A Novel Cellular Automata Based Estuarine Morphodynamic Model

Ian Bentley

MEng. MSc.

Submitted in fulfilment of the requirements for the degree of

Doctor of Philosophy

School of Engineering

College of Science and Engineering

University of Glasgow

November 2015

Volume 1 of 2

ABSTRACT

Estuaries are highly dynamic systems, subject to continuous morphological change, which results from complex interactions and feedbacks between the hydrodynamic processes, sediment transport processes and the ecology. The prediction of morphological change in estuaries is therefore difficult but is necessary to help protect a range of human interests and estuarine ecosystems. Existing methods use detailed process modelling (Bottom-Up methods) or rely on data analysis and the development simple equilibrium relationships (Top-Down methods). Bottom-Up methods are able to make accurate predictions of change over short timescales but suffer from long simulation times and an accumulation of errors when applied over medium and long timescales, while Top-Down methods are better suited for predicting long term trends in morphological behaviour. A need currently exists for new, improved methods to predict changes occurring over medium timescales (one year to several decades).

This thesis presents a new, Cellular Automata based, estuarine morphodynamic model, which divides the estuary into an array of cells and uses simplified representations of the hydrodynamic and sediment transport processes together with empirical rules to represent salt marsh ecology. The model has been developed to focus on high level interaction and feedback effects between these processes in order to identify potential medium term morphological changes that may occur in response to environmental change or engineering works.

The model has been tested using a series of sensitivity tests and idealised test scenarios for a simple generic estuary and was found to have successfully generated qualitatively realistic results. The model is robust and computationally very efficient. Further work is now needed to calibrate and verify the model using datasets from real estuaries. Future improvements may also include the addition of ocean waves, littoral wave driven sand transport and improvements to the methodology in order to further enhance the computational efficiency.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

I wish to thank the Glasgow Research Partnership in Engineering and the School of Engineering (via the James Watt Scholarship) who together provided the funding for this research. I would also like to thank my supervisor Harshinie Karunarathna for giving me the opportunity to undertake this PhD and for her continual guidance and support throughout the duration of my studies.

AUTHOR DECLARATION

I declare that no portion of the work in this thesis has been submitted in support of any application for any other degree or qualification from this or any other university or institute of learning. I also declare that the work presented in this thesis is entirely my own contribution unless otherwise stated.

Ian Bentley

Glasgow, November 2015.

LIST OF SYMBOLS

a_1, a_2 Salt marsh growth/destruction parameters (s^{-1})

A Plan area of one cell (m^2)

A_{df} Default active layer thickness (m)

A_e Active layer thickness following erosion (m)

A_d Active layer thickness following deposition (m)

A_w Peak wave near-bed orbital displacement (m)

B Parameter for calculating the wave friction factor

B_m Biomass

B_{max} Maximum biomass

B_n Elevation at base of sub-layer n (m)

c Sediment concentration

c_L Wave celerity

c_a Reference concentration

c_b Near bed sediment concentration

c_{gel} Gelling concentration

d_{gravel} Smallest grain size defined as gravel (= 2 x 10^{-3} m)

C Chezy roughness coefficient ($m^{1/2}/s$)

d_i Grain size for sediment fraction i (m)

d_s Stem diameter

d_p	Particle diameter
d_{sand}	Smallest grain size defined as sand (= 62 x 10^{-6} m)
d_{silt}	Representative grain size for silt (= 32 x 10^{-6} m)
d_{10}	10th percentile particle size (m)
d_{50}	Median particle size (m)
d_{90}	90th percentile particle size (m)
D	Deposition rate
D_s	Deposition rate due to particle settling
D_t	Deposition rate due to sediment trapping by salt marsh
D_*	Dimensionless particle size parameter
$D_{*,i}$	Dimensionless particle size parameter for fraction i
$\bar{E}$	Wave energy density (kg/s^2)
E_{mud}	Rate of erosion of mud (m/s)
f_a	Apparent bed friction coefficient
f_c	Grain friction coefficient for currents
f_c'	Instantaneous grain friction coefficient for currents
f_{cw}	Grain friction coefficient due to currents and waves
f_{cw}'	Instantaneous grain friction coefficient for currents and waves
f_w	Wave friction coefficient
$f_{w,max}$	Maximum wave friction coefficient

f'_w	Instantaneous grain friction coefficient for waves
$f'_{w,max}$	Maximum instantaneous grain friction coefficient for waves
$f_{silt,i}$	Silt factor for fraction i
F	Lateral transport factor
g	Gravitational acceleration (m/s^2)
h	Water depth (m)
h_{min}	Minimum water depth for flow calculation (m)
h_s	Water depth calculated from bed slope (m)
h_{st}	Average stem height
h_T	Total water depth calculated by the slope-based routing procedure (m)
h_{td}	Tidal water depth (m)
H	Monochromatic wave height (m)
H_{m0}	Energy based significant wave height (m)
H_{rms}	Root-mean-square wave height (m)
H_s	Significant wave height (m)
k	Wave number (m^{-1})
K	Friction parameter
k_a	Apparent bed roughness (m)
$k_{s,c}$	Current related bed roughness (m)
$k_{s,c,d}$	Current related bed roughness due to dunes (m)

$k_{s,c,mr}$ Current related bed roughness due to mega-ripples (m)

$k_{s,c,r}$ Current related bed roughness due to ripples (m)

$k_{s,grain}$ Grain roughness height (m)

$k_{s,w}$ Wave related bed roughness (m)

L Grid cell size (m)

L_w Wavelength (m)

m Dry mass of mud (kg)

m_e Erosion constant for mud (s/m)

M_a Depth of mud in the active layer before update (m)

M_b Depth of mud in the active layer following update (m)

M_e Mobility parameter

n_s Stem density per unit area

N Parameter for calculating the wave friction factor

N_s Number of positive bed slopes

N_w Number of cells to which wave energy is distributed

p_{mud} Proportion of mud

q_b Bed load per unit width (m^2/s)

$q_{b,i}$ Bed load per unit width for sediment fraction i (m^2/s)

$q_{b,x}$ Resolved bed load component along the x-axis

$q_{b,y}$ Resolved bed load component along the y-axis

q_s	Suspended sediment load per unit width (m^2/s)
R_w	Wave Reynold's number
Q	Flow (m^3/s)
Q_i	Inflow to cell (m^3/s)
Q_o	Outflow from cell (m^3/s)
s	Relative particle density
s'	Bed slope (based on bed level plus water depth in source cell)
s_L	Bed slope
s_m	Average of positive bed slopes
$\hat{s}_m$	Average of positive bed-slopes (adjusted)
s_{min}	Minimum bed slope
s_a	Salinity
$s_{a,max}$	Maximum salinity (= 1 promille)
S_a	Depth of sand in the active layer before update (m)
S_b	Depth of sand in the active layer following update (m)
t	Time (s)
$t_{s,1}$	Salt marsh model inundation time parameter 1 (%)
$t_{s,2}$	Salt marsh model inundation time parameter 2 (%)
$t_{s,max}$	Salt marsh model maximum inundation time (%)
$t_{s,min}$	Salt marsh model minimum inundation time (%)

T	Wave period (s)
T_i	Bed shear stress parameter for fraction i
T_p	Peak spectral wave period (s)
u_{10}	Wind speed at 10m above ground level (m/s)
u	Velocity (m/s)
u_c	Depth averaged current velocity (m/s)
u_e	Effective velocity (m/s)
u_{cr}	Effective critical velocity (m/s)
$u_{cr,c}$	Effective critical velocity for currents (m/s)
$u_{cr,w}$	Effective critical velocity for waves (m/s)
u_r	Velocity of return mass transport under wave (m/s)
u_*	Shear velocity (m/s)
U_w	Peak wave near-bed orbital velocity (m/s)
$U_{w,s}$	Peak wave near-bed orbital velocity computed using the significant wave height (m/s)
U_{wc}	Velocity parameter for combined waves and currents (m/s)
$U_{\delta,cw}$	Instantaneous velocity due to wave and currents at edge of wave boundary layer (m/s)
$v_{R,\delta}$	Near-bed current velocity (m/s)
v_w	Instantaneous wave near-bed orbital velocity (m/s)
w_s	Vertical exchange coefficient (m/s)

$w_{s,m}$	Particle fall velocity (m/s)
$w_{s,0}$	Particle fall velocity in clear water (m/s)
w_{50}	Median particle fall velocity (m/s)
x	Grid axis parallel to the tidal boundary
x_L	Longitudinal distance from the boundary, along the estuary centreline (m)
X	Fetch (m)
y	Grid axis perpendicular to the tidal boundary
z	Elevation (m)
z_b	Bed elevation (m)
z_{bd}	Elevation above bed (m)
z_{bdy}	Tidal water level at the model boundary (m)
z_b'	Bed elevation, adjusted to remove localised depressions (m)
z_m	Mean sea level (m)
z_{td}	Water surface elevation due to tide (m)
z_w	Water surface elevation (m)
Z	Tidal amplitude (m)
Z_n	Neap tidal amplitude (m)
Z_s	Spring tidal amplitude (m)
α	Coefficient related to the relative strength of the current and wave motion

α_{cr}	Parameter related to the relative strength of current velocity and peak wave orbital velocity.
α_{cw}	Wave-current interaction factor.
α_w	Proportion of fetch laterally reflected.
β	Coefficient related to the vertical structure of the velocity profile
β_c , β_w	Factors to account for effect of sediment particles on mixing of fluid momentum
β_{cw}	Angle between current direction and waves direction (radians)
δ	Height at which $U_{\delta,cw}$ is calculated (m)
δ_s	Thickness of near bed sediment mixing layer
δ_w	Wave boundary layer thickness
ε	Porosity
$\varepsilon_{s,c}$	Sediment mixing coefficient for currents
$\varepsilon_{s,w}$	Sediment mixing coefficient for waves
$\varepsilon_{s,cw}$	Sediment mixing coefficient for waves and currents
η	Rate of sediment trapping by plant stems
θ_c	Current direction (radians)
θ_{cw}	Instantaneous velocity direction due to combined Waves and currents (radians)
θ_s	Instantaneous sediment transport direction (radians)
θ_w	Wave direction (radians)
λ_i	Correction factor of effective grain shear stress for fraction i

μ	Mean wind speed (m/s)
μ_c	Current related efficiency factor
μ_w	Wave related efficiency factor
υ	Kinematic viscosity of water (m^2/s)
ξ_i	Hiding-exposure factor for fraction i
ρ_s	Particle density (kg/m^3)
ρ_b	Bulk density of sediment (kg/m^3)
ρ_w	Density of water (kg/m^3)
σ	Wind speed standard deviation (m/s)
τ_b	Bed shear stress (N/m^2)
$\tau_{b,cr,d_{50}}$	Critical bed shear stress based on d_{50} (N/m^2)
$\tau_{cr,marsh}$	Critical bed shear stress with adjustment for salt marsh (N/m^2)
$\tau_{cr,0}$	Critical bed shear stress (N/m^2)
$\tau_{b,c}$	Bed shear stress due to currents (N/m^2)
$\tau'_{b,c}$	Time averaged bed shear stress due to currents (N/m^2)
$\tau'_{b,cw}$	Time averaged bed shear stress due to waves and currents (N/m^2)
$\tau''_{b,cw}$	Instantaneous grain-related bed shear stress due to waves and currents (N/m^2)
$\tau_{b,w}$	Bed shear stress due to waves (N/m^2)
$\tau'_{b,w}$	Time averaged bed shear stress due to waves (N/m^2)
τ_c	Bed shear stress due to currents (N/m^2)

τ_d	Threshold bed shear stress for deposition of mud (N/m^2)
τ_e	Threshold bed shear stress for erosion (N/m^2)
$\tau_{e,max}$	Critical bed shear stress for sand/mud mixture containing 20% mud (N/m^2)
$\tau_{e,mud}$	Threshold bed shear stress for erosion of mud only (N/m^2)
$\tau_{cr,sand}$	Critical bed shear stress for sand only (N/m^2)
τ_m	Mean bed shear stress under combined waves and currents (N/m^2)
τ_{max}	Maximum bed shear stress under combined waves and currents (N/m^2)
ϕ_{floc}	Flocculation factor
$\phi_{floc,0}$	Flocculation factor at maximum salinity
ϕ_{hs}	Hindered settling factor
τ_w	Bed shear stress due to waves (N/m^2)
ψ	Current-wave mobility parameter
ω	Angular wave frequency (radians/s)
ω_{sn}	Frequency of the spring-neap tidal variation (s^{-1})

CHAPTER 1

INTRODUCTION

1.1 Project Background

Estuaries are complex natural environments that are often subject to continuous change, which may be cyclical or long term and gradual. Human interest in estuaries has historically been related to navigation, since they provide access inland from the coast and more recently, with the ever increasing size of ships, ports have been increasingly developed within the estuaries themselves. Engineering works have therefore been undertaken within estuarine waters both to construct these port facilities and to maintain their associated navigation channels; such works may include maintenance dredging and the construction of training walls. Embankment construction along the shorelines of estuaries has also been carried out, both for flood defence purposes and for land reclamation; such defences may become damaged during storms, requiring periodic maintenance. Furthermore, estuaries are frequently home to rich fishing grounds and estuarine ecosystems are also increasingly recognised as valuable in their own right.

All of these human interests in estuaries create a need for effective planning and management of the estuarine environment, to mitigate and adapt to the effects of natural changes and to minimise the impact of human activities on the environment. These interests may be affected by morphological changes which can occur naturally or as a consequence of human activity, due to man-made climate change or as an unintended consequence of engineering works (HR Wallingford, 1997).

Estuarine morphology is affected sediment movements, driven by tidal flows, fluvial flows and waves. These hydrodynamic forces are themselves affected by the morphology, via the tidal volume, bed friction and fetch lengths. Other factors, such as sediment supply, geological setting and ecology (e.g. salt marsh) also significantly influence the morphological evolution of estuaries. Due to variations in these hydrodynamic forcings

and geological settings, estuaries can vary widely in form; for example, spits, barrier beaches and tidal lagoons are common on coastlines with high wave exposure.

There is typically no clear hierarchy of cause and effect between the hydrodynamic processes, sediment movements, morphology and estuarine ecosystems, due to the many interactions and feedbacks that can occur between them and hence the evolution of estuarine systems can be highly complex in nature. In practice, this can lead to unintended consequences following engineering works and complicates the task of planning for the effects of climate change by making the potential consequences of sea level rise and changes to weather patterns difficult to predict. Effective management of estuaries therefore depends on the availability of effective tools for the prediction of potential medium and long term morphological changes, including those occurring due to climate change or as unintended consequences of human activities.

1.2 Morphological Modelling of Estuaries

The current methods available for predicting the morphological evolution of estuaries include process based "bottom-up" modelling and "top-down" methods. Bottom up models, which solve complex dynamic process equations, are well suited to making detailed, quantified predictions for localised areas and short timescales. Top down techniques on the other hand use data analysis or assume some form of equilibrium condition and are used by geomorphologists to make qualitative predictions covering large space and timescales (EMPHASYS Consortium, 2000). More recently, a number of hybrid models have been developed, which combine elements of bottom-up and top-down models in an effort to bridge the gap in capability between these model types (Huthnance et. al., 2007). Top-Down and hybrid modelling approaches tend to be more flexible in the use of empirical information than Bottom-Up models but due to their relative simplicity may not be able to fully capture the complex dynamics of estuarine systems.

Another promising approach is the application of rule based models. In the ESTSIM project (ABPmer et. al., 2008) a Boolean model was developed to represent and map the interactions of geomorphic elements and processes within different estuaries and generic estuary types. Another type of rule based model is based on Cellular Automata, where the

model domain is divided into an array of cells and deterministic rules specify changes of state for each cell, based on conditions within a local neighbourhood of nearby cells. An example of this model type is SLAMM (Warren Pinnacle Consulting, 2012), which attempts predicts the effect that sea level rise will have on coastal wetlands. Cellular automata based models are able to make simplified representations of the key physical processes in order to capture the key interactions and feedbacks that govern the overall system behaviour. Although CA grids can take a variety of forms, the majority of CA models used in geomorphology (e.g. Murray and Paola, 1994) have used a two dimensional regular grid of cells, in which each cell represents a square region of the model domain.

1.3 Project Objectives

The aim of this project has been the development of a new cellular automata based estuarine morphology model, using simplified representations of estuarine processes together with empirical rules, and able to capture the complex interactions and feedback effects that can occur between hydrodynamic forcings, sediment transport, morphology and estuarine ecology. A cellular automata is considered a suitable format for this model, due to its flexibility and ability to incorporate both process based and empirical rules. It is intended that the model should be capable of making qualitative predictions of morphological change for entire estuarine systems, over medium timescales (one year to several decades) and hence the model should be computationally efficient.

1.4 Thesis Outline

In Chapter 2 the physical processes that drive morphological change in estuaries are described in detail. These processes are related to classification systems that have been developed for estuaries including those based on morphological form. In Chapter 3 the currently available morphological modelling techniques for estuaries are described and discussed. The case for a new cellular automata based model is also made in this chapter. In Chapter 4, the model developed in this project is described in detail. In Chapters 5 and 6 results are presented from a series of sensitivity analyses and test scenarios, which were

carried out using an artificially generated generic estuary. Chapter 7 presents the conclusions drawn from this research; suggestions for future development and improvement of the model are also made in this chapter.

CHAPTER 2

ESTUARINE PROCESSES AND MORPHOLOGICAL CLASSIFICATION

2.1 Introduction

Most present day estuaries were created by the flooding of fluvial and glacial valley systems by sea level rise, following the end of the last ice age around 15,000 years ago. Present morphology is derived from these original valleys and has been shaped by processes such as tides, waves and fluvial flows (Perillo, 1995a). Estuaries are highly variable in form, with complex interactions and feedbacks occurring between morphological characteristics, hydrodynamic processes, sediment movements and ecology. No clear hierarchy of cause and effect exists in these interactions, with the morphology both influencing, and influenced by, the hydrodynamic processes (HR Wallingford, 1997).

The primary hydrodynamic processes affecting estuarine morphology are river flows, tides and waves, which erode, transport and deposit sediments. Tidally generated currents within estuaries are driven by the tidal range at the open coast, the tidal volume of the estuary and interaction with the bed. They are also affected by density driven circulation and by the rotation of the earth, via the Coriolis Effect. River flows have the greatest effect in the upper reaches of estuaries with tidal flows becoming dominant in the middle and outer regions. The outer estuary may also be significantly affected by wave-driven flows and sediment transport, leading to the development of features such as spits and barrier beaches (Dyer, 1997).

Availability of sediment significantly influences the morphology. High fluvial sediment loads tend to result in rapid sedimentation and the formation of deltas. The availability of marine sediment may influence the formation of spits, bars and barrier beaches as well as sand and mud flats in the middle and outer estuary.

Estuarine morphology is also affected by human activity, directly through activities such as land reclamation, dredging and civil engineering works and indirectly via effects such as climate change and the effect of pollution on ecosystems. Unintended negative consequences have often occurred following such changes and these can be very difficult to predict due to the complex nature of estuarine systems (HR Wallingford, 1997).

Estuaries have been classified according to a range of criteria including tidal range, tidal propagation, tidal prism, salinity structure and morphology. Although these criteria are frequently linked, each individual estuary is unique and no single classification system has been developed to include all of them.

2.2 Estuarine Processes

In this section the processes that drive the morphological development of estuaries are described. These include the hydrodynamic forcings of river flow, tides and waves, as well as the erosion, transport and deposition of sediment, the effects of estuarine ecology and various human induced changes that can occur in estuaries.

2.2.1 Tides

Tidal variation in the open ocean follows diurnal, semi-diurnal or mixed cycles, with a variation in tidal range occurring roughly every two weeks, due to the relative gravitational effects of the sun and moon: the maximum range in this cycle is known as the 'spring tide' and the minimum as the 'neap tide'. Other, smaller variations occur due to cycles in the orbit of the moon relative to the earth, including the lunar nodal cycle, which has a period of approximately 18.6 years. The declination of the moon and complex interactions with the sea bed and coast also affect the tide, resulting in a large variation in tidal range and in some locations this also results in only one tidal cycle occurring each day (diurnal tides) or a marked difference between alternate semi-diurnal tides (mixed tides) (Pethick, 1984).

Estuaries may be classified by the tidal range at the open coast as microtidal (tidal range < 2m), mesotidal (tidal range 2 to 4m) and macrotidal (tidal range > 4m) (Perillo, 1995b). At the lower end of this scale, estuarine morphology is dominated by wind and wave action,

hence lagoons, spits and bars are common features. In macrotidal estuaries, tidal currents are typically the dominant process and these estuaries are typically funnel shaped with extensive tidal flats and salt marshes, although wave action may have a significant effect near the mouth. Following similar lines estuaries can also be classified as wave dominated or tide dominated (Perillo, 1995b).

Tides in estuaries induce currents, the magnitude of which depends primarily on the tidal amplitude at the estuary mouth and the tidal prism within the estuary, which is defined as the volume of water contained within an estuary between the high and low water levels (Dyer, 1997). Tidal range is highly variable, ranging from just a few centimetres (e.g. in the Mediterranean) to over 15m (Bay of Fundy, Canada).

Tides propagate into estuaries as shallow water waves with celerity $\sqrt{(gh)}$, where g is the gravitational acceleration and h is the water depth. In some cases this wave is reflected back from the head of the estuary to create a standing wave, while in other cases, where the incoming wave is not reflected due to the geometry or frictional effects, the tide propagates solely as a progressive wave. Often the wave is reflected with significant energy dissipation, resulting in a mixture between standing and progressive waves.

For standing waves, the tidal current occurs in phase with the water level, with slack water occurring at the same time as high and low water, while currents associated with progressive tidal waves the current are out of phase by 90°, with the maximum flood and ebb currents occurring at high and low water. Where the wave is of the mixed type, the phase difference will be somewhere in between these two values (Dyer, 1997).

Tidal propagation, within estuaries, is affected by convergence effects, which tend to increase the tidal range with increasing distance from the mouth, and frictional effects, which tend to reduce the tidal range. Estuaries have been classified according to which of these effects is dominant (Dyer, 1997), as hypersynchronous, synchronous and hyposynchronous. In hypersynchronous estuaries the convergence effects dominate, producing increasing tidal range and currents with increasing distance from the mouth, while in synchronous estuaries neither effect dominates, producing a constant tidal range

over the length of the estuary, and in hyposynchronous estuaries the frictional effects dominate. Estuaries in this latter category commonly have restricted mouths.

The natural tendency for the crest of the tidal wave to travel faster than the trough, combined with the effect of friction, which has a greater influence at low depths, can result in higher peak velocities during the flood tide than during the ebb (flood dominance). The opposite situation (ebb dominance) can also occur, depending on the extent of the inter-tidal areas, and this is related to the ratio of the cross sectional area at the mouth to the water surface area within the estuary: as the tide approaches low water this value reaches a maximum, which allows a faster adjustment of the water surface within the estuary during the ebb tide and higher ebb velocities. Tidal currents within estuaries are also affected by the rotation of the earth, via the Coriolis Force, which can result in different flow paths for flood and ebb currents (Dyer, 1997). Atmospheric effects, such as wind setup, wave setup and changes in atmospheric pressure also have an effect, which is superimposed on the astronomical tide (Davison-Arnott, 2009).

Residual currents can occur in estuaries when tidal flows take a different paths during the flood and ebb tidal phases, resulting in a net residual circulation of water when averaged over the tidal cycle. Such residual flows may be related the bathymetry and non-linear friction, to horizontal geometry effects or to the Coriolis Force (Wang et. al., 1999). The first of these causes is related to flood/ebb dominance and the fact that more flow is distributed over the tidal flat during periods of greater depth, while the second cause relates to momentum effects; for example at channel bends and inlets.

2.2.2 River Flows and Salinity Effects

River flows can have a significant influence towards the head of estuaries, where the relative effects of tides and waves are diminished. Sediment is transported downstream and may contribute to infilling of the estuary or be washed out to sea by the tide. Where the fluvial sediment supply is high, a delta may be formed (Dyer, 1997; Nielson, 2009).

Fluvial sediment concentration can vary widely between rivers; for example, using sediment yields for Scottish rivers given by MacManus and Duck (1996) and mean flow

data from the Centre for Ecology and Hydrology (2013) average sediment concentrations for these rivers were found to be in the range 0.011 to 0.167 kg/m^3. Wass and Leeks (1999) calculated average suspended sediment concentrations for rivers in the Humber catchment, in the UK, which were found to be in the range 0.022 to 0.058 kg/m^3. Far higher concentrations are possible however; for example, the sediment concentration in the Yellow River, China, can be as high as 300 kg/m^3 (Xu, 2002).

High fluvial discharge can also result in stratification within an estuary, where fresh water flows out over the top of the denser saline water. Some of the saline water is drawn into the outward fresh water flow and must be replaced by seawater, resulting in a net inward flow near the bed and this can significantly affect the morphology. Estuaries have been classified according to the degree of stratification as highly stratified fjord and salt wedge estuaries, partially stratified and well mixed estuaries (Fischer et. al., 1979).

Salinity structure depends on the depth, tidal range and fluvial discharge for a given estuary, with greater stratification occurring for estuaries with smaller tidal ranges, larger depths and higher river flows. Vertical mixing of the salinity structure is caused by turbulence generated by shear at the sea bed and by turbulence generated by shear at the interface between the fresh water and saline layers (internal mixing). In highly stratified estuaries bed generated turbulence is contained entirely within the saline layer and mixing is caused only by turbulence at the interface (Dyer, 1997). Any temporary increase in turbulence can affect the salinity structure and extreme tides, waves and fluvial discharge can generate increased mixing and de-stratification, as has been observed for waves and fluvial discharge in Mobile Bay, Alabama (Schroeder et. al., 1990).

2.2.3 Waves

Waves affect estuarine morphology by eroding and transporting sediments. Wave action generates shear stresses on the bed, which can induce a net movement of sediment due to asymmetry of the velocities beneath the wave and significantly enhances transport generated by tidal and wave induced currents (Soulsby, 1997).

The ability of sea waves to move sediments depends on the wave height and period as well as the water depth. Random sea waves are typically made up of many individual waves with different heights and periods and statistical measures, such as the significant wave height (H_s) and mean zero up-crossing period (T_m), are therefore typically used to describe average conditions (Goda, 2000). Random sea waves may also be described by the wave energy spectra, which gives the distribution of wave energy as a function of the radian frequency. Common wave energy spectra are the Pierson-Moskowitz spectrum, which applies to fully developed waves in deep water and the JONSWAP spectrum, which applies to growing waves in continental shelf waters (Soulsby, 1997). The energy based significant wave height (H_{m0}) and peak spectral period (T_z) are often used in place of H_s and T_m, and give very similar values to the statistical methods described above.

Waves in estuaries may be caused by the propagation of waves from the open sea into the mouth of the estuary, which are modified by shoaling, diffraction, refraction and breaking, or may be generated locally by the wind. The height and period of locally generated waves are mainly dependant on the wind speed and duration, and the fetch (the distance of open water over which the wind blows) (US. Army Corps of Engineers, 2002).

In estuaries where waves are the dominant influence on morphology (known as wave-dominated estuaries, bar-built estuaries or coastal lagoons) sediments will typically build up near the mouth, to form a barrier beach or spit. Sediment accumulates in the inlet until the tidal current velocities are sufficiently high to erode any further sediment added by the littoral drift. The barrier is typically backed by a lagoon, which has low tidal and wave energy and may contain extensive salt marshes (Bird, 2008).

2.2.4 Sediment Transport

Movement of water over the bed creates shear stress due to friction which, if large enough to move the individual sediment grains, will induce sediment transport due to the movement of sediment particles along the bed (bed load) and the transport of sediment particles within the water column (suspended load). The sediment transport rate is dependent on flow velocity, water depth, wave conditions and sediment properties.

Biological organisms may also act to stabilise or destabilise sediments, affecting transport rates.

The movement of course sediment, in high energy environments such as exposed beaches, primarily due to the action of waves is commonly referred to as littoral sediment transport. Movement of sands and gravels can occur along coastlines due to the angle of incoming waves and this is known as long-shore transport and, where significant, this may be responsible for the development of spits and barrier beaches at the mouths of wave dominated estuaries. Cross shore sediment transport also occurs on beaches, with variations in wave conditions causing erosion and subsequent rebuilding of beaches (Nielson, 2009).

Wind may also move sediments at the shoreline to create sand dunes, which provide a natural store of sediment and may provide natural protection against flooding.

2.2.4.1 Sediment Properties

Estuarine sediments are mostly derived from eroded soil and rock, which is transported into estuaries by tidal and fluvial flows, and from organic matter. They are commonly described by their grain size distribution, which can be determined by sieve testing for particle sizes between 63μm and 75mm, while sedimentation testing is required for silt and clay particle sizes (British Standards Institution, 1990). Silts (particle sizes between 4 and 63μm) and clays (particle sizes < 4μm) behave differently to sands due to the effects of cohesion and any sediment mixture containing more than around 10% of such materials may be affected by this property (Whitehouse et. al., 2000).

Particle density is also important and for sand (composed of quartz grains), this is usually taken to be 2650 kg/m^3. The density of clay minerals is in the range 2500 to 3000 kg/m^3, although the bulk properties of clays are of greater importance. Porosity is the proportion of voids by volume in the sediment and for sands this is usually in the range 0.3 to 0.5, depending on the degree of compaction and particle size distribution (Soulsby, 1997). The dry density, which is related to the porosity and particle density, is more often used when

referring to cohesive material and this can vary considerably, from around 50 kg/m^3 to around 1500 kg/m^3 (Whitehouse et. al., 2000).

Sediments containing significant proportions of fine material (fine silt and clay) may be affected by the property of cohesion. Fine sediments are primarily composed of metal silicate type minerals, which contain electrical charges and exert electrostatic forces between grains, causing them to naturally adhere. Cohesion between particles is enhanced in saline environments due to the presence of free cations and anions, which form an electrical double layer around the particles (Whitehouse et. al., 2000).

2.2.4.2 Threshold of Motion

In slowly moving water there may be no sediment movement at all because the shear stress generated at the bed is too low to move any sediment grains. The threshold of motion occurs at the minimum velocity required to initiate the movement of sediment particles and is a property of the bed material, usually expressed as a critical bed shear stress, a critical velocity or a critical Shields parameter. Many sediment transport formulae are expressed in terms of the difference between the bed shear stress and the critical bed shear stress.

Shields (cited by Soulsby, 1997) carried out a large number of experiments in order to produce his well-known Shield's diagram, which can be used to estimate the critical shear stress for a given sediment type. The original diagram is somewhat inconvenient to use, as it contains shear stress terms on both axes, necessitating an iterative approach to obtain a solution and as a graphical method is not suited implementation within a computer program. Soulsby (1997) gives an equation allowing direct calculation of the critical shear stress given by the Shields diagram.

2.2.4.3 Bed Forms

Bed forms are regular patterns of surface features that commonly occur on mobile sandy beds and may significantly increase the hydraulic roughness of such beds. The type and dimensions of these forms depend on the sediment characteristics and flow regime. Common terms used to describe bed forms include plane bed, ripples, mega-ripples, dunes,

anti-dunes and bars; however, definitions are not well established and can vary. The term 'sand waves' is used to describe dune type bed forms in the sea (Soulsby, 1997) or as a generic term for ripples and dunes (Fredsoe and Deigaard, 1992). Smaller bed forms can also be superimposed on larger ones, to create compound forms.

2.2.4.4 Sand Transport

The transport of non-cohesive sediment is commonly divided into bed load and suspended load, where bed load is the movement of sediment particles along the bed by rolling, sliding and jumping (saltation) along the bed while the suspended load is held in suspension by the upward components of turbulent currents and is carried along by the movement of water. The bed-load transport rate is typically around 5-25% of the suspended-load transport rate; however higher proportions may be transported as bed-load where coarse materials predominate (Yang, 1996).

Sand transport occurs due to the shear stress induced by the movements of water over the bed material. When the critical shear stress is exceeded, the transport of sand grains increases as a function of the shear stress and the critical shear stress; however, this function is complex and although many formulae have been proposed none are currently universally accepted.

Bed shear stress is generated by both waves and currents and hence both of these hydrodynamic processes can generate sediment transport. Waves induce transport because higher peak orbital velocities beneath the wave crest cause a net movement of sediment in the direction of the wave. Waves also entrain sediment into the water column, where it is transported by currents, and may themselves generate currents (e.g. longshore currents) which contribute to the transport of sediments (Soulsby, 1997).

Other factors that may affect sand transport include the grain size distribution in the bed and the presence of bed forms, such as ripples or dunes. Where the bed material contains a range of sand grain sizes the smaller grains will be transported more readily than the larger ones, which can lead to sorting (e.g. fining in the downstream direction, for unidirectional flow). This effect is, to some extent, counteracted by the tendency of smaller grains to be sheltered by larger grains and the greater exposure of the larger grains (hiding and

exposure effects, Van Rijn, 2007c). Bed forms enhance sand transport rates by increasing the hydraulic roughness and hence the shear stress on the bed, through the migration of bed forms and by enhancing the entrainment of sediment. This complicates the task of calculating sediment transport rates, since the bed forms are themselves related to sediment transport rates (Van Rijn, 2007a).

2.2.4.5 Cohesive sediment transport

If silt and clay sized particles (smaller than 65μm) make up more than around 10% of the sediment, it may have cohesive properties. Fine cohesive sediments, known as muds, are common in estuaries and their behaviour is fundamentally different from that of sands. Muds may be categorised according to settling behaviour (Soulsby, cited by Whitehouse et. al., 2000) as unflocculated suspension, flocculated suspension, fluid mud and consolidating settled bed material.

Unflocculated suspension is made up of fine mud particles suspended separately in the water column; settling velocities are in the order of 0.01 mm/s. Flocculated suspension occurs in the presence of the positive and negative ions in saline water, which create an attractive force between mud particles (Pethick, 1984; Whitehouse et. al., 2000), so that they tend to adhere following collisions, to form flocs. The settling velocity of flocs ranges from 0.01 to 5 mm/s. Fluid mud occurs at very high concentrations when the space between flocs is similar to the size of the flocs themselves. Interactions between fluid mud particles are significant and the viscosity of the mixture is much higher than that of pure water but low enough to allow the mud to flow. A consolidating settled bed develops as the concentration increases further and the viscosity eventually becomes high enough to prevent the mud from flowing. At this stage the flocs are in contact with one another and settling gradually under the overlying weight.

Mud is transformed between states through the processes of erosion, transport, deposition and consolidation. Erosion occurs when the bed shear stress exceeds the threshold for erosion (τ_e); this is related to the dry density of the mud, although considerable scatter exists in this relationship. Typical values for τ_e are in the range 0.1 to 0.2 N/m^2, although it is generally higher in sand-mud mixtures, with a peak value occurring in mixtures

containing around 20% mud (Whitehouse et. al., 2000). Once in suspension, mud is transported by the processes of advection and diffusion and the transport rate due to currents is related to the velocity and concentration profiles. Mud is deposited when the bed shear stress falls below a critical value for deposition, which is typically around half that for erosion. Fluid mud is a highly viscous layer of unconsolidated mud near the bed, which can be formed during deposition or disturbance by wave action; once formed, a fluid mud layer can flow along the bed, down slopes. The settled bed will gradually consolidate as the weight of overlying material expels the trapped pore-water, causing an increase in dry density with time and depth below the surface of the bed.

2.2.5 Estuarine Ecology

Estuarine ecosystems are distinguished from freshwater and marine ecosystems by varying salinity, extensive intertidal areas and generally low average water depths. In general the number of species present decreases from the fresh water environment at the upstream limit of an estuary as the salinity increases, before increasing again as the salinity approaches that in the open sea. Estuarine environments are characterised by smaller numbers of species but abundant numbers of individual organisms (McLusky, 1981).

Plant species in estuaries may include microscopic phytoplankton and microalgae (microphytobenthos), macroalgae (seaweed) and sea grasses and as well as salt marsh and mangrove species in the intertidal areas. Animal species may include zooplankton, fish, birds and a wide range benthic organisms, which live on and within the bed and feed on live organisms and decaying plant material (detritus).

The activities of some organisms can directly affect estuarine morphology. Marsh species, sea grasses, sea weeds and microalgae tend to stabilise sediments through the development of root systems or by the productions of biofilms; these species are often referred to as biostabilisers. Other species, known as biodestabilisers may act to destabilise sediment by digging and burrowing into sediment, although this generally occurs to a lesser extent than biostabilisation (Whitehouse et. al., 2000). Since all species that form part of an ecosystem are generally interrelated through competition and predator-prey relationships it follows

that changes to the population of any particular species may have an impact on biostabilisers or biodestabilisers and therefore on estuarine morphology.

A number of estuarine species are subject to human exploitation, with estuaries often providing rich fishing grounds. Intertidal areas and saltmarshes also provide important habitats for a range of bird species, some of which may also have an indirect influence on the morphology through their feeding activity (Widdows and Brinsley, 2002).

2.2.5.1 Salt Marshes

Salt marsh vegetation is made up of a variety of salt tolerant plants (e.g. cord grass, rushes, plantains, sea-lavenders and glassworts), which grow in sheltered inter-tidal areas such as estuaries. Marsh stabilises sediment, reducing erosion, and encourages the deposition of fine sediments. Marshes can provide an effective defence against coastal flooding by dissipating wave energy and provide a unique habitat for a number of species that are not found in other environments (Townend et. al., 2010). Environmental changes, such as a rise in sea levels or increased storminess, can result in a reduction in marsh extent through mechanisms such as erosion or drowning of the marsh, which may lead to significant morphological change and increased flood risk.

Marsh species such as Spartina Alterniflora (a type of cord grass) are able to survive periodic submergence by the tide and their distribution is related to soil elevation and inundation frequency (Townend et. al., 2010). Colonisation by species such as Spartina Alterniflora typically occurs above the mean high water neaps level and at higher elevations a range of other species may be present (Packham & Willis, 1997). These marsh species are therefore sensitive to changes in relative sea level and may drown if sedimentation rates are insufficient to keep pace with sea level rise. If the marsh aggregation rate is able to keep pace with sea level rise but the near shore sedimentation rate outside of the marsh is not, then the marsh will often retreat by erosion at its seaward extent, with the formation of an abrupt scarp (Schimmer & Pizzuto, 2000). Established marsh enhances sedimentation rates by increasing the hydraulic roughness, slowing the flow of water, and by organic biomass production (Packham & Willis, 1997).

Marsh erosion due to sea level rise may be offset by landward transgression of the marsh; however, this is often prevented by the presence of sea defences, which may protect previous areas of marsh that have been converted to agricultural land, in land reclamation projects. In recent times it has been recognised that the value of the marsh in terms of ecology and flood defence may exceed the value of the defended land and some areas are now being converted back to salt marsh via a process of 'managed realignment' (Boorman, 1999).

2.2.5.2 Sea Grasses

Sea grasses are aquatic flowering plants that are able to tolerate varying degrees of salinity. They can be found in intertidal areas and depths up to around 60m but are vulnerable to desiccation and physical stresses in the upper part of this range, while the maximum depth is dependent on light penetration. Sea grasses may also be limited by their ability to take up dissolved inorganic carbon from the water and might actually benefit from rising levels of dissolved CO_2. Like salt marshes, sea grasses increase the frictional drag on tidal flows, attenuate waves, protect the substrate from erosion and enhance sedimentation (Borum et. al., 2013).

2.2.5.3 Benthic Algae

Benthic algae can be divided into macroalgae (seaweed) and microalgae (microphytobenthos). Both forms can be found in shallow estuarine intertidal and sub-tidal areas but are able to survive at greater depths than either salt marsh or sea grasses (McGlathery et. al., 2013). Seaweeds may colonise intertidal areas below the seaward limit of salt marshes and can significantly increase bed friction, thereby reducing near bed current velocities and erosion of sediment (Widdows and Brinsley, 2002). Microphytobenthos, which are common on intertidal mud flats, generate biofilms which bind sediments and can significantly reduce erosion; however, this depends on the relative abundance of algae and bio-destabilising grazers, such as molluscs and worms (Widdows and Brinsley, 2002).

2.2.5.4 Molluscs and Worms

Species such as clams, snails and ringed worms can reduce the critical shear stress for erosion through activities such as burrowing, feeding and grazing. The clam Macoma Balthica, for example, has been shown to reduce the critical shear velocity from around 0.35 m/s to less than 0.15 m/s and increase erosion rates by a factor of 10 to 100 in the Humber and Westerschelde estuaries. The presence of the snail Hydrobia Ulvae has also been shown to increase sediment erodibility in the Danish Wadden Sea (Widdows and Brinsley, 2002).

Conversely, some bivalve species, such as mussels and oysters, can form extensive reefs in the lower intertidal area. These reefs protect sediment from erosion, enhance deposition rates and increase the hydraulic roughness of the bed. Results from flume studies have shown that mussel beds in the Humber estuary can reduce sediment erosion by up to a factor of 10, although mussel beds with only partial coverage may actually increase erosion rates, by enhancing the velocity in-between the beds (Widdows and Brinsley, 2002).

2.2.6 Human Impacts

Human activity has an important influence on the morphology of estuaries through activities such as port development, dredging, land reclamation and flood defence works. Water quality is affected by the introduction of chemical pollutants, with impacts to ecology, and the rate of sea level rise is increasing due to the effects of global climate change, which may cause increased wave exposure and erosion of salt marshes (Mariotti and Fagherazzi, 2010).

Any change that significantly impacts the hydrodynamics within an estuary will cause a corresponding change to sediment transport patterns, which then feeds back into the hydrodynamic regime. Following such a disturbance an estuary will usually converge into a new state of dynamic equilibrium and the current regime of many estuaries includes human activities such as land reclamation and dredging (HR Wallingford, 1997).

Pollutants in estuaries can include ocean-borne oil and plastics, gaseous pollution from nearby industry and river-borne fertilizers and heavy metals, which can kill vegetation and

animal life or encourage the growth of algae with subsequent harmful consequences to ecosystems (Packham and Willis, 1997) and eventually to the morphodynamics.

Global temperatures are expected to rise as a result of increased levels of carbon dioxide and other greenhouse gasses in the atmosphere, causing an increase in sea levels due to melting of polar ice, glaciers and permafrost, as well as the thermal expansion of sea water. Sea level rise estimates include a wide margin of error, due to uncertainty regarding future emissions and the inherent uncertainties in the models used; the Fifth Assessment Report of the Intergovernmental Panel on Climate Change (IPCC, 2014) gives a best estimate of potential sea level rise between 0.44 and 0.74m for the end of the century, relative to 1990 levels, depending on the future scenario (demographic, social, economic, technological and environmental). Recent findings from model studies have also found that intensity of storms is likely to increase and changes in rainfall patterns and hence river discharge can be expected (Parry et. al., 2007).

Sea level rise increases the depth within estuaries (although this is often counteracted by infilling as discussed in Section 2.4), which may increase tidal flows and wave penetration, while changes to storm frequency and intensity as well as rainfall distribution will affect the wave climate, fluvial flows and storm surge events, with consequent impacts on the morphological regime. Where sea defences prevent the landward migration of inter-tidal areas, a reduction in inter-tidal sand flats, mud flats and salt marsh is likely to occur (Doody, 2004), resulting in a loss of habitats and increased flood risk.

Increased levels of atmospheric carbon dioxide can also affect plant species such as salt marsh vegetation directly, by enhancing the growth rates of some species, giving them a competitive advantage over other, less affected species (Packham and Willis, 1997). Increases in dissolved carbon dioxide in seawater similarly benefits aquatic plant species but also causes an increase in acidity, which is expected to be harmful to some species, such as molluscs (Boyd, 2011).

2.3 Morphological Classification

Morphological estuary classification systems have been proposed by Fairbridge (cited by Perillo, 1995b), Perillo (1995b) and more recently by ABPmer et. al. (2008). The EstSim classification, shown in Table 2.1, was developed in Phase 2 of the Department of Environment, Food and Rural Affairs (DEFRA) and Environmental Agency, UK joint Estuaries Research Programme (ERP). Here, estuaries are classified in terms of origin and behavioural type (ABPmer et. al., 2008).

Type	Origin	Behavioural Type
1	Glacial valley	Fjord
2		Fjard
3	Drowned river valley	Ria
4		Spit-enclosed
5		Funnel shaped
6	Marine/fluvial	Embayment
7	Drowned coastal plain	Tidal inlet

Table 2.1: ERP2 Classification System for Estuaries (ABPmer et. al., 2008)

ESTSIM defined the geomorphological elements present in each generic estuary type classified according to morphology. Following that, each of the estuary types given in Table 2.3 was represented by a systems diagram, which set out the geomorphic features typically present together with their interactions in terms of sediment exchange. Figure 2.1 shows the geomorphic elements and the linkages that may be present in different estuary types, in a systems diagram for a generic estuary. Arrows in Figure 2.1 represent flows of matter or energy between elements.

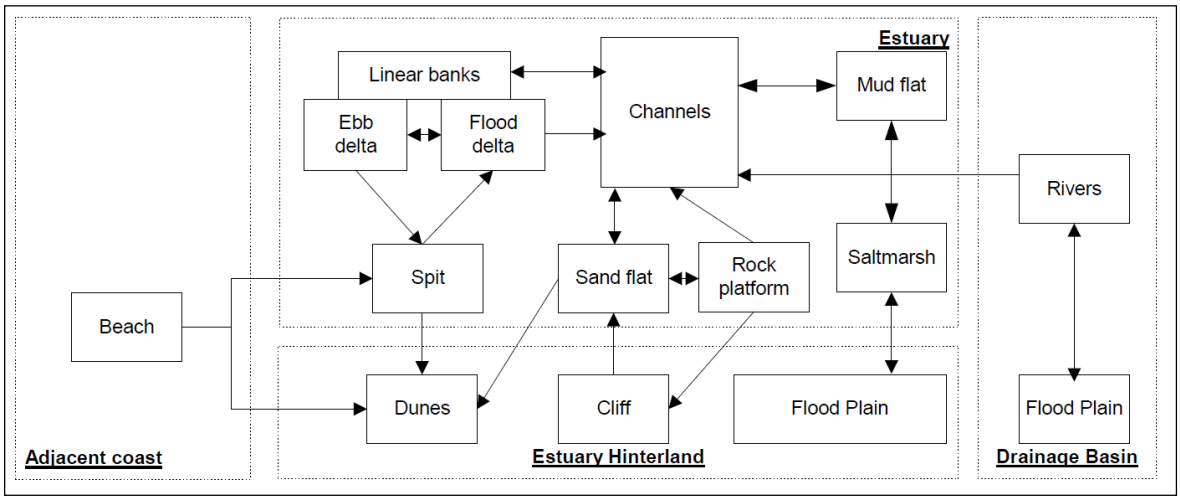

Figure 2.1: A systems diagram for a generic estuary (reproduced from ABPmer et. al., 2008. Crown copyright: (Defra); 2009)

Figure 2.1 shows how the presence of particular geomorphic features can influence the development of others. For example, beaches on adjacent coastlines provide a source of sand, which may lead to the development of spits and dunes near the mouth and, inside the estuary, the erosion of cliffs may provide a local source of sand, leading to the development of sand flats and subsequently dunes.

2.3.1.1 Drowned Glacial Valleys

Fjords occur in high relief, high latitude areas in drowned former glacial valleys, such as occur in Norway, Scotland, Greenland, Canada, Chile and New Zealand. They are long narrow, deep and steep sided with U-shaped cross sections and often contain one or more submarine sills, formed by glacial over-deepening of the fjord basin. They are typically highly stratified and deep fjords may contain isolated anoxic regions below the level of the sill (where present). Sediment inputs from marine sources are often relatively small due to the barrier effect of the sill; however they will usually retain sediments which may derive from fluvial, marine or wind-blown sources or from land-slips on adjacent slopes (Syvitski and Shaw, 1995). The EstSim systems diagram for a generic fjord is shown in Figure 2.2.

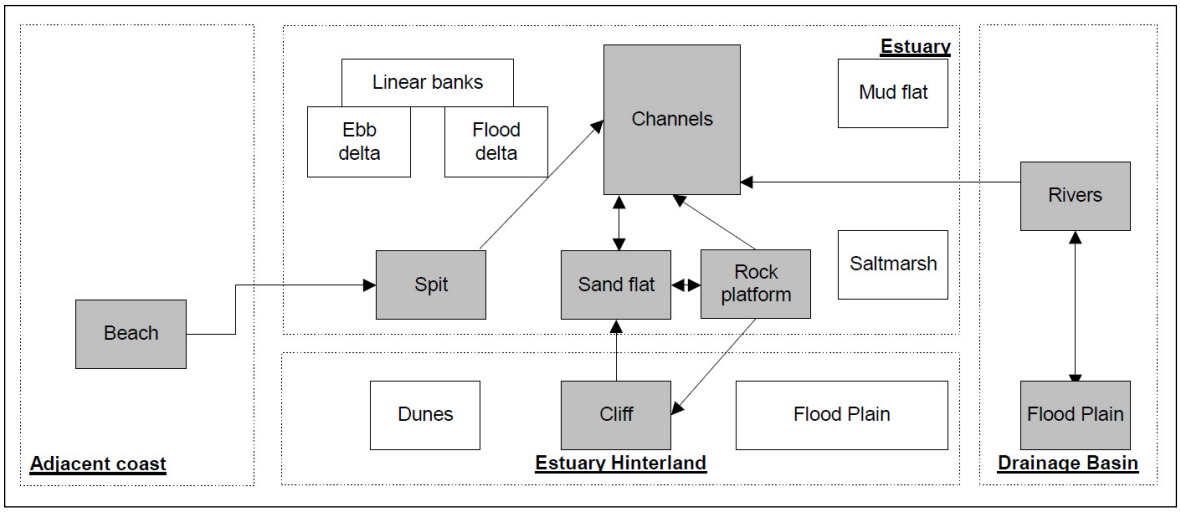

Figure 2.2: ***A systems diagram for a generic fjord (reproduced from ABPmer et. al., 2008. Crown copyright (Defra); 2009)***

Fjards occur in low relief former glacial valleys. These estuaries are more likely than fjords to fill with sediment and features such as sand flats, mud flats and salt marsh are common. They are typically located in low relief settings and have significant areas of sand and mud flats (ABPmer et. al., 2008). The EstSim systems diagram for a generic fjard is shown in Figure 2.3.

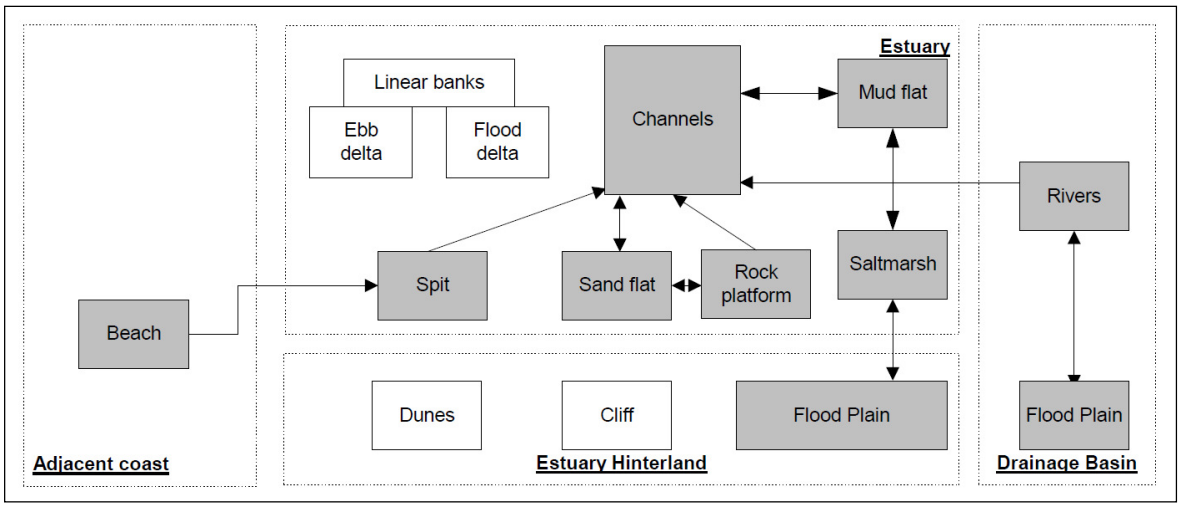

Figure 2.3: ***A systems diagram for a generic fjard (reproduced from ABPmer et. al., 2008: Crown copyright (Defra); 2009)***

2.3.1.2 Drowned River Valleys

Rias are drowned river valleys that occur on high relief coasts. They are common in South West England, South West Ireland, Brittany, Northern Spain and parts of the Chinese, Korean and Argentine coasts. These estuaries typically have substantial outer areas that are dominated by marine processes and subject to infilling by marine sands. Further inland, typical estuarine processes predominate and sediments from both marine and fluvial sources may be present, while peripheral areas are often occupied by inter-tidal mud flats. In the uppermost regions, fluvial currents and sediment inputs predominate (Castaing and Guilcher, 1995).

Other drowned river valleys exist on low relief coasts, in valleys that were cut by rivers during previous geological epochs, when sea levels were lower than at present. Where wave action is significant one or more spits may form, restricting the mouth to form a spit enclosed estuary, which would typically have flood and ebb deltas at the mouth and reduced tidal and wave energy in the middle and inner estuary. Where tides are the dominant forcing factor, a funnel shaped estuary is more likely; these estuaries typically have linear banks, elongated sand banks near the mouth and are flanked by extensive mud flats and salt marsh.

The EstSim systems diagram for a generic funnel shaped drowned river valley is given in Figure 2.4.

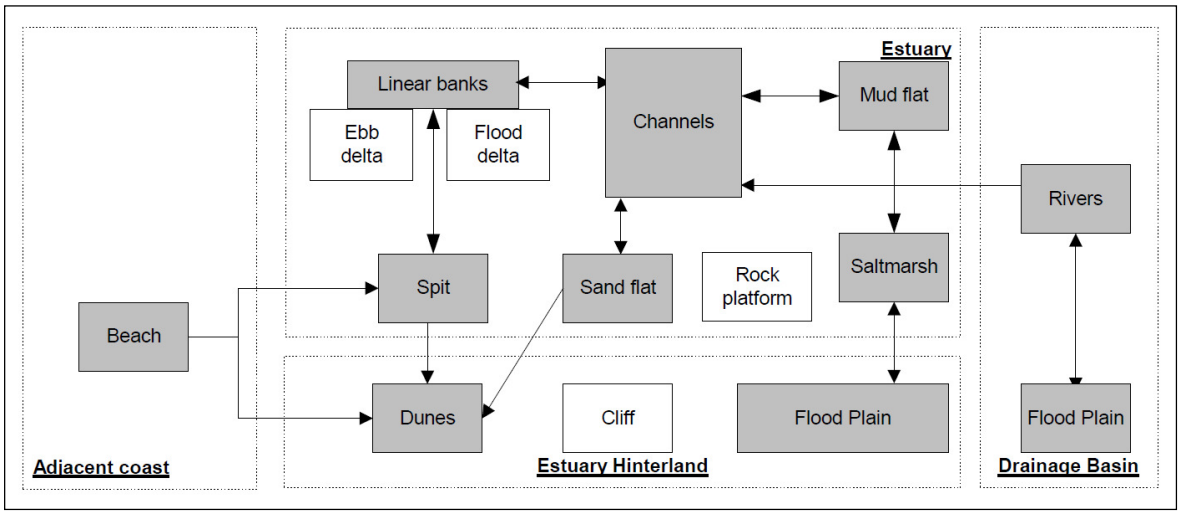

Figure 2.4: ***A systems diagram for a generic drowned river valley (reproduced from ABPmer et. al., 2008: Crown copyright (Defra); 2009)***

2.3.1.3 Embayments

Embayments are areas of marine or fluvial origin that are joined by multiple tidal rivers and have width to length ratio greater than 1 (ABPmer et. al., 2008). The EstSim systems diagram for a generic embayment is given in Figure 2.5.

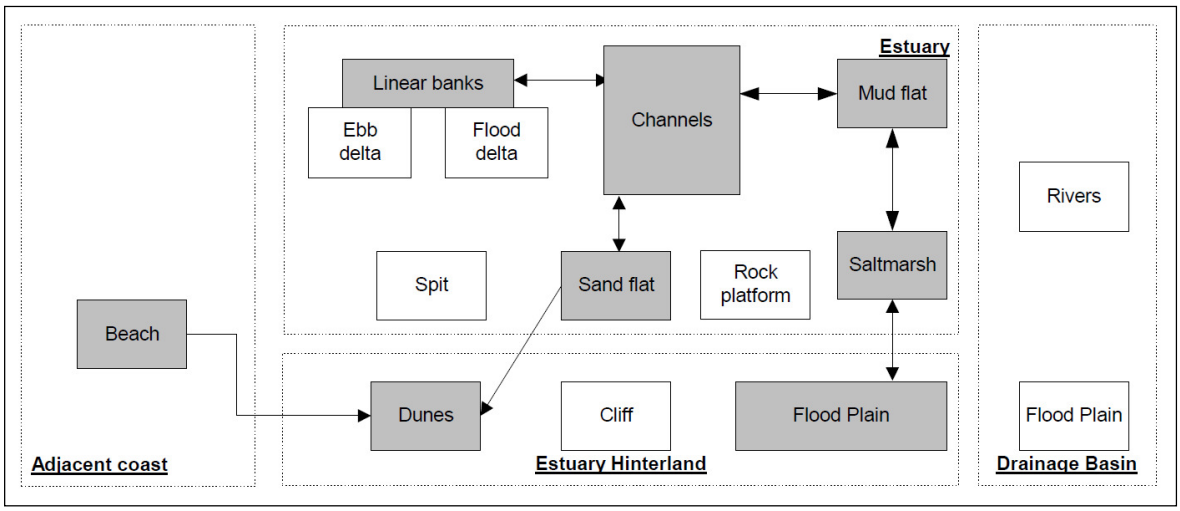

Figure 2.5: ***A systems diagram for a generic embayment (reproduced from ABPmer et. al., 2008: Crown copyright (Defra); 2009)***

2.3.1.4 Tidal Inlets

Tidal inlets occur where a low relief coastal plain has been flooded by sea level rise and separated from the coast by a barrier beach or spits, to form a lagoon. The lagoon is usually connected to the sea by one or more openings, which are principally maintained by tidal currents. This type of estuary is similar to the spit enclosed drowned river valley type, although the lagoons tend to be larger and are thus more affected by locally generated waves. Fine sediments usually accumulate within lagoons and mud flats and salt marsh are common features. In shallow areas these sediments may be re-suspended by locally generated waves and transported out of the lagoon by tidal flows and diffusion (Isla, 1995). The EstSim systems diagram for a generic tidal inlet is given in Figure 2.6.

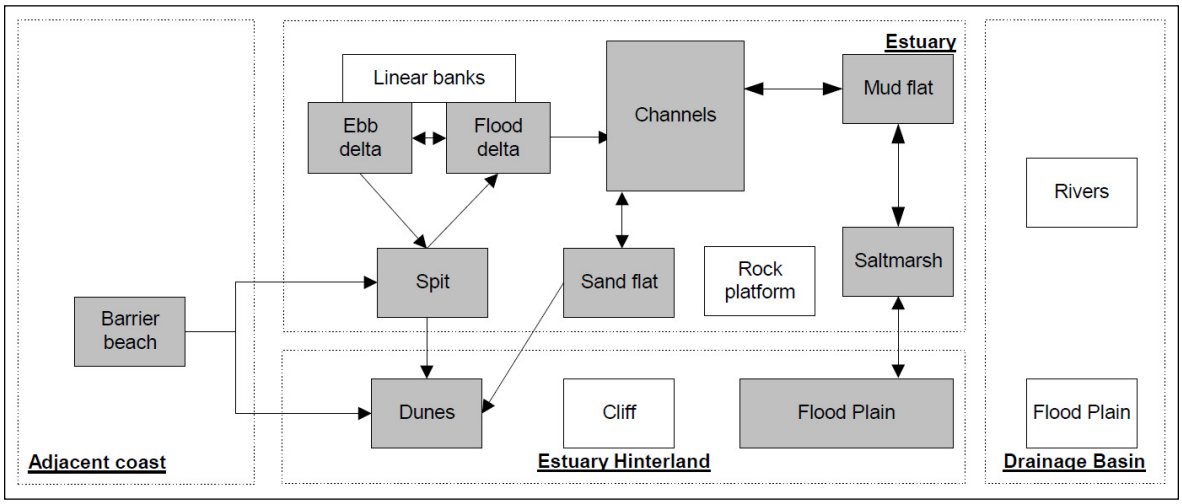

Figure 2.6: ***A systems diagram for a generic tidal inlet (reproduced from ABPmer et. al., 2008: Crown copyright (Defra); 2009)***

2.4 Morphological Evolution and Equilibrium

The concept of equilibrium in estuaries originally arose from observations that the ratios of certain physical dimensions appear to remain constant in time (O'Brien, 1931; Langbein, 1963; Escoffier, 1940); for example cross-sectional area at the mouth has been related to tidal prism at spring tide. A state of equilibrium requires a balance between sedimentary infilling and erosion and negative feedbacks, whereby increased sedimentation will tend to increase erosion and vice-versa; for example, infilling at the mouth of an estuary will reduce the cross sectional area and hence increase the velocity of tidal flows, which will eventually become strong enough to prevent further infilling. Infilling of estuaries has also been linked to flood dominance and erosion to ebb dominance because the shorter duration tidal phase with higher peak velocities will usually transport more sediment than the slower but longer lasting phase. It has been suggested that some estuaries, which have reached a state of equilibrium, may import sediment during spring tides and export sediment during neap tides (Dyer, 1997).

Estuaries are primarily depositional environments in which pre-Holocene valley systems are filled with marine and fluvial sediment until an equilibrium state has been reached. Any rise in sea level will tend to modify this equilibrium state and create accommodation space, causing further infilling to occur (Pethick, 1984; Van Goor et.al., 2003). An

equilibrium state implies a balance between the morphology and environmental variables such as tides, waves and fluvial flows. Rising sea levels affect these variables (e.g. increasing tidal flows) causing an adjustment to the morphology to maintain the equilibrium (i.e. infilling). Since neither sedimentation nor erosion can continue indefinitely, a non-equilibrium estuary must have either undergone a recent change in conditions, have a slow rate of morphological adjustment (e.g. due to a restricted sediment supply) or have some other restriction to morphological change (e.g. hard geology preventing further erosion). Equilibrium relationships have been used to develop a number of Top-Down morphological models and have been applied to individual morphological elements (e.g. channels and tidal flats) and geographical areas as well as entire estuaries. Further details of this are given in Chapter 3.

As well as sedimentary infilling and erosion, morphological adjustment to environmental forcing factors (e.g. waves, tides and river flows) includes the development of specific geomorphic features such as channels, spits, barrier beaches, sand flats, mud flats and salt marshes. The systems diagrams shown in Figures 2.1 to 2.6 show how the presence of particular forcing factors and geomorphic elements can influence the morphological evolution of an estuary and determine which other morphological elements are likely to be found. This approach has been shown to be capable of predicting which elements may be present given particular forcing variables and imposed geological conditions. Further details are given in Chapter 3.

CHAPTER 3

MORPHOLOGICAL MODELLING OF ESTUARIES

3.1 Introduction

The modelling approaches used for predicting the morphological evolution of estuaries can be categorised as Bottom-Up, Top-Down and hybrid models. Bottom-Up models are able to reproduce micro to macro-scale morphological changes over short time-scales by simulating the constituent processes at the smallest possible scales of time and space. Top-Down models, on the other hand, use data analysis and empirical rules to predict morphological change over the long term, while hybrid models use combinations of Bottom-Up and Top-Down techniques. The modelling techniques discussed in this chapter are given in Table 3.1; however, the number of available models and methods of analysis for estuaries is large and this list is not exhaustive.

Bottom-Up models are best suited to assessing short term, localised changes; for example, localised changes to sedimentation patterns. A typical application might be the determination of sedimentation rates and hence dredging requirements for a proposed port development. These models employ numerical techniques to solve partial differential equations, which describe currents, waves and sediment transport processes; however, due to an accumulation of uncertainties and the effect of complex interactions and feedbacks between the morphology, processes and ecology, these models are not well suited to predicting long term changes in estuarine morphology.

Top-Down methods include a range of data analysis techniques and theoretical analyses and although more qualitative in nature, are often better suited for the task of understanding and predicting medium to long term morphological changes. These techniques generally use information from the current and past behaviour of the estuary to make future predictions under similar conditions to those that have occurred in the past or

following a specific change, such as an increased rate of sea level rise. They can be sub-divided into data analysis methods and equilibrium based methods.

Hybrid models use combinations of Bottom-Up and Top-Down techniques to try to bridge the gap in capability between these approaches; for example, to make medium term predictions with better temporal and spatial resolution than possible using a purely Top-Down approach. Other model types include rule based models and cellular automata based models.

Bottom-Up Models		Hydrodynamic modelling Wave modelling Sediment transport modelling Particle tracking Morphological bed-updating models
Top-Down Models	**Data Analysis Techniques**	Holocene analysis Accommodation space Regression techniques Historical trend analysis Sediment budget analysis Salt marsh analysis Expert geomorphological analysis
	Equilibrium based methods	Regime relationships Form analysis Tidal asymmetry analysis Inter-tidal form analysis Estuary translation (rollover) ASMITA
Hybrid Models		Hybrid regime model Realignment model Hybrid Inverse model Hybrid energy based models Salt marsh models
Other Models		The ESTSIM prototype simulator Cellular automata based models

Table 3.1: Morphological Modelling Techniques for Estuaries

3.2 Bottom-Up Type Models

Bottom-Up, process based models use numerical schemes to solve mathematical descriptions of processes such as tidal flows, waves and sediment transport. A number of commercial software packages such as MIKE 3 (DHI, 2011), DELFT 3D (Deltares Systems, 2012) and TELEMAC (TELEMAC Consortium, n.d.), are available, which

include modules for the calculation of flow conditions, wave conditions and sediment transport.

These models are able to accurately simulate the hydraulic processes that are observed in estuaries and are best suited to predicting short term morphological change. They also tend to require significant computational effort and when applied to long term morphological simulations, uncertainties related to initial and boundary conditions, sediment transport processes and unrepresented processes (e.g. biological processes) tend to outweigh the benefits of using these models (Karunarathna et. al., 2008).

3.2.1 Hydrodynamic Modelling

The computational hydrodynamic models used to calculate flows in coastal and estuarine settings typically employ numerical solutions to the Navier-Stokes equations for incompressible fluid flow or the two dimensional (2D) shallow water equations, depending on the required output. For example, Equations (3.1) to (3.5) are the governing equations used by the basic version of the Telemac 3D model (Desombre, 2013). These equations are the three dimensional (3D) Navier Stokes equations with a free surface, changing in time. In this version of the equations any variation in density is assumed to be negligible in terms of mass conservation, pressure variation with depth is assumed to be hydrostatic (although a non-hydrostatic version of this model is also available) and density variations are not taken into account in the gravity term (the Boussinesq approximation).

$$\frac{\partial U}{\partial x}+\frac{\partial V}{\partial y}+\frac{\partial W}{\partial z}=0 \tag{3.1}$$

$$\frac{\partial U}{\partial t}+U\frac{\partial U}{\partial x}+V\frac{\partial U}{\partial y}+W\frac{\partial U}{\partial z}=-g\frac{\partial Z_s}{\partial x}+\upsilon\Delta(U)+F_x \tag{3.2}$$

$$\frac{\partial V}{\partial t}+U\frac{\partial V}{\partial x}+V\frac{\partial V}{\partial y}+W\frac{\partial V}{\partial z}=-g\frac{\partial Z_s}{\partial y}+\upsilon\Delta(V)+F_y \tag{3.3}$$

$$p = p_{atm} + \rho_0 g(Z_s - z) + \rho_0 g \int_z^{Z_s} \frac{\Delta\rho}{\rho_0} dz \tag{3.4}$$

$$\frac{\partial T}{\partial t} + U\frac{\partial T}{\partial x} + V\frac{\partial T}{\partial y} + W\frac{\partial T}{\partial z} = \upsilon\,\Delta(T) + Q \tag{3.5}$$

Equation (3.1) is a mass continuity equation, where U, V and W are the three-dimensional velocity components in the Cartesian x, y and z directions. Equations (3.2) and (3.3) are momentum equations, in which Z_s is the free surface elevation, υ is a diffusion coefficient and F_x and F_y are source terms, which include the effects of wind, the Coriolis force and bottom friction. Equation (3.4) is a hydrostatic pressure equation, where P_{atm} is the atmospheric pressure, ρ_0 is a reference density and $\Delta\rho$ is the density variation. The last equation relates to a tracer T, which may be passive or active. Active tracers relate to quantities that affect the hydrodynamics and these are primarily temperature and salinity. Q is a source or sink for the tracer.

3.2.2 Wave Modelling

Wave models can be divided into phase resolving models, which compute the movement of the sea surface at a resolution smaller than the wave length, and phase-averaged models, which compute the statistical properties of the sea surface. Examples of the phase resolving and phase averaged model types are FUNWAVE (Kirby et. al., 1998) and SWAN (Booij et. al., 1999), respectively.

Phase resolving wave models are hydrodynamic models that simulate the propagation of ocean waves. The 2D hydrodynamic modelling techniques used to simulate river tidal flows are generally not suitable for this purpose because the governing equations are only valid when the wavelength greatly exceeds the water depth. These models therefore use governing equations that include Boussinesq terms, to allow short waves to be modelled and account for deviations from hydrostatic pressure distribution due to vertical accelerations. The resolution of phase resolving models must also be smaller than the wavelength of the modelled waves, which increases the computational requirements and effectively limits their application to small areas, such as harbour developments.

This type of model is capable of accurately simulating the shoaling, refraction and diffraction of individual short waves as well as partial reflection and absorption at porous boundaries (Madsen and Warren, 1984).

Phase-averaged wave models are based on the wave Spectral Action Balance Equation and may include terms for effects such as wave generation, dissipation and propagation as well as bottom friction and depth induced breaking in shallow water (Booij et. al., 1999). Phase averaged wave models are capable of accurately simulating wave conditions at very large scales in deep and intermediate water depths but have strong limitations in certain near-shore areas, such as harbours (Rusu and Soares, 2013).

3.2.3 Sediment Transport Models

Sediment transport equations that have been developed for steady flow in rivers are often used to calculate transport by tidal currents (Soulsby, 1997). A smaller number of methods have been developed to calculate transport in the presence of combined waves and currents (e.g. Grass, cited by Soulsby, 1997 and Van Rijn, 1989).

Separate formulae are often given for bed-load and suspended-load, although some methods calculate the total sediment load directly (e.g. Engelund and Hansen, 1972, cited by Soulsby, 1997). Where calculated separately the calculated values for bed-load and suspended-load can be summed to give the total sediment load; however, it is important that the formulae used are compatible and matched at a well-defined height (Soulsby, 1997).

Van Rijn (1989) compared results from the Engelund and Hansen, Ackers and White and Van Rijn (1984) sediment transport formulae using 486 sets of river data and 120 sets of estuary data (Van Rijn, 1989). Results were expressed as a proportion of results that were accurate to within a factor of two for each method, as shown in Table 3.2. It can be seen that, for this data set at least, the Van Rijn method performs better than the other two, particularly for the estuaries data.

	Percentage of results between -50% and +100%	
Method	**River Data (486)**	**Estuary Data (120)**
Engelund-Hansen	64%	33%
Ackers-White	63%	26%
Van Rijn (1984)	76%	89%

Table 3.2: Percentage of results accurate to within a factor of 2 for different sediment transport formulae

3.2.3.1 The Method of Van Rijn

Bed load and suspended load formulae for transport by currents were developed by Van Rijn (1984a,b), following Bagnold's energy based approach and assuming that bed load is dominated by particle saltations. This method was later extended to include transport by combined currents and waves (Van Rijn, 2007a,b).

Instantaneous bed load due to combined waves and currents is given by Equation (3.6), where f_{silt} is a 'silt factor' ($f_{silt} = d_{sand} / d_{50}$, with $d_{sand} = 62\mu m$ and $f_{silt} = 1$ for $d_{50} > d_{sand}$), D_* is a dimensionless particle size parameter, $\tau'_{b,cw}$ is the instantaneous grain-related bed shear stress due to currents and waves and $\tau'_{b,cr}$ is the critical bed shear stress. The time-averaged bed load transport rate can be obtained by integrating this formula with respect to time, over the wave period.

$$q_b = 0.5 f_{silt} d_{50} D_*^{-0.3} \left(\tau'_{b,cw} / \rho\right)^{0.5} \left(\tau'_{b,cw} - \tau'_{b,cr}\right) / \tau'_{b,cr} \tag{3.6}$$

The suspended sediment transport rate is obtained by integrating the velocity and sediment concentration profiles over the water depth. Here c and u are the suspended sediment concentration and flow velocity at height z above the bed, h is the water depth and a is a reference height (related to the bed form height).

$$q_s = \int_a^h c\, u\, dz \tag{3.7}$$

Van Rijn also extended his method to allow calculation of separate transport rates for a number of particle size fractions within the bed (Van Rijn, 2007c) and also developed parametric equations for both bed load and suspended load transport (Van Rijn, 2005). Further details of these methods are given in Chapter 4.

3.2.4 Bed Updating Models

The output from hydrodynamic models can be used to calculate sediment transport rates using one of the many available sediment transport equations, which are then used update the bed levels to create a morphological bed-updating model. This is achieved using a sediment budget equation; for a 1D model this is given by Equation (3.8) (Soulsby, 1997):

$$\frac{\partial \zeta}{\partial t} = -\frac{1}{1-\varepsilon}\left(\frac{\partial q_b}{\partial x} + D - E\right) \tag{3.8}$$

where ζ is the bed level relative to an arbitrary datum and ε is the bed porosity. For sand transport over large space and time scales, the deposition (D) and erosion (E) terms can be replaced by substituting the bed load with the total load (Equation 3.9).

$$\frac{\partial \zeta}{\partial t} = \frac{1}{1-\varepsilon}\frac{\partial q_t}{\partial x} \tag{3.9}$$

In Equation (3.9) it is implicitly assumed that the total load transport can be determined from the local flow conditions and the bed composition; this type of model is known as an equilibrium model (Tayfur and Singh, 2007). For bed load and suspended sand transport this is often a reasonable assumption, since the transport rate can quickly adjust to changing flow conditions. However, silt and clay sized particles can remain in suspension for significant periods of time and in this case it may be necessary to use Equation (3.8); an advection-diffusion equation can then be used to model the transport of suspended sediment (Van Rijn, 2007d). Eulerian numerical schemes tend to produce artificial diffusion and non-physical oscillations if used to solve the advection-diffusion equation, particularly where advection is the dominant process, and alternative Eulerian-Lagrangian methods (e.g. Younes and Ackerer, 2005) are therefore commonly used. Where the local suspended sediment concentration exceeds the equilibrium concentration (that determined

by the local flow conditions and bed composition) deposition may occur, although for cohesive sediments the critical bed shear stress for deposition is usually lower than it is for erosion (Whitehouse et. al., 2000).

Morphological bed updating models have been used in conjunction with 1D hydrodynamic models to predict changes in fluvial morphology. These models must necessarily make assumptions regarding the lateral distribution of erosion and deposition of bed material and are unable to account for secondary currents such as occur at meander bends in rivers. In coastal and estuarine settings 2D or 3D hydrodynamic models are typically preferred (Papanicolaou et. al., 2008).

3.2.5 Particle Tracking Models

Particle tracking models use output from hydrodynamic models to track the movement of finite number of representative particles. They have the advantage of being able to trace the path of particles from a particular source, such as an oil spill or other source of pollution. In the simplest models each particle moves with the same velocity as the surrounding fluid, with diffusive processes represented by random perturbations to its trajectory (Dyke, 2007). Particle tracking sediment transport models are able to determine the destination of dredged spoil and trace the movement of contaminated particles; however, if sand transport is to be included, processes such as burial and re-emergence, initiation of motion, bed load transport and suspended load transport should be included, as in the SandTrack model developed by HR Wallingford (Soulsby, et. al., 2007). The SandTrack model has also been developed into a morphological bed updating model by associating each tracked particle with a volume of sediment, which is deposited on the bed as a 'lens' of sediment at specified intervals to generate the new bed morphology (Huthnance, et. al., 2007).

3.3 Top-Down Type Models

The Top-Down methods can be divided into data analysis and equilibrium based approaches. Data analysis methods include accommodation space analysis, sediment budget analysis, Holocene analysis, historical trend analysis and salt marsh analysis, while

equilibrium based approaches include regime theory, form analysis, tidal asymmetry analysis, inter-tidal form analysis and estuary translation. Expert geomorphological analysis can employ any combination of these methods, as well as Bottom-Up modelling, to gain the best possible understanding of the estuary behaviour and its probable future evolution (HR Wallingford et. al., 2006).

3.3.1 Data-Driven Models

A range of techniques are available for the analysis of current and historical data including tide and wave records, bathymetric surveys, borehole records, sedimentary records, remote sensing and geophysical surveys. Statistical, spatial and time series data analysis techniques can be used to gain an improved understanding of an estuary, for example by identifying underlying trends and cycles, associations between parameters or probability distributions (Emery and Thompson, 2001). Such methods can be used to forecast future changes by extrapolation and include statistical methods such as Empirical Orthogonal Function (EOF) analysis, which can identify temporal trends in spatial datasets (e.g. Karunarathna et. al., 2008).

Accommodation space is the estuary volume within which deposition could occur, defined as the volume between the bed and the maximum water level. Changes in bed and water levels over time therefore result in changes to the accommodation space. Where accommodation space is limited channel migration and extensive sand and mudflats are common features, whereas stable channels and the development of salt marsh are characteristic features where the accommodation space is increasing. An examination of historical changes in accommodation space can give an indication of the stability of the estuarine system, while the potential accommodation space above the maximum water level can give an indication of the future evolution of the system for a range of future sea level rise scenarios (EMPHASYS Consortium, 2000).

A sediment budget analysis can be carried out to identify and quantify the sediment fluxes, sources and sinks within an estuary, which should then balance. Data sets used include bathymetric surveys, suspended sediment concentration measurements, bed material density measurements, fluvial discharge measurements and discharge measurements at the

estuary mouth. Computational hydrodynamic and sediment transport models may also be used. Sources of sediment can include river load, marine import and erosion from cliffs, sub-tidal areas, inter-tidal areas and salt marsh within the estuary, while sinks may include dredging, marine export and deposition within sub-tidal areas, inter-tidal areas and on salt marshes (HR Wallingford et. al., 2006).

In Holocene analysis the evolution of the estuary since the end of the last ice age (approximately 15,000 years ago), is studied. Mean sea levels have risen by about 100m during this period (Perillo, 1995) and, for the majority of estuaries, this has been a major influence in their development. An investigation can make use of a variety of data and methods including borehole records, seismic surveys, particle size and mineral analysis, radio carbon dating, pollen analysis, foraminifera analysis and geoarchaelogical analysis. An improved understanding of the past evolution of an estuary can often provide insights into present form and future evolution of an estuary (ABPmer, 2008a).

Historical trend analysis complements Holocene Analysis by focussing on historical timescales, typically from around 200 years ago up to the present day. Data sources can include published papers and parliamentary records, as well as land registry archives, maps, charts, aerial photography, topographic and bathymetric surveys, remote sensing imagery and anecdotal evidence. Morphological changes can be identified by visual comparison of the data, which may be aided by GIS tools, while statistical methods can be used to generate and analyse time-series of quantitative data (e.g. volumetric changes or changes in plan form position). Where possible, observed changes are related to causes such as sea level rise or anthropogenic intervention (HR Wallingford et. al., 2006).

Salt marsh analysis is related to and may form an important component of a historical trend analysis. It is typically used to study erosion and accretion, through analysis of movement of the marsh edge and changes in elevation. Aerial photography, satellite imagery, maps and charts (historical and current) are used together with LIDAR and CASI remote sensing data. LIDAR provides accurate, high resolution elevation data while CASI data is used in conjunction with ground truthing to identify the extent and type of marsh vegetation (HR Wallingford et. al., 2006).

3.3.2 Equilibrium Based Models

These models are based on the assumption that the estuary is in a state of equilibrium and that following a perturbation (e.g. rising sea level or engineering works) it will adjust its morphology to restore the equilibrium relationship (Townend, 2005).

In regime theory a simple power law relationship is used to link hydrodynamics and morphology. A number of relationships have been developed for tidal inlets and entire estuaries. A common relationship used for tidal inlets is based on that proposed by O'Brien (1931):

$$A = f(\Omega) \tag{3.10}$$

$$A = C\,\Omega^{q} \tag{3.11}$$

Where A is the cross sectional area of the inlet and Ω is the tidal prism. The function f often takes the form given in Equation (3.11), where C and q are determined empirically or from theoretical considerations (e.g. Hughes, 2002). The regime equations for entire estuaries are given in Equation (3.12) (Langbein, 1963):

$$A \propto Q^{p},\ B \propto Q^{q},\ C \propto Q^{r} \tag{3.12}$$

where A is the cross sectional area, B is the top width and C is the mean depth. The exponents p, q and r can be derived empirically by fitting data at cross sections along the estuary. Since there can be expected to be scatter in the data around the best fit line, the estuary must be initially iterated until a close fit at all points is achieved. Alternatively, the initial error at each point can be held constant (HR Wallingford et. al., 2006).

In form analysis power law relationships are fitted to the longitudinal profile and plan form of the estuary, resulting in a simple equation to describe its three dimensional geometry based on a small number of parameters. The results can be used to study the magnitude of estuary transgression in response to sea level rise and in the design of inter-tidal channels (ABPmer, 2008b).

Tidal asymmetry analysis relates to differences in tidal current velocities during flood and ebb tides, which can cause a net movement of sediment. Tidal asymmetry has been related to morphology (Pethick, cited by ABPmer, 2008c), with deep, wide channels and inter-tidal areas below mean sea level producing flood dominance and narrow 'slot' channels with extensive high elevation mud flats producing ebb dominance. Net accretion in flood dominant estuaries and erosion in ebb dominant estuaries may then cause each type to evolve towards the other, producing a dynamic equilibrium where estuary oscillates between the two types.

Tidal asymmetry can be assessed using time-series plots of stage and velocity or using a velocity-stage plot, which shows the relative magnitude of velocities at different water levels. A number of equations have also been proposed, giving an 'asymmetry ratio' based on estuary form parameters such as channel depth, low water area, high water area and tidal amplitude (e.g. Dronkers, cited by ABPmer, 2008c).

The relationship between tidal asymmetry and morphology is complex, with peak velocity, slack duration and sediment supply all having an effect. However, a study of tidal asymmetry, how it has changed over time and how it may be affected by future changes can be a useful tool in establishing a qualitative conceptual understanding of estuary behaviour in Expert Geomorphological Analysis studies (ABPmer, 2008c).

Inter-tidal form analysis relates the cross shore profile to influences such as tidal range, wave conditions, sediment supply and sediment properties. It can provide a conceptual model for understanding the sensitivity of inter-tidal profiles to current and wave forcing and a method for predicting future changes to the inter-tidal profile caused by changes in current or wave forcing or sediment supply (HR Wallingford et. al., 2006).

A descriptive typology for mudflats was developed for the INTRMUD project (Dyer, cited by ABPmer, 2008d) based on factors including tidal range, wave energy, sediment supply, cross-shore slope, zones present (upper, middle and lower, defined relative to tidal frame), sediment density, bed-forms, organic content and biology (e.g. worms, bivalves and micro-organisms). Typology can give an idea of mud-flat behaviour through comparison with other flats of the same type.

The dominant sources of energy affecting mudflats are tidal currents (cross-shore and shore-parallel) and waves. Cross shore currents are proportional to width of the mudflat, while shore-parallel currents are related to depth. Where waves are the dominant energy source, it has been found that mudflat profiles tend to be more concave than otherwise. Equilibrium mudflat profiles can be determined based on the idea that the time averaged deposition must be equal to the time averaged erosion at all points on the profile.

Estuary translation or rollover is a general concept that describes a possible response to sea level rise, where material is eroded from inter-tidal areas near the mouth and transported to the head of the estuary resulting in a landward transgression of the estuary. As erosion in some areas provides sediment for accretion to occur in other areas, a sediment balance can be maintained while the estuary is translated landwards and upwards and the hydraulic regime is maintained whether in equilibrium or not (ABPmer, 2008e). Estuary translation has been studied using the ASMITA model (Stive et. al., 1998; Van Goor et. al., 2003; Hinkel and Klein, 2009; Hinkel et. al., 2013).

3.4 Hybrid Models

Hybrid models combine bottom-up process based modelling with top-down modelling approaches, usually with the aim of improving their long term predictive capability. The available hybrid models include coupled hydraulic and regime relationships, coupled hydraulic and entropy relationships, zero divergence of sediment flux (sediment balance) and coupled hydraulic and energy relationships (EMPHASYS Consortium, 2000). Some of the models included in this section (e.g. ASMITA) lack any direct representation of the processes but are distinguished from Top Down methods by their formulation as computational models, as opposed to methods of analysis. Others are quite similar to bottom up type models but use simplified methods to represent the processes.

3.4.1 Sediment Balance Models

These models are based on the concept of equilibrium between the prevailing hydrodynamic conditions and the morphology. If the system is disturbed from equilibrium, for example by the removal of sediment by dredging, then a sediment demand

is created, resulting in deposition. Examples of this type of model include the ESTMORPH model and ASMITA.

In the ESTMORPH model (Wang et al., 1998) an estuary is schematised as a series of cross sections, each consisting of a channel, a low tidal flat and a high tidal flat. Equilibrium morphology for each element is represented by an equilibrium cross sectional area, for the channel or an equilibrium height, for the tidal flats and is related to the tidal volume, the tidal range and the total area of the basin. Local equilibrium sediment concentrations are then calculated based on ratio of the actual cross sectional area or height to the equilibrium value and a global equilibrium concentration. Deposition occurs when the actual concentration exceeds the equilibrium concentration and vice versa. Lateral transfer of sediment between the high tidal flat, low tidal flat and the channel is modelled as a diffusive process, while sediment movement along the channel is modelled according to an advection-diffusion equation. Advection is based on a residual velocity, calculated using 1D flow model.

The ASMITA model (Stive et. al., 1998) was developed to study the morphodynamic interaction between a tidal lagoon and the adjacent coast. The model uses similar concepts to ESTMORPH and is schematised as a number of discrete morphological elements, categorised as tidal flats, channels and deltas. An equilibrium volume is defined for each element as a function of the tidal range, tidal prism and the basin area. Equilibrium concentrations are then related to the ratios of the actual volume of each element to the equilibrium volume and a global equilibrium concentration. Sediment flux between adjacent elements is then related to the difference between the sediment concentration in each element and a set of horizontal exchange coefficients, while erosion and deposition rates are related to the difference between the actual and equilibrium concentrations and a set of vertical exchange coefficients.

Sea level rise causes the actual and equilibrium volume of each element to change, as these are defined relative to the high and low water level and the tidal prism. Following such a change ASMITA will predict the response of the system of a period of years or decades. Other changes causing a disturbance from the equilibrium state may also be modelled using ASMITA, such as the removal or deposition of sediment due to dredging operations.

Guidance on the setting of values for the horizontal and vertical exchange coefficients is given by Wang (cited by Rossington and Nicholls, 2008). The rate of volume change can now be calculated for each element and the morphological evolution computed by stepping through time.

The ASMITA model was further developed during the UK Estuaries Research Programme using Matlab software, to allow models with any chosen combination of elements to be set up and run (Write et. al., 2007). Rossington and Nicholls (2008) applied ASMITA to four UK estuaries to predict the critical rate of sea level rise that would trigger losses of 25%, 50%, 75% and complete loss of the inter-tidal volume.

3.4.2 Realignment Model

A realignment model was developed during the UK Estuaries Research Programme, using a combination of bottom-up hydrodynamic modelling and top-down ASMITA-type sediment transport modelling to predict morphological change following the implementation of a managed realignment scheme (Huthnance et. al., 2007).

This model uses output from a 2DH hydrodynamic model (TELEMAC 2D) and a parametric wave model to determine spatially distributed equilibrium concentrations within the model domain, while sediment transport is modelled as a purely diffusive process. Erosion or deposition (E) is then determined at each location from the product of the sediment settling velocity and the difference between the sediment equilibrium concentration and the actual sediment concentration.

$$E = w_s\left(c_E - \bar{c}\right) \tag{3.13}$$

Time averaged local diffusion coefficients $\left(\overline{D}_x\right)$ in this model are related to the square of the velocity according to Equation (3.14), where the diffusion coefficient at the model boundary (D_{bnd}) is determined by calibration and the variables v and v_{bnd} are the instantaneous local and boundary velocities.

$$\overline{D}_x = \frac{1}{T}\sum \delta t \frac{v^2}{v_{bnd}^2} D_{bnd} \tag{3.14}$$

Since the model is intended to predict morphological change within a newly created set-back area, no empirical data is available for setting C_E and an alternative process based method was adopted instead. If deposition is assumed to occur continuously, in proportion to the time-averaged sediment concentration and the sediment settling velocity (w_s), and erosion (E) is determined according to the bed shear stresses due to waves and currents, the critical bed shear stress and an erosion rate constant (M_e), then the equilibrium concentration is equal to the time averaged concentration required for the time averaged deposition to balance the time averaged erosion:

$$C_E = \frac{M_e}{w_s}\int E\, dt \tag{3.15}$$

This model also includes a simple representation of the effects of saltmarsh, where saltmarsh is assumed to develop in all locations above a predefined elevation and the effects on erosion and deposition are included by setting the equilibrium concentration to zero in those areas. The effect of saltmarsh on currents and waves is approximated by increasing the friction coefficient in the TELEMAC model and reducing the wave heights according to Equation (3.16) (Dalrymple, 1984, cited by Spearman, 2010), where a and a_0 are the incident and attenuated wave height at distance x into the marsh.

$$\frac{a}{a_0} = \frac{1}{1+\alpha x} \tag{3.16}$$

$$\alpha = \frac{2C_D}{3\pi}\left(\frac{D}{b}\right)\left(\frac{a_0}{b}\right)\left(\sin\left(h^3\right)k\,h_{sm} + 3\sin\left(h\right)k\,h_{sm}\right)\left[\frac{4k}{3\sin\left(h\right)k\,h\left(2\sin\left(h\right)2k\,h + 2k\,h\right)}\right] \tag{3.17}$$

The coefficient α is calculated according to Equation (3.17), where h is the water depth, k is the wave number h_{sm} is the height of the marsh, C_D is the drag coefficient, D is the diameter of the saltmarsh stalks and b is a measure of the density of the marsh.

3.4.3 Inverse Hybrid Model

The inverse hybrid model is based on the premise that morphological evolution in estuaries can be characterised as a diffusive process plus a source function. This relationship is given in Equation (3.18), where h is the bed level at horizontal coordinates x and y, at time t, K is a diffusion coefficient and G is the source function.

$$\frac{\partial h}{\partial t} = K\left(\frac{\partial^2 h}{\partial x^2} + \frac{\partial^2 h}{\partial y^2}\right) + G(x, y, t) \tag{3.18}$$

The source function represents the combined effects of all non-diffusive processes on the morphological evolution and this is assumed to vary slowly in time. By inverting the model governing equation it is possible to calculate the spatial distribution of the source function using bathymetric data sets, which show the morphological change over an interval of time. Karunarathna et. al., (2008) used 20 historical bathymetric datasets for the Humber Estuary (UK), taken between the years 1851 and 2000, to calculate the evolution of the source function over time. This showed that changes to the source function were much less pronounced than changes to the bathymetry, as predicted. The source function was then extrapolated, using Empirical Orthogonal Function analysis, to allow the prediction of future changes to the morphology. Morphological predictions of the Humber Estuary given by Reeve and Karunarathna (2011) show that the Inverse Hybrid method can be successfully used to make morphological predictions.

The prediction of future changes to the morphology using the Inverse Hybrid Model is based on the assumption that future changes will not be affected by any change to the forcing conditions not reflected in the previous datasets. Therefore, the model may not be suitable for predicting changes caused by accelerated sea level rise or changes due to human activity within the estuary. It should also be noted that the success of the method and the quality and resolution of the predictions depends heavily on the availability of the data to derive the source function.

3.4.4 A Hybrid Regime Model

A hybrid regime model was also developed for the UK Estuaries Research Programme, for the prediction of long term changes in estuaries (Huthnance et. al., 2007). A 1D hydrodynamic model (ISIS or MIKE 11) is used to determine the peak flow at a series of cross sections, which represent the bathymetry. Regime relationships are then used to relate the cross-sectional area (A), top width (B) and mean hydraulic depth (H) to the peak flow at each cross section (Equation 3.12) and derive a best-fit solution for all model cross sections. Since there will be scatter about the best-fit line, morphological adjustments are made relative to the initial bathymetry (i.e. initial deviations from the best-fit line are retained at each update).

Following a change such as increased mean sea level or a change to the bathymetry representing engineering works, the 1D model is re-run to obtain the new peak flow at each cross section and the regime relationships are used to update the cross section geometries. Updating is achieved by stretching the cross sections horizontally and vertically, subject to constraints which must be specified to represent solid geology or engineering structures. The model is then iterated by re-running the 1D model and updating the geometry until the regime relationships are within a specified tolerance. Model results predict the ultimate outcome of a given change but not the timescale of the morphological adjustments.

3.4.5 Tidal Lagoon Model of Di Silvio et. al. (2010)

Di Silvio et. al. (2010) have developed a model to predict long-term morphological changes in a tidal lagoon. The model is based on the concept of transport concentration, which is defined as the tidally-averaged sediment concentration.

A simple hydrodynamic sub-model is used to derive two flow fields: the velocity field in maximum flow conditions and the residual water flux. Residual water fluxes may be caused by river flows or by differences in flow patterns between the flood and ebb tidal phases and are also adjusted in this model to account for the effect of tidal asymmetry, which can produce a net movement of sediment due to the non-linear relationship between flow velocity and sediment transport. The sediment transport components T_x and T_y are

given by Equations (3.19) and (3.20), where h_e is the effective water depth, C is the transport concentration, U_R and V_R are the depth averaged residual flow velocities and D_{ij} is a tidal dispersion tensor.

$$T_x = h_e\left(CU_R - D_{xx}\frac{\partial C}{\partial x} - D_{xy}\frac{\partial C}{\partial y}\right) \tag{3.19}$$

$$T_y = h_e\left(CV_R - D_{yx}\frac{\partial C}{\partial x} - D_{yy}\frac{\partial C}{\partial y}\right) \tag{3.20}$$

Effective water depth is related to mean sea level but is adjusted to allow for periods of non-submergence. Local values of the dispersion tensor are related to the square of the maximum velocity components.

Erosion and deposition (E) are related to the sediment concentration, an equilibrium concentration (C_{eq}) and a vertical exchange velocity (w), in accordance with Equation (3.21).

$$E = w\left(C_{eq} - C\right) \tag{3.21}$$

Equilibrium concentrations are in turn related to the long term average sediment concentration, maximum water flux, effective depth and a series of coefficients, which are obtained during model calibration. The vertical exchange velocity was set to 0.003 m/s, which is somewhat larger than the particle fall velocity.

The effect of salt marsh vegetation is included in the model by adjustments to the equilibrium concentration and vertical exchange velocity, depending on the bed elevation, which is assumed to be linked to marsh vegetation coverage. No mechanism for edge erosion of marshes is included in the model.

Di Silvio et. al. (2010) applied the model to an idealised tidal inlet and lagoon and tested for sensitivity to mesh size, wind intensity and fluvial sediment input.

3.4.6 ESTSIM Prototype Simulator

The prototype simulator described by Reeve and Karunarathna (2009) uses Boolean networks to develop a systems-based description of the geomorphological elements of an estuary. External forcings are defined as processes such as fluvial flows, tides and waves and a number of macro-scale geomorphic elements are defined, which may or may not be present depending on the type of estuary. A list of geomorphic elements for a generic estuary are given in Table 3.3.

Sub-system	Element
Adjacent coast	Beach
Estuary Hinterland	Dunes
	Cliff
	Floodplain
Drainage basin	Rivers
	Floodplain
Estuary	Linear banks
	Ebb delta
	Flood delta
	Spit
	Channels
	Sand flat
	Mud flat
	Rock platform
	Saltmarsh

Table 3.3: Geomorphic elements for a generic estuary

Rules specify the presence or otherwise of processes and geomorphic elements depending on which forcing factors and geomorphic elements are already present. For example, if both coastal cliffs (geomorphic element) and ocean waves (forcing factor) are present and there is no coastal protection, then there will be coastal cliff erosion (process). The

potential effects of changes can be assessed by adding features (e.g. coastal protection) and applying the rules until a new stable state is reached. While the simulator is not currently suitable for the evaluation of estuary management options, it provides a framework for formalising qualitative knowledge and can be used to explore potential geomorphological behaviour.

The simulator gives a possible evolutionary path for an estuary based on the positive and negative feedbacks between external forcings and morphological elements but at present is not able to distinguish between large and minor effects or indicate the timescale of potential changes (Reeve and Karunarathna, 2009).

3.4.7 Salt Marsh Ecology Models

A number of numerical salt marsh models have been developed in recent years to test hypotheses regarding salt marsh processes and to study the potential effects of different sea level rise scenarios (Fagherazzi et. al., 2012). Since regular inundation by the tide is the principle mechanism of sediment delivery to marsh platforms, their elevation is linked to mean sea level and tidal amplitude. In the 1D numerical model developed by Mariotti and Fagherazzi (2010) marsh biomass is related to elevation in a parabolic relationship, with biomass falling to zero at both the maximum and minimum elevations. Increased resistance to erosion, enhanced sedimentation and enhanced wave attenuation are then related to biomass.

Salt marsh biomass may be related to marsh elevation or the marsh inundation ratio (the proportion of time for which the marsh is submerged); however, the relationship between these parameters can be approximated as linear (Mud et. al., 2004) and hence the form of the biomass function is not affected by this distinction. Mudd et. al. (2004) proposed a linear biomass function relating biomass to the inundation ratio, in which biomass is at a maximum value when subjected to a maximum inundation ratio, $T_{i(max)}$ (i.e. the lowest elevation at which marsh can survive) and is zero at a minimum inundation ratio, $T_{i(min)}$. D'Alpaos et. al. (2007) have suggested a similar linear function for cases where the marsh is dominated by a single vegetation species (Spartina Alterniflora); however they find that in cases where a variety of halophytic species are present, a different relationship exists. In

these cases they propose a linear relationship with biomass increasing from zero, at the minimum marsh elevation (or maximum inundation ratio) to a maximum value, at the maximum marsh elevation (or minimum inundation ratio) and above. Similarly, in their numerical model for salt marsh evolution Mariotti and Fagherazzi (2010) have adopted a parabolic relationship between marsh biomass and elevation. These relationships are illustrated in Figure 3.1.

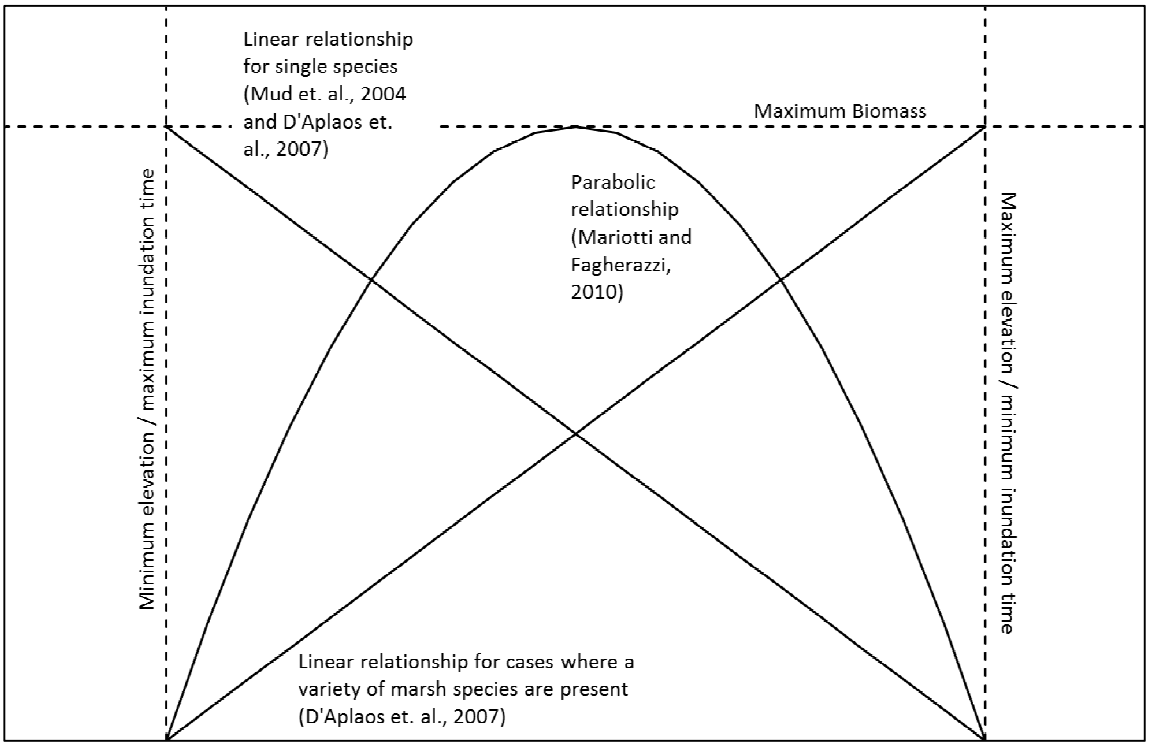

Figure 3.1: Relationships between marsh elevation / inundation time and biomass

Marsh is usually found between mean sea level and the mean high water level, although other factors such as tidal amplitude, latitude, temperature and sediment supply can influence marsh elevation (Fagherazzi, et. al., 2012). Mudd et al. (2004) have proposed a maximum value of 0.6 for $T_{i(max)}$ (expressed as a proportion of time for which the marsh is inundated), which will allow marsh growth below mean sea level.

3.4.8 Cellular Automata Based Models

Cellular automata based models have been shown capable of reproducing the macro-scale behaviour of certain physical systems without simulating the constituent processes in

detail, by focussing on higher level interactions and feedback mechanisms. Cellular automata (CA) are systems made up of arrays of cells, which are updated in a series of steps by the successive application of predefined rules. Rules specify the state of each cell based on the current state of a neighbourhood of nearby cells and for a two dimensional CA this usually includes the cell itself, its immediate orthogonal neighbours and may include its diagonal neighbours. Each successive application of the rules may represent an interval of time in some models but this is not the case for all models. CA models have been developed previously to study a wide range of processes, including river braiding, landscape evolution, aeolian dunes, tidal inlets, estuaries and coastal wetlands.

3.4.8.1 A CA Based Tidal Inlet Model

Nield and Walker (2005) developed a CA based model to derive an equilibrium morphology for an idealised tidal inlet. A Navier-Stokes hydrodynamic solver is used in this model to calculate flow velocity vectors in each cell and, where the critical velocity for sediment movement is exceeded, a sediment increment is distributed to the adjacent cells in proportion to the flow vectors. This model is designed to evolve towards an equilibrium state rather than predict the morphological evolution through time and only steady, unidirectional flows were considered. The global energy dissipation was calculated at each step and was shown to decrease as the CA progressed, suggesting this as a possible indicator of the attainment of an equilibrium morphology.

3.4.8.2 CEMCOS

CEMCOS is a process based CA based model that was developed by Dearing et. al. (2006), to simulate the morphological evolution of estuaries. Cell states are updated at each time-step according to rules based on simplified representations of tidal flows, waves and sediment transport.

The cross-sectionally averaged axial flow velocities are calculated using the method of Prandle (2003), and the transverse velocity is determined by equating the Coriolis force with the frictional drag or from continuity over the tidal flat, if this gives the higher value. An adjustment is applied to the (cross-sectionally averaged) axial velocity for each cell

based on the local depth; however no allowance is made for the effect on the velocity in surrounding cells.

Wave conditions on the open coast are calculated using Tucker's scaling, from Tucker, cited by Dearing et. al., 2006), with an adjustment applied in the near-shore area. Within the estuary wave hindcasting equations from the Shore Protection Manual are used. Comparison with the SWAN model (Booij et. al., 1999) showed that the results were improved by using an 'effective fetch', taking into account the fetch in the direction of the wind and in the ±45° directions. Sediment erosion and deposition are calculated using the equations for mud transport given by Whitehouse et. al. (2000) (See Section 4.5.2). Suspended sediment is then transported into three neighbouring cells depending on the direction of the tidal flow (flood or ebb). A volume of sediment is transported out of each cell (determined by change in water depth) and distributed equally into the three cells, excluding any that are unsubmerged. At the time of publishing the wave component had not been incorporated into the sediment transport calculations.

3.4.8.3 SLAMM

SLAMM (Sea Level Affecting Marshes Model) is a cellular automata designed to predict changes to shoreline wetlands resulting from long term sea level rise (Warren Pinnacle Consulting Ltd, 2012). This model does not attempt to simulate processes such as tides and wave action but uses a decision tree to determine changes in state for each cell at each time-step, based on relative sea level rise and sedimentation rates.

Areas protected by dykes are not permitted to change until inundated by over 2m (an adjustable model parameter). Second order effects due to spatial relationships between coastal elements are also included; for example, marsh erosion is determined based on fetch and proximity to open water leading to a change from marsh to tidal flat. Wetlands on the lee side of coastal barriers may be converted due to the effects of overwash or erosion of backshore and dune areas. Relative sea level changes are determined based on eustatic sea level rise and local historic subsidence and isostatic adjustment trends.

3.5 Review of Conventional Modelling Approaches for Estuaries

Although a wide range of morphological models are available for estuaries, the capability to make quantified predictions of change over medium time scales is still quite limited. Bottom-up models are able to reproduce the hydrodynamic and sediment transport processes with the greatest accuracy and detail but as well as being computationally demanding they can suffer from an accumulation of errors and uncertainties associated with boundary conditions, approximations and unrepresented processes. This can lead to large errors and long simulation times when long timescales are considered, although significant improvements in morphological updating techniques for medium to long term simulations have been made in recent years (e.g. the MORFAC approach; Roelvink, 2006; Lesser et. al., 2004; Ranasinghe et. al., 2011). Empirical information can be incorporated through model calibration; however, this usually limited to the setting of parameter values and they are typically unable to directly incorporate knowledge of higher level interactions, such as those specified in the ESTSIM model. Top-Down type models are able to directly use a wide range empirical information sources to predict general trends in long term behaviour but may lack the capability to predict the consequences of human induced changes or the crossing of tipping points. Hybrid models typically focus on a limited number of Bottom-Up components (e.g. hydrodynamic modelling) together with a Top-Down modelling approach (e.g. regime theory) to provide to some of the benefits of both approaches; however, none of these methods have the capability to incorporate a wide range of empirical knowledge while representing the essential mechanics of estuarine systems.

Existing cellular automata based models for river braiding and landscape evolution suggest a possible approach to building an estuarine morphology model capable of representing estuarine processes and incorporating a range of empirical information through the application of rules. It is anticipated that such a model would be capable of predicting of future trends in morphological behaviour with minimal computation effort, flexible enough to incorporate a range of human induced changes and able to identify potential step changes or other effects associated with interactions and feedbacks between the morphology, processes, geological constraints and estuarine ecology. The cellular

automata format has therefore been selected to form the basis of a new model, developed in this research, due to its computational efficiency, flexibility and the previous successes of cellular automata based models in reproducing a wide range of natural phenomena (Murray and Paola, 1994; Coulthard, 1999; Warren Pinnacle Consulting, 2012; Wolfram, 2002).

A novel, cellular automata based, estuarine morphodynamic model has been developed in this research and is described in Chapter 4.

CHAPTER 4

MODEL DESCRIPTION

4.1 Introduction

This chapter describes the development of a novel estuarine morphodynamic model based on the cellular automata format. As discussed in Chapter 1, it is the aim of this research to produce a flexible and efficient model capable of predicting medium term morphodynamic change in estuaries by concentrating on high level interactions between geomorphic elements and forcing factors. Following the review of existing estuarine morphodynamic models in Chapter 3, it was decided to use Cellular Automata as a platform for the development of the new model; however, to achieve this within a purely cellular automata based format would require that local rules capture changes to hydrodynamic processes such as tidal flows and waves resulting from changes to the morphology. Since these processes are affected by the states of a large number of non-local cells (e.g. tidal flows at the mouth are affected by the tidal prism of the entire estuary) this condition has been relaxed for the hydrodynamic and sediment transport processes. These processes are instead computed globally and represented in simplified form, in order to reduce computation times and to reduce the overall complexity of the model. Sand transport is computed according to local conditions with sediment transferred between neighbouring cells, while mud erosion and deposition are computed according to local rules and the diffusion of suspended sediment is computed globally at each time-step. Salt marsh is represented in the model according to local rules, with feedback occurring to the hydrodynamic and sediment transport model components.

4.2 Model Structure

The model incorporates simplified representations of tides, waves, sediment transport and saltmarsh ecology within a cellular automata based framework. The bathymetry is represented by a regular grid of square cells, each of which have properties of bed

elevation, flow depth and velocity, wave height, wave period and direction, sediment grain size distribution and salt marsh coverage. Non-erodible boundaries are included by setting a minimum bed elevation for each cell. The model has been developed in modular format, as shown in Figure 4.1.

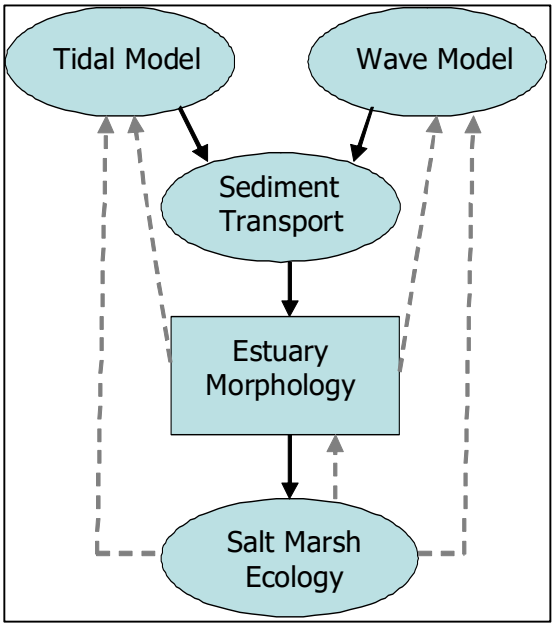

Figure 4.1: Model Structure

Cellular automata are spatially and temporally discrete computational systems. A two dimensional Cellular Automata is based on a regular grid of square cells, which have a finite number discrete potential states or a set of states composed of real numbers. Cell states are updated in a series of discrete time-steps, according to predefined rules, which specify changes to the properties of each cell based only on its current properties and the current properties of a local neighbourhood of cells. The local neighbourhood for a cell is usually defined as the four orthogonally adjacent cells or eight neighbouring cells, including the diagonally as well as the orthogonally adjacent cells (Berto and Tagliabue, 2012), as shown in Figure 4.2.

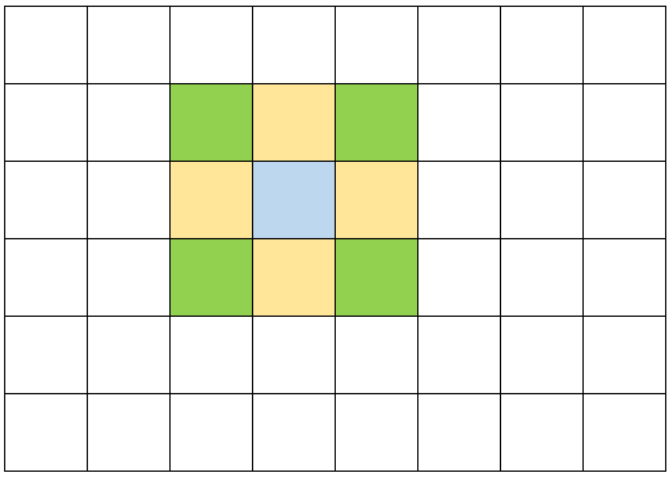

Figure 4.2: ***2D CA grid, showing the orthogonal (yellow) and diagonal neighbours (green) for one cell***

Change occurring to one particular cell can propagate through the model, to eventually affect the entire domain, as its immediate neighbours are affected followed by the neighbours of those cells and so on. Cellular Automata are able to reproduce a wide variety of complex behaviour seen in nature (Wolfram, 2002).

A flexible and pragmatic approach has been taken in developing the model and there have been a number of departures from the CA format, as discussed in Section 4.1 and described in detail in the following sections.

4.3 Calculation of Tidal and Fluvial Flow

Flow of water between cells is calculated based on mass continuity and bed friction. Other influences such as density driven circulation, secondary flows and the Coriolis Effect are not included in the present model. In order to apply mass continuity water levels must first be calculated for each cell at intervals within a complete cycle of spring to neap tides. Two methods are used to calculate water level: a slope based routing model for gravity flow in the inner estuary and a separate method for calculating tidal water levels. Both methods are applied to all cells as described in the following sections, with the larger of the two water level results carried forward for each cell, for the calculation of tidal and fluvial flows, as shown in Figure 4.3.

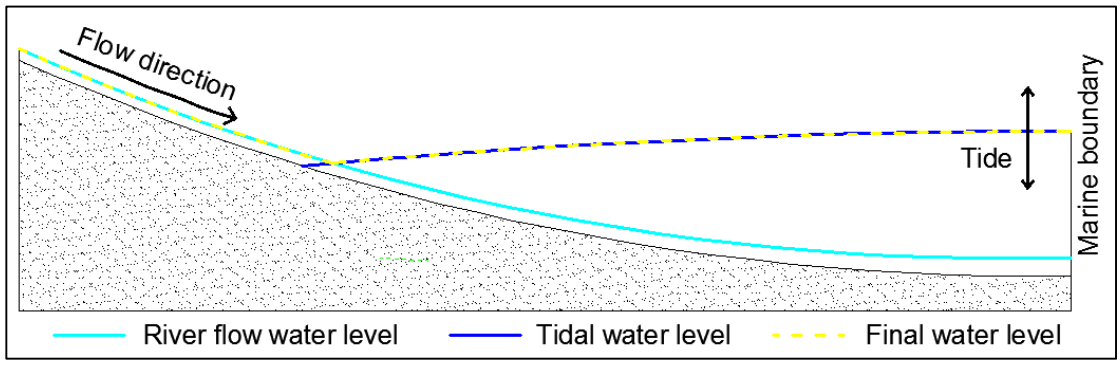

Figure 4.3: Calculated water levels

In the upper part of an estuary, depending on the phase of the tidal cycle, flow may be gravity driven and in this case it can be modelled using slope based routing models that have been developed for fluvial applications.

A slope based flow routing model has been developed to determine the water depth in each cell due only to the fluvial inflow, using an approach based on methods that have been used previously for river braiding and fluvial landscape evolution (Murray and Paola, 1994; Coulthard, 1999).

4.3.1 Depressions

Slope based routing models tend to fail at local depressions in the seabed because no positive slope exists to indicate the direction of flow from the lowest point in the depression; therefore, the following simple procedure was first developed to calculate the water level that would exist within any local depressions if they were filled with water, including a small residual slope leading to the lowest edge of the depression.

Each cell is checked to determine difference between its bed level and those of its four orthogonal neighbours. The difference between the bed level of the cell and that of the lowest orthogonal neighbour is given by Equation (4.1), where $z_{b,0}$ is the bed level of the cell in question and $z_{b,1}$ to $z_{b,4}$ are the bed elevations of the four neighbour cells, as illustrated in Figure 4.4.

$$\Delta z_b = z_{b,0} - \min\left(z_{b,1}, z_{b,2}, z_{b,3}, z_{b,4}\right) \tag{4.1}$$

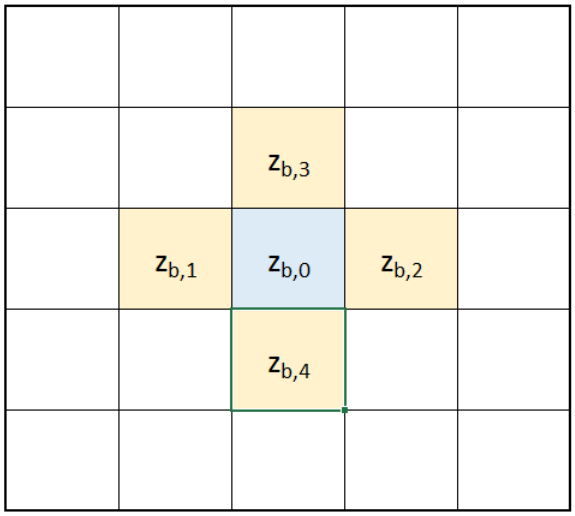

Figure 4.4: Cell locations in depression-filling procedure

If Δz_b is less than $+i/2$ (where i is a small increment, currently set to 1 x 10^{-8} m) for any cell then the depression filling procedure is started. The first step in this procedure is to raise the bed level in that cell to the maximum neighbour bed level plus i (See Figure 4.6). This rule is repeated for each cell in the model domain until all cells have bed levels that are at least $i/2$ m above the bed level of their lowest neighbour.

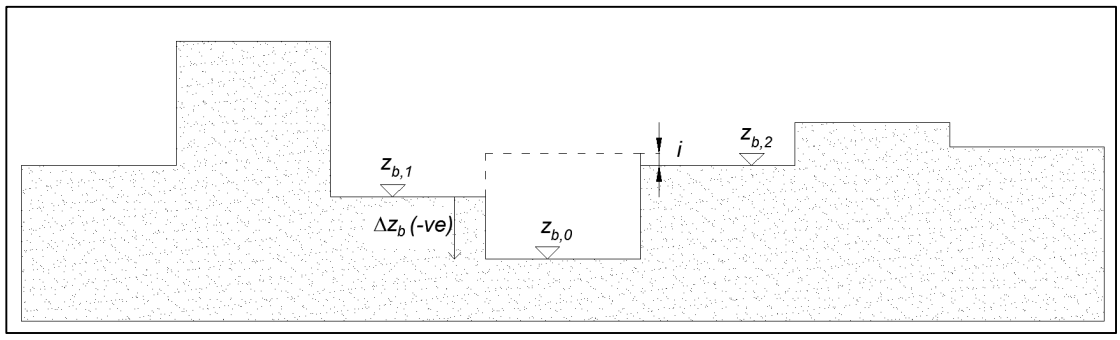

Figure 4.5: Bed level changes in depression-filling procedure

In order to reduce the computational requirements, a record is kept of all cells that have been linked to a depression and the procedure is repeated, only for those cells, until the depression is completely filled. The procedure ends when a complete cycle of checks ends without any further depressions being identified. The process is illustrated using a simple 1-dimensional example in Figure 4.6, where the numbers indicate successive iterations of the procedure.

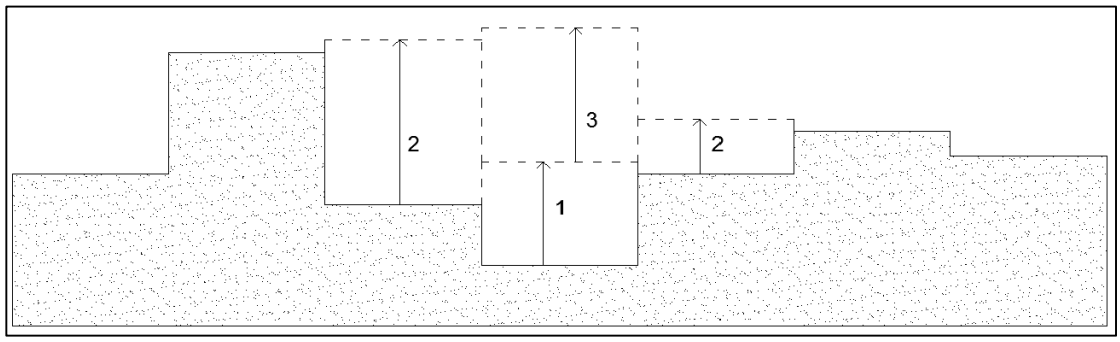

Figure 4.6: Bed level adjustments in depression-filling procedure, step 1

When the first stage of the procedure is completed, the adjusted bed levels of any cells over $1.1i$ above their lowest neighbour are reduced to the lowest neighbour bed level plus i and this step is also repeated until no further adjustments are required. This is illustrated in Figure 4.7. These modified bed levels are used only for the slope based flow routing procedure described in Section 4.3.2.

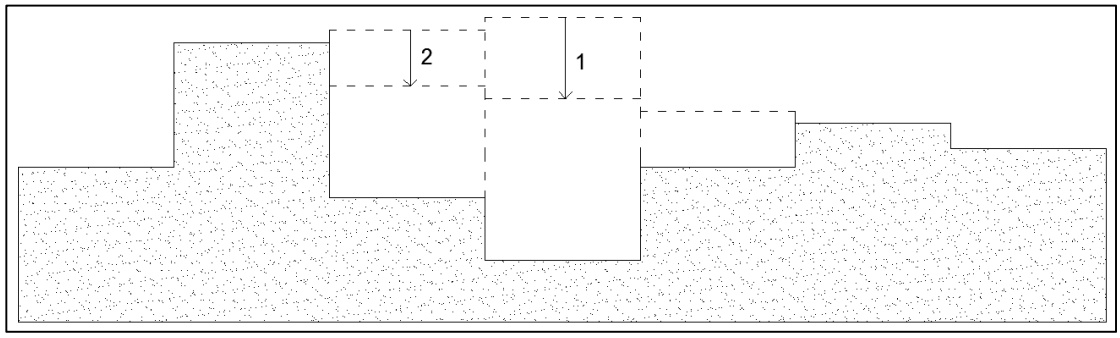

Figure 4.7: Bed level adjustments in depression-filling procedure, step 2

4.3.2 Down-Slope Flow Routing

River flow levels, above the instantaneous tidal water level (see Figure 4.3) are determined using a simple slope-based routing model. Flow paths are primarily determined based on local slopes but with some modification to allow the filling of channels (routing based only on slope results in the concentration of all flow to the lowest point in the channel bed). Routing is carried out in stages related to the four orthogonal grid directions, forwards, backwards, left to right and right to left. This process allows fluvial flows to be routed around complex channel geometries as illustrated in Figure 4.8, where dark blue cells indicate a fluvial inflow and light blue cell represent the channel geometry. The four

stages are each repeated several times if necessary to ensure that all inflow reaches the marine boundary.

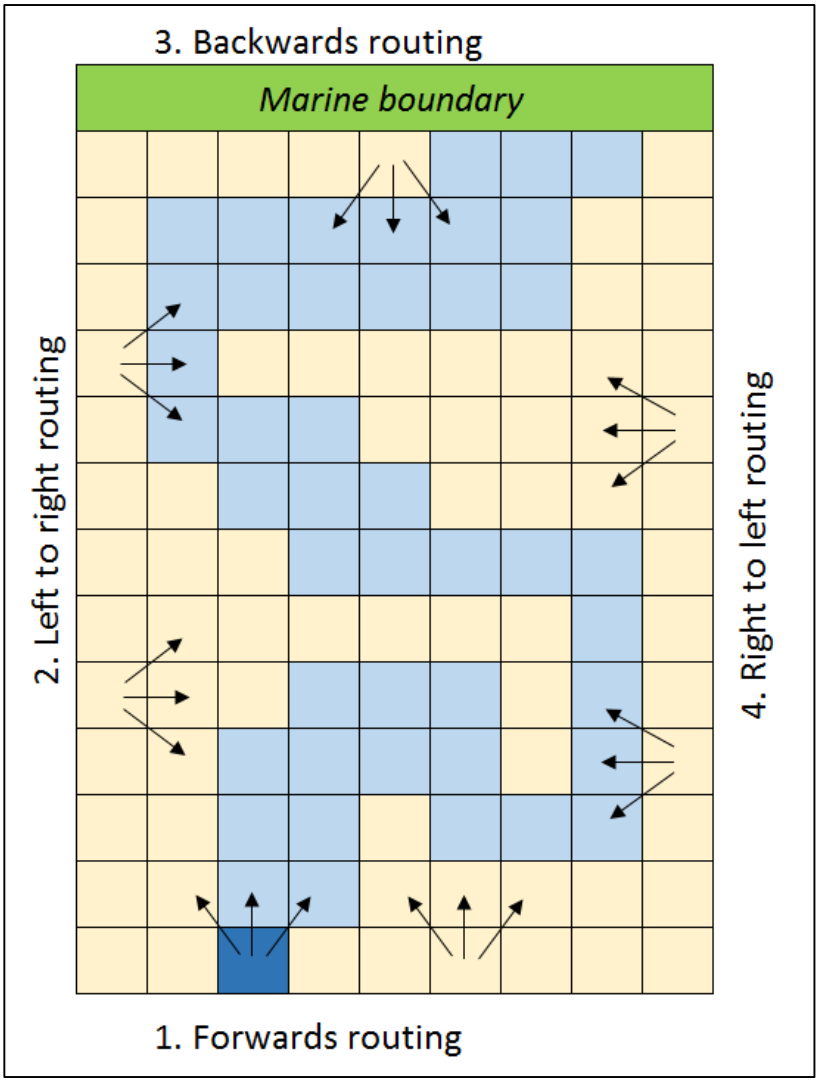

Figure 4.8: Illustration of slope based routing procedure

At each stage flow is routed one row (or column) at a time with flow from each cell routed to three cells in the next row. Flow out of each cell is routed into the three downstream adjacent cells as shown in Figure 4.9. The bed slope *s* for each of the three destination cells is first calculated from the difference in bed elevation between cells as follows:

$$s_1 = \left(z'_{b,5} - z'_{b,1}\right) / \left(L\sqrt{2}\right) \tag{4.2}$$

$$s_2 = \left(z'_{b,5} - z'_{b,2}\right)/L \tag{4.3}$$

$$s_3 = \left(z'_{b,5} - z'_{b,3}\right)/\left(L\sqrt{2}\right) \tag{4.4}$$

Where (i = 1,2,….6) are the bed elevations of the corresponding numbered cells in Figure 4.9, following removal of local depressions using the procedure described in Section 4.3.1 and L is the cell size.

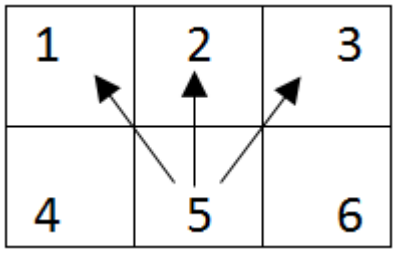

Figure 4.9: Flow directions in the slope based routing model

The water depth in each cell is calculated from the average of the positive slopes (s_m), according to Equation (4.5). This measure of slope has been found to give more stable results than alternatives, such as the greatest slope or average slope (Coulthard, 1999).

$$s_m = \left(s_1 + s_2 + s_3\right)/N_s \tag{4.5}$$

N_s is the number of positive slopes, in the range 1 to 3 and s_1, s_2 etc. are adjusted to zero if negative. Although local depressions have been removed from the lattice, since only three adjacent cell are considered at this stage it is possible for all three slopes to be negative (i.e. N_s = 0); in this case Equation (4.5) is not applied and the flow remains in cell 5, to be included in the next stage of the routing procedure (e.g. when routing flow from left to right). Otherwise, s_m is set equal to or greater than a minimum value (s_{min}); where s_{min} is an adjustable model parameter, which is related to the longitudinal bed-slope in the channel. This parameter is necessary to prevent the calculation of unrealistically large depths for small values of slope. For the simulations carried so far s_{min} has been arbitrarily set to 0.0001.

$$\hat{s}_m = \begin{cases} s_{\min} & s_m < s_{\min} \\ s_m & s_m \geq s_{\min} \end{cases} \tag{4.6}$$

The water depth h_s is then estimated using Equation (4.7), which is derived from the Chezy equation for steady uniform flow.

$$h_s = \left(\frac{Q_T}{C\,L\left(\hat{s}_m\right)^{0.5}} \right)^{2/3} \tag{4.7}$$

In this equation Q_T is given by $Q_T = \sum Q_i + \sum Q_i'$, where Q_i denotes inflows from upstream cells during the current stage of the procedure and Q_i' denotes inflows to the cell calculated during previous stages of the procedure. C is the Chezy roughness coefficient and L is the width of one cell (i.e. the cell size). The total water depth in cell 5, h_T, is given by Equation (4.8) and this is the final output for cell 5 unless a larger depth is calculated for the same cell at a later stage in the routing procedure.

$$h_T = h_s + z_b' - z_b \tag{4.8}$$

Where z_b is the bed level in cell 5 without adjustment by the depression-filling procedure. Flow is then routed to those downstream cells (Cells 1, 2 and 3 in Figure 4.9) for which the bed elevation is below the bed level of the source cell (Cell 5) plus the total depth of water (h_s) in that cell.

The outflow from cell 5 at this stage in the procedure is routed into cells 1, 2 and 3 in proportion to the recalculated slopes given by:

$$s'_1 = \left(\left(z_{b,5} + h_T \right) - z'_{b,1} \right) / \left(L\sqrt{2} \right) \tag{4.9}$$

$$s'_2 = \left(\left(z_{b,5} + h_T \right) - z'_{b,2} \right) / L \tag{4.10}$$

$$s'_3 = \left(\left(z_{b,5} + h_T \right) - z'_{b,3} \right) / \left(L\sqrt{2} \right) \tag{4.11}$$

Negative slopes are again reset to zero and the flow routed to cells in the next row are then calculated according to Equation (4.12):

$$Q_i = Q_o \frac{s'_i}{\sum_{i=1}^{3} s'_i} \tag{4.12}$$

where Q_o is the total outflow from Cell 5, in Figure 4.9 and Q_i (i = 1, 2, 3) is the inflow routed to Cells 1, 2 and 3.

This procedure is carried out in each of the four directions shown in Figure 4.8 and if necessary the entire procedure is repeated several times until all flow inputs have reached the marine boundary of the model. The process is further illustrated in Figure 4.10, where dark orange cells are those through which flow is routed at each stage, while cells highlighted in light orange cells show where flow is retained, to be routed further downstream, in another direction.

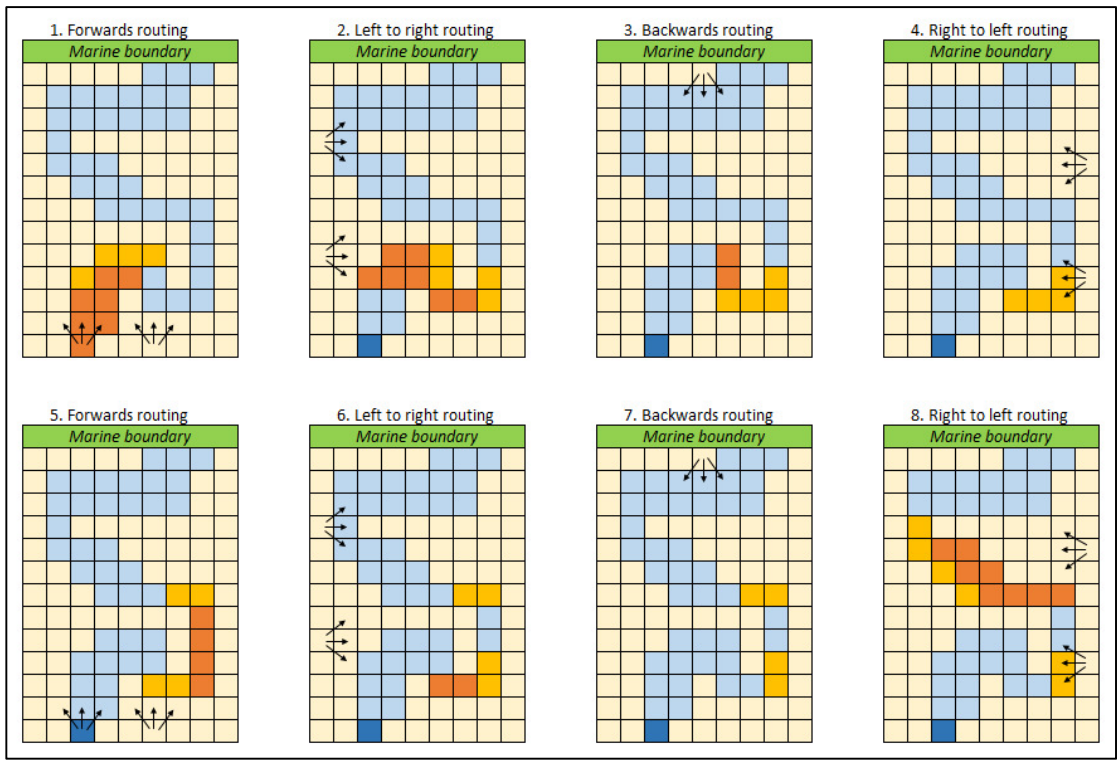

Figure 4.10: Slope based routing stages

In the example shown in Figure 4.10 the process would need to be repeated several more times to route all flow to the marine boundary. Channel meandering has not been observed to the extent shown in Figure 4.10, in the simulations carried out so far in this research; however, this example shows that the method is robust and able to cope with complex channel geometry.

4.3.3 Tidal Water Levels

If the tidal wave is assumed to propagate along the estuary with constant amplitude (a synchronous tide), water levels can be determined using Equation (4.13), following Dearing et. al. (2006):

$$z_{td} = z_m + Z\cos(kx_L - \omega t) \tag{4.13}$$

where z_{td} is the water level at time t and distance x_L from the estuary mouth, z_m is mean sea level, Z is the tidal amplitude, ω is the wave angular frequency, k is the wave number $(2\pi/c_L T)$ and c_L and T are the tidal wave celerity and period respectively. Since the celerity for shallow water waves is given by $c_L = (gh)^{0.5}$ (Linear Wave theory, where g is gravity and h is water depth; US Army Corps of Engineers, 2002), the wave number is a variable function of depth, and hence the term kx_L is replaced with $\Sigma(k_x \Delta x_L)$, where k_x is the wave number at x_L (calculated using the mean depth over cross section through the estuary) and Δx is the distance between cross sections. These values are calculated at a series of cross sections with water level at each section calculated using the sum $\Sigma(k_x \Delta x)$ for all cross sections between the one under consideration and the tidal boundary. The tidal amplitude is given by Equation (4.14).

$$Z = \frac{(Z_s + Z_n) + (Z_s - Z_n)\sin(\omega_{sn} t)}{2} \tag{4.14}$$

Where Z_n is the neap tidal amplitude, Z_s is the spring tidal amplitude and ω_{sn} is the frequency of the spring/neap tidal variation, which is approximated to exactly 29 tidal periods so that each subsequent cycle of spring to neap tides will be in phase with the first

and the calculation need only be repeated infrequently to allow for changes to the morphology.

If a more general method for calculating tidal water levels is needed (e.g. for estuaries with asynchronous tides) water levels can be determined using a numerical solution to the 1D Shallow Water Equations. In that case a 1D hydrodynamic model should be run for one complete spring-neap cycle of 29 tidal periods, with downstream boundary water levels (z_{bdy}) given by Equation (4.15).

$$z_{bdy} = Z\cos(-\omega t) \tag{4.15}$$

In this research an option has been developed to automatically generate and run a 1D numerical model using an external software package (ISIS) and read the water level results; however, the default method has been selected to be the use of Equation (4.13), due to its lower computational requirements. Whether calculated using Equation (4.13) or a numerical model, water levels are recalculated at intervals to allow for the effect of morphological changes on tidal wave propagation. Final water depths in each cell are determined from Equation (4.16).

$$h = \max(h_T, h_{td}) \tag{4.16}$$

Where h_T is the depth calculated by the slope based routing procedure (Equation 4.8) and $h_{td} = z_{td} - z_b$.

4.3.4 Interpolation of Water Levels

The calculation of tidal water levels requires the generation of cross sections, which are used either to determine the mean depth needed to apply Equation (4.13) or to generate a 1D numerical model of the estuary.

For the test scenarios described in Chapters 5 and 6, the model centreline is aligned with the grid axis and hence cross sections can be generated by taking bed levels along each row of cells; however, if the model is applied to a real estuary then it will be necessary to interpolate cross sections along lines perpendicular to the centreline of the estuary. This

processes has been automated by allowing the centreline to be defined using a series of coordinates, which are used to create a centreline comprising a series of straight sections and curves. Perpendicular lines are then interpolated from the centreline to a specified width, which can vary between sections. This process is illustrated in Figure 4.11, where the points *P1* to *P4* define the centreline.

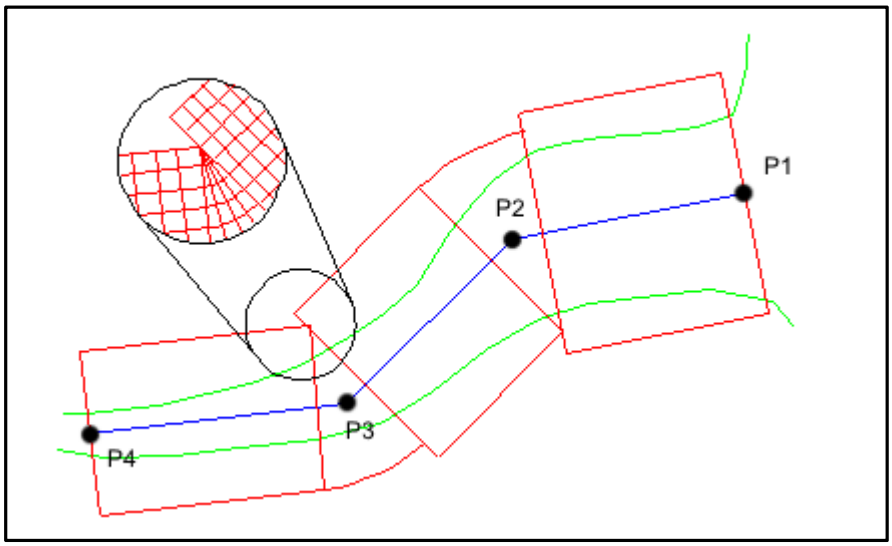

Figure 4.11: Interpolation of cross sections

The centreline is assumed to be fixed over time and hence the weightings, needed to calculate the interpolated cross sections and apply the calculated water levels to the grid cells are fixed and only need to be calculated once. The cell bed levels are divided into a regular array of triangles, with interpolated bed levels calculated at the surface of a plane formed by the triangle at that location. A similar approach is used to interpolate the calculated water levels into each grid cell.

4.3.5 Mass Continuity

Once the water levels have been calculated for each cell at each time-step, mass continuity can be applied to obtain a first estimation of flow between individual cells. During the first cycle of spring to neap tides, the flows are initially set to zero and during subsequent tidal cycles the flows are initially set to the previous values for the same stage in the spring-neap tidal cycle. In either case the flows are then adjusted to enforce mass continuity.

Cells are first ranked according to distance from the boundary in terms of connectivity via wet cells (those cells with a depth of water greater than a pre-defined minimum depth, h_{min}). All wet cells adjoining the boundary itself are assigned rank 1 and the flow direction towards the boundary recorded for each cell. Wet cells adjoining rank 1 cells, that have not already been assigned a rank, are then assigned rank 2 and the flow direction towards the rank 1 cell is recorded. This procedure is repeated until all wet cells with connectivity to the downstream boundary have been assigned a rank and flow direction.

Inflow at the upstream end of the model is assigned to cells that are specified according to the individual model setup. Starting with the highest ranked cells, outflow in the direction assigned to each cell (Q_o) is then determined based on mass continuity, according to Equation (4.17).

$$Q_o = Q_i - \left(A\Delta z_w / \Delta t\right) \tag{4.17}$$

Where Q_i is the sum of the inflows to the cell from all directions, except the outflow direction (both Q_o and Q_i may take negative values), A is the plan area of a single cell, Δz_w is the increase in water level between the start and end of the time-step (may be negative) and Δt is the time-step. Adjustment are made to flows from each ranked cell, in order of decreasing rank, until this has been completed for all cells.

Mass continuity adjustments are made at each time-step and are required because the term Δz_w may change due to changes in bed levels, since the minimum water level in a cell during a given time-step cannot be less than the bed level.

4.3.6 Flow Field Calculation

The final step in the calculation of tidal flows is an adjustment based on bed friction. Other influences such as density driven circulation, the Coriolis Effect and the effects of inertia are not included in the current method. A quasi steady-state approach is adopted and flows are adjusted using a head-balance approach, also known as the Hardy-Cross method for calculating flows in pipe networks (Featherstone and Nalluri, 1988).

This method is traditionally used to calculate head losses in pipes and thus the total hydraulic head at nodes, in pipe networks; however, only notional head losses are calculated here, in order to determine the required adjustments to the flow.

This method is an iterative procedure, in which successive adjustments are made to the flows until the total head loss around any given loop is close to zero (within a specified tolerance). Loops are formed from clusters of four adjacent cells as shown in Figure 4.12.

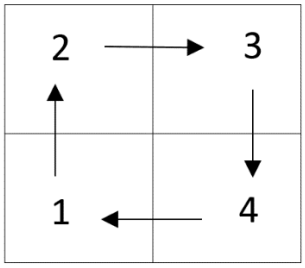

Figure 4.12: A loop of four cells used in the flow field calculation

The flow between each pair of cells in each loop is first identified, with clockwise flows assigned positive values and vice versa ($Q_1 = Q_{y,1}$, $Q_2 = Q_{x,2}$, $Q_3 = -Q_{y,4}$, $Q_4 = -Q_{x,1}$), where Q_x and Q_y are the flows crossing the right hand and upper edges of the numbered cells, respectively). A flow adjustment is then calculated using Equation (4.18).

$$\Delta Q = \frac{\sum K Q|Q|}{2\sum K|Q|} \tag{4.18}$$

The parameter K is a constant given by Equation (4.19):

$$K = \frac{1}{C^2 h^3 L} \tag{4.19}$$

where C is the Chezy coefficient, h is the water depth and L is the cell size. The correction ΔQ is then applied to the four flows in the loop. Similar corrections are made for all loops identified within the model domain and the largest correction, expressed as a proportion of the initial value of Q, is recorded. If this value is greater than the tolerance set for the model then the entire process is repeated for all loops.

Where islands of dry cells exist within the model domain, additional loops are used in the calculations. These loops are extended around any islands as shown in Figure 4.13 and flow corrections calculated and applied using the above equations.

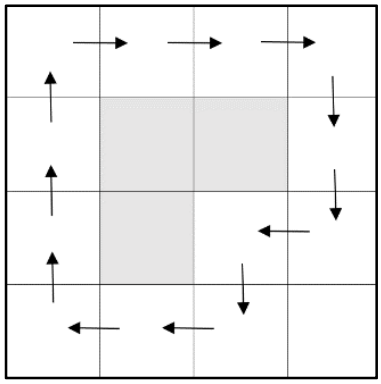

Figure 4.13: A loop of cells extended around an island of dry cells

Another special case in this procedure occurs at the tidal boundary, where the instantaneous hydraulic head is assumed to be a constant at all points on the boundary. Examples of loops created at the boundary are shown in Figure 4.14.

This procedure ensures that corrections are made for flows between any two adjacent cells. The only exception to this occurs in the case of a channel that is only one cell in width; however, in this case the flow is determined directly from the continuity calculation.

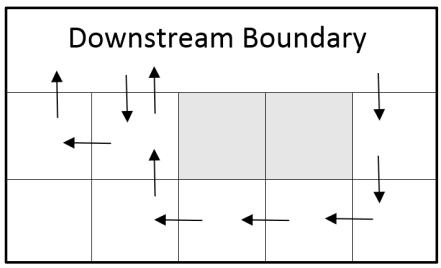

Figure 4.14: Formation of loops at the downstream boundary

4.3.7 Flow Model Verification

The flow model described above has been tested using a simple idealised estuary bathymetry. The model domain used for this test consisted of 250 rows of 21 cells (arbitrarily sized but similar to some smaller UK estuaries, such as the Deben), with each cell representing a 50m x 50m area of the bed. Results were taken at row 210 located 2 km

upstream from the marine boundary (where the marine boundary is at row 250). This location was selected to be near the marine boundary due to the higher currents expected at that location. Bed levels were set to give a uniform cross section profile for the entire length of the estuary, as shown in Figure 4.15.

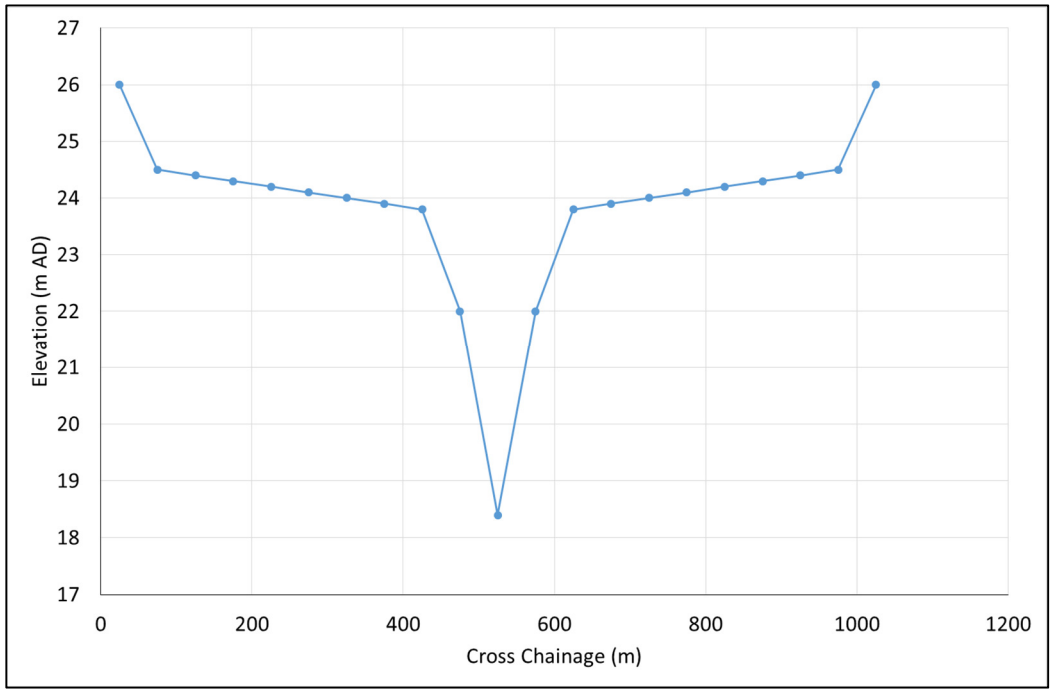

Figure 4.15: Cross section profile used for flow model verification

The mean sea level at the boundary was set to 23m AD and the tidal range to 4m. Figure 4.16 shows the axial and transverse velocities and the water depths calculated over a single tidal cycle at cross-chainage 275m (tidal flat) and Figure 4.17 shows the same results at cross-chainage 525m (channel centre).

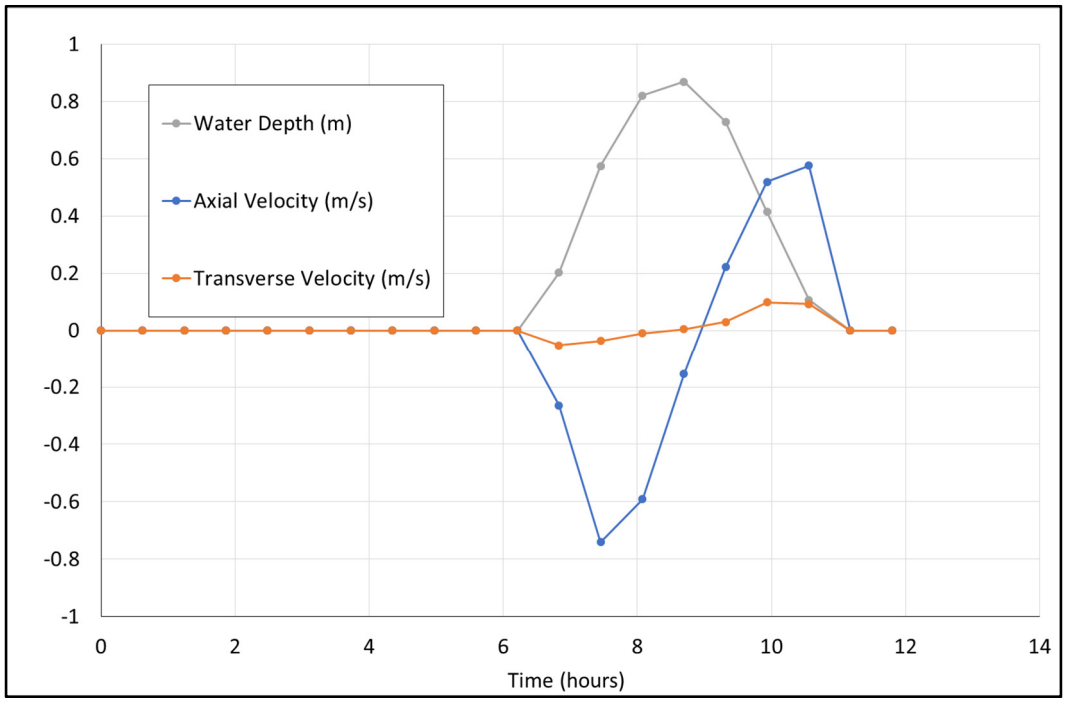

Figure 4.16: Flow calculation results at cross-chainage 275m

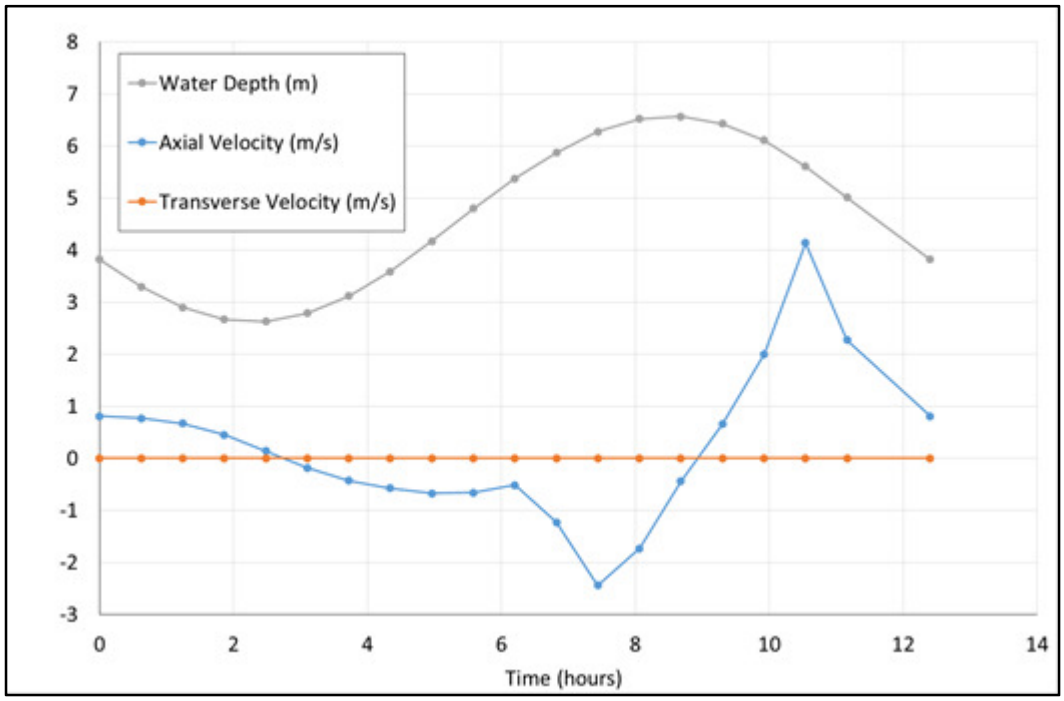

Figure 4.17: Flow calculation results at cross-chainage 525m

Figure 4.17 shows high peak axial velocities, particularly on the ebb tide. This may be due to the artificial bathymetry used for the test and in practice the high velocities might be expected to cause rapid erosion in these circumstances, increasing the dimensions of the

channel and thus reducing the peak velocity. Additionally, the depths shown in Figures 4.16 and 4.17 were generated using Equation (4.13), which may not be calculating water levels accurately in this case due to the artificial nature of the bathymetry.

To assess the impact of the method of water level calculation on these results the above simulations were repeated using the option to calculate water levels using a 1D hydrodynamic model. Figures 4.18 and 4.19 show the results generated when this option is selected.

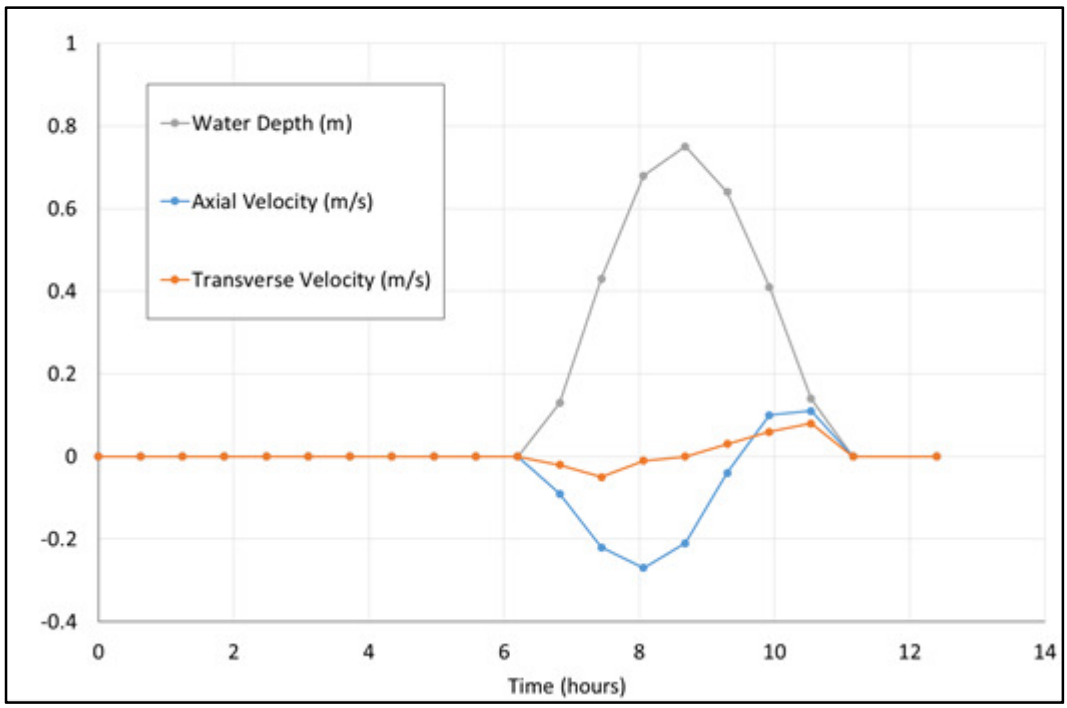

Figure 4.18: Flow calculation results at cross chainage 275m, using depths from a 1D numerical model

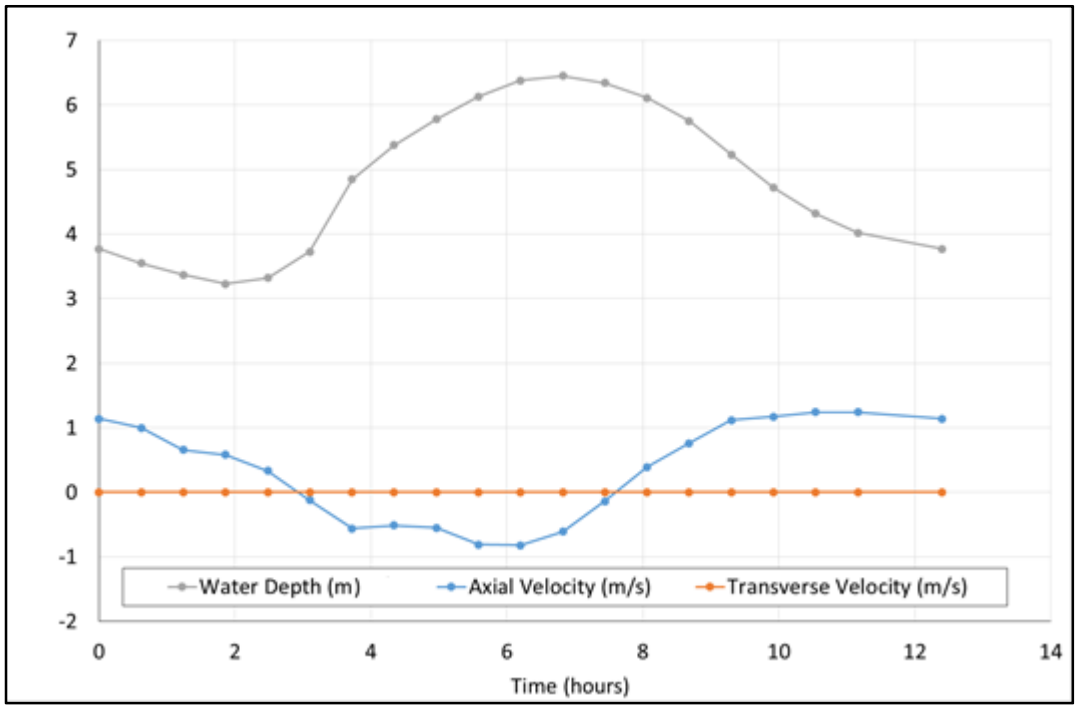

Figure 4.19: Flow calculation results at cross chainage 525m, using depths from a 1D numerical model

From Figure 4.19 it can be seen that the numerical model has predicted a steepening of the tidal curve on the flood tide. In Figures 4.18 and 4.19, peak velocities are significantly reduced from those in Figures 4.16 and 4.17, although the system still appears to be ebb dominant. These differences may be explained by better representation of the frictional effects on the propagation of the tidal wave in the 1D numerical model, causing a reduction in the rate of filling and emptying of the upstream tidal prism via the narrow channel; however, as discussed above, the bathymetry in this case is highly artificial and therefore this does not indicate that the default method for calculating water levels (Equation 4.13) is unsuitable in general.

To test the ability of the flow routing procedure to calculate flow fields around complex geometries, flow through an idealised tidal inlet was modelled. While this is not an intended application for the model, it provides a useful test, to demonstrate that the flow routing model is functioning as intended. Example flow fields, calculated by the model during the flood and ebb tides, are shown in Figures 4.20 and 4.21, respectively.

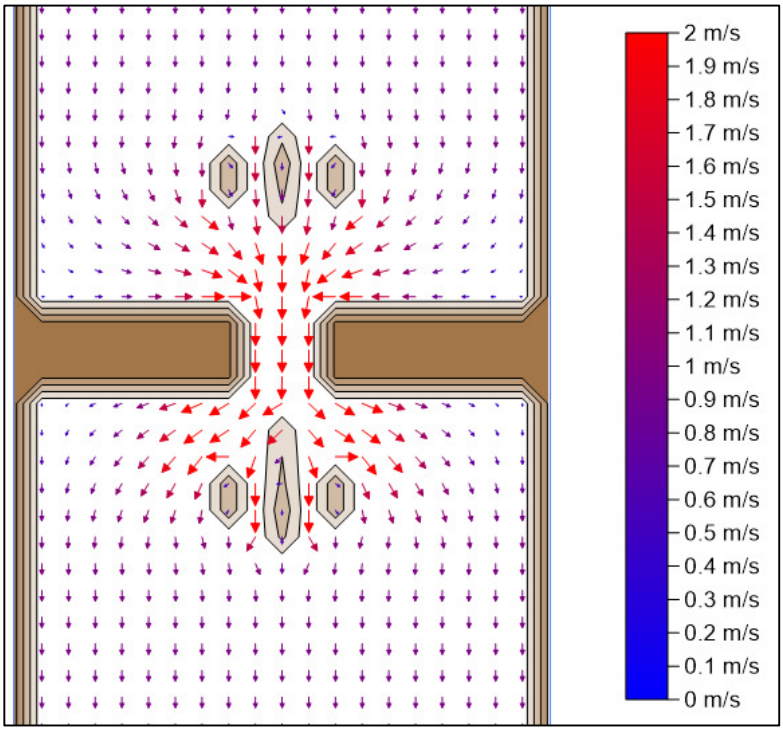

Figure 4.20: Example flow field calculated for an idealised tidal inlet: flood tide

The flow fields shown in Figures 4.20 and 4.21 show a smooth transition into and out of the inlet. As expected secondary currents are not reproduced but flow velocity is related to depth (bed level is constant is this scenario, except over the flood and ebb deltas). Velocity is also lower in the corners, adjacent to and at the left and right ends of the 'barrier beach', due to the increased distance and hence higher friction, required for flow to and from the inlet via these locations.

Some significant asymmetry is noted along the central column of vectors in both figures, even though the bathymetry is exactly symmetrical. This is due to an approximation in the way the velocity output was generated: in cases where flow is diverging to both the left and right from a single cell only the larger of the two values is included in the output file. Since in this case the bathymetry is symmetrical, flow across the left and right boundaries are almost equal and opposite but due to small errors, within the defined tolerance, one is slightly larger than the other. This affects the vector plots and the velocity magnitude used in the sediment transport calculations; however, all outflow directions are included in the allocation of sediment movements to adjacent cells (see Section 4.8).

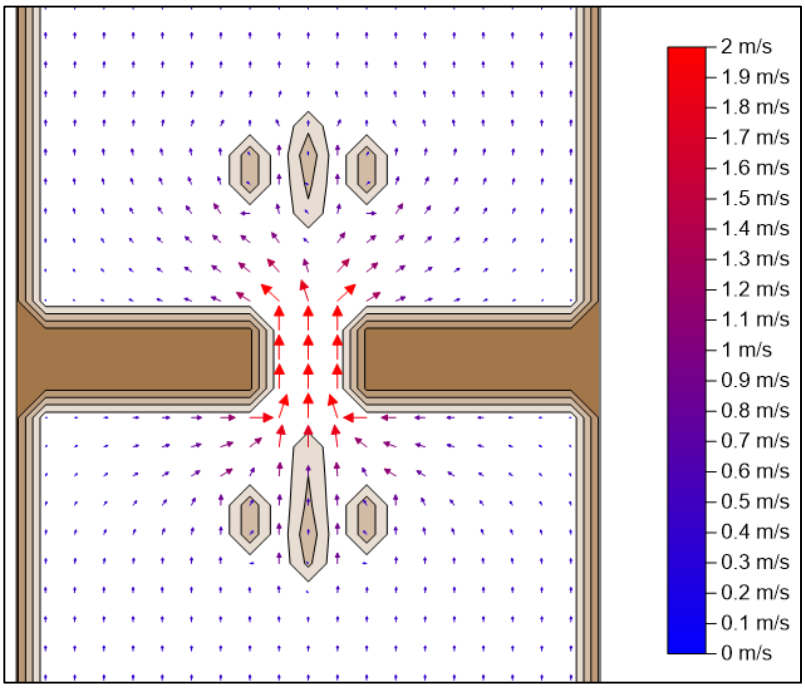

Figure 4.21: Example flow field calculated for an idealised tidal inlet: ebb tide

4.4 Locally Generated Waves

The locally generated wave height and period in the basin of the estuary are estimated using the wave hindcasting equations given in the Coastal Engineering Manual (Equations 4.21, 4.23 and 4.24; US Army Corps of Engineers, 2002). An effective fetch is estimated at each cell using a novel procedure, described below in Section 4.4.2, while wind speed and direction are generated using a simple statistical wind model. Only locally generated waves are included in the model.

4.4.1 Wind Speed and Direction

Following McWilliams and Sprevak (1982), wind speed is resolved into components in the direction of the prevailing wind and that perpendicular to the prevailing wind. A time series of prevailing wind speed is assumed to be normally distributed with a mean value of μ and standard deviation σ (where μ and σ are adjustable model parameters), while the perpendicular wind speed has zero mean and standard deviation σ.

Since it is intended that the sediment transport calculation (See Section 4.5), when averaged over the tidal cycle or spring-neap tidal cycle, should only vary according to changes in the morphology, the effects of wind speed variability must be averaged at each time-step. The distribution of prevailing wind speeds is therefore divided into 5 parts, including all wind speeds between -2σ and +2σ, as shown in Figure 4.22.

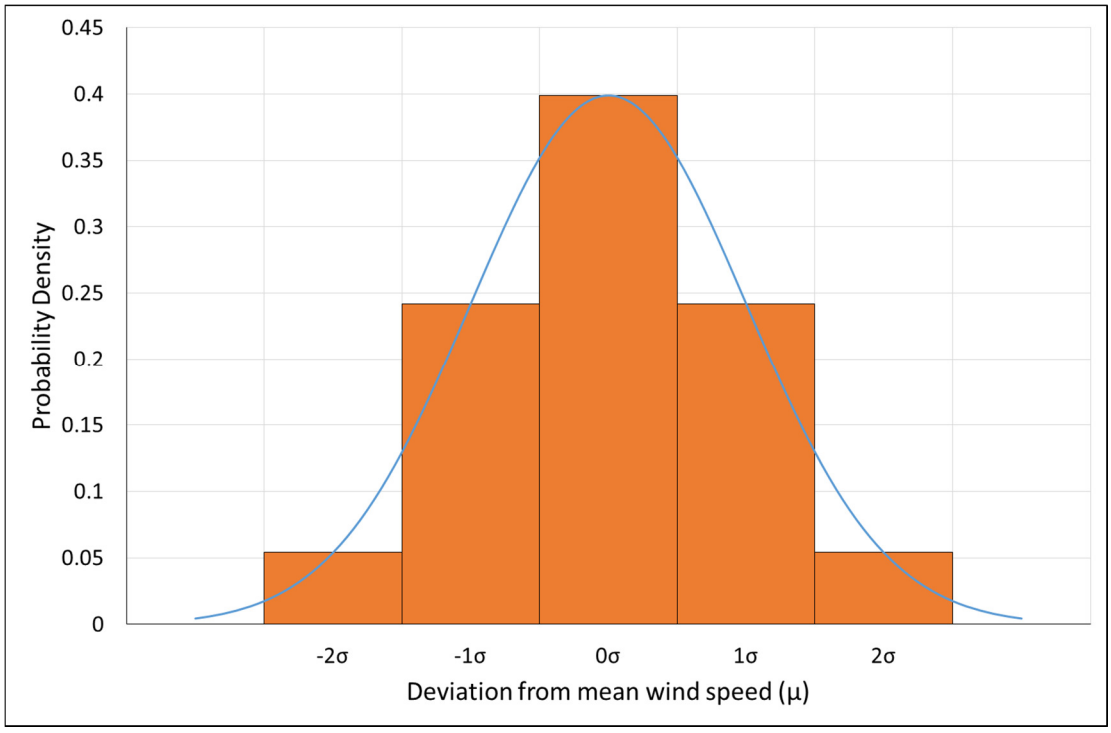

Figure 4.22: Distribution of wind speeds

The perpendicular wind speeds are similarly divided into 5 parts, giving 25 combinations of prevailing and perpendicular wind speed, which are combined by vector addition to give 25 wind speed magnitudes and directions. Probabilities are assigned to each of the 5 wind speeds in each direction according to the probability density function shown in Figure 4.22 and to the 25 wind speed combinations by multiplying the probabilities of the two components. Since the 5 wind speeds for each component do not include the full area beneath the probability density function, a small adjustment is made to ensure that the 25 probabilities add up to exactly 1.

The wave field and rates of sediment transport are calculated for all 25 wind speed combinations at each time-step. The calculated values of sediment transport are then

weighted according to their relative probabilities and summed, to give the final values of sand transport, mud erosion and mud deposition to be used to update the morphology.

4.4.2 Estimation of Wave Height and Period

Fetch is primarily a function of the bathymetry and wind speed and direction but in this model it is also affected by the depth of water, to allow for the effect of wave breaking. Fetch is estimated using an accounting procedure, starting from the 'up-wind' model boundary. This is the first or last row or column depending on the wind direction as indicated in Figure 4.23. For wind directions between -45 and +45 degrees (where a wind direction of zero degrees is aligned with the model lattice and blowing towards the model boundary) the procedure begins with row n, where each cell is assigned a boundary fetch (an adjustable model parameter). This fetch is reduced if necessary to limit the resulting wave height to the depth limited wave height (taken as $0.78H$). By reducing the fetch, as opposed to simply depth-limiting the calculated wave height, the wave heights for any 'down-wind' cells with larger depths are also reduced.

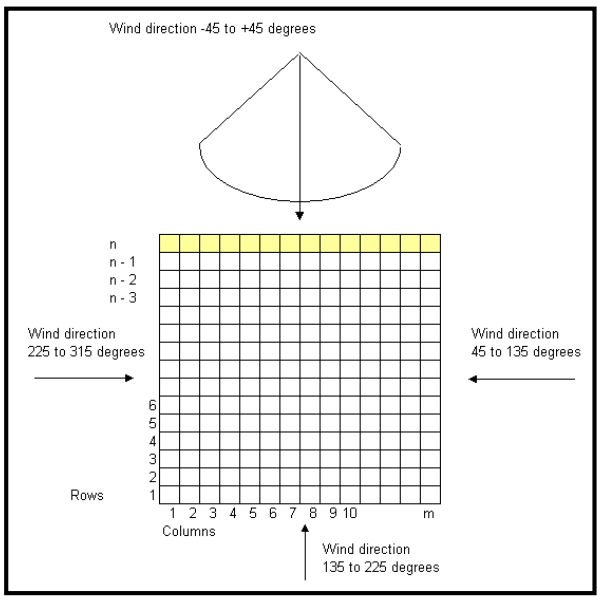

Figure 4.23: Procedure for estimating effective fetch

In this procedure, a point on the wave front in an individual cell is treated as the source of a semi-circular wavelet in an approach similar to Huygens' Principle for light waves (Huygens' Principle, 2013). Wave energy density is given by Equation (4.20) (linear wave

theory) and for fetch limited waves the energy based significant wave height can be estimated using Equation (4.21) (US Army Corps of Engineers, 2002). By combining these equations it can be seen that, for fetch limited waves, the wave energy density is proportional to the fetch (Equation 4.22).

$$\overline{E} = \frac{\rho_w g H^2}{8} \tag{4.20}$$

Where $\overline{E}$ is the wave energy density, ρ_w is the density of water, g is the gravitational acceleration and H is the wave height.

$$\frac{gH_{m0}}{u_*^2} = 0.0413 \left(\frac{gX}{u_*^2} \right)^{\frac{1}{2}} \tag{4.21}$$

Where H_{m0} is the energy based significant wave height $\left(H_{m0} \approx H_s\right)$, X is the fetch and u_* is the shear velocity (Equation 4.24).

$$\overline{E} \propto X \tag{4.22}$$

The distribution of wave energy from each cell can therefore be approximated by distributing the fetch to N_w cells in row $n-1$, in proportion to their subtended angles. For a semi-circular wavelet the energy is distributed over 180°, which would reach an infinite number of cells in the next row. Therefore, for practical reasons, the value of N_w was set to 7, covering an angle of 148°, for the simulations carried so far. If the water depth in any of the seven cells, other than the central cell, is below a predefined minimum depth then a proportion α_w of the fetch is laterally reflected; for example, if the depth in cell 1 (the left hand cell) of the seven is below the minimum value then a proportion of the fetch is reflected back into cell 3. This prevents excessive loss of fetch (i.e. wave energy) occurring laterally at the banks. When α_w is set to 1 the fetch is completely reflected but a lower value can be set if a portion of the wave energy is expected to be dissipated in shallow water.

This procedure allows the fetch to be estimated for any combination of wind direction and bathymetry and gives an approximate representation of diffraction around sand bars and

other obstacles. An offset is applied periodically in this process to account for the difference between the grid and wind directions.

The effective fetch for cells in row $n-1$ is determined from the sum of the contributions from cells in row n plus the additional fetch (the cell length L for a wind direction of zero degrees). The wave direction is assumed to be equal to the wind direction for all cells.

The procedure is repeated to obtain the fetch for cells in row $(n-2)$ and subsequent rows. Wave height and period are then calculated for each cell using Equations (4.21), (4.23) and (4.24) (US Army Corps of Engineers, 2002).

$$\frac{gT_p}{u_*} = 0.751 \sqrt[3]{\frac{gX}{u_*^2}} \tag{4.23}$$

$$u_*^2 = 0.0011 u_{10}^2 + 0.000035 u_{10}^3 \tag{4.24}$$

H_{m0} is the energy based significant wave height, T_p is the wave period, X is the fetch, u_* is the shear velocity and u_{10} is the wind speed at a height of 10m. Equation (4.24) incorporates an empirical drag coefficient and is dimensionally inconsistent; hence SI units must be used.

4.4.3 Wave Field Calculation

Wave height and period are calculated for each cell using Equations (4.21) and (4.23), using the effective fetch and wind speed procedures described above. This procedure was tested using the idealised tidal inlet and barrier beach previously used to test the flow estimation procedure (Section 4.6), for a range of wind speeds and directions. Figure 4.24 shows spatial variation of significant wave height (H_{m0}) in the model domain for an approaching wind direction of 180°, a wind speed of 10 m/s, a fetch of 10 km at the open boundary and $\alpha_w = 1$.

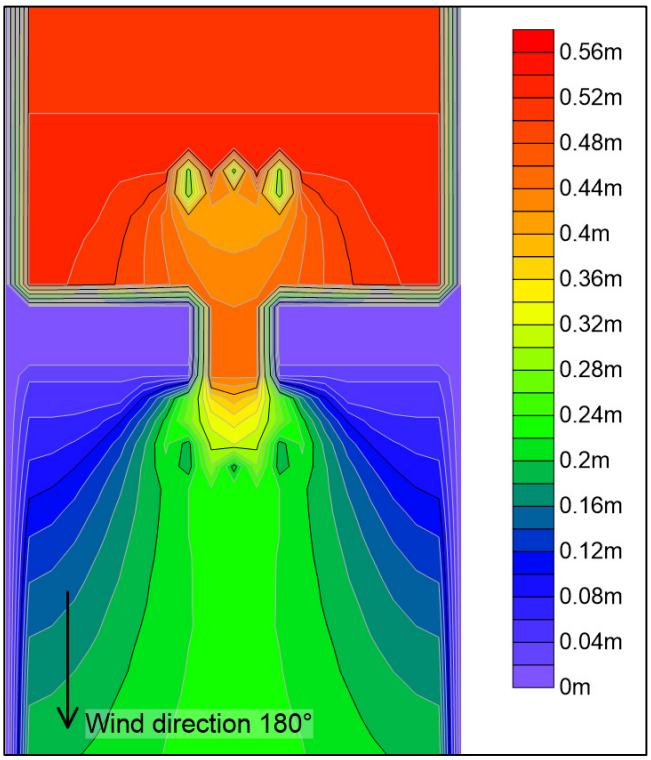

Figure 4.24: Example wave model results for an idealised tidal inlet: wind direction 180°

Figure 4.25 shows the results obtained for an approaching wind direction of 90°. The wind speed and boundary fetch are unchanged. It is noted that while the shallow areas around the flood and ebb deltas have affected the wave heights in Figure 4.24, there is no such effect in Figure 4.25 because the wave heights are already below the depth limited wave height.

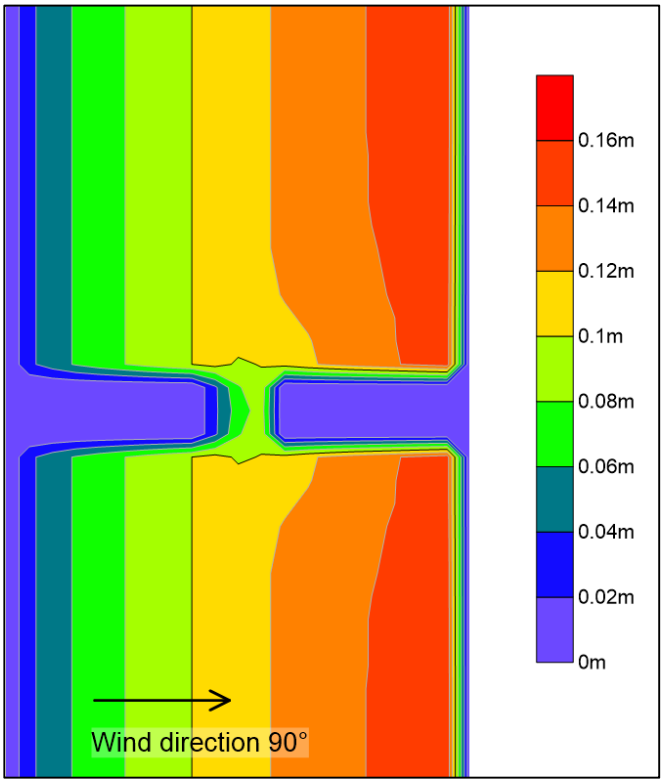

Figure 4.25: Example wave model results for an idealised tidal inlet: wind direction 90°

At a wind direction of 45° the procedure switches from calculating the fetch row by row to calculating column by column. To test the impact of this change the wave field resulting from a wind direction of 44° is compared to that resulting from a wind direction of 45°. The results from this test are shown in Figures 4.26 and 4.27 . In these figures, the wave height for a wind direction of 44° reaches around 0.44m along the right hand edge of the figure, whereas the maximum wave height of around 0.22m obtained for a wind direction of 45° is more localised along the right hand edge of the barrier beach. However, in other respects the pattern of wave heights obtained in each case is quite similar despite the differences in the method used to obtain the fetch.

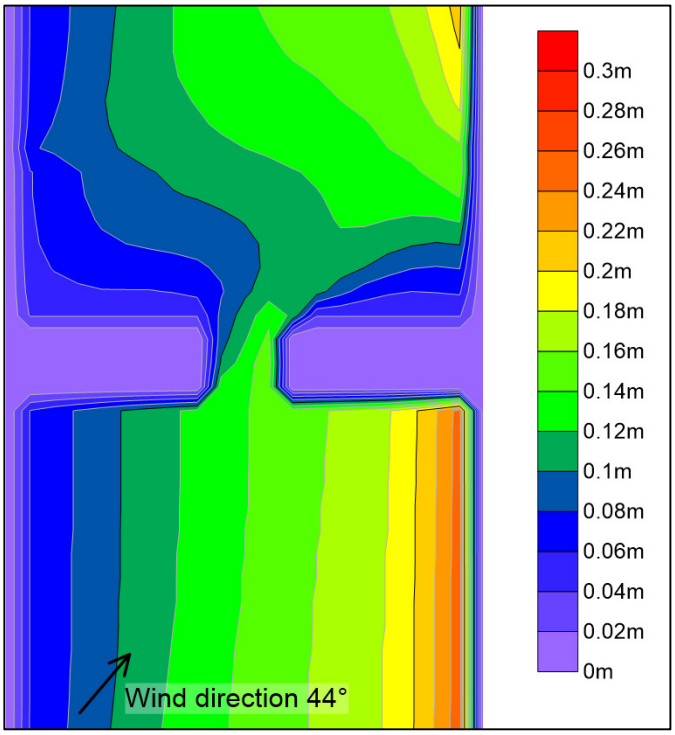

Figure 4.26: Example wave model results for an idealised tidal inlet: wind direction 44°

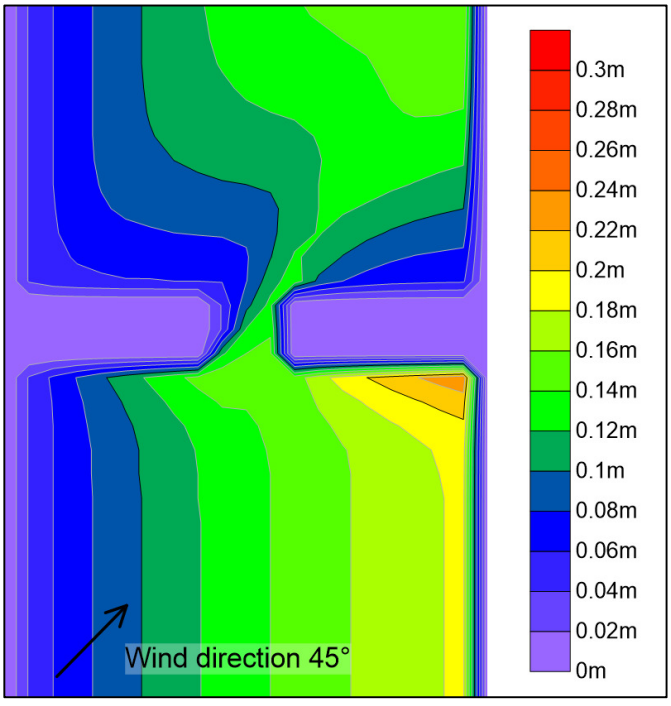

Figure 4.27: Example wave model results for an idealised tidal inlet: wind direction 45°

4.5 Sediment Transport

Sand transport rates in the model domain are calculated using the parametric Van Rijn equations (Van Rijn, 2005) when the model is setup in single-fraction mode and the Van Rijn TRANSPO2014 (TR2004) method (Van Rijn, 2007a, b, c) for the multi-fraction mode. Mud erosion and deposition are calculated according to the equations given by Whitehouse et. al. (2000), while the transport of mud is modelled as a diffusive process, based on a global diffusion coefficient and the tidally averaged rates of erosion and deposition. Both sand transport and mud erosion/deposition are based on the effects of combined waves and currents. These quantities are therefore calculated 25 times, for each of the 25 wave fields (See section 4.4.1) and the results weighted according to the relative probabilities of the wind speed and direction used to generate each set of wave conditions.

The TR2004 method has been developed for multi-fractional transport due to combined waves and currents but has greater computational requirements than the parametric method. Both the bed load and suspended load calculations require numerical integration and, in order to reduce the computational cost associated with this, the method has been re-coded and simplified. Separate transport rates are calculated for all specified fraction sizes, which may range from course sand to fine silt (grain size > 8 μm), while any clay sized particles within the transported material are assumed to be brought into suspension and are treated as mud.

4.5.1 Sand Transport

Bed load and suspended load parametric formulae given by Van Rijn (2005) are used for the single fraction model (Equations 4.25 and 4.26). These formulae are applicable to grain sizes in the range 200μm to 2000μm.

$$q_b = 0.015\left(1 - p_{mud}\right)u_c\, h M_e^{1.5}\left(d_{50}/h\right)^{1.2} \tag{4.25}$$

$$q_s = 0.012\left(1 - p_{mud}\right)u_c\, h M_e^{2.4}\left(d_{50}/h\right)\left(D_*\right)^{-0.6} \tag{4.26}$$

In these equations q_b and q_s are the volumetric bed load and suspended load transport rates per unit width of the bed, p_{mud} is the proportion of mud/clay in the bed (in the range 0 to 0.3), u is the depth averaged velocity, h is the water depth, M_e is a 'mobility parameter' and D_* is a dimensionless particle size, given by:

$$D_* = d_{50}\left[g(s-1)/\nu^2\right]^{1/3} \tag{4.27}$$

$$M_e = \frac{(u_e - u_{cr})}{\sqrt{(s-1)\,g\,d_{50}}} \tag{4.28}$$

Where u_e is an effective velocity and u_{cr} is the critical effective velocity. Both u_e and u_{cr} are adjusted for wave conditions (height and period).

$$u_e = u_c + \gamma U_w \tag{4.29}$$

$$u_{cr} = \alpha_{cr}\,u_{cr,c} + (1-\alpha)\,u_{cr,w} \tag{4.30}$$

$$U_{w,s} = \pi H_s / T_p \sinh(2\pi h / L_w) \tag{4.31}$$

Here, u_c is the depth averaged current velocity, $U_{w,s}$ is the peak wave orbital velocity from linear wave theory (US Army Corps of Engineers, 2002), computed using the significant wave height, $u_{cr,c}$ is the critical effective velocity for currents, given by Equation (4.33), $u_{cr,w}$ is the critical effective velocity for waves given by Equation (4.34), γ = 0.4 to 0.5 for irregular waves and 0.7 to 0.9 for regular waves, and $\alpha_{cr} = u_c / (u_c + U_w)$. In this research the wavelength (L_w) is computed using the approximate formula (Equation 4.32), given in the Coastal Engineering Manual (US Army Corps of Engineers, 2002).

$$L_w \approx \frac{gT_p^{\,2}}{2\pi}\sqrt{\tanh\left(\frac{4\pi^2 h}{T_p^{\,2} g}\right)} \tag{4.32}$$

$$u_{cr,c} = \begin{cases} 0.19(1+p_{mud})^{1.5}\,(d_{50})^{0.1}\,\log(12h/3d_{90}) & for\ 0.0001 < d_{50} < 0.0005m \\ 8.50(1+p_{mud})^{1.5}\,(d_{50})^{0.6}\,\log(12h/3d_{90}) & for\ 0.0005 < d_{50} < 0.0020m \end{cases} \tag{4.33}$$

$$u_{cr,w} = \begin{cases} 0.24(1+p_{mud})^{1.5}\,((s-1)g)^{0.66}(d_{50})^{0.33}(T_p)^{0.33} & for\ 0.0001 < d_{50} < 0.0005m \\ 0.95(1+p_{mud})^{1.5}\,((s-1)g)^{0.57}(d_{50})^{0.43}(T_p)^{0.14} & for\ 0.0005 < d_{50} < 0.0020m \end{cases} \quad (4.34)$$

4.5.2 Mud Transport

Mud transport is calculated using the equations for erosion, deposition and transport given by Whitehouse et. al. (2000). Erosion occurs when the maximum bed shear stress (τ_{max}) exceeds the critical bed shear stress for erosion (τ_e) and is estimated according to:

$$\frac{dm}{dt} = m_e(\tau_{\max} - \tau_e) \quad (4.35)$$

Where m is the dry mass of mud eroded per unit area, m_e is the erosion constant, which is a model parameter in the range 0.0002 to 0.003 $kgN^{-1}s^{-1}$, and τ_{max} is the maximum bed shear stress during a wave cycle. The critical bed shear stress for the erosion of mud (τ_e) is dependent on the bulk density of the bed (Whitehouse et. al., 2000); however, in the present model τ_e is a constant model parameter (typical values range from around 0.1 N/m^2 to around 0.2 N/m^2). Consolidation effects are not included in the present model.

The maximum bed shear stress is calculated based on an equation for the mean bed shear stress during a wave cycle (τ_m) that was developed for smooth beds (Whitehouse et. al., 2000):

$$\tau_{\max} = \left[(\tau_m + \tau_w \cos\beta_{cw})^2 + (\tau_w \sin\beta_{cw})^2\right]^{0.5} \quad (4.36)$$

$$\tau_m = \tau_c\left[1 + 9\left(\frac{\tau_w}{\tau_c + \tau_w}\right)^9\right] \quad (4.37)$$

where τ_c and τ_w are the bed shear stresses due to currents and waves respectively. The bed shear stress due to currents, for smooth beds, is determined from the shear velocity using $\tau_c = \rho_w u_*^2$, where ρ_w is the density of water and u_* is the shear velocity. The shear velocity is determined from the iterative solution of Equation (4.38):

$$\frac{u(z_{bd})}{u_*} = 5.5 + 2.5\ln\left(\frac{u_* z}{\upsilon}\right) \tag{4.38}$$

where $u(z_{bd})$ is the current velocity at height z_{bd} above the bed, which is taken as equal to the depth averaged current velocity at $z_{bd} = 0.37h$. The bed shear stress due to waves is given by:

$$\tau_w = 0.5\rho_w f_w U_w^2 \tag{4.39}$$

Here f_w is a wave friction factor and U_w is the peak wave near-bed orbital velocity according to linear wave theory, calculated using the equivalent monochromatic wave height H given by $H = H_{rms} = H_s\sqrt{2}$ (Whitehouse et. al., 2000).

$$U_w = \pi H / T_p \sinh(2\pi h / L_w) \tag{4.40}$$

Here T_p is the peak spectral wave period (Equation 4.23) and L_w is the wavelength (Equation 4.32). The wave friction factor for smooth beds is calculated from:

$$f_w = BR_w^{-N} \tag{4.41}$$

where the wave Reynolds number, $R_w = U_w A_w / \upsilon$ and the coefficients B and N take the values 2 and 0.5 respectively for laminar flow ($R_w < 5 \times 10^{-5}$) or 0.5 and 0.187 for smooth turbulent flow ($R_w > 5 \times 10^{-5}$). A_w is the wave semi-orbital excursion according to linear wave theory, given by Equation (4.42) and ν is the kinematic viscosity of water.

$$A_w = U_w T_p / 2\pi \tag{4.42}$$

Deposition can be calculated using $dm/dt = -(1 - \tau_b/\tau_d)\cdot c_b w_{50}$ (Whitehouse et. al., 2000), where τ_b is the shear stress at the bed, c_b is the near bed sediment concentration and w_{50} is the mean fall velocity. For $\tau_b > \tau_d$ the deposition rate is set to zero. This formula has been simplified in this research to:

$$\frac{dm}{dt} = -\left(1 - \frac{\tau_{\max}}{\tau_d}\right) \cdot c\, w_s \tag{4.43}$$

where τ_{max} is as defined above for erosion, the sediment concentration c is assumed to be uniform over the mean water depth in the cell and w_s is a vertical exchange coefficient, which is a model parameter that is related to, but not necessarily equal to, the particle fall velocity. The fall velocity (and hence w_s) is related to the sediment concentration, due to the effects of flocculation and hindered settling (Whitehouse et. al., 2000); however, a constant value is used in the present model for simplicity and a value of 5.0 x 10^{-4} m/s has been used for the simulations carried out so far (this is similar to values used for the Humber and Southampton Water in the ESTMORPH and ASMITA models: Wang, et. al., 2007; Rossington and Nicholls, 2008). The critical bed shear stress for deposition (τ_d) is a model parameter that should be typically around half the value taken for erosion (τ_e) (Whitehouse et. al., 2000).

The transport of mud within the water column occurs through the process of tidally averaged diffusion. The volume of sediment suspended within the water column is assumed to be small relative to the volume of sediment flux over the tidal period and hence sediment concentration can be estimated by balancing the diffusive flux of sediment into each cell with the net deposition, according to Equation (4.44):

$$\sum_{i=1}^{4} q_{s,i} = D_{mud} - E_{mud} \tag{4.44}$$

where $q_{s,i}$ (i = 1 to 4) is the volumetric sediment flux into a cell from each of its four orthogonal neighbours as shown in Figure 4.28, D_{mud} is the tidally averaged volumetric deposition rate and E_{mud} is the tidally averaged volumetric rate of erosion.

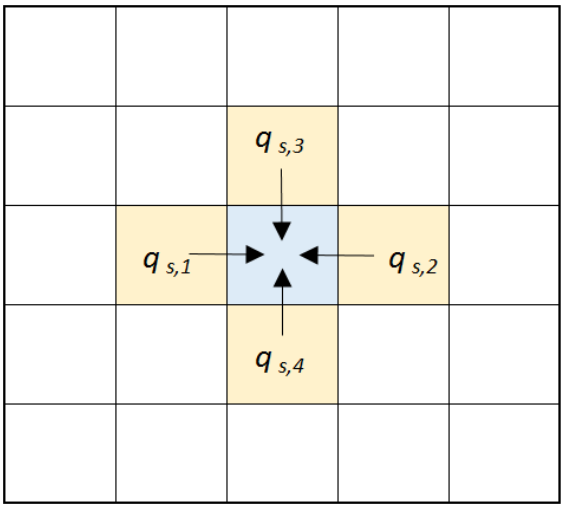

Figure 4.28: Suspended sediment flux into a cell

The sediment flux between neighbouring cells is given by Equation (4.45):

$$q_{s,i} = k_{d,i}\left(c_i - c_0\right) \tag{4.45}$$

In this equation $k_{d,i}$ is the diffusion coefficient for transport into the cell from neighbouring cell i, c_0 is the sediment concentration in the cell and c_i is the concentration in cell i. The coefficient $k_{d,i}$ is given by Equation (4.46):

$$k_{d,i} = K_d \min\left(h_i, h_0\right) \tag{4.46}$$

where K_d is a global diffusion coefficient and h_0 and h_i are the mean depth in the cell and neighbouring cell i. The value of K_d has been set with reference to values previously used for the ASMITA model (Rossington and Nicholls, 2008) and the ESTMORPH model (Wang et. al., 2007). These values range between 200 m^2/s and 2000 m^2/s and a value of 500 m^2/s has been selected for the simulations carried out so far in this research. The sediment concentration in each cell is recalculated at the end of each tidal cycle (or spring-neap cycle) according to Equation (4.47):

$$c_0 = \frac{L\sum k_{d,i}\, c_i + E_{mud}}{L\sum k_{d,i} + D'_{mud}} \tag{4.47}$$

where L is the cell size and D'_{mud} is the volumetric deposition rate divided by the previous value of c_0 (or the deposition rate that would have been calculated for a sediment concentration of 1.0). Since the results depend on the concentrations in neighbouring cells, which may have themselves changed, the sediment concentration field is solved iteratively with the calculation repeated for all cells several times until a specified convergence criteria is satisfied.

4.5.3 Combined Sand and Mud

If more than about 10% by dry weight of a sediment mixture consists of fine material then it will tend to behave as mud, with a transition occurring between 5% and 15% by dry weight (Whitehouse et. al., 2000). Based on the guidance given by Whitehouse et. al. (2000) a maximum critical bed shear stress for erosion ($\tau_{e,max}$) is assumed to occur for mixtures containing 20% fine material, with $\tau_{e,\max} = 2 \cdot \tau_{cr,sand}$ (Whitehouse et. al., 2000 suggest a multiple in the range 3 to 5 but a value of 2 is used here for consistency with results obtained using the Van Rijn TR2004 model discussed in section 4.6). The critical shear stress for erosion (τ_e) is assumed to vary linearly, between $\tau_{cr,sand}$ and $\tau_{e,max}$, for mixtures containing 0 to 20% fine material and between $\tau_{e,max}$ and $\tau_{e,\mathrm{mud}}$ for mixtures containing 20 to 100% fine material, as illustrated in Figure 4.29.

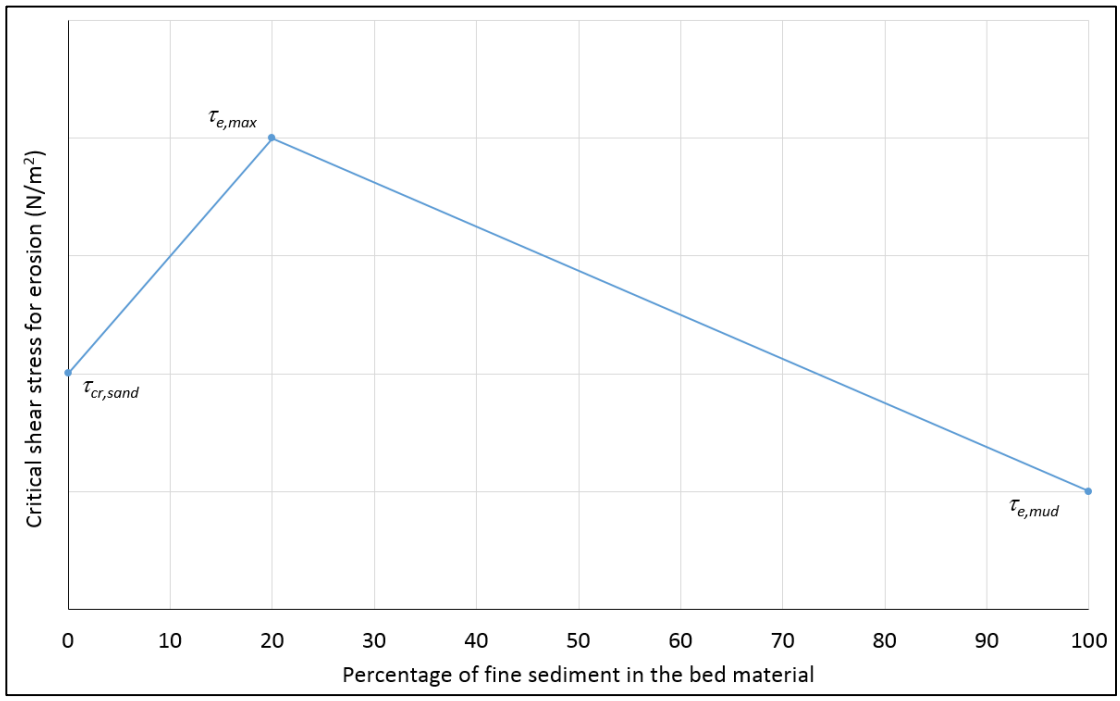

Figure 4.29: Critical shear stress for mixtures of sand and mud.

For mixtures containing less than 5% mud, sediment transport is calculated using the sand transport formulae (Equations 4.25 and 4.26). Mud contained within the transported sand is assumed to be entrained into the water column. Volumetric sand transport ($q_{t,sand}$) and volumetric mud erosion (E_{mud}) are therefore given by:

$$q_{t,sand} = (1 - p_{mud})(q_s + q_b) \tag{4.48}$$

$$E_{mud} = p_{mud}(q_s + q_b) \tag{4.49}$$

For mixtures containing more than 15% mud, the sand component of the mixture is assumed to be immobile and the rate of mud erosion (E_{mud}) is calculated using:

$$E_{mud} = \frac{p_{mud}\, m_e}{\rho_b}(\tau_{\max} - \tau_e) \tag{4.50}$$

Where ρ_b is the sediment bulk density, τ_{max} is the maximum bed shear stress during a wave cycle and τ_e is given by linear interpolation between $\tau_{cr,sand}$, $\tau_{e,max}$ and $\tau_{e,mud}$, using Equation (4.51) for $0.15 \le p_{mud} < 0.2$ or Equation (4.52) for $0.2 \le p_{mud} \le 1.0$.

$$\tau_e = \tau_{e,\max} - \left[\left(\tau_{e,\max} - \tau_{cr,sand} \right) \left(\frac{0.2 - p_{mud}}{0.2} \right) \right] \tag{4.51}$$

$$\tau_e = \tau_{e,\max} - \left[\left(\tau_{e,\max} - \tau_{e,mud} \right) \left(\frac{p_{mud} - 0.2}{0.8} \right) \right] \tag{4.52}$$

For mixtures containing between 5 and 15% mud, the volumetric sand transport is determined by interpolation between zero at 15% mud and the full value given by the sand transport formulae at 5% mud, using Equation 4.53. The mud erosion rate is determined assuming that the mud contained within the transported sand is entrained and using the mud erosion formula, which is interpolated between zero at 5% mud and the full value at 15% mud (Equation 4.54).

$$q_{t,sand} = (1 - p_{mud})(q_s + q_b) \left(\frac{0.15 - p_{mud}}{0.1} \right) \tag{4.53}$$

$$E_{mud} = \left[p_{mud} (q_s + q_b) \left(\frac{0.15 - p_{mud}}{0.1} \right) \right] + \left[\frac{p_{mud} m_e}{\rho_b} (\tau_{\max} - \tau_e) \left(\frac{p_{mud} - 0.05}{0.1} \right) \right] \tag{4.54}$$

The transport and deposition of sand and mud is calculated as described above in Sections 4.5.1 and 4.5.2.

4.5.4 Multi-Fraction Sediment Transport

An option for multi-fraction sediment transport modelling has been developed based on the Van Rijn TR2004 method (Van Rijn, 2007a, b, c), with a number of simplifications to reduce the computational requirements. The method allows a number of sand and silt fractions to be defined, each with a representative grain size (d_i) ranging in size from fine silt (8μm) to course sand (2mm), with separate transport rates calculated for each fraction. The clay fraction (i.e. sediment finer than 8μm) is treated as mud using the methods described in Section 4.5.3. Transport rates for fine silt (8μm < d_i < 32μm) are calculated using the TR2004 method but these fractions are otherwise treated as mud, with the calculated transport rate being used to determine the erosion rate.

If the proportion of clay in the bed material exceeds 30% then mud erosion is calculated using the methods described in Section 4.5.3 and applied to the clay fraction and fine silt fractions. In this case all other fractions are considered to be immobile. In this research a transition is applied for clay proportions between 20% and 30%, with the TR2004 sediment transport applied fully at 20% clay and reduced linearly to zero at 30% clay.

4.5.4.1 Hydraulic Roughness of the Bed

The hydraulic roughness is strongly affected by the presence of bed forms and, in the TR2004 method, bed form roughness can be estimated based on sediment properties, current velocity and wave conditions (Van Rijn, 2007a). Since this method does not require any numerical iteration it is computationally efficient and has therefore been incorporated into the simplified approach used in this project. Roughness values relating to waves ($k_{s,w}$) are defined separately from those relating to currents ($k_{s,c}$).

The heights of ripples, mega-ripples and dunes are estimated based on current velocity, wave conditions and median grain size. A current-wave mobility parameter (ψ) is first defined:

$$\psi = \frac{U_{wc}^2}{(s-1)g\,d_{50}} \tag{4.55}$$

where U_{wc} is a velocity parameter for combined waves and currents given by $(U_{wc})^2 = (U_w)^2 + (u_c)^2$, where U_w is the peak wave near-bed orbital velocity (Equation 4.31) and u_c is the depth averaged current velocity. The bed roughness due to ripples is then given by Equation (4.56) for all values of ψ and $d_{50} > d_{silt}$. For $d_{50} < d_{silt}$, $k_{s,c,r} = 20\,d_{silt}$ for all values of ψ ($d_{silt} = 32\mu m$).

$$k_{s,c,r} = f_{cs}\,d_{50}\left(85 - 65\tanh\left[0.015(\psi - 150)\right]\right) \tag{4.56}$$

The parameter f_{cs} is given by $f_{cs} = (0.25\,d_{gravel} / d_{50})^{0.25}$, with $f_{cs} = 1$ for $d_{50} \le 0.25\,d_{gravel}$ ($d_{gravel} = 2mm$). The bed roughness due to mega-ripples is similarly given by Equation (4.57) for $\psi < 550$ ($k_{s,c,mr} = 0$ for $\psi \ge 550$).

$$k_{s,c,mr} = 0.00002 f_{fs} h[1 - \exp(-0.05\psi)](550 - \psi) \tag{4.57}$$

Here, the parameter f_{fs} is given by $f_{fs} = d_{50} / 1.5 d_{sand}$, with $f_{fs} = 1$ for $d_{50} \geq 1.5 d_{sand}$ (d_{sand} = 62μm). Finally the bed roughness due to dunes is given by Equation (4.58) for $\psi < 600$ ($k_{s,c,d} = 0$ for $\psi \geq 600$).

$$k_{s,c,d} = 0.00008 f_{fs} h[1 - \exp(-0.02\psi)](600 - \psi) \tag{4.58}$$

Van Rijn (2007a) suggests that the overall bed roughness in relation to currents is given by the quadric summation of the values due to ripples, mega-ripples and dunes (Equation 4.59) and the overall roughness in relation to waves is given by the value due to ripples alone (Equation 4.60).

$$k_{s,c} = \left[(k_{s,c,r})^2 + (k_{s,c,mr})^2 + (k_{s,c,d})^2 \right]^{0.5} \tag{4.59}$$

$$k_{s,w} = k_{s,c,r} \tag{4.60}$$

The apparent bed roughness (k_a) is used to calculate the near bed current velocity:

$$k_a = k_{s,c} \exp\left(\frac{\gamma U_w}{\sqrt{u_c^2 + u_r^2}} \right) \tag{4.61}$$

where u_r is the velocity of the return mass transport under the wave, given by Equation (4.62). The parameter γ is given by $\gamma = 0.8 + \beta_{cw} - 0.3\beta_{cw}^{\ 2}$, where β_{cw} is the angle between the wave and current directions in radians (measured anti-clockwise from the current direction).

$$u_r = \frac{0.125\, g^{0.5} (H_s)^2}{h^{0.5}\, h_t} \tag{4.62}$$

$$h_t = h\left[0.95 - 0.35\left(\frac{H_s}{h} \right) \right] \tag{4.63}$$

Finally, the above roughness heights are used to compute a number of corresponding friction factors, as follows.

A current-related friction coefficient:

$$f_c = 0.24 \log^{-2}\left(12h/k_{s,c}\right) \tag{4.64}$$

A current-related grain friction coefficient:

$$f_c' = 0.24 \log^{-2}\left(12h/d_{90}\right) \tag{4.65}$$

A wave-related friction coefficient:

$$f_w = \exp\left(-6 + 5.2\left(A_w/k_{s,w}\right)^{-0.19}\right) \tag{4.66}$$

A wave-related grain friction coefficient:

$$f_w' = \exp\left(-6 + 5.2\left(A_w/d_{90}\right)^{-0.19}\right) \tag{4.67}$$

An apparent friction coefficient:

$$f_a = 0.24 \log^{-2}\left(12h/k_a\right) \tag{4.68}$$

Maximum values for the wave-related friction coefficients are given as: $f_{w,\max} = 0.3$ and $f'_{w,\max} = 0.05$.

4.5.4.2 The bed shear stress parameter

A bed shear stress parameter is calculated for each sediment fraction, as defined in the TR2004 method (Van Rijn, 2007c), according to:

$$T_i = \lambda_i \left[\frac{\tau_b' - \xi_i \left(d_i/d_{50}\right)\tau_{b,cr,d_{50}}}{\left(d_i/d_{50}\right)\tau_{b,cr,d_{50}}} \right] \tag{4.69}$$

where T_i is the bed shear stress parameter for fraction i, λ_i is a 'correction factor of effective grain shear stress', given by $\lambda_i = (d_i / d_{50})^{0.25}$, ξ_i is the hiding-exposure factor for fraction i (Equation 4.70), δ_i is the particle size for fraction i, d_{50} is the median particle size in the bed and $\tau_{b,cr,d_{50}}$ is the critical bed shear stress based on the median particle size. The parameter τ_b' is either the instantaneous bed shear stress due to currents and waves, for bed load calculations $(\tau_{b,cw}')$, or the time-averaged bed shear stress due to currents and waves $(\tau_{b,cw}')$, for suspended sediment load calculations.

Egiazaroff (cited by Van Rijn, 2007c) has derived a theoretical hiding exposure function to reduce the bed shear stress for smaller particles, to allow for the effect of sheltering by larger particles and to increase the bed shear stress for larger particles due to their increased exposure over that felt by the same particle in a uniform bed.

$$\xi_i = \left(\frac{\log(19)}{\log\left(19 \frac{d_i}{d_{50}}\right)} \right)^2 \tag{4.70}$$

This function tends towards infinity at $d_i = d_{50}/19$ and Kleinhans and Van Rijn (2002) have suggested that it should be modified for particles smaller than around $d_{50}/12$. Kleinhans and Van Rijn (2002) also discuss empirical hiding-exposure functions in the form $\xi_i = (d_i / d_{50})^P$, where the exponent P is in the range −1 to 0. Setting P = -1 produces equal mobility for all particles, while setting P = 0 results in uncorrected bed shear stress values. A compound function has been adopted for this project, using Egiazaroff values for $d_i > 0.19 d_{50}$ and equal mobility values for $d_i \leq 0.19 d_{50}$, as shown in Figure 4.30.

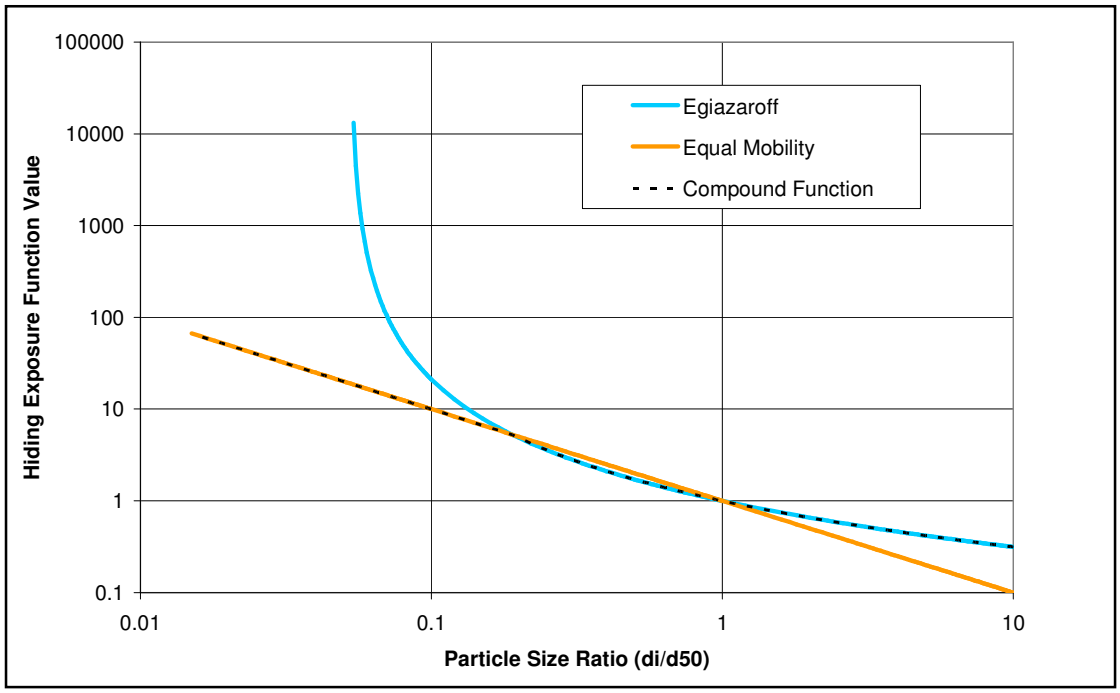

Figure 4.30: Sediment hiding exposure functions

4.5.4.3 Multi-Fraction Bed load Transport

The instantaneous volumetric bed load transport for each sediment fraction is given by:

$$q_{b,i} = 0.5\, f_{silt,i}\, d_i \left(D_{*,i}\right)^{-0.3} \left(\tau''_{b,cw} / \rho_w\right)^{0.5} \left(T_i\right) \tag{4.71}$$

where $q_{b,i}$ is the instantaneous volumetric bed load transport rate for fraction i, $f_{silt,i}$ is a 'silt factor' ($f_{silt,i} = d_{sand} / d_i$, with $d_{sand} = 62\mu m$ and $f_{silt,i} = 1$ for $d_i > d_{sand}$), d_i is the particle size, $D_{*,i}$ is the dimensionless particle size parameter for fraction i (Equation 4.27), $\tau''_{b,cw}$ is the instantaneous grain-related bed shear stress due to currents and waves, given by Equation (4.72), ρ_w is the density of water and T_i is a bed shear stress parameter for fraction i (Equation 4.69).

$$\tau''_{b,cw} = 0.5 \rho_w\, f_{cw} \left(U_{\delta,cw}\right)^2 \tag{4.72}$$

In this equation f_{cw} is the grain friction coefficient due to currents and waves, given by Equation (4.73), and $U_{\delta,cw}$ is the instantaneous velocity due to currents and waves at the edge of the wave boundary layer (Equation 4.79).

$$f'_{cw} = \alpha\beta f'_c + (1-\alpha) f'_w \tag{4.73}$$

where α is a coefficient related to the relative strength of the current and wave motion, β is a coefficient related to the vertical structure of the velocity profile, f'_c is the grain friction coefficient for currents and f'_w is the grain friction coefficient for waves (Section 4.5.4.1).

$$\alpha = u_c / (u_c + U_w) \tag{4.74}$$

$$\beta = 0.25\left[\frac{-1+\ln(30h/k_{s,c})}{\ln(30\delta/k_{s,c})}\right]^2 \tag{4.75}$$

Here, $k_{s,grain}$ is the grain roughness height ($k_{s,grain} = 1 d_{90}$), $k_{s,c}$ is the current related bed roughness (Equation 4.59). The parameter δ is the height above the bed at which $U_{\delta,cw}$ is calculated:

$$\delta = \max(3\delta_w, k_{s,c}) \tag{4.76}$$

where δ_w is the wave boundary layer thickness, given by:

$$\delta_w = 0.072 A_w (A_w / k_{s,w})^{0.25} \tag{4.77}$$

Where $k_{s,w}$ is the wave related bed roughness (Equation 4.60) and A_w is the peak wave near bed orbital displacement (Equation 4.42).

The instantaneous bed load transport rate must be integrated over a wave cycle in order to calculate time-averaged bed load transport rate. A simplified approach to this has been adopted in this research, in order to minimise the computational requirements, in which the instantaneous near bed orbital velocity due to the wave (v_w) is assumed to be a sinusoidal function of time, varying between $+U_w$ and $-U_w$. The wave is divided into eight equal time periods: two periods each for near bed orbital velocities of zero, $0.707 U_w$, and $-0.707 U_w$ and one period each for the velocities U_w and $-U_w$, as illustrated in Figure 4.31.

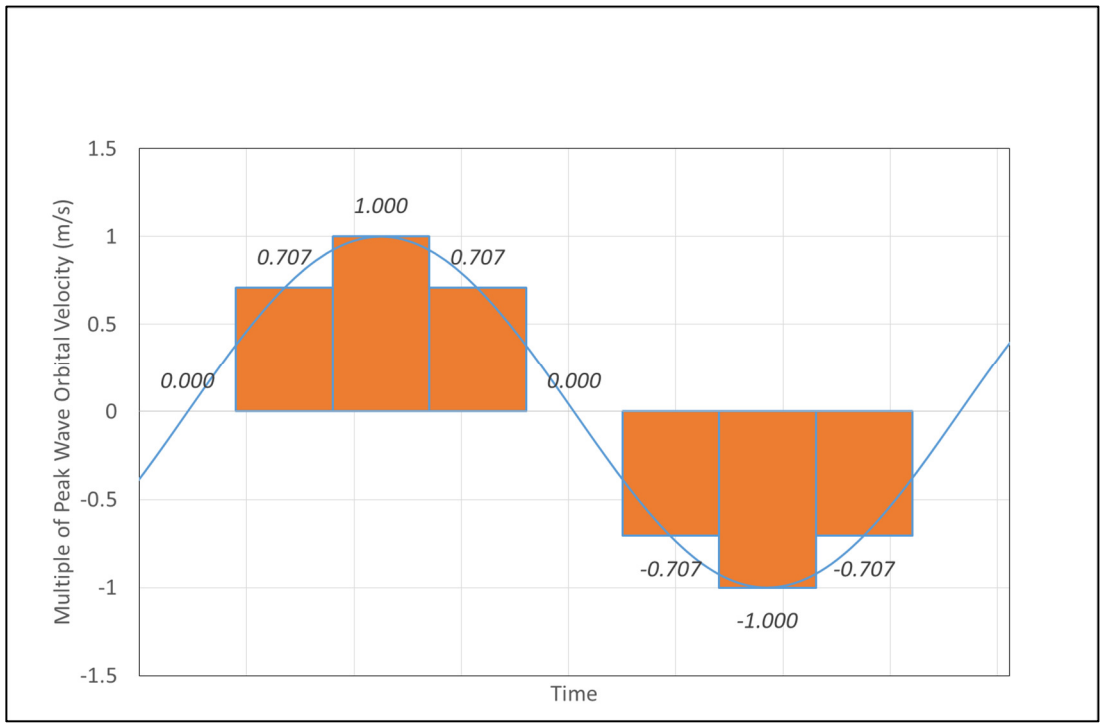

Figure 4.31: Division of wave cycle for bed load calculation

The significant wave height (H_s) and peak spectral wave period (T_p) are used as the representative wave characteristics (Van Rijn, 1993). The wave length (L_w) is calculated using Equation (4.32).

Near bed velocities due to the waves must be combined with the near bed current velocity ($v_{R,\delta}$) to determine $U_{\delta,cw}$ for each of the eight time periods.

$$v_{R,\delta} = \frac{u_c \ln\left(30\delta_w / k_a\right)}{-1 + \ln\left(30h/k_a\right)} \tag{4.78}$$

The total near bed velocity magnitude for each time period is then obtained from vector addition of the wave and current components:

$$U_{\delta,cw} = \sqrt{\left(v_w \sin\theta_w + v_{R,\delta}\sin\theta_c\right)^2 + \left(v_w \cos\theta_w + v_{R,\delta}\cos\theta_c\right)^2} \tag{4.79}$$

$$\theta_{cw} = \tan^{-1}\left(\frac{v_w \sin\theta_w + v_{R,\delta}\sin\theta_c}{v_w \cos\theta_w + v_{R,\delta}\cos\theta_c}\right) \tag{4.80}$$

where v_w is the instantaneous wave orbital velocity (0, 0.707 U_w, U_w etc.), θ_w is the wave direction and θ_c is the current direction (measured clockwise from the grid axis pointing towards the downstream boundary). The resulting sediment transport direction (θ_s) is given by $\theta_s = \theta_{cw}$ for $v_w \cos\theta_w + v_{R,\delta} \cos\theta_c \geq 0$ and $\theta_s = \theta_{cw} + \pi$ (radians) for $v_w \cos\theta_w + v_{R,\delta} \cos\theta_c < 0$. The time-averaged bed load along the grid axes is then given by:

$$q_{b,x} = \frac{\sum_{j=1}^{8} q_{b,j} \sin \theta_{s,j}}{8} \tag{4.81}$$

$$q_{b,y} = \frac{\sum_{j=1}^{8} q_{b,j} \cos \theta_{s,j}}{8} \tag{4.82}$$

Where $q_{b,j}$ and $q_{x,j}$ are the respective values of q_b (Equation 4.71) and θ_s for each of the eight instantaneous bed load calculations.

4.5.4.4 Multi-Fraction Suspended Sediment Transport

The suspended sediment transport rate is calculated in the TR2004 method by integrating the suspended sediment concentration together with the velocity profile, over the depth (Van Rijn, 2007b,c), in accordance with Equation (4.83):

$$q_s = \int_a^h c_i \, u \, dz \tag{4.83}$$

where c and u are the sediment concentration and current velocity at height z above the bed, and a is the reference height, which is given by $a = 0.5\ k_{s,c,r}$, with a minimum value of 0.01. A reference concentration (c_a) is calculated for each fraction at $z = a$:

$$c_a = 0.015 f_{silt,i} (d_i / a) (D_{*,i})^{-0.3} (T_i)^{1.5} \tag{4.84}$$

where $f_{silt,i}$, d_i and $D_{*,i}$ are defined as above for bed load. The transport stage parameter (T_i) is given by Equation (4.69) with $\tau_b' = \tau_{b,cw}'$, given by:

$$\tau'_{b,cw} = \tau'_{b,c} + \tau'_{b,w} \tag{4.85}$$

$$\tau'_{b,c} = \mu_c \alpha_{cw} \tau_{b,c} \tag{4.86}$$

$$\tau'_{b,w} = \mu_w \tau_{b,w} \tag{4.87}$$

where μ_c and μ_w are the current related efficiency factor (related to the grain size d_{90} and the bed roughness) and the wave related efficiency factor (related to the dimensionless particle size D_*). The 'wave-current interaction factor' (α_{cw}) is based on the bed roughness and the wave conditions.

$$\mu_c = f_c'/f_c \tag{4.88}$$

$$\mu_w = 0.7 D_{*,i} \tag{4.89}$$

With $\mu_{w,min} = 0.14$ for $D_* \geq 5$ and $\mu_{w,max} = 0.35$ for $D_* = 2$.

$$\alpha_{cw} = \left(\frac{\ln(90\delta_w/k_a)}{\ln(90\delta_w/k_{s,c})} \right)^2 \left(\frac{-1+\ln(30h/k_{s,c})}{-1+\ln(30h/k_a)} \right)^2 \tag{4.90}$$

The bed shear stresses due to currents ($\tau_{b,c}$) and waves ($\tau_{b,w}$) are given in Equations (4.91) and (4.92) (Van Rijn, 2006).

$$\tau_{b,c} = \frac{1}{8} \rho \, f_c (u_c)^2 \tag{4.91}$$

$$\tau_{b,w} = \frac{1}{4} \rho \, f_w (U_w)^2 \tag{4.92}$$

The parameters f_c', f_c, f_w', f_w, k_a and $k_{s,c}$ are as defined in Section 4.5.4.1. The formula for the concentration profile includes the effects of particle flocculation and hindered settling in fluid-sediment mixtures.

$$\frac{dc_i}{dz} = -\frac{w_{s,m}\, c_i}{\phi_{d,i}\, \varepsilon_{s,cw,i}} \tag{4.93}$$

$$w_{s,m} = \phi_{floc}\, \phi_{hs}\, w_{s,0} \tag{4.94}$$

Here, $\phi_{floc,i}$ is a flocculation factor, ϕ_{hs} is a hindered settling factor, $w_{s,0}$ is the particle fall velocity in clear water, based on the suspended sediment particle size and $\varepsilon_{s,cw}$ is a sediment mixing coefficient for combined waves and currents. $\varepsilon_{s,cw}$ and ϕ_{floc} are given by:

$$\varepsilon_{s,cw} = \left[(\varepsilon_{s,c})^2 + (\varepsilon_{s,w})^2 \right]^{0.5} \tag{4.95}$$

$$\phi_{floc} = \left((\phi_{floc,0} - 1)(Sa / Sa_{\max}) \right) + 1 \tag{4.96}$$

where Sa is the salinity and Sa_{max} = 1 promille.

$$\phi_{floc,0} = \left(4 + \log(2c / c_{gel})\right)^{E_{floc}} \tag{4.97}$$

$$E_{floc} = (d_{sand} / d_{50}) - 1 \tag{4.98}$$

$$\phi_{hs} = (1 - 0.65c / c_{gel})^5 \tag{4.99}$$

The parameters $\varepsilon_{s,c}$ and $\varepsilon_{s,w}$ are the sediment mixing coefficients for currents and waves, c is the total sediment concentration and c_{gel} is the gelling concentration (taken as 0.65).

$$\varepsilon_{s,c} = \phi_d \beta_c \varepsilon_{f,c} \tag{4.100}$$

$$\varepsilon_{f,c} = 4\frac{z}{h}\left(1-\frac{z}{h}\right)\varepsilon_{f,c,\max} \tag{4.101}$$

$$\varepsilon_{f,c,\max} = 0.25\kappa u_{*,c} h \tag{4.102}$$

If $z > 0.5h$, $\varepsilon_{f,c} = \varepsilon_{f,c,max}$.

$$\varepsilon_{s,w} = \varepsilon_{s,w,bed} + \left(\varepsilon_{s,w,\max} - \varepsilon_{s,w,bed}\right)\left(\frac{z-\delta_s}{0.5h-\delta_s}\right) \tag{4.103}$$

$$\varepsilon_{s,w,\max} = 0.035\gamma_{br} h H_s / T_p \tag{4.104}$$

$$\varepsilon_{s,w,bed} = 0.018\gamma_{br}\beta_w \delta_s U_w \tag{4.105}$$

$$\delta_s = 0.3h\left(H_s/h\right)^{0.5} \tag{4.106}$$

If $z > 0.5h$, $\varepsilon_{s,w,} = \varepsilon_{s,w,max}$ and if $z < \delta_s$, $\varepsilon_{s,w} = \varepsilon_{s,w,bed}$. The parameter ϕ_d is an empirical factor to account for the damping effect of sediment particles on flow turbulence, β_c and β_w are factors to account for the effect of the sediment particles on the mixing of fluid momentum, δ_s is the thickness of the near-bed sediment mixing layer and γ_{br} is an empirical coefficient related to wave breaking.

$$\phi_d = \phi_{fs}\left(1 + \left(c/c_{gel}\right)^{0.8} - 2\left(c/c_{gel}\right)^{0.4}\right) \tag{4.107}$$

$$\phi_{fs} = d_{50}/1.5d_{sand} \tag{4.108}$$

If $d_{50} > 1.5\,d_{sand}$, $\phi_{fs} = 1.5\,d_{sand}$.

$$\beta_c = 1 + 2\left(w_s/u_{*,c}\right)^2 \tag{4.109}$$

$$\beta_w = 1 + 2\left(w_s/u_{*,w}\right)^2 \tag{4.110}$$

$$\gamma_{br} = 1 + \left(\left(H_s/h\right) - 0.4\right)^{0.5} \tag{4.111}$$

If $H_s/h < 0.4$, $\gamma_{br} = 1$. Outside the wave boundary layer ($z \geq 3\delta_w$), the velocity profile is given by:

$$v_{R,z} = \frac{u_c \ln(30z/k_a)}{-1 + \ln(30h/k_a)} \tag{4.112}$$

Inside the wave boundary layer ($z < 3\delta_w$), the velocity profile is given by:

$$v_{R,z} = \frac{v_\delta \ln(30z/k_a)}{\ln(90\delta_w/k_{s,c})} \tag{4.113}$$

with:

$$v_\delta = \frac{u_c \ln(90\delta_w/k_a)}{-1 + \ln(30h/k_a)} \tag{4.114}$$

In this research the integration calculation has been simplified to minimise the number of computation steps required. Beginning at $z = a$, the increments δ_z are set to give $\delta_c = c / 2$, with $\delta_{z,max} = 0.1$m or ($h - z_{bd}$), whichever is lower, and $\delta_c = c$ for $c < 1 \times 10^{-6}$. A predictor-corrector method is used to improve the accuracy of the integration.

4.5.5 Comparison of Sediment Transport Methods

The TR2004 point model is available as a Fortran computer program and is applicable to a wide range of silt and sand sized particles (8 to 1000μm). It has been verified using field data for particle sizes above 60μm (Van Rijn, 2007c). Table 4.1 gives results from a number of test calculations, which have been carried out to compare results from the TR2004 model to the parametric equations given in Section 4.5.1.

D_{50} (μm)	Velocity (m/s)	H_s (m) (T_p = 10s)	Volumetric Transport (m^2/s)		
			TR2004 (Full)	Parametric	Ratio
1000	0.5	0.0	2.71×10^{-7}	0	0%
1000	0.5	0.5	2.56×10^{-5}	1.84×10^{-5}	72%
1000	1.0	0.0	2.87×10^{-5}	7.11×10^{-5}	248%
1000	1.0	0.5	1.03×10^{-4}	2.11×10^{-4}	205%
1000	1.5	0.0	1.67×10^{-4}	4.71×10^{-4}	282%
1000	1.5	0.5	2.72×10^{-4}	8.71×10^{-4}	320%
250	0.5	0.0	3.10×10^{-6}	6.95×10^{-6}	224%
250	0.5	0.5	8.75×10^{-5}	7.83×10^{-5}	89%
250	1.0	0.0	1.77×10^{-4}	2.95×10^{-4}	167%
250	1.0	0.5	6.71×10^{-4}	7.46×10^{-4}	114%
250	1.5	0.0	9.28×10^{-4}	1.66×10^{-3}	179%
250	1.5	0.5	2.55×10^{-3}	2.90×10^{-3}	114%
50	0.5	0.0	5.02×10^{-5}	1.35×10^{-5}	26%
50	0.5	0.5	2.63×10^{-4}	3.07×10^{-4}	117%
50	1.0	0.0	1.71×10^{-3}	9.44×10^{-4}	55%
50	1.0	0.5	2.83×10^{-3}	2.69×10^{-3}	95%
50	1.5	0.0	1.15×10^{-2}	5.59×10^{-3}	49%
50	1.5	0.5	1.42×10^{-2}	1.03×10^{-2}	73%

Table 4.1: Comparison results for single fraction sediment transport

Table 4.1 shows that the parametric equations generally agree with TR2004 method to within a factor of 3 for the test calculations, including for the 50 μm grain size, which is below the minimum recommended particle size for the parametric model.

Tables 4.20 and 4.30 show results from the TR2004 method and the simplified version used in this project for sediment divided into four fractions. For these test results it can be seen that the simplified method agrees well with the TR2004 method with similar differences in the results to those between the TR2004 method and the parametric equations, for single fraction transport.

H = 10m, v = 1.0 m/s, H_s = 0m, T_p = 10s				
Fraction Size (µm)	**Proportion (%)**	**Volumetric Transport (q_t)**		
		TR2004 (Full)	**TR2004 (Simplified)**	**Ratio**
50	25	1.01×10^{-3}	7.35×10^{-4}	73%
100	25	2.94×10^{-4}	1.28×10^{-4}	43%
200	25	3.18×10^{-5}	1.56×10^{-5}	49%
500	25	4.77×10^{-6}	1.96×10^{-6}	41%

Table 4.2: Comparison results for multi-fraction sediment transport (1)

H = 10m, v = 1.0 m/s, H_s = 0.5, T_p = 10s				
Fraction Size (µm)	**Proportion (%)**	**Volumetric Transport (q_t)**		
		TR2004 (Full)	**TR2004 (Simplified)**	**Ratio**
50	25	1.74×10^{-3}	1.56×10^{-3}	90%
100	25	5.58×10^{-4}	3.65×10^{-4}	65%
200	25	8.82×10^{-5}	7.42×10^{-5}	95%
500	25	1.74×10^{-5}	1.15×10^{-5}	66%

Table 4.3: Comparison results for multi-fraction sediment transport (2)

4.5.6 Lateral Down-slope Sediment Transport

Additional down slope transport is included from orthogonally adjacent cells with higher bed levels, into the current cell, as shown in Figure 4.32.

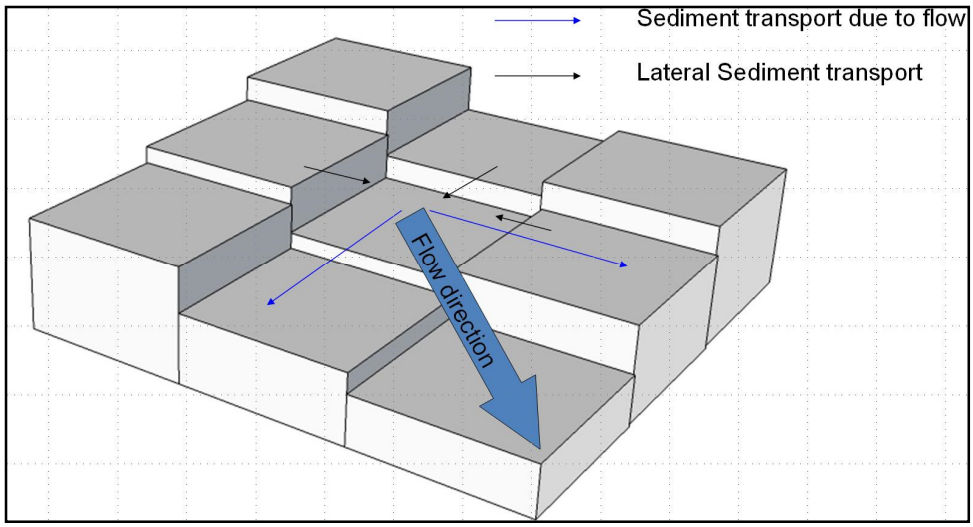

Figure 4.32: Sediment transport directions

Lateral sand transport into a cell is calculated using Equations (4.25) and (4.26), using the current velocity and wave conditions in the present cell and the sediment properties (i.e.

the proportion of mud in the bed) in the adjacent cell (the source of sediment). The down slope sand transport is given by:

$$q_{lat,s,i} = \frac{F_{lat}\,\Delta z_b\left(q_{b,lat,i} + q_{s,lat,i}\right)}{L} \tag{4.115}$$

where $q_{lat,s,i}$ is the lateral sand transport from cell i (i = 1, 2, 3, 4) into the cell 0 (as defined in Figure 4.4), $q_{b,lat,i}$ and $q_{s,lat,i}$ are the bed load and suspended load calculated using the current velocity and wave conditions in cell 0 and the sediment properties in cell i, F_{lat} is a scaling factor (set to 1.0 in the simulations carried out so far) and Δz_b is the difference in bed level between cell i and cell 0 (set to zero if cell 0 has a higher bed level than cell i). The downslope transport of mud is given by:

$$q_{lat,m,i} = F_{lat}\,\Delta z_b\,\frac{E_{mud,lat,i}}{\rho_b\,L} \tag{4.116}$$

where $E_{mud,lat,i}$ is the rate of mud erosion calculated using Equation (4.35) and the current velocity, wave properties and sediment properties from cells 0 and i as described above for lateral sand transport.

4.5.7 Morphological Updating

Since the proportion of mud in the bed in each cell can change over time, the bed composition is stored in a series of layers, which are updated at each time-step. A top, active layer, stores the depth of sand and mud in the bed and its thickness is increased or decreased by the deposition or erosion occurring at each time-step. The calculated sand transport for each cell is divided between its four orthogonal neighbours according to:

$$q_{t,i} = \frac{Q_{o,i}\;q_t}{\sum Q_{o,i}} \tag{4.117}$$

where $q_t = q_s + q_b$, $Q_{o,i}$ (i = 1, 2, 3, 4) are positive outflows from the cell into each of the four orthogonal neighbours ($Q_{o,i}$ is set to zero for inflows) and $q_{t,i}$ is the volumetric sand transport rate, per unit width of bed, into each of the four neighbouring cells. An increase

in the depth of sand in the active layer $(\Delta S'_a)$, due to transport out of the cell into its orthogonal neighbours (will be a negative value in this case) is then calculated according to:

$$\Delta S'_a = -\frac{\Delta t \sum q_{t,i}}{(1-\varepsilon) L} \tag{4.118}$$

where ε is the porosity of the bed (taken to be constant regardless of bed composition) and Δt is the time-step. If $-\Delta S_a$ is greater than the total depth of sand available in the active layer then it is set equal to minus the available depth and the values of $q_{t,i}$ are proportionally reduced to maintain mass conservation of sediment. A similar adjustment is made if $-\Delta S_a$ plus $-\Delta M_a$ (see below) would cause the bed level to be reduced to a value below the minimum value set for the cell (representing non-erodible geology). When this has been completed for all cells the total increase in the depth of sand can be calculated for each cell, according to:

$$\Delta S_a = \frac{\Delta t \sum q'_{t,j}}{(1-\varepsilon) L} + \Delta S'_a \tag{4.119}$$

where $(j = 1, 2, 3, 4)$ are the volumetric sand transport rates into the cell from each of its orthogonal neighbours. An increase in the depth of mud (ΔM_a) in the active layer is given by:

$$\Delta M_a = -\frac{\Delta t \left(E_{mud} - D_{mud}\right)}{\rho_b} \tag{4.120}$$

where ρ_b is the sediment bulk density. Here ΔM_a can be positive if deposition is greater than erosion for the cell; however, if $-\Delta M_a$ is greater than the available depth of mud in the active layer (M_a), then the value of E_{mud} is reduced according to:

$$E_{mud} = \rho_b \left(M_a + D_{mud}\right) / \Delta t \tag{4.121}$$

Again, a reduction is made if $-\Delta S_a$ plus $-\Delta M_a$ would cause the cell bed level to be reduced to a value below the minimum value for the cell. Once the increases in the depths of sand

and mud in the active layer have been calculated, the bed level for the cell in increased according to:

$$\Delta z_b = \Delta S_a + \Delta M_a \tag{4.122}$$

Once the changes to the depths of sand and mud in the active layer have been calculated for all cells, the active layer is restored to a default thickness by transferring material to or from a series of sub-layers, which have constant thickness. The base elevation of the active layer in each cell varies with the bed level, while the sub-layer elevations are fixed. Only sub-layers with elevations fully or partly below the active layer are used at any given time and only the proportion of mud (p_{mud}) is stored for each sub-layer. The number of sub-layers is determined based on model parameters for maximum and minimum bed elevations and the sub-layer thickness. Figure 4.33 shows the procedure for updating the active layer and sub-layers following erosion.

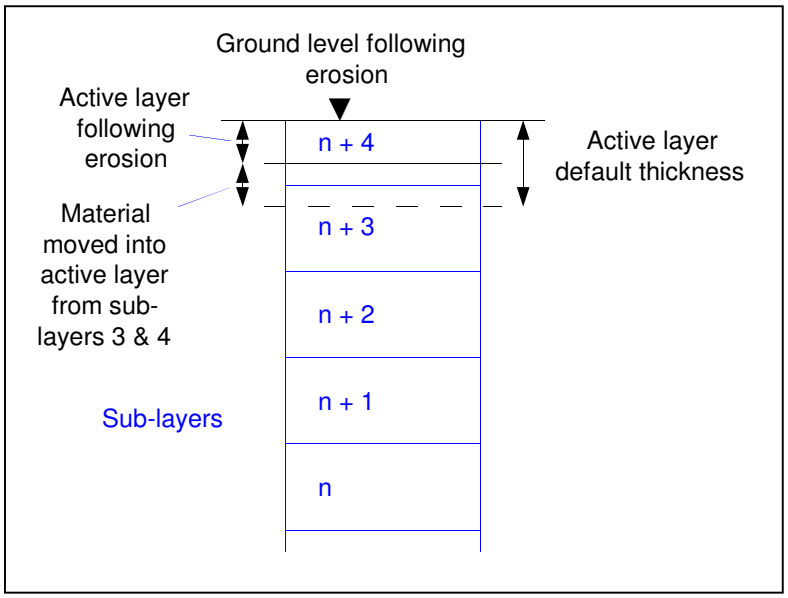

Figure 4.33: Active layer updating procedure, following erosion

In the example scenario shown in Figure 4.33 the active layer is restored to default thickness by the transfer of material from sub-layers $n+3$ and $n+4$, into the active layer, in accordance with Equations (4.123) and (4.124).

$$S_b = S_a + (1 - p_{mud,n+4})(z_b - A_e - B_{n+4}) + (1 - p_{mud,n+3})(B_{n+4} - z_b + A_{df}) \quad (4.123)$$

$$M_b = M_a + p_{mud,n+4}(z_b - A_e - B_{n+4}) + p_{mud,n+3}(B_{n+4} - z_b + A_{df}) \quad (4.124)$$

Where M_b and S_b are the depths of mud and sand per unit area in the active layer following the update to default thickness, $p_{mud,n+3}$ and $p_{mud,n+4}$ are the proportion of mud in sub-layers $n+3$ and $n+4$, z_b is the bed level, A_e is the active layer thickness following erosion, B_{n+4} is the level at the base of sub-layer $n+4$ and A_{df} is the default thickness of the active layer. A similar procedure is used for updating the active layer following deposition, shown in Figure 4.34.

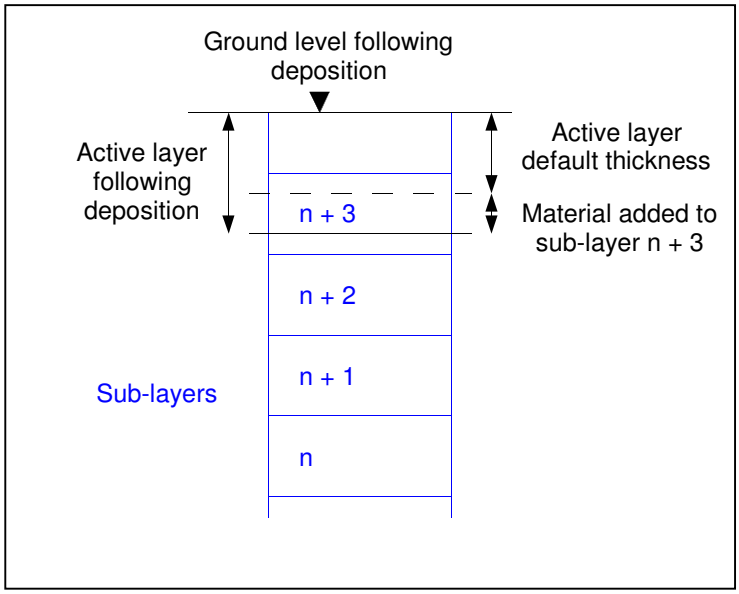

Figure 4.34: Active layer updating procedure, following deposition

In this example the active layer is restored to its default thickness by the transfer of material from the active layer to sub-layer $n+3$ according to:

$$p_{mud,n+3} = \frac{p_{mud,n+3,d}(z_b - A_d - B_{n+3}) + \frac{M_a}{A_d}(A_d - A_{df})}{z_b - A_d - B_{n+3}} \quad (4.125)$$

Where $p_{mud,n+3}$ is the proportion of mud in sub-layer $n+3$ following the update, $p_{mud,n+3,d}$ is the proportion of mud in sub-layer $n+3$ before update, following deposition and A_d is

the thickness of the active layer following deposition. In this case the active layer is reduced to default thickness while maintaining the relative proportions of sand and mud:

$$S_b = \frac{S_a \; A_{df}}{A_d} \tag{4.126}$$

$$M_b = \frac{M_a \; A_{df}}{A_d} \tag{4.127}$$

For multi-fraction sediment transport the active layer stores a depth for each sediment fraction, while each sub-layer stores the proportion of each fraction within the bed material. Updating of the active layer and sub-layers follows a similar procedure to that described above for a single sand fraction.

4.5.8 Sediment Boundary Conditions

The import or export of fine sediment at the upstream and downstream model boundaries is determined from a boundary sediment concentration, specified as a depth of settled bed material per metre depth of water. Fine sediment crossing the boundary is then calculated according to the diffusion calculations, as described in Section 4.5.2.

Sand input to the model domain at the tidal boundary is specified as a constant time-averaged input applied to the boundary cells at each time-step. Sand is added along the boundary at each time-step in proportion to the calculated sand transport rate (in this case assuming the bed is composed of 100% sand) or, if the sand transport rates are very small, in proportion to the maximum bed shear stress due to combined waves and currents (τ_{max}). At present it is a requirement that the marine boundary of the model is aligned with the CA grid, such that it is formed by the last row of grid cells. The addition of sand at one end of the model boundary was considered in order to represent a supply of sand due to long shore transport on the adjacent coast; however, the present model is unable to reproduce the littoral transport processes needed to redistribute the sediment in a realistic manner because sand transport by waves is not represented (only enhancement by waves of sand transport due to currents). Hence, sand supply due to long shore transport is not included in the present model.

The export of sand during the ebb tidal phase is based on the calculated sediment transport rates at the boundary.

4.6 Salt Marsh

Since regular inundation by tides is the principal mechanism of sediment delivery to marsh platforms, their elevation is linked to mean sea level and tidal amplitude. In the present model marsh extent and biomass are primarily dependent on elevation (via inundation time).

The salt marsh representation used for this project is based on elements of the 1D numerical salt marsh model developed by Mariotti and Fagherazzi (2010). Marsh biomass potential is based on inundation time, averaged over a tidal cycle or a spring-neap tidal cycle in this research, which is linked to mean sea level and tidal propagation as well as marsh elevation. This relationship is implemented in the model using a simple piecewise function as illustrated by the example in Figure 4.35, allowing both relationships proposed by Alpaos et. al. (2006) to be represented (see Section 3.4.7).

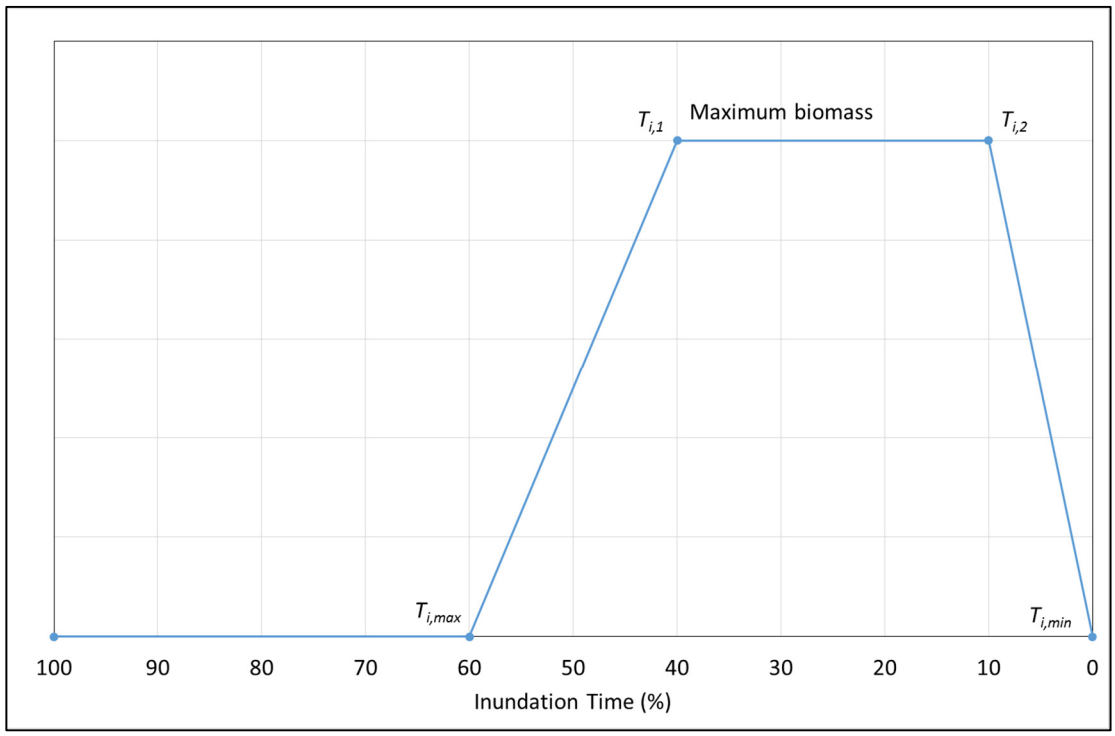

Figure 4.35: Biomass function

Biomass potential increases from zero, where the average inundation time is at the maximum value of $T_{i,max}$, to a maximum value where the average inundation falls to $T_{i,1}$. For inundation times between $T_{i,1}$ and $T_{i,2}$ the biomass remains at the maximum value and above $T_{i,2}$ the biomass falls back to zero, at $T_{i,min}$. The variables $T_{i,max}$, $T_{i,1}$, $T_{i,2}$ and $T_{i,min}$ are adjustable model parameters.

In this research the inundation time is determined from the number of time-steps for which a positive water depth exists in the cell, during a complete cycle of spring to neap tides. The following simple equation is used to update the biomass when the biomass potential is different to current biomass in the cell:

$$B_{m,t} = B_{m,t-1} + \left(B_{p,t} - B_{t-1}\right)^{a_1 t_s} \tag{4.128}$$

Where $B_{m,t}$ is the updated biomass at the current time-step, $B_{m,t-1}$ is the biomass at the previous time-step, $B_{p,t}$ is the biomass potential at the current time-step, t_s is the duration of one time-step in seconds and a_1 is a model parameters (arbitrarily set to 1 x 10^{-6} at present, which allows B_m to reach B_p within a period of a few weeks). Marsh is usually found between mean sea level and the mean high water level, although other factors such as tidal amplitude, latitude, temperature and sediment supply can influence marsh elevation (Fagherazzi, et. al., 2012). Mudd et al. (2004) have proposed a maximum value of 0.6 for $T_{i,max}$ (expressed as a proportion of time for which the marsh is inundated), which will allow a small amount of marsh growth below mean sea level, and a value of 0.02 for $T_{i,min}$.

Increased resistance to erosion, enhanced sedimentation and enhanced wave attenuation are related to biomass, using the relationships given by Mariotti and Fagherazzi (2010):

$$\tau_{cr,marsh} = \tau_{cr,0}\left(1 + k_{veg} B_m / B_{\max}\right) \tag{4.129}$$

where $\tau_{cr,marsh}$ is the adjusted critical bed shear stress, $\tau_{cr,0}$ is the critical shear stress without marsh, k_{veg} is a model parameter, B_m is the biomass and B_{max} is the maximum biomass. Mariotti and Fagherazzi (2010) adopted values of 5 and 2000 for k_{veg} and B_{max}, respectively.

$$D = D_s + D_t \tag{4.130}$$

Here, D is the enhanced deposition rate, D_s is the deposition rate due to settling and D_t is the deposition rate due to sediment trapping by marsh vegetation, given by:

$$D_t = c u_c \eta d_s n_s h_{st} \tag{4.131}$$

where c is the sediment concentration, η is the rate of sediment trapping by plant stems, d_s is the stem diameter, n_s is the stem density per unit area and h_{st} is the average stem height. The parameters η, d_s, n_s and h_{st} are given by:

$$\eta = 0.224 \left(\frac{u d_s}{\upsilon} \right)^{0.718} \left(\frac{d_p}{d_s} \right)^{2.08} \tag{4.132}$$

$$n_s = 250 B_m^{\,0.3032} \tag{4.133}$$

$$h_{st} = 0.0609 B_m^{\,0.1876} \tag{4.134}$$

$$d_s = 0.0006 B_m^{\,0.3} \tag{4.135}$$

where d_p is the particle diameter and ν is the kinematic viscosity of water. Mariotti and Fagherazzi (2010) give the following relationship to calculate the attenuation of wave height when propagating over salt marsh:

$$H_{reduction}(\%) = 3 \frac{B_m}{B_{\max}} L_{Att} \tag{4.136}$$

where $H_{reduction}$ is the percentage reduction in wave height and L_{Att} is the distance over which the wave propagates. When applied to the CA model with a cell size of 50m this formula gives a percentage reduction of more than 100% for a single cell and in this research the following exponential function is used instead:

$$H_{reduction}(\%) = 100\left(1 - 0.97^{L_{Att}}\right)\frac{B_m}{B_{\max}} \tag{4.137}$$

The wave height applied within each cell is then taken to be the average of the wave height entering the cell (i.e. before attenuation) and the wave height leaving the cell (reduced according to Equation 4.137). Increased hydraulic roughness was estimated with reference to the Manning's n values given by Chow (1959) for vegetated and non-vegetated conditions, using the approximation $C \approx 1 / n$. A linear relationship was assumed, with Chezy C values of 10 and 50 for maximum and zero biomass respectively.

In this research it has been assumed that if the enhanced critical bed shear stress for the marsh is exceeded then the marsh is quickly destroyed by erosion. In this case marsh biomass is reduced according to:

$$B_{m,t} = \frac{B_{m,t-1}}{(T_m + 1)^{a_2 t_s}} \tag{4.138}$$

Where a_2 is model parameter (arbitrarily set to 2.78 x 10^{-4} at present, to make the exponent approximately equal to 1.0 for a time-step of 1 hour) and T_m is a transport parameter given by:

$$T_m = \frac{\tau_{\max} - \tau_{cr,marsh}}{\tau_{cr,marsh}} \tag{4.139}$$

where τ_{max} is given by Equation (4.36).

4.7 Model Implementation and Results

The model has been developed as a Fortran computer programme, which is compiled as a standalone executable console application to run on computers using the Microsoft Windows 8.1 operating system. A complete listing of the program source code is provided in Appendix A.

Input to the model is provided in a series of text files:

- A 'control file', specifying general model parameters such as time-step, run duration, tidal range, wave conditions, sediment properties etc., links to other input files and output file details.

- Initial bathymetry, minimum bed levels, initial salt marsh coverage and initial sediment composition are provided in ERSI ASCII grid format.

- For multi-fraction simulations an additional sediment control file is used to specify the properties of each sediment fraction.

Examples input files are provided in Appendix B.

Output files are generated in ERSI ASCII grid format at intervals during each model run, including bathymetry, salt marsh coverage and sediment composition in the active layer. Wave height and velocity field grids may also be optionally generated. A sediment mass balance file is also generated, giving changes in the total volumes of sand and mud together model domain with the total sand and mud input and output volumes. Example output files are provided in Appendix C.

The performance of the model has been assessed using a series of sensitivity tests and hypothetical future scenarios. These tests are described and their results presented in the following chapters.

CHAPTER 5

SENSITIVITY TESTING

5.1 Introduction

A series of sensitivity tests were performed to assess the performance of the model described in Chapter 4 and to investigate its sensitivity to a range of input conditions and model settings. In each test one model parameter was modified, relative to a baseline scenario.

The purpose of this sensitivity testing is to assess the behaviour of the model in response to variations in a range of input parameters. At present the model has not been tested against any field or laboratory data and these tests are therefore needed to provide a qualitative assessment of model performance. Results from the tests will show whether the model is behaving as expected; for example, increases in wave energy or tidal flows are expected to cause increased erosion, while sedimentation rates are expected to be influenced by the boundary sediment concentration and diffusion coefficient. The tests have been divided into six categories as shown in Table 5.1.

Category	Sensitivity Tests
General model settings	• Time-step • Cell size
Hydrodynamic settings	• Tidal range • Fluvial inflow • Wave energy
Initial conditions	• Initial bathymetry • Initial sediment composition
Boundary conditions	• Non-erodible bathymetry • Sand input at marine boundary • Suspended sediment concentration at boundary
Sediment properties	• Grain size (sand) • Critical shear stress for erosion & deposition of mud • Multi-fraction sand transport (comparison with single sand fraction results) • Sediment diffusion coefficient
Salt marsh	• Biomass function parameters

Table 5.1: List of sensitivity tests

A standard set of default model parameter values was used for the baseline and all sensitivity test scenarios, with only one parameter varied for each sensitivity test. The baseline parameter values are given in Table 5.2.

Category	Parameter	Default Value
Model parameters	Cell size (L)	50 m
	Time-step (t_s)	0.621 hours
Hydrodynamics	Fluvial inflow (Q_f)	10 m^3/s
	Spring and neap tidal range (Z_s, Z_n)	4 m, 2m
	Prevailing wind speed (μ)	5 m/s
	Wind speed standard deviation (σ)	5 m/s
Sediment properties (sand)	10^{th} percentile particle size (d_{10})	100 µm
	Median particle size (d_{50})	150 µm
	90^{th} percentile particle size (d_{90})	200 µm
Sediment properties (mud)	Critical shear stress for erosion (τ_e)	0.12 N/m^2
	Critical shear stress for deposition (τ_d)	0.06 N/m^2
	Diffusion coefficient (K)	500 $m^{-2}s^{-1}$
Boundary conditions	Sand input at marine boundary (S_{in})	5 x 10^{-5} m^3/s
	Volumetric suspended sediment concentration at marine boundary (C_{ds})	1 x 10^{-4}
Salt marsh	(Salt marsh not included in the baseline scenario)	-
Other parameters (not included in sensitivity testing)	Number of rows in model lattice (n_r)	250
	Number of columns in model lattice (n_c)	21
	Mean sea level (Z_{msl})	23 m AD
	Prevailing wind direction (relative to the estuary axis at the marine boundary, with zero indicating a seaward direction)	0°
	Fetch at marine boundary, for on-shore wind directions (X_{bdy})	10 km
	Maximum Chezy hydraulic roughness (no marsh)	50 $m^{1/2}/s$
	Minimum Chezy hydraulic roughness (maximum marsh biomass)	10 $m^{1/2}/s$
	Particle density (sand) (ρ_s)	2650 kg/m^3
	Bulk density (mud) ($\rho_{b)}$	1200 kg/m^3
	Sediment porosity (ε)	0.4
	Minimum depth (h_{min})	0.05m
	Scaling factor for lateral sediment transport (F_{lat})	1
	Critical shear stress factor for 20% mud (over the value for 100% sand) ($\tau_{e,max}/\tau_{cr,sand}$)	2
	Active layer thickness (A_{df})	0.2m
	Sub-layer thickness (B_{df})	0.2m

Table 5.2: Baseline parameter values

A simple generic estuary bathymetry was created, to provide initial conditions for the sensitivity testing, comprising an array of 250 rows and 21 columns of 50m x 50m cells, forming an initial bathymetry 12.5 km in length and 1.05 km wide. Figure 5.1 shows the

non-erodible and initial bed profiles at the marine boundary (the estuary mouth), representing the estuary basin and surface of the erodible sediment.

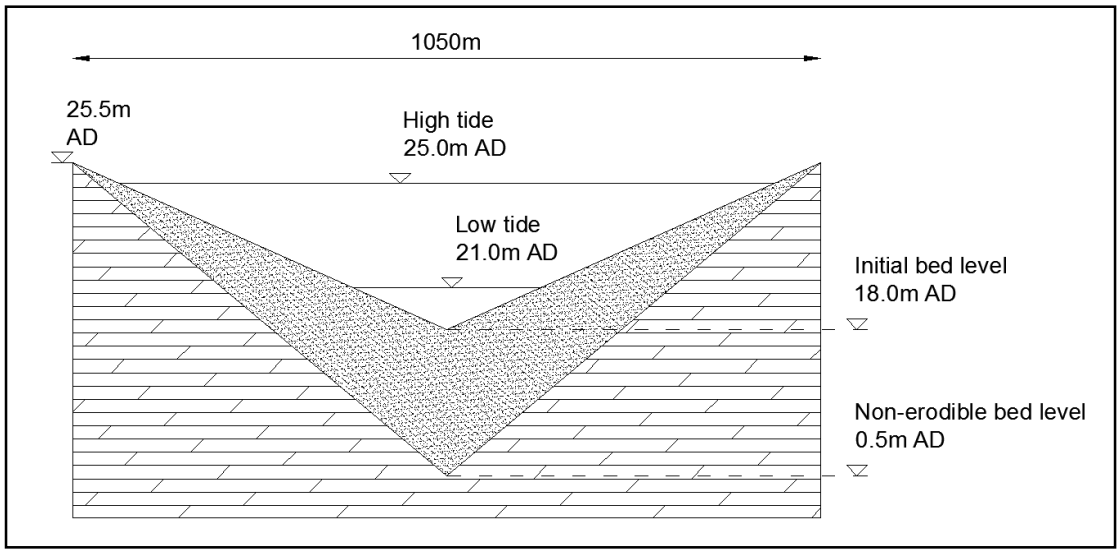

Figure 5.1: Cross section showing the baseline initial bathymetry at the estuary mouth

The non-erodible bed levels and initial bed levels are increased by 0.1 m and 0.03 m respectively with each row, moving upstream, so that both the non-erodible and the initial bed levels in the centre of the channel, at the upstream limit are 25.40 m AD and 25.47 m AD respectively (above an arbitrary datum), as illustrated in Figure 5.2. The maximum cross section levels are limited to 26.0 m AD (this is above the maximum water level and hence does not affect the simulation results).

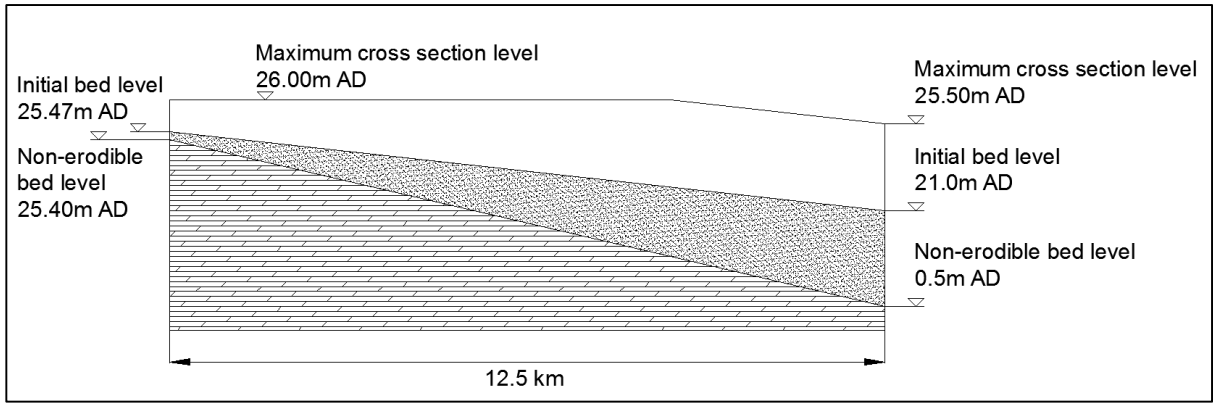

Figure 5.2: Long section showing the baseline initial bathymetry

These dimensions were arbitrarily selected but are intended to be broadly similar to some UK estuaries (e.g. the Deben). Similarly, the model inputs are broadly typical of some

estuaries in the UK; for example, the spring tidal range is larger than that in the Deben Estuary (spring tidal range in the Deben is around 3.5m) but less than the tidal range in the Humber Estuary (spring and neap tidal ranges in the Humber are around 7m and 3.5m respectively). Likewise, the fluvial inflow of 10 m^3/s is within the wide range of mean flows entering UK estuaries (mean flow entering the Deben is below 1 m^3/s; mean flow in the River Trent, one of the two larger rivers entering the Humber Estuary, is around 90 m^3/s; mean flow entering Ribble Estuary is around 33 m^3/s). Wind conditions are given by McWilliams and Sprevak (1982) for Aldergrove in Northern Ireland, where the mean prevailing wind speed (μ) varies seasonally between zero and 7 knots (0 – 3.6 m/s) and the standard deviation of the wind speed (σ) varies between 7 and 8 knots (3.6 – 4.1 m/s). The adopted values of 5.0 m/s for both μ and σ are therefore similar to, if slightly higher than, those at Aldergrove.

For consistency, model results will generally be provided in the form of a time-series plot showing changes in sediment volume and final cross section profiles at chainages 5.0, 7.5, 10.0 and 12.5 km (measured from the upstream model boundary). Additional results will be provided where necessary to illustrate the outcome of each test.

5.2 Baseline Scenario

An initial model simulation was carried out using the generic bathymetry described above and the default model parameters listed in Table 5.2. This simulation includes tidal flows, fluvial flows and locally generated waves (but not salt marsh). Sediment transport was modelled using a single sand fraction and a mud fraction.

A baseline simulation period of 430,000 hours (approximately 50 years) was selected for practical reasons due to time required to complete each model run (approximately 12 hours for one 50 year simulation) and it is accepted that equilibrium conditions will not be established in the estuary during this period. The changes in sediment volumes occurring during the baseline simulation are shown in Figure 5.3.

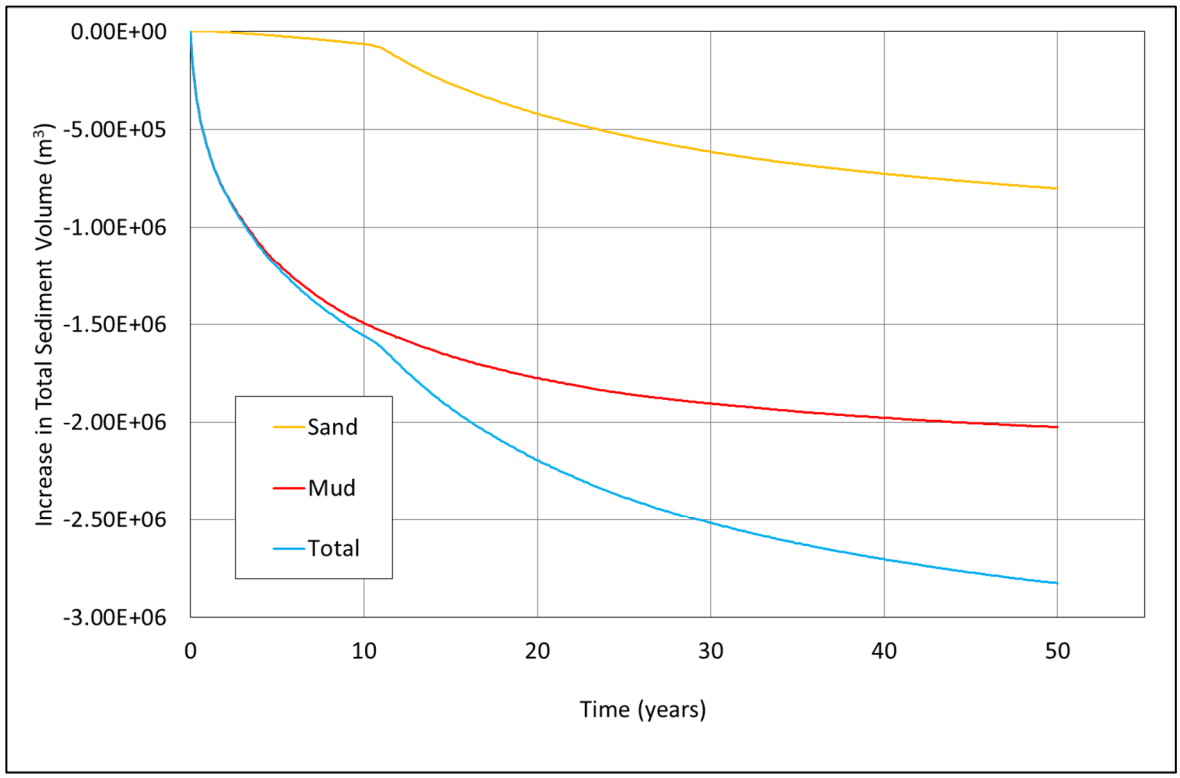

Figure 5.3: Change in sediment volumes during the baseline simulation

From Figure 5.3 it can be seen that the volumes of both sand and mud contained within the estuary have been significantly reduced during the simulation period. This reflects the artificial nature of the initial bathymetry and the consequent initial adjustment based on the imposed hydrodynamic forcings (tides, waves and fluvial inflow); however, the rate of change is steadily reduced towards the end of the simulation, indicating that an equilibrium bathymetry may eventually be reached following a sufficient simulation period. This is further illustrated in the contour plots showing the initial conditions and baseline results, in Figure 5.4.

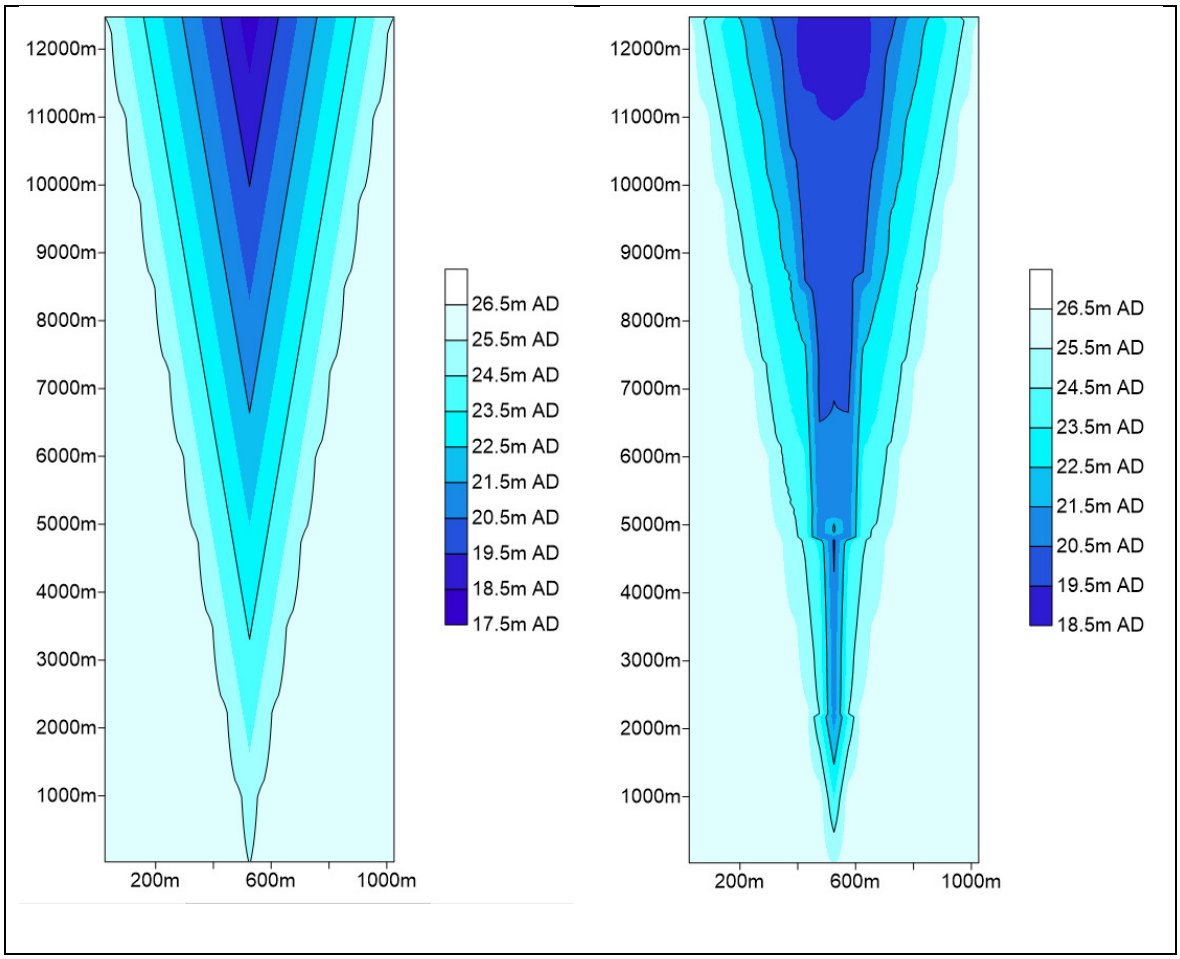

Figure 5.4: Contour plots showing the initial bathymetry (left) and baseline results (right)

Figure 5.4 confirms that significant erosion of sediment has occurred, with the deepest areas of the estuary being significantly increased in area. Cross section profiles at chainages 2,500, 5,000m, 7,500m and 10,000m are given in Figure 5.5

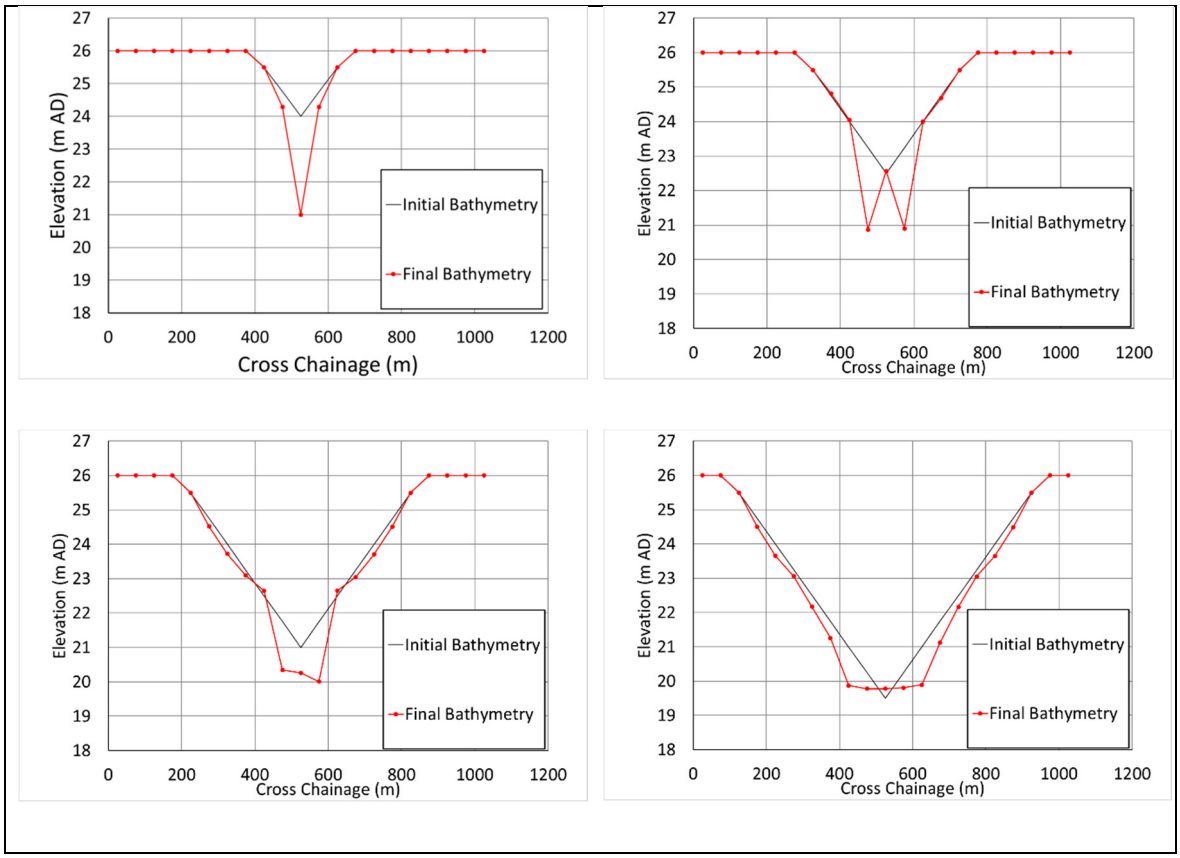

Figure 5.5: Initial conditions and baseline cross section results at chainage 2,500m (top left), 5,000m (top right), 7,500m (bottom left) and 10,000m (bottom right)

From Figure 5.5 it can again be seen that the main process occurring during the baseline simulation is erosion, as the morphology has adjusted from the initial conditions. The erosion has also generally been concentrated in the deeper areas, creating a central channel, particularly at chainages 5,000m and 10,000m. At chainage 5,000m twin parallel low-water channels have been formed.

5.3 General Model Settings

5.3.1 Time-step

The model time-step must be set to divide exactly into the tidal period and the baseline time-step of 0.621 gives exactly 20 time-steps per tidal cycle. To assess the sensitivity to time-step the baseline scenario was repeated using time-steps of 1.242 hours (10 per tidal

cycle), 0.311 hours (40 per tidal cycle) and 0.155 hours (80 per tidal cycle). The change in total sediment volume occurring during these simulations in given in Figure 5.6.

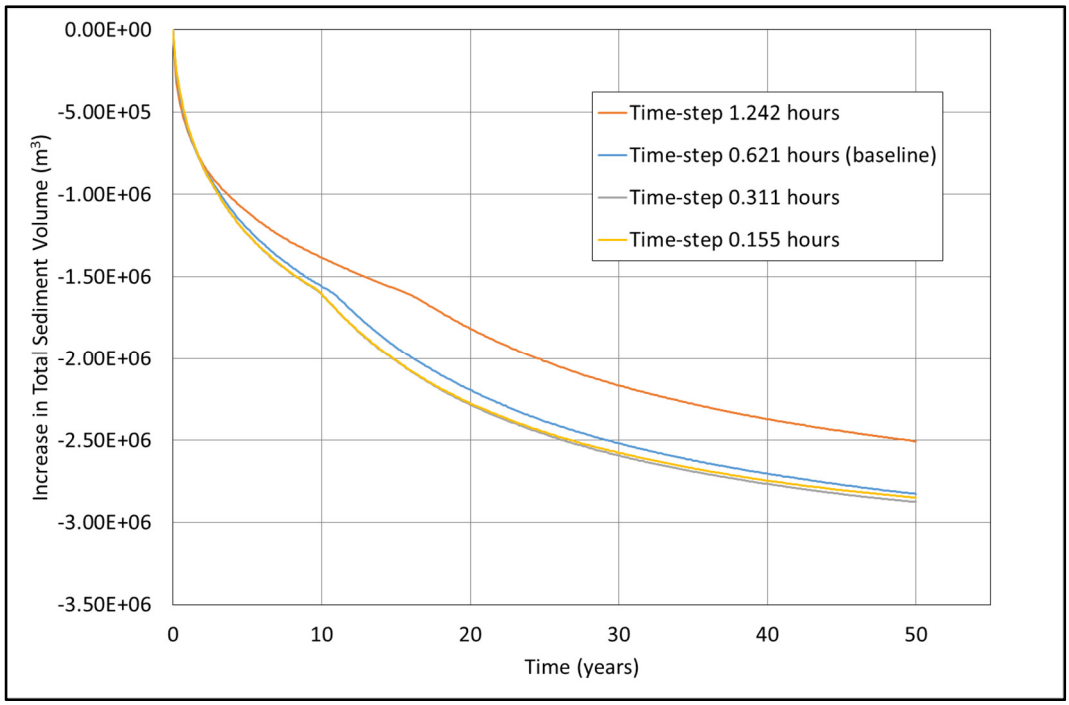

Figure 5.6: Change in total sediment volume: sensitivity to time-step

Based on Figure 5.6 reducing the time-step from the baseline value has only a small effect on the results, while increasing the time-step results in a notable reduction in the modelled rate of erosion. Cross section profiles are given in Figures 5.7 to 5.10.

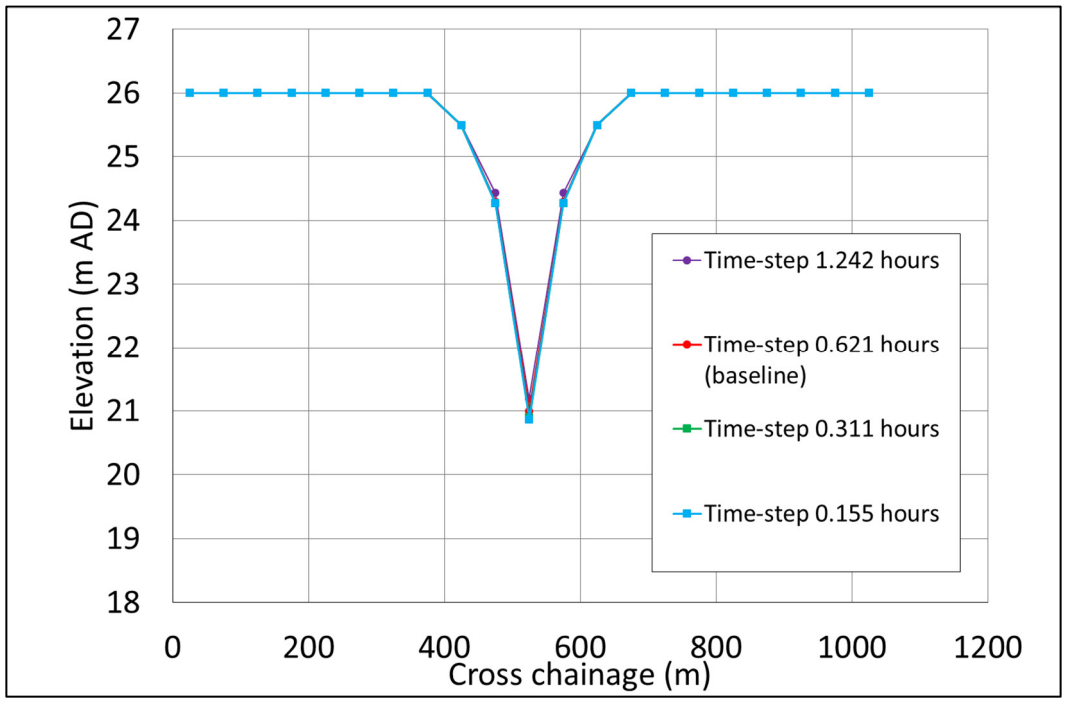

Figure 5.7: Cross section results at chainage 2,500m: sensitivity to time-step

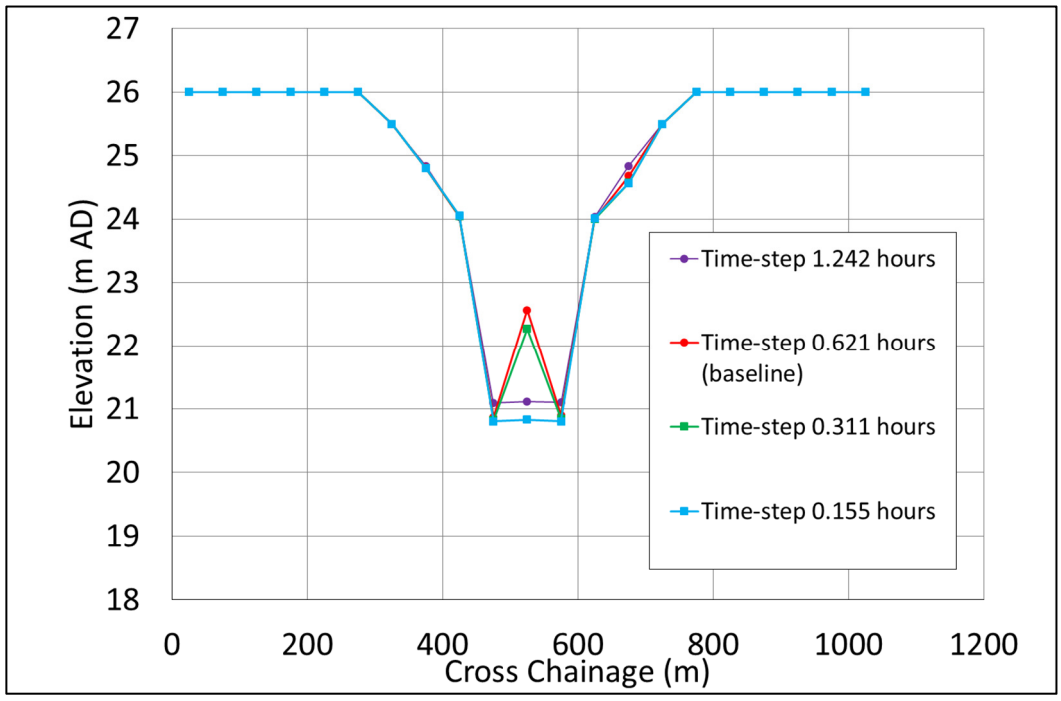

Figure 5.8: Cross section results at chainage 5,000m: sensitivity to time-step

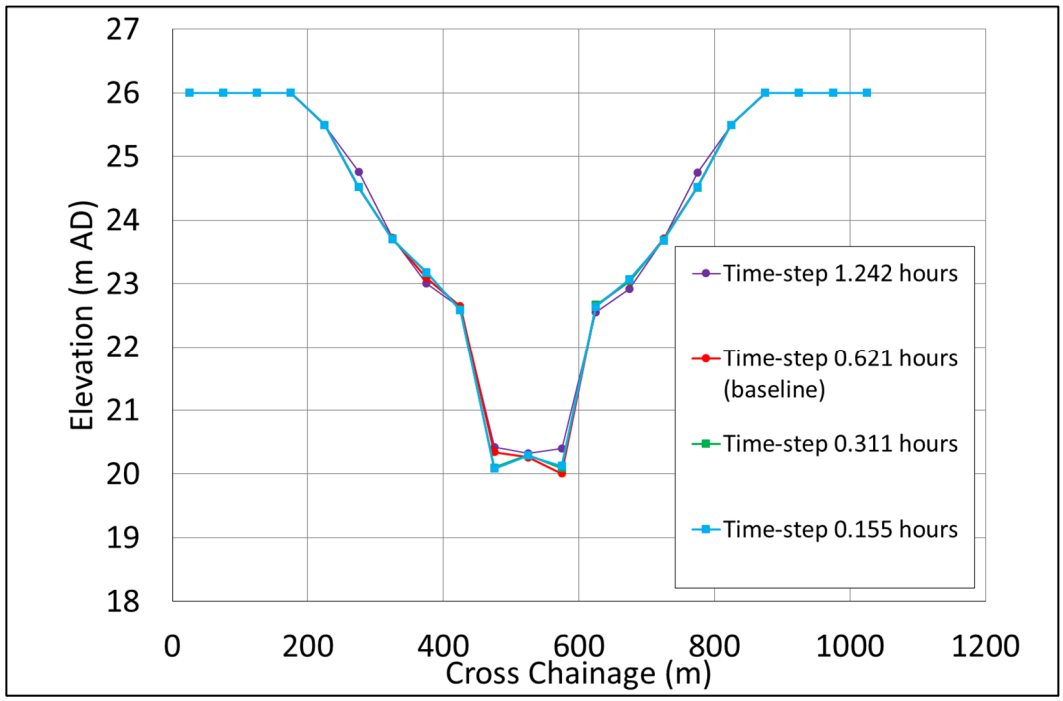

Figure 5.9: Cross section results at chainage 7,500m: sensitivity to time-step

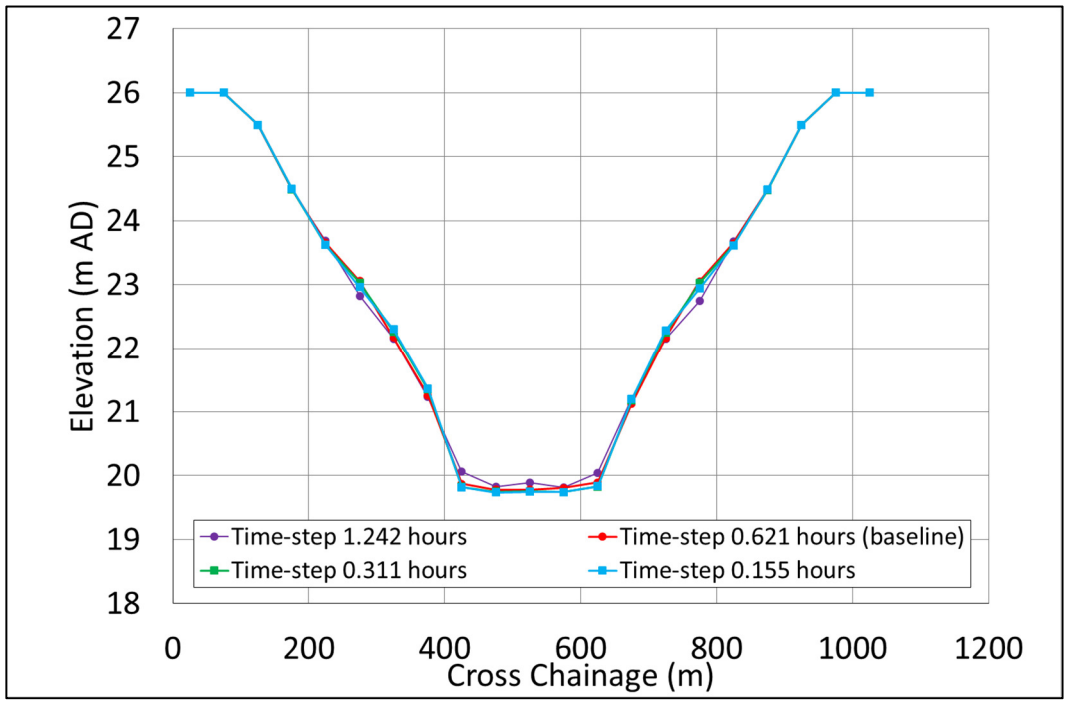

Figure 5.10: Cross section results at chainage 10,000m: sensitivity to time-step

Figures 5.7 to 5.10 confirm that less erosion has taken place for the simulation with 1.242 hour time-steps. This result may be explained by averaging of the velocity during a single time-step, which effectively reduces the peak ebb and flood velocities and hence the peak

erosion rates during each tidal period. This effect is negligible for the shorter time-steps but becomes significant as the time-step increases. The central bar at chainage 5,000m appears to be sensitive to the time-step since it is not present in the results for time-steps of 1.242 hours or 0.155 hours; however, this feature has actually only moved a short distance downstream in these simulations.

5.3.2 Cell size

Model runs have been carried out with the cell size increased to 70m and reduced to 30m, from the baseline value of 50m, in order to assess the sensitivity of the results to variation in this parameter. Figure 5.11 shows the change in total sediment volume occurring during these simulations.

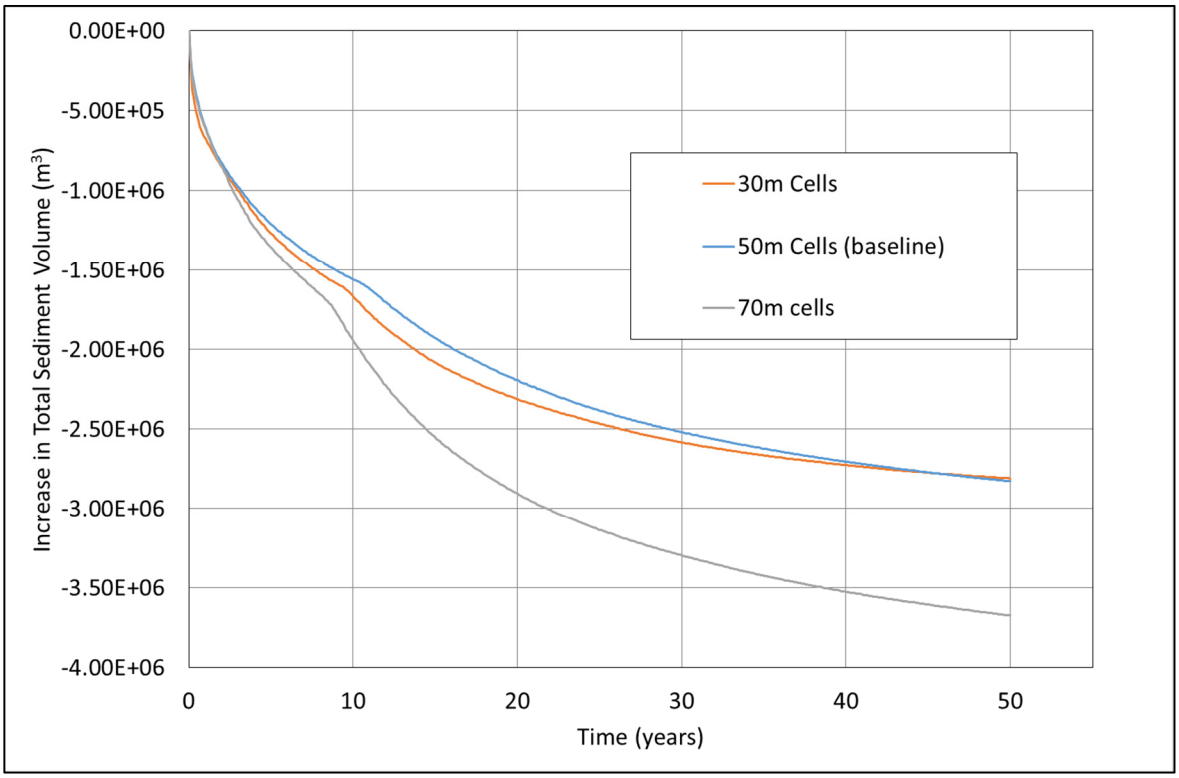

Figure 5.11: Change in total sediment volume: sensitivity to cell size

Figure 5.11 shows that the rate of erosion is notably higher when the cell size is increased to 70m, while the difference between the erosion rates for 30m and 50m cell sizes is much lower. Figures 5.12 to 5.15 show cross section results for these simulations.

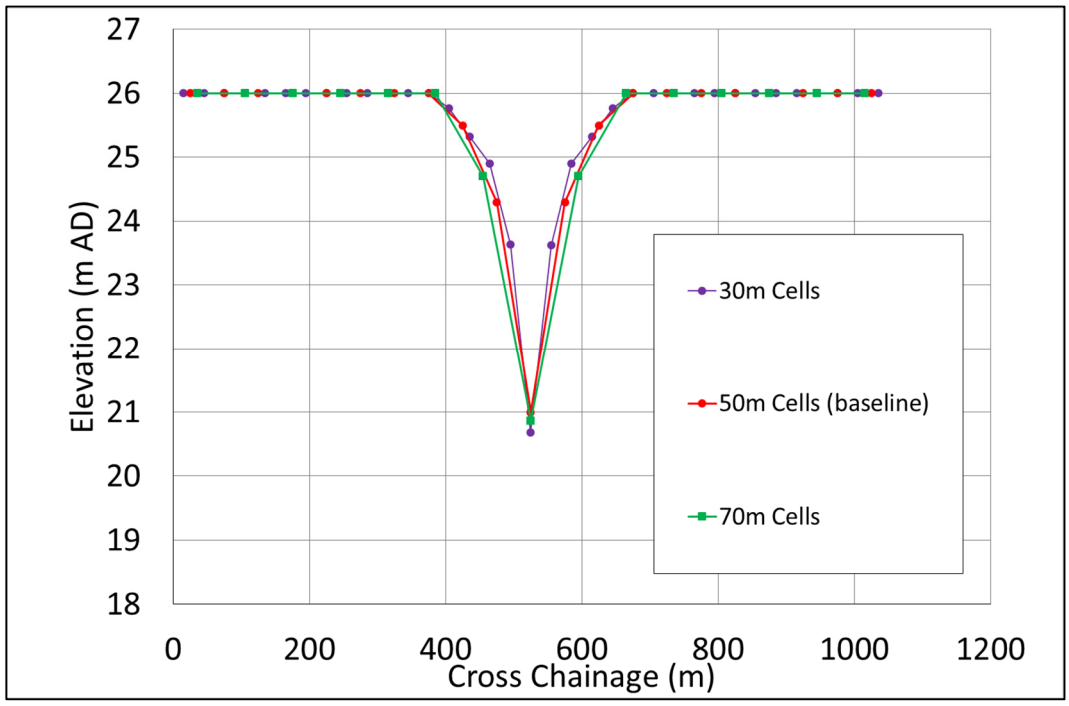

Figure 5.12: Cross section results at chainage 2,500m: sensitivity to cell size

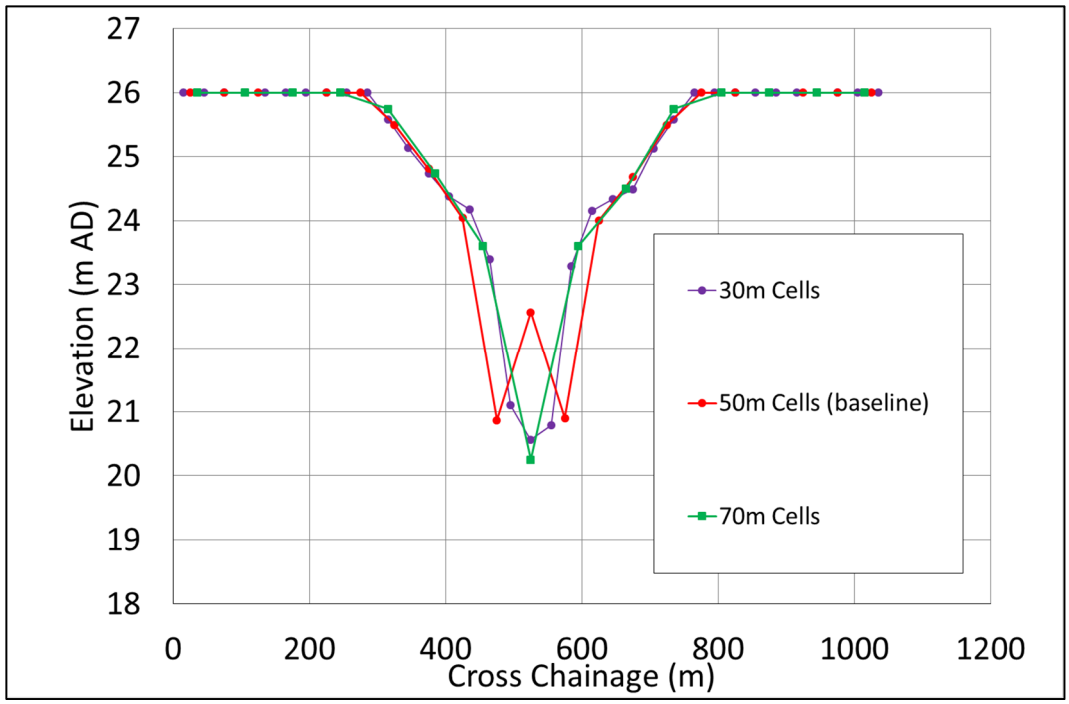

Figure 5.13: Cross section results at chainage 5,000m: sensitivity to cell size

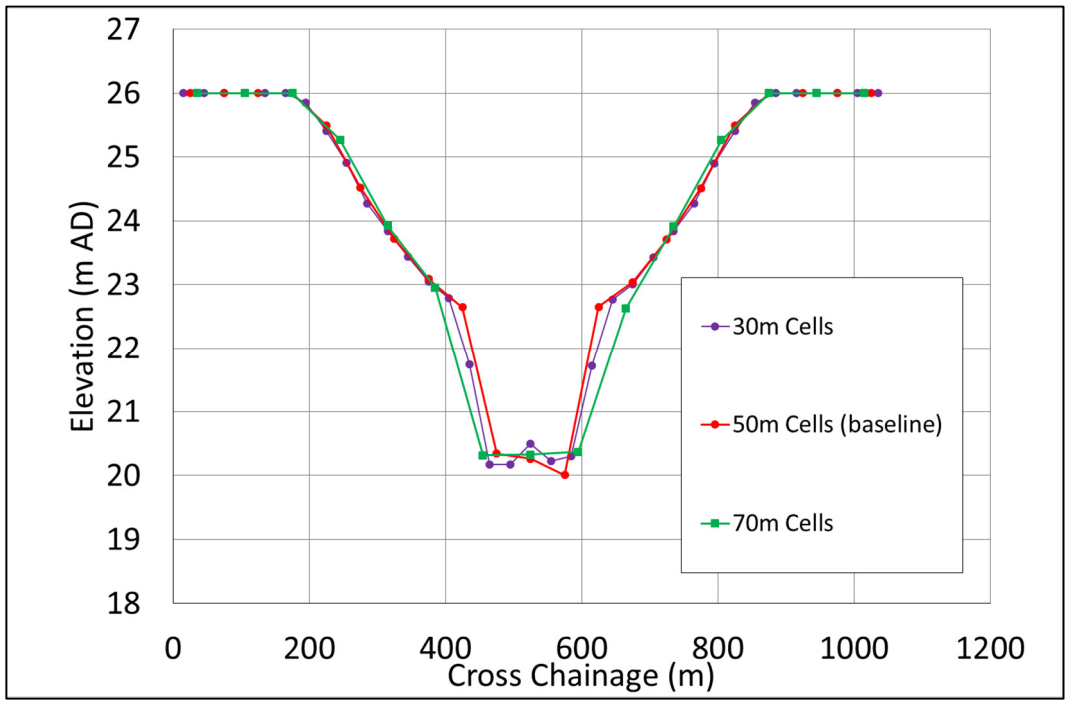

Figure 5.14: Cross section results at chainage 7,500m: sensitivity to cell size

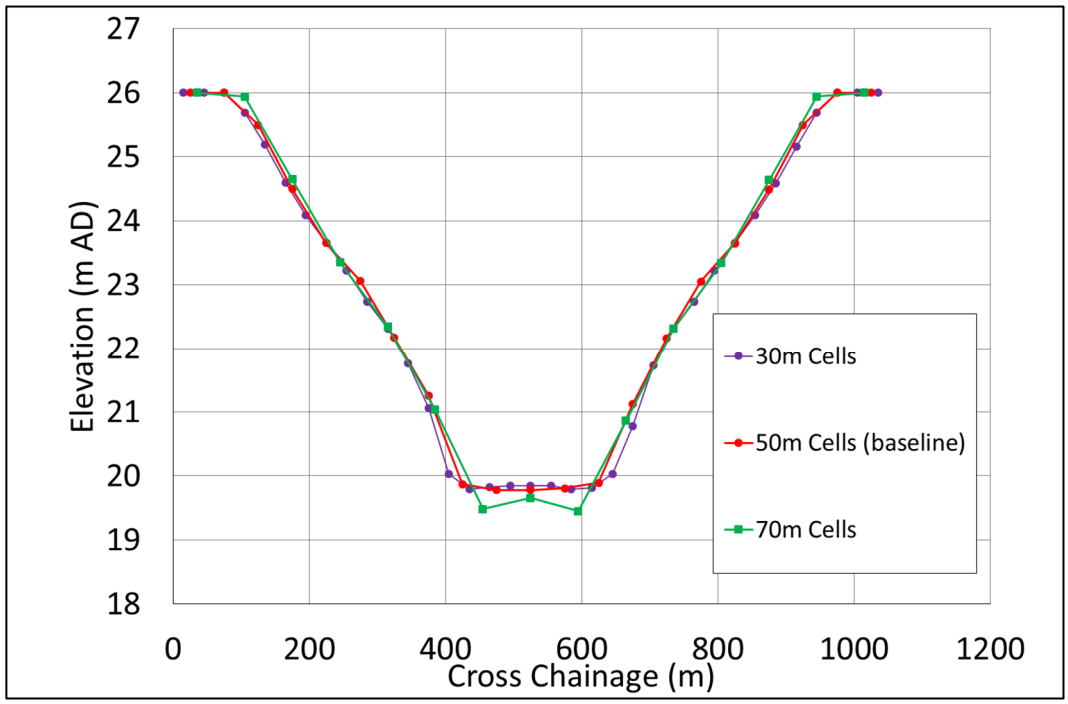

Figure 5.15: Cross section results at chainage 10,000m: sensitivity to cell size

The cross section results show that the results are not very sensitive to cell size, within the range tested. The channel width in the upper reaches is limited to a minimum width of one cell, which may contribute to the greater erosion shown in Figure 5.11, for the simulation

using 70m cells, due to the increased tidal prism in the upper reaches and consequent increase in the downstream tidal flow.

5.4 Hydrodynamic Settings

5.4.1 Fluvial Inflow

The sensitivity of the model to the fluvial inflow (a constant model parameter in the present model) has been tested by performing simulations with the inflow reduced to 5 m^3/s and increased to 20 m^3/s, from the baseline value of 10 m^3/s. The change in total sediment volume during these simulations is given in Figure 5.16.

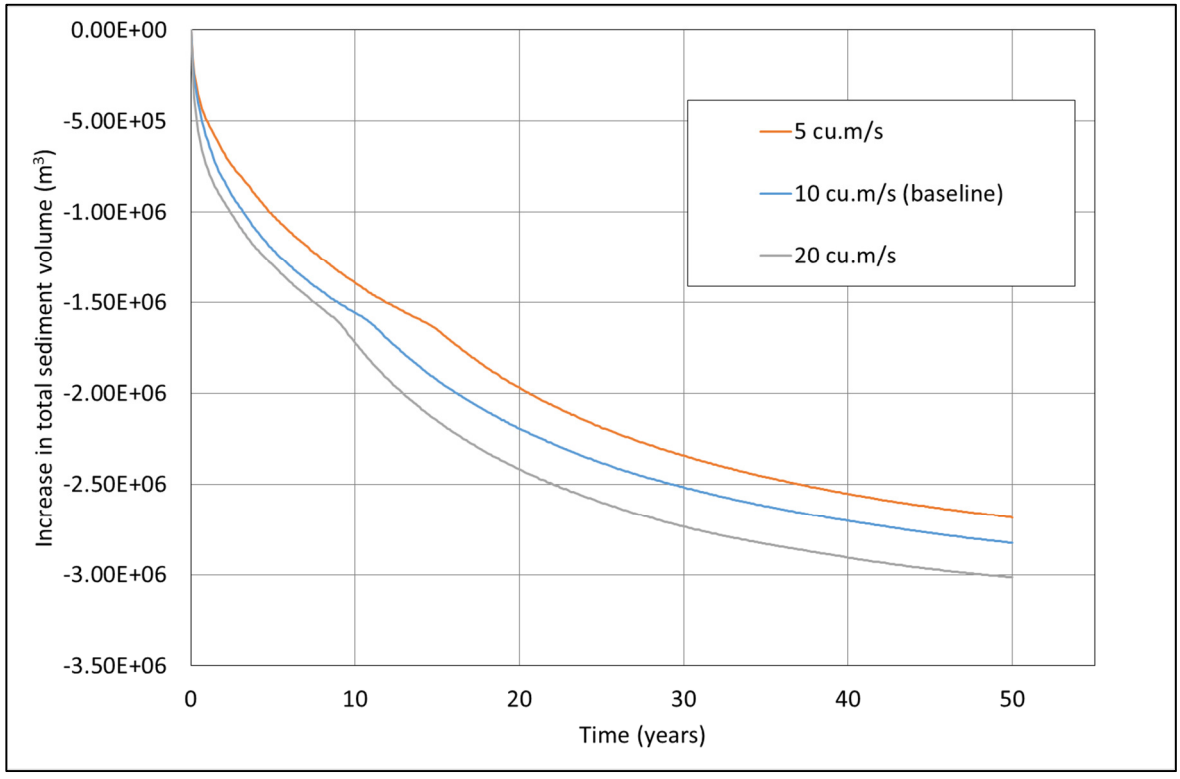

Figure 5.16: Change in total sediment volume: sensitivity to fluvial inflow

From Figure 5.16 it can be seen that greater erosion occurs with increasing fluvial inflow. Cross section results for these simulations are given in Figures 5.17 to 5.20 and these show that the increased erosion for higher fluvial inflow is concentrated in the central channel. The effect from the fluvial inflow is expected to be greatest in the inner estuary, since the relative influence of tidal flow is lower in that area and this is generally

confirmed by these results. The channel bed at chainage 2,500m is only slightly above the specified minimum bed level of 20.5m AD at that location, which may explain why the bed at this cross section appears to be less sensitive to fluvial inflow than that at chainage 5,000m.

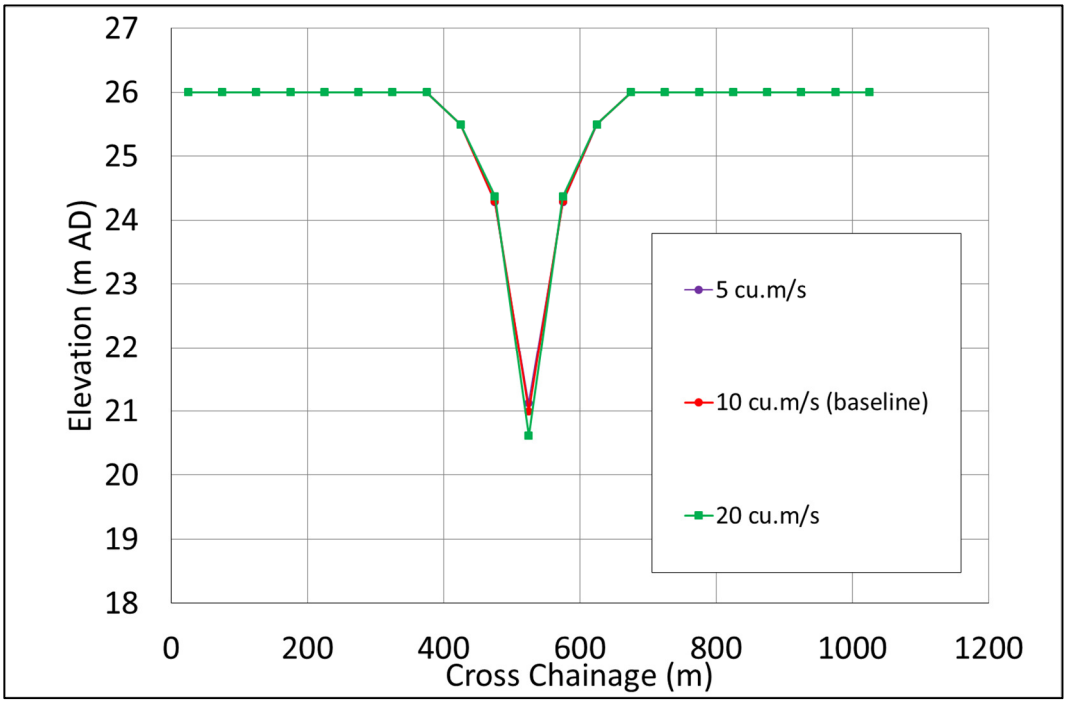

Figure 5.17: Cross section results at chainage 2,500m: sensitivity to fluvial inflow

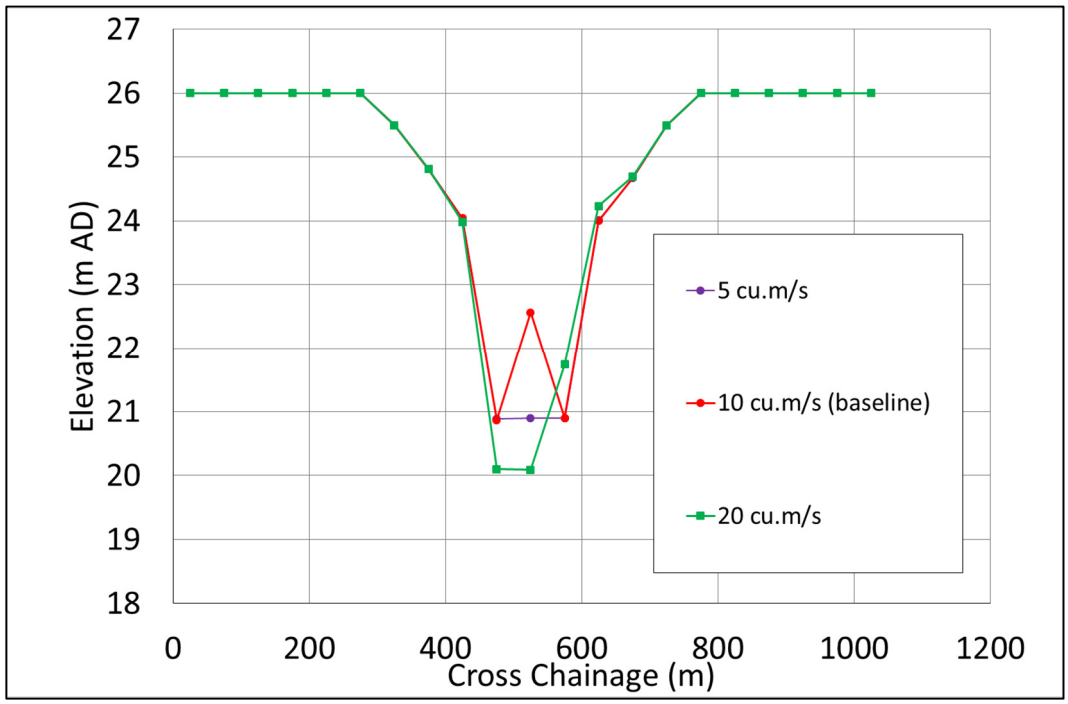

Figure 5.18: Cross section results at chainage 5,000m: sensitivity to fluvial inflow

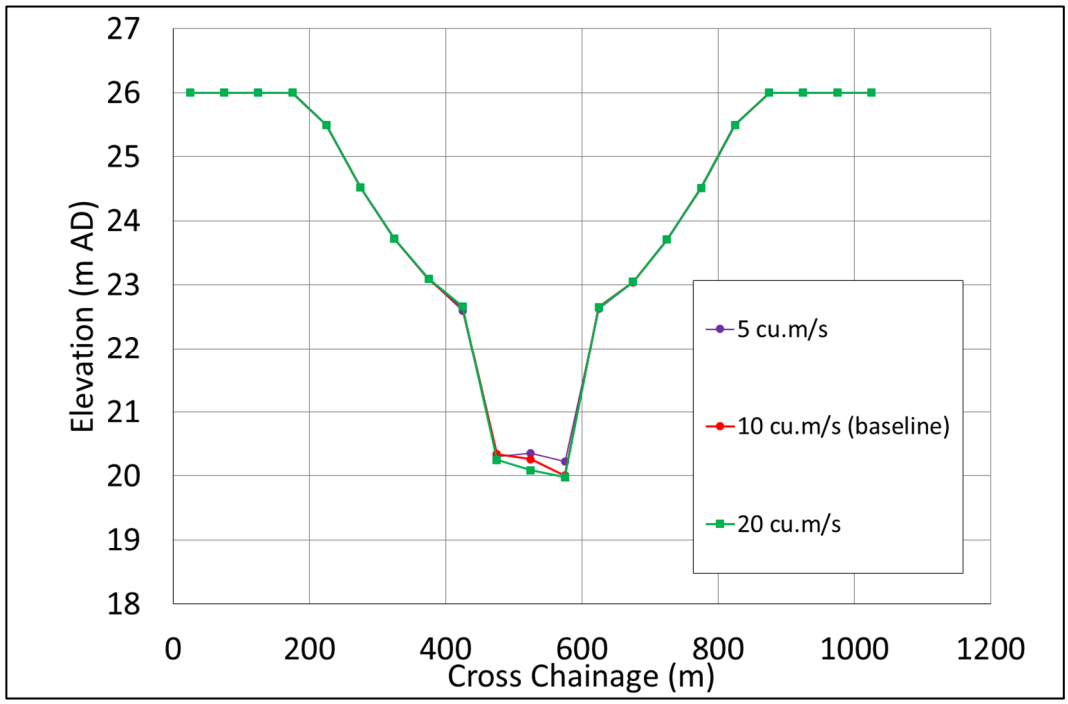

Figure 5.19: Cross section results at chainage 7,500m: sensitivity to fluvial inflow

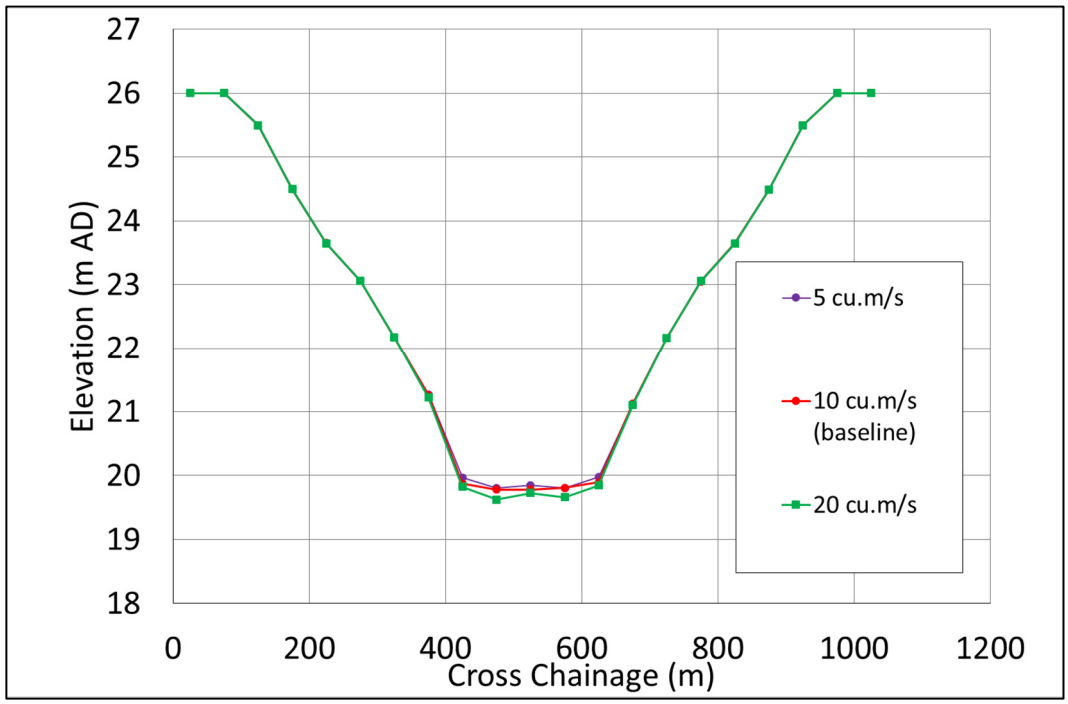

Figure 5.20: Cross section results at chainage 10,000m: sensitivity to fluvial inflow

5.4.2 Tidal Range

Sensitivity to the applied tidal range has been tested by performing simulations with the spring tidal range increased to 5m and reduced to 3m, from the baseline value of 4m. In each case the neap tidal range was set equal to half the spring tidal range, in common with the baseline simulation. Figure 5.21 shows the change in total sediment volume occurring during these simulations.

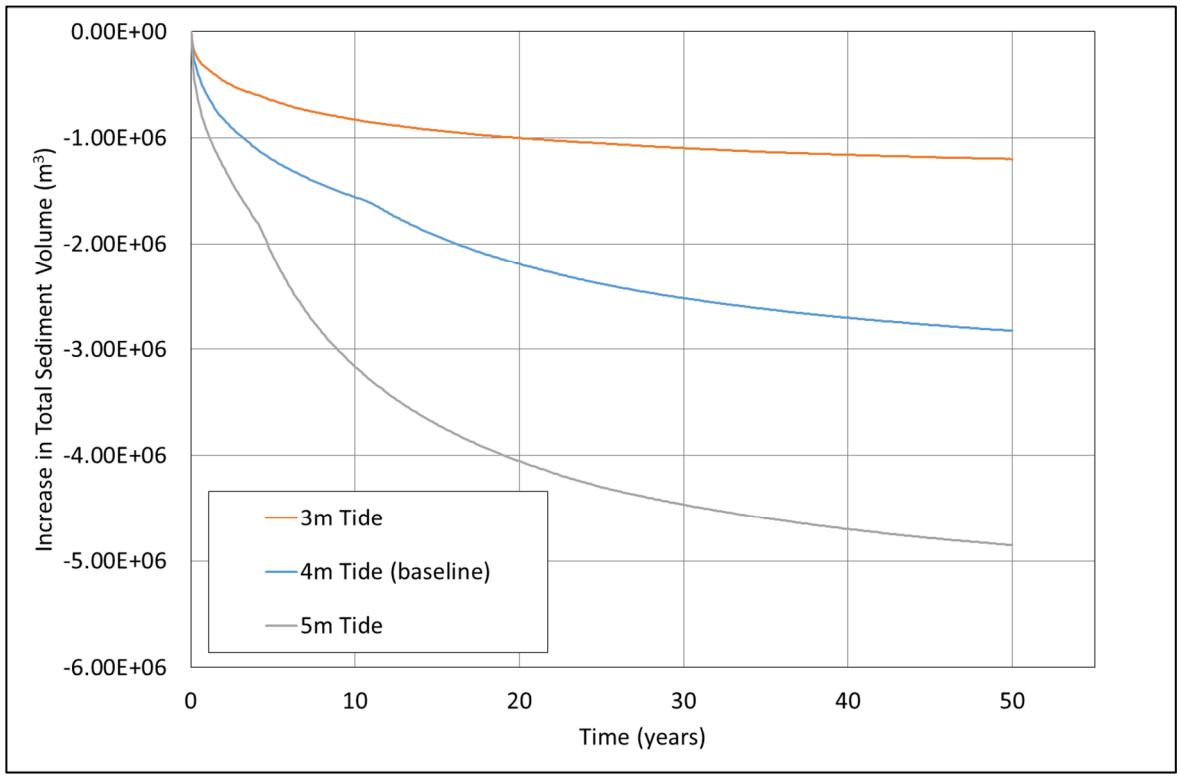

Figure 5.21: Change in total sediment volume: sensitivity to tidal range

Figure 5.21 shows that the rate of erosion is significantly influenced by the tidal range. The cross sections given in Figures 5.21 to 5.24 show that the increased erosion with increasing tidal range is concentrated in the channel and is greatest near the marine boundary, as expected due to the increased influence of tides in this area. It is noted that only relatively minor changes have occurred to bed levels in the inter-tidal areas, outside the channel.

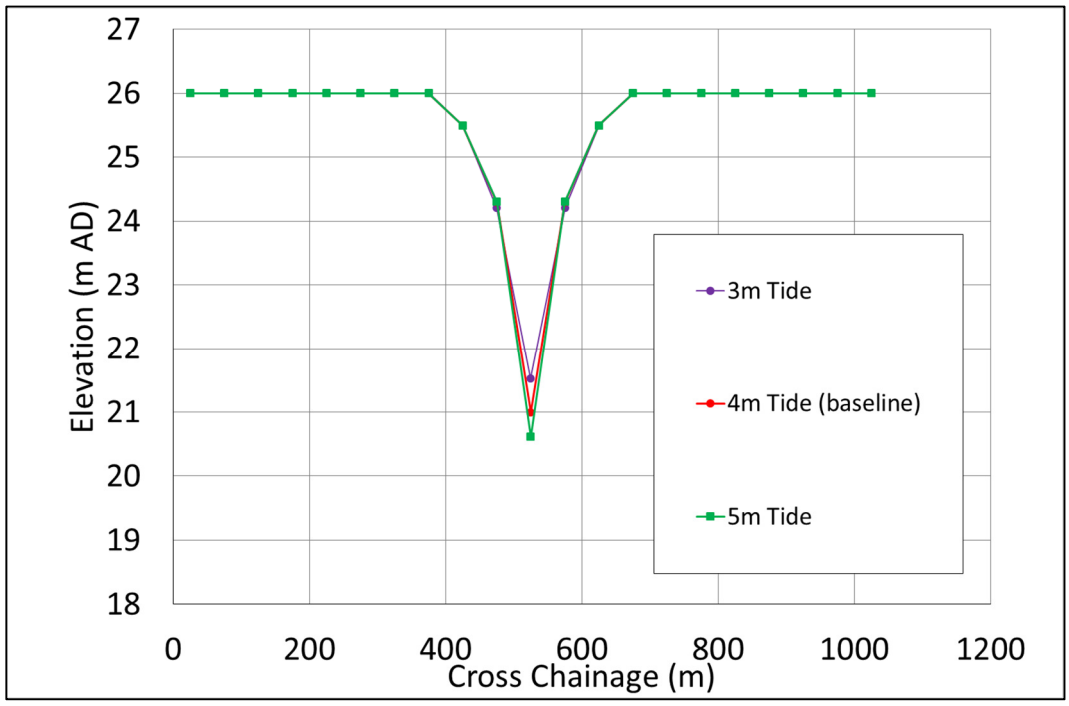

Figure 5.22: Cross section results at chainage 2,500m: sensitivity to tidal range

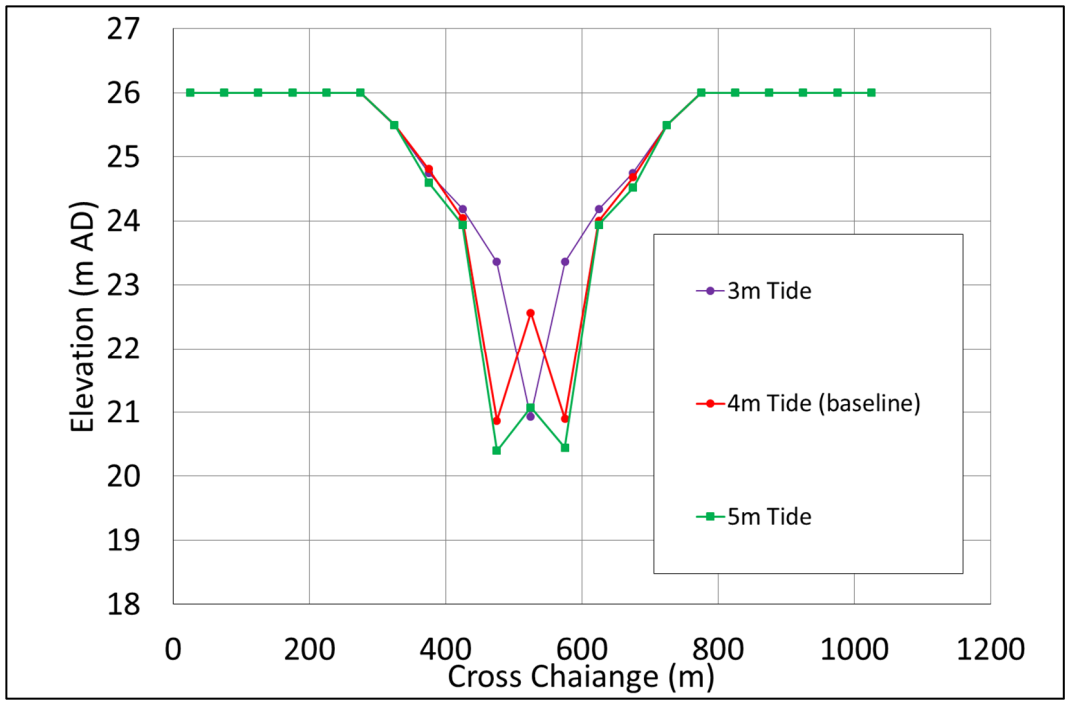

Figure 5.23: Cross section results at chainage 5,000m: sensitivity to tidal range

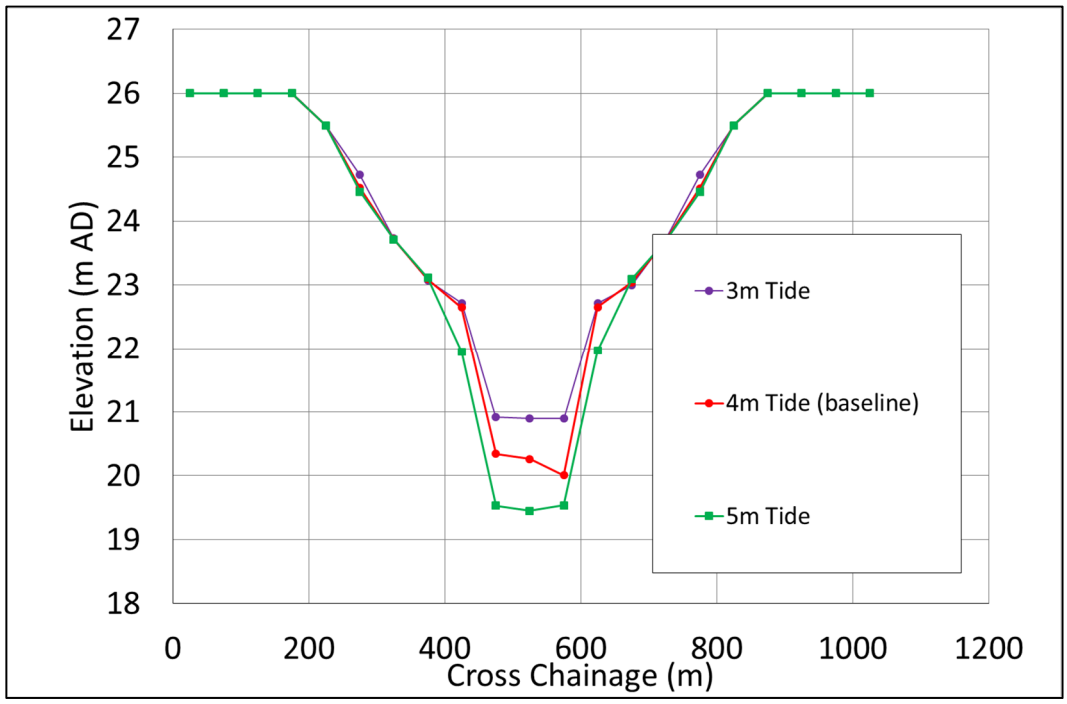

Figure 5.24: Cross section results at chainage 7,500m: sensitivity to tidal range

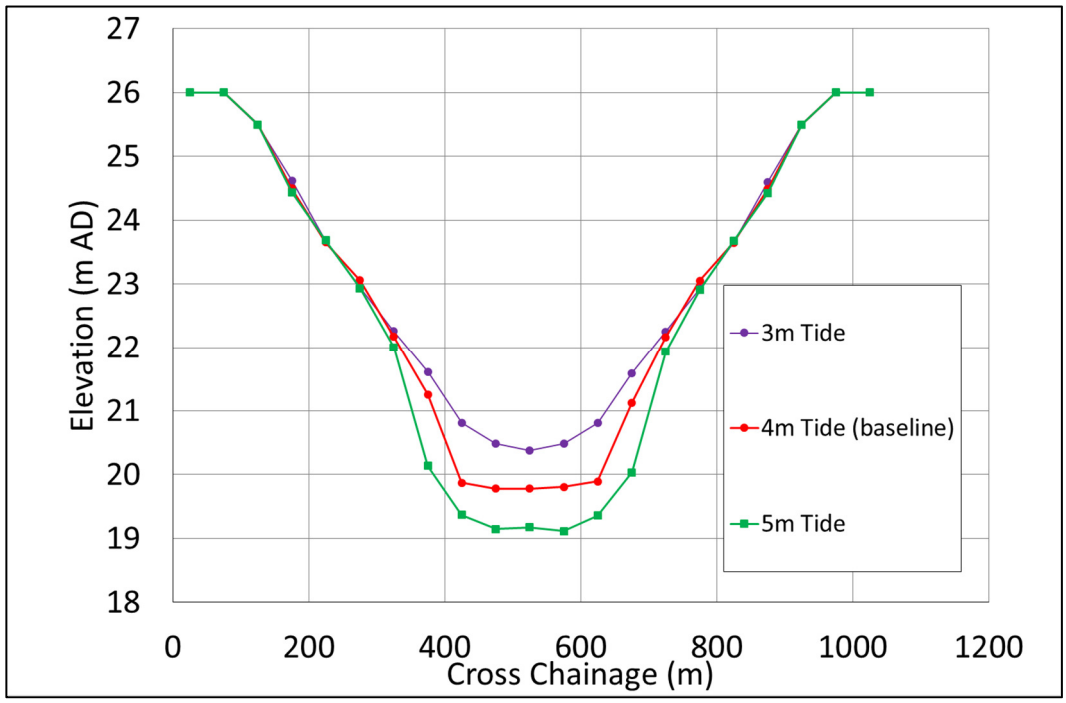

Figure 5.25: Cross section results at chainage 10,000m: sensitivity to tidal range

5.4.3 Waves

To assess the influence of waves on the model results the prevailing wind speed and the standard deviation of the wind speed have each been reduced to 2.5 m/s and increased to 7.5 m/s, from the baseline value of 5 m/s (applied to both the prevailing wind speed and the standard deviation of the wind speed for each test). The change in total sediment volume during these simulations is given in Figure 5.26.

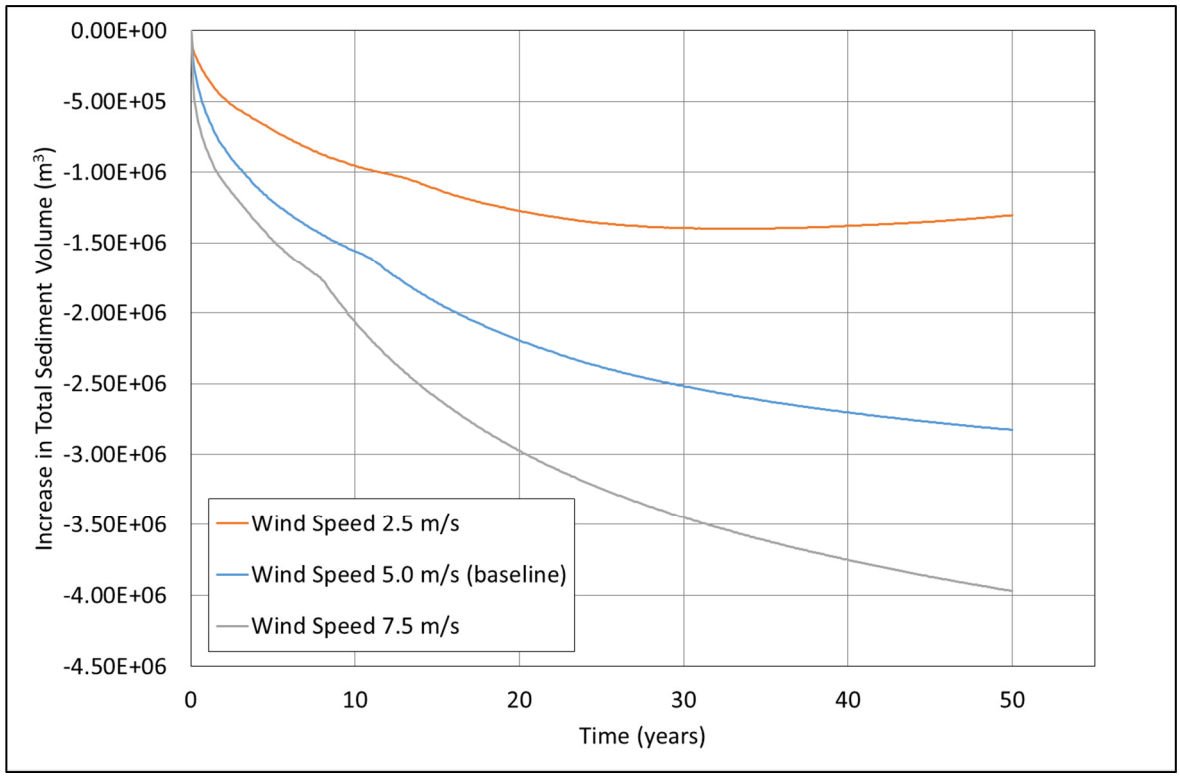

Figure 5.26: Change in total sediment volume: sensitivity to wave energy

Figure 5.26 shows the model results to be significantly influenced by the wave settings and confirms that the rate of erosion is greater for higher wind speeds (and hence higher wave energy). It is also noted that for the lowest wind speed setting of 2.5 m/s the total sediment volume has begun to increase towards the end of the simulation, indicating that the total rate of erosion has dropped below the total rate of sediment deposition within the model domain.

The cross sections given in Figures 5.26 to 5.29 show that the greatest effect from the change in wave energy occurs in the inter-tidal areas and that only minor changes to the

depth of the channel have occurred. This is the expected pattern of influence from wave energy, since the peak wave orbital velocity is inversely related to the water depth (Equation 4.40).

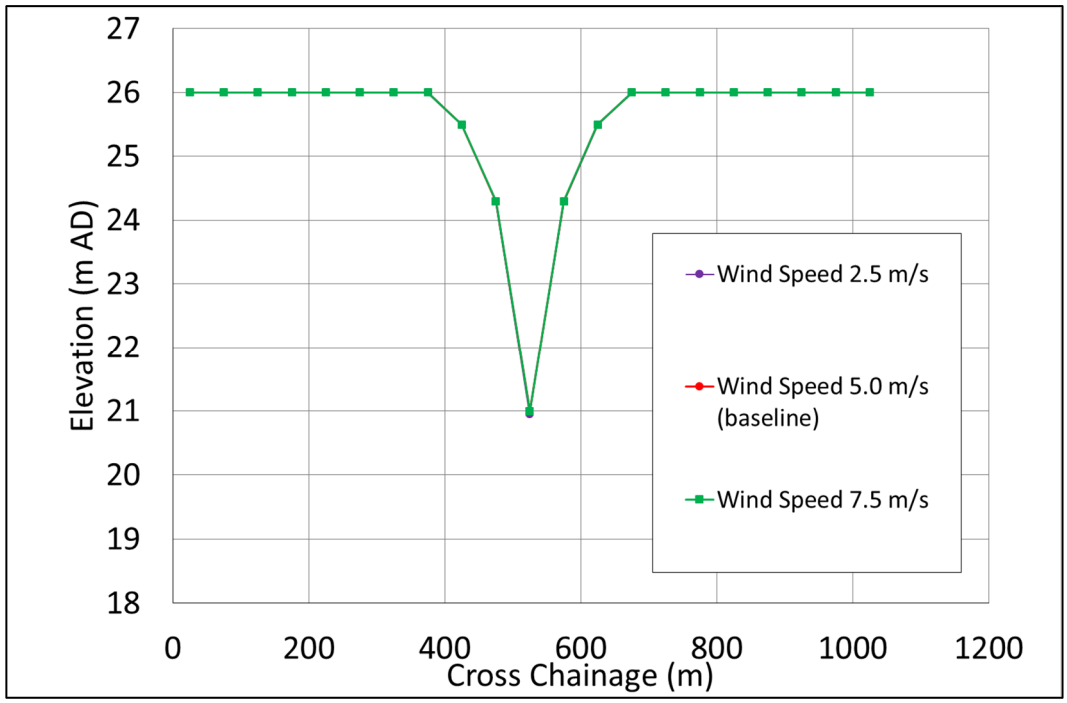

Figure 5.27: Cross section results at chainage 2,500m: sensitivity to wave energy

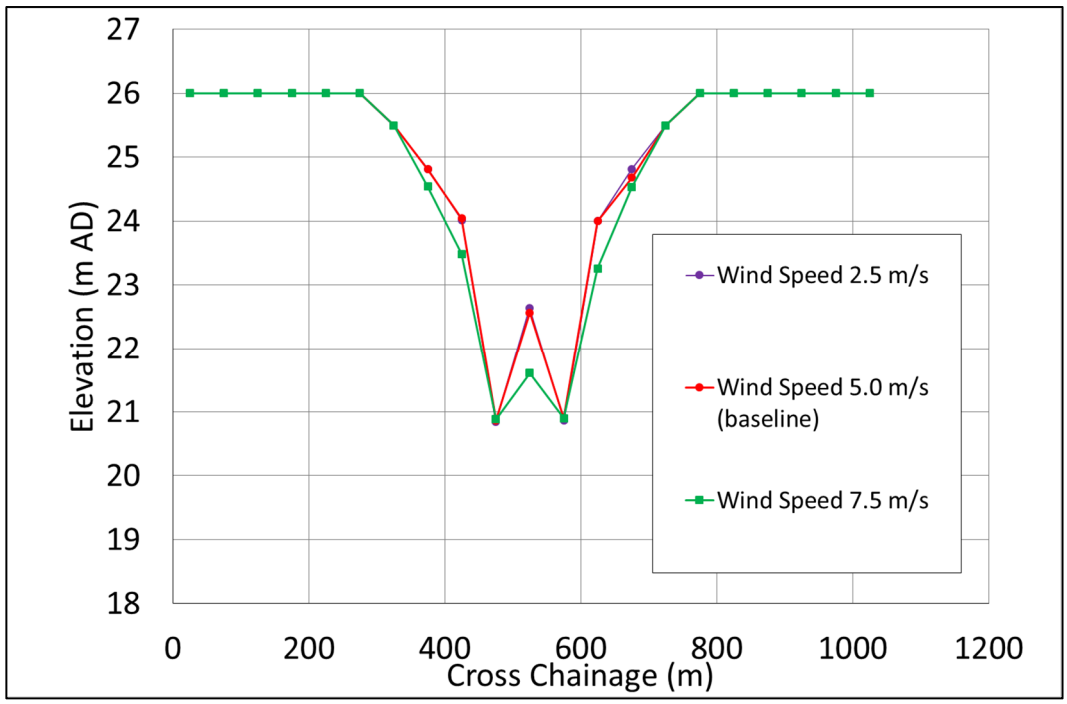

Figure 5.28: Cross section results at chainage 5,000m: sensitivity to wave energy

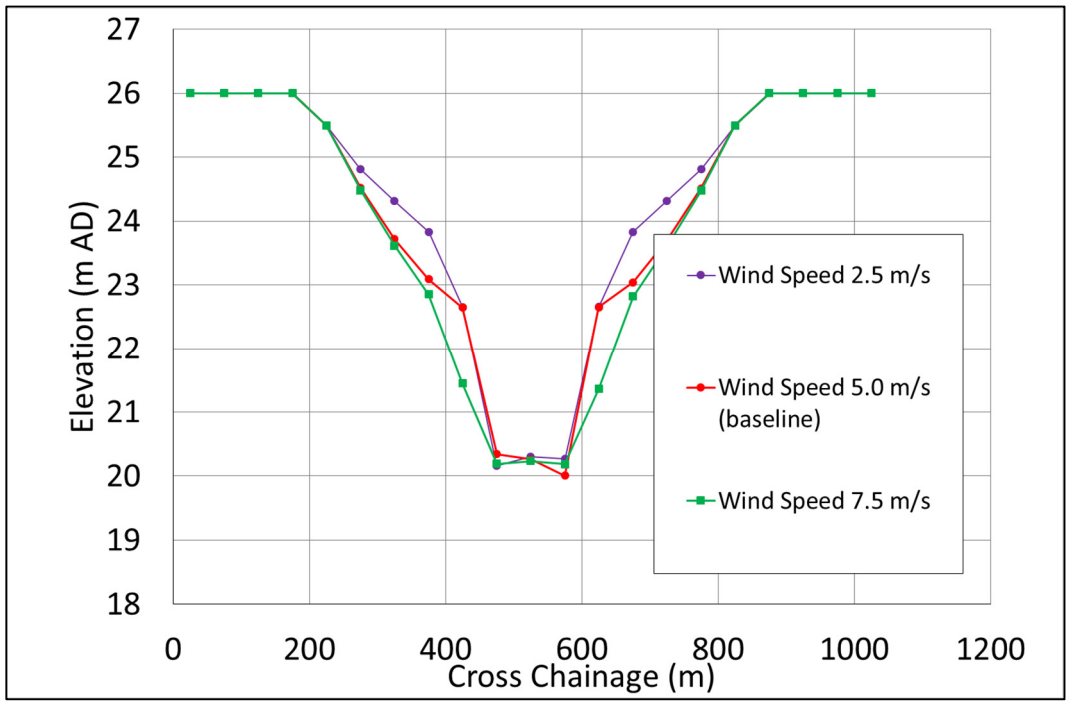

Figure 5.29: Cross section results at chainage 7,500m: sensitivity to wave energy

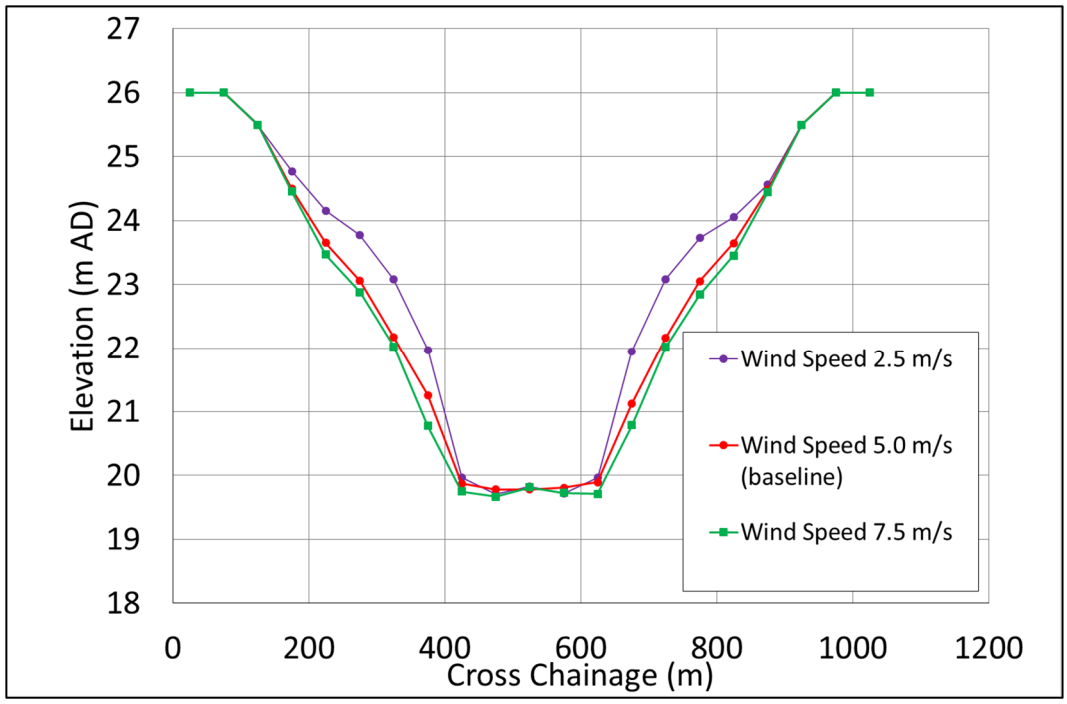

Figure 5.30: Cross section results at chainage 10,000m: sensitivity to wave energy

5.5 Initial Model Settings

It is expected that the morphology in each simulation will evolve towards an equilibrium state, depending only on the imposed hydrodynamic conditions, sediment inputs and geological constraints (minimum bed levels) and it is therefore anticipated that any changes applied to the initial conditions, relative to the baseline initial conditions, will be reduced over time. This has been tested using sensitivity analyses in which the initial bathymetry and sediment composition are varied from that in the baseline simulation.

5.5.1 Initial Bathymetry

Sensitivity to changes in the initial bathymetry was tested by varying the lowest initial bed level in the cross section at the estuary mouth (set to 18.0m AD in the baseline scenario as shown in Figure 5.1). This value has been increased to 19 m AD and reduced to 17 m AD from the baseline value of 18 m AD. Bed levels for the cells on the left and right edges of this cross section were left unchanged at 25.5 m AD and bed levels for the intermediate cells interpolated between this value and the new minimum bed level. Similarly the minimum bed level at the upstream end of the model domain was unchanged, with the bed levels for the intermediate cells re-interpolated between these values, over the length of the estuary. Figure 5.31 shows the change in total sediment volume occurring during these simulations. In this case the volumes are given relative to the baseline initial sediment volume.

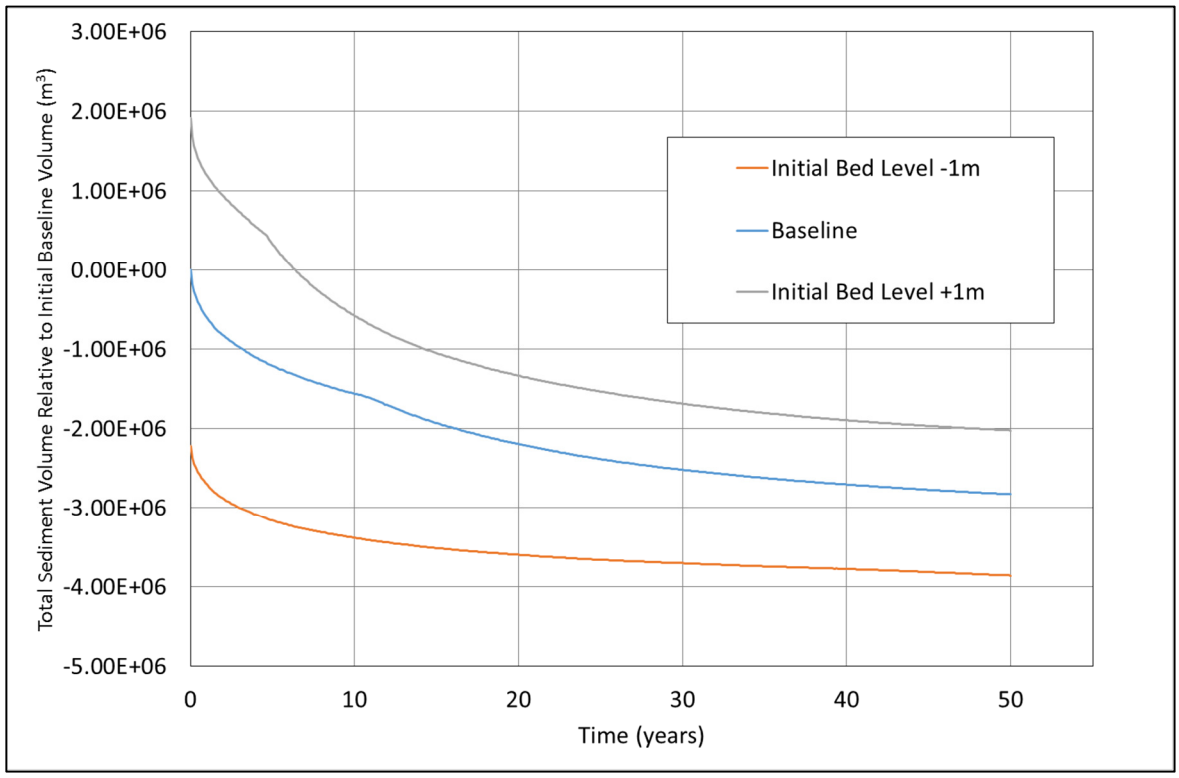

Figure 5.31: Change in total sediment volume relative to the initial sediment volume in the baseline scenario: sensitivity to initial bed levels

Figure 5.31 shows that the difference in total sediment volume is reduced over time as expected, although this figure suggests that a much longer simulation period would be required for the sediment volumes in the three simulations to converge.

Figures 5.32 to 5.35 show the cross section results for these simulations together with the initial bed levels (shown as grey dashed lines). These figures show that bed levels in the channel have largely equalised by the end of the simulations, while much of the original difference in bed levels in the inter-tidal areas has remained. It is noted that the overall bed level changes in the channel are generally larger than those in the inter-tidal areas and this faster adjustment to the hydrodynamic conditions, due to the greater tidal velocities, may explain the greater convergence of bed levels in these areas.

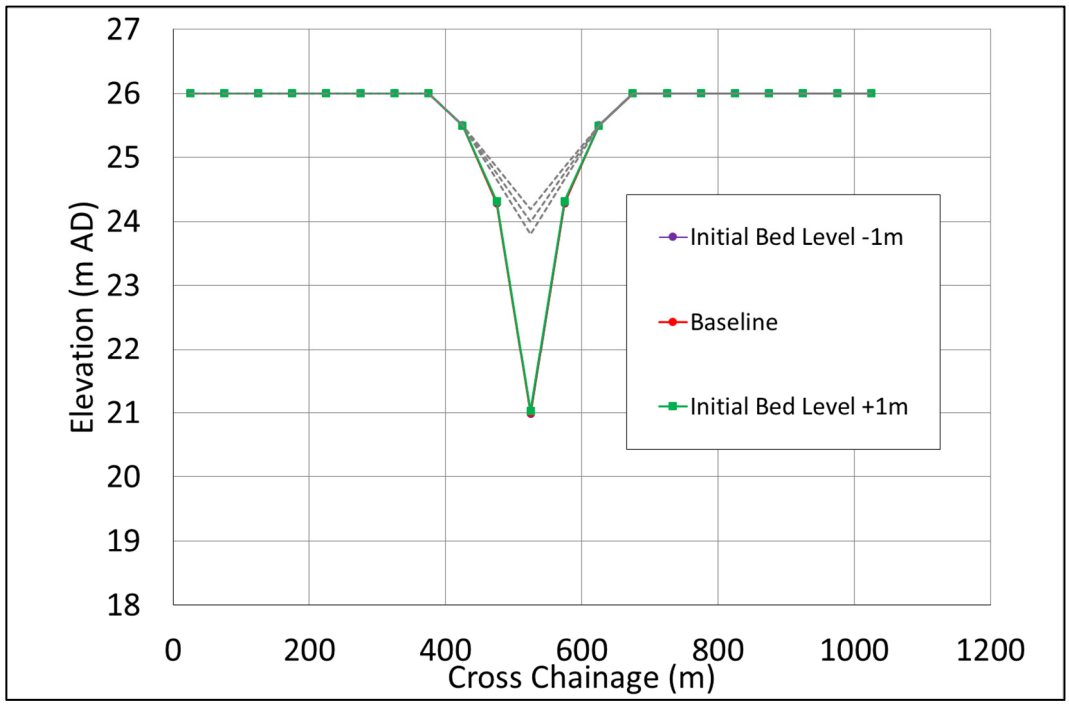

Figure 5.32: Cross section results at chainage 2,500m: sensitivity to initial bed levels

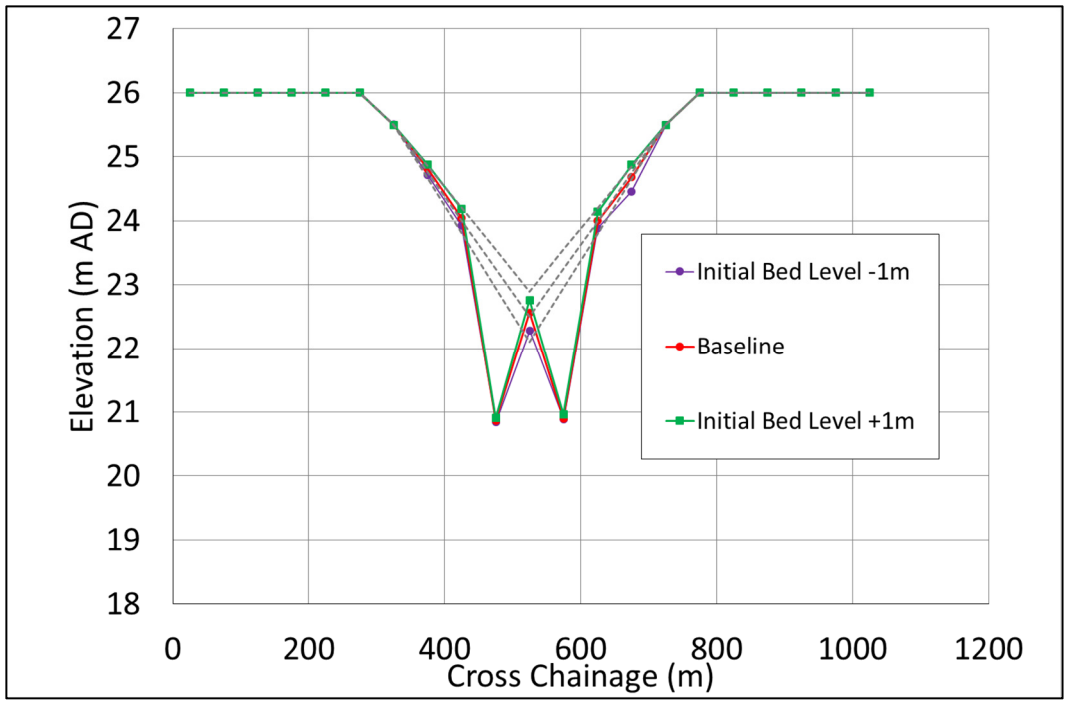

Figure 5.33: Cross section results at chainage 5,000m: sensitivity to initial bed levels

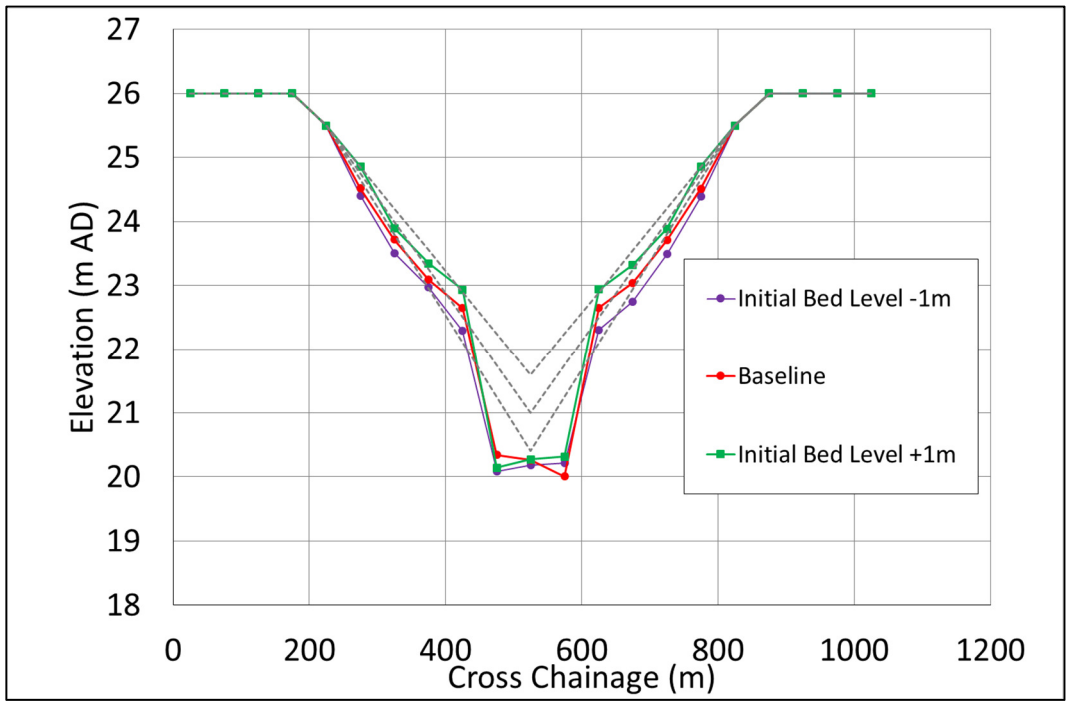

Figure 5.34: Cross section results at chainage 7,500m: sensitivity to initial bed levels

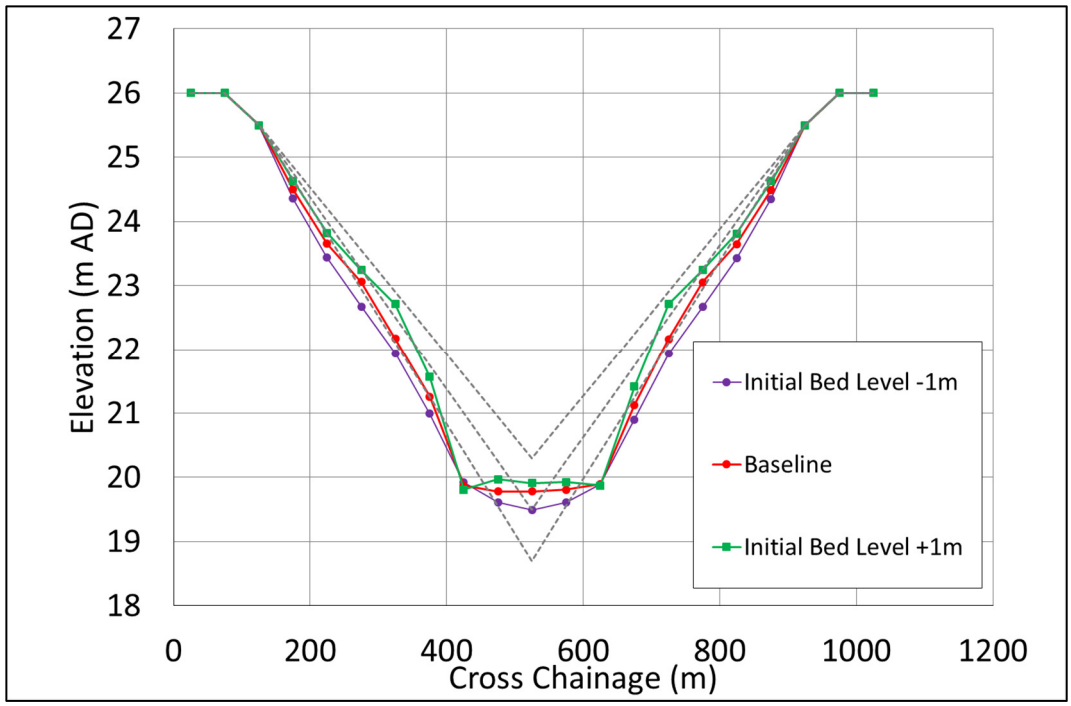

Figure 5.35: Cross section results at chainage 10,000m: sensitivity to initial bed levels

5.5.2 Initial Sediment Composition

The initial sediment composition for the baseline scenario is composed of 50% sand and 50% mud for all cells and sediment layers. To assess the effect of this setting on the final morphology, simulations have been carried with the initial sediment composition set to 100% sand and to 100% mud. The change in total sediment volume during these simulations in shown in Figure 5.36

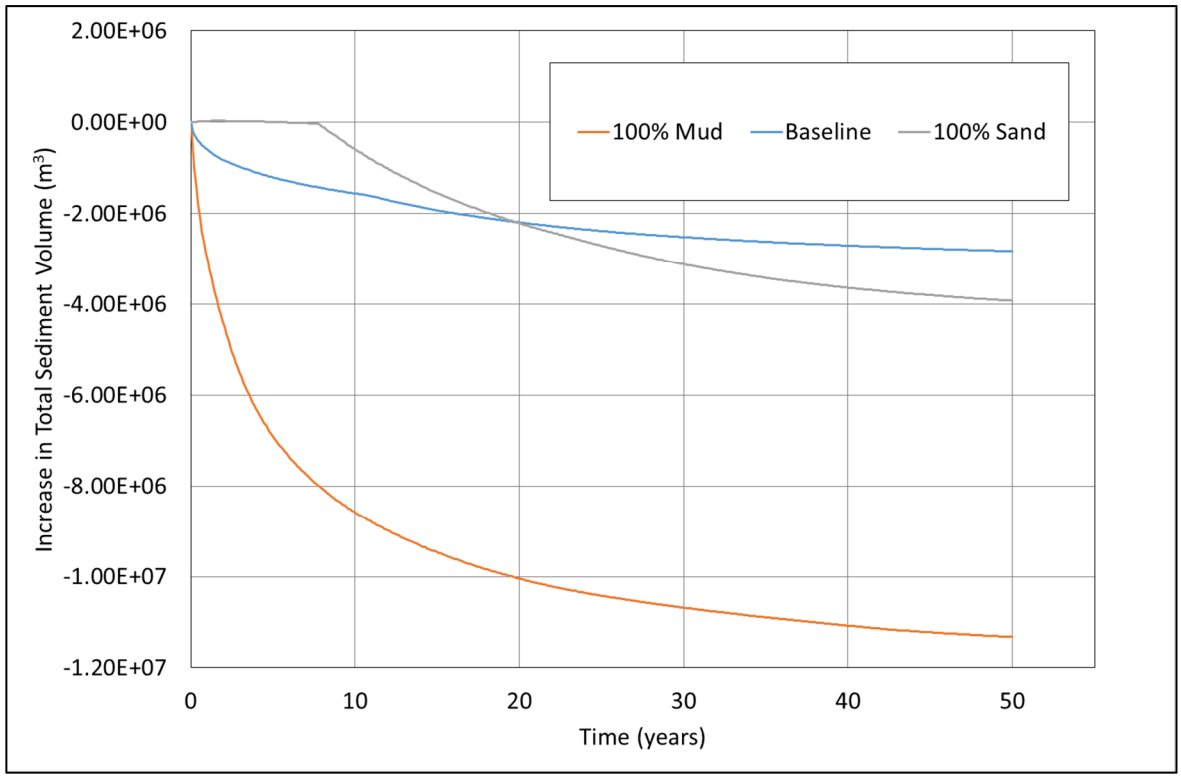

Figure 5.36: Change in total sediment volume: sensitivity to initial sediment composition

Figure 5.36 shows that the rate of erosion is significantly higher when the initial sediment is composed of 100% mud, which is expected due to the lower critical shear stress for the erosion of mud type sediment. Towards the end of the simulation the erosion in the simulation for 100% sand has also exceeded that in the baseline simulation. This is also consistent with the model formulation, which applies a higher critical shear stress to some mixtures of sand and mud than to sediment composed entirely of sand (peak critical shear stress occurs for mixtures containing 20% mud).

The cross section results given in Figures 5.37 to 5.40 confirm that increased erosion has occurred for both the mud only and the sand only simulations; however, some increased deposition has also occurred, in the inner estuary, for the mud only simulation. This is consistent with a higher local sediment concentration occurring due to the increased erosion in the channel.

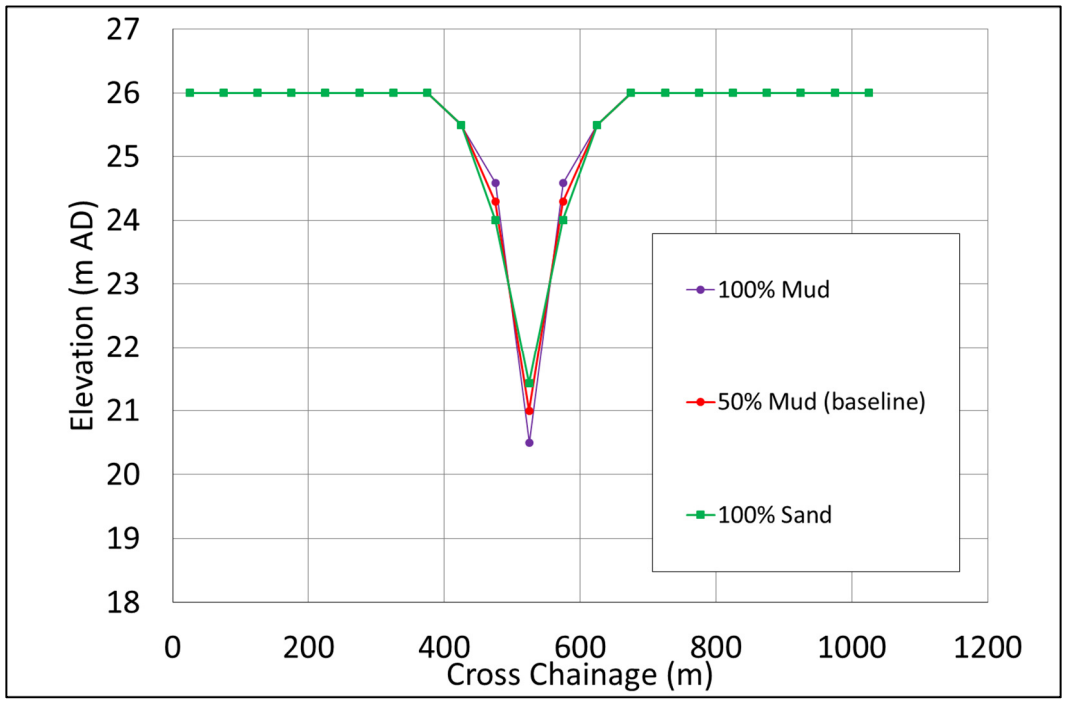

Figure 5.37: Cross section results at chainage 2,500m: sensitivity to initial bed levels

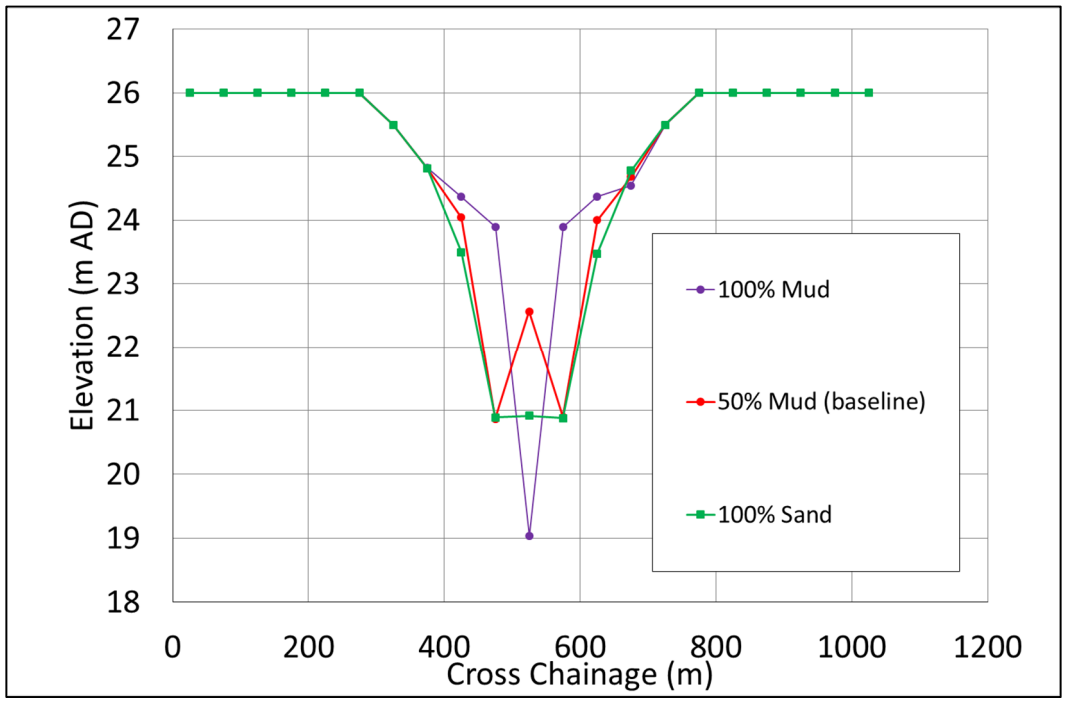

Figure 5.38: Cross section results at chainage 5,000m: sensitivity to initial bed levels

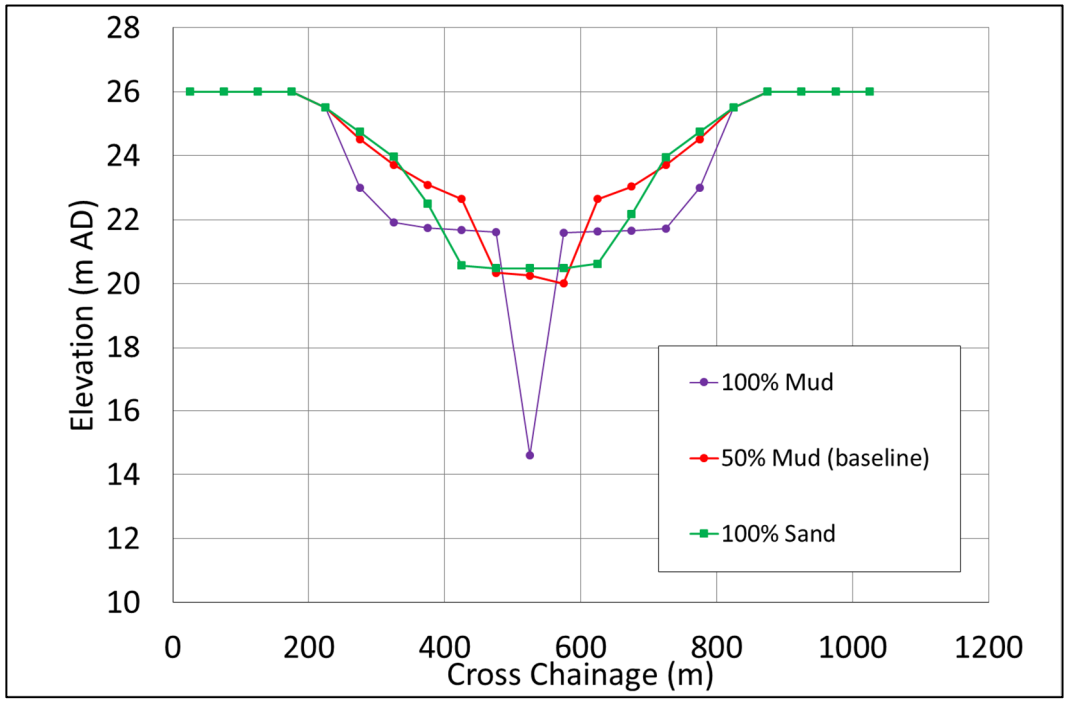

Figure 5.39: Cross section results at chainage 7,500m: sensitivity to initial bed levels

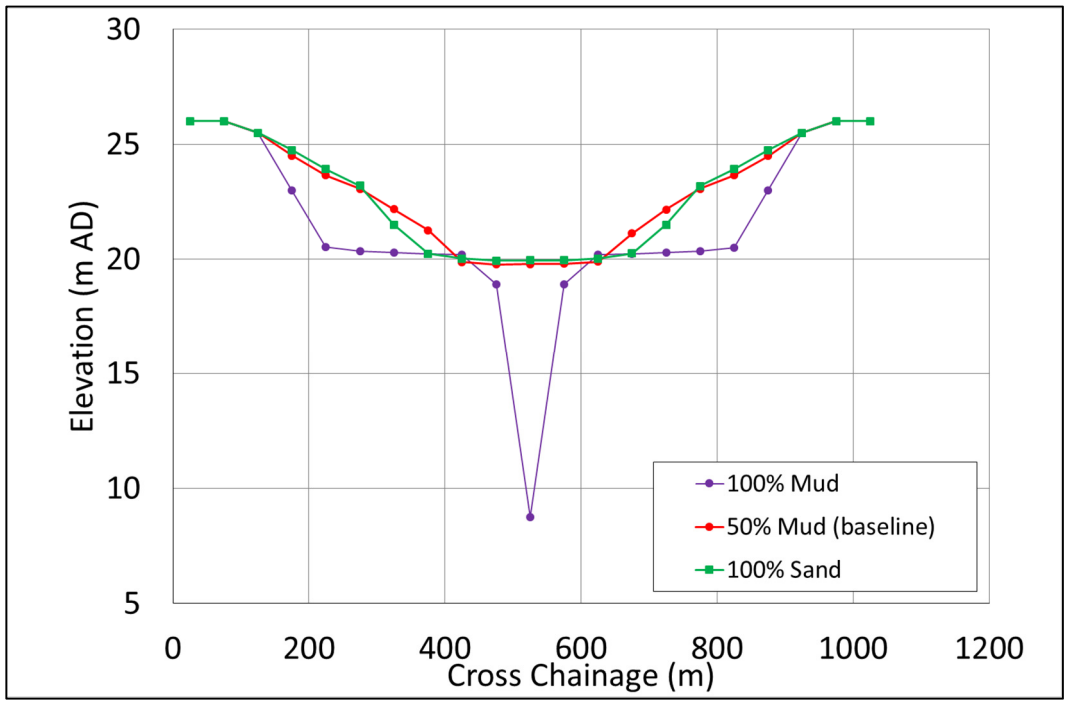

Figure 5.40: Cross section results at chainage 10,000m: sensitivity to initial bed levels

5.6 Boundary Conditions

5.6.1 Minimum Bed Levels

Minimum bed levels are set for each simulation to represent hard non-erodible geology and to define the space within which sediment movements can occur. The minimum bed levels in the baseline simulation are as defined in Section 5.1 and illustrated in Figure 5.1 for the estuary mouth. To assess the impact of these minimum levels on the results a simulation has been carried out with the minimum bed levels reduced to 5m AD for the majority of the cells. In this simulation the minimum bed levels for cells along the boundary is unchanged from the baseline values, creating a rectangular 'bathtub' in which sediment movements can occur. Near the upstream end of the model the minimum bed levels are increased steeply, over around 20 cells (1000m), so that the minimum levels along the upstream boundary match those in the baseline simulation. The change in total sediment volume during this simulation is given in Figure 5.41.

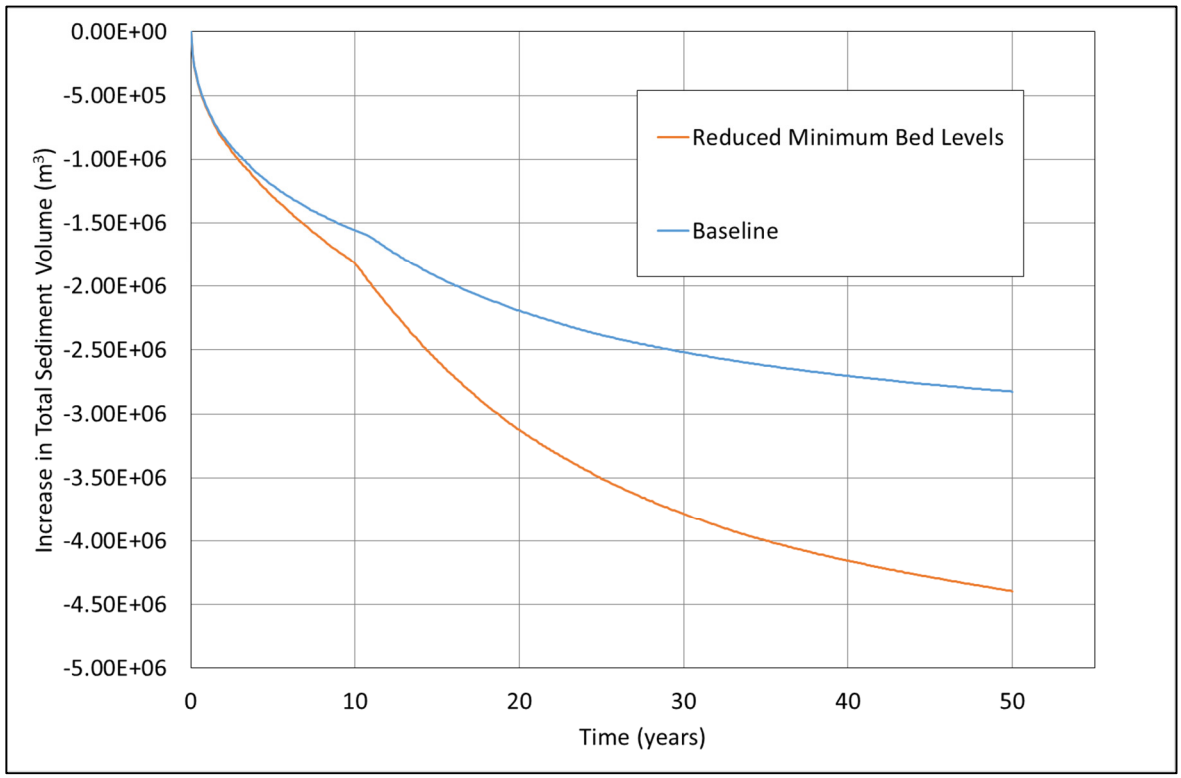

Figure 5.41: Change in total sediment volume: sensitivity to minimum bed levels

Figure 5.41 shows that the minimum bed levels do have a significant influence on the rate of erosion occurring during the baseline simulation. Cross section results are given in Figures 5.42 to 5.46 .

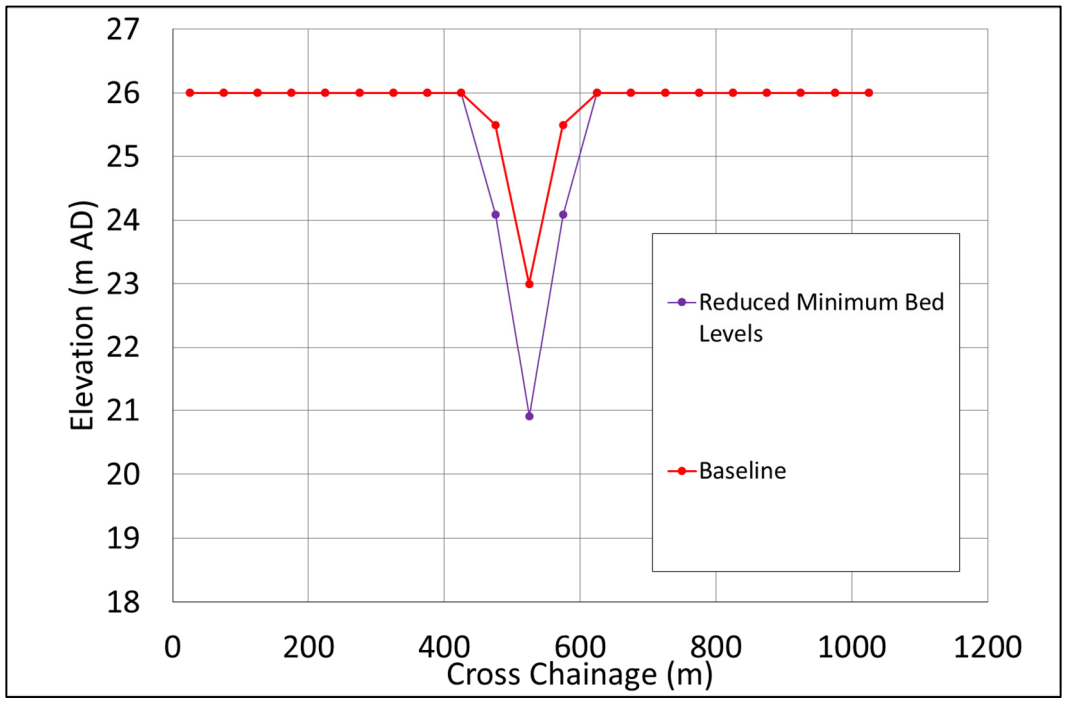

Figure 5.42: Cross section results at chainage 1,250m: sensitivity to minimum bed levels

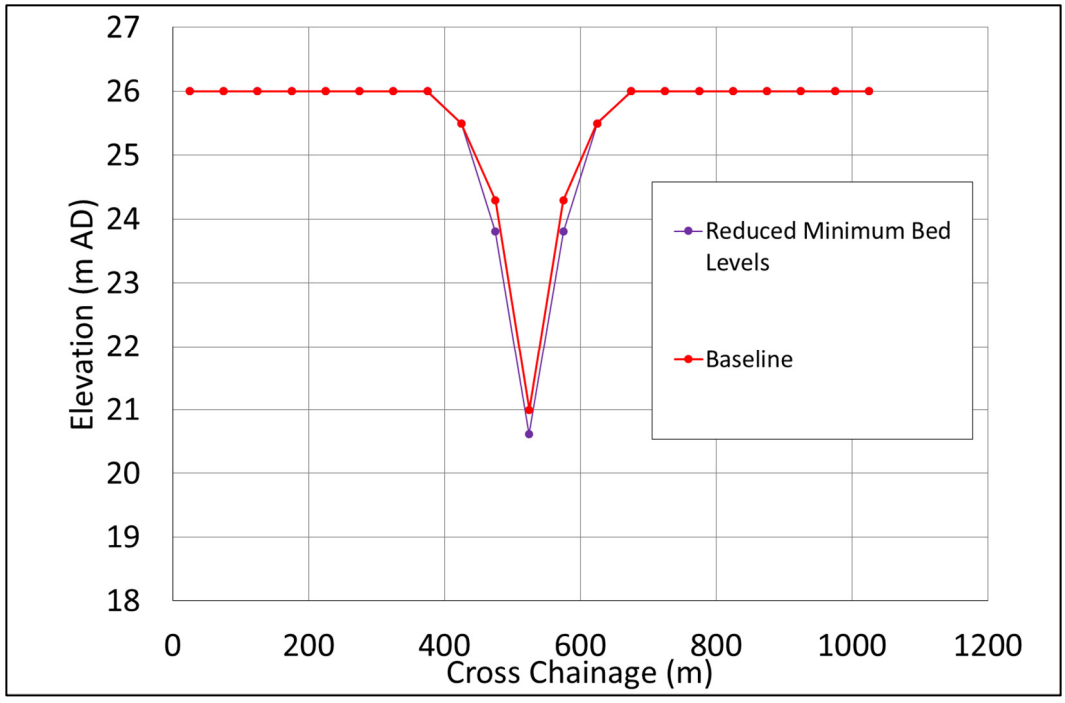

Figure 5.43: Cross section results at chainage 2,500m: sensitivity to minimum bed levels

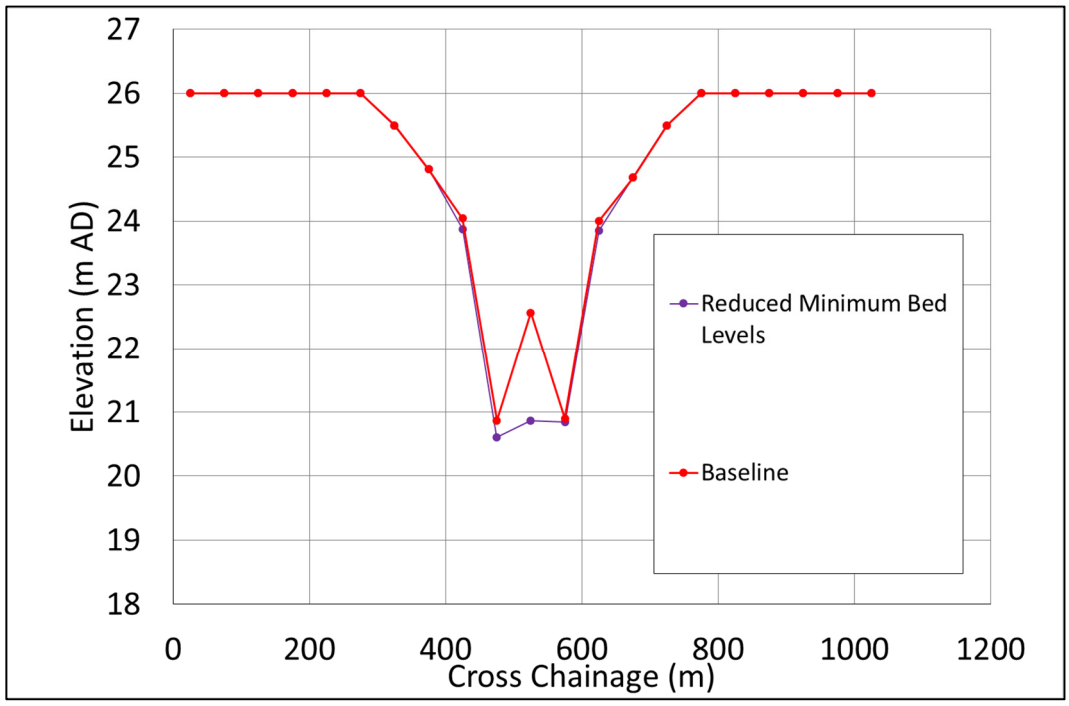

Figure 5.44: Cross section results at chainage 5,000m: sensitivity to minimum bed levels

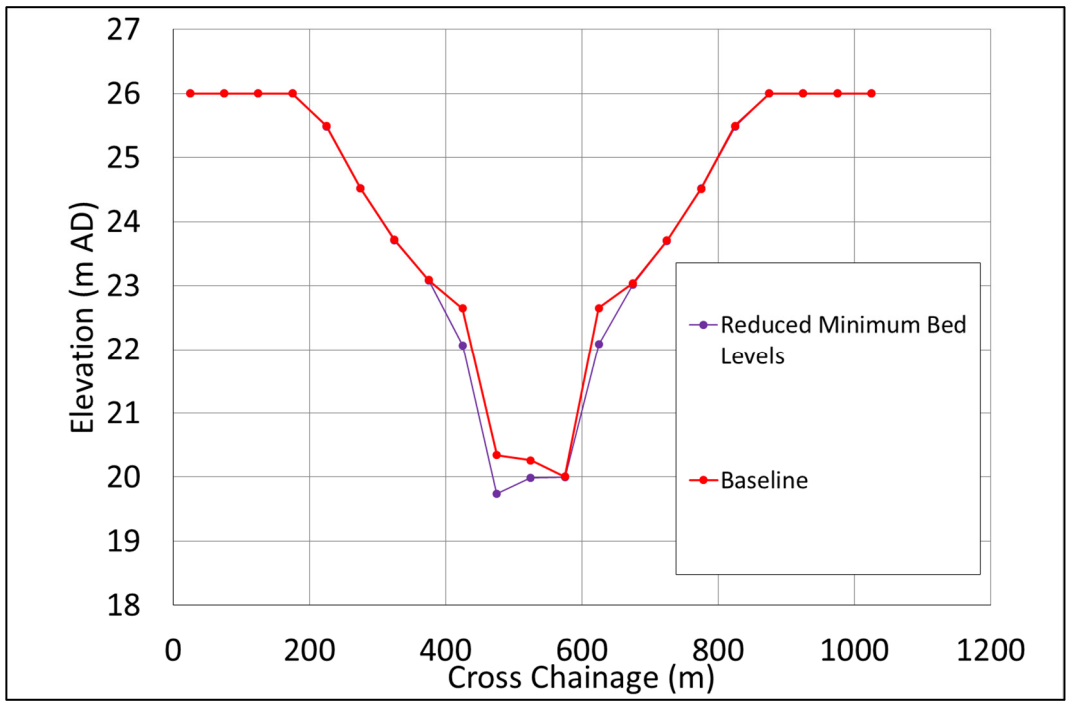

Figure 5.45: Cross section results at chainage 7,500m: sensitivity to minimum bed levels

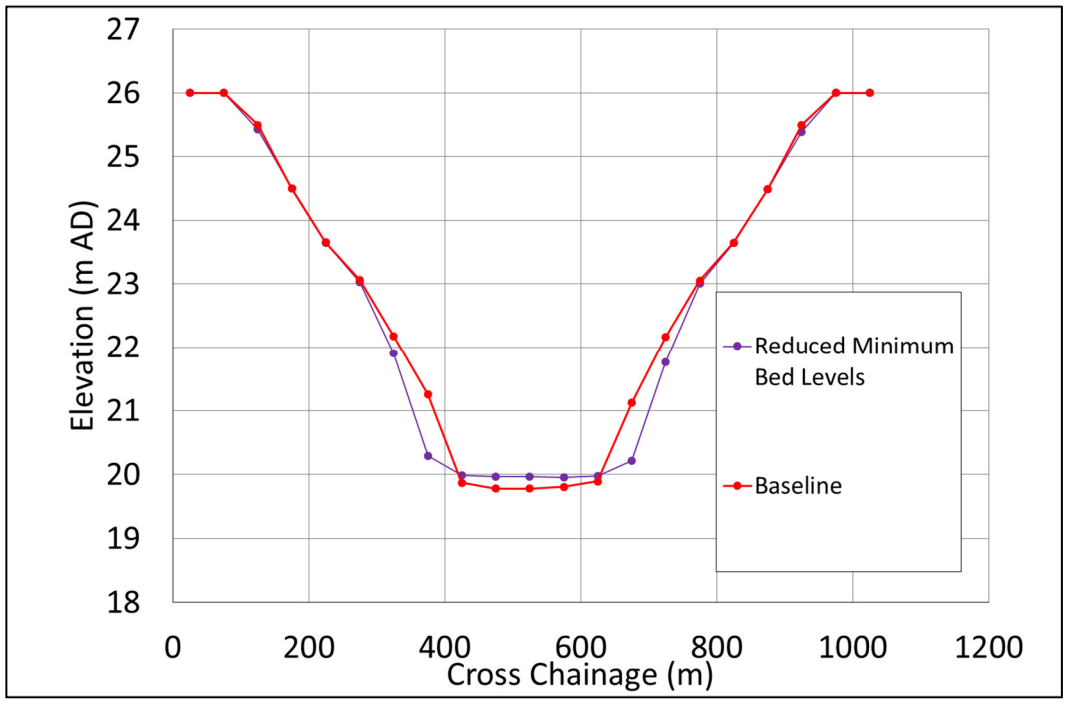

Figure 5.46: Cross section results at chainage 10,000m: sensitivity to minimum bed levels

From Figures 5.42 to 5.46 it can be seen that the width and the depth of the channel is affected at all cross sections, even though erosion has only been directly limited by the minimum bed levels near the upstream end of the model domain, in the baseline simulation. This can be explained by the increased tidal prism due to the increased channel dimensions near the upstream end of the model and the consequent increase in tidal flows.

5.6.2 Sand Input at the Marine Boundary

Sand is added to cells on the marine boundary at a specified constant rate, at each time-step, in proportion to the calculated sand transport rate (calculated for a sediment composition of 100% sand) or, if the total calculated sand transport rate along the boundary is zero, in proportion to the bed shear stress. The sensitivity of the model results to the specified rate of sand import has been tested by carrying out model runs with the sand import rate reduced to zero and increased by a factor of 10, to 5 x 10^{-4} m^3/s. Figure 5.47 shows the change in total sediment volume during these simulations.

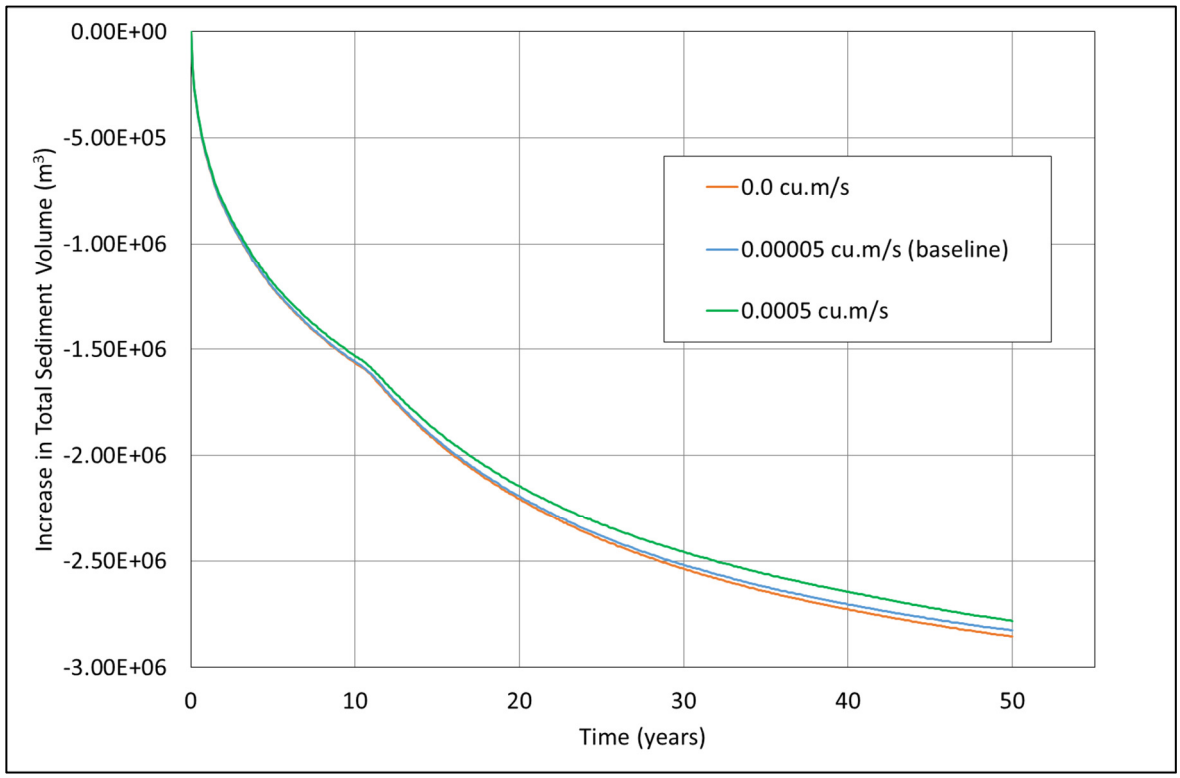

Figure 5.47: Change in total sediment volume: sensitivity to sand input rate

Figure 5.47 shows that changes in the rate of sand import have only a minor effect on the rates of erosion and deposition within the estuary. It is expected that increasing the rate at which sand is added to the boundary cells will reduce water depth, and hence increase the flow velocity, until the time-averaged net transport rate away from the boundary (both upstream and downstream) is equal to the rate at which sediment is added. Figures 5.48 and 5.49 show the cross section results at chainage 10,000m and at the boundary (chainage 12,500m). Results further upstream (e.g. at chainages 2,500m, 5,000m and 7,500m) did not show any significant difference from the baseline results.

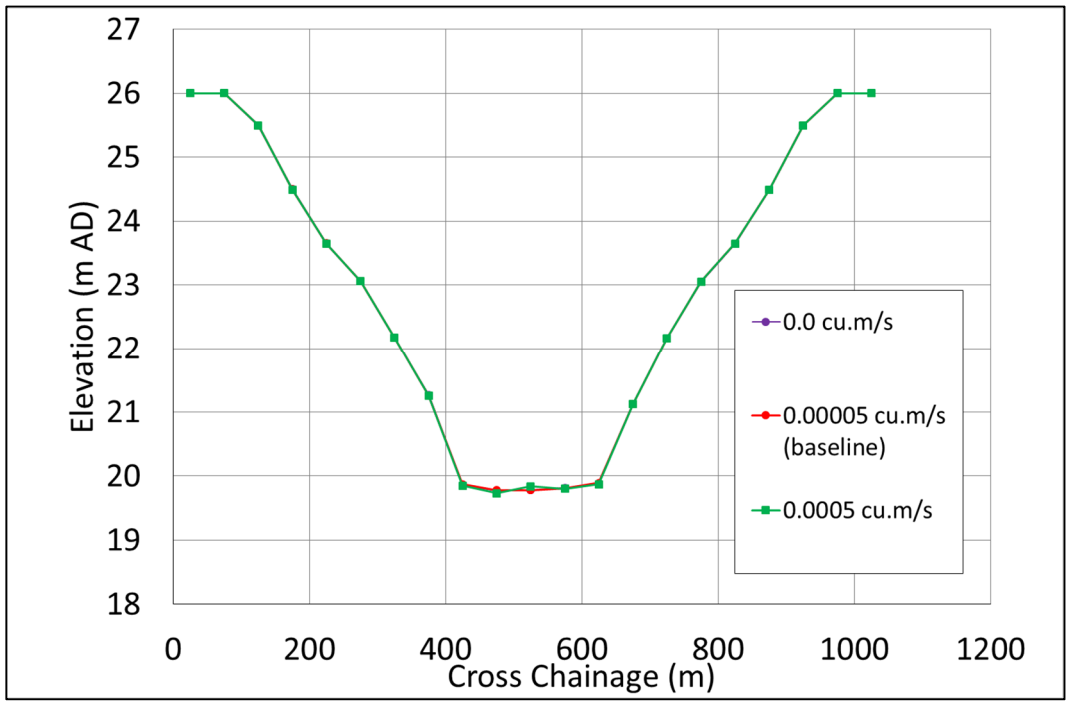

Figure 5.48: Cross section results at chainage 10,000m: sensitivity to sand input rate

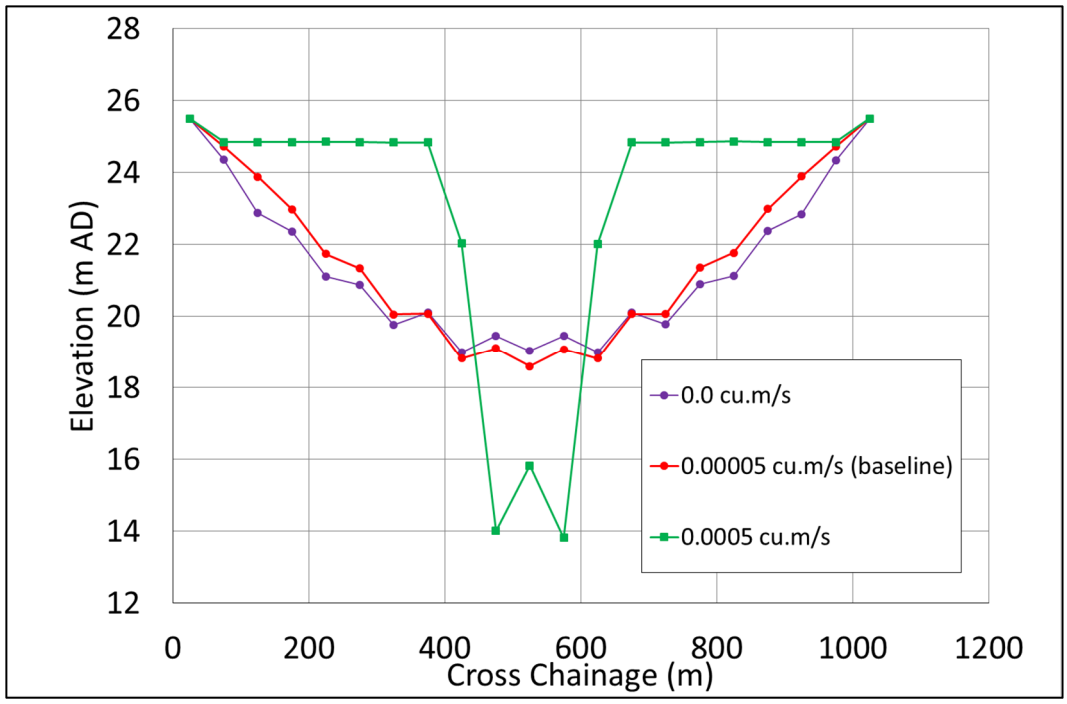

Figure 5.49: Cross section results at chainage 12,500m: sensitivity to sand input rate

Figures 5.48 and 5.49 show that the main effect on the bathymetry is along the boundary itself, with almost no effect occurring at chainage 10,000m. When the sand input rate is increased, a barrier is formed with a restricted inlet. This is similar to the observed

behaviour in wave dominated estuaries, where sand is introduced at the mouth by wave driven littoral sediment transport.

5.6.3 Sensitivity to Boundary Sediment Concentration

The suspended sediment concentration at the boundary directly influences the sediment concentration and hence the deposition rate within individual cells. Model runs have been carried out with the boundary sediment concentration increased or decreased by a factor of 10 to assess the sensitivity of the model results to this parameter. The change in total sediment volume during these model runs is given in Figure 5.49.

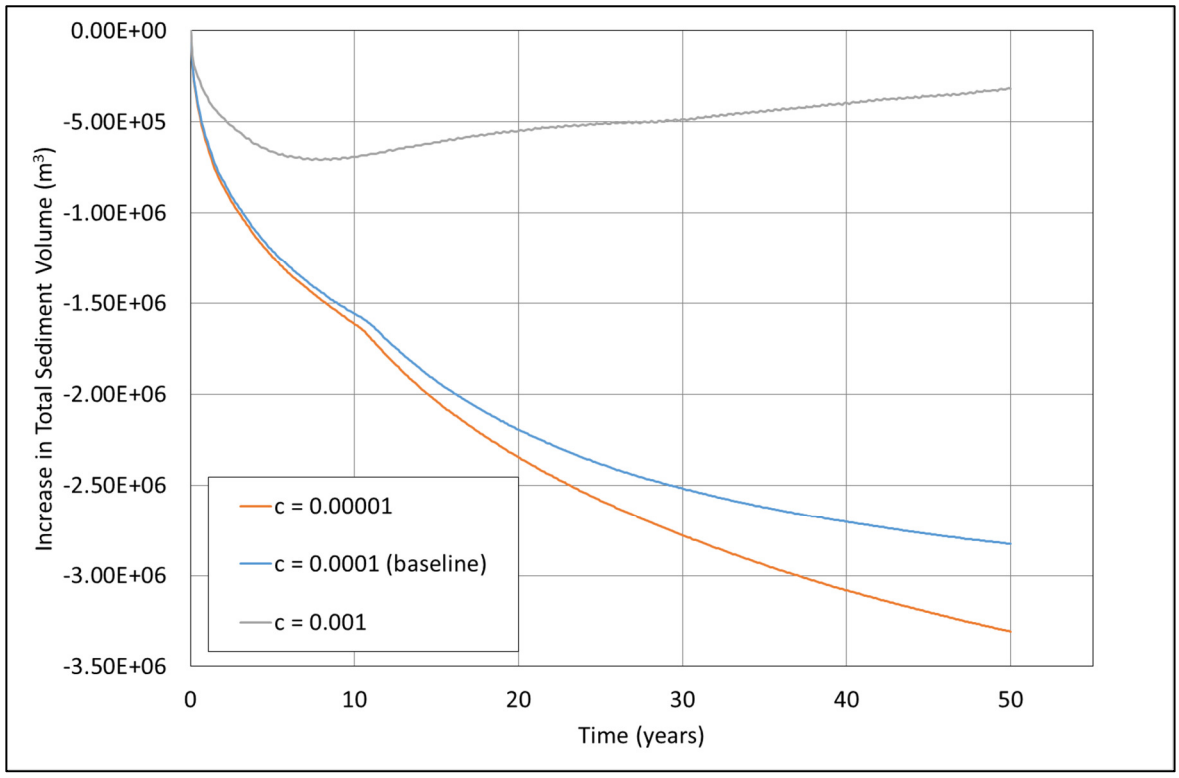

Figure 5.50: Change in total sediment volume: sensitivity to boundary sediment concentration

Figure 5.50 shows that increasing the boundary sediment concentration has a significant effect on the sediment volume, causing volumes to increase following an initial period of erosion. Reducing the boundary concentration has a smaller effect, causing increased

erosion and/or reduced deposition as expected. Final cross section profiles for these simulations are given in Figures 5.51 to 5.54 .

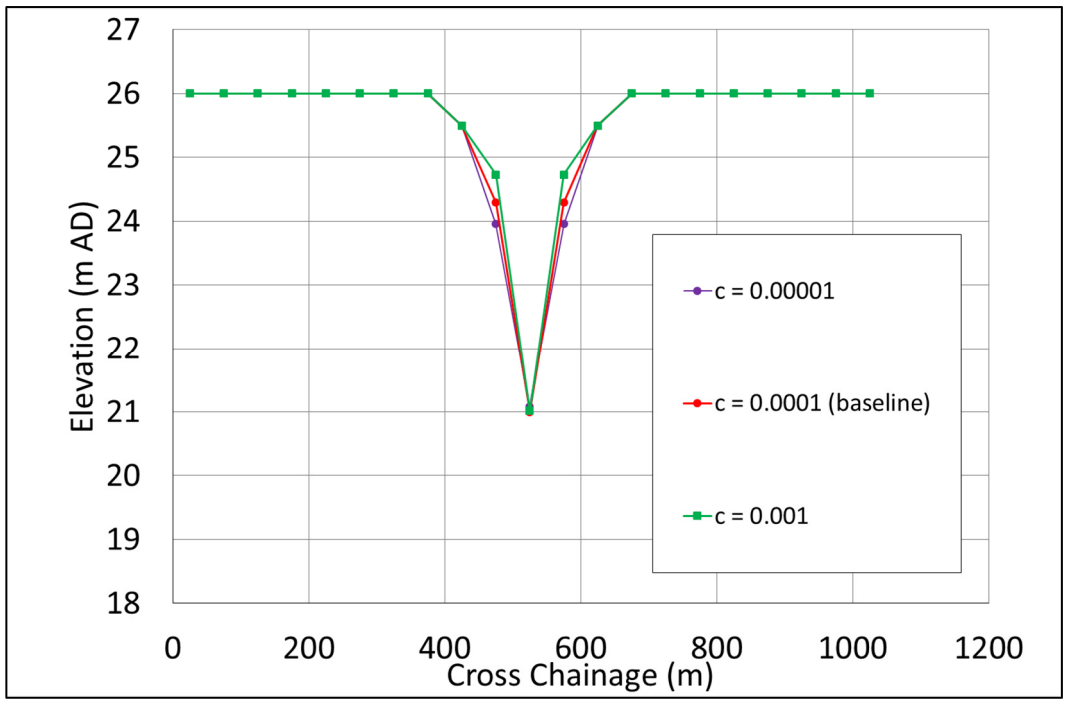

Figure 5.51: Cross section results at chainage 2,500m: sensitivity to boundary sediment input

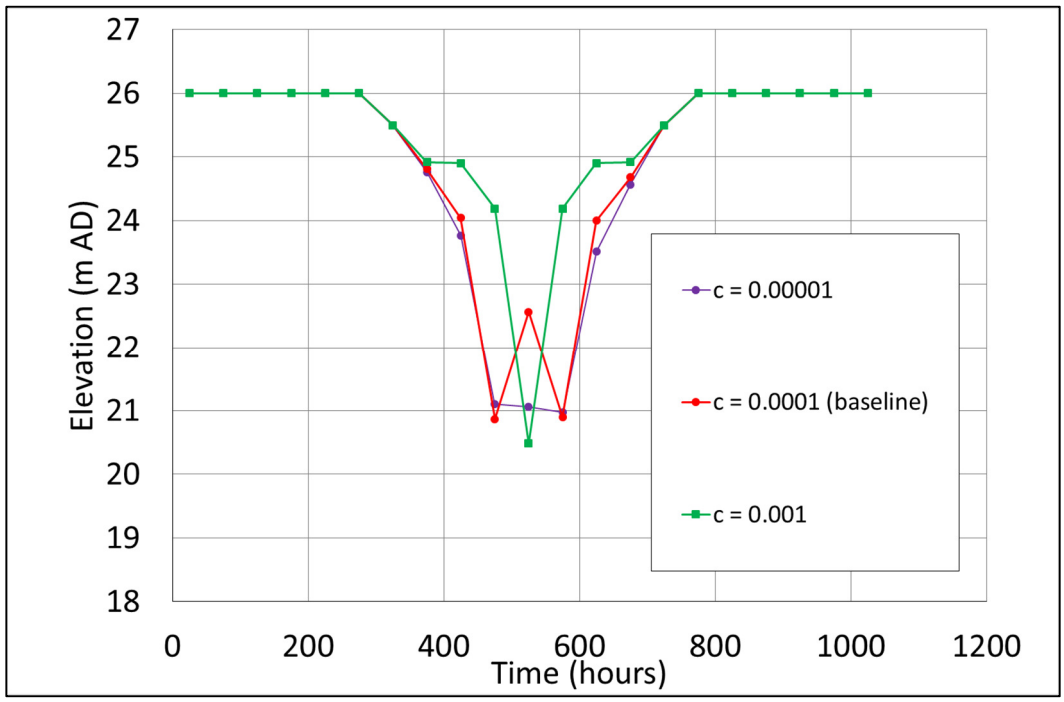

Figure 5.52: Cross section results at chainage 5,000m: sensitivity to boundary sediment input

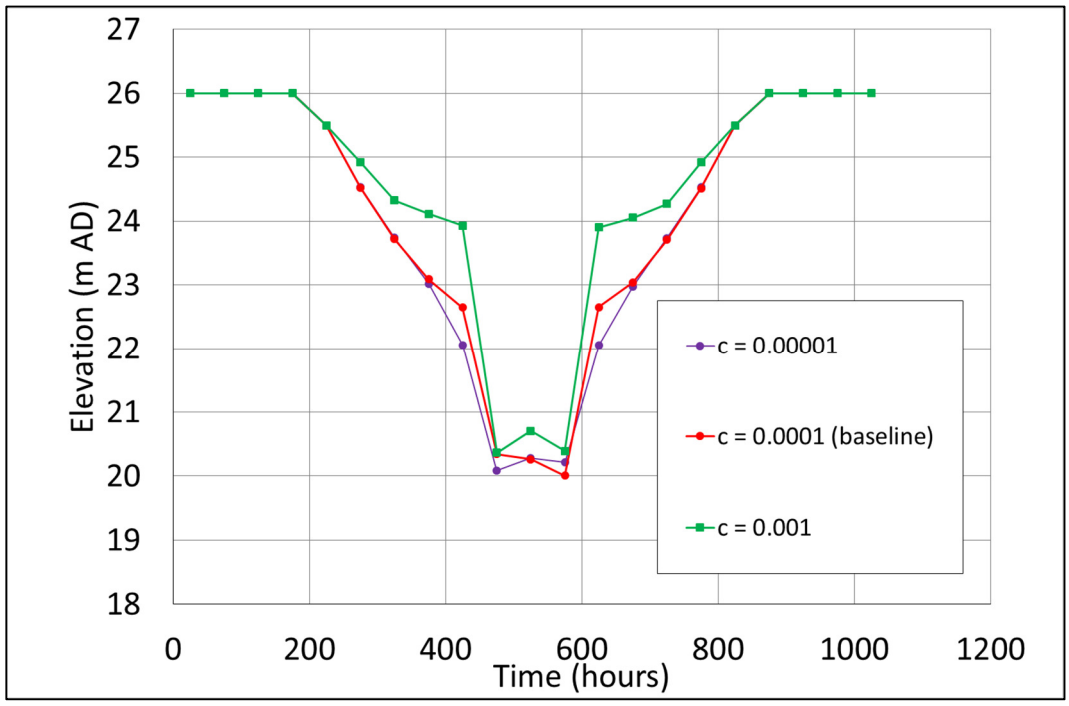

Figure 5.53: Cross section results at chainage 7,500m: sensitivity to boundary sediment input

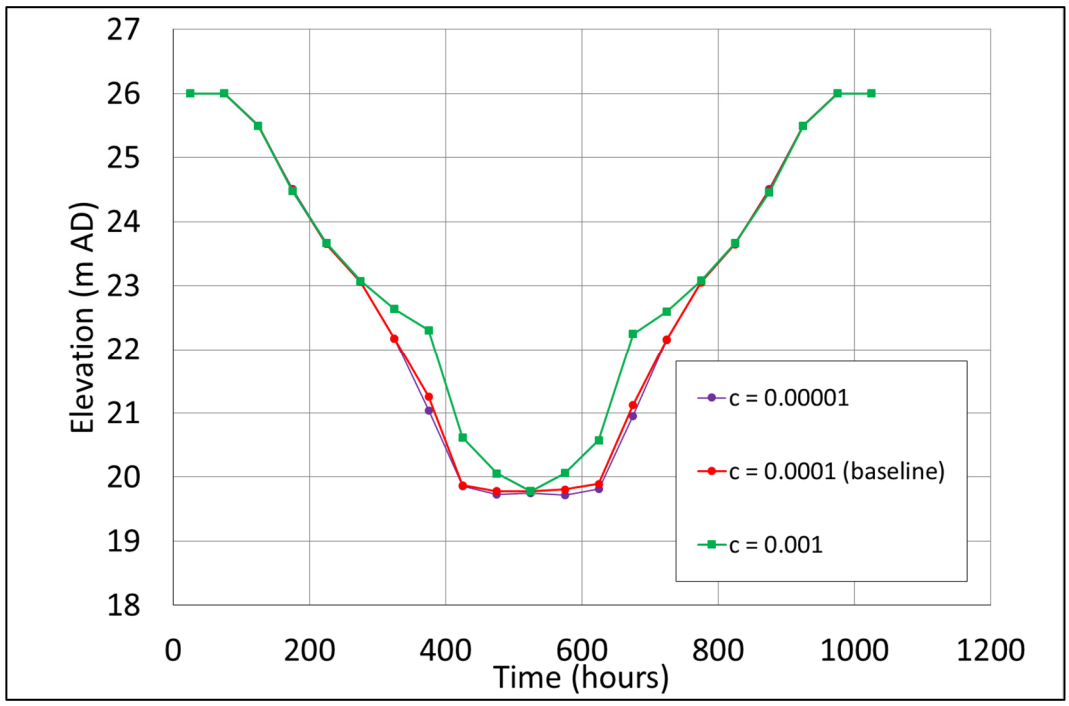

Figure 5.54: Cross section results at chainage 10,000m: sensitivity to boundary sediment input

From these figures it can be seen that increasing the sediment concentration at the boundary causes increased deposition, particularly in the inter-tidal areas. This reduces the

tidal volume and consequently downstream tidal flows are reduced, resulting in reduced channel dimensions near the mouth.

5.7 Sediment Properties

5.7.1 Sand Grain Size

To assess the impact of the representative sand grain size on the model results, simulations have been carried out with the median grain size (d_{50}) reduced to 100 µm and increased to 300 µm. In each case the d_{10} and d_{90} parameters have been adjusted to maintain the same ratio to d_{50} as in the baseline scenario, as shown in Table 5.3. The change in total sediment volume during these simulations is shown in Figure 5.55.

	d_{10}	d_{50}	d_{90}
Baseline	100 µm	150 µm	200 µm
Test 1	66 µm	100 µm	133 µm
Test 2	200 µm	300 µm	400 µm

Table 5.3: Sand grain sizes used in sensitivity testing

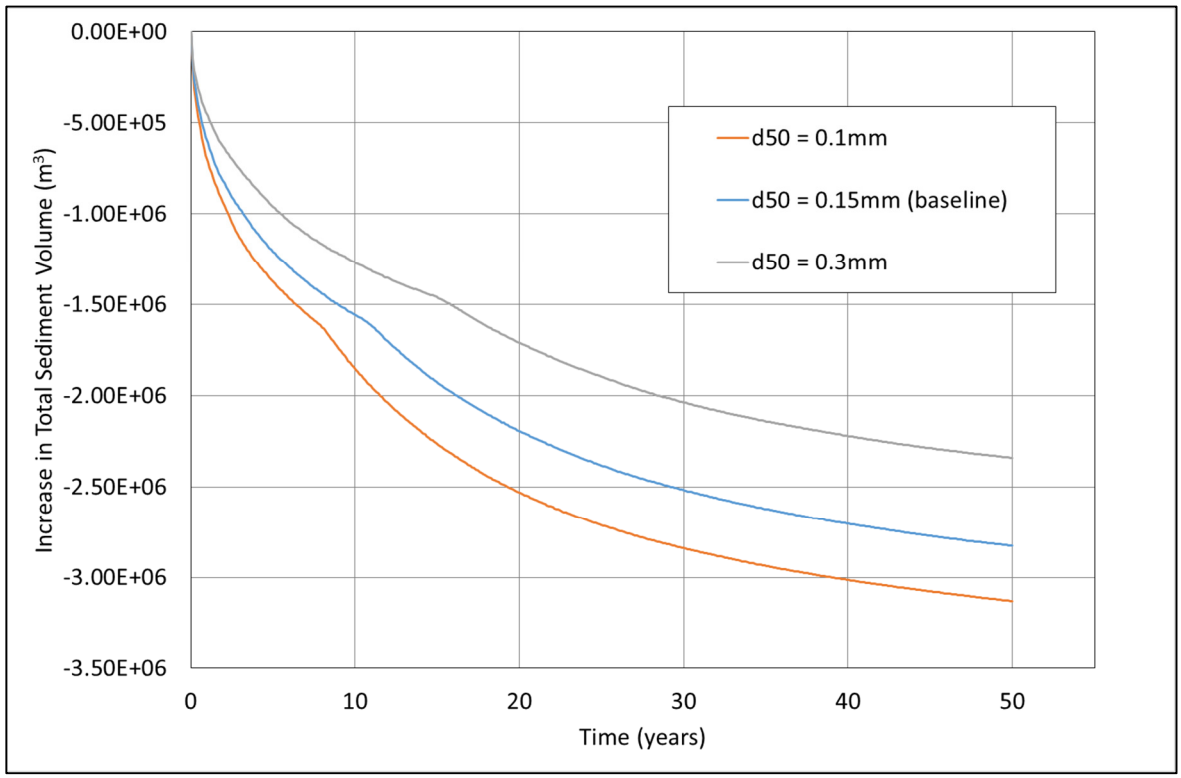

Figure 5.55: Change in total sediment volume: sensitivity to sand grain size

Figure 5.55 shows that, as might be expected, the rate of erosion is increased when the grain size in reduced (and vice versa), due to the increase in transport rates for smaller grain sizes. Figures 5.56 to 5.59 show cross section results for these simulations.

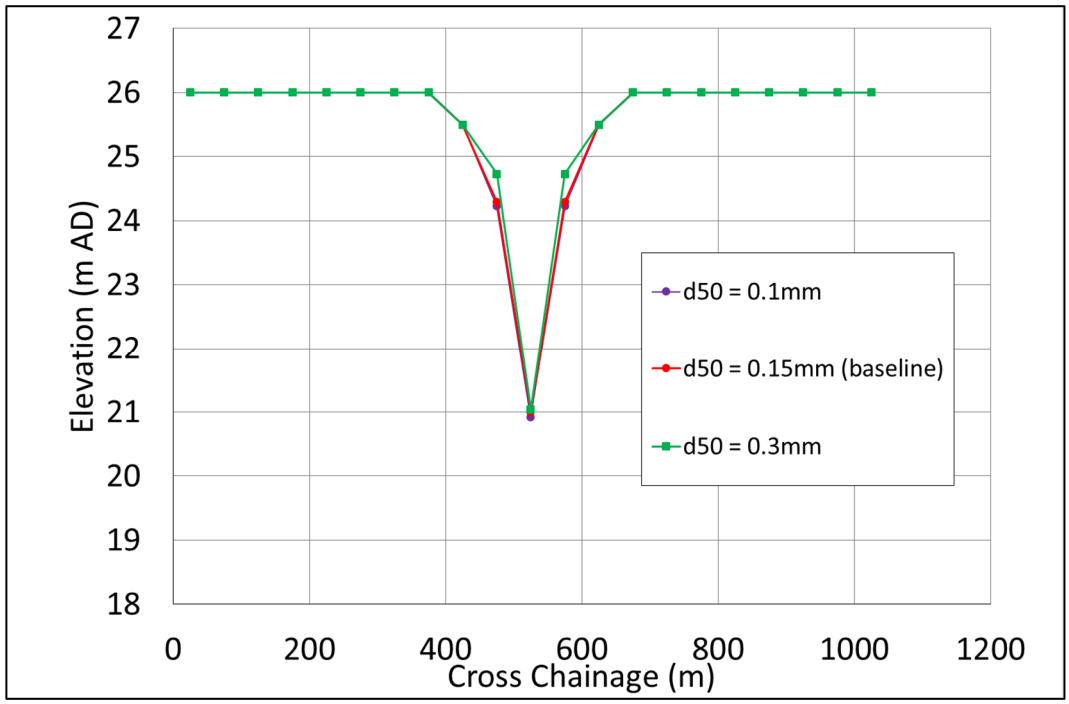

Figure 5.56: Cross section results at chainage 2,500m: sensitivity to sand grain size

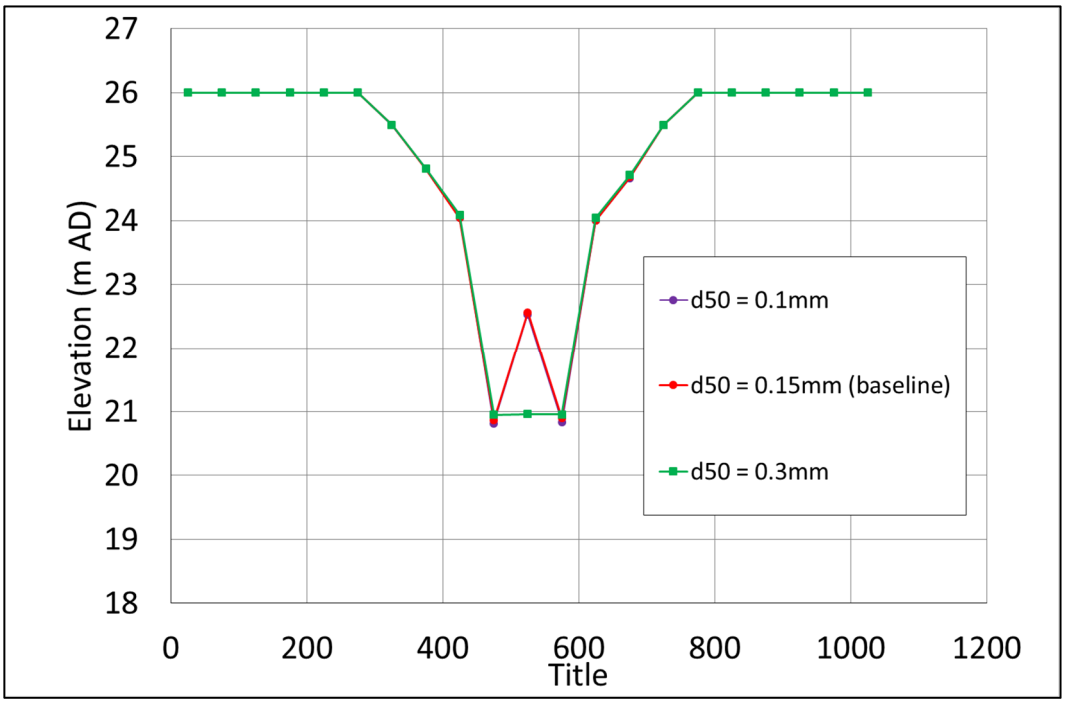

Figure 5.57: Cross section results at chainage 5,000m: sensitivity to sand grain size

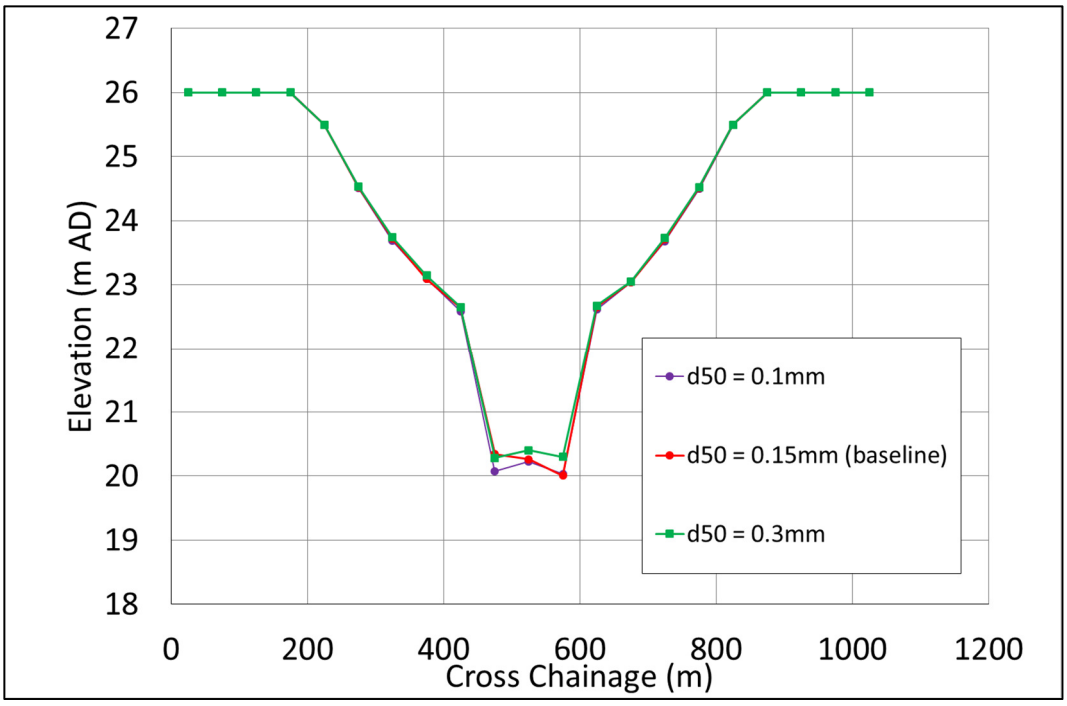

Figure 5.58: Cross section results at chainage 7,500m: sensitivity to sand grain size

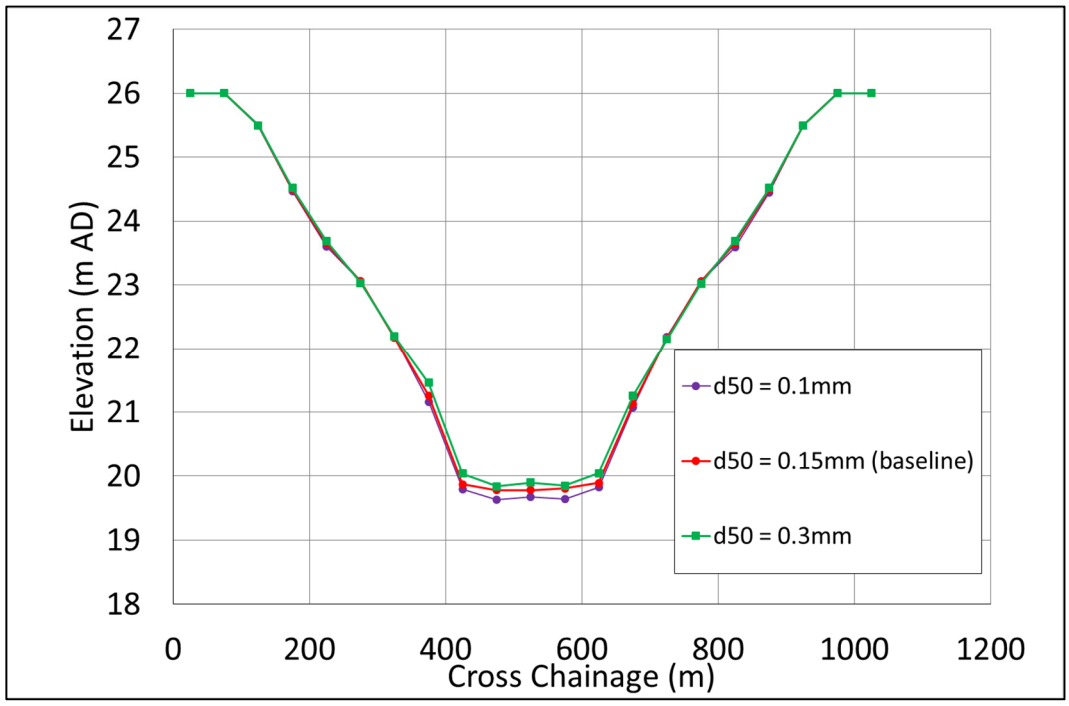

Figure 5.59: Cross section results at chainage 10,000m: sensitivity to sand grain size

Figures 5.56 to 5.59 show that the greatest difference between these simulations has occurred in the channel; however, the results are not shown to be very sensitive to the representative grain size parameter.

5.7.2 Critical Shear Stress for the Erosion and Deposition of Mud

Erosion of mud occurs in the model when the bed shear stress due to combined waves and currents (τ_{max}) exceeds the critical shear stress for erosion (τ_e), which is a fixed model parameter. Similarly, deposition can only occur when τ_{max} is below the critical shear stress for deposition (τ_d), which is typically around half the critical shear stress for erosion. To assess the sensitivity of the model results to this parameter, model runs have been performed with τ_e and τ_d reduced by 50% and increased by 50% from their baseline values of 0.12 N/m^2 (τ_e) and 0.06 N/m^2 (τ_d). Figure 5.60 shows the change in total sediment volume during these simulations.

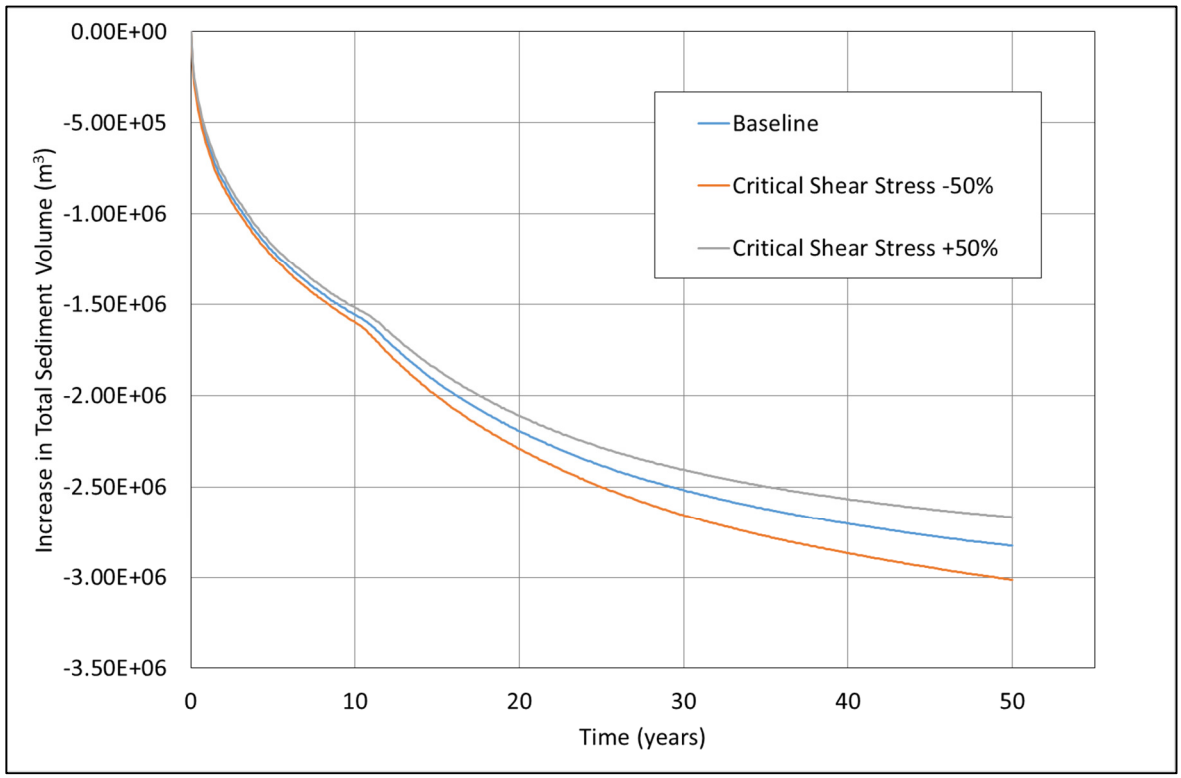

Figure 5.60: Change in total sediment volume: sensitivity to the critical shear stresses for erosion and deposition of mud

Figure 5.60 shows that the reduction in total sediment volume is enhanced when the critical shear stresses are reduced and vice versa. This is expected since reducing the critical shear stress for erosion will increase erosion rates while reducing the critical shear stress for deposition will make deposition less likely to occur and reduce deposition rates.

Figures 5.61 to 5.64 show the cross section results for these simulations. It is noted that only minor changes have occurred to the final bed levels, which is expected since the sediment composition in the initial conditions contains a mixture of 50% sand and 50% mud. Therefore, an increase in the erosion of mud will increase the proportion of sand in the bed material and hence limit the amount of subsequent erosion.

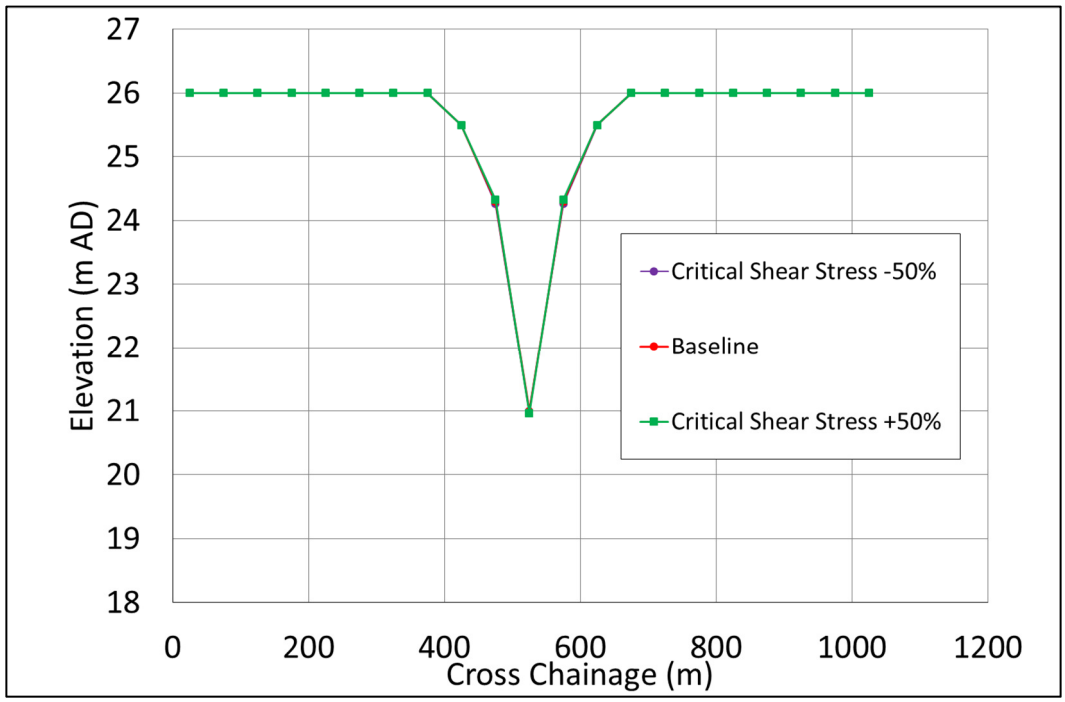

Figure 5.61: Cross section results at chainage 2,500m: sensitivity to critical shear stresses for erosion and deposition of mud

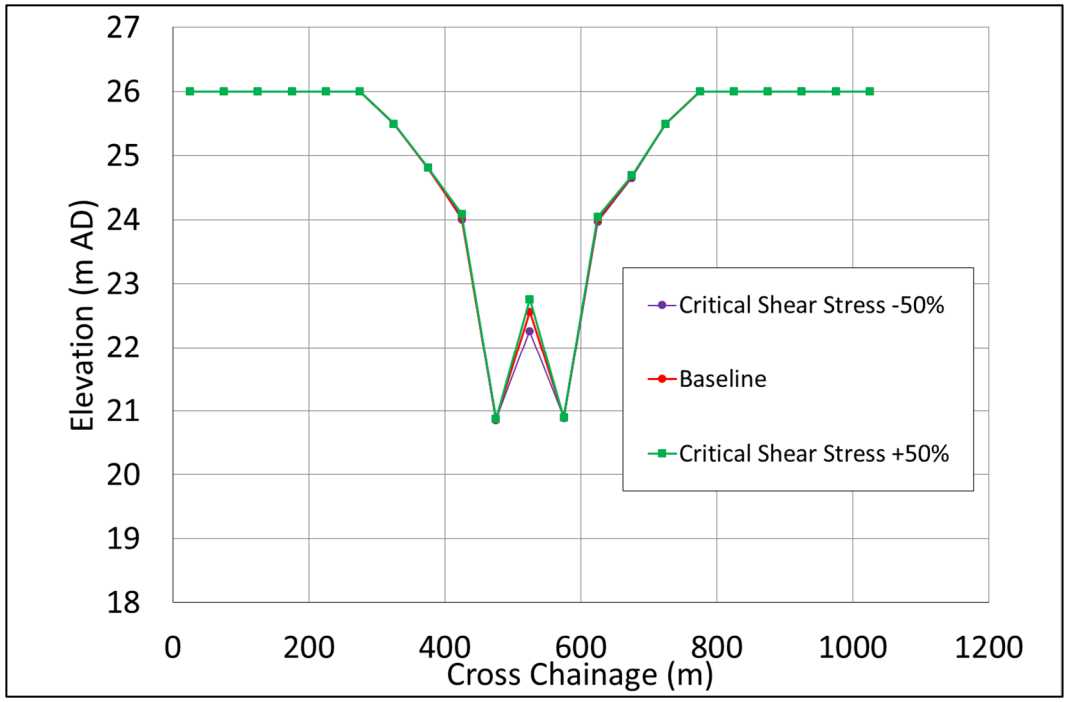

Figure 5.62: Cross section results at chainage 5,000m: sensitivity to critical shear stresses for erosion and deposition of mud

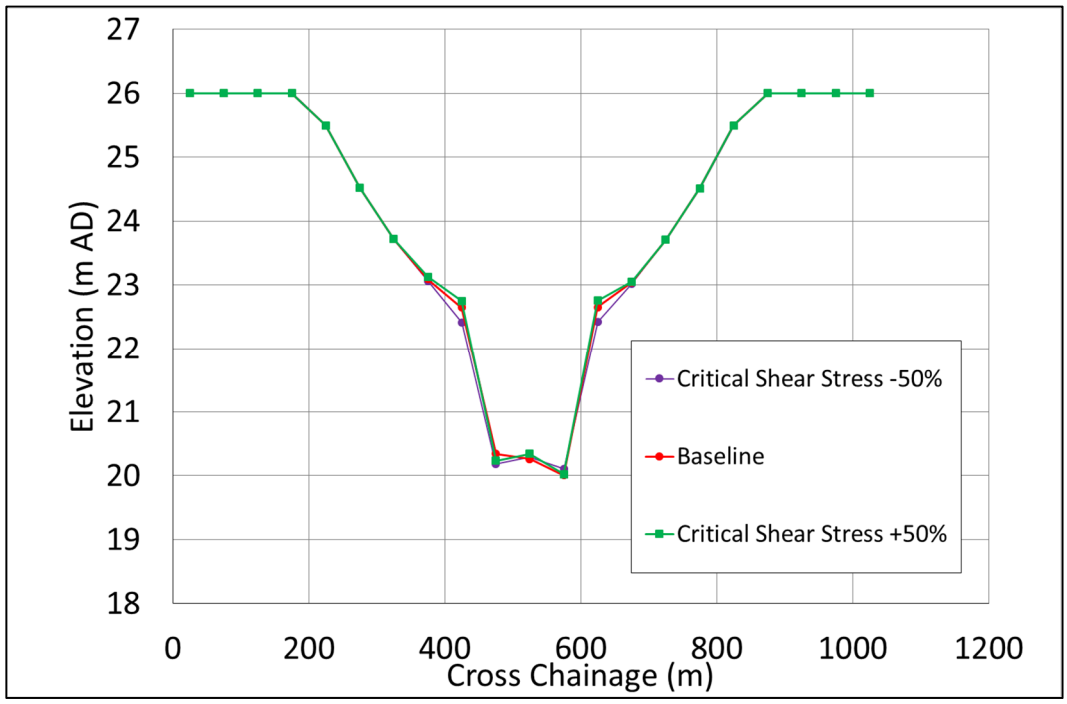

Figure 5.63: Cross section results at chainage 7,500m: sensitivity to critical shear stresses for erosion and deposition of mud

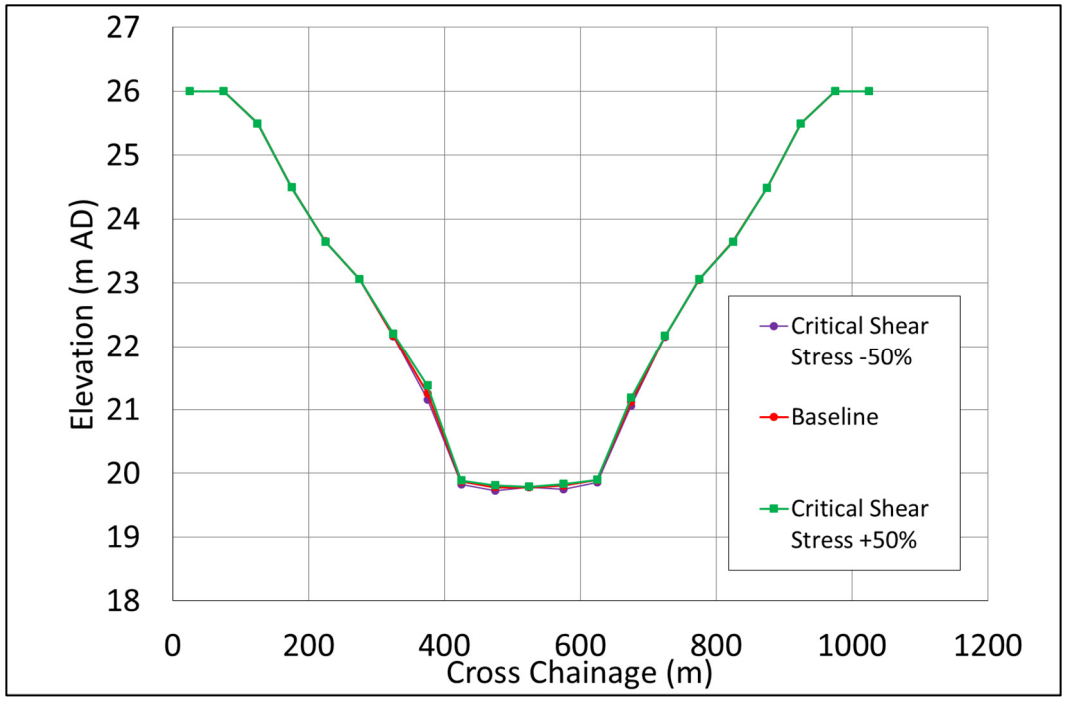

Figure 5.64: Cross section results at chainage 10,000m: sensitivity to critical shear stresses for erosion and deposition of mud

5.7.3 Multi-Fraction Sediment Transport

A multi-fraction simulation was performed, in which the sediment was divided into three sand fractions and a single mud fraction. The sand fraction sizes are given in Table 5.4 and are defined to be broadly consistent with the single fraction sediment parameters used in the baseline scenario.

Sand Fraction	Minimum particle size (μm)	Minimum particle size (μm)	Representative particle size (μm)
1	75	125	96.8
2	125	175	147.9
3	175	225	198.4

Table 5.4: Sand fraction sizes in the multi-fraction sediment sensitivity test

The change in sediment volume for each fraction is compared to the baseline results in Figure 5.65, where it can be seen that significantly higher erosion rates have occurred in the multi-fraction simulations. Sand erosion rates for Fractions 1 and 2 are both similar to the total sand erosion in the baseline scenario even though individually they represent a smaller proportion of the bed material. At present the reasons for this discrepancy have not been identified; however, the critical bed shear stress and sand transport rates are calculated using a different method and differences in these values may affect the final morphology. Also, transport rate of the smallest sand grain size in the multi-fraction approach is expected to be higher than those for single fraction sand, which may increase the erosion rate as this fraction is selectively eroded. Figures 5.66 to 5.69 show cross section results for the multi-fraction simulation.

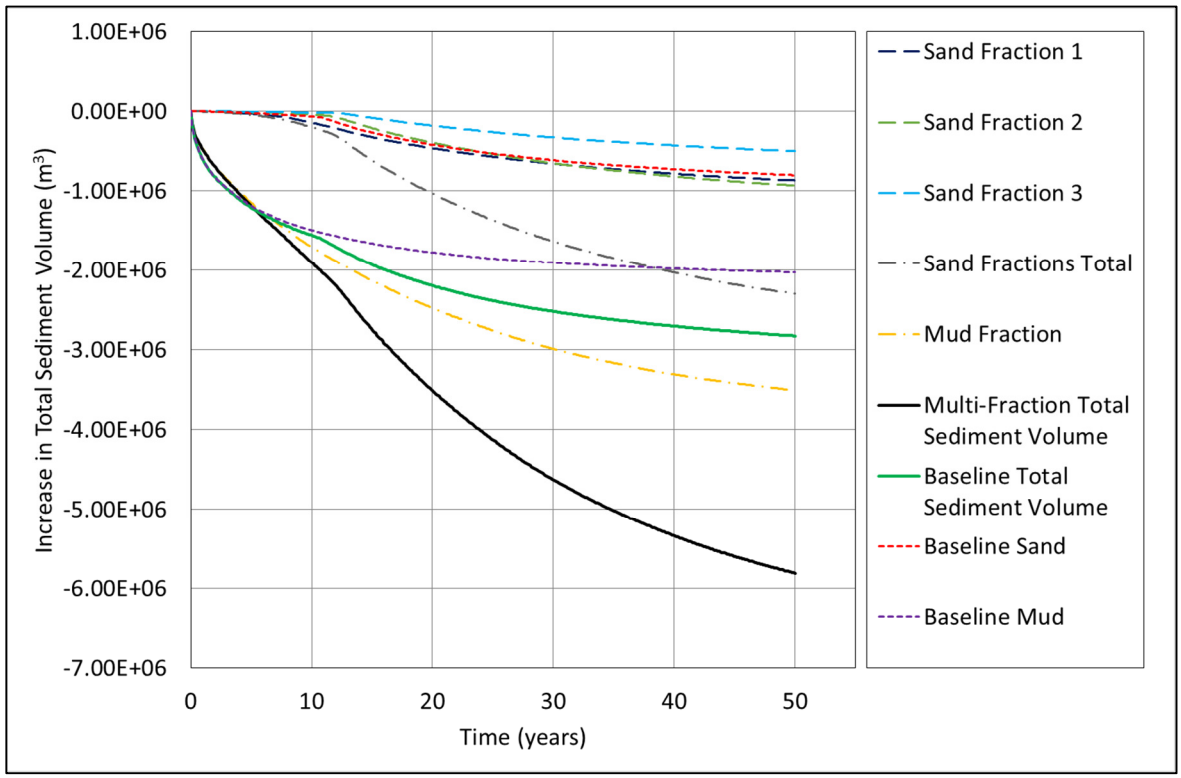

Figure 5.65: Change in total sediment volume: multi-fraction sediment transport

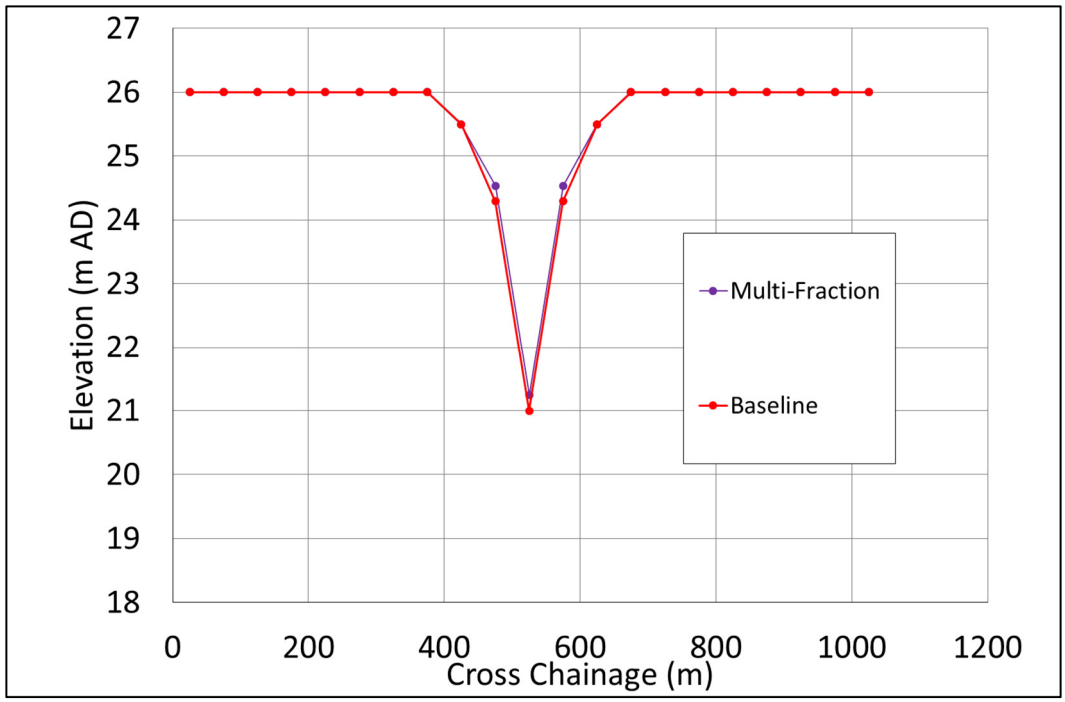

Figure 5.66: Cross section results at chainage 2,500m: multi-fraction sediment transport

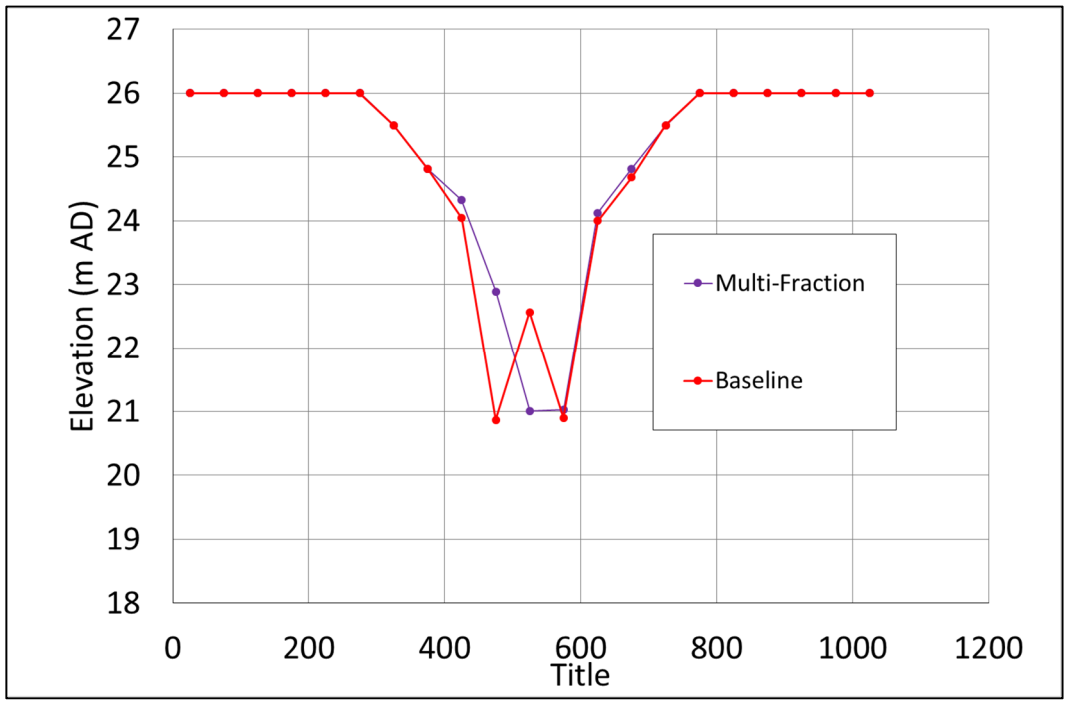

Figure 5.67: Cross section results at chainage 5,000m: multi-fraction sediment transport

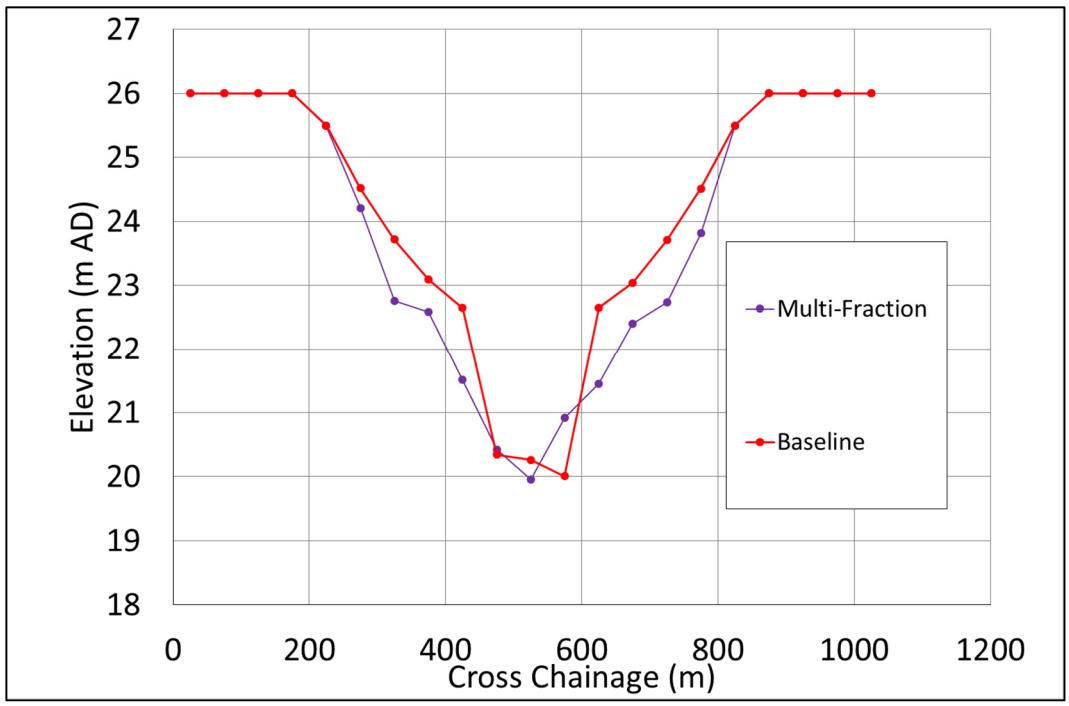

Figure 5.68: Cross section results at chainage 7,500m: multi-fraction sediment transport

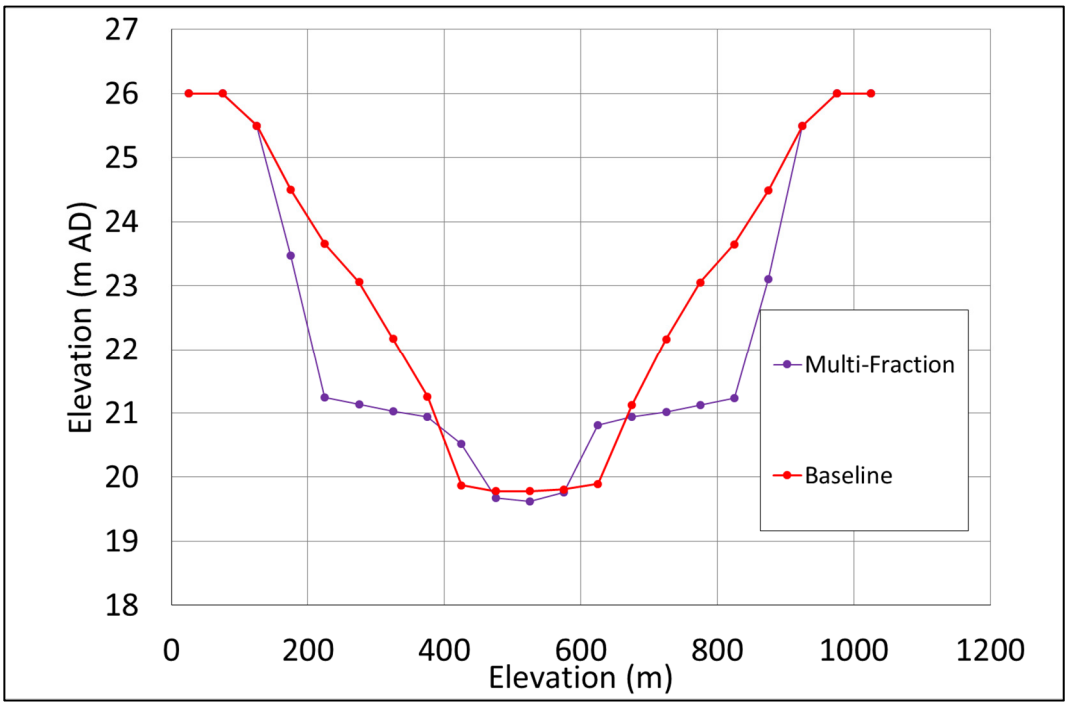

Figure 5.69: Cross section results at chainage 10,000m: multi-fraction sediment transport

Figures 5.66 to 5.69 show that the increased erosion in the multi-fraction simulation is concentrated near the marine boundary, where the wave energy is greatest. Final bed levels in the inner estuary are actually slightly higher for the multi-fraction approach.

Figure 5.70 shows the distribution of sediments at the end of the simulation period, as the calculated median grain size in the active layer. This shows that the median grain size greater in the outer estuary and in the channel bed, where the bed shear stress is highest.

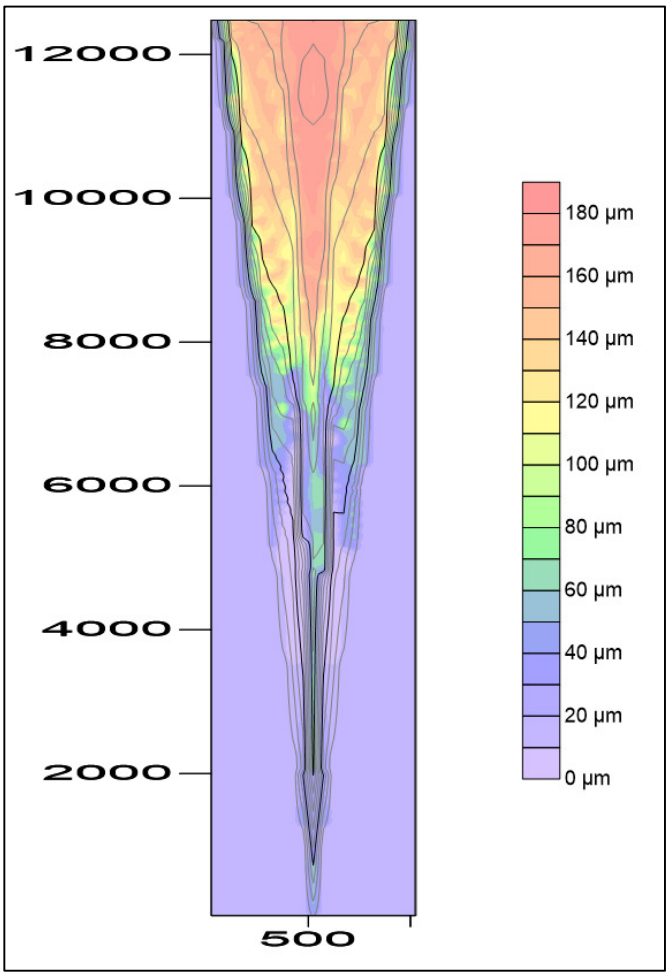

Figure 5.70: Contour plot showing the bathymetry (contour lines) and the median grain size in the active layer (shading)

5.7.4 Diffusion Coefficient

In this research the transport of suspended sediment has been modelled as a purely diffusive process and the diffusion coefficient (K_d) in the model has been set to 500 m^2/s for the baseline scenario (see section 4.5.2). To assess the sensitivity of the model results to this parameter simulations have been carried out with K_d increased to 2500 m^2/s, reduced to 100 m^2/s and 10 m^2/s. The change in total sediment volume during these simulations is given in Figure 5.71.

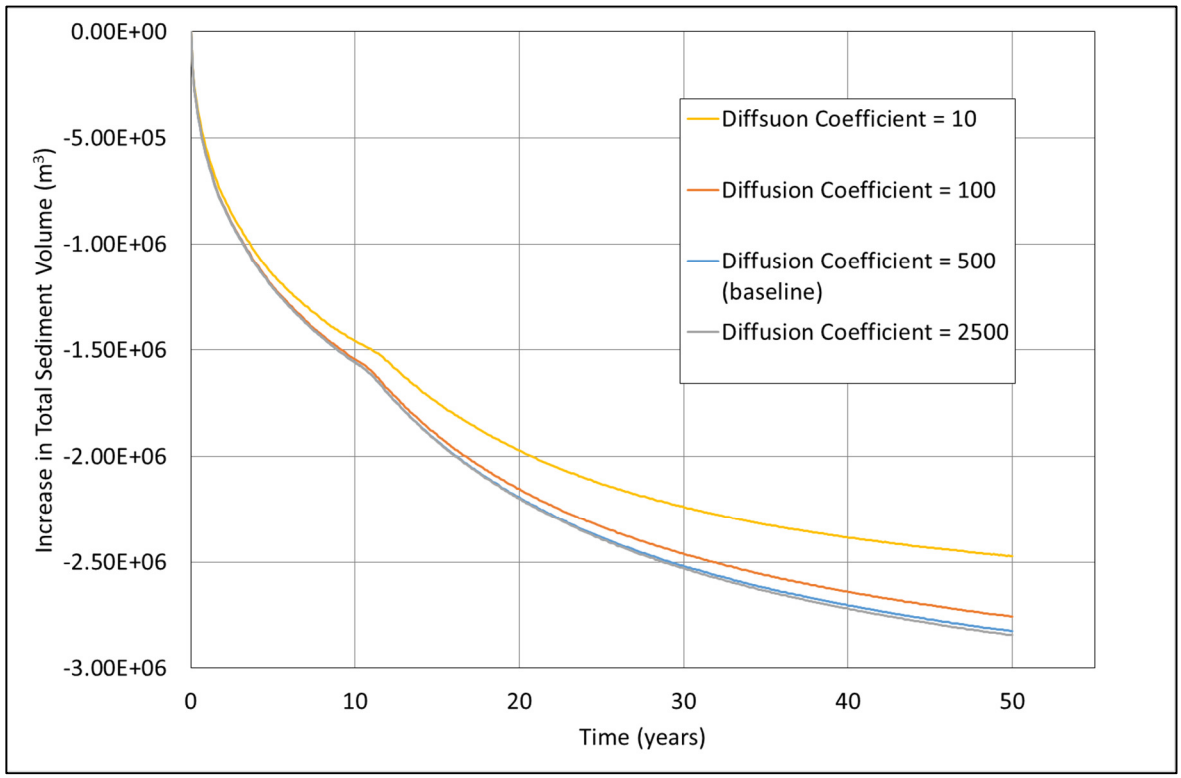

Figure 5.71: Change in total sediment volume: sensitivity to the diffusion coefficient

Figure 5.71 shows that the reduction in total sediment volume during these simulations is less when the diffusion coefficient is reduced. This is as expected, since significant erosion occurs, which will tend to increase local sediment concentrations and consequently increase the rate of deposition in nearby cells. When the diffusion coefficient is lower, this effect is enhanced.

Increasing the diffusion coefficient to 2500 m^2/s had little effect on the results because at 500 m^2/s the coefficient is already high enough to prevent the local sediment concentration from deviating significantly from the boundary sediment concentration.

The cross section results, given in Figures 5.72 to 5.75 indicate that the increased deposition has occurred in the inter-tidal areas, for reduced values of K_d.

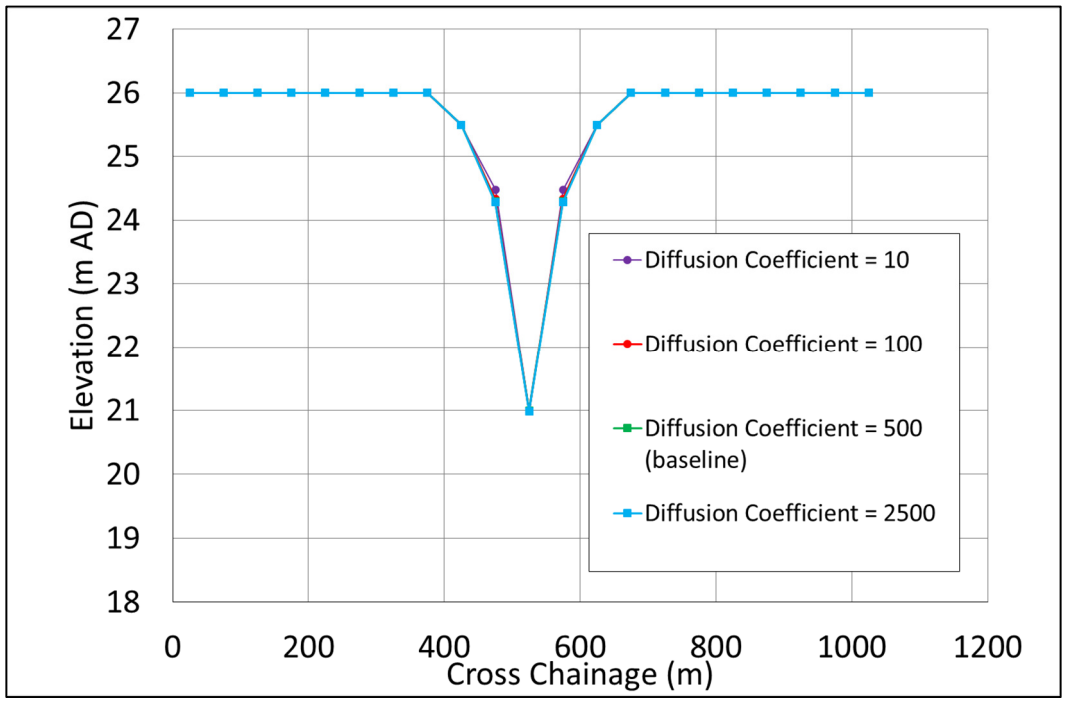

Figure 5.72: Cross section results at chainage 2,500m: sensitivity to the diffusion coefficient

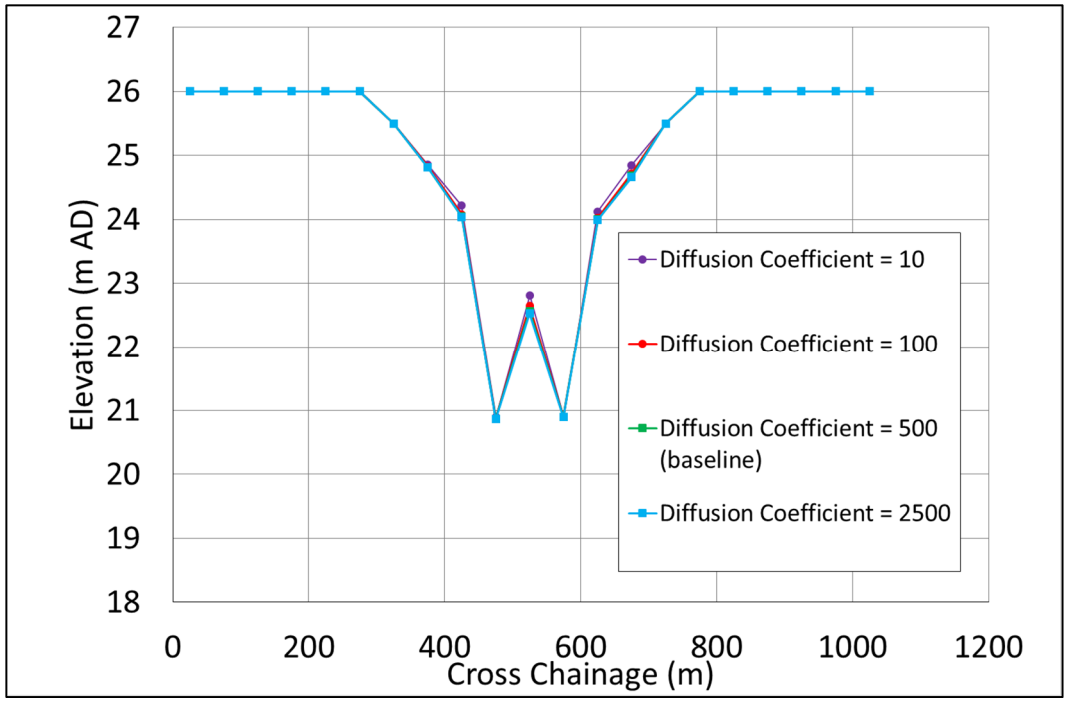

Figure 5.73: Cross section results at chainage 5,000m: sensitivity to the diffusion coefficient

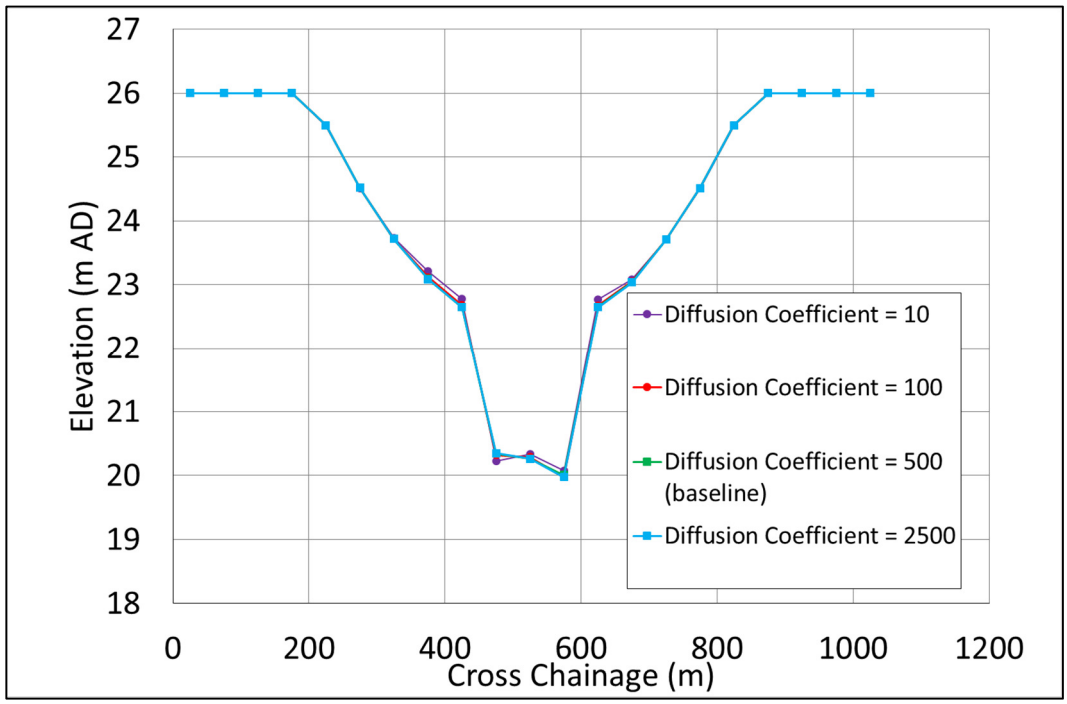

Figure 5.74: Cross section results at chainage 7,500m: sensitivity to the diffusion coefficient

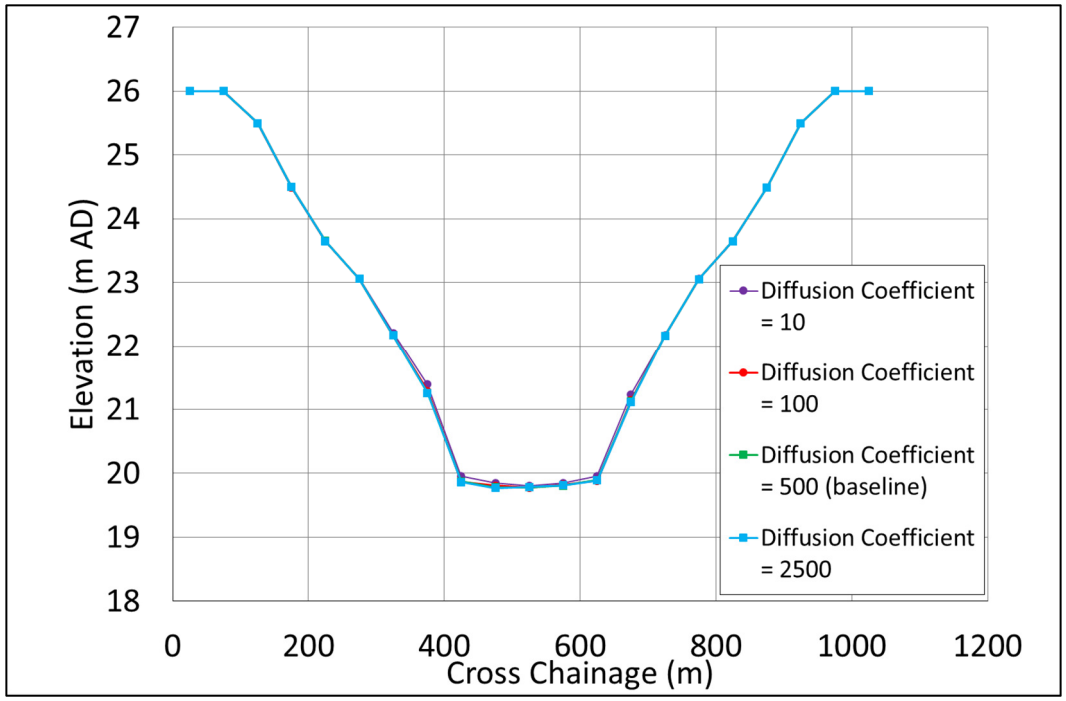

Figure 5.75: Cross section results at chainage 10,000m: sensitivity to the diffusion coefficient

5.8 Salt Marsh

Salt marsh has not been included in the baseline simulation; however, to assess the effect of the salt marsh component of the model three biomass functions have been modelled. The parameters $T_{i,max}$, $T_{i,min}$, $T_{i,1}$ and $T_{i,2}$ for each profile are listed in Table 5.5 (see Section 4.6 for discussion of biomass functions). Biomass Functions 1 and 2 correspond to the alternative profiles proposed by D'Alpaos et. al. (2007), while profile 3 is an approximation of the quadratic profile adopted by Morris (2006). In each case the maximum biomass is set to be 2000 g/m^2.

Biomass Function	$T_{i,max}$ (%)	$T_{i,1}$ (%)	$T_{i,2}$ (%)	$T_{i,min}$ (%)
1	50	49	48	0
2	50	2	1	0
3	50	35	15	0

Table 5.5: Biomass function parameters

Biomass Function 1 is similar to the linear function adopted by Mudd (2004), while Biomass Function 2 is similar to the function proposed by D'Alpaos et. al. (2007), for cases where a variety of marsh species are present and Biomass Function 3 is intended as an approximation of the parabolic function adopted by Mariotti and Fagherazzi (2010) (see Section 3.4.7). The change in total sediment volume during each simulation is given in Figure 5.76.

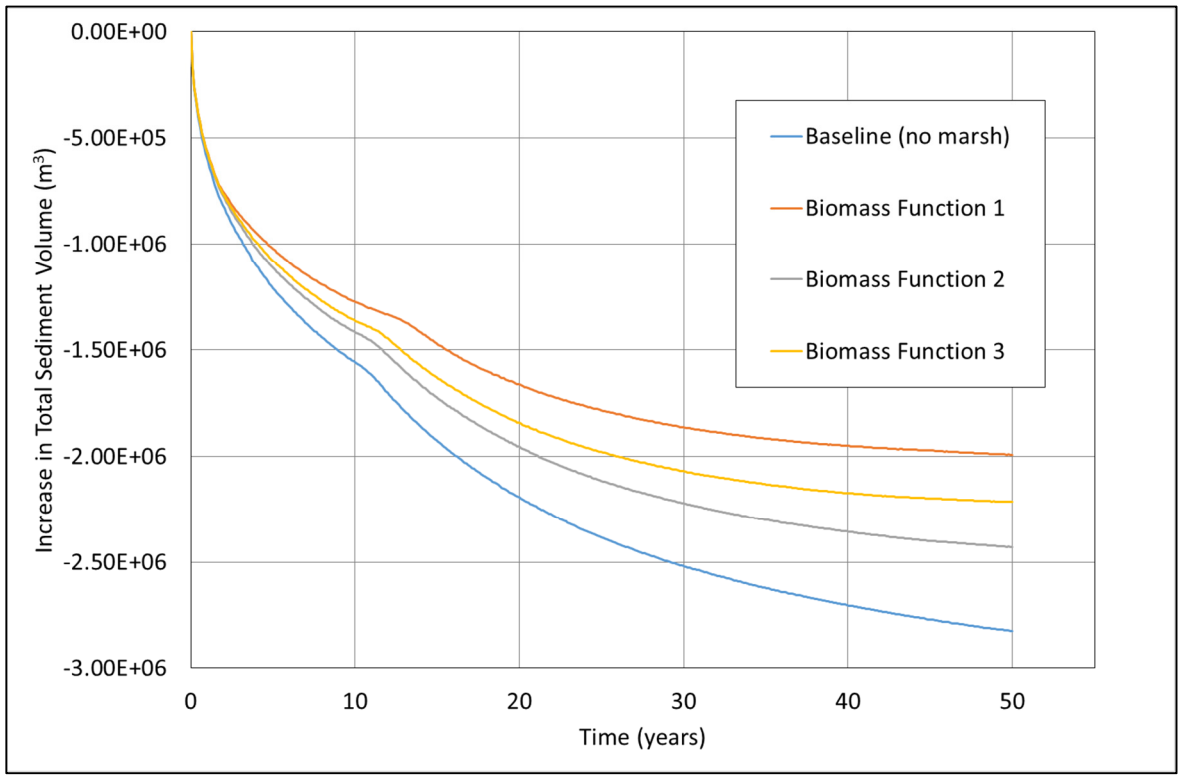

Figure 5.76: Change in total sediment volume: sensitivity to salt marsh biomass distribution

Figure 5.76 shows that all three salt marsh biomass functions have resulted in higher final total sediment volumes, with Function 1 having the greatest effect and Function 2 the least effect. This is further illustrated by the cross section results given in Figures 5.77 to 5.80 , which also confirm that the differences in bed levels between each set of results are primarily located in the inter-tidal areas. Biomass Function 1 produces the greatest effect because the biomass for this function is greatest at the lowest elevation and since this area is more frequently inundated than marsh at higher elevations, the enhanced sedimentation has a greater effect; however, it is possible that for a longer simulation period one of the other biomass functions would give the greatest total amount of sedimentation. Biomass Function 2, for example, will produce increasing sedimentation rates with increasing marsh elevation.

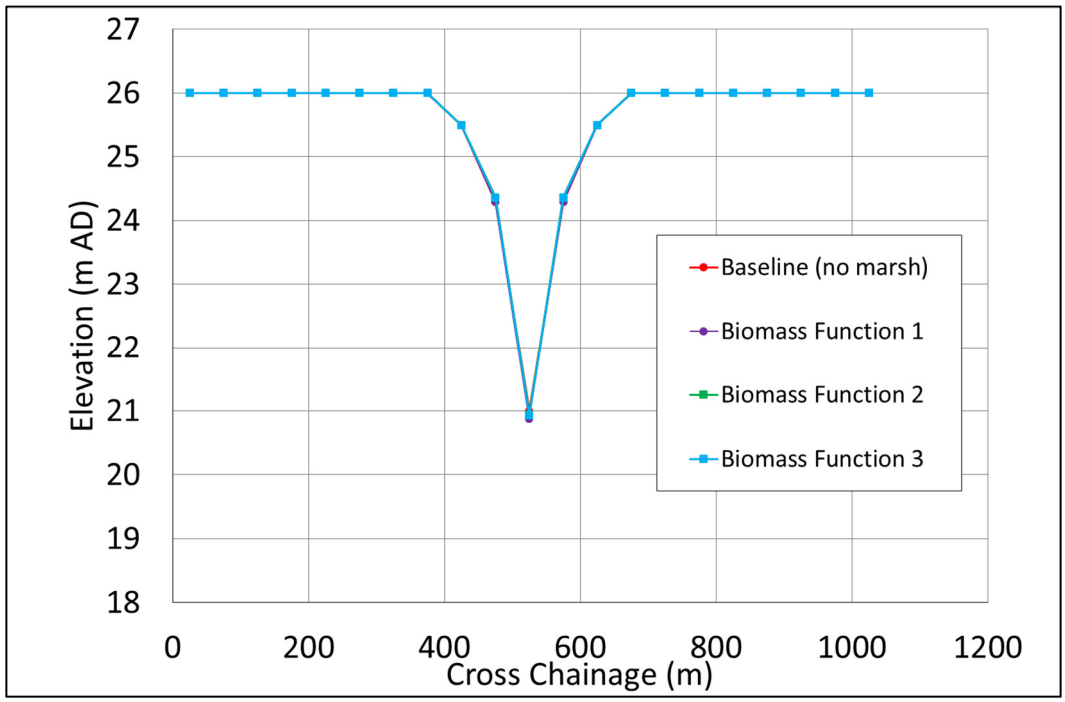

Figure 5.77: Cross section results at chainage 2,500m: sensitivity to salt marsh biomass function

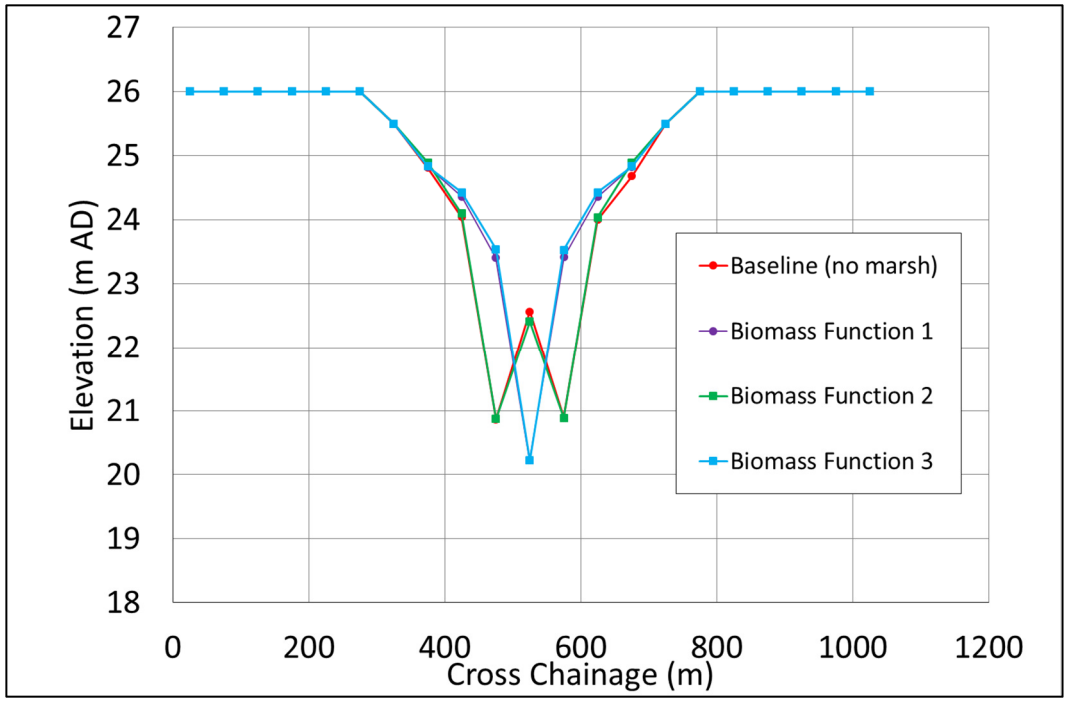

Figure 5.78: Cross section results at chainage 5,000m: sensitivity to salt marsh biomass function

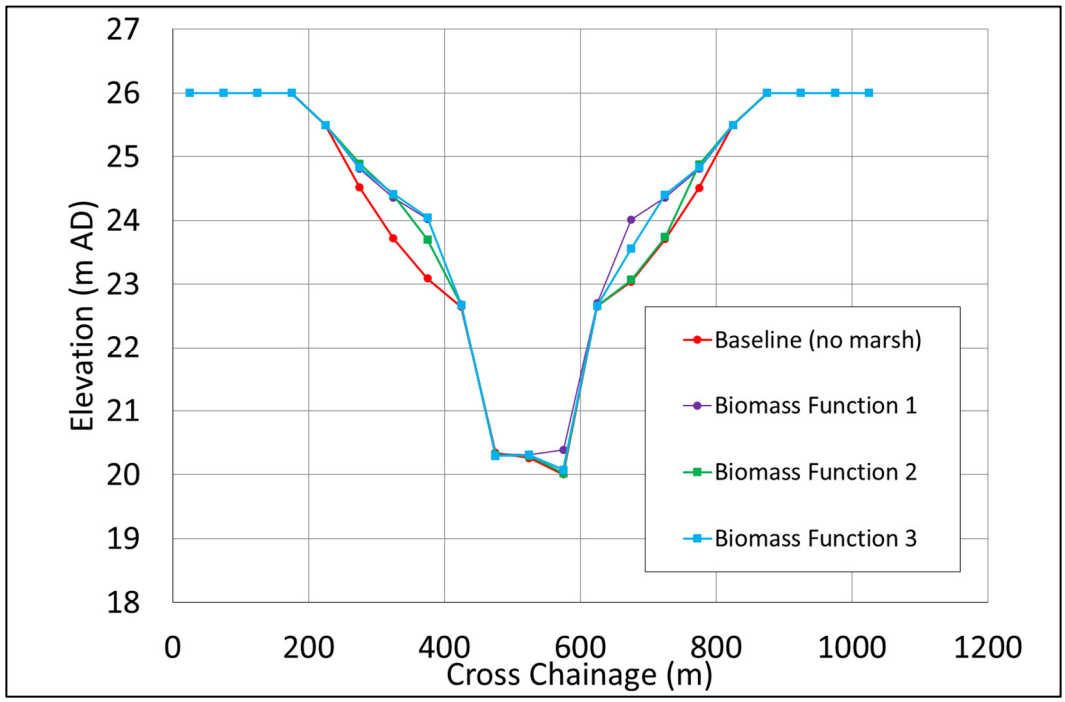

Figure 5.79: Cross section results at chainage 7,500m: sensitivity to salt marsh biomass function

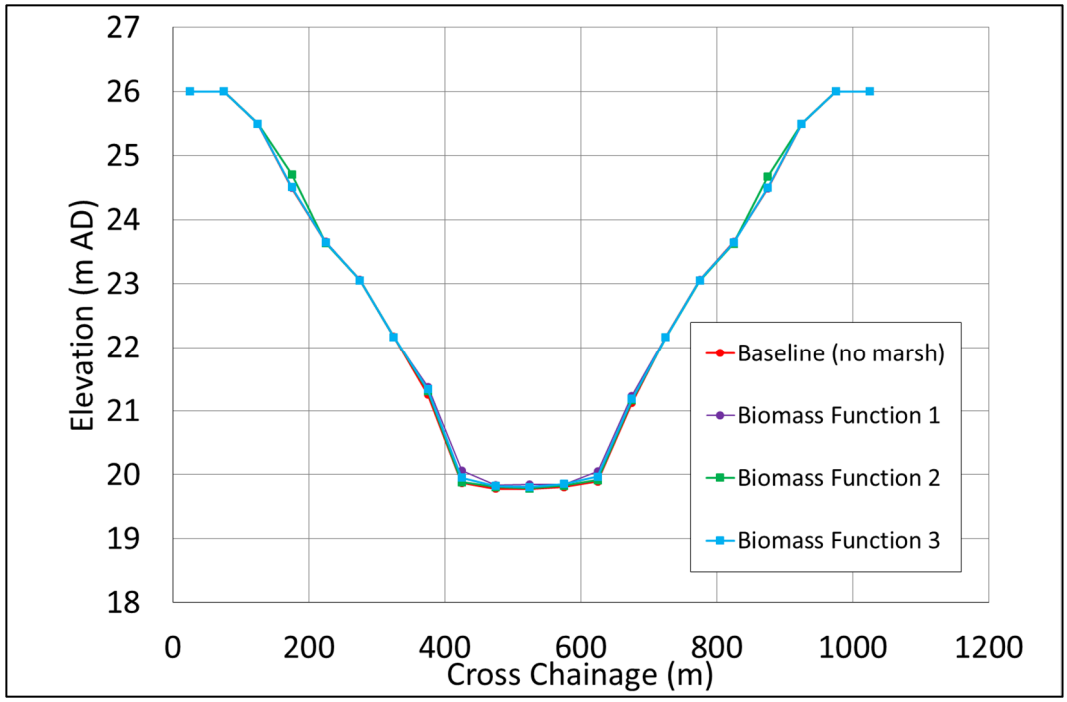

Figure 5.80: Cross section results at chainage 10,000m: sensitivity to salt marsh biomass function

5.9 Sensitivity Analysis Summary

The sensitivity analyses have confirmed that the model results are not very sensitive to the time-step provided that it is sufficiently short to resolve the variation in tidal flows during a single tidal cycle. For 20 and above time-steps per tidal cycle the results are not sensitive to the time-step, while reducing this value to 10 has a small but noticeable effect. Similarly the results were not very sensitive to the adopted cell size provided that this is small enough to adequately resolve the central channel.

The model responded as expected, in qualitative terms, to variations in the hydrodynamic forcings, with increases in fluvial flow predominantly affecting the channel dimensions in the inner estuary, tidal range primarily affecting the channel dimensions in the outer estuary and wave energy primarily affecting bed levels in the inter-tidal areas.

It was anticipated that the influence of the initial model settings would be reduced over time, depending on the applied forcing. In fact the final model results were found to be quite sensitive to the imposed initial settings; however, the differences from the baseline scenario were reduced over time, indicating that with a sufficient simulation period the results should converge towards a bathymetry depending only on the applied forcings, sediment inputs and geological constraints, as expected.

The sensitivity to the minimum bed levels (representing geological constraints) was found to be localised near the upstream end of the model, where channel erosion had been limited by the minimum levels; however, a smaller effect was found throughout the model domain and this is thought to be caused by the effect of the erosion in the upstream section on the tidal prism and hence tidal flows.

The results were found to be significantly sensitive to the boundary sediment concentration, as expected; however, the sensitivity to the rate of sand input at the boundary was quite low and it was found that the effect of the sand input was localised at the marine boundary. It is noted that, at present the model does not include sand transport due to waves (although the enhancement by waves of transport due to currents is included) and this may be necessary to improve model performance in relation to sand movements in the outer estuary.

The model also responded as expected to changes to the sediment properties (sand grain size and critical shear stress for the erosion and deposition of mud). The difference between the results for the multi-fraction simulation and baseline simulation were greater than expected and the reasons for this have not been determined at present; however, the results multi-fraction simulation are otherwise as expected, qualitatively.

Finally the salt marsh biomass function has a significant effect on the inter-tidal areas, due to the enhanced sedimentation, increased hydraulic roughness and wave attenuation. This behaviour is also qualitatively as expected.

CHAPTER 6

FUTURE SCENARIO TESTING

6.1 Introduction

The ability of the model to predict the consequences of future morphodynamic changes in an estuary has been assessed using three hypothetical test scenarios, comprising an accelerated sea level rise scenario, a dredging scenario and an engineering work scenario, as described in Table 6.1.

To minimise any effect from the initial conditions and establish, as far as possible, equilibrium conditions in the estuary an initial 500 year simulation was carried out, with the results used to generate initial conditions for a baseline future scenario and each of the scenarios listed in Table 6.1, as illustrated by Figure 6.1.

Salt marsh was included in all the scenario simulations, using the salt marsh parameters $T_{i,max} = 0.5$, $T_{i,1} = 0.49$, $T_{i,2} = 0.48$ and $T_{i,min} = 0$. These parameters correspond to Biomass Function 1, as described in Section 5.8, which was chosen due to its tendency to give a stable marsh elevation relative to mean sea level (Biomass Function 2, for example, was found to have a strong tendency to cause sedimentation up to the maximum water level).

Scenario	Type	Description
1	Accelerated sea level rise	Rate of sea level rise increased to 10 mm/year
2	Channel dredging	100m wide, 10m deep channel created along the centre of the estuary, extending 3.5 km from the mouth.
3	Engineering work	Creation of a harbour area in the outer estuary.

Table 6.1: Future Scenarios

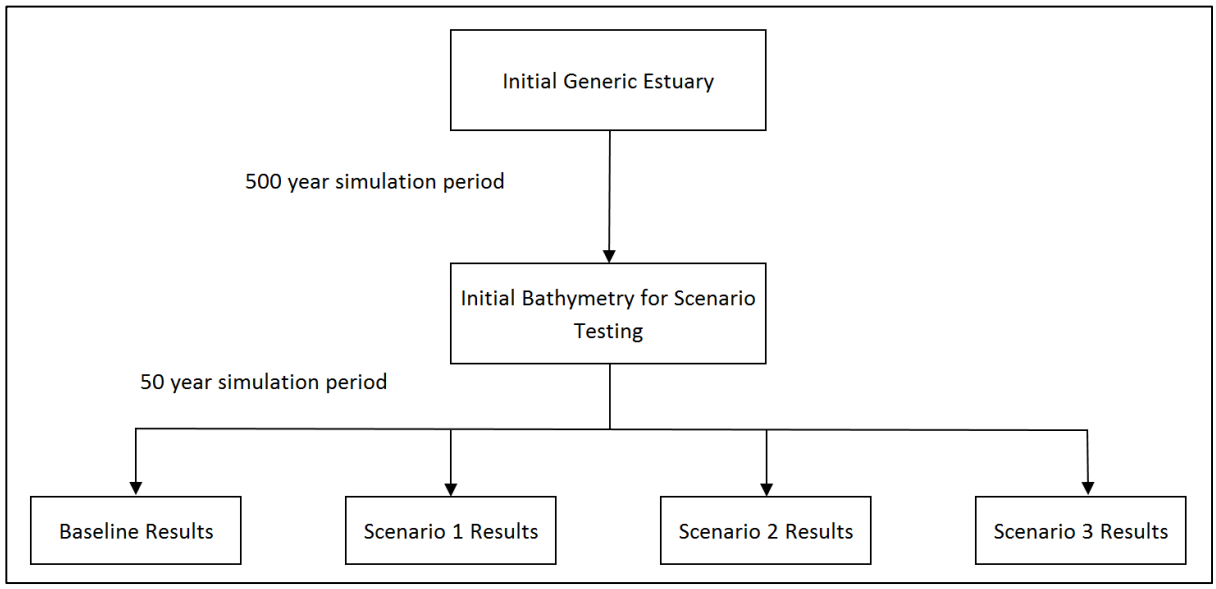

Figure 6.1: Future scenario simulations

6.2 Initial Conditions

An initial 500 year simulation was carried out to generate initial conditions for the baseline and scenario simulations as illustrated in Figure 6.1. The initial conditions and model parameters for the 500 year simulation were the same as for the sensitivity testing baseline described in Section 5.1, except that the initial mean sea level was reduced to 22m AD (to allow space for sea level rise to occur) and salt marsh was included in the simulation, as described above. A constant rate of sea level rise, of 2 mm/yr, was applied during the first 500 year simulation period, to give a final mean sea level of 23m AD. Figure 6.2 shows the change in sediment volume during this simulation.

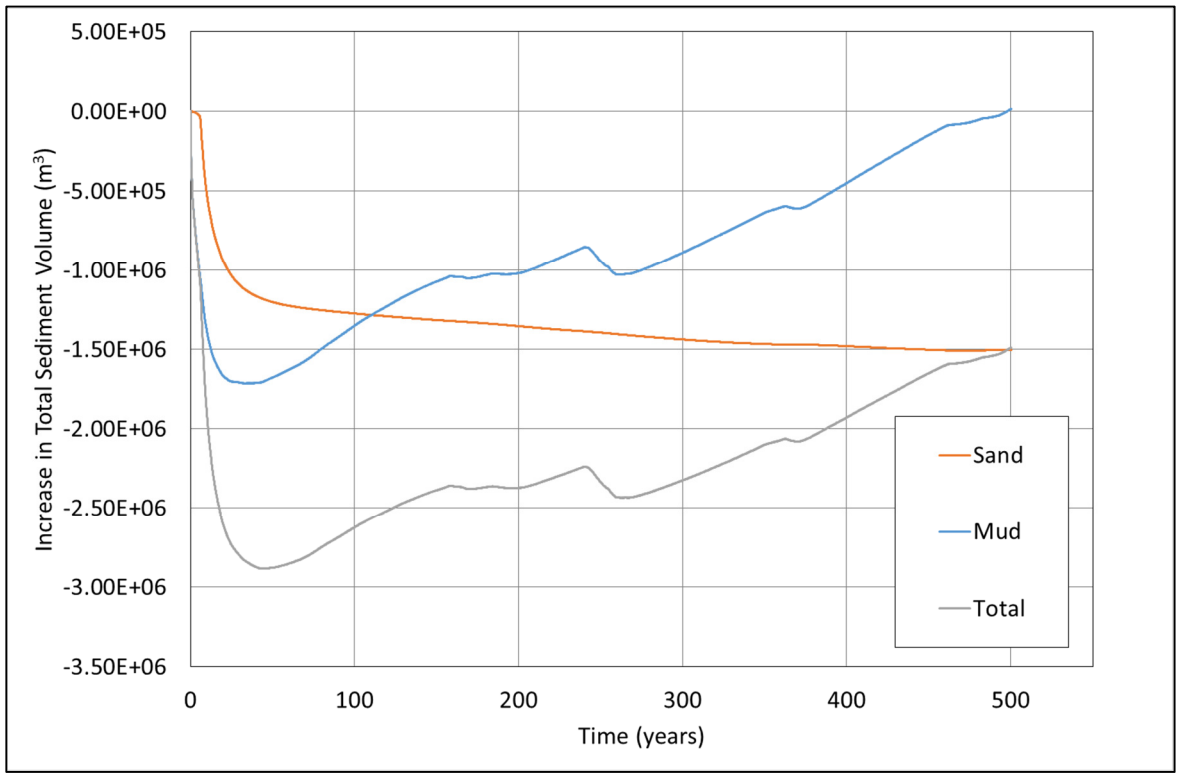

Figure 6.2: Change in total sediment volume during the initial 500 year simulation

Here it can be seen that following an initial period of rapid adjustment from the initial conditions, the rate of reduction in the volume of sand within the model domain has slowed and has appears to have stabilised at approximately 1.5 million cubic metres of eroded sediment. For context, if averaged over 1000 50m x 50m cells (20% of the model domain area) this corresponds to an eroded depth of 0.6m. Basin filling would normally be expected to occur due to sea level rise (Section 2.4); however, erosion occurs in this case due to the artificial nature of initial conditions. The model was also found to be unable to import significant volumes of sand in most cases and this is thought to be because the model tends to overestimate flow velocities during the ebb tide.

Following a similar initial adjustment the volume of mud within the model domain has increased steadily for the remainder of the simulation period, due to deposition on the tidal flats, and has returned to its initial value by the end of the simulation period. Contour plots showing the final bathymetry and depth of mud in the active layer are given in Figure 6.3, below. It is noted that the steady increase in the volume of mud, in Figure 6.2, is interrupted by a significant dip in the volume of mud, occurring after around 240 years. Further investigation revealed that an area of tidal flat at around chainage 9,000m was

eroded during this period and subsequently recreated. The reasons for this event have not been determined; however, this provides an illustration of the complex behaviour that can be generated by the model, due to interactions and feedback between the morphology and hydrodynamic processes.

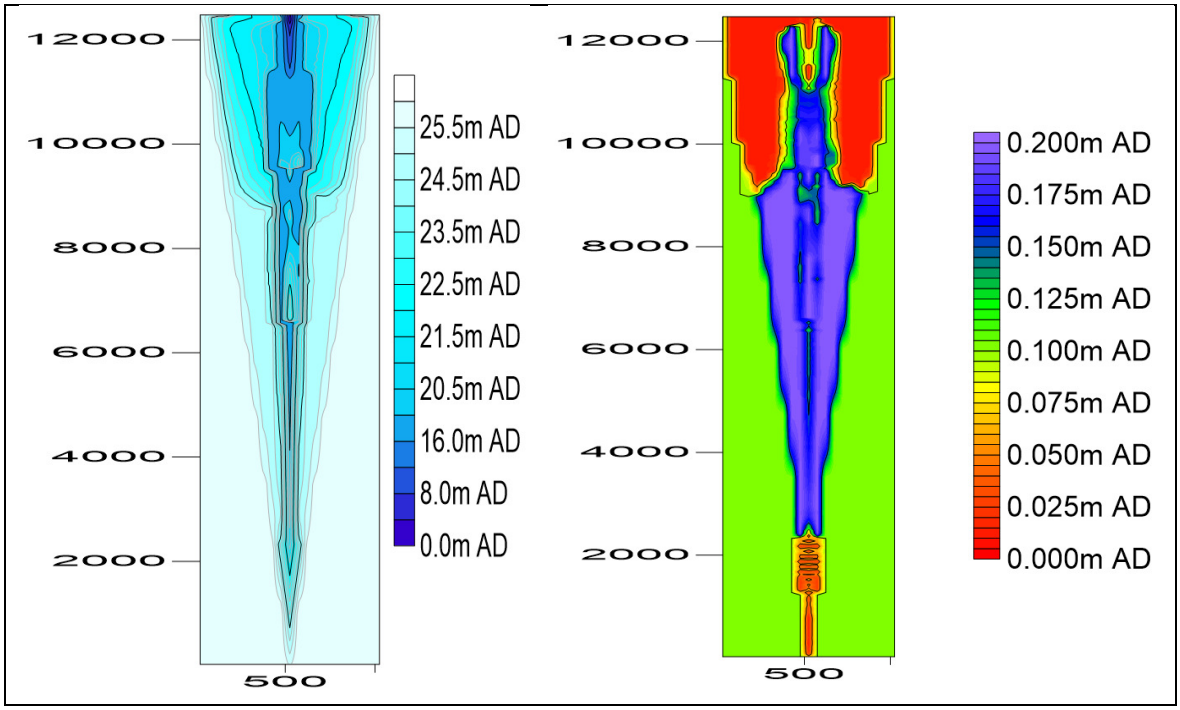

Figure 6.3: Bathymetry (left) and depth of mud in active layer following 500 year simulation period

From Figure 6.3, deposition of mud has occurred on the tidal flats in the middle part of the estuary. The bed of the channel is composed of mainly sand in the upper 2 km of the estuary and for around 1 km upstream from the marine boundary. Elsewhere, sediment in the deepest part of the channel contains between 10% and 50% sand. The inter-tidal areas in the outer estuary, where the wave energy is greatest, are also composed of mainly sand. The distribution of salt marsh biomass at the end of the 500 year simulation period is shown in Figure 6.4.

Finally, it is noted that the continual increase in the volume of mud throughout the simulation period is expected, as a consequence of the 2 mm/year the sea level rise applied during this simulation.

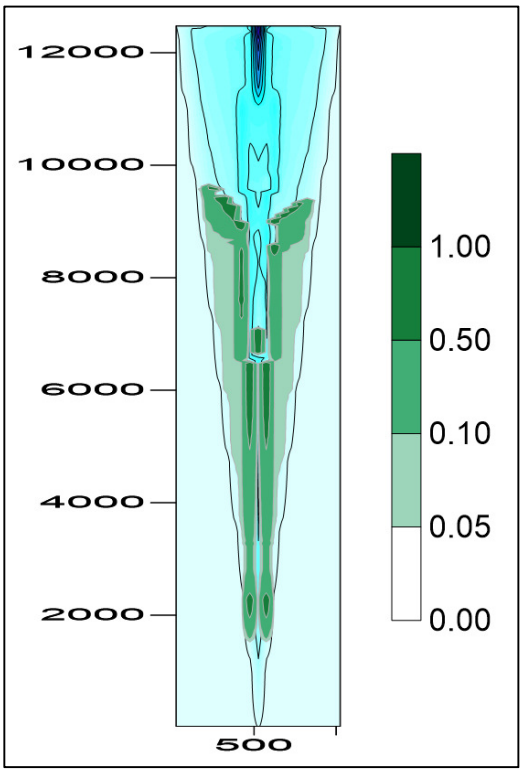

Figure 6.4: ***Salt marsh biomass distribution (shades of green) as a proportion of the maximum biomass density (2000 g/m^2) following the initial 500 year simulation period***

6.3 Baseline Scenario

In the baseline scenario the initial 500 year simulation was continued for an additional period of 50 years, with the rate of sea level rise maintained at 2 mm/year. Figure 6.5 shows the change in total sediment volume occurring during this simulation.

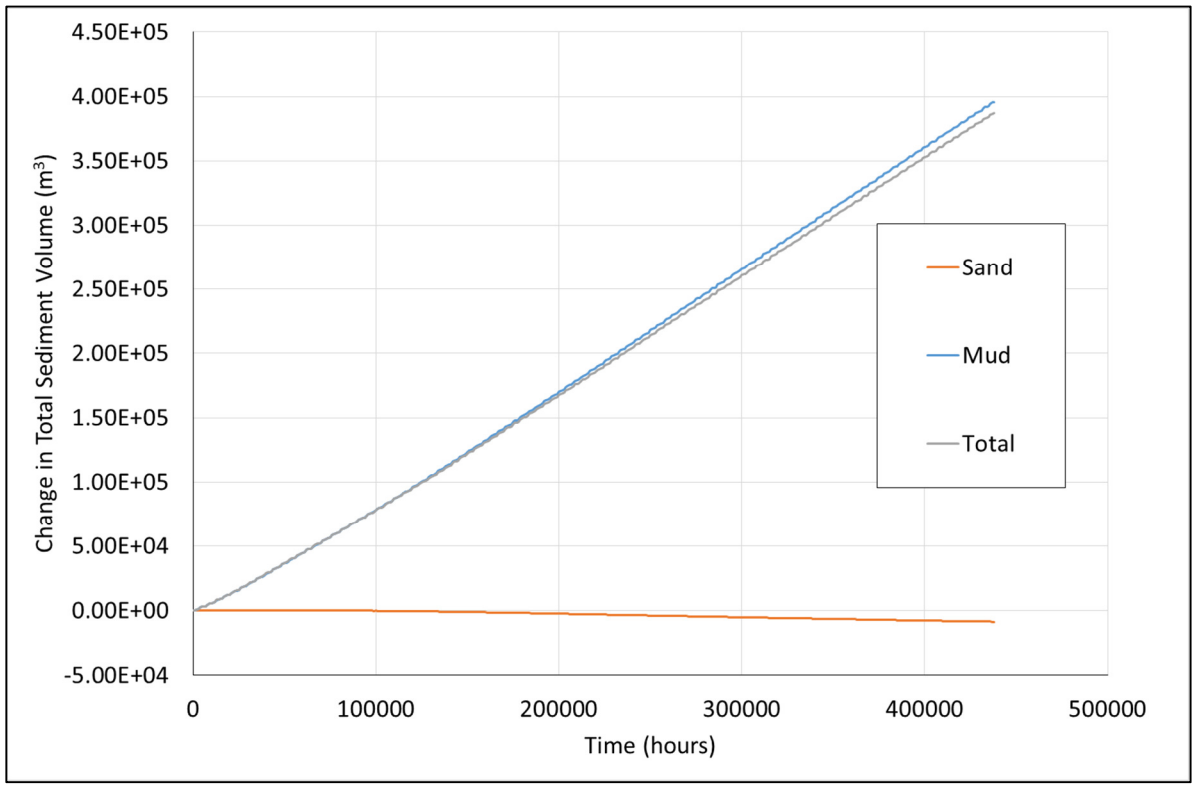

Figure 6.5: Change in total sediment volume during the baseline simulation

Sand volume has remained almost constant during this simulation, while the rate of mud deposition has increased to roughly 8,000 m^3/year, from an average of around 4,000 m^3/year during the initial 500 year simulation. At present the reasons for this are unclear; however, the rate of deposition in the initial simulation is quite varied and it is possible that this would have occurred anyway, had that simulation been continued. Further details of the baseline results will be included in the following sections, for comparison with the results from each test scenario.

6.4 Accelerated Sea Level Rise Scenario

In this scenario the rate of sea level rise was increased to 10 mm/year for a period of 50 years. This is higher than the estimate given in the IPCC Fifth Assessment Report (Section 2.2.6) and is selected for illustrative purposes only. Increasing the mean sea level causes water depths to increase, leading to a reduction in flow velocity and a reduction in near bed wave orbital velocities; hence deposition is expected to occur. Therefore, with a sufficient

supply of sediment, deposition rates might be expected to mirror the rate of sea level rise. The change in total sediment volume during this simulation is shown in Figure 6.6.

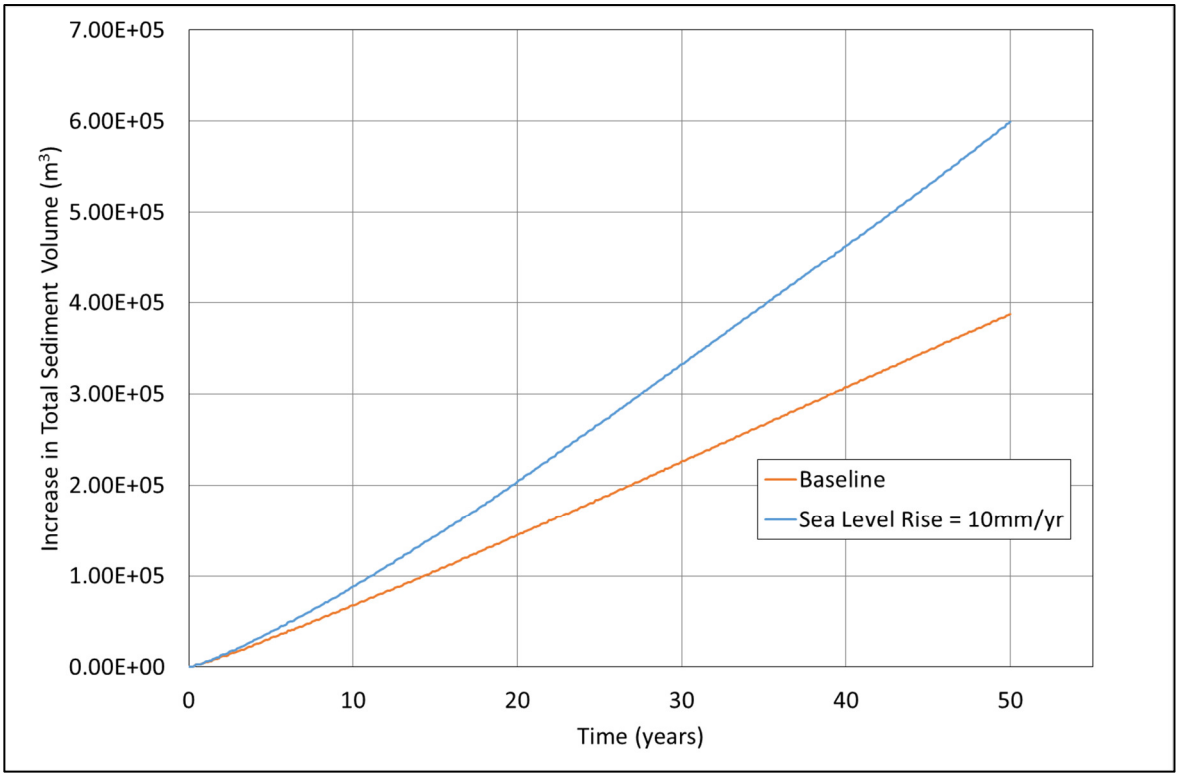

Figure 6.6: Accelerated sea level rise: change in total sediment volume

Figure 6.6 confirms that the rate of deposition has indeed increased under conditions of accelerated sea level rise. Figures 6.70 to 6.10 show cross section results from these simulations.

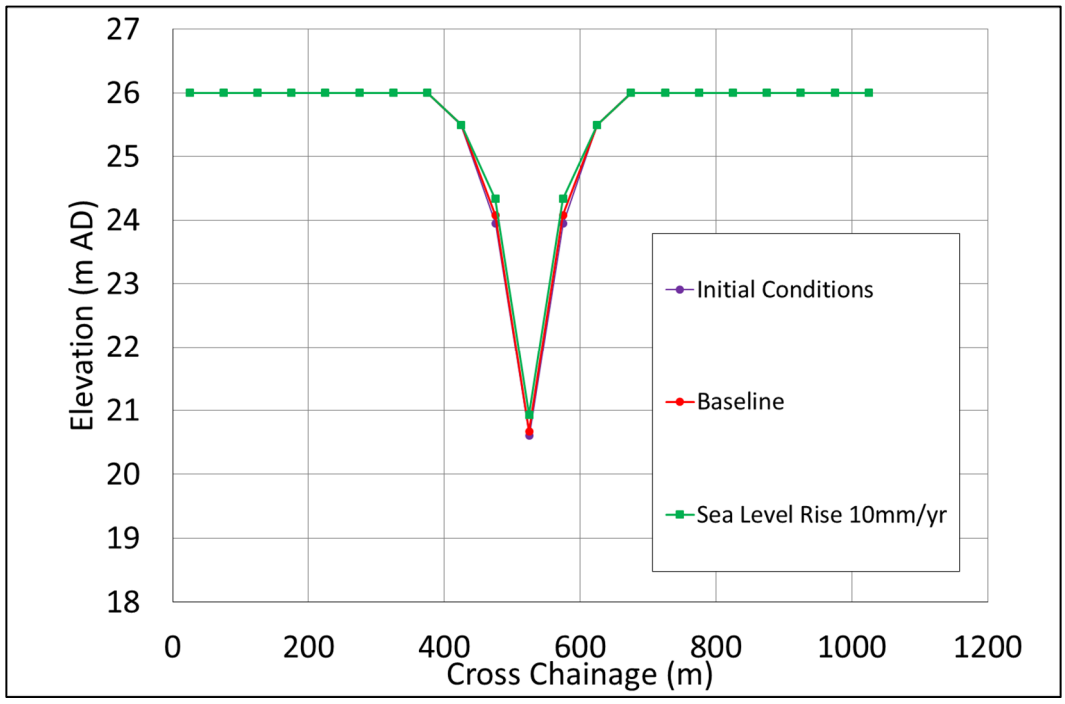

Figure 6.7: Cross section results at chainage 2,500m: accelerated sea level rise

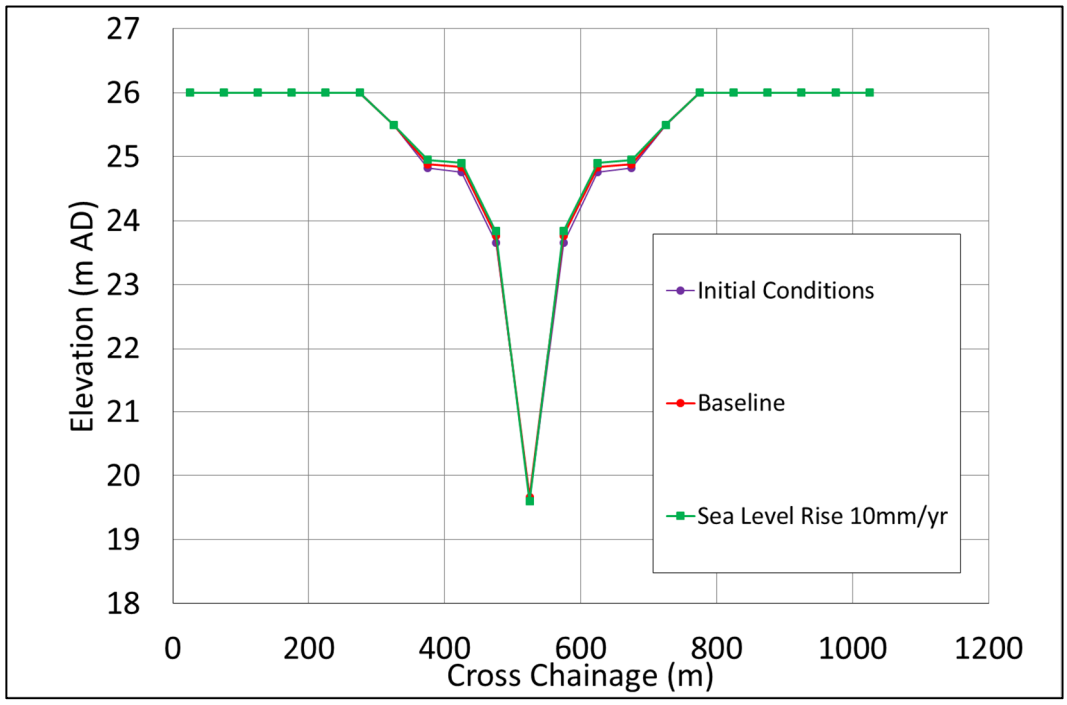

Figure 6.8: Cross section results at chainage 5,000m: accelerated sea level rise

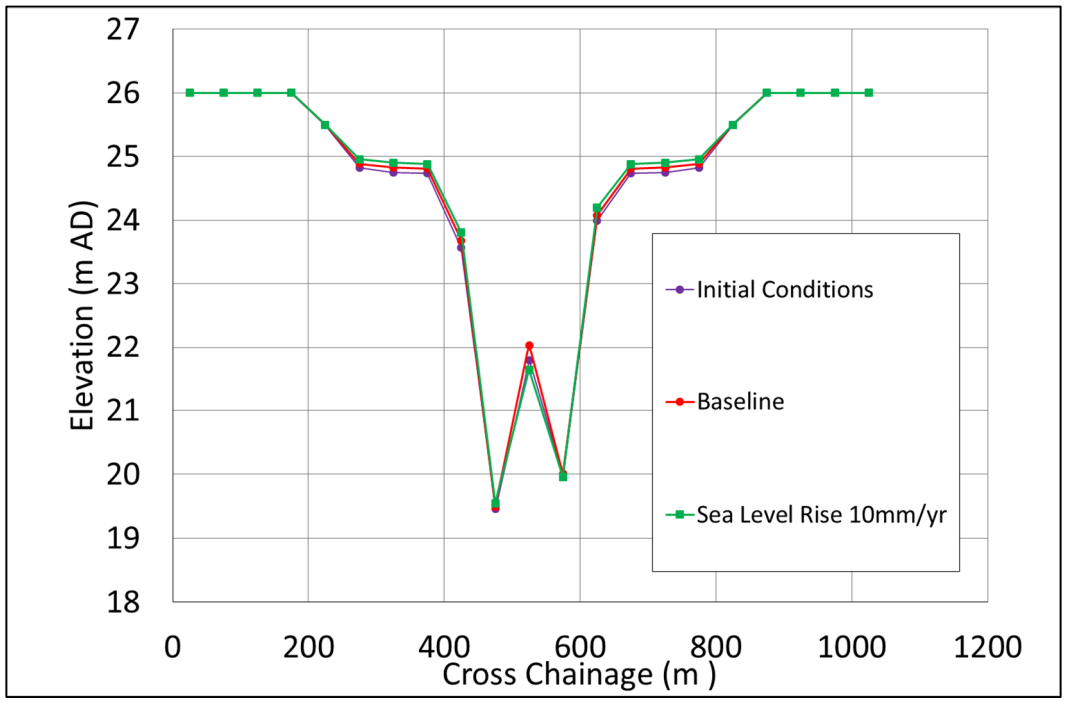

Figure 6.9: Cross section results at chainage 7,500m: accelerated sea level rise

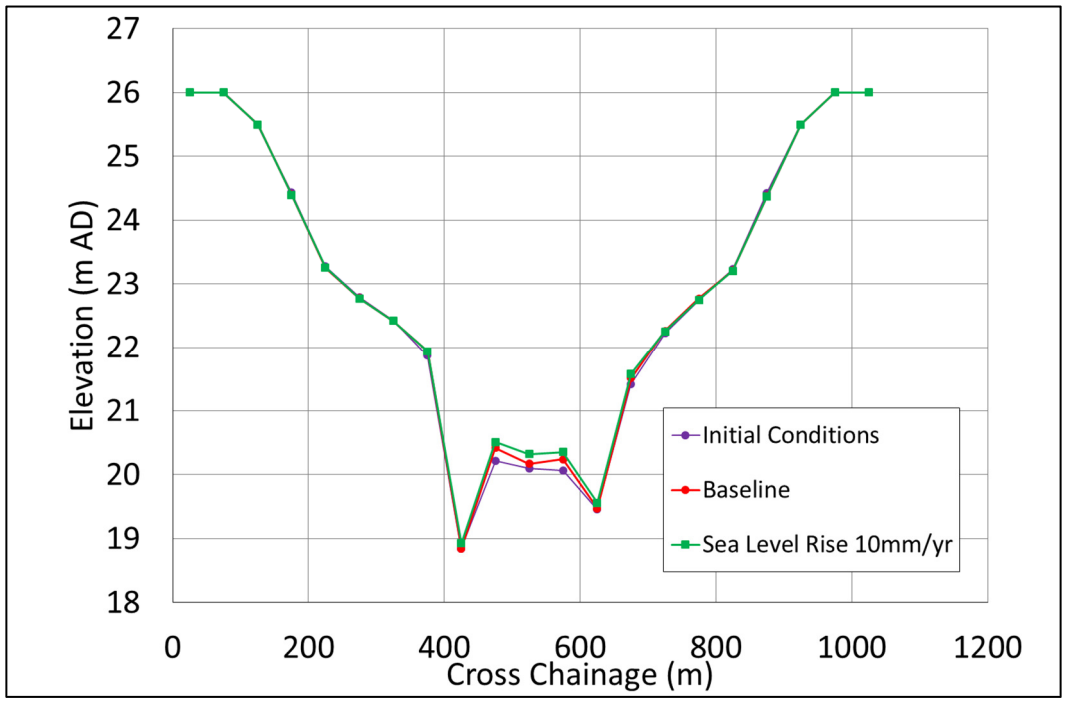

Figure 6.10: Cross section results at chainage 10,000m: accelerated sea level rise

Total sea level rise during the baseline simulation is 100mm and deposition on the marsh was found to be in range 60 to 120 mm, with the higher values occurring on the lower marsh and lower values occurring at the landward edge of the marsh. Sedimentation in the

middle of the marsh was found to be around 80 mm. While the total deposition is generally higher in the accelerated sea level rise scenario, typical values were only around 150mm and maximum values around 380 mm. Since the total sea level rise in this scenario is 500mm, this indicates that aggradation of the marsh may be limited by the availability of sediment. In this case sedimentation is not keeping pace with sea level rise and this can result in the eventual drowning of the marsh. If the marsh is unable to spread landwards (e.g. due to the presence of defences) this scenario would also lead to a loss of marsh area (coastal squeeze).

6.5 Dredging Scenario

In this scenario a navigation channel is dredged to a depth of 10m below the spring low tide level, with a new bed level of 11m AD. The channel is 50m (1 cell) in width and extends for 5 km upstream from the marine boundary. The changes in total sediment volume during this simulation are shown relative to the initial volume in the baseline scenario in Figure 6.11.

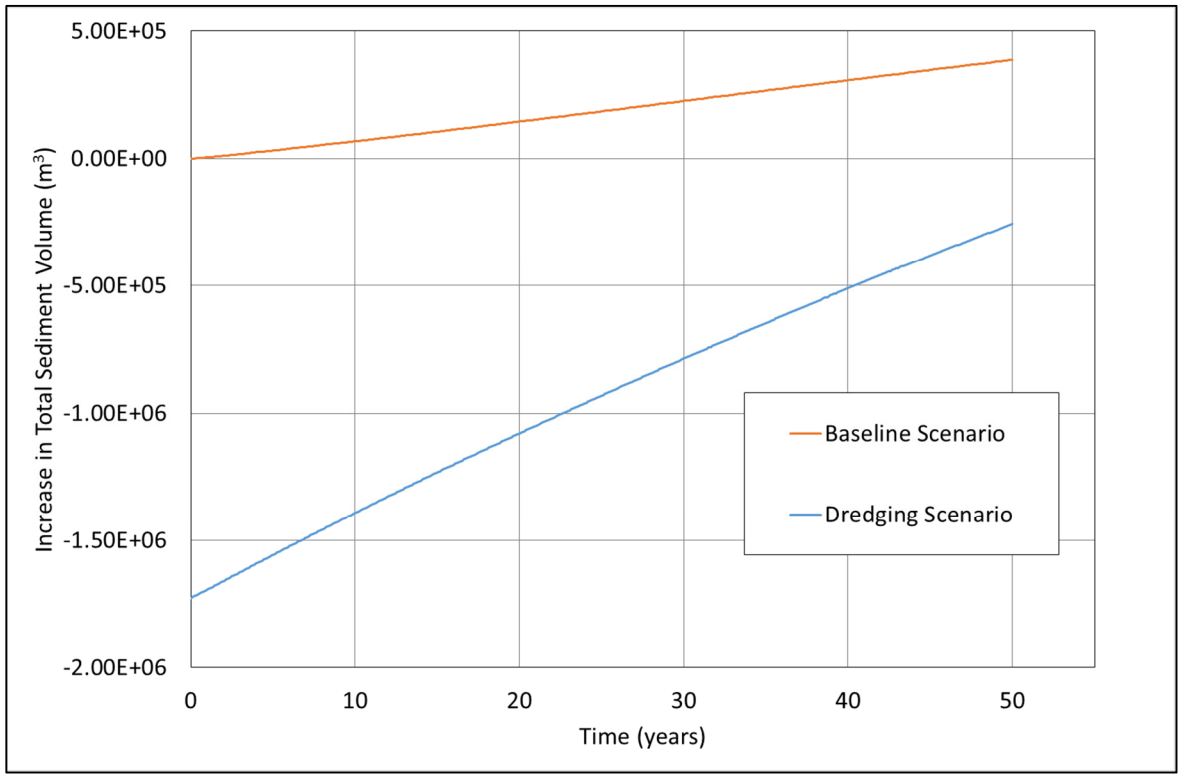

Figure 6.11: Dredging scenario: change in total sediment volume

Approximately 1.75 million cubic metres of sediment was removed from the initial conditions in this scenario, to create the dredged channel and by the end of the simulation period the difference in total sediment volume between the baseline and dredging scenarios has narrowed to around 0.65 million cubic metres. The dredged channel has therefore resulted in enhanced sedimentation. Figures 6.12 to 6.14 show cross section results at chainages 8,000m, 10,000m and 12,000m.

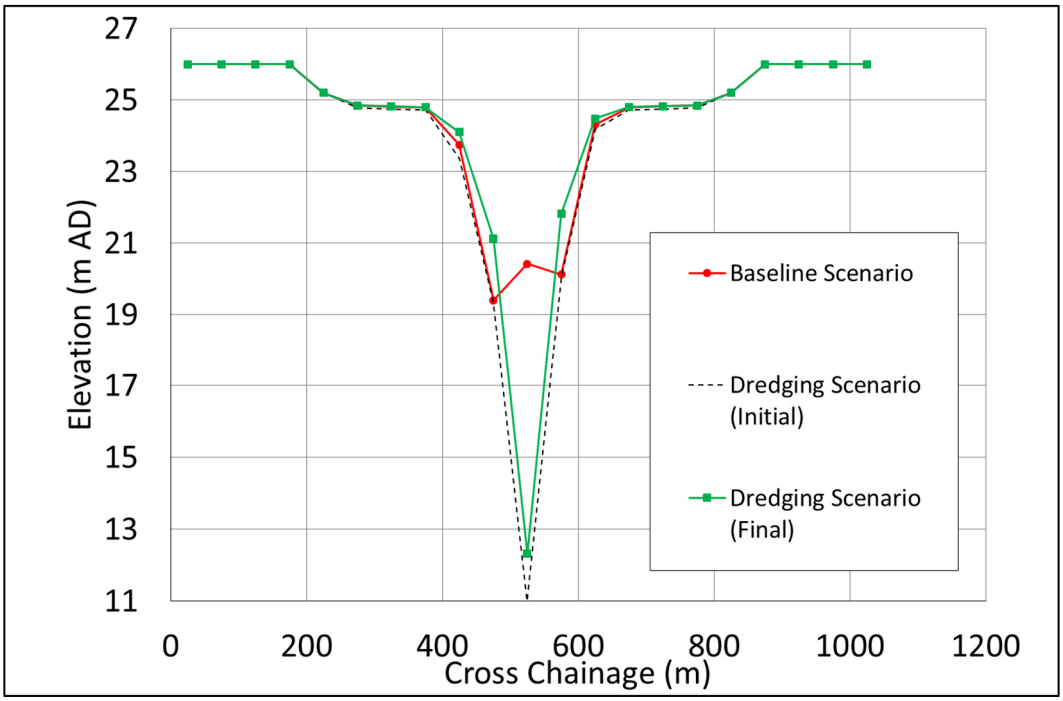

Figure 6.12: Cross section results at chainage 8,000m: dredging scenario

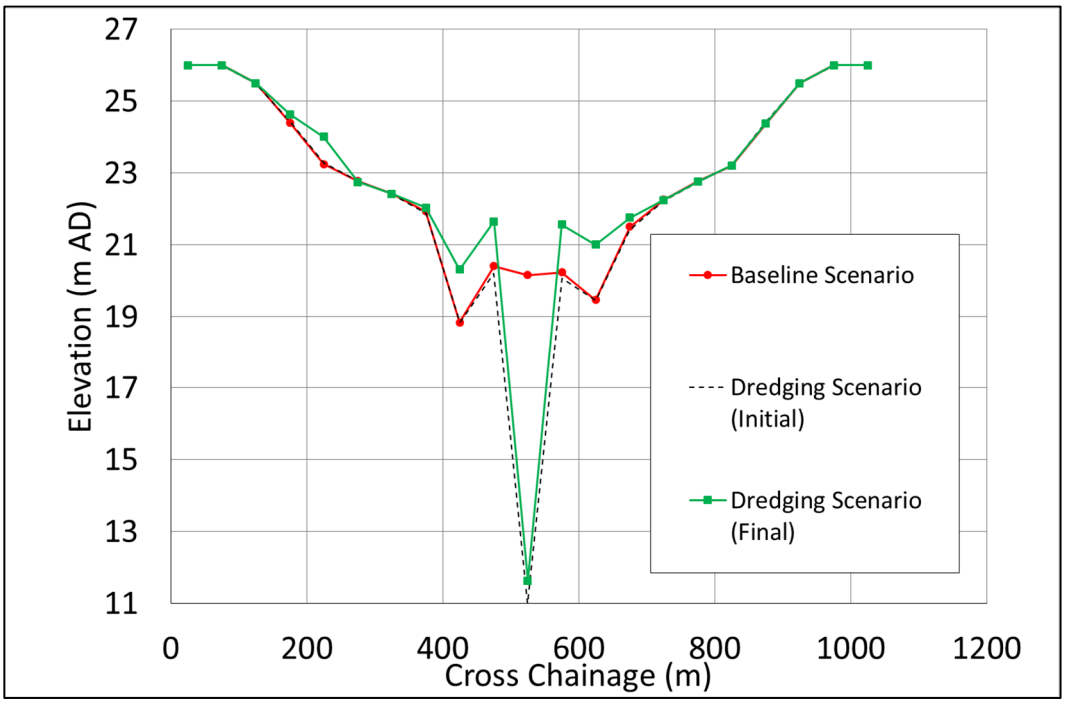

Figure 6.13: Cross section results at chainage 10,000m: dredging scenario

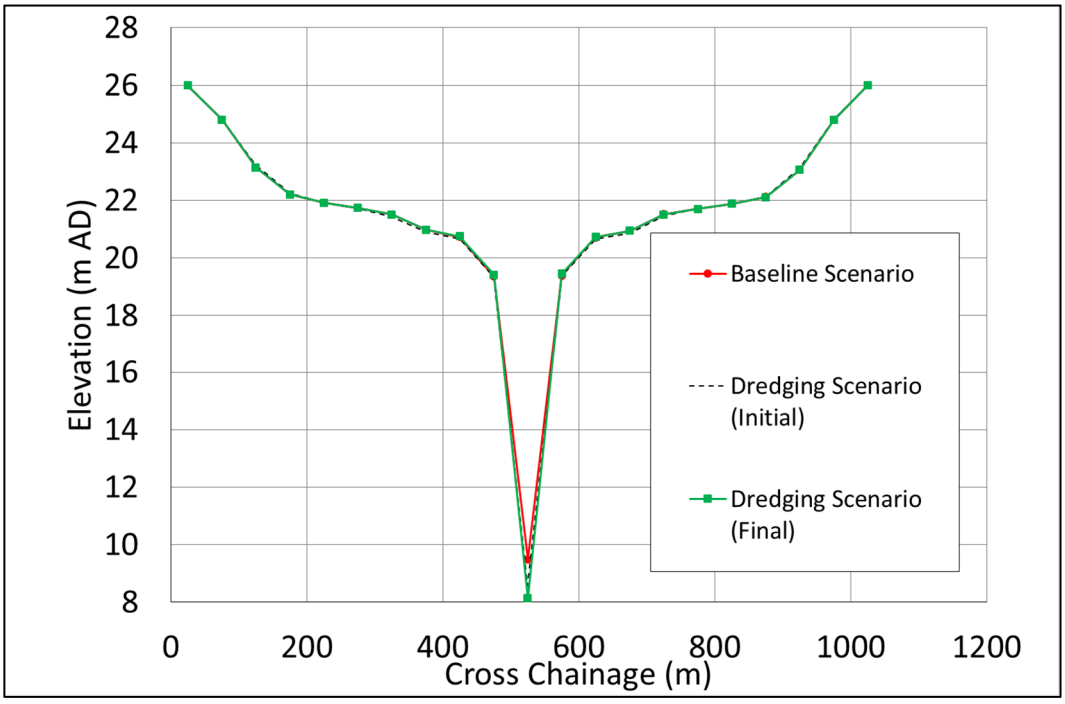

Figure 6.14: Cross section results at chainage 12,000m: dredging scenario

While some deposition has occurred in the channel, at chainages 8,000m and 10,000m, a greater part of the enhanced deposition has occurred in the sub-tidal area adjacent to the

channel. This is probably due to a reduction in flow velocity (and hence bed shear stress) in these cells, due to increased flow capacity in the dredged channel.

The increase in flow within the channel will tend to cause increased velocity and sediment transport within the channel, which would cause sediment to be drawn into the deep channel, due to downslope lateral transport; however, this effect is counteracted by the increase in depth, which reduces the velocity, due to the increase in cross sectional area, and reduces the shear stress due to waves on the bed. In this case the latter effect is dominant and infilling of the channel has occurred due to the deposition of fine suspended sediment (mud). Example time series of axial velocity and depth in the channel, at the end of the simulation period, are given in Figure 6.15, for a single tidal cycle, at spring tide.

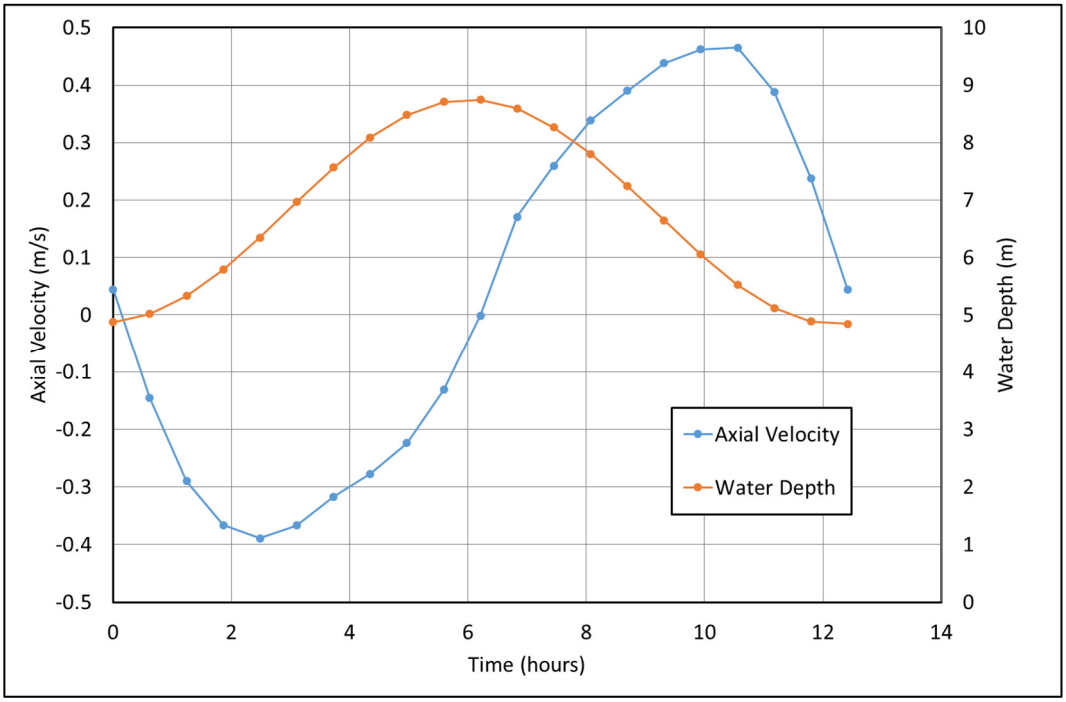

Figure 6.15: Dredging scenario: velocity and depth time-series in the channel, at chainage 12,000m, at spring tide

Figure 6.16 shows an example velocity vector field from the beginning of the simulation period, giving tidal flow velocities at the beginning of the ebb tide. Flow velocities are generally quite low at this time but are lower in the dredged channel (chainage > 7,500m) than in the unmodified channel further upstream. Green shading in this figure relates to salt marsh biomass density.

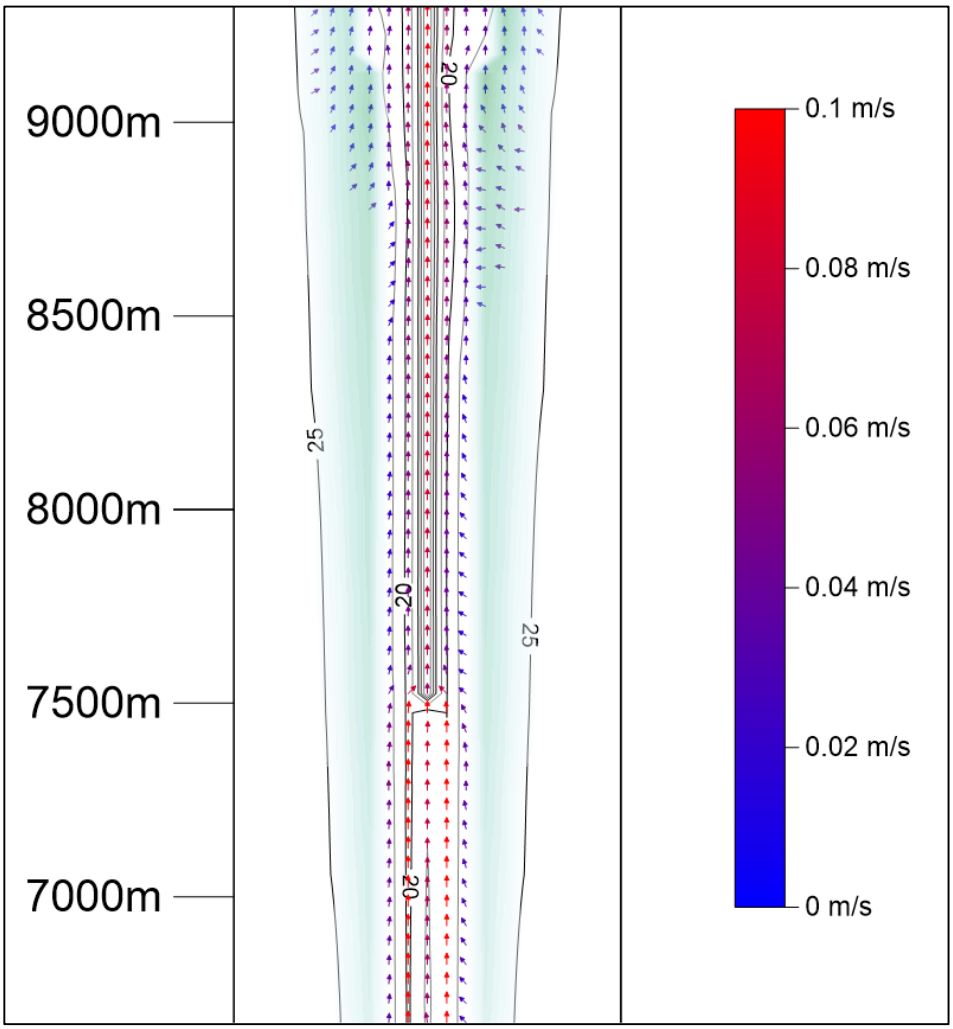

Figure 6.16: Dredging scenario: example vector field taken during the ebb tide at the beginning of the simulation period.

At chainage 12,000m the initial bed levels are unchanged from those in the baseline scenario, since they are already below the level required for the dredged channel. Results at this location are similar to the baseline scenario results; however, the channel depth has been reduced by around 0.5m in the dredging scenario and increased by around 0.9m in the baseline scenario. This is probably caused by a reduction in the supply of eroded sand to this area, in the dredging scenario, due to increased depths and reduced transport from further upstream.

Figure 6.17 shows the final bathymetry and depth of mud in the active layer at the end of the dredging scenario simulation, confirming that the deposition occurring both within and

adjacent to the dredged channel is composed predominantly of mud (the dredged channel extends from chainage 7,500m to chainage 12,500m).

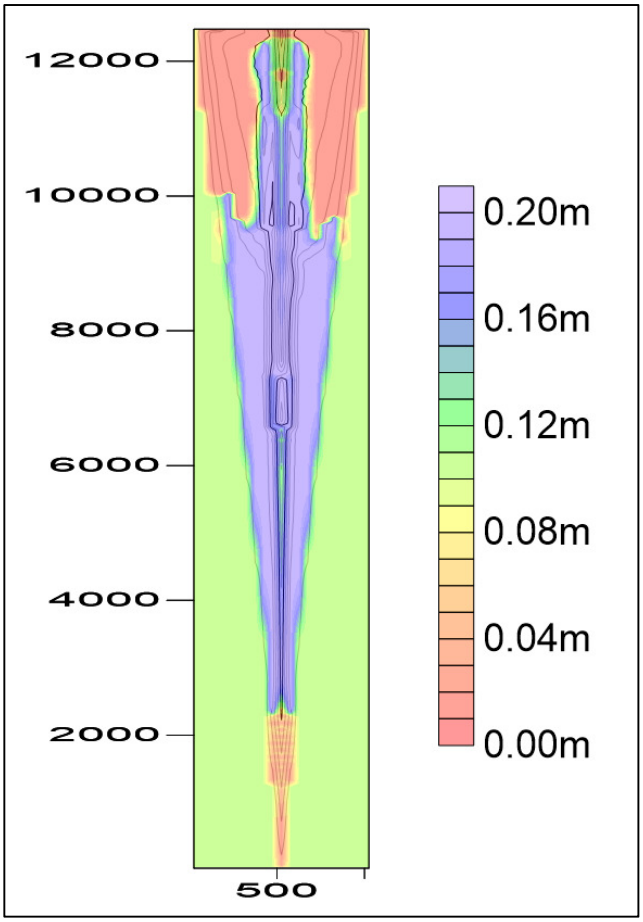

Figure 6.17: Contour plot showing the final bathymetry (contour lines) and depth of mud in the active layer (shading) (out of 0.2m total active layer thickness)

6.6 Harbour Development Scenario

This scenario considers the construction of a harbour in the outer estuary and tests the ability of the model to predict sedimentation patterns and consequent changes to the morphology over a period of 50 years. Figure 6.18 shows the modified initial bathymetry with an example vector field giving flow velocities during the flood tide. Figure 6.19 shows the modified bathymetry with wave heights for a wind speed of 15 m/s and direction of zero degrees (i.e. wind blowing directly towards the marine boundary, at chainage 12,500m).

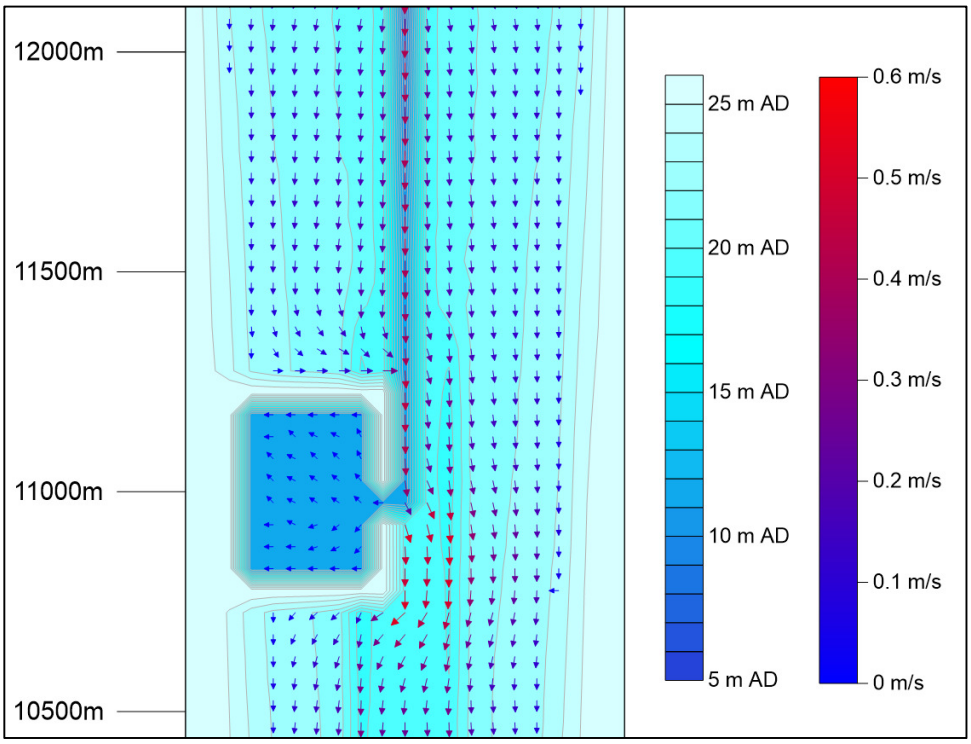

Figure 6.18: Harbour development scenario: modified bathymetry and example flow velocity field

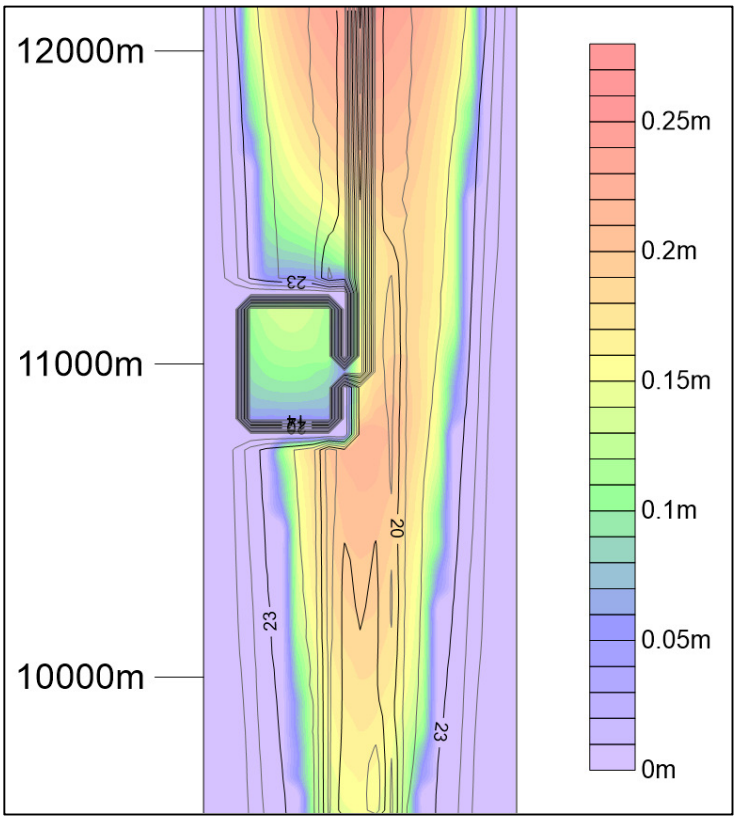

Figure 6.19: Harbour development scenario: modified bathymetry (contour lines) and wave heights for a wind speed of 15 m/s and direction of zero degrees (shading).

In this scenario the harbour walls were created by raising the initial and minimum bed levels in the relevant cells to 26.0m AD. Bed levels within the harbour were reduced to 11.0m AD and the bed of the channel leading into the harbour was also lowered to 11.0m AD. The change in sediment volume during this simulation is shown in Figure 6.20.

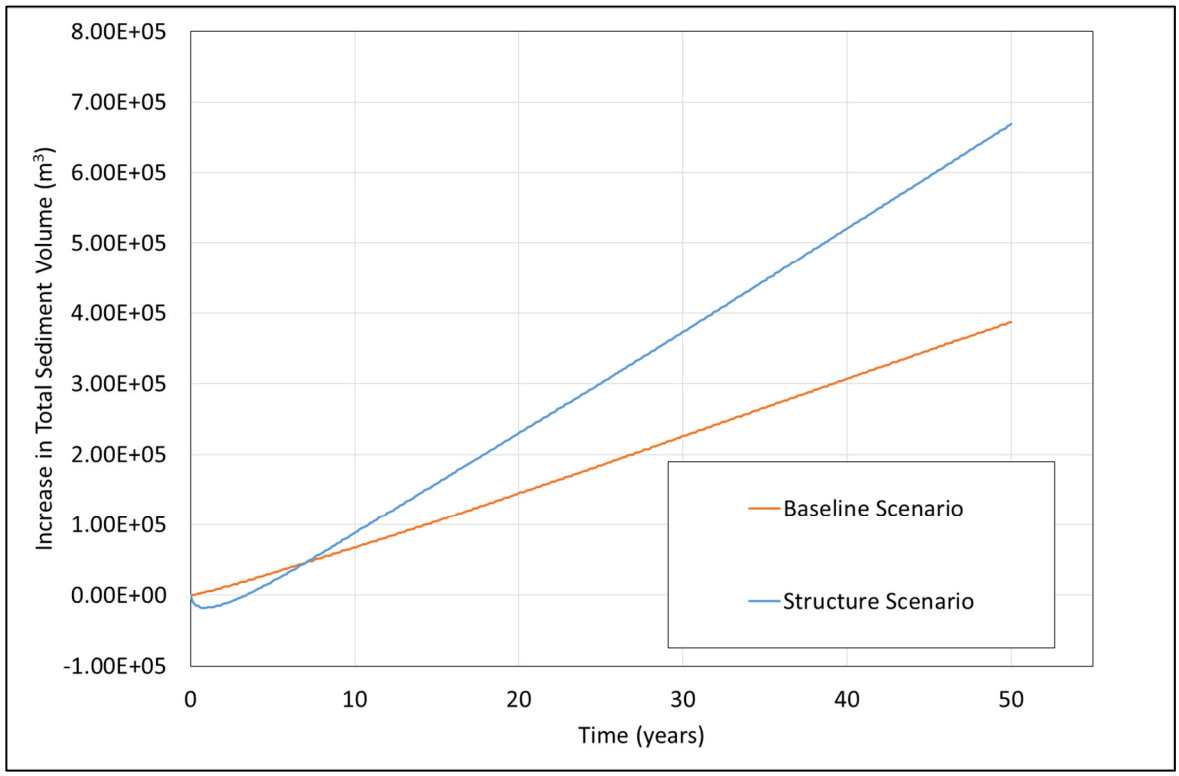

Figure 6.20: Harbour development scenario: change in total sediment volume

Figure 6.20 shows that, following an initial reduction in sediment volume, sedimentation has occurred at a faster rate in the harbour development scenario, resulting in a final sediment volume significantly greater than in the baseline scenario. Figure 6.21 shows the final bathymetry in the vicinity of the harbour, together with an example velocity vector field, this time during the ebb tide.

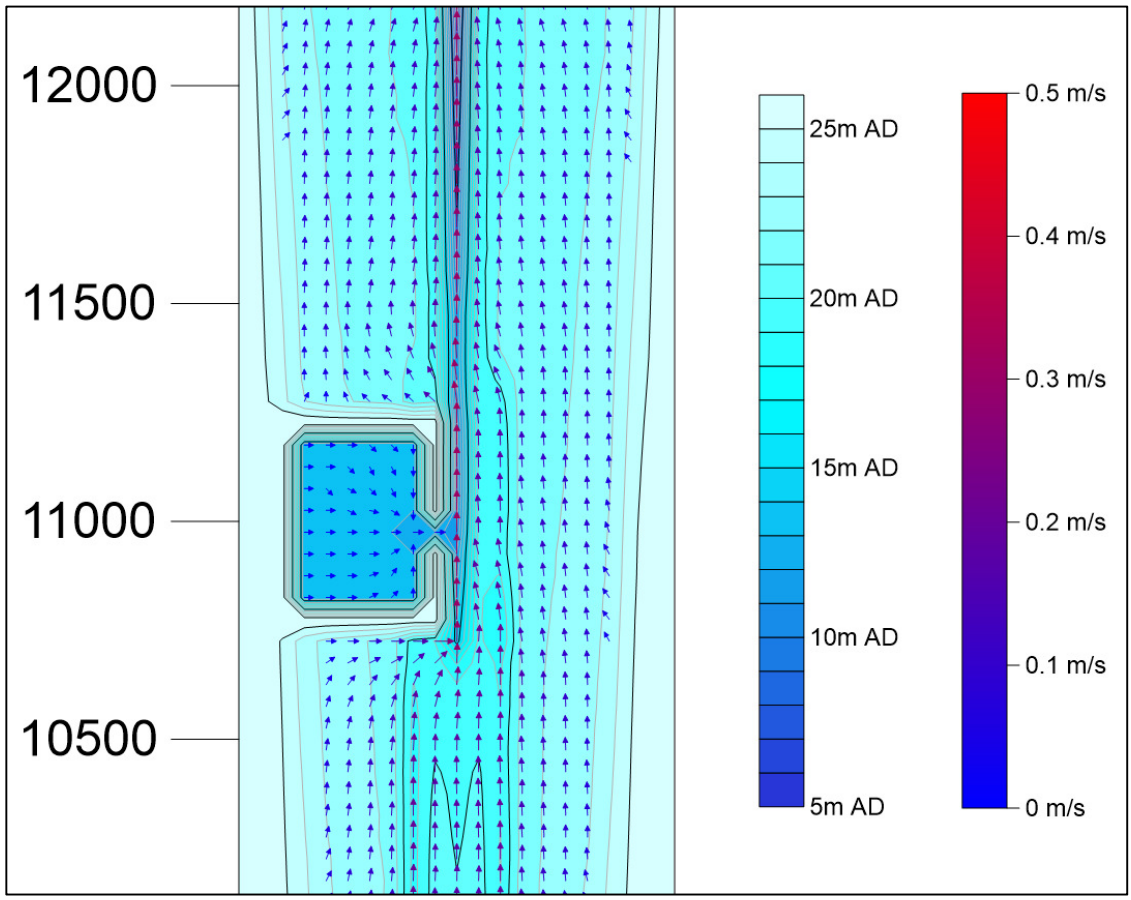

Figure 6.21: Harbour development scenario: final bathymetry

Approximately two metres of sedimentation has occurred within the harbour due to sheltering from waves and low flow velocities. Sedimentation within the harbour is significantly higher than observed on the marsh areas in the accelerated sea level rise scenario because the bed is continuously submerged in this case. The channel leading into the harbour has retained its original depth and has been extended beyond the harbour entrance, as shown in Figure 6.22, due to increased tidal flow velocities in this area. These changes are further illustrated by cross sections at chainages 10,900m (upstream of the harbour entrance) and 11,100m (downstream of the harbour entrance) in Figures 6.22 and 6.23 .

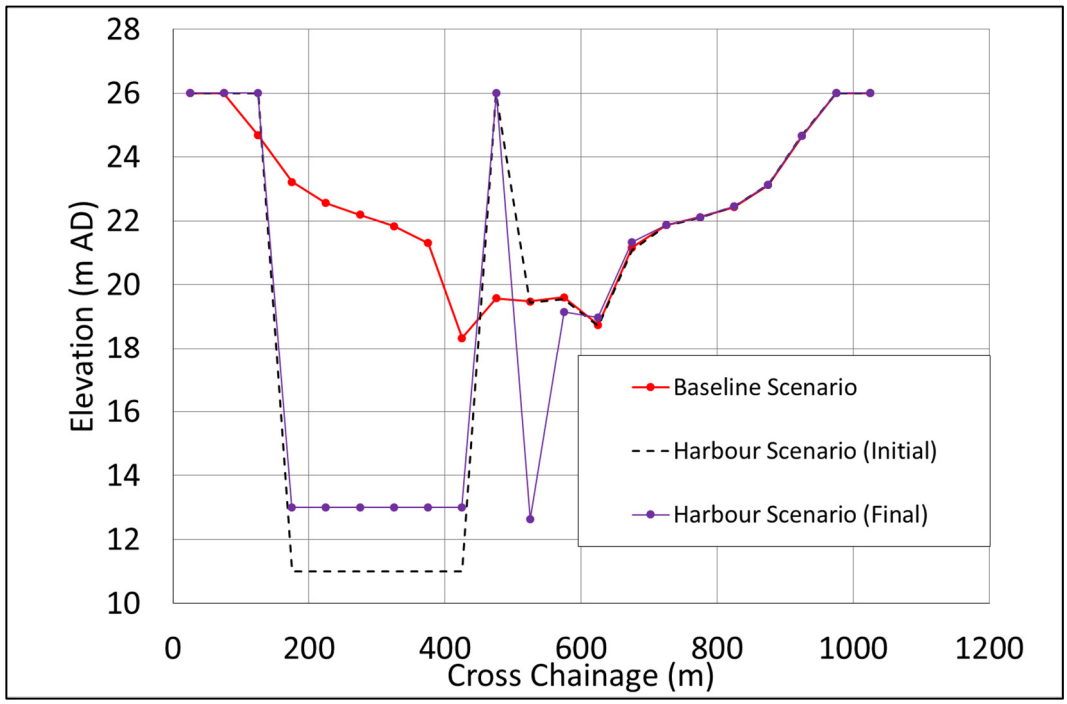

Figure 6.22: Harbour development scenario: cross section results at chainage 10,900m

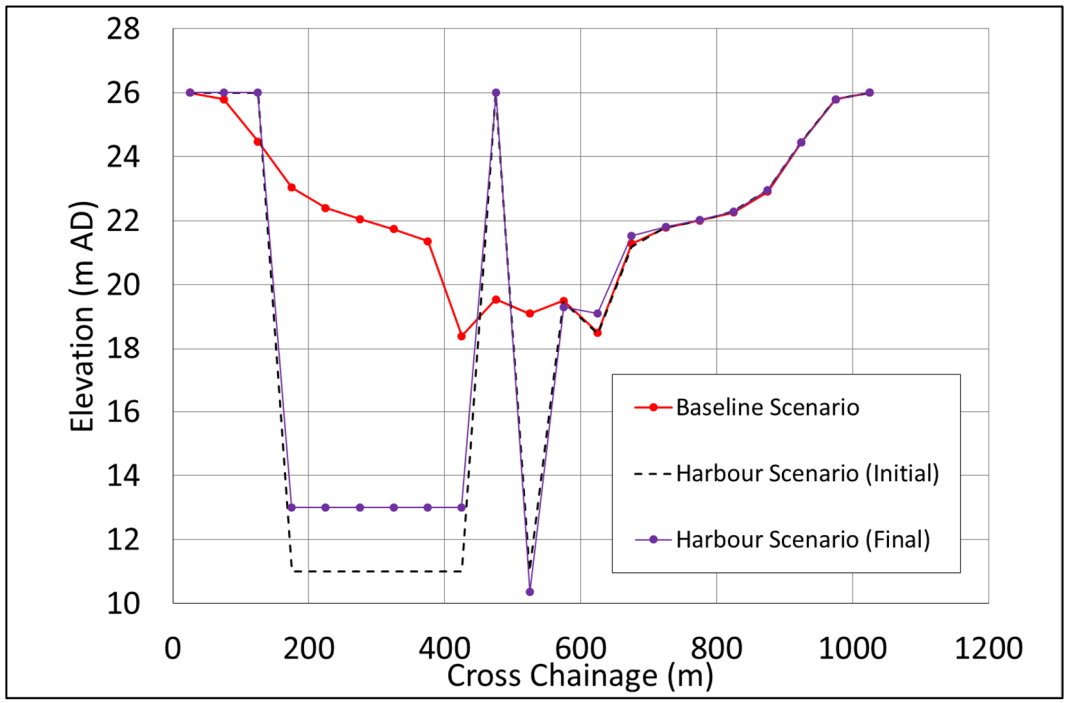

Figure 6.23: Harbour development scenario: cross section results at chainage 11,100m

Another change induced by the harbour construction has occurred in the area around chainage 9,500m. This is the location of the furthest extent of the salt marsh in the direction of the marine boundary. The marsh in this area was found to expand at this

location in both the baseline and harbour development scenarios but by a greater amount in the harbour scenario and this is assumed to be a caused by reduced wave penetration for landward wind directions (i.e. wind blowing upstream from the marine boundary). Figure 6.24 shows the difference in results at chainage 9,400m.

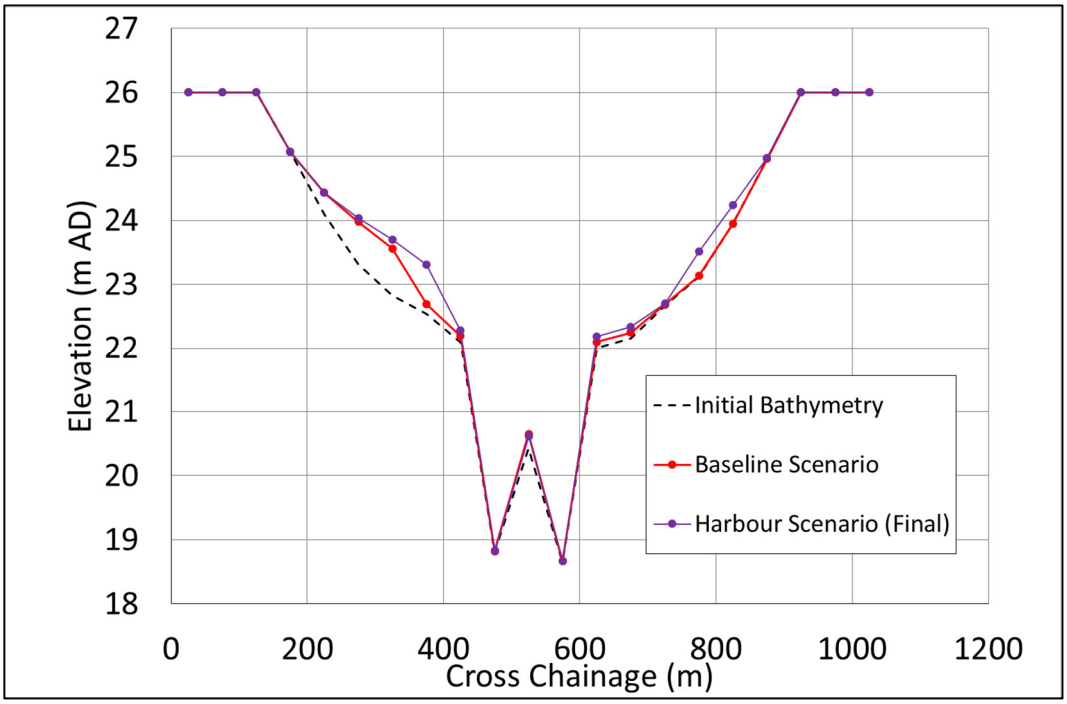

Figure 6.24: Structure scenario: cross section results showing increased deposition at chainage 9,400m

6.7 Comparison with the Deben Estuary

While the overall dimensions of the generic estuary used in the sensitivity and scenario testing was based on the Deben estuary, the results were not expected to reproduce the behaviour of this estuary due to differences in the initial bathymetry and hydrodynamic forcings. However, a comparison with the real behaviour of the Deben over the last several decades may still be useful.

Recent papers on the Deben (e.g. Birningham and French, 2006) have focussed on the ebb tidal delta and little information on the behaviour of the middle and inner estuary is available in the literature. Frostick and McCave (1979) report that significant seasonal variation occurs to mud flat levels in the Deben but that no discernible longer-term trend in mud flat levels could be identified.

Aerial photography showing a section of the estuary is given in Figure 6.25 for the year 1945 and in Figure 6.26 for the year 2013. Apparent differences in the width of the channel between the two images are likely to be related to the tidal water level at the time each photograph was taken, rather than any change to the morphology, and the extents of the mud flats and salt marsh otherwise do not appear to have changed significantly. Therefore, the limited available evidence does not suggest any trend of morphological change over the last few decades. This is contrary to the results of the baseline sensitivity test in Chapter 5 (which shows significant erosion) but is consistent with the baseline scenario in Chapter 6, which shows accretion on the marsh and flats broadly keeping pace with sea level rise, as discussed in Section 6.4.

Figure 6.25: Aerial photography showing the Deben Estuary in 1945

Figure 6.26: Aerial photography showing the Deben Estuary in 2013

6.8 Discussion

Tidal flows, wave, sediment transport and salt marsh are represented in the model using approximate, simplified procedures; however, these procedures have been shown able to capture a number of key interaction and feedback effects, including:

- Mass continuity of tidal flow: any reduction in cross sectional area or increase in the upstream tidal prism causes the cross sectionally averaged velocity to increase (and vice versa), which will tend to prevent any further reduction in area by increasing the shear stress on the bed and hence reducing deposition or generating erosion (a negative feedback).

- Bed friction: a localised increase in bed level will tend to reduce the velocity due to the relative increase in the effect of bed friction and the consequent diversion of flow into deeper water, which may lead to increased deposition (a positive feedback).

- Changes to the cross-sectionally averaged depth affect the tidal wave propagation and hence influence peak velocities on the flood and ebb tides. In principle this can allow the model to capture processes such as a change from ebb dominance to flood dominance.

- The relationship between the morphology, fetch, waves and sediment transport: where the wave height exceeds the depth limited wave height, both the local wave height and down-wind wave heights are reduced; hence, erosion is reduced or deposition enhanced in these areas. This allows the sheltering effect of features such as shoreline irregularities, spits or artificial structures to be represented (even when they are submerged).

- The relationship between salt marsh and the morphology. Where the bed elevation is in the range for which salt marsh growth occurs, bed friction is increased, wave height is reduced and sedimentation is enhanced.

The above interactions control geomorphic features such as channels, tidal flats and salt marshes. Examples of this in the test scenarios include enhanced sedimentation on the tidal flat, due to accelerated sea level rise or sheltering from waves, and changes to the channel depth due to the redirection of flow around a structure.

The method used to calculate tidal and fluvial flows is based only on mass continuity and bed friction but this simplicity has resulted in a model that is robust and computationally

efficient, while maintaining some of the principal feedback relationships with the morphology. Secondary flows at meander bends, the Coriolis Effect and density driven circulation are not represented in the present model and it has been assumed that these effects will have only a minor influence on the morphology, without inducing any changes to the long term behaviour of the system; however, it is acknowledged that in some cases density driven circulation can affect long term behaviour, by inducing a net inward movement of sediment near the bed (e.g. in the Mersey Estuary: Thomas et. al., 2002).

Tidal water levels for the simulations described in this chapter were calculated using Equation 4.13, which assumes a synchronous tidal regime. It is noted that in some cases this assumption may not be appropriate and an option to use a 1D numerical model to calculate water levels has been developed. This option has only a small effect on the computational efficiency and provides a more general tidal wave propagation solution and should therefore be considered during any future use of the model.

The method adopted for the flow field calculation (the Hardy-Cross method) has been shown to be reliable and robust; however, model testing has shown that computation times for this method increase rapidly when the number of cells is increased. This method was chosen initially for ease of implementation during model development and it is noted that more computationally efficient methods exist for solving this type of problem; for example, those described by Wood and Charles (cited by Featherstone and Nalluri, 1988) and Kutija (1995). Improvement of this component should therefore be investigated during any future development of the model.

The method used to calculate the wave height and period is simple and robust. It also incorporates approximations of the effects of wave breaking in shallow water and diffraction around obstacles. Wave energy spreading laterally to the edges of the submerged area is assumed to be partially reflected (by an amount specified by fixed model parameter) and when this reflection is set to 100% the method tends to give a result close to the straight-line fetch as recommended in the Coastal Engineering Manual (US Army Corps of Engineers, 2002). Although the Coastal Engineering Manual does not recommend the modification of fetch lengths for narrow fetches, it seems likely that some

wave energy will be lost due to breaking in shallow areas, in cases where the shore profile is not very steep.

Only locally generated waves are included in the present model. Higher average wave energy can be applied in the outer estuary by increasing the fetch length assumed at the marine boundary; however, the outer regions of estuaries are often affected by ocean waves with different distributions of wave height, period and direction to those generated locally by the wind. Ocean waves could be incorporated into the model using a parametric method such as that proposed by Tucker (cited by Dearing et. al., 2005) to derive local wave conditions based on the local depth and the deep water wave conditions, which would need to be provided as an input to the model.

Well established transport formulae have been adopted in this research; however, uncertainties associated with these methods are combined with assumptions and approximations associated with their application within the model. For example, the method used to calculate the erosion and deposition of mud has been simplified by assuming a constant bulk density (ignoring consolidation effects) and the treatment of mixed sediment has been simplified by assuming a simple linear relationship between the relative proportion of sand and mud, and the critical bed shear stress for erosion. The treatment of lateral downslope transport has also been greatly simplified and further work is needed to determine the effect of these simplifications on the morphological results.

At present the model does not include the transport of sand by waves (only enhancement by waves of transport due to currents is included). Since wave driven transport is typically important in the outer estuary, this limitation is likely to affect the model results in that area.

Salt marsh extent in the present model is related only to inundation time; however, other factors such as wave exposure and sediment type may also affect the growth and stability of salt marsh. A robust calibration using field data will allow the validity of the model to be assessed and aid in the identification of other predictors of marsh coverage, which can be used to further improve the model.

In all three of the test scenarios described in this chapter, the results have confirmed that the model is able to predict future changes to the morphology resulting from an imposed change in conditions. The model has successfully captured the qualitative morphological change; for example, marsh aggradation was expected to increase due to accelerated sea level rise because this increases the depth of water and consequently reduces the shear stress on the bed. An increase in water depth also increases the accommodation space (see Section 3.3.1). Similarly, deposition within the channel in Scenario 2, the harbour in Scenario 3 and erosion of the channel adjacent to the harbour in Scenario 3 are realistic when considering the likely effect on bed shear stresses. Further work is needed however, to assess the quantitative performance of the model and its ability to reproduce observed changes in real estuaries, as discussed in the next chapter.

As intended this model falls between existing Top-Down and Hybrid type methods and traditional Bottom-Up, processed based modelling, in terms of computation time, input data requirements and output generated. The model predicts spatial variations in the evolution of the morphology, in constrast to 'lumped' Top-Down and hybrid models, such as ASMITA and since the model represents hydrodynamic and ecological processes it should also offer greater insight into the mechanisms causing particular morphological behaviour than is possible using empirically based Top-Down type methods.

Model run times are much shorter than for Bottom-Up type models (the 50 year simulations included in Chapters 5 and 6 took approximately 12 hours to complete using a desktop computer with an Intel i7 processor), which should also enable rapid testing of multiple scenarios to assess the sensitivity of the morphology to variations in the inputs (e.g. variations in future rates of sea level rise). Input files can be generated using GIS software such as ArcMap from available datasets by interpolating between known data points to create a Triangular Irregular Network (TIN) model of the ground surface, underlying bedrock, initial distribution of sediment types and salt marsh coverage, and exporting the resulting surfaces in ESRI ASCII grid format. Other inputs are sediment properties (d_{50}, d_{90}, porosity and critical shear stresses for the erosion and deposition of mud), mean river discharge, wind speed and standard deviation and the fetch to be applied at the open boundary.

Calibration of model runs may be achieved by varying some of the above input parameters, where there is uncertainty in the adopted values or by varying a number of other model parameters, including:

- The sediment diffusion coefficient (D).
- The boundary equilibrium concentration (C_{ds}).
- Rate of sand input at the marine boundary (S_{in})
- The scaling factor for lateral sediment transport (F_{lat}).
- The critical shear stress scaling factor for mixtures of sand and mud ($\tau_{e,max}/\tau_{cr,sand}$).
- Minimum and maximum values for the Chezy roughness coefficient (C).

The cell size, time-step and active and sub-layer thickness can also adjusted if required to suit the requirement of a particular simulation.

Water levels may optionally be calculated using a 1D hydrodynamic model (as opposed to Equation 4.13) if the hydraulic modelling software ISIS is installed on the computer to be used for the simulations. Run times for the 1D model are only a few seconds and these are repeated infrequently during the simulation. Cross sections are generated from each row of bathymetry data and the free version of ISIS is limited to 200 cross sections; however, an option has been included to represent the bathymetry using fewer cross sections than the number of rows in the model domain (e.g. one in every two rows). A further option has been developed to allow the use of cross sections interpolated perpendicular to a predefined centreline, as described in Section 4.3.4.

CHAPTER 7

CONCLUSIONS AND RECOMMENDATIONS

7.1 Conclusions

This thesis describes the development and validation of a Cellular Automata based morphodynamic model for the prediction of medium term morphological change in estuaries. This includes the development of new, simplified and efficient methods for the estimation tidal flows and waves and the adaption of existing methods for the sediment transport and salt marsh model components. The simplified methods used in this research have resulted in a model that is robust and computationally very efficient, while maintaining the principal interactions and feedbacks between the hydrodynamic processes, morphology and salt marsh ecology.

The sensitivity testing and test scenarios confirm that the model performance is qualitatively very encouraging. The model is not unduly sensitive to the computation time-step or cell size and it responds in a realistic manner to changes in the hydrodynamic forcings, sediment properties and boundary conditions. The model has also been shown to be capable of reproducing the evolution of some common geomorphic features (e.g. channel, tidal flats and salt marsh) and generating a realistic response to imposed changes, such as sea level rise or engineering works.

In contrast to existing empirical Top-Down and hybrid type models, this approach has the potential to make better predictions of future change due to its ability to incorporate interaction and feedback effects between processes and the morphology that may not be very well represented in the past behaviour of an estuary. While the model is not expected to be able to predict precise morphological changes, it should be able to indicate trends, as well as potential interactions, feedbacks and tipping points. At the time and space scales of interest, existing detailed process models have similar limitations but typically have far greater computational and data requirements. The flexibility of the cellular automata

approach also makes it relatively straight forward to incorporate additional processes and empirical rules where required.

At present the model has only been used to simulate hypothetical scenarios in a simple generic estuary. Therefore, the next stage in its development should include calibration and verification using empirical datasets from one or more real estuaries. It is expected that to reproduce the past behaviour of a real estuary some elements of the model components will need to be improved and refined. Representation of additional processes may also be needed, such as ocean waves or additional ecological processes. It is believed that, following the completion of this additional work, the model has the potential to become useful tool to aid strategic management and planning within estuaries.

7.2 Recommendations for Further Work

While the present model has been shown to produce realistic behaviour in a simple generic estuary, a robust calibration and verification process is needed to develop the capability to predict changes in real estuaries, including the timing and extent of such changes. This process is expected to include refinement of existing model components and may involve the addition new processes to the model.

First, the flow, wave and sediment transport model components should be tested individually against short term simulations using existing process models to confirm that these processes are adequately represented.

The sediment transport approach then needs to be assessed against established process based morphological models and, if necessary, modified to ensure that these processes are adequately represented. In particular, the methods developed to handle mixed sediment, down slope transport and the transport of fine suspended sediment need to be evaluated in this way.

The morphological evolution predicted by the model can be assessed using one or more datasets giving, as a minimum, a complete set of bathymetric data for an estuary as well as river flows, tidal range and wind speed characteristics. Additional information on geology, sediment characteristics and spatial extent of salt marsh may also be incorporated, if

available. When tested using a data for a real estuary the model should not produce significant large scale and rapid changes to the morphology, although some initial adjustment is expected due to the approximations and simplifications included in the model. The model should be able to reproduce observed trends, such as the infilling of channels or erosion of mud flats. The predictive capability of the model can be evaluated against short term simulations using existing process based models and longer term predictions using Top-Down type models, such as ASMITA.

Specific improvements that can be can made to the model during calibration may include the addition of ocean waves in the outer estuary and wave driven sediment transport. These processes are important in the outer parts of some estuaries and are not represented in present model. The salt marsh model component may be improved by refining the rules to better represent the observed marsh extents in real estuaries and where other estuarine organisms (e.g. micro algae) are found to have a significant effect on the morphology further rules may be added to predict their extents and effects.

REFERENCES

ABPmer, 2008a. Holocene Analysis. http://www.estuary-guide.net/pdfs/holocene_analysis.pdf (Last accessed, March, 2013).

ABPmer, 2008b. Form Analysis. http://www.estuary-guide.net/pdfs/form_analysis.pdf (Last accessed March, 2013)

ABPmer, 2008c. Tidal Asymmetry Analysis. http://www.estuary-guide.net/pdfs/tidal_asymmetry_analysis.pdf (Last accessed March, 2013).

ABPmer, 2008d. Inter Tidal Form Analysis. http://www.estuary-guide.net/pdfs/intertidal_form_analysis.pdf (Last accessed March, 2013).

ABPmer, 2008e. Estuary Translation (Rollover). http://www.estuary-guide.net/pdfs/estuary_translation.pdf (Last accessed March, 2013).

ABPmer, University of Plymouth, University College London, Discovery Software, HR Wallingford and Delft Hydraulics, 2008. Development and Demonstration of Systems-Based Estuary Simulators. R&D Technical Report FD2117/TR, Joint Defra/EA Flood and Coastal Erosion Risk Management R&D Programme.

Berto, F. and Tagliabue, J., 2012. Cellular Automata. In Edward N. Zalta (ed.), The Stanford Encyclopedia of Philosophy (Summer 2012 Edition). http://plato.stanford.edu/archives/sum2012/entries/cellular-automata (last accessed December 2014).

Bird, E., 2008. Coastal Geomorphology, An Introduction. 2nd Edition. Wiley. Chichester.

Booij, N., Ris, R.C. and Holthuijsen, L.H., 1999. A third-generation wave model for coastal regions. 1. Model description and validation. Journal of geophysical research, 104, C4, 7649-7666.

Boorman, L.A., 1999. Salt marshes – present functioning and future change. Mangroves and Salt Marshes 3: 227-241

Borum, J., Gruber, R. K. and Kemp, W. M., 2013. Seagrass and related submersed vascular plants, in Day, W. D., Jr, Crump, B. C., Kemp, W. M. and Yanez-Arancibia, A (Editors), 2013. Estuarine Ecology, 2nd Edition. Wiley-Blackwell, Hoboken, New Jersey.

Boyd, P.W., 2011. Beyond ocean acidification. Nature Geoscience, 4, 1-2.

British Standards Institution, 1990. BS 1377-2: Methods of test for soils for civil engineering purposes, Part 2: Classification tests. BSI, London.

Burningham, H. and French, J., 2006. Morphodynamic behaviour of a mixed sand-gravel ebb-tidal delta: Deben estuary, Suffolk, UK. Marine Geology 225 (2006) 23-44.

Castaing, P. and Guilcher, A., 1995. Geomorphology and sedimentology of rias, in Perillo, G.M.E. (Ed). Geomorphology and Sedimentology of Estuaries, Elsevier Science B.V. Amsterdam.

Centre for Ecology and Hydrology, 2013. http://www.ceh.ac.uk/data/index.html (Last accessed May 2015)

Chow, V.T., 1959. Open Channel Hydraulics. The Blackburn Press. Caldwell, New Jersey.

Coulthard, T. J., 1999. Modelling Upland Catchment Response to Holocene Environmental Change. Unpublished PhD Thesis, School of Geography, University of Leeds, U.K. 181pp.

D'Alpaos, A., Lanzoni, S., Marani, M. and Rinaldo, A., 2007. Landscape evolution in tidal embayments: Modelling the interplay of erosion, sedimentation, vegetation dynamics. Journal of Geophysical Research: Earth Surface (2003–2012), 112(F1).

Dalrymple, R.W. and Rhodes, R.N., 1995. Estuarine Dunes and Bars, in Perillo, G.M.E. (Ed). Geomorphology and Sedimentology of Estuaries, Elsevier Science B.V. Amsterdam.

Davison-Arnott, R., 2009. Introduction to Coastal Processes and Geomorphology. Cambridge University Press, Cambridge, UK.

Dearing J, Richmond N, Plater A, Wolf J, Prandle D, Coulthard T., 2006. Modelling approaches for coastal simulation based on cellular automata: the need and potential. Philos Trans R Soc A: Math, Phys Eng Sci 364(1841):1051–1071

Decombre, J., 2013. TELEMAC Modelling System, 3D hydrodynamics, TELMAC 3D Software, Release 6.2, Operating Manual. EDF R&D, 2013.

Deltares Systems, 2012. http://www.deltaressystems.com (Last accessed February 2013)

DHI, 2011. http://www.dhisoftware.com (Last accessed February 2013).

Di Silvio, G., Dall'Angelo, C., Bonaldo, D and Fasolato, G., 2010. Long-term model of planimetric and bathymetric evolution of a tidal lagoon. Continental Shelf Research, 30, 894-903.

Doody, J.P., 2004. 'Coastal squeeze' - an historical perspective. Journal of Coastal Conservation, 10/1-2, 129-138

Dyer, K.R., 1997. Estuaries A Physical Introduction. John Wiley and Sons Ltd., Chichester, England. ISBN 0-471-97470-6

Dyke, P., 2007. Modeling Coastal and Offshore Processes. Imperial College Press, London.

Emery, W.J. and Thompson. R.E., 2001. Data analysis methods in physical oceanography. Elsevier. Oxford, UK.

Escoffier, E.F., 1940. The Stability of Tidal Inlets, Shore and Beach, 8, 114-115.

EMPHASYS Consortium, 2000. A Guide to Prediction of Morphological Change within Estuarine Systems. Estuaries Research Programme, Phase 1, HR Wallingford, Report TR 114.

Fagherazzi, S., Kirwan, M.L., Mudd, S.M., Guntenspergen, G.R., Temmerman, S., D'Alpaos, A., van de Koppel, J., Rybczyk, J.M., Reyes, E., Craft, C. and Clough, J., 2012. Numerical models of salt marsh evolution: ecological, geomorphic, and climatic factors. Reviews of Geophysics, 50, RG1002.

Featherstone, R. and Nalluri, C., 1998. Civil Engineering Hydraulics. 2nd Edition. BSP Professional Books, Blackwell Scientific Publications Ltd. Oxford. ISBN 0-632-02201-9

Fischer H.B., List, E.J., Koh., R.C.Y., Imberger, J. and Brooks, N.H., 1979. Mixing in Inland and Coastal Waters. Academic Press. London.

Fredsoe, J. and Deigaard, R. 1992. Mechanics of Coastal Sediment Transport. Wold Scientific Publishing Co. Pte. Ltd. Singapore. ISBN 981-02-0840-5

Frostick, L.E. and McCave, I.N., 1979. Seasonal Shifts of Sediment Within an Estuary Mediated by Algal Growth. Estuarine and Coastal Marine Science (1979) 9, 569-576

Goda, Y., 2000. Random Seas and the Design of Martime Structures. 2nd Edition. World Scientific. London.

Hervouet, J.-M., 2000. TELEMAC modelling system: an overview. Hydrol. Process., 14: 2209–2210.

Hinkel, J. and Klein, R.J., 2009. Integrating knowledge to assess coastal vulnerability to sea-level rise: The development of the DIVA tool. Global Environmental Change, 19(3), 384-395.

Hinkel, J., Nicholls, R.J., Tol, R.S., Wang, Z.B., Hamilton, J.M., Boot, G. and Klein, R. J., 2013. A global analysis of erosion of sandy beaches and sea-level rise: An application of DIVA. Global and Planetary Change, 111, 150-158.

HR Wallingford. 1997. Estuaries: The case for research into morphology and processes. HR Wallingford Report SR 478.

HR Wallingford, ABPmer, and Pethick. J., 2006. Review and Formalisation of Geomorphological Concepts and Approaches for Estuaries. R&D Technical Report

FD2116/TR2. London, U.K Department of Environment, Food and Rural Affairs (DEFRA) and the Environment Agency (EA).

Hughes, S.A., 2002. Equilibrium cross sectional area at tidal inlets. Journal of Coastal Research, 18(1), 160-174.

Hui, H., 2007. Non-hydrostatic Numerical Modeling of Hydraulic Jump over a Weir. M.Sc. Thesis, Delft University of Technology, Delft.

Huthnance, J.M., Karunarathna, G., Lane, A., Manning, A.J., Norton, P., Reeve, D., Spearman, J., Soulsby, R.L., Townend, I.H., Wolf, J. and Wright, A., 2007. Development of estuary morphological models, R&D Technical Report FD2107/TR, Joint Defra/EA Flood and Coastal Erosion Risk Management R&D Programme.

Huygens' principle, 2013. Encyclopædia Britannica Online. Retrieved 05 April, 2013, from http://www.britannica.com/EBchecked/topic/277804/Huygens-principle

IPCC, 2014. Climate Change 2014: Impacts, Adaptation, and Vulnerability. Part A: Global and Sectoral Aspects. Contribution of Working Group II to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change [Field, C.B., V.R. Barros, D.J. Dokken, K.J. Mach, M.D. Mastrandrea, T.E. Bilir, M. Chatterjee, K.L. Ebi, Y.O. Estrada, R.C. Genova, B. Girma, E.S. Kissel, A.N. Levy, S. MacCracken, P.R. Mastrandrea, and L.L. White (eds.)]. Cambridge University Press, Cambridge, United Kingdom and New York, NY, USA, 1132 pp.

Isla, F.I., 1995. Coastal lagoons, in Perillo, G.M.E. (Ed). Geomorphology and Sedimentology of Estuaries, Elsevier Science B.V. Amsterdam.

Karunarathna, H. and Reeve, D.E., 2008. A Boolean Approach to Prediction of Long-term Evolution of Estuary Morphology. Journal of Coastal Research, Vol. 24(2B), pp. 51-61.

Karunarathna, G.H., Reeve, D. and Spivack, M. 2008. Long term morphodynamic evolution of estuaries: An inverse problem. Estuarine, Coastal and Shelf Science 77 (2008), 385-395.

Kirby, J. T., Wei, G., Chen, Q., Kennedy, A. B., & Dalrymple, R. A., 1998. Funwave 1.0, fully nonlinear boussinesq wave model documentation and user's manual. Center for Applied Coastal Research, University of Delaware.

Kleinhans, M. G., and van Rijn, L.C., 2002. Stochastic prediction of sediment transport in sand-gravel bed rivers. J. Hydraul. Eng. , 128 :4, 412–425.

Kutija, V., 1995. A generalized method for the solution of flows in networks. Journal of hydraulic research, 33(4), 535-554.

Langbein, W.R., 1963. The Hydraulic Geometry of a small estuary. Bulletin of International Association for Scientific Hydrology, 8, 84-94.

Lesser, G., Roelvink, J.A., Van Kester, J.A.T.M. and Stelling, G.S. 2004. Development and validation of a three-dimensional morphological model. Coastal Engineering 51, 883–915

Madsen, P. and Warren, I., 1984. Performance of a numerical short-wave model. Coastal Eng. 8, 73–93.

Mariotti G, Fagherazzi S (2010) A numerical model for the coupled long-term evolution of salt marshes and tidal flats. J Geophys Res 115:F01004

McGlathery, K. J., Sundback, K. and Fong, P, 2013. Estuarine Benthic Algae, in Day, W. D., Jr, Crump, B. C., Kemp, W. M. and Yanez-Arancibia, A (Editors), 2013. Estuarine Ecology, 2nd Edition. Wiley-Blackwell, Hoboken, New Jersey.

McLusky, D. S., 1981. The Estuarine Ecosystem. 2nd Edition. Blackie Academic and Professional, Glasgow.

McManus, J., Duck, R.W., 1996. Regional variations of fluvial sediment yield in eastern Scotland. In: Walling, D., Webb, B. (Eds.), Erosion and Sediment Yield: Global and Regional Perspectives (Proceedings of the Exeter Symposium, July 1996). IAHS Publ., 236. IAHS, Wallingford, United Kingdom, pp. 157–161.

McWilliams, B and Sprevak, D., 1982. The simulation of hourly wind speed and direction, Mathematics and Computers in Simulation, Volume 24, Issue 1, Pages 54-59

Morris, J.T., 2006. Competition among marsh macrophytes by means of geomorphological displacement in the intertidal zone. Estuarine, Coastal and Shelf Science, 69(3), 395-402.

Mudd, S. M., S. Fagherazzi, J. T. Morris, and D. J. Furbish (2004), Flow, sedimentation, and biomass production on a vegetated salt marsh in South Carolina: Toward a predictive model of marsh morphologic and ecologic evolution, in The Ecogeomorphology of Tidal Marshes, Coastal Estuarine Stud., vol. 59, edited by S. Fagherazzi, M. Marani, and L. K. Blum, pp. 165–187, AGU, Washington, D. C.

Murray, A. B. and Paola, C. 1994. 'A cellular model of braided rivers', Nature, 371, 54–57.

Neild, J.M. and Walker, D.J., 2005. Two-dimensional equilibrium morphological modelling of a tidal inlet: an entropy based approach. Ocean Dynamics (2005) 55, 549-558.

Nielson, P., 2009. Coastal and Estuarine Processes. Advanced Series on Ocean Engineering – Volume 29. World Scientific. London.

O'Brien, M.P., 1931. Estuary tidal prism related to entrance areas. Civil Engineering, 1(8), 738-739.

Packham, J.R. and Willis, A.J., 1997. Ecology of Dunes, Salt Marsh and Shingle. Chapman and Hall. London.

Papanicolaou, A. N., Elhakeem, M., Krallis, G., Prakash, S., & Edinger, J. (2008). Sediment transport modeling review—current and future developments. Journal of Hydraulic Engineering, 134(1), 1-14.

Parry, M.L., Canziani, J.P., Palutikof, J.P., van der Linden, P.J. and Hanson, C.E., 2007. Contribution of Working Group II to the Fourth Assessment Report of the Intergovernmental Panel on Climate Change. Cambridge University Press, Cambridge, UK, 976 pp

Perillo, G. M. E., 1995a. Geomorphology and sedimentology of estuaries: an introduction, in Perillo, G.M.E. (Ed). Geomorphology and Sedimentology of Estuaries, Elsevier Science B.V. Amsterdam.

Perillo, G. M. E., 1995b. Definitions and geomorphologic classifications of estuaries, in Perillo, G.M.E. (Ed). Geomorphology and Sedimentology of Estuaries, Elsevier Science B.V. Amsterdam.

Pethick, J., 1984. An Introduction to Coastal Geomorphology. Edward Arnold Ltd., London.

Prandle, D., 2003: Relationships between Tidal Dynamics and Bathymetry in Strongly Convergent Estuaries. J. Phys. Oceanogr., 33, 2738–2750.

Ranasinghe, R., Swinkels, C., Luijendijk, A., Roelvink, D., Bosboom, J., Stive, M. and Walstra, D., 2011. Morphodynamic upscaling with the MORFAC approach: Dependencies and sensitivities. Coastal engineering, 58(8), 806-811.

Reeve, D.E. and Karunarathna, H., 2009. On the Prediction of Long-term Evolution of Morphodynamic Response of Estuarine Systems to Sea Level Rise and Human Interference. Continental Shelf Research, Vol. 29, pp. 938-950

Reeve, D.E., and Karunarathna, H., 2011. A statistical–dynamical method for predicting estuary Morphology. Ocean Dynamics (2011) 61:1033–1044

Roelvink, J.A., 2006. Coastal morphodynamic evolution techniques. Coastal Engineering, 53(2), 277-287.

Rossington, S.K. and Nicholls, R.J., 2008. Morphological behaviour of UK estuaries under conditions of accelerating sealevel rise, in Dohmen-Janssen C.M. and Hulscher, S.J.M.H. (Eds). River, Coastal and Estuarine Morphodynamics: RCEM 2007, Taylor & Francis, Netherlands.

Rusu, E. and Soares, C.G., 2013. Modeling Waves in Open Coastal Areas and Harbors with Phase-Resolving and Phase-Averaged Models. Journal of Coastal Research: Volume 29, Issue 6: pp. 1309 – 1325.

Schrodoer, W.W., Dinnnel, S.P. and Wiseman Jr, W.J., 1990. Salinity Stratification in a River Dominated Estuary. Estuaries, Vol 13, No. 2, 145-154.

Schwimmer, R.A. and Pizzuto, J.E., 2000. A model for the evolution of marsh shorelines. Journal of Sedimentary Research, Vol. 70, Nr. 5, 1026-1035

Soulsby, R.L., 1997. Dynamics of Marine Sands. A manual for practical applications, Thomas Telford, London.

Soulsby, R.L., Mead, C.T. and Wild, B.R., 2007. A model for simulating the dispersal tracks of sand grains in coastal areas: 'SandTrack'. Geographical Society, London, Special Publications, 274, 65-72.

Spearman, J., 2010. The development of a tool for estimating the morphological evolution of managed realignment sites. Continental Shelf Research 31, S199-S210.

Stive, M.J.F.; Wang, Z.B.; Capobianco, M.; Ruol, P.; Buijsman, M.C. (1998). Morphodynamics of a tidal lagoon and the adjacent coast, in: Dronkers, J. et al. (Ed.) (1998). Physics of estuaries and coastal seas: proceedings of the 8th International Biennial Conference on physics of estuaries and coastal seas, The Hague, Netherlands 9-12 September 1996. pp. 397-407

SWAN Team, 2006. SWAN Technical Documentation, SWAN Cycle III, Version 40.51. Delft University of Technology, Delft.

Syvitski, J.P.M. and Shaw, J., 1995. Sedimentology and geomorphology of Fjords, in Perillo, G.M.E. (Ed). Geomorphology and Sedimentology of Estuaries, Elsevier Science B.V. Amsterdam.

Tayfur, G. and Singh, V. P., 2007. Kinematic wave model for transient bed profiles in alluvial channels under nonequilibrium conditions. Water resources research, 43, W12412.

Telemac Consortium, n.d. http://www.opentelemac.org (Last accessed February, 2013).

Thomas, C.G., Spearman, J.R. and Turnbull, M.J., 2002. Historical morphological change in the Mersey estuary. Continental Shelf Research, 22, 1775-1794

Townend, I., 2005. An examination of empirical stability relationships for UK estuaries. Journal of Coastal Research (2005): 1042-1053.

Townend, I. H., C. A. Fletcher, M. A. F. Knaapen, and S. K. Rossington (2010), A review of salt marsh dynamics, Water Environ. J., 24, 1–12.

US. Army Corps of Engineers, 2002. Coastal Engineering Manual. Engineer Manual 1110-2-1110, US. Army Corps of Engineers, Washington, D.C.

Van Goor, M.A., Zitman, T.J., Wang, Z.B., Stive, M.J.F., 2003. Impact of sea-level rise on the morphological equilibrium state of tidal inlets. Marine Geology 202, 211–227.

Van Rijn, L.C., 1984a. Sediment transport, Part I: Bed load transport. J. Hydraul. Eng., 110:10, 1431–1456

Van Rijn, L.C., 1984b. Sediment transport, Part II: Suspended load transport. J. Hydraul. Eng., 110:11, 1613–1641.

Van Rijn, L.C., 1989. Handbook of sediment transport by currents and waves. Report H 461, Delft Hydraulics.

Van Rijn, L.C., 1993. Principles of Sediment Transport in Rivers Estuaries and Coastal Seas. Aqua Publications, Blokzijl, Netherlands

Van Rijn, L.C., 2005. Principles of Sedimentation and Erosion Engineering in Rivers Estuaries and Coastal Seas. Aqua Publications, Blokzijl, Netherlands.

Van Rijn, L.C., 2007a. Unified view of sediment transport by currents and waves: I. Initiation of motion, bed roughness and bed load transport, J. Hydraul. Eng., 133:6, 649–667.

Van Rijn, L.C., 2007b. Unified view of sediment transport by currents and waves. II: Suspended transport. J. Hydraul. Eng., 133:6, 668–689.

Van Rijn, L.C., 2007c. Unified view of sediment transport by currents and waves, III: Graded beds. J. Hydraul. Eng., 133:7, 761–775.

Van Rijn, L.C., 2007d. Unified view of sediment transport by currents and waves, IV: Application of Morphodynamic Model. J. Hydraul. Eng., 133:7, 776–793.

Wang, Z. B., Fokkink, R. J. and Langerak, A. 1998. A dynamic-empirical model for estuarine morphology, In: Dronkers, J., Sheffers, M. B. A. M. (Eds), Physics of Estuaries and Coastal Seas, Balkema, Rotterdam, pp. 279-286.

Wang, Z. B., Jeuken, C. and De Vriend., H. J., 1999. Tidal asymmetry and residual sediment transport in estuaries. A literature study and applications to the Western Scheldt. WL| Delft Hydraulics report Z 2749.

Wang, Z. B., de Vriend, H. J., Stive, M. J. F. and Townend, I. H., 2007. On the parameter setting of semi-empirical long-term morphological models for estuaries and tidal lagoons. In: Proceedings of 5th IAHR Symposium on River, Coastal and Estuarine Morphodynamics. IAHR. (2007)

Warren Pinnacle Consulting, 2012. SLAMM 6.2 Technical Documentation, Sea Level Affecting Marshes Model, Version 6.2 beta. Warren Pinnacle Consulting Inc., Waitsfield VT, USA.

Wass, P.D. and Leeks, G.J.L., 1999. Suspended sediment fluxes in the Humber catchment, UK. Hydrol. Process. 13, 935-953

Whitehouse, R., Soulsby, R., Roberts, W. and Mitchener, H., 2000. Dynamics of estuarine muds. A manual for practical applications. Thomas Telford. London.

Widdows, J. and Brinsley, M., 2002. Impact of biotic and abiotic processes on sediment dynamics and the consequences to the structure and functioning of the intertidal zone. Journal of Sea Research. 48. 143-156.

Wofram, S., 2002. A New Kind of Science, Wolfram Media, Champaign, IL, USA.

Write, A.P., Townend I.H. and Rossington, K., 2007. ASMITA Manual Version 1.3, Joint Defra/EA Flood and Coastal Erosion Risk Management R&D Programme, R&D Technical Report FD2107/TR. Defra, London.

Xu, J., 2002. Implications of relationships among suspended sediment size, water discharge, and suspended sediment concentration, the Yellow River basin, China. Catena, 49, 289-307

Yang, C.T., 1996. Sediment transport: theory and practice. Reprint edition 2003 with corrections, Krieger Publishing Company, Forida, USA.

Younes, A. and Ackerer, P., 2005. Solving the advection-diffusion equation with the Eulerian-Lagrangian localized adjoint method on unstructured meshes and non uniform time stepping. Journal of Computational Physics, 208, 384-402.

A Novel Cellular Automata Based Estuarine Morphodynamic Model

Ian Bentley

MEng. MSc.

Submitted in fulfilment of the requirements for the degree of

Doctor of Philosophy

School of Engineering

University of Glasgow

Volume 2 of 2

Appendix A

FORTRAN SOURCE CODE LISTING

FORTRAN Source Code

Modules:

MAIN PROGRAM

Program	Purpose	Page Nr.
ESTUARY_MODEL	Main program	248
Subroutines		
SED_OUT	Write grid data to file (sediment)	281
GRID_OUT	Write grid data to file (salt marsh / velocity data)	282
GRID_OUTL	Write grid data to file (bathymetry)	283
Functions		
GETSIZE1	Calculate d_{10}, d_{50}, d_{90} etc. for output to file	284

```
!CA Estuary Model Main Program
!Last update 07/02/2015

PROGRAM ESTUARY_MODEL

USE ISIS
USE SANDa
USE SAND_MUD
USE FLOW1
USE MARSH
USE DATA_INOUT
USE WAVES
USE CLG1

IMPLICIT NONE

!Array to store the bed level grid data
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:), ALLOCATABLE :: GRID
!Bed levels in interpolated grid
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:), ALLOCATABLE :: CLG_BED

REAL,  DIMENSION(:,:),            ALLOCATABLE   :: FGRID !Non-erodable bed levels
REAL,  DIMENSION(:,:,:),   ALLOCATABLE :: QX    !RH cell boundary flow
REAL,  DIMENSION(:,:,:),   ALLOCATABLE :: QY    !'Top' cell boundary flow

REAL,    DIMENSION(:,:),     ALLOCATABLE :: VX    !RH cell bdy velocity
REAL,  DIMENSION(:,:),     ALLOCATABLE :: VY    !Top cell bdy velocity
!Water level at start of each time-step, for each cell
REAL,  DIMENSION(:,:,:),   ALLOCATABLE   :: WL
!Array to store water level data from steady state model
!run with normal depth downstream boundary
!(if ISIS used to obtain water level)
REAL,  DIMENSION(:),       ALLOCATABLE   :: WL_SS


REAL,    DIMENSION(:,:),     ALLOCATABLE :: H   !Water depths in each cell
REAL,    DIMENSION(:,:),     ALLOCATABLE :: H1  !Depth from slope routing

!Time averaged water depth at each cell (over one time-step)
REAL (SELECTED_REAL_KIND(15,307)),   DIMENSION(:,:), ALLOCATABLE :: MEAN_H
!Array to store the sediment fraction in the sub layers
REAL (SELECTED_REAL_KIND(15,307)),   DIMENSION(:,:,:,:),ALLOCATABLE :: SED
!Array to store the active layer for each cell
REAL (SELECTED_REAL_KIND(15,307)),   DIMENSION(:,:,:), ALLOCATABLE :: ALYR
```

```
!Total sediment in model domain for each fraction
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:), ALLOCATABLE :: MF_SED_TOT
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:),ALLOCATABLE :: MF_SED_TOT1
!Suspended sediment concentration
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:,:), ALLOCATABLE :: SC
!Cumulative mud deposition for each cell
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:),   ALLOCATABLE :: DTOT
!Culumative mud erosion for each cell
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:),   ALLOCATABLE :: ETOT
!Multi-fraction cumulative deposition
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:,:), ALLOCATABLE :: MFDTOT
!Multi-fraction cumulative erosion
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:,:), ALLOCATABLE :: MFETOT

REAL,   DIMENSION(:),       ALLOCATABLE :: FI    !mean fraction sizes
REAL,   DIMENSION(:),       ALLOCATABLE :: WS    !particle fall velocities
REAL,   DIMENSION(:),       ALLOCATABLE :: FMIN   !min fraction sizes
REAL,   DIMENSION(:),       ALLOCATABLE :: FMAX   !maxfraction sizes
REAL,   DIMENSION(:),       ALLOCATABLE :: PRI    !fraction proportions
REAL,   DIMENSION(:),       ALLOCATABLE :: XSEC_CH !ISIS XSEC chainages
REAL,    DIMENSION(:,:),     ALLOCATABLE :: TIDE    !tidal boundary data
REAL,   DIMENSION(:,:),     ALLOCATABLE :: MIN_WL  !min water levels
REAL,   DIMENSION(:,:),     ALLOCATABLE :: MAX_WL  !max WL for each cell
REAL,   DIMENSION(:),       ALLOCATABLE :: ROW_CH  !row ch. in int. grd.
REAL,   DIMENSION(:,:),     ALLOCATABLE :: C       !Chezy coefficent
REAL,   DIMENSION(:,:),     ALLOCATABLE :: BM     !fraction of max biomass
REAL,   DIMENSION(:,:,:),   ALLOCATABLE :: WAVE_H    !Wave height
REAL,   DIMENSION(:,:,:),   ALLOCATABLE :: WAVE_T    !Wave period
REAL,   DIMENSION(:,:,:),   ALLOCATABLE :: WAVE_DIR  !Wave direction
REAL,   DIMENSION(:,:),     ALLOCATABLE :: STYPE !Predef. sed. composition
REAL,   DIMENSION(:,:),     ALLOCATABLE :: SED_IN !Boundary sediment input
REAL,   DIMENSION(:),       ALLOCATABLE :: MISCV  !Misc. input variables
REAL,   DIMENSION(:),       ALLOCATABLE :: KX     !Wave number x chainage
REAL,   DIMENSION(:),       ALLOCATABLE :: QI     !Fluvial inflow(s)

!Predefined points defining cenreline for interpolated grid (x&y coords)
REAL,  DIMENSION(:,:),     ALLOCATABLE :: SETUP_PTS
!Directions of centreline sections
REAL,   DIMENSION(:),       ALLOCATABLE :: DTN
!x & y Coordinates of start of straight line sections of interpolated grid
REAL,   DIMENSION(:,:),     ALLOCATABLE :: LINE_START
!x & y coordinates of origin points for arc segments of interpolated grid
REAL,   DIMENSION(:,:),     ALLOCATABLE :: ARC_ORIGIN
!Interpolated grid cell lengths for each straight line section
REAL,   DIMENSION(:),       ALLOCATABLE :: LINE_INC
!Interpolated grid cell lengths (in radians) for each arc section
REAL,   DIMENSION(:),       ALLOCATABLE :: ARC_INC
!x & y coordinates of each interpolated grid cell
REAL,   DIMENSION(:,:,:),   ALLOCATABLE :: NG_LOC
!Area of each interpolated grid cell
REAL,  DIMENSION(:,:),     ALLOCATABLE :: NG_AREA
!Length of each interpolated grid cell
REAL,   DIMENSION(:,:),     ALLOCATABLE :: CELL_LEN
!Weightings for calculating CLG bed levels from source grid
REAL,   DIMENSION(:,:,:),   ALLOCATABLE :: BLWT
!Orientation of interpolated grid cells
REAL,   DIMENSION(:),       ALLOCATABLE :: ROW_DTN
!Weighting for interpolating WL in each cell, from int. grd water WLs
REAL,   DIMENSION(:,:,:),   ALLOCATABLE :: CLG_WT
!Interpolated grid water levels
REAL,   DIMENSION(:,:),     ALLOCATABLE :: CLG_WL
!Width of each strainght line & arc section of interpolated grid
REAL,   DIMENSION(:),       ALLOCATABLE :: CLG_WIDTH
!Initial Percentage mud in each cell, loaded from inup file
REAL,   DIMENSION(:,:),     ALLOCATABLE :: PMUD_FILE
```

```
!Probablility for each wave height/period/direction
REAL,    DIMENSION(:),       ALLOCATABLE :: P
REAL,    DIMENSION(:),       ALLOCATABLE :: WSP  !Wind speed
REAL,    DIMENSION(:),       ALLOCATABLE :: WD   !Wind direction

!Total sediment input for each fraction
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:), ALLOCATABLE :: MF_SED_IN
!Total sediment output for each fraction
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:), ALLOCATABLE :: MF_SED_OUT
!Total suspended sediment volume for each fraction
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:), ALLOCATABLE :: MF_SSVOL

REAL,    DIMENSION(6) :: PMUD !%mud for single-fraction+mud sediment types

!Initial sediment types in each cell (types defined by STYPE)
INTEGER, DIMENSION(:,:), ALLOCATABLE :: SED_TYPE
INTEGER, DIMENSION(:),   ALLOCATABLE :: QI_R !Row nr. for fluvial inflows
INTEGER, DIMENSION(:),   ALLOCATABLE :: QI_C !Col nr. for fluvial inflows

!Interpolated grid: length of line sections (interpolated grid cells)
INTEGER, DIMENSION(:),   ALLOCATABLE :: LINE_LEN
!Interpolated grid: length of arc sections (interpolated grid cells)
INTEGER, DIMENSION(:),   ALLOCATABLE :: ARC_LEN
!Interpolated grid: number of columns in each line / arc pair
INTEGER, DIMENSION(:),   ALLOCATABLE :: NG_COLS
!Interpolated grid: number of columns in each row
INTEGER, DIMENSION(:),   ALLOCATABLE :: ROW_WTH
!X Coordinate (row/col in source grid) for bed level weightings
INTEGER, DIMENSION(:,:,:), ALLOCATABLE :: WT_XCOORDS
!Y Coordinate (row/col in source grid) for bed level weightings
INTEGER, DIMENSION(:,:,:), ALLOCATABLE :: WT_YCOORDS
!Interpolated grid X & Y coordinates for nearest 3 IG cells
!to a given regular grid cell
INTEGER, DIMENSION(:,:,:), ALLOCATABLE :: XWT
INTEGER, DIMENSION(:,:,:), ALLOCATABLE :: YWT

REAL (SELECTED_REAL_KIND(15,307)) VOL      !Total sediment volume
REAL (SELECTED_REAL_KIND(15,307)) SAND_IN  !Total sand input
REAL (SELECTED_REAL_KIND(15,307)) SAND_OUT !Total sand output
REAL (SELECTED_REAL_KIND(15,307)) MUD_IN   !Total mud input
REAL (SELECTED_REAL_KIND(15,307)) MUD_OUT  !Total mud output
REAL (SELECTED_REAL_KIND(15,307)) DATUM    !Datum for total sed. vol. calc
REAL (SELECTED_REAL_KIND(15,307)) SAND_TOT !Total sand volume
REAL (SELECTED_REAL_KIND(15,307)) MUD_TOT  !Total mud volume
REAL (SELECTED_REAL_KIND(15,307)) R_LYRS   !Number of sublayers
REAL (SELECTED_REAL_KIND(15,307)) ALYR_BASE !Active layer base Level
REAL (SELECTED_REAL_KIND(15,307)) MAXCOR    !Max concentration correction
REAL (SELECTED_REAL_KIND(15,307)) CTOL      !Cencentration tolerance
REAL (SELECTED_REAL_KIND(15,307)) MAXER     !Max erosion limited by min BL
REAL (SELECTED_REAL_KIND(15,307)) ETOT1     !Total erosion
!Neighbouring cell sediment concentrations
REAL (SELECTED_REAL_KIND(15,307)) C1, C2, C3, C4, SC1
!Neighbouring cell diffusion coefficients
REAL (SELECTED_REAL_KIND(15,307)) K1, K2, K3, K4

!Number of hours for ISIS warmup
!(to avoid initial conditions affect on results)
REAL ISIS_WARMUP

REAL GRID_SIZE                  !Grid cell size in metres
REAL GRID_WIDTH               !Grid width
REAL GRID_LENGTH        !Grid length

REAL TIMESTEP           !time-step in hours
REAL TSS                !timestep in seconds
```

```
REAL RUN_TIME           !run time in hours
REAL RT_YRS             !run time in years
REAL TIME               !Current time from start of run (hrs)
REAL ISIS_RT            !ISIS model run time
REAL LM_SOL             !Lunar month relative to sun

REAL TIDAL_PERIOD       !Tidal period in hours
REAL TDR_SP             !Spring tidal range
REAL TDR_NP             !Neap tidal range
REAL HTIDE              !Spring high tide level
REAL TD_RANGE           !Current tidal range
REAL MID_TIDE           !Mean sea level
!Tidal period & frequency + frequency of spring-neap cycle
REAL PT, FT, FSN
!Total upstream fluvial inflow (sum of QI() point inflows)
REAL INFLOW

REAL PS                 !Sediment porosity
!Sediment properties (sizes in micrometres / density in kg/m3)
REAL D10, D50, D90, DS
REAL SMIN, SMAX         !Minumum/maximum levels for bed composition array
REAL SM1, SM2           !Salt marsh max biomass points 1 & 2

!Total proportion of all sediment fractions, excluding the clay fraction
REAL (SELECTED_REAL_KIND(15,307)) :: SED_TOT
!Sub-layer and active layer thickness
REAL (SELECTED_REAL_KIND(15,307)) LYR_TH, ALYR_TH
!Min & max slopes used in slope based flow routing + routing exponent
REAL S_MIN, S_MAX, XP

!Max salt marsh biomass + min & max submergence time (proportion of total)
REAL BMAX, SUB_MIN, SUB_MAX
REAL KB  !Organogenic sediment production for salt marsh (m/yr)

!Grid cell size for 2nd & subsequent grid files - for consistancy check
REAL GS
!Height of additional points added to ends of xsecs used in ISIS model
REAL WALL_HT
!Wind direction, wind speed and fetch as model boundary (input value)
REAL WIND_DIR, WIND_SPD, BDY_FETCH
REAL ESD            !Standard deviation of random changes to wind speed
REAL SPD, SPSD, PWS !prevailing windspeed direction, standard dev. & mean

REAL MIN_H, MAX_V   !Min water depth, max velocity

REAL RESULTS_INT            !Inverval between grid results output (hours)
REAL RESULTS_TIME           !Time of next grid results output (hours)
!Intervals for grid results output
REAL SED_RESULTS_INT, V_RESULTS_INT1, SED_RESULTS_TIME, V_RESULTS_TIME
REAL DS_SED_INPUT           !Sand input at downstream boundary

REAL LTFR                   !Lateral sediment tranport coefficient
REAL DUM1                   !Dummy input / output

REAL (SELECTED_REAL_KIND(15,307)) SMIN1, SMAX1 !Min & max sublayer levels

REAL DMUD    !Bulk density of mud
REAL TE, TD  !Critical bed shear stress for erosion / deposition, for muds
REAL TCRF  !Max bed shear stress scaling factor (for mud proportion = 20%)

REAL C_MIN, C_MAX, KD !Min / max Chezy coefficient / diffusion coefficient

REAL HTOT      !Sum of water depths over a single row / cross section
REAL GRID_MIN  !Minimum bed level over a single row / cross section
REAL HMEAN     !Mean water depth over a single row / cross section
```

```
REAL CT        !Tidal wave celerity

REAL DSSC                !downstream boundary suspended mud concentration

!Prefered cell width in interpolated grid
!(actual width must be integer factor of grid width)
REAL NG_SIZE
REAL C_TIME !Time when water levels next recalculated
REAL C_INT  !interval between water level calculations (hours)
REAL XYMIN  !Minimum XWT or YWT value
REAL MASSTM !Time of next mas balance output,
REAL MASSINT !interval between mass balance outputs

REAL PTOT       !Sum of wind/wave proabilities
REAL WS1, WS2   !Prevailing wind speed / transverse wind speed
REAL P1, P2     !Probabilities for prevailing & transverse wind speeds
REAL WREF       !Lateral wave reflection parameter
REAL SLR        !Sea level rise parameter

REAL :: DSAND = 0.000062    !Minimum grain size classes as sand

INTEGER NR_RESULTS  !Number of locations at which results will be output
INTEGER TOTAL_TIMESTEPS  !Total number of timesteps the model will run for

INTEGER N, M, B         !Counters
INTEGER ROW, COL        !Current row/column in loop
INTEGER ROWS, COLS      !Total number of rows/columns in source grid
INTEGER STLEN           !Integer to hold number of chacacters in string
INTEGER TSPT            !Timesteps per tidal cycle
INTEGER TSPRP !Timesteps / tidal period (tidal cycle or spring/neap cycle)
INTEGER XSECS           !Number of cross sections in ISIS model
INTEGER XSEC_SP         !ISIS cross section spacing in grid rows
INTEGER DBP             !Number of points in ISIS downstream boundary
INTEGER T, T1, T2       !Timestep counters
INTEGER TS_NR           !Total timesteps
INTEGER ST_LEN          !character string length
INTEGER NWL          !Number of water levels to be calculated for each cell
INTEGER NR_FRACTIONS    !Number of sediment fractions
INTEGER NR_LAYERS       !Total number of sub-layers
INTEGER N_LYRS          !Number of sub-layers below bed level
INTEGER RESULTS_DIR_LEN !Number of characters in results directory path
INTEGER RS, CS          !Number of rows / cols in grid input file
INTEGER NR_TYPES        !Number of pre-defined sediment types
INTEGER ZZ              !Switch ISIS error check switch (on if zz = 1)
INTEGER V_RESULTS_INT2, V_RESULTS_NR, VRC1, VRC2 !Velocity output params
INTEGER DS_SED_TYPE, DSSC_TYPE  !Boundary sediment types (multi-fraction)
INTEGER NV     !Number of misc. model variables to read from control file
INTEGER IO              !File status
INTEGER RUN_MODE        !Run mode (single or multi-fraction)
INTEGER QI_N            !Number of inflow points
INTEGER NG_ROWS, NR_PTS, NG_COLS_MAX, ROW_OS !Interpolated grid parameters
INTEGER MASS_FILE       !Mass balance file number
INTEGER TS_FILE         !Time-series output file number

INTEGER LOG_FILE !Log file number
INTEGER MAXITR !Maximum number of iterations of calculate flow field
INTEGER NMUD   !Number of fractions classed as mud (multi-fraction model)
INTEGER NWS1   !Number of wind speeds in prevailing/transverse directions
INTEGER NWS2        !Number of wind speed combinations
INTEGER N1, N2, N3  !Counters
INTEGER ITT         !Number of iterations
INTEGER TS_ROW, TS_COL !Row & column numbers for timeseries output

CHARACTER (LEN = 200) CONTRL_FN     !Filename for model control file
CHARACTER (LEN = 200) GRID_FN       !Filename for source grid data
```

```
CHARACTER (LEN = 200) FGRID_FN      !Filename for min bed lvl grid input
CHARACTER (LEN = 200) STYPE_FN      !File for sediment type grid input
CHARACTER (LEN = 200) SED_FN        !Fn for sed data (multifraction mode)
CHARACTER (LEN = 200) RESULTS_DIR   !Directory path for results output
CHARACTER (LEN = 200) RESULTS_FN    !Filename for results output
CHARACTER (LEN = 200) ISIS_FN       !Filename for ISIS DAT file
CHARACTER (LEN = 200) ISIS_FN_SS    !Filename for ISIS DAT (steady state)
CHARACTER (LEN = 200) ISIS_SS_ZZS   !ISIS steady state results filename
CHARACTER (LEN = 200) ISIS_OUT_FN   !Filename for ISIS csv output file
CHARACTER (LEN = 200) MARSH_FN      !Initial marsh grid input filename
CHARACTER (LEN = 200) MASS_BALANCE_FN   !Mass balance output filename
CHARACTER (LEN = 200) LOG_FILE_FN   !Log file filename
CHARACTER (LEN = 200) TS_FILE_FN    !Time-series file filename
CHARACTER (LEN = 23) CHR            !Temp character string
CHARACTER (LEN = 23) CHR1           !Temp character string
CHARACTER (LEN = 23) CHR2           !Temp character string
CHARACTER (LEN = 5) RP              !'True' for spring/neap tides
CHARACTER (LEN = 5) IS_SF           !String input 'true' or 'false'
CHARACTER (LEN = 5) IS_MARSH        !String input 'true' or 'false'
CHARACTER (LEN = 5) IS_ISIS         !String input 'true' or 'false'
CHARACTER (LEN = 5) IS_WAVES        !String input 'true' or 'false'
CHARACTER (LEN = 5) IS_BDR          !String input 'true' or 'false'
CHARACTER (LEN = 5) IS_SDR          !String input 'true' or 'false'
CHARACTER (LEN = 5) IS_VLR          !String input 'true' or 'false'
CHARACTER (LEN = 5) IS_INT          !String input 'true' or 'false'
CHARACTER (LEN = 5) IS_MUD          !String input 'true' or 'false'
CHARACTER (LEN = 5) IS_TS           !String input 'true' or 'false'
CHARACTER STR

LOGICAL INTERPOLATE         !True if ISIS iunterpolates to be added
LOGICAL MF, MRSH            !True for multi-fraction / marsh component used
LOGICAL ERROR               !True if ISIS error found
LOGICAL USE_ISIS, USE_CLG   !True if ISIS used / if interpolated grid used
LOGICAL USE_WAVES           !True if wave model used
LOGICAL TS_RESULTS          !True if time-series results selected
!True if bed level / sediment / velocity results to be output
LOGICAL BD_RESULTS, SED_RESULTS, V_RESULTS, WV_RESULTS
LOGICAL MUDFILE !True if percentage mud in each cell loaded from file

!True if tidal water level above cross section bed level
LOGICAL, DIMENSION(:), ALLOCATABLE :: TDWL

LM_SOL = LUNAR_MONTH * (365.0 / 364.0)
TIDAL_PERIOD = 12 / (1 - (1 / LM_SOL))

!Retrieve control filename, which must be supplied as argument
CALL GETARG (1, CONTRL_FN)
IF (CONTRL_FN == "") THEN
    !Control file for debugging
    CONTRL_FN = &
"C:\Users\iwben_000\Documents\Model\Test_Dec14\Flow_TS_Example\Flow_TS_Example.cr
l"
END IF


!----------------READ MODEL PARAMETER AND BED LEVEL FILES-----------------

!Read data from control file
OPEN (1, FILE = CONTRL_FN, STATUS = "OLD", IOSTAT = IO)
IF (IO /= 0) THEN
    PRINT*, "Control file not found"
    READ (*,10) str
    STOP
END IF
```

```
READ (1, 20) RT_YRS
READ (1, 150) TSPT

RUN_TIME = RT_YRS * 8766   !Assuming 365.25 days / year

TIMESTEP = TIDAL_PERIOD / TSPT
TSS = TIMESTEP * 3600

READ (1, 10) GRID_FN    !Filename for initial bed levels file
STLEN = 1
DO
    STLEN = STLEN + 1
    IF (STLEN == 113 .OR. GRID_FN(STLEN:STLEN) == '"') EXIT
END DO
GRID_FN = GRID_FN(2:STLEN - 1)    !Remove quotes from string

READ (1, 10) FGRID_FN   !Filename for minimum bed levels file
STLEN = 1
DO
    STLEN = STLEN + 1
    IF (STLEN == 133 .OR. FGRID_FN(STLEN:STLEN) == '"') EXIT
END DO
FGRID_FN = FGRID_FN(2:STLEN - 1)  !Remove quotes from string

!Read input setting to use/not use ISIS model for water level calculation
READ (1, 10) IS_ISIS
IF (IS_ISIS == "True") THEN
    USE_ISIS = .TRUE.
ELSE
    USE_ISIS = .FALSE.
END IF

!Read ISIS model settings
!Period between start of ISIS run and period used for results
READ (1, 20) ISIS_WARMUP
!(e.g. for XSEC_SP = 2 ISIS cross section for every second grid row)
READ (1, 220) XSEC_SP
!Setting to add ISIS interpolates between cross sections
READ (1, 10) IS_INT
IF (IS_INT == "True") THEN
    INTERPOLATE = .TRUE.
ELSE
    INTERPOLATE = .FALSE.
END IF
READ (1, 10) ISIS_FN   !Filename for ISIS dat file
STLEN = 1
DO
    STLEN = STLEN + 1
    IF (STLEN == 113 .OR. ISIS_FN(STLEN:STLEN) == '"') EXIT
END DO
ISIS_FN = ISIS_FN(2:STLEN - 1)    !Remove quotes from string
READ (1, 10) ISIS_FN_SS   !Filename for ISIS steady state dat file
STLEN = 1
DO
    STLEN = STLEN + 1
    IF (STLEN == 113 .OR. ISIS_FN_SS(STLEN:STLEN) == '"') EXIT
END DO
ISIS_FN_SS = ISIS_FN_SS(2:STLEN - 1)    !Remove quotes from string

!Hydraulics boundary conditions
READ (1, 10) RP
READ (1, 190, ADVANCE = 'NO') HTIDE
READ (1, 190, ADVANCE = 'NO') TDR_SP
READ (1, 190) TDR_NP
```

```
!Wave & wind model inputs
READ (1, 10) IS_WAVES
IF (IS_WAVES == "True") THEN
    USE_WAVES = .TRUE.
ELSE
    USE_WAVES = .FALSE.
END IF

READ (1, 310, ADVANCE = 'NO') SPD
READ (1, 190, ADVANCE = 'NO') PWS
READ (1, 190, ADVANCE = 'NO') SPSD
READ (1, 330) BDY_FETCH

!Single or multi-fraction
READ (1, 10) IS_SF
IF (IS_SF == "True") THEN
    MF = .FALSE.
ELSE
    MF = .TRUE.
END IF

!Sediment model options
IF (MF) THEN
    RUN_MODE = 2
ELSE
    RUN_MODE = 1
END IF

!Sediment parameters
READ (1, 190, ADVANCE = 'NO') D10
READ (1, 190, ADVANCE = 'NO') D50
READ (1, 190) D90
READ (1, 10) SED_FN
STLEN = 1
DO
    STLEN = STLEN + 1
    IF (STLEN == 113 .OR. SED_FN(STLEN:STLEN) == '"') EXIT
END DO
SED_FN = SED_FN(2:STLEN - 1)      !Remove quotes from string
READ (1, 10) STYPE_FN
STLEN = 1
DO
    STLEN = STLEN + 1
    IF (STLEN == 113 .OR. STYPE_FN(STLEN:STLEN) == '"') EXIT
END DO
STYPE_FN = STYPE_FN(2:STLEN - 1)  !Remove quotes from string

!Sediment boundary conditions
READ (1, 20, ADVANCE = 'NO') DS_SED_INPUT
READ (1, 160) DS_SED_TYPE
READ (1, 330, ADVANCE = 'NO') DSSC
READ (1, 160) DSSC_TYPE

!Salt marsh inputs
READ (1, 10) IS_MARSH
IF (IS_MARSH == "True") THEN
    MRSH = .TRUE.
ELSE
    MRSH = .FALSE.
END IF
READ (1, 10) MARSH_FN

STLEN = 1
DO
    STLEN = STLEN + 1
```

```
    IF (STLEN == 113 .OR. MARSH_FN(STLEN:STLEN) == '"') EXIT
END DO

MARSH_FN = MARSH_FN(2:STLEN - 1) !Remove quotes from string
READ (1, 170, ADVANCE = 'NO') BMAX
READ (1, 170, ADVANCE = 'NO') SUB_MIN
READ (1, 170, ADVANCE = 'NO') SM1
READ (1, 170, ADVANCE = 'NO') SM2
READ (1, 170) SUB_MAX

!Slope routing parameters
READ (1, 280, ADVANCE = 'NO') XP
READ (1, 290) S_MIN

!Results output options
READ (1, 10) IS_BDR
IF (IS_BDR == "True") THEN
    BD_RESULTS = .TRUE.
ELSE
    BD_RESULTS = .FALSE.
END IF

READ (1, 260) RESULTS_INT
READ (1, 10) IS_SDR
IF (IS_SDR == "True") THEN
    SED_RESULTS = .TRUE.
ELSE
    SED_RESULTS = .FALSE.
END IF
READ (1, 260) SED_RESULTS_INT
READ (1, 10) IS_VLR
IF (IS_VLR == "True") THEN
    V_RESULTS = .TRUE.
ELSE
    V_RESULTS = .FALSE.
END IF
READ (1, 260, ADVANCE = 'NO') V_RESULTS_INT1
READ (1, 270, ADVANCE = 'NO') V_RESULTS_INT2
READ (1, 270) V_RESULTS_NR

READ (1, 10) IS_TS
IF (IS_TS == "True") THEN
    TS_RESULTS = .TRUE.
ELSE
    TS_RESULTS = .FALSE.
END IF
READ (1, 270, ADVANCE = 'NO') TS_ROW
READ (1, 270) TS_COL

READ (1, 10) RESULTS_DIR
STLEN = 1
DO
    STLEN = STLEN + 1
    IF (STLEN == 113 .OR. RESULTS_DIR(STLEN:STLEN) == '"') EXIT
END DO
RESULTS_DIR = RESULTS_DIR(2:STLEN - 1)  !Remove quotes from string

!Read the miscellaneous variable inputs
READ (1, 160) NV
IF (NV > 0) THEN
    ALLOCATE (MISCV(NV))
    DO N = 1, NV
        READ (1, 300) MISCV(N)
    END DO
END IF
```

```
CLOSE (1)

IF (RP == "True") THEN
  !Set options for uniform tide
  TSPRP = TSPT
  ISIS_RT = ISIS_WARMUP + 13
  NWL = TSPT
ELSE
  !Set options for variable spring-neap tide
  TSPRP = TSPT * 29
  NWL = TSPRP
  ISIS_RT = ISIS_WARMUP + 380
END IF

DBP = INT (ISIS_RT * 5) + 1

!Read data from text files

!Initial bed levels, including number of rows &
!columns of cells in the model domain and cell size
CALL DATA_INPUT(GRID, 1, ROWS, COLS, GRID_SIZE, GRID_FN)

!Minimum bed levels
CALL DATA_INPUT(FGRID, 1, RS, CS, GS, FGRID_FN)
IF (RS /= ROWS .OR. CS /= COLS .OR. GS /= GRID_SIZE) THEN
    PRINT*, "Inconsistent data. Number of rows/columns or "
    PRINT*, "cell size in fixed bed file does not match bed data file"
    READ*, CHR
    STOP
END IF

!Initial marsh coverage
IF (MRSH) THEN
    CALL DATA_INPUT(BM, 1, RS, CS, GS, MARSH_FN)
    IF (RS /= ROWS .OR. CS /= COLS .OR. GS /= GRID_SIZE) THEN
        PRINT*, "Inconsistent data. Number of rows/columns or cell size "
        PRINT*, "in initial marsh file does not match bed data file"
        READ*, CHR
        STOP
    END IF
ELSE
    ALLOCATE (BM(ROWS, COLS))
    BM = 0
END IF

!Dimension arrays
ALLOCATE (C(ROWS, COLS))
ALLOCATE (QX(ROWS, COLS, TSPRP))
ALLOCATE (QY(ROWS, COLS, TSPRP))

RESULTS_DIR_LEN = LEN_TRIM(RESULTS_DIR)



!Assign miscellaneous variables

DS = MISCV(3)
PS = MISCV(4)
C = MISCV(5)
MIN_H = MISCV(6)
MAX_V = MISCV(7)

LTFR = MISCV(9)
ESD = MISCV(10)
```

```
KB = MISCV(11)
TE = MISCV(12)
TD = MISCV(13)
DMUD = MISCV(14)
SMIN1 = MISCV(15)
SMAX1 = MISCV(16)
PMUD(1) = MISCV(17) / 100
PMUD(2) = MISCV(18) / 100
PMUD(3) = MISCV(19) / 100
PMUD(4) = MISCV(20) / 100
PMUD(5) = MISCV(21) / 100
PMUD(6) = MISCV(22) / 100
IF (.NOT. MF) THEN
    ALYR_TH = MISCV(23)
    LYR_TH = MISCV(24)
END IF
C_MIN = MISCV(25)
C_MAX = MISCV(26)
KD = MISCV(27)

TCRF = MISCV(32)

QI_N = MISCV(49)
ALLOCATE (QI_R(QI_N))
ALLOCATE (QI_C(QI_N))
ALLOCATE (QI(QI_N))
INFLOW = 0
DO N = 1, QI_N
    QI_R(N) = MISCV((N * 3) + 47)
    QI_C(N) = MISCV((N * 3) + 48)
    QI(N) = MISCV((N * 3) + 49)
    INFLOW = INFLOW + QI(N)
END DO

S_MAX = MISCV(54)
MAXITR = MISCV(55)

IF (INT(MISCV(33) + 0.001) == 1) THEN
    USE_CLG = .TRUE.
ELSE
    USE_CLG = .FALSE.
END IF

IF (MISCV(58) > 0.999) THEN
    MUDFILE = .TRUE.
ELSE
    MUDFILE = .FALSE.
END IF

WREF = MISCV(59)
SLR = MISCV(60)

IF (USE_CLG == .TRUE.) THEN
    NR_PTS = INT(MISCV(34) + 0.001)
    ALLOCATE (SETUP_PTS(NR_PTS, 2))
    ALLOCATE (CLG_WIDTH(NR_PTS - 1))
    ALLOCATE (NG_COLS(NR_PTS - 1))

    DO N = 1, NR_PTS
        SETUP_PTS(N, 1) = MISCV((N * 2) + 33)
        SETUP_PTS(N, 2) = MISCV((N * 2) + 34)
        IF (N < NR_PTS) THEN
            CLG_WIDTH(N) = MISCV(N + 44)
        END IF
    END DO
```

```
    NG_COLS_MAX = 0
    DO N = 1, NR_PTS - 1
      NG_COLS(N) = INT((CLG_WIDTH(N) / NG_SIZE)+ 0.99)
      IF (NG_COLS(N) > NG_COLS_MAX) THEN
        NG_COLS_MAX = NG_COLS(N)
      END IF
    END DO

END IF

!Initial sediment types
IF (MUDFILE == .TRUE.) THEN
    CALL DATA_INPUT(PMUD_FILE, 1, RS, CS, GS, STYPE_FN)
    IF (RS /= ROWS .OR. CS /= COLS .OR. GS /= GRID_SIZE) THEN
        PRINT*, "Inconsistent data. Number of rows/columns or cell size "
        PRINT*, "in sediment type file does not match bed data file"
        READ*, CHR
        STOP
    END IF
ELSE
    CALL DATA_INPUT(SED_TYPE, 1, RS, CS, GS, STYPE_FN)
    IF (RS /= ROWS .OR. CS /= COLS .OR. GS /= GRID_SIZE) THEN
        PRINT*, "Inconsistent data. Number of rows/columns or cell size "
        PRINT*, "in sediment type file does not match bed data file"
        READ*, CHR
        STOP
    END IF
END IF

!************************Initialise variables*****************************

ZZ = 0
QX = 0
QY = 0

RESULTS_TIME = 0
SED_RESULTS_TIME = 0
V_RESULTS_TIME = 0


C_INT = 1000
C_TIME = C_INT
NG_SIZE = GRID_SIZE / 2.0

IF (MISCV(53) > 0.99) THEN
    WV_RESULTS = .TRUE.
ELSE
    WV_RESULTS = .FALSE.
END IF

!Mass balance total etc
MASSINT = 1000
SAND_IN = 0
SAND_OUT = 0
MUD_IN = 0
MUD_OUT = 0
MASS_BALANCE_FN = RESULTS_DIR(1:RESULTS_DIR_LEN) // "\MASS_BALANCE.TXT"
LOG_FILE_FN = RESULTS_DIR(1:RESULTS_DIR_LEN) // "\LOG.TXT"
TS_FILE_FN = RESULTS_DIR(1:RESULTS_DIR_LEN) // "\TS.TXT"
LOG_FILE = 11
MASS_FILE = 10
TS_FILE = 9
MASSTM = MASSINT
```

```
!***************************************************************************

!Read sediment data from file
IF (MF) THEN

    OPEN (1, FILE = SED_FN, STATUS = "OLD", IOSTAT = IO)
    IF (IO /= 0) THEN
        PRINT*, "Sediment data file not found"
        READ (*,10) str
        STOP
    END IF
    READ (1, 160) NR_FRACTIONS
    READ (1, 20, ADVANCE = 'NO') SMIN
    READ (1, 20) SMAX
    READ (1, 170, ADVANCE = 'NO') LYR_TH
    READ (1, 170) ALYR_TH
    NR_LAYERS = ((SMAX - SMIN) / LYR_TH) + 1

    ALLOCATE (ALYR(ROWS, COLS, NR_FRACTIONS + 1))
    ALLOCATE (SED(ROWS, COLS, NR_LAYERS, NR_FRACTIONS + 1))
    ALLOCATE (FMIN(NR_FRACTIONS + 1))
    ALLOCATE (FMAX(NR_FRACTIONS + 1))
    ALLOCATE (FI(NR_FRACTIONS + 1))
    ALLOCATE (WS(NR_FRACTIONS + 1))
    ALLOCATE (PRI(NR_FRACTIONS + 1))
    ALLOCATE (MF_SED_IN(NR_FRACTIONS + 1))
    ALLOCATE (MF_SED_OUT(NR_FRACTIONS + 1))
    ALLOCATE (MF_SSVOL(NR_FRACTIONS + 1))
    ALLOCATE (MF_SED_TOT(NR_FRACTIONS + 1))
    ALLOCATE (MF_SED_TOT1(NR_FRACTIONS + 1))

    READ (1, 10) CHR            !Column headings not read
    FMIN(1) = 1E-6
    FMAX(1) = 8E-6
    DO N = 2, NR_FRACTIONS + 1
        READ (1, 180, ADVANCE = 'NO') M     !Fraction number
        !Min fraction size in micrometres
        READ (1, 210, ADVANCE = 'NO') FMIN(N)
        READ (1, 210) FMAX(N)   !Max fraction size in micrometres
        FI(N) = SQRT(FMIN(N) * FMAX(N))
    END DO
    FMIN(1) = 1.25E-7
    FMAX(1) = 8E-6
    FI(1) = 1E-6
    READ (1, 160) NR_TYPES

    ALLOCATE (STYPE(NR_TYPES, NR_FRACTIONS))

    DO N = 1, NR_TYPES
        READ (1, 180, ADVANCE = 'NO') B
        DO M = 1, NR_FRACTIONS - 1
            READ (1, 170, ADVANCE = 'NO') STYPE(N, M)
        END DO
        READ (1, 170) STYPE(N, NR_FRACTIONS)
    END DO


    CLOSE (1)

    ALLOCATE (SED_IN(NR_FRACTIONS + 1, 2))

    DO N = 2, NR_FRACTIONS + 1
        SED_IN(N, 2) = DS_SED_INPUT * STYPE(DS_SED_TYPE, N - 1)
    END DO
```

```
    MF_SED_IN = 0
    MF_SED_OUT = 0
    MF_SSVOL = 0

    NMUD = 1
    DO N = 2, NR_FRACTIONS
        IF (FI(N) < DSAND) THEN
            NMUD = NMUD + 1
        END IF
    END DO

    ALLOCATE (SC(ROWS, COLS, NMUD))
    IF (MF) THEN
        ALLOCATE (MFDTOT(ROWS, COLS, NMUD))
        ALLOCATE (MFETOT(ROWS, COLS, NMUD))
        MFDTOT = 0
        MFETOT = 0
    END IF

END IF

!Dimension arrays for sand + mud model
IF (RUN_MODE == 1) THEN
    NMUD = 1
    NR_LAYERS = ((SMAX1 - SMIN1) / LYR_TH) + 1
    ALLOCATE (ALYR(ROWS, COLS, 2))
    ALLOCATE (SED(ROWS, COLS, NR_LAYERS, 1))
    ALLOCATE (SC(ROWS, COLS, 1))
ELSE
    NMUD = 1
    ALLOCATE (SC(ROWS, COLS, 1))
END IF

!Initialise all sediment layers &
!calculate WALL_HT based on max cell height
WALL_HT = 0
DO ROW = 1, ROWS
    DO COL = 1, COLS
        IF (GRID(ROW, COL) > WALL_HT) WALL_HT = GRID(ROW, COL)
        IF (RUN_MODE == 1) THEN
            IF (MUDFILE == .TRUE.) THEN
                ALYR(ROW, COL, 1) = (1.0 - PMUD_FILE(ROW, COL)) * ALYR_TH
                ALYR(ROW, COL, 2) = PMUD_FILE(ROW, COL) * ALYR_TH
                DO N = 1, NR_LAYERS
                    SED(ROW, COL, N, 1) = PMUD_FILE(ROW, COL)
                END DO
            ELSE
                ALYR(ROW, COL, 1) = (1.0 - PMUD(SED_TYPE(ROW, COL))) * &
                    ALYR_TH
                ALYR(ROW, COL, 2) = PMUD(SED_TYPE(ROW, COL)) * ALYR_TH
                DO N = 1, NR_LAYERS
                    SED(ROW, COL, N, 1) = PMUD(SED_TYPE(ROW, COL))
                END DO
            END IF
        ELSE
            IF (RUN_MODE == 2) THEN
                SED_TOT = 0
                DO N = 2, NR_FRACTIONS + 1
                    ALYR(ROW, COL, N) = STYPE(SED_TYPE(ROW, COL), N - 1) &
                        * ALYR_TH
                    SED_TOT = SED_TOT + STYPE(SED_TYPE(ROW, COL), N - 1)
                    DO M = 1, NR_LAYERS
                        SED(ROW, COL, M, N) = &
                            STYPE(SED_TYPE(ROW, COL), N - 1)
```

```
                    END DO
                END DO
                ALYR(ROW, COL, 1) = (1.0 - SED_TOT) * ALYR_TH
                DO M = 1, NR_LAYERS
                    SED(ROW, COL, M, 1) = 1.0 - SED_TOT
                END DO
            END IF
        END IF
    END DO
END DO
WALL_HT = WALL_HT + 10


!-------------------------------------------------------------------------
GRID_WIDTH = COLS * GRID_SIZE
GRID_LENGTH = ROWS * GRID_SIZE

IF (USE_ISIS) THEN
    ALLOCATE (TIDE(DBP, 2))
ELSE
    ALLOCATE (TIDE(NWL, 2))
END IF
ALLOCATE (WL(ROWS, COLS, NWL))
ALLOCATE (TDWL(TSPRP))
ALLOCATE (WL_SS(ROWS))
ALLOCATE (VX(ROWS, COLS))
ALLOCATE (VY(ROWS, COLS))
ALLOCATE (H(ROWS, COLS))
ALLOCATE (H1(ROWS, COLS))
ALLOCATE (MEAN_H(ROWS, COLS))
ALLOCATE (MIN_WL(ROWS, COLS))
ALLOCATE (MAX_WL(ROWS, COLS))
ALLOCATE (KX(TSPRP))
ALLOCATE (DTOT(ROWS, COLS))
ALLOCATE (ETOT(ROWS, COLS))

IF (USE_CLG == .TRUE.) THEN
    ALLOCATE (DTN(NR_PTS - 1))
    ALLOCATE (LINE_START(NR_PTS - 1, 2))
    ALLOCATE (ARC_ORIGIN(NR_PTS - 2, 2))
    ALLOCATE (LINE_INC(NR_PTS - 1))
    ALLOCATE (LINE_LEN(NR_PTS - 1))
    ALLOCATE (ARC_INC(NR_PTS - 2))
    ALLOCATE (ARC_LEN(NR_PTS - 2))

    CALL SETUP_CLG (NG_ROWS, NG_COLS, GRID_SIZE, NG_SIZE, SETUP_PTS, &
                    NR_PTS, DTN, LINE_START, ARC_ORIGIN, LINE_INC, &
                    LINE_LEN, ARC_INC, ARC_LEN)

    ALLOCATE (ROW_WTH(NG_ROWS))
    ALLOCATE (CLG_BED(NG_ROWS, NG_COLS_MAX))
    ALLOCATE (NG_LOC(NG_ROWS, NG_COLS_MAX, 2))
    ALLOCATE (NG_AREA(NG_ROWS, NG_COLS_MAX))
    ALLOCATE (CELL_LEN(NG_ROWS, NG_COLS_MAX))
    ALLOCATE (WT_XCOORDS(NG_ROWS, NG_COLS_MAX, 3))
    ALLOCATE (WT_YCOORDS(NG_ROWS, NG_COLS_MAX, 3))
    ALLOCATE (BLWT(NG_ROWS, NG_COLS_MAX, 3))
    ALLOCATE (ROW_DTN(NG_ROWS))
    ALLOCATE (ROW_CH(NG_ROWS))
    ALLOCATE (CLG_WL(NG_ROWS, TSPRP))

    ALLOCATE (XWT(ROWS, COLS, 3))
    ALLOCATE (YWT(ROWS, COLS, 3))
    ALLOCATE (CLG_WT(ROWS, COLS, 3))
```

```
CALL CREATE_CLG (ROWS, COLS, NG_ROWS, NG_COLS, NG_COLS_MAX, ROW_WTH, &
                 GRID_LENGTH, GRID_WIDTH, GRID_SIZE, NR_PTS, &
                 LINE_START, ARC_ORIGIN, LINE_INC, LINE_LEN, &
                 ARC_INC, ARC_LEN, ROW_DTN, ROW_CH, NG_SIZE, NG_LOC, &
                 NG_AREA, DTN, CELL_LEN, WT_XCOORDS, WT_YCOORDS, BLWT)
CALL GET_CLG_WTGS (LINE_START, DTN, GRID_SIZE, NG_SIZE, NG_LOC, &
                   LINE_INC, ARC_INC, LINE_LEN, ARC_LEN, ARC_ORIGIN, &
                   NR_PTS, ROWS, COLS, NG_ROWS, NG_COLS, ROW_WTH, &
                   XWT, YWT, CLG_WT)

!Populate CLG bed levels
CLG_BED = 9999
DO ROW = 1, NG_ROWS
    ROW_OS = (NG_COLS_MAX - ROW_WTH(ROW)) / 2
    DO COL = 1 + ROW_OS, NG_COLS_MAX - ROW_OS

        IF (BLWT(ROW, COL, 1) >= -1 ) THEN

            CLG_BED(ROW, COL) = (GRID(WT_XCOORDS(ROW, COL, 1), &
             WT_YCOORDS(ROW, COL, 1)) * BLWT(ROW, COL, 1)) + &
             (GRID(WT_XCOORDS(ROW, COL, 2), WT_YCOORDS(ROW, COL, 2)) &
             * BLWT(ROW, COL, 2)) + (GRID(WT_XCOORDS(ROW, COL, 3), &
             WT_YCOORDS(ROW, COL, 3)) * BLWT(ROW, COL, 3))

        END IF

    END DO
END DO



IF (USE_ISIS) THEN
    !Set up ISIS model
    IF ((NG_ROWS - 1) - &
      (INT((NG_ROWS - 1) / XSEC_SP) * XSEC_SP) > 0) THEN
        XSECS = INT((NG_ROWS - 1) / XSEC_SP) + 2
    ELSE
        XSECS = INT((NG_ROWS - 1) / XSEC_SP) + 1
    END IF

    ALLOCATE(XSEC_CH(XSECS + 1))
    XSEC_CH(XSECS + 1) = 1E25

    CALL CALC_TIDE (TIDE, TDR_SP, TDR_NP, HTIDE, DBP)

    CALL WRITE_ISIS (CLG_BED, C, C_MAX, NG_SIZE, ROW_CH, XSEC_CH, &
                     XSEC_SP, NG_ROWS, NG_COLS_MAX, ISIS_FN_SS, &
                     TIDE, DBP, INFLOW, .TRUE., .FALSE., WALL_HT)

    CALL RUN_ISIS (1, ISIS_FN_SS, ISIS_RT, .TRUE.)

    ERROR = CHECK_ISIS(1, ISIS_FN_SS)
    IF (ERROR) THEN
        PRINT 10, &
        "ISIS STEADY-STATE RUN ERROR. ENTER 'C' TO CONTINUE"
        DO
            READ (*,*) STR
            IF (STR == 'C' .OR. STR == 'c') EXIT
        END DO
    END IF

    STLEN = LEN_TRIM(ISIS_FN_SS)
    ISIS_SS_ZZS = ISIS_FN_SS(1:STLEN - 3) // "zzs"
    CALL READ_SS (1, ISIS_SS_ZZS, NG_ROWS, XSEC_SP, WL_SS)
```

```
        CALL WRITE_ISIS (CLG_BED, C, C_MAX, NG_SIZE, ROW_CH, XSEC_CH, &
                XSEC_SP, NG_ROWS, NG_COLS_MAX, ISIS_FN, TIDE, DBP, &
                INFLOW, .FALSE., INTERPOLATE, WALL_HT, WL_SS(ROWS))

        CALL RUN_ISIS (1, ISIS_FN, ISIS_RT, .FALSE.)

        ERROR = CHECK_ISIS(1, ISIS_FN)
        IF (ERROR) THEN
            PRINT 10, "ISIS UNSTEADY RUN ERROR. ENTER 'C' TO CONTINUE"
            DO
                READ (*,*) STR
                IF (STR == 'C' .OR. STR == 'c') EXIT
            END DO
        END IF

        STLEN = LEN_TRIM(ISIS_FN)
        ISIS_OUT_FN = ISIS_FN(1:STLEN - 3) // "csv"

        CALL READ_ISIS(1, ISIS_OUT_FN, ISIS_WARMUP, TSPRP, NG_ROWS, &
                XSEC_SP, WL(:, 1,:), RP, XSEC_CH, ROW_CH)

        DO ROW = 1, ROWS
            DO COL = 1, COLS
                WL(ROW, COL,:) = WL(ROW, 1,:)
            END DO
        END DO
    ELSE
        PT = 44709.87089    !Tidal period
        FT = 2 * PI / PT    !Tidal frequency
        FSN = 2 * PI / (PT * 29) !Spring/neap frequency
        MID_TIDE = HTIDE - (TDR_SP / 2.0)
        TDWL = .TRUE.
        KX = 0
        DO ROW = NG_ROWS, 1, -1
            GRID_MIN = 9999
            DO COL = 1, NG_COLS_MAX
                IF (CLG_BED(ROW, COL) < GRID_MIN) GRID_MIN = &
                    CLG_BED(ROW, COL)
            END DO
    !        WL_SS(ROW) = GRID_MIN + MIN_H
            DO N = 1, TSPRP
                IF (TDWL(N)) THEN
                    IF (ROW < NG_ROWS) THEN
                        HTOT = 0
                        M = 0
                        DO COL = 1, NG_COLS_MAX
                            IF (CLG_WL(ROW + 1, N) > &
                                CLG_BED(ROW + 1, COL)) THEN
                                HTOT = HTOT + CLG_WL(ROW + 1, N) - &
                                    CLG_BED(ROW + 1, COL)
                                M = M + 1
                            END IF
                        END DO
                        IF (M > 0) THEN
                            HMEAN = HTOT / M
                        ELSE
                            HMEAN = 0
                        END IF
                    ELSE
                        HMEAN = 999
                    END IF

                    IF (ROW < NG_ROWS) THEN
                        IF (HMEAN > 0.9 * MIN_H) THEN
                            CT = SQRT(G * HMEAN)
```

```
                            KX(N) = KX(N) + (2 * PII * (ROW_CH(ROW + 1)- &
                                    ROW_CH(ROW)) / (CT * PT))
                        END IF
                    END IF
                    IF (HMEAN > MIN_H) THEN
                        IF (RP == "True") THEN
                            TD_RANGE = TDR_SP / 2.0
                        ELSE
                            TD_RANGE = ((TDR_SP + TDR_NP) / 4.0) + &
                          (((TDR_SP - TDR_NP) / 4.0) * SIN(FSN * N * TSS))
                        END IF
                        CLG_WL(ROW, N) = MID_TIDE + &
                        (TD_RANGE * SIN(KX(N) - (FT * N * TSS)))
                    ELSE
                        CLG_WL(ROW, N) = -9999
                    END IF
                    IF (CLG_WL(ROW, N) < GRID_MIN + MIN_H) THEN
                        TDWL(N) = .FALSE.
                        CLG_WL(ROW, N) = GRID_MIN + MIN_H
                    END IF
                ELSE
                    CLG_WL(ROW, N) = GRID_MIN + MIN_H
                END IF
            END DO
        END DO
    END IF


    DO ROW = 1, ROWS
        DO COL = 1, COLS
            XYMIN = MIN(XWT(ROW, COL, 1), XWT(ROW, COL, 2), &
                    XWT(ROW, COL, 3), YWT(ROW, COL, 1), &
                    YWT(ROW, COL, 2), YWT(ROW, COL, 3))
            IF (XYMIN > 1E-6) THEN
                DO T = 1, TSPRP

WL(ROW, COL, T) =    (CLG_WL(YWT(ROW, COL, 1), T) * CLG_WT(ROW, COL, 1))+ &
                     (CLG_WL(YWT(ROW, COL, 2), T) * CLG_WT(ROW, COL, 2))+ &
                     (CLG_WL(YWT(ROW, COL, 3), T) * CLG_WT(ROW, COL, 3))

                END DO
            ELSE
                WL(ROW, COL,:) = -9999
            END IF
        END DO
    END DO
ELSE

    !If interpolated grid not used, use cross sections based on
    !successive rows of cells
    ALLOCATE (ROW_CH(ROWS))
    DO N = 1, ROWS
      ROW_CH(N) = (N - 0.5) * GRID_SIZE
    END DO
    !USE_ISIS = .FALSE.
    IF (USE_ISIS) THEN
        !Set up ISIS model
        IF ((ROWS - 1) - (INT((ROWS - 1) / XSEC_SP) * XSEC_SP) > 0) THEN
          XSECS = INT((ROWS - 1) / XSEC_SP) + 2
        ELSE
          XSECS = INT((ROWS - 1) / XSEC_SP) + 1
        END IF

        ALLOCATE(XSEC_CH(XSECS + 1))
        XSEC_CH(XSECS + 1) = 1E25
```

```
        CALL CALC_TIDE (TIDE, TDR_SP, TDR_NP, HTIDE, DBP)

        CALL WRITE_ISIS (GRID, C, C_MAX, GRID_SIZE, ROW_CH, XSEC_CH, &
                         XSEC_SP, ROWS, COLS, ISIS_FN_SS, TIDE, DBP, &
                         INFLOW, .TRUE., .FALSE., WALL_HT)

        CALL RUN_ISIS (1, ISIS_FN_SS, ISIS_RT, .TRUE.)

        ERROR = CHECK_ISIS(1, ISIS_FN_SS)
        IF (ERROR) THEN
            PRINT 10, "ISIS STEADY-STATE RUN ERROR. ENTER 'C' TO CONTINUE"
            DO
                READ (*,*) STR
                IF (STR == 'C' .OR. STR == 'c') EXIT
            END DO
        END IF

        STLEN = LEN_TRIM(ISIS_FN_SS)
        ISIS_SS_ZZS = ISIS_FN_SS(1:STLEN - 3) // "zzs"
        CALL READ_SS (1, ISIS_SS_ZZS, ROWS, XSEC_SP, WL_SS)

        CALL WRITE_ISIS (GRID, C, C_MAX, GRID_SIZE, ROW_CH, XSEC_CH, &
                    XSEC_SP, ROWS, COLS, ISIS_FN, TIDE, DBP, &
                    INFLOW, .FALSE., INTERPOLATE, WALL_HT, WL_SS(ROWS))

        CALL RUN_ISIS (1, ISIS_FN, ISIS_RT, .FALSE.)

        ERROR = CHECK_ISIS(1, ISIS_FN)
        IF (ERROR) THEN
            PRINT 10, "ISIS UNSTEADY RUN ERROR. ENTER 'C' TO CONTINUE"
            DO
                READ (*,*) STR
                IF (STR == 'C' .OR. STR == 'c') EXIT
            END DO
        END IF

        STLEN = LEN_TRIM(ISIS_FN)
        ISIS_OUT_FN = ISIS_FN(1:STLEN - 3) // "csv"

        CALL READ_ISIS(1, ISIS_OUT_FN, ISIS_WARMUP, TSPRP, ROWS, &
                XSEC_SP, WL(:, 1,:), RP, XSEC_CH, ROW_CH)

        DO ROW = 1, ROWS
            DO COL = 1, COLS
                WL(ROW, COL,:) = WL(ROW, 1,:)
            END DO
        END DO

    ELSE
        PT = 44709.87089    !Tidal period
        FT = 2 * PI / PT    !Tidal frequency
        FSN = 2 * PI / (PT * 29) !Spring/neap frequency
        MID_TIDE = HTIDE - (TDR_SP / 2.0)
        TDWL = .TRUE.
        KX = 0
        DO ROW = ROWS, 1, -1
            GRID_MIN = 1E9
            DO COL = 1, COLS
                IF (GRID(ROW, COL) < GRID_MIN) GRID_MIN = GRID(ROW, COL)
            END DO

            DO N = 1, TSPRP
                IF (TDWL(N)) THEN
                    IF (ROW < ROWS) THEN
```

```
                        HTOT = 0
                        M = 0
                        DO COL = 1, COLS
                            IF (WL(ROW + 1, 1, N) > &
                                    GRID(ROW + 1, COL)) THEN
                                HTOT = HTOT + WL(ROW + 1, 1, N) - &
                                        GRID(ROW + 1, COL)
                                M = M + 1
                            END IF
                        END DO
                        IF (M > 0) THEN
                            HMEAN = HTOT / M
                        ELSE
                            HMEAN = 0
                        END IF
                    ELSE
                        HMEAN = 999
                    END IF

                    IF (ROW < ROWS) THEN
                        IF (HMEAN > 0.9 * MIN_H) THEN
                            CT = SQRT(G * HMEAN)
                            KX(N) = KX(N) + &
                                    (2* PII * GRID_SIZE / (CT * PT))
                        END IF
                    END IF
                    IF (HMEAN > MIN_H) THEN
                        IF (RP == "True") THEN
                            TD_RANGE = TDR_SP / 2.0
                        ELSE
                            TD_RANGE = ((TDR_SP + TDR_NP) / 4.0) + &
                     (((TDR_SP - TDR_NP) / 4.0) * SIN(FSN * N * TSS))
                        END IF
                        WL(ROW,:, N) = MID_TIDE + (TD_RANGE * SIN(KX(N) - &
                                        (FT * N * TSS)))
                    ELSE
                        WL(ROW,:, N) = -9999
                    END IF
                    IF (WL(ROW, 1, N) < GRID_MIN + MIN_H) THEN
                        TDWL(N) = .FALSE.
                        WL(ROW,:, N) = GRID_MIN + MIN_H
                    END IF
                ELSE
                    WL(ROW,:, N) = GRID_MIN + MIN_H
                END IF
            END DO
        END DO
    END IF
END IF

!Calculate max and min water level for each cell
MIN_WL = 1E10
MAX_WL = -1E10
DO  ROW = 1, ROWS
    DO COL = 1, COLS
        DO T = 1, TSPRP
            IF (WL(ROW, COL, T) < MIN_WL(ROW, COL)) THEN
                MIN_WL(ROW, COL) = WL(ROW, COL, T)
            END IF
            IF (WL(ROW, COL, T) > MAX_WL(ROW, COL)) THEN
                MAX_WL(ROW, COL) = WL(ROW, COL, T)
            END IF
        END DO
    END DO
END DO
```

```
OPEN (LOG_FILE, FILE = LOG_FILE_FN)
WRITE (LOG_FILE, 10) CONTRL_FN

!Mass balance header + initial calculation and output
OPEN (MASS_FILE, FILE = MASS_BALANCE_FN)

WRITE (MASS_FILE, 10, ADVANCE = 'NO') "                TIME             "
WRITE (MASS_FILE, 10, ADVANCE = 'NO') "SAND IN            SAND OUT      "
WRITE (MASS_FILE, 10, ADVANCE = 'NO') "        MUD IN              "
WRITE (MASS_FILE, 10, ADVANCE = 'NO') "MUD OUT TOTAL SAND VOLUME    "
WRITE (MASS_FILE, 10) "TOTAL MUD VOLUME  TOTAL CURRENT VOL."

IF (TS_RESULTS) THEN
    OPEN (TS_FILE, FILE = TS_FILE_FN)
    WRITE (TS_FILE, 10) "Time series output for:"
    WRITE (TS_FILE, 10) CONTRL_FN
    WRITE (TS_FILE, 10, ADVANCE = 'NO') "ROW = "
    WRITE (TS_FILE, 120) TS_ROW
    WRITE (TS_FILE, 10, ADVANCE = 'NO') "COL = "
    WRITE (TS_FILE, 120) TS_COL
    WRITE (TS_FILE, 10) "           TIME             VX             VY
H"
END IF

DATUM = -100
VOL = 0
SAND_TOT = 0
MUD_TOT = 0
DO ROW = 1, ROWS
    DO COL = 1, COLS
        VOL = VOL + ((GRID(ROW, COL) - SMIN1) * GRID_SIZE * GRID_SIZE)
    END DO
END DO
WRITE (MASS_FILE, 340, ADVANCE = 'NO') 0
IF (RUN_MODE == 1) THEN
    WRITE (MASS_FILE, 340, ADVANCE = 'NO') SAND_IN
    WRITE (MASS_FILE, 340, ADVANCE = 'NO') SAND_OUT
    WRITE (MASS_FILE, 340, ADVANCE = 'NO') MUD_IN
    WRITE (MASS_FILE, 340, ADVANCE = 'NO') MUD_OUT
    DO ROW = 1, ROWS
        DO COL = 1, COLS

            ALYR_BASE = GRID(ROW, COL) - ALYR_TH
            R_LYRS = (ALYR_BASE - SMIN1) / LYR_TH
            N_LYRS = INT(R_LYRS)

            DO N = 1, N_LYRS
                SAND_TOT = SAND_TOT + ((1 - SED(ROW, COL, N, 1)) * &
                           LYR_TH * GRID_SIZE * GRID_SIZE)
                MUD_TOT = MUD_TOT + (SED(ROW, COL, N, 1) * LYR_TH * &
                          GRID_SIZE * GRID_SIZE)
            END DO

            SAND_TOT = SAND_TOT + ((1 - SED(ROW, COL, N_LYRS + 1, 1)) * &
                    LYR_TH * (R_LYRS - N_LYRS) * GRID_SIZE * GRID_SIZE)
            MUD_TOT = MUD_TOT + (SED(ROW, COL, N_LYRS + 1, 1) * LYR_TH * &
                    (R_LYRS - N_LYRS) * GRID_SIZE * GRID_SIZE)

            SAND_TOT = SAND_TOT + (ALYR(ROW, COL, 1) * GRID_SIZE * &
                       GRID_SIZE)
            MUD_TOT = MUD_TOT + (ALYR(ROW, COL, 2) * GRID_SIZE * &
                       GRID_SIZE)
```

```
        END DO
    END DO
    WRITE (MASS_FILE, 340, ADVANCE = 'NO') SAND_TOT
    WRITE (MASS_FILE, 340, ADVANCE = 'NO') MUD_TOT

ELSE
    IF (RUN_MODE == 2) THEN

        DO N = 1, NR_FRACTIONS + 1
            WRITE (MASS_FILE, 340, ADVANCE = 'NO') MF_SED_IN(N)
            WRITE (MASS_FILE, 340, ADVANCE = 'NO') MF_SED_OUT(N)
        END DO

        MF_SED_TOT = 0
        DO ROW = 1, ROWS
            DO COL = 1, COLS

                ALYR_BASE = GRID(ROW, COL) - ALYR_TH
                R_LYRS = (ALYR_BASE - SMIN1) / LYR_TH
                N_LYRS = INT(R_LYRS)

                DO N = 1, NR_FRACTIONS + 1
                    DO M = 1, N_LYRS
                        MF_SED_TOT(N) = MF_SED_TOT(N) + &
    (SED(ROW, COL, M, N) * LYR_TH * GRID_SIZE * GRID_SIZE)
                    END DO
                    MF_SED_TOT(N) = MF_SED_TOT(N) + &
                        (SED(ROW, COL, N_LYRS + 1, N) * &
                        LYR_TH * (R_LYRS - N_LYRS) * &
                        GRID_SIZE * GRID_SIZE)
                    MF_SED_TOT(N) = MF_SED_TOT(N) + (ALYR(ROW, COL, N) &
                                    * GRID_SIZE * GRID_SIZE)
                END DO

            END DO
        END DO

        DO N = 1, NR_FRACTIONS + 1
            WRITE (MASS_FILE, 340, ADVANCE = 'NO') MF_SED_TOT(N)
        END DO
        MF_SED_TOT1 = MF_SED_TOT

    END IF
END IF
WRITE (MASS_FILE, 340) VOL

!Wind speed, direction & probability
NWS1 = 2
NWS2 = ((NWS1 * 2) + 1) * ((NWS1 * 2) + 1)
N1 = -NWS1 - 1
N2 = -NWS1
PTOT = 0


ALLOCATE (P(NWS2))
ALLOCATE (WSP(NWS2))
ALLOCATE (WD(NWS2))
ALLOCATE (WAVE_H(ROWS, COLS, NWS2))
ALLOCATE (WAVE_T(ROWS, COLS, NWS2))
ALLOCATE (WAVE_DIR(ROWS, COLS, NWS2))

DO N3 = 1, NWS2
    IF (N1 < 2) THEN
        N1 = N1 + 1
    ELSE
```

```
            N1 = -NWS1
            N2 = N2 + 1
        END IF
        WS1 = (REAL(N1) * SPSD) + PWS
        WS2 = REAL(N2) * SPSD
        P1 = (1 / SQRT(2 * PII)) * EXP(-(REAL(N1) ** 2) / 2)
        P2 = (1 / SQRT(2 * PII)) * EXP(-(REAL(N2) ** 2) / 2)
        P(N3) = P1 * P2
        PTOT = PTOT + P(N3)
        WSP(N3) = SQRT((WS1**2.0) + (WS2**2.0))
        IF (WS1 > 0) THEN
            IF (ABS(WS1) > 1E-6) THEN
                WD(N3) = ATAN(WS2 / WS1) + (SPD * PII / 180)
            ELSE
                IF (WS2 > 0) THEN
                    WD(N3) = (PII / 2.0) + (SPD * PII / 180)
                ELSE
                    WD(N3) = -(PII / 2.0) + (SPD * PII / 180)
                END IF
            END IF
        ELSE
            IF (ABS(WS1) > 1E-6) THEN
                WD(N3) = ATAN(WS2 / WS1) + (SPD * PII / 180) + PII
            ELSE
                IF (WS2 > 0) THEN
                    WD(N3) = -(PII / 2.0) + (SPD * PII / 180) + PII
                ELSE
                    WD(N3) = (PII / 2.0) + (SPD * PII / 180) + PII
                END IF
            END IF
        END IF

        IF (WD(N3) < 0) WD(N3) = WD(N3) + (2.0 * PII)
        IF (WD(N3) > (2.0 * PII)) WD(N3) = WD(N3) - (2.0 * PII)
    END DO

    DO N = 1, NWS2
        P(N) = P(N) / PTOT
    END DO
    !----------------------------MAIN PROGRAM LOOP START----------------------


    !Run model for required time period

    T1 = 1
    T2 = 2
    TIME = 0
    TS_NR = 1
    SC = DSSC
    DTOT = 0
    ETOT = 0

    DO WHILE (TIME <= RUN_TIME)

        !Calculate fluvial & tidal flow for each cell
        CALL CALCFLOW (GRID, GRID_SIZE, WL(:,:,T1), WL(:,:,T2), QX(:,:,T1), &
        QY(:,:,T1), H, MIN_H, H1, C, TSS, ROWS, COLS, QI, QI_N, QI_R, QI_C, &
        S_MIN, S_MAX, XP, MAXITR, LOG_FILE, TIME)

        !Calculate wave height and period for each cell
        DO N = 1, NWS2
            IF (WSP(N) > 1E-3) THEN
                CALL CALC_WAVES (ROWS, COLS, GRID_SIZE, H(:,:), MIN_H, &
                WSP(N), WD(N), BDY_FETCH, WAVE_H(:,:,N), WAVE_T(:,:,N), &
                WAVE_DIR(:,:,N), BM, WREF)
```

```
    ELSE
        WAVE_H(:,:,N) = 0
        WAVE_T(:,:,N) = 0
        WAVE_DIR(:,:,N) = 0
    END IF
    DO ROW = 1, ROWS
        DO COL = 1, COLS
            IF (WAVE_H(ROW, COL, N) < 1E-3) WAVE_H(ROW, COL, N) = 1E-3
            IF (WAVE_T(ROW, COL, N) < 1E-3) WAVE_T(ROW, COL, N) = 1E-3
        END DO
    END DO
END DO


!Calculate mean water depth in each cell
DO ROW = 1, ROWS
    DO COL = 1, COLS

        IF (MAX_WL(ROW, COL) > GRID(ROW, COL) + H1(ROW, COL)) THEN
            IF (MIN_WL(ROW, COL) > GRID(ROW, COL) + H1(ROW, COL)) THEN
                MEAN_H(ROW, COL) = ((MAX_WL(ROW, COL) + &
                MIN_WL(ROW, COL)) / 2) - GRID(ROW, COL)
            ELSE
                MEAN_H(ROW, COL) = (MAX_WL(ROW, COL) - &
                GRID(ROW, COL) + H1(ROW, COL)) / 2
            END IF
        ELSE
            IF (H1(ROW, COL) > MIN_H) THEN
                MEAN_H(ROW, COL) = H1(ROW, COL)
            ELSE
                MEAN_H(ROW, COL) = 0
            END IF
        END IF
        IF (H(ROW, COL) / 2 > MEAN_H(ROW, COL)) &
        MEAN_H(ROW, COL) = H(ROW, COL) / 2
    END DO
END DO

!Sediment transport and morphological update calculation

IF (RUN_MODE == 1) THEN
    !Single fraction sand + mud
    CALL CALC_MUDST_SF(GRID, FGRID, ALYR, SED(:,:,:,1), ALYR_TH, &
         LYR_TH, NR_LAYERS, SMIN1, ROWS, COLS, TSS, TSPRP, &
         GRID_SIZE, QX(:,:,T1), QY(:,:,T1), H, MEAN_H, MIN_H, D50, &
         D90, DS, PS, KD, DMUD, TCRF, WAVE_H, WAVE_T, WAVE_DIR, &
         BM, BMAX, DSSC, DS_SED_INPUT, VX, VY, MAX_V, LTFR, TE, &
         TD, SC(:,:,1), NWS2, P, DTOT, ETOT, SAND_IN, SAND_OUT, &
         MUD_IN, MUD_OUT)
ELSE

    !Multi-fraction
    CALL CALC_ST_MF(GRID, ALYR, SED, FGRID, ROWS, COLS, GRID_SIZE, &
         LYR_TH, ALYR_TH, NR_FRACTIONS, SMIN, NR_LAYERS, FI, FMIN, &
         FMAX, STYPE, SED_IN, DSSC, DS, DMUD, PS, TSS, QX(:,:,T1), &
         QY(:,:,T1), H, MIN_H, MEAN_H, WAVE_H, WAVE_T, WAVE_DIR, &
         BMAX, BM, KB, LTFR, VX, VY, TSPRP, KD, TD, TE, MF_SED_IN, &
         MF_SED_OUT, DTOT, ETOT, MF_SSVOL, NWS2, P, SC(:,:,1))

END IF


IF (T1 == TSPRP) THEN
    !At end of tidal cycle or spring/neap cycle
    !Recalculate suspended sediment concentrations
```

```
MAXCOR = 1
CTOL = 0.01 / (ROWS + COLS)
ITT = 0
DO WHILE ((MAXCOR > CTOL) .AND. (ITT < 10000))
    ITT = ITT + 1
    MAXCOR = 0
    DO ROW = 1, ROWS
        DO COL = 1, COLS
            IF (MEAN_H(ROW, COL) > MIN_H) THEN

                !No erosion below minimum bed level
                MAXER = GRID(ROW, COL) - FGRID(ROW, COL)
                IF (MAXER < 0) MAXER = 0

                IF (ETOT(ROW, COL) > MAXER) THEN
                    ETOT1 = MAXER
                ELSE
                    ETOT1 = ETOT(ROW, COL)
                END IF

                IF (ROW < ROWS) THEN
                    C1 = SC(ROW + 1, COL, 1)
                    IF (MEAN_H(ROW + 1, COL) > MIN_H) THEN
                        K1 = KD * MIN(MEAN_H(ROW + 1, COL), &
                        MEAN_H(ROW, COL)) / GRID_SIZE
                    ELSE
                        K1 = 0
                    END IF
                ELSE
                    C1 = DSSC
                    K1 = KD * MEAN_H(ROW, COL) / GRID_SIZE
                END IF

                IF (ROW > 1) THEN
                    C2 = SC(ROW - 1, COL, 1)
                    IF (MEAN_H(ROW - 1, COL) > MIN_H) THEN
                        K2 = KD * MIN(MEAN_H(ROW - 1, COL), &
                        MEAN_H(ROW, COL)) / GRID_SIZE
                    ELSE
                        K2 = 0
                    END IF
                ELSE
                    K2 = 0
                END IF

                IF (COL < COLS) THEN
                    C3 = SC(ROW, COL + 1, 1)
                    IF (MEAN_H(ROW, COL + 1) > MIN_H) THEN
                        K3 = KD * MIN(MEAN_H(ROW, COL + 1), &
                        MEAN_H(ROW, COL)) / GRID_SIZE
                    ELSE
                        K3 = 0
                    END IF
                ELSE
                    K3 = 0
                END IF

                IF (COL > 1) THEN
                    C4 = SC(ROW, COL - 1, 1)
                    IF (MEAN_H(ROW, COL - 1) > MIN_H) THEN
                        K4 = KD * MIN(MEAN_H(ROW, COL - 1), &
                        MEAN_H(ROW, COL)) / GRID_SIZE
                    ELSE
                        K4 = 0
```

```
                        END IF
                    ELSE
                        K4 = 0
                    END IF

                    IF ((K1 + K2 + K3 + K4) > 1E-6 * KD) THEN
                        SC1 = ((K1 * C1) + (K2 * C2) + (K3 * C3) + &
                        (K4 * C4) + (ETOT1 / (TSS * TSPRP))) / &
                        ((K1 + K2 + K3 + K4) - (DTOT(ROW, COL) / &
                        (TSS * TSPRP)))
                    ELSE
                        SC1 = 0
                    END IF

                    IF (SC(ROW, COL, 1) > (1E-6 * DSSC)) THEN
                        IF (ABS((SC1 - SC(ROW, COL, 1)) / &
                        SC(ROW, COL, 1)) > MAXCOR) THEN
                            MAXCOR = ABS((SC1 - SC(ROW, COL, 1)) / &
                            SC(ROW, COL, 1))
                        END IF
                    ELSE
                        IF (SC1 > (1E-6 * DSSC)) THEN
                            MAXCOR = CTOL + 9999
                        ELSE
                            SC1 = 0
                        END IF
                    END IF

                    SC(ROW, COL, 1) = SC1

                END IF
            END DO

        END DO
    END DO
    DTOT = 0
    ETOT = 0

    !If calculation ended without converging write details to log file
    IF (ITT >= MAXITR) THEN
        WRITE (LOG_FILE, 10, ADVANCE = 'NO') &
        "CONCENTRATION FIELD FAILED TO CONVERGE AFTER "
        WRITE (LOG_FILE, 70, ADVANCE = 'NO') MAXITR
        WRITE (LOG_FILE, 10, ADVANCE = 'NO') " ITERATIONS AT TIME: "
        WRITE (LOG_FILE, 60, ADVANCE = 'NO') TIME
        WRITE (LOG_FILE, 10, ADVANCE = 'NO') " HOURS. MAX C RATIO = "
        WRITE (LOG_FILE, 300) MAXCOR
    END IF


END IF

!Salt marsh extent calculation
IF (MRSH) THEN
    CALL CALC_MARSH(ROWS, COLS, C, C_MIN, C_MAX, BM, GRID, &
    WL(:,:,T1), WL(:,:,T2), TSS, TIME, SUB_MIN, SUB_MAX, SM1, SM2)
END IF

!Water levels recalculated at intervals of C_INT
IF (TIME > C_TIME) THEN

    !Increase in water levels due to sea level rise

    HTIDE = HTIDE + ((SLR / 1000) * (C_INT / 8760))
```

```
        C_TIME = C_TIME + C_INT

        IF (USE_CLG == .TRUE.) THEN

            IF (USE_ISIS) THEN


            ELSE

                !Populate CLG bed levels
                CLG_BED = 9999
                DO ROW = 1, NG_ROWS
                    ROW_OS = (NG_COLS_MAX - ROW_WTH(ROW)) / 2
                    DO COL = 1 + ROW_OS, NG_COLS_MAX - ROW_OS

                        IF (BLWT(ROW, COL, 1) >= -1 ) THEN

                            CLG_BED(ROW, COL) = &
                            (GRID(WT_XCOORDS(ROW, COL, 1), &
                            WT_YCOORDS(ROW, COL, 1)) * &
                            BLWT(ROW, COL, 1)) + &
                            (GRID(WT_XCOORDS(ROW, COL, 2), &
                            WT_YCOORDS(ROW, COL, 2)) * &
                            BLWT(ROW, COL, 2)) + &
                            (GRID(WT_XCOORDS(ROW, COL, 3), &
                            WT_YCOORDS(ROW, COL, 3)) * &
                            BLWT(ROW, COL, 3))

                        END IF

                    END DO
                END DO

                PT = 44709.87089
                FT = 2 * PI / PT
                FSN = 2 * PI / (PT * 29)
                MID_TIDE = HTIDE - (TDR_SP / 2.0)
                TDWL = .TRUE.
                KX = 0
                DO ROW = NG_ROWS, 1, -1
                        GRID_MIN = 9999
                    DO COL = 1, NG_COLS_MAX
                        IF (CLG_BED(ROW, COL) < GRID_MIN) GRID_MIN = &
                        CLG_BED(ROW, COL)
                    END DO
!                    WL_SS(ROW) = GRID_MIN + MIN_H
                    DO N = 1, TSPRP
                        IF (TDWL(N)) THEN
                            IF (ROW < NG_ROWS) THEN
                                HTOT = 0
                                M = 0
                                DO COL = 1, COLS
                                    IF (CLG_WL(ROW + 1, N) > &
                                    CLG_BED(ROW + 1, COL)) THEN
                                        HTOT = HTOT + &
                                        CLG_WL(ROW + 1, N) - &
                                        CLG_BED(ROW + 1, COL)
                                        M = M + 1
                                    END IF
                                END DO
                                IF (M > 0) THEN
                                    HMEAN = HTOT / M
                                ELSE
                                    HMEAN = 0
                                END IF
```

```
                            ELSE
                                HMEAN = 9999
                            END IF

                            IF (ROW < NG_ROWS) THEN
                                IF (HMEAN > 0.9 * MIN_H) THEN
                                    CT = SQRT(G * HMEAN)
                                    KX(N) = KX(N) + (2 * PII * &
                                    (ROW_CH(ROW + 1) - ROW_CH(ROW)) / &
                                    (CT * PT))
                                END IF
                            END IF
                            IF (HMEAN > MIN_H) THEN
                                IF (RP == "True") THEN
                                    TD_RANGE = TDR_SP / 2.0
                                ELSE
                                    TD_RANGE = ((TDR_SP + TDR_NP) / 4.0) &
                                    + (((TDR_SP - TDR_NP) / 4.0) * &
                                    SIN(FSN * N * TSS))
                                END IF
                                CLG_WL(ROW, N) = MID_TIDE + (TD_RANGE * &
                                SIN(KX(N) - (FT * N * TSS)))
                            ELSE
                                CLG_WL(ROW, N) = -9999
                            END IF
                            IF (CLG_WL(ROW, N) < GRID_MIN + MIN_H) THEN
                                TDWL(N) = .FALSE.
                                CLG_WL(ROW, N) = GRID_MIN + MIN_H
                            END IF
                        ELSE
                            CLG_WL(ROW, N) = GRID_MIN + MIN_H
                        END IF
                    END DO
                END DO
            END IF

            DO ROW = 1, ROWS
                DO COL = 1, COLS
                    XYMIN = MIN(XWT(ROW, COL, 1), XWT(ROW, COL, 2), &
                    XWT(ROW, COL, 3), YWT(ROW, COL, 1), &
                    YWT(ROW, COL, 2), YWT(ROW, COL, 3))
                    IF (XYMIN > 0) THEN
                        DO T = 1, TSPRP

WL(ROW, COL, T) = (CLG_WL(YWT(ROW, COL, 1), T) * CLG_WT(ROW, COL, 1)) + &
                  (CLG_WL(YWT(ROW, COL, 2), T) * CLG_WT(ROW, COL, 2)) + &
                  (CLG_WL(YWT(ROW, COL, 3), T) * CLG_WT(ROW, COL, 3))

                        END DO
                    ELSE
                        WL(ROW, COL,:) = -9999
                    END IF
                END DO
            END DO
        ELSE

            IF (USE_ISIS) THEN

                CALL WRITE_ISIS (GRID, C, C_MAX, GRID_SIZE, ROW_CH, &
                XSEC_CH, XSEC_SP, ROWS, COLS, ISIS_FN_SS, TIDE, DBP, &
                INFLOW, .TRUE., .FALSE., WALL_HT)

                CALL RUN_ISIS (1, ISIS_FN_SS, ISIS_RT, .TRUE.)

                ERROR = CHECK_ISIS(1, ISIS_FN_SS)
```

```
        STLEN = LEN_TRIM(ISIS_FN_SS)
        ISIS_SS_ZZS = ISIS_FN_SS(1:STLEN - 3) // "zzs"
        IF (ERROR) THEN
            IF (ZZ == 1) THEN
                PRINT 10, &
              "ISIS STEADY-STATE RUN ERROR. ENTER 'C' TO CONTINUE"
                DO
                    READ (*,*) STR
                    IF (STR == 'C' .OR. STR == 'c') EXIT
                END DO
                CALL READ_SS (1, ISIS_SS_ZZS, ROWS, XSEC_SP, &
                WL_SS)
            END IF
        ELSE
            CALL READ_SS (1, ISIS_SS_ZZS, ROWS, XSEC_SP, WL_SS)
        END IF

        CALL WRITE_ISIS (GRID, C, C_MAX, GRID_SIZE, ROW_CH, &
        XSEC_CH, XSEC_SP, ROWS, COLS, ISIS_FN, TIDE, DBP, &
        INFLOW, .FALSE., INTERPOLATE, WALL_HT, WL_SS(ROWS))

        CALL RUN_ISIS (1, ISIS_FN, ISIS_RT, .FALSE.)

        ERROR = CHECK_ISIS(1, ISIS_FN)
        IF (ERROR) THEN
            IF (ZZ == 1) THEN
                PRINT 10, &
                "ISIS UNSTEADY RUN ERROR. ENTER 'C' TO CONTINUE"
                DO
                    READ (*,*) STR
                    IF (STR == 'C' .OR. STR == 'c') EXIT
                END DO
                CALL READ_ISIS(1, ISIS_OUT_FN, ISIS_WARMUP, &
                      TSPRP, ROWS, XSEC_SP, WL(:, 1,:), RP, &
                      XSEC_CH, ROW_CH)
            END IF
        ELSE
            CALL READ_ISIS(1, ISIS_OUT_FN, ISIS_WARMUP, TSPRP, &
                 ROWS, XSEC_SP, WL(:, 1,:), RP, XSEC_CH, ROW_CH)
        END IF

        DO ROW = 1, ROWS
            DO COL = 1, COLS
                WL(ROW, COL,:) = WL(ROW, 1,:)
            END DO
        END DO

    ELSE

        PT = 44709.87089
        FT = 2 * PI / PT
        FSN = 2 * PI / (PT * 29)
        MID_TIDE = HTIDE - (TDR_SP / 2.0)
        TDWL = .TRUE.
        KX = 0
        DO ROW = ROWS, 1, -1
                GRID_MIN = 1E9
            DO COL = 1, COLS
                IF (GRID(ROW, COL) < GRID_MIN) GRID_MIN = &
                   GRID(ROW, COL)
            END DO

            DO N = 1, TSPRP
                IF (TDWL(N)) THEN
                    IF (ROW < ROWS) THEN
```

```
                                HTOT = 0
                                M = 0
                                DO COL = 1, COLS
                                    IF (WL(ROW + 1, 1, N) > &
                                    GRID(ROW + 1, COL)) THEN
                                        HTOT = HTOT + WL(ROW + 1, 1, N)- &
                                               GRID(ROW + 1, COL)
                                        M = M + 1
                                    END IF
                                END DO
                                IF (M > 0) THEN
                                    HMEAN = HTOT / M
                                ELSE
                                    HMEAN = 0
                                END IF
                            ELSE
                                HMEAN = 999
                            END IF

                            IF (ROW < ROWS) THEN
                                IF (HMEAN > 0.9 * MIN_H) THEN
                                    CT = SQRT(G * HMEAN)
                                    KX(N) = KX(N) + (2* PII * GRID_SIZE &
                                    / (CT * PT))
                                END IF
                            END IF
                            IF (HMEAN > MIN_H) THEN
                                IF (RP == "True") THEN
                                    TD_RANGE = TDR_SP / 2.0
                                ELSE
                                    TD_RANGE = ((TDR_SP + TDR_NP) / 4.0) &
                                    + (((TDR_SP - TDR_NP) / 4.0) * &
                                    SIN(FSN * N * TSS))
                                END IF
                                WL(ROW,:, N) = MID_TIDE + (TD_RANGE * &
                                SIN(KX(N) - (FT * N * TSS)))
                            ELSE
                                WL(ROW,:, N) = -9999
                            END IF
                            IF (WL(ROW, 1, N) < GRID_MIN + MIN_H) THEN
                                TDWL(N) = .FALSE.
                                WL(ROW,:, N) = GRID_MIN + MIN_H
                            END IF
                        ELSE
                            WL(ROW,:, N) = GRID_MIN + MIN_H
                        END IF
                    END DO
                END DO
            END IF
        END IF
    END IF

    IF (TS_RESULTS) THEN
        IF (TS_NR <= TSPRP) THEN
            WRITE (TS_FILE, 60, ADVANCE = 'NO') TIME
            WRITE (TS_FILE, 60, ADVANCE = 'NO') VX(TS_ROW, TS_COL)
            WRITE (TS_FILE, 60, ADVANCE = 'NO') VY(TS_ROW, TS_COL)
            WRITE (TS_FILE, 60) H(TS_ROW, TS_COL)
        ELSE
            CLOSE (TS_FILE)
            TS_RESULTS = .FALSE.
        END IF
    END IF

    IF (BD_RESULTS) THEN
```

```
        !Output of current bed level data at specified intervals
        IF (TIME >= RESULTS_TIME) THEN

            WRITE (CHR, 120) INT(TIME)
            CHR = ADJUSTL(CHR)
            RESULTS_FN = RESULTS_DIR(1:RESULTS_DIR_LEN) // &
            "\GRID_" // TRIM(CHR) // ".ASC"
            CALL GRID_OUTL(ROWS, COLS, GRID_SIZE, GRID, RESULTS_FN)
            IF (WV_RESULTS == .TRUE.) THEN
                DO N = 1, NWS2
                    WRITE (CHR1, 180) N
                    CHR1 = ADJUSTL(CHR1)
                    RESULTS_FN = RESULTS_DIR(1:RESULTS_DIR_LEN) // &
                    "\WAVEH_" // TRIM(CHR) // "_" // TRIM(CHR1) // ".ASC"
                    CALL GRID_OUT(ROWS, COLS, GRID_SIZE, &
                                  WAVE_H(:,:,N), RESULTS_FN)
                END DO
            END IF

            IF (MRSH == .TRUE.) THEN
                RESULTS_FN = RESULTS_DIR(1:RESULTS_DIR_LEN) // "\MARSH_" &
                                          // TRIM(CHR) // ".ASC"
                CALL GRID_OUT(ROWS, COLS, GRID_SIZE, BM, RESULTS_FN)
            END IF
            RESULTS_TIME = RESULTS_TIME + RESULTS_INT

        END IF
    END IF

    IF (TIME >= MASSTM) THEN
        !Output mass balance totals to file
        VOL = 0
        SAND_TOT = 0
        MUD_TOT = 0
        DO ROW = 1, ROWS
            DO COL = 1, COLS
                VOL = VOL + ((GRID(ROW, COL) - SMIN1) * GRID_SIZE * &
                      GRID_SIZE)
            END DO
        END DO
        WRITE (MASS_FILE, 340, ADVANCE = 'NO') TIME
        IF (RUN_MODE == 1) THEN
            WRITE (MASS_FILE, 340, ADVANCE = 'NO') SAND_IN
            WRITE (MASS_FILE, 340, ADVANCE = 'NO') SAND_OUT
            WRITE (MASS_FILE, 340, ADVANCE = 'NO') MUD_IN
            WRITE (MASS_FILE, 340, ADVANCE = 'NO') MUD_OUT
            DO ROW = 1, ROWS
                DO COL = 1, COLS

                    ALYR_BASE = GRID(ROW, COL) - ALYR_TH
                    R_LYRS = (ALYR_BASE - SMIN1) / LYR_TH
                    N_LYRS = INT(R_LYRS)

                    DO N = 1, N_LYRS
                        SAND_TOT = SAND_TOT + ((1 - SED(ROW, COL, N, 1)) &
                                   * LYR_TH * GRID_SIZE * GRID_SIZE)
                        MUD_TOT = MUD_TOT + (SED(ROW, COL, N, 1) * &
                                  LYR_TH * GRID_SIZE * GRID_SIZE)
                    END DO

                    SAND_TOT = SAND_TOT + &
                        ((1 - SED(ROW, COL, N_LYRS + 1, 1)) * LYR_TH * &
                        (R_LYRS - N_LYRS) * GRID_SIZE * GRID_SIZE)
                    MUD_TOT = MUD_TOT + (SED(ROW, COL, N_LYRS + 1, 1) * &
                      LYR_TH * (R_LYRS - N_LYRS) * GRID_SIZE * GRID_SIZE)
```

```
                    SAND_TOT = SAND_TOT + (ALYR(ROW, COL, 1) * &
                               GRID_SIZE * GRID_SIZE)
                    MUD_TOT = MUD_TOT + (ALYR(ROW, COL, 2) * GRID_SIZE * &
                              GRID_SIZE)

                END DO
            END DO

            WRITE (MASS_FILE, 340, ADVANCE = 'NO') SAND_TOT
            WRITE (MASS_FILE, 340, ADVANCE = 'NO') MUD_TOT

        ELSE
            IF (RUN_MODE == 2) THEN

                DO N = 1, NR_FRACTIONS + 1
                    WRITE (MASS_FILE, 340, ADVANCE = 'NO') MF_SED_IN(N)
                    WRITE (MASS_FILE, 340, ADVANCE = 'NO') MF_SED_OUT(N)
                END DO

                MF_SED_TOT = 0
                DO ROW = 1, ROWS
                    DO COL = 1, COLS

                        ALYR_BASE = GRID(ROW, COL) - ALYR_TH
                        R_LYRS = (ALYR_BASE - SMIN1) / LYR_TH
                        N_LYRS = INT(R_LYRS)

                        DO N = 1, NR_FRACTIONS + 1
                            DO M = 1, N_LYRS
                                MF_SED_TOT(N) = MF_SED_TOT(N) + &
                                (SED(ROW, COL, M, N) * LYR_TH * &
                                GRID_SIZE * GRID_SIZE)
                            END DO
                            MF_SED_TOT(N) = MF_SED_TOT(N) + &
                                (SED(ROW, COL, N_LYRS + 1, N) * &
                                LYR_TH * (R_LYRS - N_LYRS) * GRID_SIZE * &
                                GRID_SIZE)
                            MF_SED_TOT(N) = MF_SED_TOT(N) + &
                               (ALYR(ROW, COL, N) * GRID_SIZE * GRID_SIZE)
                        END DO

                    END DO
                END DO

                DO N = 1, NR_FRACTIONS + 1
                    WRITE (MASS_FILE, 340, ADVANCE = 'NO') MF_SED_TOT(N)
                END DO

            END IF
        END IF
        WRITE (MASS_FILE, 340) VOL
        MASSTM = MASSTM + MASSINT

    END IF


    !Output active layer composition to file
    IF (TIME >= SED_RESULTS_TIME) THEN
        WRITE (CHR, 120) INT(TIME)
        CHR = ADJUSTL(CHR)
        RESULTS_FN = RESULTS_DIR(1:RESULTS_DIR_LEN) // "\SED_" // &
                     TRIM(CHR) // ".ASC"
        CALL SED_OUT(ROWS, COLS, MF, FI, FMIN, FMAX, GRID_SIZE, ALYR, &
                     ALYR_TH, NR_FRACTIONS, 0.5, RESULTS_FN)
```

```
        SED_RESULTS_TIME = SED_RESULTS_TIME + SED_RESULTS_INT
    END IF



    IF (V_RESULTS) THEN

        !Output velocity results to file

        IF (TIME >= V_RESULTS_TIME - 1E-5) THEN
            VRC1 = 1
            VRC2 = V_RESULTS_INT2
            V_RESULTS_TIME = V_RESULTS_TIME + V_RESULTS_INT1
        END IF

        IF (VRC1 <= V_RESULTS_NR) THEN
            IF (VRC2 == V_RESULTS_INT2) THEN

                WRITE (CHR1, 120) INT(TIME)
                CHR1 = ADJUSTL(CHR1)
                WRITE (CHR2, 120) VRC1
                CHR2 = ADJUSTL(CHR2)
                RESULTS_FN = RESULTS_DIR(1:RESULTS_DIR_LEN) // "\VX_" // &
                             TRIM(CHR1) // "_" // TRIM(CHR2) // ".ASC"

                CALL GRID_OUT(ROWS, COLS, GRID_SIZE, VX, RESULTS_FN)

                RESULTS_FN = RESULTS_DIR(1:RESULTS_DIR_LEN) // "\VY_" // &
                             TRIM(CHR1) // "_" // TRIM(CHR2) // ".ASC"

                CALL GRID_OUT(ROWS, COLS, GRID_SIZE, VY, RESULTS_FN)

                VRC1 = VRC1 + 1
                VRC2 = 1

            ELSE
                VRC2 = VRC2 + 1
            END IF
        END IF

    END IF

    !Increment the time-step
    IF (T2 == 1) THEN
        T1 = 1
        T2 = 2
    ELSE
        IF (T2 < TSPRP) THEN
            T1 = T1 + 1
            T2 = T2 + 1
        ELSE
            T1 = TSPRP
            T2 = 1
        END IF
    END IF

    TIME = TIME + TIMESTEP

    TS_NR = TS_NR + 1

    PRINT 110, TIME

END DO
```

```
!--------------------MAIN PROGRAM LOOP END-------------------------------


CLOSE (MASS_FILE)
CLOSE (LOG_FILE)

PRINT *,"Run Completed"
READ *, STR


10 FORMAT (A)
20 FORMAT (F8.0)
30 FORMAT (I11)
60 FORMAT (F15.3)
70 FORMAT (I10)
110 FORMAT ("Time= ", F15.3)
120 FORMAT (I8)
150 FORMAT (I5)
160 FORMAT (I3)
170 FORMAT (F6.0)
180 FORMAT (I2)
190 FORMAT (F6.0)
200 FORMAT (F4.0)
210 FORMAT (F11.0)
220 FORMAT (I1)
260 FORMAT (F11.2)
270 FORMAT (I9)
280 FORMAT (F6.2)
290 FORMAT (F6.4)
300 FORMAT (F14.6)
310 FORMAT (F7.0)
320 FORMAT (F5.0)
330 FORMAT (F9.0)
340 FORMAT (F20.6)
!*********************************************************************
!                  END OF MAIN PROGRAM
!*********************************************************************

CONTAINS

!Results output subroutines

SUBROUTINE SED_OUT(ROWS, COLS, MF, FI, FMIN, FMAX, GRID_SIZE, SED, TH, &
                   NFR, FR, FILENAME)

REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:,:), INTENT(IN) :: SED
REAL, DIMENSION(:), INTENT(IN) :: FI
REAL, DIMENSION(:), INTENT(IN) :: FMIN
REAL, DIMENSION(:), INTENT(IN) :: FMAX

REAL, INTENT(IN) :: GRID_SIZE
REAL, INTENT(IN) :: FR

REAL (SELECTED_REAL_KIND(15,307)), INTENT(IN) :: TH

INTEGER, INTENT(IN) :: ROWS
INTEGER, INTENT(IN) :: COLS
INTEGER, INTENT(IN) :: NFR

LOGICAL, INTENT(IN) :: MF

REAL, DIMENSION(:), ALLOCATABLE :: PI

REAL D50
```

```
INTEGER ROW, COL

CHARACTER (LEN = 200) FILENAME       !Filename for output file

IF (MF) THEN
    ALLOCATE (PI(NFR + 1))
END IF

!Open output file
OPEN (1, FILE = FILENAME)

!Write header information
WRITE (1, 10) "ncols", COLS
WRITE (1, 10) "nrows", ROWS
WRITE (1, 10) "xllcorner", 0
WRITE (1, 10) "yllcorner", 0
WRITE (1, 30) "cellsize", GRID_SIZE
WRITE (1, 10) "nodata_value", -9999

!Write grid data to file
DO     ROW = ROWS, 1, -1
       DO     COL = 1, COLS - 1
           IF (MF) THEN
               DO N = 1, NFR + 1
                   PI(N) = SED(ROW, COL, N) / TH
               END DO
               D50 = GETSIZE1(FI, FMIN, FMAX, PI, NFR, FR) * 1E6
                  WRITE (1, 20, ADVANCE = 'NO') D50
              ELSE
                  WRITE (1, 20, ADVANCE = 'NO') SED(ROW, COL, 2)
              END IF
       END DO
    !Write last column and advance to next line
    IF (MF) THEN
        DO N = 1, NFR
            PI(N) = SED(ROW, COLS, N) / TH
        END DO
        D50 = GETSIZE1(FI, FMIN, FMAX, PI, NFR, FR) * 1E6
           WRITE (1, 20) D50
       ELSE
           WRITE (1, 20) SED(ROW, COLS, 2)
       END IF

END DO

CLOSE (1)

10     FORMAT (A, T15, I9.1)
20     FORMAT (F12.3)
30     FORMAT (A, T15, F9.3)

END SUBROUTINE

SUBROUTINE GRID_OUT(ROWS, COLS, GRID_SIZE, GRID, FILENAME)

REAL,  DIMENSION(:,:),            INTENT(IN)    :: GRID

REAL, INTENT(IN) :: GRID_SIZE

INTEGER, INTENT(IN) :: ROWS
INTEGER, INTENT(IN) :: COLS

CHARACTER (LEN = 200), INTENT (IN) :: FILENAME !Filename for output file

INTEGER ROW, COL
```

```
!Open output file
OPEN (1, FILE = FILENAME)

!Write header information
WRITE (1, 10) "ncols", COLS
WRITE (1, 10) "nrows", ROWS
WRITE (1, 10) "xllcorner", 0
WRITE (1, 10) "yllcorner", 0
WRITE (1, 30) "cellsize", GRID_SIZE
WRITE (1, 10) "nodata_value", -9999

!Write grid data to file
DO     ROW = ROWS, 1, -1

       DO     COL = 1, COLS - 1
              WRITE (1, 20, ADVANCE = 'NO') GRID (ROW, COL)
       END DO
    !Write last column and advance to next line
    WRITE (1, 20) GRID (ROW, COLS)

END DO

CLOSE (1)

10     FORMAT (A, T15, I9.1)
20     FORMAT (F12.3)
30     FORMAT (A, T15, F9.3)

END SUBROUTINE

SUBROUTINE GRID_OUTL(ROWS, COLS, GRID_SIZE, GRID, FILENAME)

REAL (SELECTED_REAL_KIND(15,307)),      DIMENSION(:,:),     INTENT(IN) :: GRID

REAL, INTENT(IN) :: GRID_SIZE

INTEGER, INTENT(IN) :: ROWS
INTEGER, INTENT(IN) :: COLS
CHARACTER (LEN = 200), INTENT (IN) :: FILENAME !Filename for output file

INTEGER ROW, COL

!Open output file
OPEN (1, FILE = FILENAME)

!Write header information
WRITE (1, 10) "ncols", COLS
WRITE (1, 10) "nrows", ROWS
WRITE (1, 10) "xllcorner", 0
WRITE (1, 10) "yllcorner", 0
WRITE (1, 30) "cellsize", GRID_SIZE
WRITE (1, 10) "nodata_value", -9999

!Write grid data to file
DO     ROW = ROWS, 1, -1

       DO     COL = 1, COLS - 1
              WRITE (1, 20, ADVANCE = 'NO') GRID (ROW, COL)
       END DO
    !Write last column and advance to next line
    WRITE (1, 20) GRID (ROW, COLS)

END DO
```

```
CLOSE (1)

10     FORMAT (A, T15, I9.1)
20     FORMAT (F12.3)
30     FORMAT (A, T15, F9.3)

END SUBROUTINE

!Function to calculate sediment size
!statistics based on fraction proportions
FUNCTION GETSIZE1(FI, FMIN, FMAX, PI, NFR, FR)

REAL, DIMENSION(:), INTENT(IN) :: PI
REAL, DIMENSION(:), INTENT(IN) :: FI
REAL, DIMENSION(:), INTENT(IN) :: FMIN
REAL, DIMENSION(:), INTENT(IN) :: FMAX

REAL, INTENT(IN) :: FR

INTEGER, INTENT(IN) :: NFR

REAL, DIMENSION(:), ALLOCATABLE :: PCUM

REAL GETSIZE1, FRA, FR1, D50

INTEGER N

ALLOCATE (PCUM(NFR + 1))

!Median size for sand & silt particles
D50 = GETSIZE(FI, FMIN, FMAX, PI, NFR + 1, 0.5)

!Weighted geometric mean of sand and mud particle sizes
GETSIZE1 = (D50 ** (1 - PI(1))) * (FI(1) ** PI(1))

END FUNCTION


END PROGRAM
```

Module DATA_INOUT

Subroutines	Purpose	Page nr.
REAL_INPUT	Input ASCII data to REAL	**285**
DOUBLE_INPUT	Input ASCII data to double precision REAL	**286**
INTEGER_INPUT	Input ASCII data to INTEGER	**287**

```
MODULE DATA_INOUT

IMPLICIT NONE

INTERFACE DATA_INPUT
    MODULE PROCEDURE REAL_INPUT, DOUBLE_INPUT, INTEGER_INPUT
END INTERFACE

CONTAINS

SUBROUTINE REAL_INPUT(DAT, F, ROWS, COLS, GS, FN)

REAL, DIMENSION(:,:), ALLOCATABLE, INTENT(OUT) :: DAT

REAL, INTENT(OUT) :: GS

INTEGER, INTENT(IN)      :: F
INTEGER, INTENT(OUT)     :: ROWS
INTEGER, INTENT(OUT)     :: COLS

CHARACTER (200), INTENT (IN)            :: FN

INTEGER ROW, COL, IO

CHARACTER (LEN = 23) CHR
CHARACTER STR

!Open ASCII grid file and read number of rows/columns
OPEN (F, FILE = FN, STATUS = "OLD", IOSTAT = IO)
IF (IO /= 0) THEN
    PRINT*, "Could not open file"
    PRINT*, FN
    READ (*,10) STR
    STOP
END IF

READ (F, 40, ADVANCE = 'NO') CHR !Discard 1st 12 characters
READ (F, 30) COLS                        !Read number of columns
READ (F, 40, ADVANCE = 'NO') CHR !Discard 1st 12 characters of next line
READ (F, 30) ROWS                        !Read number of rows
READ (F, 10) CHR                        !Discard x origin coordinate as not
needed
READ (F, 10) CHR                        !Discard y origin coordinate as not
needed
READ (F, 40, ADVANCE = 'NO') CHR !Discard 1st 12 characters of next line
READ (F, 50) GS                          !Read size of one grid square
READ (F, 10) CHR                  !No data value (discarded in current version)

ALLOCATE (DAT(ROWS, COLS))

!Read data from ASCII file
DO ROW = ROWS, 1, -1
```

```
  DO COL = 1, COLS - 1
    READ (F, 20, ADVANCE = 'NO') DAT(ROW, COL)
  END DO
  READ (F, 20) DAT(ROW, COLS)
END DO

CLOSE (F)


10 FORMAT (A)
20 FORMAT (F8.0)
30 FORMAT (I11)
40 FORMAT (A12)
50 FORMAT (F20.0)


END SUBROUTINE


SUBROUTINE DOUBLE_INPUT(DAT, F, ROWS, COLS, GS, FN)

REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:), &
    ALLOCATABLE, INTENT(OUT) :: DAT

REAL, INTENT(OUT) :: GS

INTEGER, INTENT(IN)     :: F
INTEGER, INTENT(OUT)    :: ROWS
INTEGER, INTENT(OUT)    :: COLS

CHARACTER (200), INTENT (IN)            :: FN

INTEGER ROW, COL

CHARACTER (LEN = 23) CHR

!Open ASCII grid file and read number of rows/columns
OPEN (F, FILE = FN)
READ (F, 40, ADVANCE = 'NO') CHR !Discard 1st 12 characters
READ (F, 30) COLS                        !Read number of columns
READ (F, 40, ADVANCE = 'NO') CHR !Discard 1st 12 characters of next line
READ (F, 30) ROWS                        !Read number of rows
READ (F, 10) CHR                        !Discard x origin coordinate as not
needed
READ (F, 10) CHR                        !Discard y origin coordinate as not
needed
READ (F, 40, ADVANCE = 'NO') CHR !Discard 1st 12 characters of next line
READ (F, 50) GS                          !Read size of one grid square
READ (F, 10) CHR                  !No data value (discarded in current version)

ALLOCATE (DAT(ROWS, COLS))

!Read data from ASCII file
DO ROW = ROWS, 1, -1
  DO COL = 1, COLS - 1
    READ (F, 20, ADVANCE = 'NO') DAT(ROW, COL)
  END DO
  READ (F, 20) DAT(ROW, COLS)
END DO

CLOSE (F)


10 FORMAT (A)
20 FORMAT (F8.0)
```

```
30 FORMAT (I11)
40 FORMAT (A12)
50 FORMAT (F20.0)


END SUBROUTINE


SUBROUTINE INTEGER_INPUT(DAT, F, ROWS, COLS, GS, FN)

INTEGER, DIMENSION(:,:), ALLOCATABLE, INTENT(OUT) :: DAT

REAL, INTENT(OUT) :: GS

INTEGER, INTENT(IN)     :: F
INTEGER, INTENT(OUT)    :: ROWS
INTEGER, INTENT(OUT)    :: COLS

REAL DAT_IN

CHARACTER (200), INTENT (IN)            :: FN

INTEGER ROW, COL

CHARACTER (LEN = 23) CHR

!Open ASCII grid file and read number of rows/columns
OPEN (F, FILE = FN)
READ (F, 40, ADVANCE = 'NO') CHR !Discard 1st 12 characters
READ (F, 30) COLS                        !Read number of columns
READ (F, 40, ADVANCE = 'NO') CHR !Discard 1st 12 characters of next line
READ (F, 30) ROWS                        !Read number of rows
READ (F, 10) CHR                        !Discard x origin coordinate as not
needed
READ (F, 10) CHR                        !Discard y origin coordinate as not
needed
READ (F, 40, ADVANCE = 'NO') CHR !Discard 1st 12 characters of next line
READ (F, 50) GS                          !Read size of one grid square
READ (F, 10) CHR                  !No data value (discarded in current version)

ALLOCATE (DAT(ROWS, COLS))

!Read data from ASCII file
DO ROW = ROWS, 1, -1
  DO COL = 1, COLS - 1
    READ (F, 20, ADVANCE = 'NO') DAT_IN
    DAT(ROW, COL) = INT(DAT_IN)
  END DO
  READ (F, 20) DAT_IN
  DAT(ROW, COLS) = INT(DAT_IN)
END DO

CLOSE (F)


10 FORMAT (A)
20 FORMAT (F8.0)
30 FORMAT (I11)
40 FORMAT (A12)
50 FORMAT (F20.0)


END SUBROUTINE

END MODULE
```

Module FLOW1


```
MODULE FLOW1

IMPLICIT NONE

REAL, PARAMETER :: SQR2 = 1.414213562

CONTAINS

SUBROUTINE CALCFLOW(GRID, GS, WL1, WL2, QX, QY, H, MIN_H, H1, C, TSS, &
                    ROWS, COLS, QI, QI_N, QI_R, QI_C, S_MIN, S_MAX, XP, &
                    MAXITR, LOG_FILE, TIME)

!Bed levels
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:), INTENT (IN) :: GRID

!Array to store water level data for start of timestep
REAL,  DIMENSION(:,:),            INTENT (IN)         :: WL1
!Array to store water level data for end of timestep
REAL,  DIMENSION(:,:),            INTENT (IN)         :: WL2
!Transverse flow into cell on right
REAL,  DIMENSION(:,:),     INTENT (INOUT)      :: QX
!Axial flow entering cells from upstream row
REAL,  DIMENSION(:,:),     INTENT (INOUT)      :: QY
!Water depth for each cell
REAL,  DIMENSION(:,:),         INTENT (OUT)    :: H
!Water depth for each cell from slope based routing
REAL,  DIMENSION(:,:),         INTENT (OUT)    :: H1
!Chezy roughness coefficient for each cell
REAL,   DIMENSION(:,:),           INTENT (IN)         :: C
!Upstream inflow cells in row 1(constant flow in m3/s for each cell)
REAL,   DIMENSION(:),       INTENT (IN)     :: QI

INTEGER, DIMENSION(:), INTENT(IN) :: QI_R   !Row numbers for river inflows
INTEGER, DIMENSION(:), INTENT(IN) :: QI_C   !Col numbers for river inflows

REAL, INTENT (IN)          :: TSS       !Timestep in seconds
REAL, INTENT (IN)          :: GS        !Grid cell size
REAL, INTENT (IN)       :: S_MIN        !Min slope value used to calculate
```

```
                                        !depth in slope based routing
REAL, INTENT (IN)       :: S_MAX        !Max slope value used to calculate
                                        !depth in slope based routing
REAL, INTENT (IN)       :: XP           !Slope routing model parameter
REAL, INTENT (IN)       :: TIME         !Simulation time (hours)
REAL, INTENT (IN)       :: MIN_H        !Minimum water depth

INTEGER, INTENT (IN)       :: ROWS         !Number of rows in CA domain
INTEGER, INTENT (IN)    :: COLS         !Number of columns in CA domain
INTEGER, INTENT (IN)    :: QI_N         !Number of river inflow cells
INTEGER, INTENT (IN)    :: MAXITR       !Maximum nr. of iterations
INTEGER, INTENT (IN)    :: LOG_FILE     !Log file number

!Change in flow due to change in storage or inflow
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:), ALLOCATABLE :: DQ

!Adjusted bed levels, with 'holes' filled to ensure continuous
!downslope from any cell to the model boundary
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:), &
    ALLOCATABLE, SAVE :: AGRID

REAL, DIMENSION(:,:), ALLOCATABLE :: K      !Bed friction paramater

!0 = Non-tidal, 1 - 4 = tidal with continuity corrections routed to:
!1=row+1, 2=col+1, 3=row-1 & 4=col-1
INTEGER, DIMENSION(:,:),    ALLOCATABLE :: TD
!Row coordinate for cells in continuity search (by search level)
INTEGER, DIMENSION(:,:),    ALLOCATABLE :: RL
!Column coordinate for cells in continuity search (by search level)
INTEGER, DIMENSION(:,:),    ALLOCATABLE :: CL
!Row coordinate for cells in continuity search (sequential)
INTEGER, DIMENSION(:),      ALLOCATABLE :: RN
!Column coordinate for cells in continuity search (sequential)
INTEGER, DIMENSION(:),      ALLOCATABLE :: CN

REAL (SELECTED_REAL_KIND(15,307)) QTOL, QCORR, QCORR1, QCORR2, QCORR3
REAL (SELECTED_REAL_KIND(15,307)) QCORR4, MAXQCORR

REAL WLMIN, WLMAX, WLMEAN, KX, KY
REAL K1, K2, K3, K4
REAL HLQ1, HLQ2, HLQ3, HLQ4
REAL HL1, HL2, HL3, HL4
REAL HLQT, HLT
REAL QCHECKMIN, VTOL
REAL SWL, WL_1, WL_2

INTEGER ROW, COL, N, N1, N2, L, M, M1
INTEGER RW, RW1, RW2, RW3
INTEGER TD1
INTEGER ITT

LOGICAL, SAVE :: FIRST_EXEC = .TRUE.

integer aa
character str

IF (FIRST_EXEC == .TRUE.) THEN
    FIRST_EXEC = .FALSE.
    ALLOCATE(AGRID(ROWS, COLS))
END IF

VTOL = 0.01

N = 3 * (ROWS + COLS)
```

```
ALLOCATE (DQ(ROWS, COLS))
ALLOCATE (K(ROWS, COLS))
ALLOCATE (TD(ROWS, COLS))
ALLOCATE (RL(N, N))
ALLOCATE (CL(N, N))
ALLOCATE (RN(ROWS * COLS))
ALLOCATE (CN(ROWS * COLS))


!FILL 'HOLES' (DEPRESSIONS)
CALL FILL_HOLES(GRID, AGRID, ROWS, COLS)

!ESTIMATE WATER DEPTH DUE TO FLUVIAL GRAVITY FLOW
CALL S_ROUT(GRID, AGRID, C, ROWS, COLS, GS, QI, QI_N, &
            QI_R, QI_C, H1, MIN_H, S_MIN, S_MAX)

!Calculate water depths
DO ROW = 1, ROWS

    DO COL = 1, COLS

        SWL = GRID(ROW, COL) + H1(ROW, COL)

        WL_1 = MAX(WL1(ROW, COL), SWL)
        WL_2 = MAX(WL2(ROW, COL), SWL)

        WLMIN = MIN(WL_1, WL_2)
        WLMAX = MAX(WL_1, WL_2)

        IF (WLMAX - GRID(ROW, COL) < MIN_H + 1E-6) THEN
            H(ROW, COL) = 0
            DQ(ROW, COL) = 0
        ELSE
            IF (WLMIN > GRID(ROW, COL)) THEN
                WLMEAN = (WL_1 + WL_2) / 2.0
                DQ(ROW, COL) = (WL_1 - WL_2) * (GS ** 2) / TSS
            ELSE
                IF (WL_1 > GRID(ROW, COL)) THEN
                    WLMEAN = (WL_1 + GRID(ROW, COL)) / 2.0
                    DQ(ROW, COL) = (WL_1 - GRID(ROW, COL)) * &
                        (GS ** 2) / TSS
                ELSE
                    WLMEAN = (WL_2 + GRID(ROW, COL)) / 2.0
                    DQ(ROW, COL) = (GRID(ROW, COL) - WL_2) * &
                        (GS ** 2) / TSS
                END IF
            END IF
            IF (WLMEAN - GRID(ROW, COL) < MIN_H + 1E-6) THEN
                H(ROW, COL) = 0
                DQ(ROW, COL) = 0
            ELSE
                H(ROW, COL) = WLMEAN - GRID(ROW, COL)
            END IF
        END IF
    END DO
END DO


!ADD RIVER FLOW
DO N = 1, QI_N
    DQ(QI_R(N), QI_C(N)) = DQ(QI_R(N), QI_C(N)) + QI(N)
END DO

!CALCULATE FRICTION PARAMETERS
DO ROW = 1, ROWS
```

```
    DO COL = 1, COLS
        IF (H(ROW, COL) > 0.001) THEN
            K(ROW, COL) = 1 / ((C(ROW, COL) ** 2) * &
                (H(ROW, COL) ** 3) * GS)
        ELSE
            K(ROW, COL) = 0
        END IF
    END DO
END DO

!IDENTIFY ALL CELLS WITH HYDRAULIC CONNECTION TO THE BOUNDARY
!AND IDENTIFY CELL ORDER AND FLOW DIRECTIONS FOR CONTINUITY CALCULATION
TD = 0
N = 0
M = 0
DO COL = 1, COLS
    IF (H(ROWS, COL) > MIN_H) THEN
        TD(ROWS, COL) = 1
        N = N + 1
        M = M + 1
        RL(1, N) = ROWS
        CL(1, N) = COL
        RN(M) = ROWS
        CN(M) = COL
    END IF
END DO

L = 1   !L is the search level
!RN/CN contain lists rows/cols of connected cells starting in order of
!distance from boundary RL/CL contain list of the same cells but
!organised by search level TD records which neighbouring cell is
!hydraulically connected and in a lower level
!(i.e. closer to the boundary)
DO WHILE (N > 0)
    N1 = N
    N = 0   !N = number of cells hydrauically connected to previous level
    L = L + 1
    !Check each hydraulically connect cell found in previous level for
    !new connected cells
    DO N2 = 1, N1
        IF (RL(L - 1, N2) > 1) THEN
            !If row > 1 check cell for new connected cell in row - 1
            IF ((H(RL(L - 1, N2) - 1, CL(L - 1, N2)) > MIN_H) .AND. &
                TD(RL(L - 1, N2) - 1, CL(L - 1, N2)) == 0) THEN
                TD(RL(L - 1, N2) - 1, CL(L - 1, N2)) = 1
                N = N + 1
                M = M + 1
                RL(L, N) = RL(L - 1, N2) - 1
                CL(L, N) = CL(L - 1, N2)
                RN(M) = RL(L, N)
                CN(M) = CL(L, N)
            END IF
        END IF
        IF (CL(L - 1, N2) > 1) THEN
            !If col > 1 check cell for new connected cell in col - 1
            IF ((H(RL(L - 1, N2), CL(L - 1, N2) - 1) > MIN_H) .AND. &
                TD(RL(L - 1, N2), CL(L - 1, N2) - 1) == 0) THEN
                TD(RL(L - 1, N2), CL(L - 1, N2) - 1) = 2
                N = N + 1
                M = M + 1
                RL(L, N) = RL(L - 1, N2)
                CL(L, N) = CL(L - 1, N2) - 1
                RN(M) = RL(L, N)
                CN(M) = CL(L, N)
            END IF
```

```
        END IF
        IF (RL(L - 1, N2) < ROWS) THEN
            !If row > rows check cell for new connected cell in row + 1
            IF ((H(RL(L - 1, N2) + 1, CL(L - 1, N2)) > MIN_H) .AND. &
                TD(RL(L - 1, N2) + 1, CL(L - 1, N2)) == 0) THEN
                TD(RL(L - 1, N2) + 1, CL(L - 1, N2)) = 3
                N = N + 1
                M = M + 1
                RL(L, N) = RL(L - 1, N2) + 1
                CL(L, N) = CL(L - 1, N2)
                RN(M) = RL(L, N)
                CN(M) = CL(L, N)
            END IF
        END IF
        IF (CL(L - 1, N2) < COLS) THEN
            !If col < cols check cell for new connected cell in col + 1
            IF ((H(RL(L - 1, N2), CL(L - 1, N2) + 1) > MIN_H) .AND. &
                TD(RL(L - 1, N2), CL(L - 1, N2) + 1) == 0) THEN
                TD(RL(L - 1, N2), CL(L - 1, N2) + 1) = 4
                N = N + 1
                M = M + 1
                RL(L, N) = RL(L - 1, N2)
                CL(L, N) = CL(L - 1, N2) + 1
                RN(M) = RL(L, N)
                CN(M) = CL(L, N)
            END IF
        END IF
    END DO
END DO

!CONTINUITY
!MAKE FLOW CORRECTIONS TO ENSURE MASS CONTINUITY BETWEEN ACTIVE CELLS
!Starting with furthest cell from the boundary
DO M1 = M, 1, -1

    ROW = RN(M1)
    COL = CN(M1)
    !Correction is determined by summing all inflows, outflows and
    !change in storage using values from previous calculations
    !(set to zero at first iteration)
    QCORR = DQ(ROW, COL)
    IF (ROW > 1) THEN
        IF (TD(ROW - 1, COL) > 0) THEN
            QCORR = QCORR + QY(ROW - 1, COL)
        END IF
    END IF
    IF (COL > 1) THEN
        IF (TD(ROW, COL - 1) > 0) THEN
            QCORR = QCORR + QX(ROW, COL - 1)
        END IF
    END IF
    IF (ROW < ROWS) THEN
        IF (TD(ROW + 1, COL) > 0) THEN
            QCORR = QCORR - QY(ROW, COL)
        ELSE
            QY(ROW, COL) = 0
        END IF
    ELSE
        QCORR = QCORR - QY(ROW, COL)
    END IF
    IF (COL < COLS) THEN
        IF (TD(ROW, COL + 1) > 0) THEN
            QCORR = QCORR - QX(ROW, COL)
        ELSE
            QX(ROW, COL) = 0
```

```
        END IF
    ELSE
        QX(ROW, COL) = 0
    END IF

    !Apply flow correction
    TD1 = TD(ROW, COL)
    IF (TD1 == 1) THEN
        QY(ROW, COL) = QY(ROW, COL) + QCORR
    ELSE
        IF (TD1 == 2) THEN
            QX(ROW, COL) = QX(ROW, COL) + QCORR
        ELSE
            IF (TD1 == 3) THEN
                QY(ROW - 1, COL) = QY(ROW - 1, COL) - QCORR
            ELSE
                QX(ROW, COL - 1) = QX(ROW, COL - 1) - QCORR
            END IF
        END IF
    END IF
END DO

QTOL = 0.01 !Calc. ends when max correction is within 1% of previous value
ITT = 1
MAXQCORR = QTOL + 1.0

!Head balance calculation
DO WHILE ((MAXQCORR > QTOL) .AND. (ITT < MAXITR))

    ITT = ITT + 1

    MAXQCORR = 0
    DO ROW = 1, ROWS
        DO COL = 1, COLS
            IF (TD(ROW, COL) > 0) THEN
                IF (ROW < ROWS) THEN
                    IF (COL < COLS) THEN
                        IF ((TD(ROW + 1, COL) > 0) .AND. &
                            (TD(ROW + 1, COL + 1) > 0) .AND. &
                            (TD(ROW, COL + 1)) > 0) THEN

                            !HEAD BALANCE ON 'LOCAL LOOP'
                            K1 = (K(ROW, COL) + &
                                K(ROW + 1, COL)) / 2.0
                            K2 = (K(ROW + 1, COL) + &
                                K(ROW + 1, COL + 1)) / 2.0
                            K3 = (K(ROW + 1, COL + 1) + &
                                K(ROW, COL + 1)) / 2.0
                            K4 = (K(ROW, COL + 1) + &
                                K(ROW, COL)) / 2.0

                            HLQ1 = K1 * ABS(QY(ROW, COL))
                            HLQ2 = K2 * ABS(QX(ROW + 1, COL))
                            HLQ3 = K3 * ABS(QY(ROW, COL + 1))
                            HLQ4 = K4 * ABS(QX(ROW, COL))
                            HL1 = HLQ1 * QY(ROW, COL)
                            HL2 = HLQ2 * QX(ROW + 1, COL)
                            HL3 = HLQ3 * QY(ROW, COL + 1)
                            HL4 = HLQ4 * QX(ROW, COL)

                            HLQT = HLQ1 + HLQ2 + HLQ3 + HLQ4
                            HLT = HL1 + HL2 - HL3 - HL4

                            IF (HLQT > 1E-15) THEN
                                QCORR = -HLT / (2 * HLQT)
```

```
                ELSE
                    QCORR = 0
                END IF

                QCHECKMIN = VTOL * GS * &
                    MIN(H(ROW, COL), H(ROW + 1, COL))
                IF (ABS(QY(ROW, COL)) > QCHECKMIN) THEN
                    QCORR1 = ABS(QCORR / QY(ROW, COL))
                ELSE
                    QCORR1 = ABS(QCORR / QCHECKMIN)
                END IF
                QCHECKMIN = VTOL * GS * &
                    MIN(H(ROW, COL + 1), H(ROW + 1, COL + 1))
                IF (ABS(QY(ROW, COL + 1)) > QCHECKMIN) THEN
                    QCORR2 = ABS(QCORR / QY(ROW, COL + 1))
                ELSE
                    QCORR2 = ABS(QCORR / QCHECKMIN)
                END IF
                QCHECKMIN = VTOL * GS * &
                    MIN(H(ROW, COL), H(ROW, COL + 1))
                IF (ABS(QX(ROW, COL)) > QCHECKMIN) THEN
                    QCORR3 = ABS(QCORR / QX(ROW, COL))
                ELSE
                    QCORR3 = ABS(QCORR / QCHECKMIN)
                END IF
                QCHECKMIN = VTOL * GS * &
                    MIN(H(ROW + 1, COL), H(ROW + 1, COL + 1))
                IF (ABS(QX(ROW + 1, COL)) > QCHECKMIN) THEN
                    QCORR4 = ABS(QCORR / QX(ROW + 1, COL))
                ELSE
                    QCORR4 = ABS(QCORR / QCHECKMIN)
                END IF
                IF (MAX(QCORR1, QCORR2, QCORR3, QCORR4) > &
                    MAXQCORR) MAXQCORR = &
                    MAX(QCORR1, QCORR2, QCORR3, QCORR4)

                IF (MAXQCORR > 0.1) THEN
                    AA = 1
                END IF

                QY(ROW, COL) = QY(ROW, COL) + QCORR
                QX(ROW + 1, COL) = QX(ROW + 1, COL) + QCORR
                QY(ROW, COL + 1) = QY(ROW, COL + 1) - QCORR
                QX(ROW, COL) = QX(ROW, COL) - QCORR

            ELSE
                !If one or more cells in the local group is
                !not submerged then: IDENTIFY ALTERNATIVE
                !LOOP TO THE RIGHT IF POSSIBLE
                !Only do this step if cells at (row + 1, col)
                !and (row + 1, col + 1) are 'active cells'
                !(i.e. submerged)
                !(This condition must occur at least once for
                !any given loop)
                IF ((TD(ROW + 1, COL) > 0) .AND. &
                    (TD(ROW + 1, COL + 1) > 0)) THEN

                    CALL LONGLOOP &
                        (ROW, COL, ROWS, COLS, QX, QY, K, TD)

                END IF
            END IF

        END IF !(IF COL < COLS)
    ELSE
```

```
                    !IF (ROW == ROWS) - Cells on tidal boundary
                    IF (COL < COLS) THEN
                        IF (TD(ROW, COL + 1) > 0) THEN

                            !HEAD BALANCE ON 'LOCAL LOOP' AT BOUNDARY
                            K1 = K(ROW, COL)
                            K3 = K(ROW, COL + 1)
                            K4 = RMS(K(ROW, COL + 1), K(ROW, COL))
                            HLQ1 = K1 * ABS(QY(ROW, COL))
                            HLQ3 = K3 * ABS(QY(ROW, COL + 1))
                            HLQ4 = K4 * ABS(QX(ROW, COL))
                            HL1 = HLQ1 * QY(ROW, COL)
                            HL3 = HLQ3 * QY(ROW, COL + 1)
                            HL4 = HLQ4 * QX(ROW, COL)

                            HLQT = HLQ1 + HLQ3 + HLQ4
                            HLT = HL1 - HL3 - HL4

                            IF (HLQT > 1E-15) THEN
                                QCORR = -HLT / (2 * HLQT)
                            ELSE
                                QCORR = 0
                            END IF

                            QCHECKMIN = VTOL * GS * H(ROW, COL)
                            IF (ABS(QY(ROW, COL)) > QCHECKMIN) THEN
                                QCORR1 = ABS(QCORR / QY(ROW, COL))
                            ELSE
                                QCORR1 = ABS(QCORR / QCHECKMIN)
                            END IF
                            QCHECKMIN = VTOL * GS * H(ROW, COL + 1)
                            IF (ABS(QY(ROW, COL + 1)) > QCHECKMIN) THEN
                                QCORR2 = ABS(QCORR / QY(ROW, COL + 1))
                            ELSE
                                QCORR2 = ABS(QCORR / QCHECKMIN)
                            END IF
                            QCHECKMIN = VTOL * GS * MIN(H(ROW, COL), &
                                H(ROW, COL + 1))
                            IF (ABS(QX(ROW, COL)) > QCHECKMIN) THEN
                                QCORR3 = ABS(QCORR / QX(ROW, COL))
                            ELSE
                                QCORR3 = ABS(QCORR / QCHECKMIN)
                            END IF
                            IF (MAX(QCORR1, QCORR2, QCORR3) > MAXQCORR) &
                                MAXQCORR = MAX(QCORR1, QCORR2, QCORR3)
                            IF (MAXQCORR >0.1) THEN
                                AA = 1
                            END IF
                            QY(ROW, COL) = QY(ROW, COL) + QCORR
                            QY(ROW, COL + 1) = QY(ROW, COL + 1) - QCORR
                            QX(ROW, COL) = QX(ROW, COL) - QCORR
                        ELSE
                            CALL LONGLOOP &
                                (ROW, COL, ROWS, COLS, QX, QY, K, TD)
                        END IF
                    END IF
                END IF
            ELSE
                !If cell is not submerged set flow to zero
                QX(ROW, COL) = 0
                QY(ROW, COL) = 0
            END IF
        END DO
    END DO
END DO
```

```
!If calculation ended without converging write details to log file
IF (ITT >= MAXITR) THEN
    WRITE (LOG_FILE, 10, ADVANCE = 'NO') &
        "FLOW FIELD FAILED TO CONVERGE AFTER "
    WRITE (LOG_FILE, 20, ADVANCE = 'NO') MAXITR
    WRITE (LOG_FILE, 10, ADVANCE = 'NO') " ITERATIONS AT TIME: "
    WRITE (LOG_FILE, 30, ADVANCE = 'NO') TIME
    WRITE (LOG_FILE, 10, ADVANCE = 'NO') " HOURS. MAX Q RATIO = "
    WRITE (LOG_FILE, 40) MAXQCORR
END IF

10 FORMAT (A)
20 FORMAT (I10)
30 FORMAT (F15.3)
40 FORMAT (F15.10)

END SUBROUTINE


SUBROUTINE LONGLOOP(ROW, COL, ROWS, COLS, QX, QY, K, TD)
!Subroutine to calculate flow corrections for loops around
!islands of non-submerged cells

REAL, DIMENSION(:,:), INTENT(INOUT) :: QX   !Flows in x direction
REAL, DIMENSION(:,:), INTENT(INOUT) :: QY   !Flows in y direction

REAL, DIMENSION(:,:), INTENT(IN)    :: K    !Friction parameters

INTEGER, DIMENSION(:,:), INTENT(IN) :: TD   !Hydraulic connectivity info

INTEGER, INTENT(IN) :: ROW      !Current row
INTEGER, INTENT(IN) :: COL      !Current column
INTEGER, INTENT(IN) :: ROWS     !Total rows
INTEGER, INTENT(IN) :: COLS     !Total columns

INTEGER, DIMENSION(:), ALLOCATABLE :: RN
INTEGER, DIMENSION(:), ALLOCATABLE :: CN
INTEGER, DIMENSION(:), ALLOCATABLE :: RQN
INTEGER, DIMENSION(:), ALLOCATABLE :: CQN
INTEGER, DIMENSION(:), ALLOCATABLE :: DXN
INTEGER, DIMENSION(:), ALLOCATABLE :: DYN

LOGICAL, DIMENSION(:,:), ALLOCATABLE :: VI

REAL DXO, DYO
REAL K1, K2
REAL HLQ1, HLQ2, HL1, HL2, HLQT, HLT
REAL QCORR

INTEGER N, N1, MIN_N
INTEGER ROW1, COL1, COL2

LOGICAL FIN, LP

character str

N = 10 * (ROWS + COLS)

ALLOCATE (VI(ROWS, COLS))
ALLOCATE (RN(N))    !Row number of cells in loop
ALLOCATE (CN(N))    !Column number of cells in loop
ALLOCATE (RQN(N))   !Row number of flow between cells in loop
                    !y component is between row r and row r + 1
                    !x component is between row c and row c + 1
```

```
ALLOCATE (CQN(N))    !Row number of flow between cells in loop
ALLOCATE (DXN(N))    !1 for x direction flow or zero for y direction
ALLOCATE (DYN(N))    !Zero for x direction flow or 1 for y direction

VI = .FALSE.
RN(1) = ROW
CN(1) = COL
RQN(1) = ROW
CQN(1) = COL
DXN(1) = 0
DYN(1) = 1
FIN = .FALSE.
LP = .FALSE.

VI(ROW, COL) = .TRUE.

IF (ROW == ROWS) THEN
    COL2 = COL
    FIN = .TRUE.
    DO WHILE (COL2 < COLS)
        COL2 = COL2 + 1
        IF ((TD(ROW, COL2) > 0) .AND. (VI(ROW, COL2) == .FALSE.)) THEN
            RN(2) = ROWS
            CN(2) = COL2
            RQN(2) = ROWS
            CQN(2) = COLS
            DXN(2) = 0
            DYN(2) = -1
            VI(ROW, COL2) = .TRUE.
            FIN = .FALSE.
            MIN_N = 1
            N = 2
            COL2 = COLS
        END IF
    END DO
ELSE
    !Conditons for calling the subroutine specify that the cells above and
    !above right are submerged, hence the 2nd & 3rd cells in the loop are
    !known
    RN(2) = ROW + 1
    CN(2) = COL
    RN(3) = ROW + 1
    CN(3) = COL + 1
    RQN(2) = ROW + 1
    CQN(2) = COL
    DXN(2) = 1
    DYN(2) = 0
    VI(ROW + 1, COL) = .TRUE.
    VI(ROW + 1, COL + 1) = .TRUE.
    MIN_N = 2
    N = 3
END IF

!Fin is set to true if the loop has been closed or has reached
!left (col == 1), right (col == cols)or bottom (row = 1) boundaries
!and is unable to proceed (i.e. no loop posible)
!If n is reduced to min_n no loop is possible
DO WHILE ((FIN == .FALSE.) .AND. (N > MIN_N))

    ROW1 = RN(N)
    COL1 = CN(N)

    !If the current cell and the previous cell are on the boundary row
    !and flow direction flow previous cell is not in the x direction
    !then the loop includes the boundary
```

```
IF ((RN(N) == ROWS) .AND. (RN(N - 1) == ROWS) .AND. &
    (DXN(N - 1) == 0)) THEN
    DXO = 0
    DYO = -1
ELSE
    DXO = DXN(N - 1)
    DYO = DYN(N - 1)
END IF

!If flow from previous cell is positive & in x direction
!i.e. from cell to the left (col - 1), then
!1st check cell below (row - 1) then cell to right (col + 1)
!then cell above (always trying to 'turn right' to find
!shortest clockwise loop)
IF (DXO == 1) THEN
    IF (ROW1 > 1) THEN
        !If cell in row (r - 1) is submerged and
        !not already included in the loop
        !then this is the next cell in the loop
        !VI set to TRUE for cells already in the loop
        IF ((TD(ROW1 - 1, COL1) > 0) .AND. &
            (VI(ROW1 - 1, COL1) == .FALSE.) ) THEN
            RQN(N) = ROW1 - 1
            CQN(N) = COL1
            DXN(N) = 0
            DYN(N) = -1
            N = N + 1
            RN(N) = ROW1 - 1
            CN(N) = COL1
            VI(ROW1 - 1, COL1) = .TRUE.
        ELSE
            IF (COL1 < COLS) THEN
                !Check if cell on right is the first cell in the loop
                !i.e. loop is closed
                IF ((ROW1 == RN(1)) .AND. (COL1 + 1 == CN(1))) THEN
                    FIN = .TRUE.
                    LP = .TRUE.
                    RQN(N) = ROW1
                    CQN(N) = COL1
                    DXN(N) = 1
                    DYN(N) = 0
                ELSE
                    !If cell on right is submerged & not already
                    !part of the loop
                    !this is the next cell in the loop
                    IF ((TD(ROW1, COL1 + 1) > 0) .AND. &
                        (VI(ROW1, COL1 + 1) == .FALSE.)) THEN
                        RQN(N) = ROW1
                        CQN(N) = COL1
                        DXN(N) = 1
                        DYN(N) = 0
                        N = N + 1
                        RN(N) = ROW1
                        CN(N) = COL1 + 1
                        VI(ROW1, COL1 + 1) = .TRUE.
                    ELSE
                        !Check if cell on above is the first
                        !cell in the loop
                        !i.e. loop is closed
                        IF (ROW1 < ROWS) THEN
                            IF ((ROW1 + 1 == RN(1)) .AND. &
                                (COL1 == CN(1))) THEN
                                FIN = .TRUE.
                                LP = .TRUE.
                                RQN(N) = ROW1
```

```
                                    CQN(N) = COL1
                                    DXN(N) = 0
                                    DYN(N) = 1
                                ELSE
                                    !If cell on above is submerged & not
                                    !already part of the loop
                                    !this is the next cell in the loop
                                    IF ((TD(ROW1 + 1, COL1) > 0) .AND. &
                                        (VI(ROW1 + 1, COL1) == .FALSE.)) &
                                        THEN
                                        RQN(N) = ROW1
                                        CQN(N) = COL1
                                        DXN(N) = 0
                                        DYN(N) = 1
                                        N = N + 1
                                        RN(N) = ROW1 + 1
                                        CN(N) = COL1
                                        VI(ROW1 + 1, COL1) = .TRUE.
                                    ELSE
                                        !If cell below, on right and above
                                        !are all unsubmerged then
                                        !the current cell is a deadend, so
                                        !the procedure goes back to
                                        !the previous step. VI now set to
                                        !true for the current cell so
                                        !it will not be checked again
                                        N = N - 1
                                    END IF
                                END IF
                            ELSE
                                !If ROW1 == ROWS the loop has reached the
                                !boundary, which is a special case
                                !loop can continue from 1st unsubmerged
                                !cell to the right
                                COL2 = COL1
                                FIN = .TRUE.
                                DO WHILE (COL2 < COLS)
                                    COL2 = COL2 + 1
                                    IF ((TD(ROW1, COL2) > 0) .AND. &
                                        (VI(ROW1, COL2) == .FALSE.)) THEN
                                        RQN(N) = ROW1
                                        CQN(N) = COL1
                                        DXN(N) = 0
                                        DYN(N) = 1

                                        N = N + 1
                                        RN(N) = ROW1
                                        CN(N) = COL2
                                        FIN = .FALSE.
                                        COL2 = COLS
                                        VI(ROW1, COL2) = .TRUE.
                                    END IF
                                END DO
                            END IF
                        END IF
                    END IF
                ELSE
                    !If procedure has reached right hand boundary
                    !and cannot proceed downward then no loop possible
                    FIN = .TRUE.
                END IF
            END IF
        ELSE
            !This situation should not occur
            FIN = .TRUE.
```

```
    END IF
ELSE
    !If flow from previous cell is negative & in y direction
    !i.e. from cell above (row + 1), then
    !1st check cell to left (col - 1) then cell below (row - 1)
    !then cell to right (always trying to 'turn right' to find
    !shortest clockwise loop)
    IF (DYO == -1) THEN
        IF (COL1 > 1) THEN
            IF ((TD(ROW1, COL1 - 1) > 0) .AND. &
                (VI(ROW1, COL1 - 1) == .FALSE.)) THEN
                RQN(N) = ROW1
                CQN(N) = COL1 - 1
                DXN(N) = -1
                DYN(N) = 0
                N = N + 1
                RN(N) = ROW1
                CN(N) = COL1 - 1
                VI(ROW1, COL1 - 1) = .TRUE.
            ELSE
                IF (ROW1 > 1) THEN
                    IF ((TD(ROW1 - 1, COL1) > 0) .AND. &
                        (VI(ROW1 - 1, COL1) == .FALSE.)) THEN
                        RQN(N) = ROW1 - 1
                        CQN(N) = COL1
                        DXN(N) = 0
                        DYN(N) = -1
                        N = N + 1
                        RN(N) = ROW1 - 1
                        CN(N) = COL1
                        VI(ROW1 - 1, COL1) = .TRUE.
                    ELSE
                        IF (COL1 < COLS) THEN
                            IF ((ROW1 == RN(1)) .AND. &
                                (COL1 + 1 == CN(1))) THEN
                                FIN = .TRUE.
                                LP = .TRUE.
                                RQN(N) = ROW1
                                CQN(N) = COL1
                                DXN(N) = 1
                                DYN(N) = 0
                            ELSE
                                IF ((TD(ROW1, COL1 + 1) > 0) .AND. &
                                    (VI(ROW1, COL1 + 1) == .FALSE.)) &
                                    THEN
                                    RQN(N) = ROW1
                                    CQN(N) = COL1
                                    DXN(N) = 1
                                    DYN(N) = 0
                                    N = N + 1
                                    RN(N) = ROW1
                                    CN(N) = COL1 + 1
                                    VI(ROW1, COL1 + 1) = .TRUE.
                                ELSE
                                    IF (ROW1 == ROWS) THEN
                                        COL2 = COL1
                                        FIN = .TRUE.
                                        DO WHILE (COL2 < COLS)
                                            COL2 = COL2 + 1
                                            IF ((TD(ROW1, COL2) > 0) &
                                                .AND. (VI(ROW1, COL2)&
                                                 == .FALSE.)) THEN
                                                CN(N) = COL2
                                                FIN = .FALSE.
                                                COL2 = COLS
```

```
                                                VI(ROW1, COL2) = &
                                                    .TRUE.
                                            END IF
                                        END DO
                                    ELSE
                                        N = N - 1
                                    END IF
                                END IF
                            END IF
                        ELSE
                            FIN = .TRUE.
                        END IF
                    END IF
                ELSE
                    FIN = .TRUE.
                END IF
            END IF
        ELSE
            FIN = .TRUE.
        END IF
    ELSE
        !If flow from previous cell is negative & in x direction
        !i.e. from cell on right (col + 1), then
        !1st check cell above (row + 1) then cell to left (col - 1)
        !then cell below (always trying to 'turn right' to find
        !shortest clockwise loop)
        IF (DXO == -1) THEN
            IF (ROW1 < ROWS) THEN
                IF ((ROW1 + 1 == RN(1)) .AND. (COL1 == CN(1))) THEN
                    FIN = .TRUE.
                    LP = .TRUE.
                    RQN(N) = ROW1
                    CQN(N) = COL1
                    DXN(N) = 0
                    DYN(N) = 1
                ELSE
                    IF ((TD(ROW1 + 1, COL1) > 0) .AND. &
                        (VI(ROW1 + 1, COL1) == .FALSE.)) THEN
                        RQN(N) = ROW1
                        CQN(N) = COL1
                        DXN(N) = 0
                        DYN(N) = 1

                        N = N + 1
                        RN(N) = ROW1 + 1
                        CN(N) = COL1
                        VI(ROW1 + 1, COL1) = .TRUE.
                    ELSE
                        IF (COL1 > 1) THEN
                            IF ((TD(ROW1, COL1 - 1) > 0) .AND. &
                                (VI(ROW1, COL1 - 1) == .FALSE.)) THEN
                                RQN(N) = ROW1
                                CQN(N) = COL1 - 1
                                DXN(N) = -1
                                DYN(N) = 0

                                N = N + 1
                                RN(N) = ROW1
                                CN(N) = COL1 - 1
                                VI(ROW1, COL1 - 1) = .TRUE.
                            ELSE
                                IF (ROW1 > 1) THEN
                                    IF ((TD(ROW1 - 1, COL1) > 0) &
                                        .AND. (VI(ROW1 - 1, COL1) == &
                                        .FALSE.)) THEN
```

```
                                    RQN(N) = ROW1 - 1
                                    CQN(N) = COL1
                                    DXN(N) = 0
                                    DYN(N) = -1

                                    N = N + 1
                                    RN(N) = ROW1 - 1
                                    CN(N) = COL1
                                    VI(ROW1 - 1, COL1) = .TRUE.
                                ELSE
                                    N = N - 1
                                END IF
                            ELSE
                                FIN = .TRUE.
                            END IF
                        END IF
                    ELSE
                        FIN = .TRUE.
                    END IF
                END IF
            END IF
        ELSE
            COL2 = COL1
            FIN = .TRUE.
            DO WHILE (COL2 < COLS)
                COL2 = COL2 + 1
                IF ((TD(ROW1, COL2) > 0) .AND. &
                    (VI(ROW1, COL2) == .FALSE.)) THEN
                    RQN(N) = ROW1
                    CQN(N) = COL1
                    DXN(N) = 0
                    DYN(N) = 1
                    N = N + 1
                    RN(N) = ROW1
                    CN(N) = COL2
                    FIN = .FALSE.
                    COL2 = COLS
                    VI(ROW1, COL2) = .TRUE.
                END IF
            END DO
        END IF
    ELSE
        !If flow from previous cell is positive & in y direction
        !(only remaining option) i.e. from cell below (row - 1),
        !then 1st check cell on right (col + 1) then cell above
        !(row + 1) then cell to left (always trying to
        !'turn right' to find shortest clockwise loop)
        IF (COL1 < COLS) THEN
            IF ((ROW1 == RN(1)) .AND. (COL1 + 1 == CN(1))) THEN
                FIN = .TRUE.
                LP = .TRUE.
                RQN(N) = ROW1
                CQN(N) = COL1
                DXN(N) = 1
                DYN(N) = 0
            ELSE
                IF ((TD(ROW1, COL1 + 1) > 0) .AND. &
                    (VI(ROW1, COL1 + 1) == .FALSE.)) THEN
                    RQN(N) = ROW1
                    CQN(N) = COL1
                    DXN(N) = 1
                    DYN(N) = 0
                    N = N + 1
                    RN(N) = ROW1
                    CN(N) = COL1 + 1
```

```
            VI(ROW1, COL1 + 1) = .TRUE.
        ELSE
            IF ((ROW1 + 1 == RN(1)) .AND. &
                (COL1 == CN(1))) THEN
                FIN = .TRUE.
                LP = .TRUE.
                RQN(N) = ROW1
                CQN(N) = COL1
                DXN(N) = 0
                DYN(N) = 1
            ELSE
                IF (ROW1 < ROWS) THEN
                    IF ((TD(ROW1 + 1, COL1) > 0) .AND. &
                        (VI(ROW1 + 1, COL1) == .FALSE.)) &
                        THEN
                        RQN(N) = ROW1
                        CQN(N) = COL1
                        DXN(N) = 0
                        DYN(N) = 1
                        N = N + 1
                        RN(N) = ROW1 + 1
                        CN(N) = COL1
                        VI(ROW1 + 1, COL1) = .TRUE.
                    ELSE
                        IF (COL1 > 1) THEN
                            IF ((TD(ROW1, COL1 - 1) > 0) &
                                .AND. (VI(ROW1, COL1 - 1)&
                                == .FALSE.)) THEN
                                RQN(N) = ROW1
                                CQN(N) = COL1 - 1
                                DXN(N) = -1
                                DYN(N) = 0
                                N = N + 1
                                RN(N) = ROW1
                                CN(N) = COL1 - 1
                                VI(ROW1, COL1 - 1) = &
                                    .TRUE.
                            ELSE
                                N = N - 1
                            END IF
                        ELSE
                            FIN = .TRUE.
                        END IF
                    END IF
                ELSE
                    COL2 = COL1
                    FIN = .TRUE.
                    DO WHILE (COL2 < COLS)
                        COL2 = COL2 + 1
                        IF ((TD(ROW1, COL2) > 0) .AND. &
                            (VI(ROW1, COL2) == .FALSE.)) &
                            THEN
                            RQN(N) = ROW1
                            CQN(N) = COL1
                            DXN(N) = 0
                            DYN(N) = 1
                            N = N + 1
                            RN(N) = ROW1
                            CN(N) = COL2
                            FIN = .FALSE.
                            COL2 = COLS
                            VI(ROW1, COL2) = .TRUE.
                        END IF
                    END DO
                END IF
```

```
                            END IF
                        END IF
                    END IF
                ELSE !(IF COL=COLS)
                    FIN = .TRUE.
                END IF

            END IF !IF (DXN(N - 1) == -1)
        END IF !IF (DYN(N - 1) == -1)
    END IF !IF (DXN(N - 1) == 1)
END DO !DO WHILE ((FIN == .FALSE.) .AND. (N > 2))

!If a closed loop was found then calculate the flow correction
IF (LP == .TRUE.) THEN
    HLT = 0
    HLQT = 0
    DO N1 = 1, N
        IF ((RN(N1) < ROWS) .OR. (DYN(N1) < 1)) THEN
            IF (DXN(N1) == 0) THEN
                K1 = RMS(K(RN(N1), CN(N1)), K(RN(N1) + DYN(N1), CN(N1)))
                HLQ1 = K1 * ABS(QY(RQN(N1), CQN(N1)))
                HL1 = DYN(N1) * HLQ1 * QY(RQN(N1), CQN(N1))
            ELSE
                K1 = RMS(K(RN(N1), CN(N1)), K(RN(N1), CN(N1) + DXN(N1)))
                HLQ1 = K1 * ABS(QX(RQN(N1), CQN(N1)))
                HL1 = DXN(N1) * HLQ1 * QX(RQN(N1), CQN(N1))
            END IF
            HLT = HLT + HL1
            HLQT = HLQT + HLQ1
        ELSE
            !Special case where the loop includes the tidal boundary
            K1 = K(RN(N1), CN(N1))
            K2 = K(RN(N1 + 1), CN(N1 + 1))
            HLQ1 = K1 * ABS(QY(RQN(N1), CQN(N1)))
            HLQ2 = K2 * ABS(QY(RQN(N1 + 1), CQN(N1 + 1)))
            HL1 = HLQ1 * QY(RQN(N1), CQN(N1))
            HL2 = HLQ2 * QY(RQN(N1 + 1), CQN(N1 + 1))
            HLT = HLT + HL1 - HL2
            HLQT = HLQT + HLQ1 + HLQ2
        END IF
    END DO

    IF (HLQT < 1E-9) THEN
        !Avoid possible divide by zero error
        QCORR = 0
    ELSE
        QCORR = -HLT / (2 * HLQT)
    END IF

    !Apply flow correction
    DO N1 = 1, N
        IF ((RN(N1) < ROWS) .OR. (DYN(N1) < 1)) THEN
            QX(RQN(N1), CQN(N1)) = QX(RQN(N1), CQN(N1)) + &
                (DXN(N1) * QCORR)
            QY(RQN(N1), CQN(N1)) = QY(RQN(N1), CQN(N1)) + &
                (DYN(N1) * QCORR)
        ELSE
            QY(ROWS, CN(N1)) = QY(ROWS, CN(N1)) + QCORR
            QY(ROWS, CN(N1 + 1)) = QY(ROWS, CN(N1 + 1)) - QCORR
        END IF
    END DO

END IF

END SUBROUTINE
```

```
SUBROUTINE S_ROUT(GRID, AGRID, C, ROWS, COLS, GS, QI, QI_N, &
    QI_R, QI_C, H1, MIN_H, SMIN, S_MAX)
!Subroutine to estimate flow depths due to gravity flow of the
!fluvial inflow

!Bed level data
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:), INTENT (IN) :: GRID
!Adjusted bed levels after depression filling
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:), INTENT (IN) :: AGRID

REAL, DIMENSION(:,:), INTENT (OUT) :: H1   !Slope routing Flow depth
REAL, DIMENSION(:,:), INTENT (IN)  :: C    !Chezy roughness
REAL, DIMENSION(:),   INTENT (IN)  :: QI   !Inflow values

INTEGER, DIMENSION(:), INTENT (IN) :: QI_R  !Row number for each inflow
INTEGER, DIMENSION(:), INTENT (IN) :: QI_C  !Column number for each inflow

REAL, INTENT(IN) :: SMIN    !Minimum slope used in depth calculation
REAL, INTENT(IN) :: S_MAX   !Maximum slope used in depth calculation
REAL, INTENT(IN) :: MIN_H   !Minimum depth
REAL, INTENT(IN) :: GS      !Cell size (m)

INTEGER, INTENT(IN)     :: ROWS !Total number of rows
INTEGER, INTENT(IN)     :: COLS !Total number of columns
INTEGER, INTENT(IN)     :: QI_N !Number of inflows

REAL, DIMENSION(:,:), ALLOCATABLE :: QIN !Flow routed into a cell
REAL, DIMENSION(:,:), ALLOCATABLE :: QT !Total flow routed through a cell

REAL (SELECTED_REAL_KIND(15,307)) BL, BLF, BLB, BLL, BLR
REAL (SELECTED_REAL_KIND(15,307)) BLFL, BLFR, BLBL, BLBR
REAL (SELECTED_REAL_KIND(15,307)) BLLF, BLLB, BLRF, BLRB
REAL (SELECTED_REAL_KIND(15,307)) SF, SB, SL, SR, SAV
REAL (SELECTED_REAL_KIND(15,307)) SFL, SFR, SBL, SBR
REAL (SELECTED_REAL_KIND(15,307)) SLF, SLB, SRF, SRB
REAL (SELECTED_REAL_KIND(15,307)) SMAX

REAL QI1, QT1, QOUT
REAL H2, H3
REAL WL, WLSF, WLSB, WLSL, WLSR
REAL WLSFL, WLSFR, WLSBL, WLSBR
REAL WLSLF, WLSLB, WLSRF, WLSRB
REAL WLSTOT
REAL QF, QB, QL, QR
REAL QFL, QFR, QBL, QBR
REAL QLF, QLB, QRF, QRB
REAL MINQ, QLIM, QOUT_TOT, QIN_TOT, QIN_TOT1
REAL QIN_TOT2, Q_TOT, QIN_TOT3, DQ_TOT

INTEGER ROW, COL, N, ITT, ITTT

INTEGER R, CL

CHARACTER STR
REAL AA

LOGICAL, SAVE :: FIRST_EXEC = .TRUE.

H1 = 0

ALLOCATE (QIN(ROWS, COLS))
ALLOCATE (QT(ROWS, COLS))
```

```
QIN  = 0
QT = 0

MINQ = 1E-2
QLIM = MINQ / (REAL(ROWS * COLS) * 2.0)

QOUT_TOT = 0
QIN_TOT = 0
DO N = 1, QI_N
    QIN(QI_R(N), QI_C(N)) = QIN(QI_R(N), QI_C(N)) + QI(N)
    QT(QI_R(N), QI_C(N)) = QT(QI_R(N), QI_C(N)) + QI(N)
    QIN_TOT = QIN_TOT + QI(N)
END DO

!Repeat procedure until all inflow has been routed to the
!downstream boundary (at row = rows)
DO WHILE(ABS(1 - (QOUT_TOT / QIN_TOT)) > 1E-4)

    !Forward routing from row 1 to row = rows
    DO ROW = 1, ROWS - 1
        DO COL = 1, COLS

            QI1 = QIN(ROW, COL)
            QT1 = MIN(QT(ROW, COL), QIN_TOT)
            IF (QI1 > QLIM) THEN

                BL = MAX(GRID(ROW, COL), AGRID(ROW, COL))   !Bed level
                !Depth of water filling any depression
                H3 = MAX(AGRID(ROW, COL) - GRID(ROW, COL), 0.0)
                !Bed level in adjacent cell in row = row + 1
                BLF = MAX(GRID(ROW + 1, COL), AGRID(ROW + 1, COL))
                IF (COL > 1) THEN
                    !Bed level in cell in next row & to left
                    BLFL = MAX(GRID(ROW + 1, COL - 1), &
                        AGRID(ROW + 1, COL - 1))
                ELSE
                    BLFL = 9999
                END IF
                IF (COL < COLS) THEN
                    !Bed level in cell in next row & to right
                    BLFR = MAX(GRID(ROW + 1, COL + 1), &
                        AGRID(ROW + 1, COL + 1))
                ELSE
                    BLFR = 9999
                END IF

                !Calculate bed slopes to 3 destination cells
                SF = (BL - BLF) / GS
                SFL = (BL - BLFL) / (GS * SQR2)
                SFR = (BL - BLFR) / (GS * SQR2)

                !SAV is average of positive slopes or -1
                !for 3 negative values
                SAV = AVSLP(SF, SFL, SFR)

                !SMAX is maximum of 3 slopes
                SMAX = MAX(SF, SFL, SFR)

                !If SMAX positive
                IF (SMAX > 1E-12) THEN
                    !Ensure SAV is within predefined limits
                    IF (SAV < SMIN) SAV = SMIN
                    IF (SAV > S_MAX) SAV = S_MAX

                    !Calculate water depth based on SAV
```

```
H2 = CBRT(((QT1 / (C(ROW, COL) * GS)) ** 2) &
    / REAL(SAV))
IF (H2 < MIN_H * 1.01) H2 = MIN_H * 1.01

!Calculate water level
WL = BL + H2

!Flow only routed to (row + 1) if water level is above
!bed level in adjacent cell, (row + 1, col)
IF (WL > BLF) THEN

    !If bed level is below that in the adjacent cell,
    !reduce the flow routed forward in proportion
    !to the difference between the depth above the
    !adjacent cell and that in the current cell
    IF (BL > BLF) THEN
        QOUT = QI1
    ELSE
        QOUT = QI1 * (WL - BLF) / H2
    END IF
    IF (QOUT > QI1) QOUT = QI1

    !Calculate slopes based in water level in current
    !cell and bed levels in adjacent cell
    WLSF = (WL - BLF) / GS
    WLSFL = (WL - BLFL) / (GS * SQR2)
    WLSFR = (WL - BLFR) / (GS * SQR2)

    !WLSF must be positive, set other slopes to zero
    !if negative
    IF (WLSFL < 0) WLSFL = 0
    IF (WLSFR < 0) WLSFR = 0

    !Sum of three slopes
    WLSTOT = WLSF + WLSFL + WLSFR

    !Set outflow to three cells in proportion
    !to the slopes
    QFL = QOUT * WLSFL / WLSTOT
    QF = QOUT * WLSF / WLSTOT
    QFR = QOUT * WLSFR / WLSTOT

    !Prevent flow from diverging to give values below
    !threshold MINQ
    IF (QFL < 1.01 * MINQ) THEN
        IF (QFR < 1.01 * MINQ) THEN
            !If QFL and QFR both negative and forward
            !bedslope is positive route all flow
            !directly positive or, if forward bedslope
            !zero or negative no flow routed forwards
            IF (SF > 1E-12) THEN
                QF = QF + QFL + QFR
            ELSE
                QF = 0
            END IF
            QFL = 0
            QFR = 0
        ELSE
            !If QFL only below threshold and max of
            !SF, SFR positive then redistribute
            !flow to forward and forward-right cells
            IF (MAX(SF, SFR) > 1E-12) THEN
                QF = QF + (QFL / 2.0)
                QFR = QFR + (QFL / 2.0)
            ELSE
```

```
                                QF = 0
                                QFR = 0
                            END IF
                            QFL = 0
                        END IF
                    ELSE
                        !If QFR only below threshold and max of SF,
                        !SFL positive then redistribute
                        !flow to forward and forward-left cells
                        IF (QFR < 1.01 * MINQ) THEN
                            IF (MAX(SF, SFL) > 1E-12) THEN
                                QF = QF + (QFR / 2.0)
                                QFL = QFL + (QFR / 2.0)
                            ELSE
                                QF = 0
                                QFL = 0
                            END IF
                            QFR = 0
                        END IF
                    END IF

                    !If QF above thresold value, calculate inflows
                    !to 3 cells in next row QT is the total flow
                    !routed through the cell, used to calculate
                    !the depth
                    IF (QF > MINQ) THEN

                        QIN(ROW + 1, COL) = QIN(ROW + 1, COL) + QF
                        QT(ROW + 1, COL) = QT(ROW + 1, COL) + QF

                        IF (COL > 1) THEN
                            QIN(ROW + 1, COL - 1) = &
                                QIN(ROW + 1, COL - 1) + QFL
                            QT(ROW + 1, COL - 1) = &
                                QT(ROW + 1, COL - 1) + QFL
                        END IF
                        IF (COL < COLS) THEN
                            QIN(ROW + 1, COL + 1) = &
                                QIN(ROW + 1, COL + 1) + QFR
                            QT(ROW + 1, COL + 1) = &
                                QT(ROW + 1, COL + 1) + QFR
                        END IF

                        !QIN to current cell reduced
                        !by sum of outflows
                        QIN(ROW, COL) = QIN(ROW, COL) - QOUT
                    ELSE
                        !If QF below thresold then all flow routed
                        !to orthogonal neighbour with lowest bed level
                        CALL MAXSLOPE(ROW, COL, ROWS, COLS, GRID, &
                            AGRID, QI1, QIN, QT)
                        QIN(ROW, COL) = 0

                    END IF

                    !Set water depth -
                    !this is the slope routing output
                    IF (QT1 > MINQ) H1(ROW, COL) = WL - GRID(ROW, COL)

                END IF
            END IF
        END IF
    END DO
END DO
```

```
!For row = rows depth calculated based on max slope
!All flow from this row is routed across the boundary and added
!to the total outflow
DO COL = 1, COLS
    QOUT_TOT = QOUT_TOT + QIN(ROWS, COL)
    H2 = CBRT(((QIN(ROWS, COL) / (C(ROWS, COL) * GS)) ** 2) / S_MAX)
    IF (H2 > H1(ROWS, COL)) H1(ROWS, COL) = H2
END DO
QIN(ROWS,:) = 0

!Any inflow remaining in the model domain is now routed from left to
!right using the same procedure
DO COL = 1, COLS - 1
    DO ROW = 1, ROWS

        QI1 = QIN(ROW, COL)
        QT1 = MIN(QT(ROW, COL), QIN_TOT)
        IF (QI1 > QLIM) THEN

            BL = MAX(GRID(ROW, COL), AGRID(ROW, COL))
            H3 = MAX(AGRID(ROW, COL) - GRID(ROW, COL), 0.0)
            BLR = MAX(GRID(ROW, COL + 1), AGRID(ROW, COL + 1))
            IF (ROW < ROWS) THEN
                BLRF = MAX(GRID(ROW + 1, COL + 1), &
                    AGRID(ROW + 1, COL + 1))
            ELSE
                BLRF = -9999
            END IF
            IF (ROW > 1) THEN
                BLRB = MAX(GRID(ROW - 1, COL + 1), &
                    AGRID(ROW - 1, COL + 1))
            ELSE
                BLRB = 9999
            END IF

            SR = (BL - BLR) / GS
            SRF = (BL - BLRF) / (GS * SQR2)
            SRB = (BL - BLRB) / (GS * SQR2)

            SAV = AVSLP(SR, SRF, SRB)

            SMAX = MAX(SR, SRF, SRB)

            IF (SMAX > 1E-12) THEN
                IF (SAV < SMIN) SAV = SMIN
                IF (SAV > S_MAX) SAV = S_MAX

                H2 = CBRT(((QT1 / (C(ROW, COL) * GS)) ** 2) &
                    / REAL(SAV))
                IF (H2 < MIN_H * 1.01) H2 = MIN_H * 1.01

                WL = BL + H2

                IF (WL > BLR) THEN

                    IF (BL > BLR) THEN
                        QOUT = QI1
                    ELSE
                        QOUT = QI1 * (WL - BLR) / H2
                    END IF
                    IF (QOUT > QI1) QOUT = QI1

                    WLSR = (WL - BLR) / GS
                    WLSRF = (WL - BLRF) / (GS * SQR2)
                    WLSRB = (WL - BLRB) / (GS * SQR2)
```

```
IF (WLSRF < 0) WLSRF = 0
IF (WLSRB < 0) WLSRB = 0

WLSTOT = WLSR + WLSRF + WLSRB

QRF = QOUT * WLSRF / WLSTOT
QR = QOUT * WLSR / WLSTOT
QRB = QOUT * WLSRB / WLSTOT

IF (QRF < 1.01 * MINQ) THEN
    IF (QRB < 1.01 * MINQ) THEN
        IF (SR > 1E-12) THEN
            QR = QR + QRF + QRB
        ELSE
            QR = 0
        END IF
        QRF = 0
        QRB = 0
    ELSE
        IF (MAX(SR, SRB) > 1E-12) THEN
            QR = QR + (QRF / 2.0)
            QRB = QRB + (QRF / 2.0)
        ELSE
            QR = 0
            QRB = 0
        END IF
        QRF = 0
    END IF
ELSE
    IF (QRB < 1.01 * MINQ) THEN
        IF (MAX(SR, SRF) > 1E-12) THEN
            QR = QR + (QRB / 2.0)
            QRF = QRF + (QRB / 2.0)
        ELSE
            QR = 0
            QRF = 0
        END IF
        QRB = 0
    END IF
END IF

IF (QR > MINQ) THEN
    QIN(ROW, COL + 1) = QIN(ROW, COL + 1) + QR
    QT(ROW, COL + 1) = QT(ROW, COL + 1) + QR

    IF (ROW < ROWS) THEN
        QIN(ROW + 1, COL + 1) = &
            QIN(ROW + 1, COL + 1) + QRF
        QT(ROW + 1, COL + 1) = &
            QT(ROW + 1, COL + 1) + QRF
    END IF
    IF (ROW > 1) THEN
        QIN(ROW - 1, COL + 1) = &
            QIN(ROW - 1, COL + 1) + QRB
        QT(ROW - 1, COL + 1) = &
            QT(ROW - 1, COL + 1) + QRB
    END IF
    QIN(ROW, COL) = QIN(ROW, COL) - QOUT
ELSE
    CALL MAXSLOPE(ROW, COL, ROWS, COLS, &
        GRID, AGRID, QI1, QIN, QT)
    QIN(ROW, COL) = 0
END IF
```

```
                    IF (QT1 > MINQ) H1(ROW, COL) = WL - GRID(ROW, COL)

                END IF
            END IF
        END IF
    END DO
END DO

!Any inflow remaining in the model domain is now routed from right to
!left using the same procedure
DO COL = COLS, 2, -1
    DO ROW = 1, ROWS

        QI1 = QIN(ROW, COL)
        QT1 = MIN(QT(ROW, COL), QIN_TOT)
        IF (QI1 > QLIM) THEN

            BL = MAX(GRID(ROW, COL), AGRID(ROW, COL))
            H3 = MAX(AGRID(ROW, COL) - GRID(ROW, COL), 0.0)
            BLL = MAX(GRID(ROW, COL - 1), AGRID(ROW, COL - 1))
            IF (ROW < ROWS) THEN
                BLLF = MAX(GRID(ROW + 1, COL - 1), &
                    AGRID(ROW + 1, COL - 1))
            ELSE
                BLLF = -9999
            END IF
            IF (ROW > 1) THEN
                BLLB = MAX(GRID(ROW - 1, COL - 1), &
                    AGRID(ROW - 1, COL - 1))
            ELSE
                BLLB = 9999
            END IF

            SL = (BL - BLL) / GS
            SLF = (BL - BLLF) / (GS * SQR2)
            SLB = (BL - BLLB) / (GS * SQR2)

            SAV = AVSLP(SL, SLF, SLB)

            SMAX = MAX(SL, SLF, SLB)

            IF (SMAX > 1E-12) THEN
                IF (SAV < SMIN) SAV = SMIN
                IF (SAV > S_MAX) SAV = S_MAX

                H2 = CBRT(((QT1 / (C(ROW, COL) * GS)) ** 2) &
                    / REAL(SAV))
                IF (H2 < MIN_H * 1.01) H2 = MIN_H * 1.01

                WL = BL + H2

                IF (WL > BLL) THEN

                    IF (BL > BLL) THEN
                        QOUT = QI1
                    ELSE
                        QOUT = QI1 * (WL - BLL) / H2
                    END IF
                    IF (QOUT > QI1) QOUT = QI1

                    WLSL = (WL - BLL) / GS
                    WLSLF = (WL - BLLF) / (GS * SQR2)
                    WLSLB = (WL - BLLB) / (GS * SQR2)

                    IF (WLSLF < 0) WLSLF = 0
```

```
            IF (WLSLB < 0) WLSLB = 0

            WLSTOT = WLSL + WLSLF + WLSLB

            QLF = QOUT * WLSLF / WLSTOT
            QL = QOUT * WLSL / WLSTOT
            QLB = QOUT * WLSLB / WLSTOT

            IF (QLF < 1.01 * MINQ) THEN
                IF (QLB < 1.01 * MINQ) THEN
                    IF (SL > 1E-12) THEN
                        QL = QL + QLF + QLB
                    ELSE
                        QL = 0
                    END IF
                    QLF = 0
                    QLB = 0
                ELSE
                    IF (MAX(SL, SLB) > 1E-12) THEN
                        QL = QL + (QLF / 2.0)
                        QLB = QLB + (QLF / 2.0)
                    ELSE
                        QL = 0
                        QLB = 0
                    END IF
                    QLF = 0
                END IF
            ELSE
                IF (QLB < 1.01 * MINQ) THEN
                    IF (MAX(SL, SLF) > 1E-12) THEN
                        QL = QL + (QLB / 2.0)
                        QLF = QLF + (QLB / 2.0)
                    ELSE
                        QL = 0
                        QLF = 0
                    END IF
                    QLB = 0
                END IF
            END IF

            IF (QL > MINQ) THEN
                QIN(ROW, COL - 1) = QIN(ROW, COL - 1) + QL
                QT(ROW, COL - 1) = QT(ROW, COL - 1) + QL
                IF (ROW < ROWS) THEN
                    QIN(ROW + 1, COL - 1) = &
                        QIN(ROW + 1, COL - 1) + QLF
                    QT(ROW + 1, COL - 1) = &
                        QT(ROW + 1, COL - 1) + QLF
                END IF
                IF (ROW > 1) THEN
                    QIN(ROW - 1, COL - 1) = &
                        QIN(ROW - 1, COL - 1) + QLB
                    QT(ROW - 1, COL - 1) = &
                        QT(ROW - 1, COL - 1) + QLB
                END IF
                QIN(ROW, COL) = QIN(ROW, COL) - QOUT
            ELSE
                CALL MAXSLOPE(ROW, COL, ROWS, COLS, &
                    GRID, AGRID, QI1, QIN, QT)
                QIN(ROW, COL) = 0
            END IF

            IF (QT1 > MINQ) H1(ROW, COL) = WL - GRID(ROW, COL)

        END IF
```

```
            END IF
        END IF
    END DO
END DO


!Any inflow remaining in the model domain is now routed backwards
!using the same procedure
DO ROW = ROWS, 2, -1
    DO COL = 1, COLS

        QI1 = QIN(ROW, COL)
        QT1 = MIN(QT(ROW, COL), QIN_TOT)
        IF (QI1 > QLIM) THEN

            BL = MAX(GRID(ROW, COL), AGRID(ROW, COL))
            H3 = MAX(AGRID(ROW, COL) - GRID(ROW, COL), 0.0)
            BLB = MAX(GRID(ROW - 1, COL), AGRID(ROW - 1, COL))
            IF (COL > 1) THEN
                BLBL = MAX(GRID(ROW - 1, COL - 1), &
                    AGRID(ROW - 1, COL - 1))
            ELSE
                BLBL = 9999
            END IF
            IF (COL < COLS) THEN
                BLBR = MAX(GRID(ROW - 1, COL + 1), &
                    AGRID(ROW - 1, COL + 1))
            ELSE
                BLBR = 9999
            END IF

            SB = (BL - BLB) / GS
            SBL = (BL - BLBL) / (GS * SQR2)
            SBR = (BL - BLBR) / (GS * SQR2)

            SAV = AVSLP(SB, SBL, SBR)

            SMAX = MAX(SB, SBL, SBR)

            IF (SMAX > 1E-12) THEN
                IF (SAV < SMIN) SAV = SMIN
                IF (SAV > S_MAX) SAV = S_MAX

                H2 = CBRT(((QT1 / (C(ROW, COL) * GS)) ** 2) &
                    / REAL(SAV))
                IF (H2 < MIN_H * 1.01) H2 = MIN_H * 1.01

                WL = BL + H2

                IF (WL > BLB) THEN

                    IF (BL > BLB) THEN
                        QOUT = QI1
                    ELSE
                        QOUT = QI1 * (WL - BLB) / H2
                    END IF
                    IF (QOUT > QI1) QOUT = QI1

                    WLSB = (WL - BLB) / GS
                    WLSBL = (WL - BLBL) / (GS * SQR2)
                    WLSBR = (WL - BLBR) / (GS * SQR2)

                    IF (WLSBL < 0) WLSBL = 0
                    IF (WLSBR < 0) WLSBR = 0
```

```
                    WLSTOT = WLSB + WLSBL + WLSBR

                    QBL = QOUT * WLSBL / WLSTOT
                    QB = QOUT * WLSB / WLSTOT
                    QBR = QOUT * WLSBR / WLSTOT

                    IF (QBL < 1.01 * MINQ) THEN
                        IF (QBR < 1.01 * MINQ) THEN
                            IF (SB > 1E-12) THEN
                                QB = QB + QBL + QBR
                            ELSE
                                QB = 0
                            END IF
                            QBL = 0
                            QBR = 0
                        ELSE
                            IF (MAX(SB, SBR) > 1E-12) THEN
                                QB = QB + (QBL / 2.0)
                                QBR = QBR + (QBL / 2.0)
                            ELSE
                                QB = 0
                                QBR = 0
                            END IF
                            QBL = 0
                        END IF
                    ELSE
                        IF (QBR < 1.01 * MINQ) THEN
                            IF (MAX(SB, SBL) > 1E-12) THEN
                                QB = QB + (QBR / 2.0)
                                QBL = QBL + (QBR / 2.0)
                            ELSE
                                QB = 0
                                QBL = 0
                            END IF
                            QBR = 0
                        END IF
                    END IF

                    IF (QB > MINQ) THEN
                        QIN(ROW - 1, COL) = QIN(ROW - 1, COL) + QB
                        QT(ROW - 1, COL) = QT(ROW - 1, COL) + QB
                        IF (COL > 1) THEN
                            QIN(ROW - 1, COL - 1) = &
                                QIN(ROW - 1, COL - 1) + QBL
                            QT(ROW - 1, COL - 1) = &
                                QT(ROW - 1, COL - 1) + QBL
                        END IF
                        IF (COL < COLS) THEN
                            QIN(ROW - 1, COL + 1) = &
                                QIN(ROW - 1, COL + 1) + QBR
                            QT(ROW - 1, COL + 1) = &
                                QT(ROW - 1, COL + 1) + QBR
                        END IF
                        QIN(ROW, COL) = QIN(ROW, COL) - QOUT
                    ELSE
                        CALL MAXSLOPE(ROW, COL, ROWS, COLS, GRID, &
                            AGRID, QI1, QIN, QT)
                        QIN(ROW, COL) = 0
                    END IF

                    IF (QT1 > MINQ) H1(ROW, COL) = WL - GRID(ROW, COL)

                END IF
            END IF
        END IF
```

```
        END DO
    END DO
END DO

END SUBROUTINE

!Depression filling
SUBROUTINE FILL_HOLES(GRID, AGRID, ROWS, COLS)

REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:), INTENT(IN)    :: GRID
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:), INTENT(INOUT):: AGRID

INTEGER, INTENT(IN) :: ROWS
INTEGER, INTENT(IN) :: COLS

INTEGER, DIMENSION(:), ALLOCATABLE :: RWL
INTEGER, DIMENSION(:), ALLOCATABLE :: CLL

REAL (SELECTED_REAL_KIND(15,307)) NBEDB, NBEDF, NBEDL, NBEDR
REAL (SELECTED_REAL_KIND(15,307)) CBED, NBED_MIN, NBED_MAX

LOGICAL, DIMENSION(:,:), ALLOCATABLE :: F1
LOGICAL, DIMENSION(:,:), ALLOCATABLE :: F2

REAL INC

INTEGER ROW, COL, ROW1, COL1, N, M

LOGICAL, DIMENSION(:,:), ALLOCATABLE, SAVE :: ACTIVE

LOGICAL, SAVE :: FIRST_EXEC = .TRUE.

LOGICAL CHANGE, F3

IF (FIRST_EXEC == .TRUE.) THEN
    FIRST_EXEC = .FALSE.
    ALLOCATE (ACTIVE(ROWS, COLS))
    ACTIVE = .TRUE.
    AGRID = -9999
END IF

ALLOCATE (RWL(ROWS * COLS))
ALLOCATE (CLL(ROWS * COLS))
ALLOCATE (F1(ROWS, COLS))
ALLOCATE (F2(ROWS, COLS))

F1 = .FALSE.

CHANGE = .TRUE.
INC = 1E-8


DO ROW = 1, ROWS - 1
    DO COL = 1, COLS

        IF (F1(ROW, COL) == .FALSE.) THEN

            !Calculate the highest and lowest adjusted bed levels (agrid)
            !of othogonal neighbours
            CALL CELLCHECK(GRID, AGRID, ROW, COL, ROWS, COLS, CBED, &
                NBED_MIN, NBED_MAX)

            !If bed level is less than inc/2 (small value) above lowest
            !neighbour raise adjusted bed level
            IF (CBED  - NBED_MIN < INC / 2) THEN
```

```
RWL(1) = ROW
CLL(1) = COL

N = 1
CHANGE = .TRUE.
F2 = .FALSE.
F3 = .FALSE.

F1(ROW, COL) = .TRUE.
F2(ROW, COL) = .TRUE.

!Recheck all adjusted cells until no changes were
!made on last iteration
DO WHILE (CHANGE == .TRUE.)

    CHANGE = .FALSE.
    M = 0

    !Raise adjusted bed level and check neighbouring
    !cells & ajdust if necessary
    DO WHILE (M < N)

        M = M + 1

        ROW1 = RWL(M)
        COL1 = CLL(M)

        IF (F3 == .TRUE.) THEN
            CALL CELLCHECK(GRID, AGRID, ROW1, COL1, &
                ROWS, COLS, CBED, NBED_MIN, NBED_MAX)
        ELSE
            F3 = .TRUE.
        END IF

        IF (CBED  - NBED_MIN < INC / 2) THEN

            AGRID(ROW1, COL1) = NBED_MAX + INC

            CHANGE = .TRUE.

            IF (ROW1 > 1) THEN
                IF (F2(ROW1 - 1, COL1) == .FALSE.) THEN
                    N = N + 1
                    RWL(N) = ROW1 - 1
                    CLL(N) = COL1
                    F1(ROW1 - 1, COL1) = .TRUE.
                    F2(ROW1 - 1, COL1) = .TRUE.
                END IF
            END IF
            IF (ROW1 < ROWS - 1) THEN
                IF (F2(ROW1 + 1, COL1) == .FALSE.) THEN
                    N = N + 1
                    RWL(N) = ROW1 + 1
                    CLL(N) = COL1
                    F1(ROW1 + 1, COL1) = .TRUE.
                    F2(ROW1 + 1, COL1) = .TRUE.
                END IF
            END IF
            IF (COL1 > 1) THEN
                IF (F2(ROW1, COL1 - 1) == .FALSE.) THEN
                    N = N + 1
                    RWL(N) = ROW1
                    CLL(N) = COL1 - 1
                    F1(ROW1, COL1 - 1) = .TRUE.
```

```
                                F2(ROW1, COL1 - 1) = .TRUE.
                            END IF
                        END IF
                        IF (COL1 < COLS) THEN
                            IF (F2(ROW1, COL1 + 1) == .FALSE.) THEN
                                N = N + 1
                                RWL(N) = ROW1
                                CLL(N) = COL1 + 1
                                F1(ROW1, COL1 + 1) = .TRUE.
                                F2(ROW1, COL1 + 1) = .TRUE.
                            END IF
                        END IF
                    END IF
                END DO
            END DO
        END IF
    END IF
END DO
END DO

F1 = .FALSE.
CHANGE = .TRUE.

DO ROW = 1, ROWS - 1
    DO COL = 1, COLS

        IF (F1(ROW, COL) == .FALSE.) THEN

            CALL CELLCHECK(GRID, AGRID, ROW, COL, ROWS, COLS, CBED, &
                NBED_MIN, NBED_MAX)

            !If ajusted bed level more than 1.1Inc (small value) above
            !lowest neighbour Reduce to lowest neighbour bed level + INC
            !If adjuted bed level becomes lower than the actual bed level
            !then it is ignored in the slope routing calculation
            IF (AGRID(ROW, COL) > NBED_MIN + (1.1 * INC)) THEN

                RWL(1) = ROW
                CLL(1) = COL

                N = 1
                CHANGE = .TRUE.
                F2 = .FALSE.
                F3 = .FALSE.

                F1(ROW, COL) = .TRUE.
                F2(ROW, COL) = .TRUE.

                DO WHILE (CHANGE == .TRUE.)

                    CHANGE = .FALSE.
                    M = 0

                    DO WHILE (M < N)

                        M = M + 1

                        ROW1 = RWL(M)
                        COL1 = CLL(M)

                        IF (F3 == .TRUE.) THEN
                            CALL CELLCHECK(GRID, AGRID, ROW1, COL1, &
                                ROWS, COLS, CBED, NBED_MIN, NBED_MAX)
                        ELSE
                            F3 = .TRUE.
```

```
                        END IF

                        IF (AGRID(ROW1, COL1) > NBED_MIN + (1.1 * INC)) &
                            THEN

                            AGRID(ROW1, COL1) = NBED_MIN + INC

                            CHANGE = .TRUE.

                            IF (ROW1 > 1) THEN
                                IF (F2(ROW1 - 1, COL1) == .FALSE.) THEN
                                    N = N + 1
                                    RWL(N) = ROW1 - 1
                                    CLL(N) = COL1
                                    F1(ROW1 - 1, COL1) = .TRUE.
                                    F2(ROW1 - 1, COL1) = .TRUE.
                                END IF
                            END IF
                            IF (ROW1 < ROWS - 1) THEN
                                IF (F2(ROW1 + 1, COL1) == .FALSE.) THEN
                                    N = N + 1
                                    RWL(N) = ROW1 + 1
                                    CLL(N) = COL1
                                    F1(ROW1 + 1, COL1) = .TRUE.
                                    F2(ROW1 + 1, COL1) = .TRUE.
                                END IF
                            END IF
                            IF (COL1 > 1) THEN
                                IF (F2(ROW1, COL1 - 1) == .FALSE.) THEN
                                    N = N + 1
                                    RWL(N) = ROW1
                                    CLL(N) = COL1 - 1
                                    F1(ROW1, COL1 - 1) = .TRUE.
                                    F2(ROW1, COL1 - 1) = .TRUE.
                                END IF
                            END IF
                            IF (COL1 < COLS) THEN
                                IF (F2(ROW1, COL1 + 1) == .FALSE.) THEN
                                    N = N + 1
                                    RWL(N) = ROW1
                                    CLL(N) = COL1 + 1
                                    F1(ROW1, COL1 + 1) = .TRUE.
                                    F2(ROW1, COL1 + 1) = .TRUE.
                                END IF
                            END IF
                        END IF
                    END DO
                END DO
            END IF
        END IF
    END DO
END DO


END SUBROUTINE


SUBROUTINE CELLCHECK(GRID, AGRID, ROW, COL, ROWS, &
    COLS, CBED, NBED_MIN, NBED_MAX)
!Subroutine to calculate max and min neighbour bed levels

REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:), INTENT(IN)    :: GRID
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:), INTENT(IN)    :: AGRID

REAL (SELECTED_REAL_KIND(15,307)), INTENT(OUT)    :: CBED
```

```
REAL (SELECTED_REAL_KIND(15,307)), INTENT(OUT)    :: NBED_MIN
REAL (SELECTED_REAL_KIND(15,307)), INTENT(OUT)    :: NBED_MAX

INTEGER, INTENT(IN) :: ROWS
INTEGER, INTENT(IN) :: COLS
INTEGER, INTENT(IN) :: ROW
INTEGER, INTENT(IN) :: COL

REAL (SELECTED_REAL_KIND(15,307)) NBEDB, NBEDF, NBEDL, NBEDR

CBED = MAX(AGRID(ROW, COL), GRID(ROW, COL))

IF (ROW > 1) THEN
    NBEDB = MAX(AGRID(ROW - 1, COL), GRID(ROW - 1, COL))
ELSE
    NBEDB = CBED
END IF
IF (ROW < ROWS) THEN
    NBEDF = MAX(AGRID(ROW + 1, COL), GRID(ROW + 1, COL))
ELSE
    NBEDF = CBED
END IF
IF (COL > 1) THEN
    NBEDL = MAX(AGRID(ROW, COL - 1), GRID(ROW, COL - 1))
ELSE
    NBEDL = CBED
END IF
IF (COL < COLS) THEN
    NBEDR = MAX(AGRID(ROW, COL + 1), GRID(ROW, COL + 1))
ELSE
    NBEDR = CBED
END IF

NBED_MIN = MIN(NBEDB, NBEDF, NBEDL, NBEDR)
NBED_MAX = MAX(NBEDB, NBEDF, NBEDL, NBEDR)

END SUBROUTINE

SUBROUTINE MAXSLOPE(ROW, COL, ROWS, COLS, GRID, AGRID, QI, QIN, QT)
!Subroutine to route all flow to lowest orthoginal neighbour

REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:), INTENT(IN)    :: GRID
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:), INTENT(IN)    :: AGRID

REAL, DIMENSION(:,:), INTENT(INOUT) :: QIN
REAL, DIMENSION(:,:), INTENT(INOUT) :: QT

REAL, INTENT(IN) :: QI

REAL (SELECTED_REAL_KIND(15,307)) DF, DB, DL, DR

INTEGER, INTENT(IN) :: ROWS
INTEGER, INTENT(IN) :: COLS
INTEGER, INTENT(IN) :: ROW
INTEGER, INTENT(IN) :: COL

REAL (SELECTED_REAL_KIND(15,307)) FBED, BBED, LBED, RBED

DF = 1E-10
DB = 2E-10
DL = 3E-10
DR = 4E-10

IF (ROW > 1) THEN
    BBED = MAX(AGRID(ROW - 1, COL), GRID(ROW - 1, COL)) + DB
```

```
IF (ROW < ROWS) THEN
    FBED = MAX(AGRID(ROW + 1, COL), GRID(ROW + 1, COL)) + DF
    IF (COL > 1) THEN
        LBED = MAX(AGRID(ROW, COL - 1), GRID(ROW, COL - 1)) + DL
        IF (COL < COLS) THEN
            RBED = MAX(AGRID(ROW, COL + 1), GRID(ROW, COL + 1)) + DR

            IF (BBED < MIN(FBED, RBED, LBED)) THEN
                QIN(ROW - 1, COL) = QIN(ROW - 1, COL) + QI
                QT(ROW - 1, COL) = QT(ROW - 1, COL) + QI
            ELSE
                IF (RBED < MIN(FBED, BBED, LBED)) THEN
                    QIN(ROW, COL + 1) = QIN(ROW, COL + 1) + QI
                    QT(ROW, COL + 1) = QT(ROW, COL + 1) + QI
                ELSE
                    IF (LBED < MIN(FBED, BBED, RBED)) THEN
                        QIN(ROW, COL - 1) = QIN(ROW, COL - 1) + QI
                        QT(ROW, COL - 1) = QT(ROW, COL - 1) + QI
                    ELSE
                        QIN(ROW + 1, COL) = QIN(ROW + 1, COL) + QI
                        QT(ROW + 1, COL) = QT(ROW + 1, COL) + QI
                    END IF
                END IF
            END IF
        ELSE
            IF (BBED < MIN(FBED, LBED)) THEN
                QIN(ROW - 1, COL) = QIN(ROW - 1, COL) + QI
                QT(ROW - 1, COL) = QT(ROW - 1, COL) + QI
            ELSE
                IF (LBED < MIN(FBED, BBED)) THEN
                    QIN(ROW, COL - 1) = QIN(ROW, COL - 1) + QI
                    QT(ROW, COL - 1) = QT(ROW, COL - 1) + QI
                ELSE
                    QIN(ROW + 1, COL) = QIN(ROW + 1, COL) + QI
                    QT(ROW + 1, COL) = QT(ROW + 1, COL) + QI
                END IF
            END IF
        END IF
    ELSE
        IF (COL < COLS) THEN
            RBED = MAX(AGRID(ROW, COL + 1), GRID(ROW, COL + 1)) + DR

            IF (BBED < MIN(FBED, RBED)) THEN
                QIN(ROW - 1, COL) = QIN(ROW - 1, COL) + QI
                QT(ROW - 1, COL) = QT(ROW - 1, COL) + QI
            ELSE
                IF (RBED < MIN(FBED, BBED)) THEN
                    QIN(ROW, COL + 1) = QIN(ROW, COL + 1) + QI
                    QT(ROW, COL + 1) = QT(ROW, COL + 1) + QI
                ELSE
                    QIN(ROW + 1, COL) = QIN(ROW + 1, COL) + QI
                    QT(ROW + 1, COL) = QT(ROW + 1, COL) + QI
                END IF
            END IF
        ELSE
            IF (BBED < FBED) THEN
                QIN(ROW - 1, COL) = QIN(ROW - 1, COL) + QI
                QT(ROW - 1, COL) = QT(ROW - 1, COL) + QI
            ELSE
                QIN(ROW + 1, COL) = QIN(ROW + 1, COL) + QI
                QT(ROW + 1, COL) = QT(ROW + 1, COL) + QI
            END IF
        END IF
    END IF
ELSE
```

```
        IF (COL > 1) THEN
            LBED = MAX(AGRID(ROW, COL - 1), GRID(ROW, COL - 1)) + DL
            IF (COL < COLS) THEN
                RBED = MAX(AGRID(ROW, COL + 1), GRID(ROW, COL + 1)) + DR

                IF (BBED < MIN(RBED, LBED)) THEN
                    QIN(ROW - 1, COL) = QIN(ROW - 1, COL) + QI
                    QT(ROW - 1, COL) = QT(ROW - 1, COL) + QI
                ELSE
                    IF (RBED < MIN(BBED, LBED)) THEN
                        QIN(ROW, COL + 1) = QIN(ROW, COL + 1) + QI
                        QT(ROW, COL + 1) = QT(ROW, COL + 1) + QI
                    ELSE
                        QIN(ROW, COL - 1) = QIN(ROW, COL - 1) + QI
                        QT(ROW, COL - 1) = QT(ROW, COL - 1) + QI
                    END IF
                END IF
            ELSE
                IF (BBED < LBED) THEN
                    QIN(ROW - 1, COL) = QIN(ROW - 1, COL) + QI
                    QT(ROW - 1, COL) = QT(ROW - 1, COL) + QI
                ELSE
                    QIN(ROW, COL - 1) = QIN(ROW, COL - 1) + QI
                    QT(ROW, COL - 1) = QT(ROW, COL - 1) + QI
                END IF
            END IF
        ELSE
            IF (COL < COLS) THEN
                RBED = MAX(AGRID(ROW, COL + 1), GRID(ROW, COL + 1)) + DR

                IF (BBED < RBED) THEN
                    QIN(ROW - 1, COL) = QIN(ROW - 1, COL) + QI
                    QT(ROW - 1, COL) = QT(ROW - 1, COL) + QI
                ELSE
                    QIN(ROW, COL + 1) = QIN(ROW, COL + 1) + QI
                    QT(ROW, COL + 1) = QT(ROW, COL + 1) + QI
                END IF
            ELSE
                QIN(ROW - 1, COL) = QIN(ROW - 1, COL) + QI
                QT(ROW - 1, COL) = QT(ROW - 1, COL) + QI
            END IF
        END IF
    END IF
ELSE
    IF (ROW < ROWS) THEN
        FBED = MAX(AGRID(ROW + 1, COL), GRID(ROW + 1, COL)) + DF
        IF (COL > 1) THEN
            LBED = MAX(AGRID(ROW, COL - 1), GRID(ROW, COL - 1)) + DL
            IF (COL < COLS) THEN
                RBED = MAX(AGRID(ROW, COL + 1), GRID(ROW, COL + 1)) + DR
                IF (RBED < MIN(FBED, LBED)) THEN
                    QIN(ROW, COL + 1) = QIN(ROW, COL + 1) + QI
                    QT(ROW, COL + 1) = QT(ROW, COL + 1) + QI
                ELSE
                    IF (LBED < MIN(FBED, RBED)) THEN
                        QIN(ROW, COL - 1) = QIN(ROW, COL - 1) + QI
                        QT(ROW, COL - 1) = QT(ROW, COL - 1) + QI
                    ELSE
                        QIN(ROW + 1, COL) = QIN(ROW + 1, COL) + QI
                        QT(ROW + 1, COL) = QT(ROW + 1, COL) + QI
                    END IF
                END IF
            ELSE
                IF (LBED < FBED) THEN
                    QIN(ROW, COL - 1) = QIN(ROW, COL - 1) + QI
```

```
                    QT(ROW, COL - 1) = QT(ROW, COL - 1) + QI
                ELSE
                    QIN(ROW + 1, COL) = QIN(ROW + 1, COL) + QI
                    QT(ROW + 1, COL) = QT(ROW + 1, COL) + QI
                END IF
            END IF
        ELSE
            IF (COL < COLS) THEN
                RBED = MAX(AGRID(ROW, COL + 1), GRID(ROW, COL + 1)) + DR
                IF (RBED < FBED) THEN
                    QIN(ROW, COL + 1) = QIN(ROW, COL + 1) + QI
                    QT(ROW, COL + 1) = QT(ROW, COL + 1) + QI
                ELSE
                    QIN(ROW + 1, COL) = QIN(ROW + 1, COL) + QI
                    QT(ROW + 1, COL) = QT(ROW + 1, COL) + QI
                END IF

            ELSE
                QIN(ROW + 1, COL) = QIN(ROW + 1, COL) + QI
                QT(ROW + 1, COL) = QT(ROW + 1, COL) + QI
            END IF
        END IF
    ELSE
        IF (COL > 1) THEN
            LBED = MAX(AGRID(ROW, COL - 1), GRID(ROW, COL - 1)) + DL
            IF (COL < COLS) THEN
                RBED = MAX(AGRID(ROW, COL + 1), GRID(ROW, COL + 1)) + DR
                IF (RBED < LBED) THEN
                    QIN(ROW, COL + 1) = QIN(ROW, COL + 1) + QI
                    QT(ROW, COL + 1) = QT(ROW, COL + 1) + QI
                ELSE
                    QIN(ROW, COL - 1) = QIN(ROW, COL - 1) + QI
                    QT(ROW, COL - 1) = QT(ROW, COL - 1) + QI
                END IF
            ELSE
                QIN(ROW, COL - 1) = QIN(ROW, COL - 1) + QI
                QT(ROW, COL - 1) = QT(ROW, COL - 1) + QI
            END IF
        ELSE
            IF (COL < COLS) THEN
                QIN(ROW, COL + 1) = QIN(ROW, COL + 1) + QI
                QT(ROW, COL + 1) = QT(ROW, COL + 1) + QI
            END IF
        END IF
    END IF
END IF


END SUBROUTINE

REAL (SELECTED_REAL_KIND(15,307)) FUNCTION AVSLP(S1, S2, S3)
!Function to calculate the average positive slope

REAL (SELECTED_REAL_KIND(15,307)), INTENT(IN) :: S1
REAL (SELECTED_REAL_KIND(15,307)), INTENT(IN) :: S2
REAL (SELECTED_REAL_KIND(15,307)), INTENT(IN) :: S3

INTEGER N

REAL (SELECTED_REAL_KIND(15,307)) ST, AS

ST = 0
N = 0

IF (S1 > 1E-12) THEN
```

```
    N = 1
    ST = S1
END IF

IF (S2 > 1E-12) THEN
    N = N + 1
    ST = ST + S2
END IF

IF (S3 > 1E-12) THEN
    N = N + 1
    ST = ST + S3
END IF

IF (N > 0) THEN
    AS = ST / REAL(N)
ELSE
    AS = -1
END IF

AVSLP = AS
END FUNCTION

REAL FUNCTION RMS(R1, R2)
!Function to calculate the route-mean-square of two numbers

REAL, INTENT(IN) :: R1
REAL, INTENT(IN) :: R2

RMS = SQRT(((R1 ** 2) + (R2 ** 2)) / 2)

END FUNCTION

REAL FUNCTION CBRT(NUM)
!Function to calculate the cube-root of a number

REAL, INTENT(IN) :: NUM

CBRT = NUM ** (1.0 / 3.0)
END FUNCTION


END MODULE
```

Module WAVES

Subroutines	Purpose	Page nr.
CALC_WAVES	Calculate wave height and period for each cell	325

```
MODULE WAVES

IMPLICIT NONE

REAL, PRIVATE, PARAMETER :: G = 9.80665
REAL, PRIVATE, PARAMETER :: PI = 3.14159265358979
REAL, PRIVATE, PARAMETER :: BI = 0.78       !Breaker index

CONTAINS

SUBROUTINE CALC_WAVES(ROWS, COLS, GRID_SIZE, DEPTH, MIN_H, WIND_SPD, &
    WIND_DIR, BDY_FETCH, WAVE_H, WAVE_T, WAVE_DIR, BM, WREF)

REAL,    DIMENSION(:,:), INTENT(IN)  :: DEPTH
REAL,    DIMENSION(:,:), INTENT(IN)  :: BM
REAL,    DIMENSION(:,:), INTENT(OUT) :: WAVE_H
REAL,    DIMENSION(:,:), INTENT(OUT) :: WAVE_T
REAL,    DIMENSION(:,:), INTENT(OUT) :: WAVE_DIR

REAL,        INTENT(IN) :: WIND_SPD
REAL,        INTENT(IN) :: WIND_DIR
REAL,        INTENT(IN) :: BDY_FETCH
REAL,        INTENT(IN) :: GRID_SIZE
REAL,        INTENT(IN) :: MIN_H
REAL,        INTENT(IN) :: WREF

INTEGER,     INTENT(IN) :: ROWS
INTEGER,     INTENT(IN) :: COLS

REAL,    DIMENSION(:,:), ALLOCATABLE :: FETCH
REAL,    DIMENSION(:,:), ALLOCATABLE :: DIR_TOT
REAL,    DIMENSION(:,:), ALLOCATABLE :: DMG_TOT
REAL,    DIMENSION(:,:), ALLOCATABLE :: OS_TOT

REAL,    DIMENSION(7) :: FETCH2
REAL,    DIMENSION(7) :: DIR_TOT1
REAL,    DIMENSION(7) :: DMG_TOT1
REAL,    DIMENSION(7) :: OS_TOT1

REAL,    DIMENSION(4) :: F
REAL,    DIMENSION(7) :: DOS

REAL DF, CD, UF, AV_OS, RES_OS, OFFSET, RES, FRC
REAL H_MAX, MAX_FETCH, WAVE_LEN, WAVE_C, OS_MINH
REAL MAXH, H_MAX1, MAX_FETCH1, WIND_DIR1, FETCH1
REAL DD, DG, RES_OS1, ADD_OS, ADJ, MIN_D, MAX_D
REAL FRM, WAVEH1, WAVEH2, WAVET1, WAVET2

INTEGER ROW, COL, ROW1, COL1, N, RST, CST, DROW, DCOL, R
INTEGER FR, FC, LR, LC, D, OS, D1

LOGICAL BY_ROWS

ALLOCATE (FETCH(ROWS, COLS))
```

```
ALLOCATE (DIR_TOT(ROWS, COLS))
ALLOCATE (DMG_TOT(ROWS, COLS))
ALLOCATE (OS_TOT(ROWS, COLS))


!WIND_DIR1 = PI * WIND_DIR / 180
WIND_DIR1 = WIND_DIR
WAVE_DIR = WIND_DIR1

IF (COS(WIND_DIR1) > 0.70710678) THEN
    BY_ROWS = .TRUE.
    RST = 1
    DF = GRID_SIZE / COS(WIND_DIR1)
    FR = 1
    LR = ROWS
    DG = 0
ELSE
    IF (COS(WIND_DIR1) < -0.70710678) THEN
        BY_ROWS = .TRUE.
        RST = -1
        DF = -GRID_SIZE / COS(WIND_DIR1)
        FR = ROWS
        LR = 1
        DG = PI
    ELSE
        BY_ROWS = .FALSE.
        IF (SIN(WIND_DIR1) > 0) THEN
            CST = 1
            DF = GRID_SIZE / SIN(WIND_DIR1)
            FC = 1
            LC = COLS
            DG = PI / 2
        ELSE
            CST = -1
            DF = -GRID_SIZE / SIN(WIND_DIR1)
            FC = COLS
            LC = 1
            DG = 3 * PI / 2
        END IF
    END IF
END IF

FETCH = 0
IF (BY_ROWS) THEN
    IF (RST == 1) THEN
        FETCH(1,:) = 0
        ADJ = 0.0
     ELSE
        FETCH(ROWS,:) = BDY_FETCH
        ADJ = PI
    END IF
ELSE
    IF (CST == 1) THEN
        FETCH(:, 1) = BDY_FETCH
        ADJ = 0
    ELSE
        FETCH(:, COLS) = BDY_FETCH
        ADJ = PI
    END IF
END IF

CD = 0.001 * (1.1 + (0.035 * WIND_SPD))     !Drag coefficient
UF = CD * (WIND_SPD**2)                     !Friction velocity squared

!Fractions based on subtended angles of 4 of 7 cells
```

```
!(last 3 same as 1st 3)
F(1) = 0.358722
F(2) = 0.200831
F(3) = 0.080270
F(4) = 0.039538

IF (BY_ROWS) THEN

    IF (WIND_DIR1 > 3 * PI / 2) WIND_DIR1 = WIND_DIR1 - (2 * PI)

    AV_OS = 0
    DO ROW = FR, LR, RST

        OS = 0

        !Offset is applied in
        IF (AV_OS > 0.999) THEN
            AV_OS = AV_OS - 1.0
            FETCH(ROW, 1) = FETCH(ROW, 1) + BDY_FETCH
            OS = 1
        ELSE
            IF (AV_OS < -0.999) THEN
                AV_OS = AV_OS + 1.0
                FETCH(ROW, COLS) = FETCH(ROW, COLS) + BDY_FETCH
                OS = -1
            END IF
        END IF
        AV_OS = AV_OS + (TAN(WIND_DIR1) * RST)

        DO COL = 1, COLS

            !Fetch increased one cell
            FETCH(ROW, COL) = FETCH(ROW, COL) + DF

            !Fetch limited to value that gives depth limited wave height
            H_MAX = BI * DEPTH(ROW, COL)
            MAX_FETCH = (UF / G) * (((G * H_MAX) / (0.0413 * UF))**2.0)
            IF (FETCH(ROW, COL) > MAX_FETCH) FETCH(ROW, COL) = MAX_FETCH

            WAVEH1 = (UF * 0.0413 / G) * ((G * FETCH(ROW, COL) / UF)**0.5)
            WAVET1 = (SQRT(UF) * 0.751 / G) * &
                ((G * FETCH(ROW, COL) / UF)**0.33333333)

            !Fetch reduction due to marsh attenuation
            FRM = ((1 - (0.03 * BM(ROW, COL))) ** GRID_SIZE) ** 2.0
            FETCH(ROW, COL) = FETCH(ROW, COL) * FRM

            WAVEH2 = (UF * 0.0413 / G) * ((G * FETCH(ROW, COL) / UF)**0.5)
            WAVET2 = (SQRT(UF) * 0.751 / G) * &
                ((G * FETCH(ROW, COL) / UF)**0.33333333)

            WAVE_H(ROW, COL) = (WAVEH1 + WAVEH2) / 2.0
            WAVE_T(ROW, COL) = (WAVET1 + WAVET2) / 2.0

            !Check for cells below min depth
            !(wave assumed to be reflected from channel edges)
            !MIN_D is the lowest column number fetch
            !can be ditributed to - 1
            !MAX_D is the highest column number fetch
            !can be distributed to + 1
            MIN_D = 0
            MAX_D = COLS + 1
            DO D = COL - 3 + OS, COL - 1 + OS
                IF (D > 0) THEN
                    IF (DEPTH(ROW, D) < MIN_H) THEN
```

```
                    MIN_D = D
                END IF
            END IF
        END DO
        DO D = COL + 3 + OS, COL + 1 + OS, -1
            IF (D <= COLS) THEN
                IF (DEPTH(ROW, D) < MIN_H) THEN
                    MAX_D = D
                END IF
            END IF
        END DO

        IF (ROW /= LR) THEN

            FETCH1 = FETCH(ROW, COL)

            !Calculate fetch distributed to 7 cells in next row
            !WREF is the proportion of the fetch reflected
            DO D = - 3, + 3

                FRC = F(ABS(D) + 1)
                DCOL = COL + D + OS
                IF (DCOL <= MIN_D) THEN
                    DCOL = (2 * MIN_D) - DCOL + 1
                    FRC = FRC * WREF

                    IF (DCOL >= MAX_D) THEN
                        DCOL = (2 * MAX_D) - DCOL - 1
                        FRC = FRC * WREF
                        IF (DCOL <= MIN_D) THEN
                            DCOL = (2 * MIN_D) - DCOL + 1
                            FRC = FRC * WREF
                        END IF
                    END IF
                ELSE
                    IF (DCOL >= MAX_D) THEN
                        DCOL = (2 * MAX_D) - DCOL - 1
                        FRC = FRC * WREF
                        IF (DCOL <= MIN_D) THEN
                            DCOL = (2 * MIN_D) - DCOL + 1
                            FRC = FRC * WREF
                            IF (DCOL >= MAX_D) THEN
                                DCOL = (2 * MAX_D) - DCOL - 1
                                FRC = FRC * WREF
                            END IF
                        END IF
                    END IF
                END IF

                !Calculate fetch distributed to cells in next row
                IF ((DCOL > 0) .AND. (DCOL <= COLS)) THEN
                    FETCH(ROW + RST, DCOL) = FETCH(ROW + RST, DCOL) &
                        + (FETCH1 * FRC)
                END IF

            END DO

        END IF

    END DO
END DO

ELSE

    !If wind angle between 45 & 135 degrees or between 225 & 315 degrees
```

```
!Calculate fetch column by column
IF (WIND_DIR1 < 0) WIND_DIR1 = WIND_DIR1 - (2 * PI)
AV_OS = 0

DO COL = FC, LC, CST

    OS = 0
    IF (AV_OS > 0.999) THEN
        AV_OS = AV_OS - 1.0
        OS = 1
        FETCH(1, COL) = FETCH(1, COL) + BDY_FETCH
    ELSE
        IF (AV_OS < -0.999) THEN
            AV_OS = AV_OS + 1.0
            FETCH(ROWS, COL) = FETCH(ROWS, COL) + BDY_FETCH
            OS = -1
        END IF
    END IF
    AV_OS = AV_OS - (TAN(WIND_DIR1 - (PI / 2)) * CST)

    DO ROW = 1, ROWS


        !Fetch increased one cell
        FETCH(ROW, COL) = FETCH(ROW, COL) + DF

        !Fetch limited to value that gives depth limited wave height
        H_MAX = BI * DEPTH(ROW, COL)
        MAX_FETCH = (UF / G) * (((G * H_MAX) / (0.0413 * UF))**2.0)
        IF (FETCH(ROW, COL) > MAX_FETCH) FETCH(ROW, COL) = MAX_FETCH

        WAVEH1 = (UF * 0.0413 / G) * ((G * FETCH(ROW, COL) / UF)**0.5)
        WAVET1 = (SQRT(UF) * 0.751 / G) * &
            ((G * FETCH(ROW, COL) / UF)**0.33333333)

        !Fetch reduction due to marsh attenuation
        FRM = ((1 - (0.03 * BM(ROW, COL))) ** GRID_SIZE) ** 2.0
        FETCH(ROW, COL) = FETCH(ROW, COL) * FRM

        WAVEH2 = (UF * 0.0413 / G) * ((G * FETCH(ROW, COL) / UF)**0.5)
        WAVET2 = (SQRT(UF) * 0.751 / G) * &
            ((G * FETCH(ROW, COL) / UF)**0.33333333)

        WAVE_H(ROW, COL) = (WAVEH1 + WAVEH2) / 2.0
        WAVE_T(ROW, COL) = (WAVET1 + WAVET2) / 2.0

        !Check for cells below min depth
        !(wave assumed to be reflected from channel edges)
        MIN_D = 0
        MAX_D = ROWS + 1
        DO D = ROW - 3 + OS, ROW - 1 + OS
            IF (D > 0) THEN
                IF (DEPTH(D, COL) < MIN_H) THEN
                    MIN_D = D
                END IF
            END IF
        END DO
        DO D = ROW + 3 + OS, ROW + 1 + OS, -1
            IF (D <= ROWS) THEN
                IF (DEPTH(D, COL) < MIN_H) THEN
                    MAX_D = D
                END IF
            END IF
        END DO
```

```
            IF (COL /= LC) THEN

                FETCH1 = FETCH(ROW, COL)

                DO D = - 3, + 3

                    FRC = F(ABS(D) + 1)
                    DROW = ROW + D + OS
                    IF (DROW <= MIN_D) THEN
                        DROW = (2 * MIN_D) - DROW + 1
                        FRC = FRC * WREF
                        IF (DROW >= MAX_D) THEN
                            DROW = (2 * MAX_D) - DROW - 1
                            FRC = FRC * WREF
                            IF (DROW <= MIN_D) THEN
                                DROW = (2 * MIN_D) - DROW + 1
                                FRC = FRC * WREF
                            END IF
                        END IF
                    ELSE
                        IF (DROW >= MAX_D) THEN
                            DROW = (2 * MAX_D) - DROW - 1
                            FRC = FRC * WREF
                            IF (DROW <= MIN_D) THEN
                                DROW = (2 * MIN_D) - DROW + 1
                                FRC = FRC * WREF
                                IF (DROW >= MAX_D) THEN
                                    DROW = (2 * MAX_D) - DROW - 1
                                    FRC = FRC * WREF
                                END IF
                            END IF
                        END IF
                    END IF

                    IF ((DROW > 0) .AND. (DROW <= ROWS)) THEN
                        FETCH(DROW, COL + CST) = FETCH(DROW, COL + CST) &
                            + (FETCH1 * FRC)
                    END IF
                END DO


            END IF

        END DO
    END DO
END IF

END SUBROUTINE

END MODULE
```

Module SAND_MUD

Subroutines	Purpose	Page nr.
MUD_ER	Calculate erosion of mud	331
CALC_MUDST_SF	Calculate sediment transport and update bed levels	334
Functions		
SAND_TR	Calculate single fraction sand transport rate	333
CALC_TMX	Calculate bed shear stress	351

```
MODULE SAND_MUD

IMPLICIT NONE

!Standard value for gravitational acceleration
REAL, PRIVATE, PARAMETER :: G = 9.80665
!Density of sea water (at salinity = 35ppt, temperature = 10 degrees C)
REAL, PRIVATE, PARAMETER :: DWA = 1027
!Kinematic viscosity of sea water (temperature/salinity as above)
REAL, PRIVATE, PARAMETER :: KV = 1.36E-6
REAL, PRIVATE, PARAMETER :: KPA = 0.4    !Von Karmans constant
REAL, PRIVATE, PARAMETER :: PII = 3.14159265358979

CONTAINS

!Subroutine to calculate erosion depth of mud
SUBROUTINE MUD_ER(U, HS, TP, WCD, H, PMUD, DMUD, TEMUD, TEMAX, TESAND, &
                  BM, TSS, EM, TMAX, TECR, MF)

REAL, INTENT(IN) :: U       !Depth averaged current velocity
REAL, INTENT(IN) :: HS      !Significant wave height
REAL, INTENT(IN) :: TP      !Peak spectral wave period
REAL, INTENT(IN) :: WCD     !Wave / current direction
REAL, INTENT(IN) :: H       !Water depth
REAL, INTENT(IN) :: PMUD    !Fraction of mud in bed
REAL, INTENT(IN) :: DMUD    !Bulk density of mud
REAL, INTENT(IN) :: TEMUD   !Critical shear stress for erosion of mud
REAL, INTENT(IN) :: TEMAX   !Critical shear stress for erosion of 20% mud
REAL, INTENT(IN) :: TESAND  !Critical shear stress for erosion of sand
REAL, INTENT(INOUT) :: BM   !Salt marsh biomass as fraction of max
REAL, INTENT(IN) :: TSS     !Model timestep in seconds

REAL (SELECTED_REAL_KIND(15,307)), INTENT(OUT) :: EM     !Erosion

REAL, INTENT(OUT) :: TMAX   !Bed shear stress
REAL, INTENT(OUT) :: TECR   !Critical bed shear stress

LOGICAL, INTENT(IN) :: MF   !True if called from multi-fraction module

REAL UST         !Shear velocity
REAL TC          !Bed shear stress due to current
REAL TW          !Bed shear stress due to waves
REAL WAVE_L      !Wave length
REAL UD, AD      !Peak wave orbital velocity and excursion
REAL HRMS        !Root-mean-square wave height
!Cycle-mean bed shear stress for smooth beds
!(Whitehouse et. al. 'Dynamics of Marine Muds' pg 55
```

```
REAL TME
!Bed shear stress adjustment based on salt-marsh biomass present
REAL TCFSM
REAL E1          !Calculated erosion rate (kg/m2/s)
REAL ME   !Erosion constant (Whitehouse et. al. Fig 16/Table 8, Pg.69)
REAL RW          !Wave Reynolds number
REAL N1, B1      !Friction factor parameters
REAL FWS         !Smooth bed friction factor
REAL TECR_SM     !Critical bed shear stress including effect of salt marsh
REAL T !Difference between bed shear stress and critical bed shear stress
!Parameters used in marsh destruction model
!(shear stress ratio / timestep in hours)
REAL T1, R

INTEGER N !Counter

ME = 0.002
UST = 1
DO N = 1, 6
    !Whitehouse et. al. 'Dynamics of Marine Muds' Eqn 3.6
    !Taking velocity at 0.37h = depth averaged velocity
    UST = U / (5.5 + 2.5 * LOG(UST * 0.37 * H / KV))
END DO
TC = DWA * (UST ** 2)

!wavelength
!Using approximate solution (+-10%) from CEM pt. 2 Chap. 4
IF (TP > 1) THEN
    WAVE_L = (G * (TP ** 2) / (2 * PII)) * ((TANH(4 * (PII ** 2) * &
             H / ((TP ** 2) * G))) ** 0.5)
    HRMS = HS / 1.414
    IF (H / WAVE_L < 2.0) THEN
        UD = PII * HRMS / (TP * SINH(2 * PII * H / WAVE_L))
        AD = 0.5 * UD * TP / PII
        RW = UD * AD / KV
        IF (RW <= 5E5) THEN
            B1 = 2
            N1 = 0.5
        ELSE
            B1 = 0.0521
            N1 = 0.187
        END IF
        IF (RW > 1E-3) THEN
            FWS = B1 * (RW ** -N1)
            TW = 0.5 * DWA * FWS * (UD ** 2)
        ELSE
            TW = 0
        END IF
    ELSE
        TW = 0
    END IF
ELSE
    TW = 0
END IF

TME = TC * (1 + 9 * ((TW / (TC + TW)) ** 9)) !Whitehouse et. al. Eqn 3.28
TMAX = SQRT(((TME + TW * COS(WCD)) ** 2) + ((TW * SIN(WCD)) ** 2)) !Eq3.26

IF (MF == .FALSE.) THEN
    TCFSM = 1.0 + (4.0 * BM)

!Interpolate critical bed shear stress based on Whitehouse et. al. Fig 31
    IF (PMUD > 0.2) THEN
        TECR = 1.25 * ((TEMUD * (PMUD - 0.2)) + (TEMAX * (1 - PMUD)))
    ELSE
```

```
        TECR = 5.0 * (((0.2 - PMUD) * TESAND) + (PMUD * TEMAX))
    END IF

    TECR_SM = TCFSM * TECR
ELSE
    !Multi-fraction module - marsh effect already included
    TECR_SM = TEMUD
END IF

IF (TMAX > TECR_SM) THEN
    T = (TMAX - TECR_SM)
    E1 = ME * T
    IF ((PMUD > 0.15) .OR. (MF == .FALSE.)) THEN
        EM = E1 * TSS / DMUD
    ELSE
        EM = 10.0 * (PMUD - 0.05) * E1 * TSS / DMUD
    END IF
    !Marsh destruction model
    T1 = T / TECR_SM
    R = TSS / 3600
    BM = BM / ((T1 + 1) ** R)
ELSE
    EM = 0
END IF


END SUBROUTINE

REAL (SELECTED_REAL_KIND(15,307)) FUNCTION SAND_TR(U, HS, TP, H, DSTR, &
                                                   D50, D90, DS, PMUD, BM)

REAL, INTENT(IN) :: U       !Depth averaged current velocity
REAL, INTENT(IN) :: HS      !Significant wave height
REAL, INTENT(IN) :: TP      !Peak spectral wave period
REAL, INTENT(IN) :: H       !Water depth
REAL, INTENT(IN) :: DSTR    !Dimensionless particle size
REAL, INTENT(IN) :: D50     !Median particle size for sand
REAL, INTENT(IN) :: D90     !90 percentile particle size
REAL, INTENT(IN) :: DS      !Density of sand particles
REAL, INTENT(IN) :: PMUD    !Percentage mud in active layer
REAL, INTENT(IN) :: BM      !Biomass (fraction of maximum)

REAL DSAND      !Minimum sand particle size
REAL GAMMA, Y, KH
REAL S          !Relative density for sand particles
REAL FSILT      !Silt factor
REAL UCRC, UCRW !Critical depth averaged velocity for currents / waves
REAL UW         !PEak wave oribital velocity
!Critical bed shear stress adjustement based on salt-marsh biomass present
REAL TCFSM
REAL UE         !Effective velocity
REAL QB, QS     !Bed load / suspended load
REAL DSTR1      !Dimensionless particle size
REAL ALPHA, UCR, ME


DSAND = 0.000062
GAMMA = 0.4
S = DS / DWA
IF (D50 > DSAND) THEN
    FSILT = 1.0
ELSE
    FSILT = DSAND / D50
END IF
```

```
IF (DSTR < 1.0) THEN
    DSTR1 = D50 * (((S - 1) * G / (KV ** 2)) ** (1.0 / 3.0))
ELSE
    DSTR1 = DSTR
END IF

IF (D50 < 0.0005) THEN
    UCRC = 0.19 * ((1 + PMUD) ** 1.5) * (D50 ** 0.1) * LOG10(12 * H / &
          (3 * D90))
    UCRW = 0.24 * ((1 + PMUD) ** 1.5) * (((S - 1) * G) ** 0.66) * &
          (D50 ** 0.33) * (TP ** 0.33)
ELSE
    UCRC = 8.5 * ((1 + PMUD) ** 1.5) * (D50 ** 0.6) * &
           LOG10(12 * H / (3 * D90))
    UCRW = 0.95 * ((1 + PMUD) ** 1.5) * (((S - 1) * G) ** 0.57) * &
          (D50 ** 0.43) * (TP ** 0.14)
END IF


Y = 4.02 * H / (TP ** 2)
KH = SQRT(Y) * (1.0 + (0.166 * Y) + (0.031 * (Y ** 2)))
IF (KH < 10) THEN
    UW = PII * HS / (TP * SINH(KH))
ELSE
    UW = 0
END IF

IF (U > 1E-3) THEN
    ALPHA = U / (U + UW)
ELSE
    ALPHA = 0
END IF
UCR = (ALPHA * UCRC) + ((1 - ALPHA) * UCRW)


!Critical shear stress adjustment for Salt marsh
TCFSM = 1.0 + (BM * 4)
!Critical velocity adjustment - taking shear stress as
!proportional to velocity squared
UCR = UCR * SQRT(TCFSM)

UE = U + (GAMMA * UW)

IF (UE > UCR) THEN
    ME = (UE - UCR) / SQRT((S - 1) * G * D50)
    QB = 0.015 * (1 - PMUD) * FSILT * U * H * (ME ** 1.5) * &
        ((D50 / H) ** 1.2)
    QS = 0.012 * (1 - PMUD) * FSILT * U * H * (ME ** 2.4) * (D50 / H) * &
        (DSTR1 ** -0.6)
ELSE
    QB = 0
    QS = 0
END IF
SAND_TR = QB + QS


END FUNCTION


!-------------------------------------------------------------------------
!Subroutine to calculate morphological updates including sand transport, &
!mud erosion / deposition and sand-mud mixtures
SUBROUTINE CALC_MUDST_SF(GRID, FGRID, ALYR, SED, ALYR_TH, LYR_TH, &
           NR_LAYERS, BASE, ROWS, COLS, TSS, TSPRP, GRID_SIZE, QX, QY, &
           H, MEAN_H, H_MIN, D50I, D90I, DS, PS, KD, DMUD, TCRF, WAVEH, &
```

```
            WAVET, WAVE_DIR, BM, BM_MAX, DSSC, DS_SAND_INPUT, VX1, VY1, &
            MAX_V, LTFR, TE, TD, SC, NWS2, P, DTOT, ETOT, SAND_IN, &
            SAND_OUT, MUD_IN, MUD_OUT)

!Array to to store bed level data
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:),  INTENT(INOUT) :: GRID
!Active layer composition
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:,:),INTENT(INOUT) :: ALYR
!Sublayer composition
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:,:),INTENT(INOUT) :: SED
!Suspended sediment
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:),  INTENT(INOUT) :: SC
!Mean water depth for timestep
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:),INTENT(IN)    :: MEAN_H
!Cumulative deposition
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:),  INTENT(INOUT) :: DTOT
!Cumulative erosion
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:),  INTENT(INOUT) :: ETOT

REAL, DIMENSION(:,:), INTENT(IN) :: FGRID !non-erodable bed levels
REAL, DIMENSION(:,:), INTENT(IN) :: QX    !x dir. flow at each cell
REAL, DIMENSION(:,:), INTENT(IN) :: QY    !y dir. flow at each cell

REAL, DIMENSION(:,:), INTENT(OUT) :: VX1  !x dir velcoity
REAL, DIMENSION(:,:), INTENT(OUT) :: VY1  !y dir velocity

REAL, DIMENSION(:,:), INTENT(IN)  :: H    !Water depth at each cell

REAL, DIMENSION(:,:,:), INTENT(IN)    :: WAVEH      !Wave height
REAL, DIMENSION(:,:,:), INTENT(IN)    :: WAVET      !Wave period
REAL, DIMENSION(:,:,:), INTENT(IN)    :: WAVE_DIR   !Wave direction
REAL, DIMENSION(:,:),   INTENT(INOUT) :: BM !Biomass as fr. of max biomass
REAL, DIMENSION(:),     INTENT(IN)    :: P  !Probability

!Cumulative sand input at the boundary
REAL (SELECTED_REAL_KIND(15,307)), INTENT(INOUT) :: SAND_IN
!Cumulative sand output at the boundary
REAL (SELECTED_REAL_KIND(15,307)), INTENT(INOUT) :: SAND_OUT
!Cumulative mud input at the boundary
REAL (SELECTED_REAL_KIND(15,307)), INTENT(INOUT) :: MUD_IN
!Cumulative mud output at the boundary
REAL (SELECTED_REAL_KIND(15,307)), INTENT(INOUT) :: MUD_OUT
!Active layer default thickness
REAL (SELECTED_REAL_KIND(15,307)), INTENT(IN) :: ALYR_TH
!Sub-layer thickness
REAL (SELECTED_REAL_KIND(15,307)), INTENT(IN) :: LYR_TH
!Minimum level for sub-layers
REAL (SELECTED_REAL_KIND(15,307)), INTENT(IN) :: BASE

REAL, INTENT(IN) :: DS              !Sand particle density
REAL, INTENT(IN) :: PS              !Sand porosity
REAL, INTENT(IN) :: D50I            !Median particle size for sand
REAL, INTENT(IN) :: D90I            !90 percentile particle size for sand
REAL, INTENT(IN) :: KD              !Diffusion coefficient
REAL, INTENT(IN) :: DMUD            !Bulk density for mud
REAL, INTENT(IN) :: TSS             !Timestep (s)
REAL, INTENT(IN) :: GRID_SIZE       !Grid cell size
!Sand input at downstream boundary (m3/s)
REAL, INTENT(IN) :: DS_SAND_INPUT
REAL, INTENT(IN) :: MAX_V  !Maximum velocity limit (model parameter ~2m/s)
REAL, INTENT(IN) :: LTFR   !Lateral transport scaling factor
REAL, INTENT(IN) :: TE     !Critical shear stress for erosion of mud
REAL, INTENT(IN) :: TD     !Critical shear stress for deposition of mud
REAL, INTENT(IN) :: BM_MAX !Maximum biomass for saltmarsh
```

```
REAL, INTENT(IN) :: DSSC  !Volume conc. of mud at dsbdy
REAL, INTENT(IN) :: TCRF  !Shear stress factor at 20% mud
REAL, INTENT(IN) :: H_MIN !Minimum water depth

INTEGER, INTENT(IN) :: ROWS         !Number of CA rows
INTEGER, INTENT(IN) :: COLS         !Number of CA columns
INTEGER, INTENT(IN) :: NR_LAYERS    !Number of sub-layers
INTEGER, INTENT(IN) :: TSPRP !Nr of timesteps in a tide/spring/neap cycle
INTEGER, INTENT(IN) :: NWS2         !Number of wave fields

REAL, DIMENSION(:,:), ALLOCATABLE :: VX        !x dir velocity
REAL, DIMENSION(:,:), ALLOCATABLE :: VY        !y dir velocity

REAL,   DIMENSION(:,:),     ALLOCATABLE :: SGRID !Init bed this timestep
REAL,   DIMENSION(:,:),     ALLOCATABLE :: CV   !Velocity magnitude
REAL,   DIMENSION(:,:),     ALLOCATABLE :: CVX  !Unsigned(+ve) x dir vel.
REAL,   DIMENSION(:,:),     ALLOCATABLE :: CVY  !Unsigned(+ve) y dir vel.

!Maximum bed shear stress under combined currents and waves
REAL,   DIMENSION(:,:,:),   ALLOCATABLE :: TMX
!Active layer composition (temp)
REAL,   DIMENSION(:,:,:),   ALLOCATABLE :: ALYR1
!Mud proportion in active layer
REAL,   DIMENSION(:,:),     ALLOCATABLE :: PMUD
!Critical bed shear stress
REAL,   DIMENSION(:,:),     ALLOCATABLE :: TECR
REAL,   DIMENSION(:),       ALLOCATABLE :: TMXTOTR
REAL,   DIMENSION(:),       ALLOCATABLE :: QTC

REAL (SELECTED_REAL_KIND(15,307)) :: QTOT       !Total flow in bdy row
REAL (SELECTED_REAL_KIND(15,307)) :: ALYR_TOT   !Active layer total depth
REAL (SELECTED_REAL_KIND(15,307)) :: DL         !Bed level change
REAL (SELECTED_REAL_KIND(15,307)) :: SUB_LYR_FR !Thickness of top sub-lyr
REAL (SELECTED_REAL_KIND(15,307)) :: S_LYRS     !Nr sub-lyrs below a.lyr
REAL (SELECTED_REAL_KIND(15,307)) :: STH        !Sed depth below a.lyr
REAL (SELECTED_REAL_KIND(15,307)) :: FR         !Fraction
REAL (SELECTED_REAL_KIND(15,307)) :: QT         !Tot sand trprt. (Bl+Susp)
REAL (SELECTED_REAL_KIND(15,307)) :: QTN    !Sand transport (wave field n)
REAL (SELECTED_REAL_KIND(15,307)) :: MAXE       !Maximum erosion depth
REAL (SELECTED_REAL_KIND(15,307)) :: QM_SAND    !Depth of transported sand

!Bed level differences between neighbouring cells and max of these values
REAL (SELECTED_REAL_KIND(15,307)) BDF, BDB, BDL, BDR, BDMAX
!Lateral erosion in each direction and total lateral erosion
REAL (SELECTED_REAL_KIND(15,307)) ELATF, ELATB, ELATL, ELATR, ELATT
!Lateral sand transport in each of four grid directions + total
REAL (SELECTED_REAL_KIND(15,307)) QMLATF, QMLATB, QMLATL, QMLATR, QMLATT
!Total lateral sediment transport, reduction factors for lateral transport
REAL (SELECTED_REAL_KIND(15,307)) LATT, LATR, QMR, ELR
!Depth of sand transported in x direction / y direction
REAL (SELECTED_REAL_KIND(15,307)) QMX, QMY
!Maximum erosion, total erosion
REAL (SELECTED_REAL_KIND(15,307)) MAXER, ERTOT
!Erosion reduction factor, lateral mud transport
REAL (SELECTED_REAL_KIND(15,307)) ERFR, EL1, ELN
!Mud erosion depths
REAL (SELECTED_REAL_KIND(15,307)) EM, EM1, EM2, EMN
!Fractions of advection in x direction / y direction
REAL (SELECTED_REAL_KIND(15,307)) FRX, FRY
!Depth ratios
REAL (SELECTED_REAL_KIND(15,307)) HRB, HRF, HRL, HRR
!Fr. of suspended sediment transported by diffusion/fr. not transported
REAL (SELECTED_REAL_KIND(15,307)) FD, FD1
!Sum of bed shear stress values in boundary row
REAL (SELECTED_REAL_KIND(15,307)) TMX_TOT
```

```
!Depth of sediment transported by diffusion in 4 directions
!(equivalent depth of settled bed)
REAL (SELECTED_REAL_KIND(15,307)) DDF, DDB, DDL, DDR
REAL (SELECTED_REAL_KIND(15,307)) DEP1, DEP2, DEPN
REAL (SELECTED_REAL_KIND(15,307)) QTTOT

REAL CELL_AREA  !Plan area of one cell
REAL NL         !Number of sub-layers to be updated
REAL MIN_V      !Minimum velocity
REAL HS, TP     !Wave height, wave period
REAL DLWH       !depth limited wave height (ratio to water depth)
REAL NST, HST, DST !Salt marsh - nr. stalks, stalk ht, stalk dia
REAL KB         !Organogenic sediment production rate
REAL CURR_DIR   !Current direction
REAL WC_DIR     !angle between current and wave directions
REAL ME         !Erosion constant for mud
REAL BMS, SED_TRP, FRSM, FRD !Salt marsh parameters
REAL WS                      !Particle fall velcoity (mud)
REAL TCR_MAX, TCR_SAND !Critical shear stress at 20% mud / for sand only
REAL D50, D90   !D50 (median) and D90 particle sizes in metres
REAL DSTR       !dimensionless particle size parameter
REAL S_SAND     !Relative density for sand
REAL SPCR       !Critical shields parameter for sand
!Max. proportion of suspended sediment
!transported between adjacent by diffusion in one timestep
REAL FDMAX
!Fraction of sand transport in x direction / y direction
REAL ALYR_FR1, ALYR_FR2
REAL TECRN

REAL :: NULL = 0 !Dummy input to subroutine
REAL :: MT = 0   !Time from last organogenic sediment addition (seconds)

INTEGER ROW, COL                    !Row, column number
INTEGER N, M                        !Counters
INTEGER SUB_LYR_NR                  !Sub-layer number


ALLOCATE (SGRID(ROWS, COLS))
ALLOCATE (CV(ROWS, COLS))
ALLOCATE (CVX(ROWS, COLS))
ALLOCATE (CVY(ROWS, COLS))
ALLOCATE (VX(ROWS, COLS))
ALLOCATE (VY(ROWS, COLS))
ALLOCATE (TMX(ROWS, COLS, NWS2))
ALLOCATE (ALYR1(ROWS, COLS, 2))
ALLOCATE (PMUD(ROWS, COLS))
ALLOCATE (TECR(ROWS, COLS))
ALLOCATE (TMXTOTR(COLS))
ALLOCATE (QTC(COLS))

D50 = D50I * 0.000001
D90 = D90I * 0.000001
ALYR1 = 0
S_SAND = DS / DWA
DSTR = ((G * (S_SAND - 1) / (KV ** 2)) ** (1.0 / 3.0)) * D50
SPCR = (0.3 / (1 + (1.2 * DSTR))) + (0.055 * (1 - EXP(-0.02 * DSTR)))
TCR_SAND = SPCR * G * (DS - DWA) * D50
TCR_MAX = TCR_SAND * TCRF

KB = 0.002
DLWH = 0.78
SGRID = GRID
MIN_V = 0.001
CELL_AREA = GRID_SIZE * GRID_SIZE
```

```
WS = 1.3E-5
ME = 0.001
FDMAX = 0.2

!Calculate velocity for each cell
!based on positive outflow values at cell boundary
DO ROW = 1, ROWS
    DO COL = 1, COLS

        PMUD(ROW, COL) = ALYR(ROW, COL, 2) / ALYR_TH

        IF (H(ROW, COL) > H_MIN) THEN
            IF (COL == 1) THEN
                VX(ROW, COL) = QX(ROW, COL) / (2.0 * H(ROW, COL) * &
                               GRID_SIZE)
            ELSE
                IF (QX(ROW, COL) > 0) THEN
                    IF (QX(ROW, COL - 1) < 0) THEN
                        IF (ABS(QX(ROW, COL)) > ABS(QX(ROW, COL - 1))) &
                              THEN
                            VX(ROW, COL) = QX(ROW, COL) / &
                              (H(ROW, COL) * GRID_SIZE)
                        ELSE
                            VX(ROW, COL) = QX(ROW, COL - 1) / &
                              (H(ROW, COL) * GRID_SIZE)
                        END IF
                    ELSE
                        VX(ROW, COL) = QX(ROW, COL) / &
                          (H(ROW, COL) * GRID_SIZE)
                    END IF
                ELSE
                    IF (QX(ROW, COL - 1) < 0) THEN
                        VX(ROW, COL) = QX(ROW, COL - 1) / &
                          (H(ROW, COL) * GRID_SIZE)
                    ELSE
                        VX(ROW, COL) = 0
                    END IF
                END IF
            END IF

            IF (ROW == 1) THEN
                VY(ROW, COL) = QY(ROW, COL) / (H(ROW, COL) * GRID_SIZE)
            ELSE
                IF (QY(ROW, COL) > 0) THEN
                    IF (QY(ROW - 1, COL) < 0) THEN
                        IF (ABS(QY(ROW - 1, COL)) > ABS(QY(ROW, COL))) &
                              THEN
                            VY(ROW, COL) = QY(ROW - 1, COL) / &
                              (H(ROW, COL) * GRID_SIZE)
                        ELSE
                            VY(ROW, COL) = QY(ROW, COL) / &
                              (H(ROW, COL) * GRID_SIZE)
                        END IF
                    ELSE
                        VY(ROW, COL) = QY(ROW, COL) / &
                          (H(ROW, COL) * GRID_SIZE)
                    END IF
                ELSE
                    IF (QY(ROW - 1, COL) < 0) THEN
                        VY(ROW, COL) = QY(ROW - 1, COL) / &
                          (H(ROW, COL) * GRID_SIZE)
                    ELSE
                        VY(ROW, COL) = 0
                    END IF
                END IF
```

```
            END IF

            CV(ROW, COL) = SQRT((VX(ROW, COL) ** 2) + (VY(ROW, COL) ** 2))
            CVX(ROW, COL) = ABS(VX(ROW, COL))
            CVY(ROW, COL) = ABS(VY(ROW, COL))
        ELSE
            VX(ROW, COL) = 0
            VY(ROW, COL) = 0
            CV(ROW, COL) = 0
            CVX(ROW, COL) = 0
            CVY(ROW, COL) = 0
        END IF
        IF (CV(ROW, COL) > MAX_V) THEN
            VX(ROW, COL) = MAX_V * VX(ROW, COL) / CV(ROW, COL)
            VY(ROW, COL) = MAX_V * VY(ROW, COL) / CV(ROW, COL)
            CV(ROW, COL) = MAX_V
            CVX(ROW, COL) = ABS(VX(ROW, COL))
            CVY(ROW, COL) = ABS(VY(ROW, COL))
        END IF
    END DO
END DO

VX1=VX
VY1=VY

!************************************************************************
!******************Sediment Transport Calculation ***********************
!************************************************************************
DO ROW = 1, ROWS
    DO COL = 1, COLS

        IF (H(ROW, COL) > H_MIN .AND. CV(ROW, COL) > MIN_V) THEN

            !Erosion of mud
            TMX(ROW, COL,:) = 0
            TECR(ROW, COL) = 0
            IF (CVY(ROW, COL) < 1E-6) THEN
                IF (VX(ROW, COL) > 0) THEN
                    CURR_DIR = PII / 2
                ELSE
                    CURR_DIR = -PII / 2
                END IF
            ELSE
                IF (VY(ROW, COL) > 0) THEN
                    CURR_DIR = ATAN(VX(ROW, COL) / VY(ROW, COL))
                ELSE
                    CURR_DIR = ATAN(VX(ROW, COL) / &
                        VY(ROW, COL)) + (PII / 2)
                END IF
            END IF

            EM1 = 0
            DO N = 1, NWS2
                !Calculate angle between wave and current direction
                WC_DIR = WAVE_DIR(ROW, COL, N) - CURR_DIR
                HS = WAVEH(ROW, COL, N)
                TP = WAVET(ROW, COL, N)
                CALL MUD_ER(CV(ROW, COL), HS, TP, WC_DIR, H(ROW, COL), &
                            PMUD(ROW, COL), DMUD, TE, TCR_MAX, TCR_SAND, &
                            BM(ROW, COL), TSS, EMN, TMX(ROW, COL, N), &
                            TECRN, .FALSE.)
                EM1 = EM1 + (EMN * P(N))
            END DO
            TECR(ROW, COL) = TECRN
```

```
IF (PMUD(ROW, COL) < 0.15) THEN

    !Sand transport
    QT = 0
    DO N = 1, NWS2
        HS = WAVEH(ROW, COL, N)
        TP = WAVET(ROW, COL, N)
        QTN = SAND_TR(CV(ROW, COL), HS, TP, H(ROW, COL), &
             DSTR, D50, D90, DS, PMUD(ROW, COL), BM(ROW, COL))
        QT = QT + (QTN * P(N))
    END DO

    IF (PMUD(ROW, COL) < 0.05) THEN
        QM_SAND = QT * TSS / ((1 - PS) * GRID_SIZE)
        EM1 = 0
    ELSE
        QM_SAND = 10.0 * (0.15 - PMUD(ROW, COL)) * QT * &
                  TSS / ((1 - PS) * GRID_SIZE)
        EM1 = EM1 * 10.0 * (PMUD(ROW, COL) - 0.05)
    END IF
    EM2 = PMUD(ROW, COL) * QM_SAND
    QM_SAND = (1 - PMUD(ROW, COL)) * QM_SAND
ELSE
    QM_SAND = 0
    EM2 = 0
END IF

!Total erosion, including that associated with sand transport
EM = EM1 + EM2

!Sand transport resolved components
QMX = QM_SAND * CVX(ROW, COL) / CV(ROW, COL)
QMY = QM_SAND * CVY(ROW, COL) / CV(ROW, COL)

!Limit mud erosion + sand transport
!to prevent erosion below minimum bed level
MAXER = GRID(ROW, COL) - FGRID(ROW, COL)
IF (MAXER < 0) MAXER = 0
ERTOT = EM + QMX + QMY
IF (ERTOT > MAXER) THEN
    ERFR = MAXER / ERTOT
    EM = EM * ERFR
    QMX = QMX * ERFR
    QMY = QMY * ERFR
END IF

!Sand transport into neighbouring cells
!Ensuring no sediment transported into cells
!with less than minimum depth
IF (VY(ROW, COL) < 0) THEN
    IF (ROW > 1) THEN
        IF (H(ROW - 1, COL) < H_MIN) THEN
            QMY = 0
        END IF
    END IF
ELSE
    IF (ROW < ROWS) THEN
        IF (H(ROW + 1, COL) < H_MIN) THEN
            QMY = 0
        END IF
    END IF
END IF
IF (VX(ROW, COL) < 0) THEN
    IF (COL > 1) THEN
        IF (H(ROW, COL - 1) < H_MIN) THEN
```

```
                QMX = 0
            END IF
        END IF
    ELSE
        IF (COL < COLS) THEN
            IF (H(ROW, COL + 1) < H_MIN) THEN
                QMX = 0
            END IF
        END IF
    END IF

    !Add eroded mud to mud out total
    !and remove from active layer
    !(limited by total mud in active layer)
    IF (ALYR(ROW, COL, 2) > EM) THEN
        MUD_OUT = MUD_OUT + (EM * CELL_AREA)
        ETOT(ROW, COL) = ETOT(ROW, COL) + EM
        ALYR(ROW, COL, 2) = ALYR(ROW, COL, 2) - EM
    ELSE
        MUD_OUT = MUD_OUT + (ALYR(ROW, COL, 2) * CELL_AREA)
        ETOT(ROW, COL) = ETOT(ROW, COL) + ALYR(ROW, COL, 2)
        ALYR(ROW, COL, 2) = 0
    END IF

    !Remove transported sand from active layer
    !(limited by total sand in active layer)
    IF (ALYR(ROW, COL, 1) > QMX + QMY) THEN
        ALYR(ROW, COL, 1) = ALYR(ROW, COL, 1) - (QMX + QMY)
    ELSE
        IF (QMX + QMY > 1E-6) THEN
            ALYR_FR1 = QMX / (QMX + QMY)
            ALYR_FR2 = QMY / (QMX + QMY)
            QMX = ALYR(ROW, COL, 1) * ALYR_FR1
            QMY = ALYR(ROW, COL, 1) * ALYR_FR2
        END IF
        ALYR(ROW, COL, 1) = 0
    END IF

    !Add transported sand to active layer in neighbouring cells
    IF (VY(ROW, COL) < 0) THEN
        IF (ROW > 1) THEN
            ALYR1(ROW - 1, COL, 1) = ALYR1(ROW - 1, COL, 1) + QMY
        ELSE
            SAND_OUT = SAND_OUT + (QMY * CELL_AREA)
        END IF
    ELSE
        IF (ROW < ROWS) THEN
            ALYR1(ROW + 1, COL, 1) = ALYR1(ROW + 1, COL, 1) + QMY
        ELSE
            SAND_OUT = SAND_OUT + (QMY * CELL_AREA)
        END IF
    END IF

    IF (VX(ROW, COL) < 0) THEN
        IF (COL > 1) THEN
            ALYR1(ROW, COL - 1, 1) = ALYR1(ROW, COL - 1, 1) + QMX
        END IF
    ELSE
        IF (COL < COLS) THEN
            ALYR1(ROW, COL + 1, 1) = ALYR1(ROW, COL + 1, 1) + QMX
        END IF
    END IF

ELSE
    !Cririca bed shear stress calculated for all cells
```

```
            !(May be needed for dry cells, for lateral transport
            !calculation) Interpolate critical bed shear stress
            !based on Whitehouse et. al. Fig 31
            IF (PMUD(ROW, COL) > 0.2) THEN
                TECR(ROW, COL) = 1.25 * ((TE * (PMUD(ROW, COL) - 0.2)) + &
                                 (TCR_MAX * (1 - PMUD(ROW, COL))))
            ELSE
                TECR(ROW, COL) = 5.0 * (((0.2 - PMUD(ROW, COL)) * &
                                 TCR_SAND) + (PMUD(ROW, COL) * TCR_MAX))
            END IF
            TMX(ROW, COL,:) = 0
        END IF !(H < H_MIN OR V < MIN V)

    END DO !(COLS)
END DO    !(ROWS)




!Lateral sediment transport
DO ROW = 1, ROWS
    DO COL = 1, COLS

        !Forward lateral transport (to row + 1)
        IF (ROW < ROWS) THEN

            BDF = GRID(ROW, COL) - GRID(ROW + 1, COL)
            IF (BDF > 0) THEN
                IF (H(ROW + 1, COL) > H_MIN &
                   .AND. CV(ROW + 1, COL) > MIN_V) THEN
                    IF (PMUD(ROW, COL) > 0.05) THEN
                        EL1 = 0
                        DO N = 1, NWS2
                            IF (TMX(ROW + 1, COL, N) > &
                                TECR(ROW, COL)) THEN
                                EM = LTFR * ME * (TMX(ROW + 1, COL, N) - &
                                     TECR(ROW, COL)) * TSS / DMUD
                            ELSE
                                EM = 0
                            END IF
                            IF (PMUD(ROW, COL) > 0.15) THEN
                                ELN = EM * BDF / GRID_SIZE
                            ELSE
                                ELN = 10.0 * (PMUD(ROW, COL) - 0.05) * &
                                      EM * BDF / GRID_SIZE
                            END IF
                            EL1 = EL1 + (ELN * P(N))
                        END DO
                    ELSE
                        EL1 = 0
                    END IF

                    IF (PMUD(ROW, COL) < 0.15) THEN
                        QT = 0
                        DO N = 1, NWS2
                            HS = WAVEH(ROW + 1, COL, N)
                            TP = WAVET(ROW + 1, COL, N)
                            QTN = SAND_TR(CV(ROW + 1, COL), HS, TP, &
                                  H(ROW + 1, COL), DSTR, D50, D90, DS, &
                                  PMUD(ROW, COL), NULL)
                            QT = QT + (QTN * P(N))
                        END DO
                        IF (PMUD(ROW, COL) < 0.05) THEN
```

```
                        QMLATF = LTFR * QT * TSS * BDF / &
                                 (GRID_SIZE ** 2)
                    ELSE
                        QMLATF = LTFR * 10.0 * &
                                 (0.15 - PMUD(ROW, COL)) * QT * TSS * &
                                 BDF / (GRID_SIZE ** 2)
                    END IF
                ELSE
                    QMLATF = 0
                END IF
                ELATF = EL1 + (PMUD(ROW, COL) * QMLATF)
                QMLATF = (1 - PMUD(ROW, COL)) * QMLATF
            ELSE
                ELATF = 0
                QMLATF = 0
            END IF
        ELSE
            BDF = 0
            ELATF = 0
            QMLATF = 0
        END IF
    ELSE
        BDF = 0
        ELATF = 0
        QMLATF = 0
    END IF

    !Backward lateral transport (to row - 1)
    IF (ROW > 1) THEN
        BDB = GRID(ROW, COL) - GRID(ROW - 1, COL)
        IF (BDB > 0) THEN
            IF (H(ROW - 1, COL) > H_MIN .AND. &
                CV(ROW - 1, COL) > MIN_V) THEN
                IF (PMUD(ROW, COL) > 0.05) THEN
                    EL1 = 0
                    DO N = 1, NWS2
                        IF (TMX(ROW - 1, COL, N) > &
                            TECR(ROW, COL)) THEN
                            EM = LTFR * ME * (TMX(ROW - 1, COL, N) - &
                                 TECR(ROW, COL)) * TSS / DMUD
                        ELSE
                            EM = 0
                        END IF
                        IF (PMUD(ROW, COL) > 0.15) THEN
                            ELN = EM * BDB / GRID_SIZE
                        ELSE
                            ELN = 10.0 * (PMUD(ROW, COL) - 0.05) * &
                                  EM * BDB / GRID_SIZE
                        END IF
                        EL1 = EL1 + (ELN * P(N))
                    END DO
                ELSE
                    EL1 = 0
                END IF

                IF (PMUD(ROW, COL) < 0.15) THEN
                    QT = 0
                    DO N = 1, NWS2
                        HS = WAVEH(ROW - 1, COL, N)
                        TP = WAVET(ROW - 1, COL, N)
                        QTN = SAND_TR(CV(ROW - 1, COL), HS, TP, &
                              H(ROW - 1, COL), DSTR, D50, D90, DS, &
                              PMUD(ROW, COL), NULL)
                        QT = QT + (QTN * P(N))
                    END DO
```

```
                    IF (PMUD(ROW, COL) < 0.05) THEN
                        QMLATB = LTFR * QT * TSS * BDB / &
                                 (GRID_SIZE ** 2)
                    ELSE
                        QMLATB = LTFR * 10.0 * &
                                (0.15 - PMUD(ROW, COL)) * QT * &
                                TSS * BDB / (GRID_SIZE ** 2)
                    END IF
                ELSE
                    QMLATB = 0
                END IF
                ELATB = EL1 + (PMUD(ROW, COL) * QMLATB)
                QMLATB = (1 - PMUD(ROW, COL)) * QMLATB
            ELSE
                ELATB = 0
                QMLATB = 0
            END IF
        ELSE
            BDB = 0
            ELATB = 0
            QMLATB = 0
        END IF
    ELSE
        BDB = 0
        ELATB = 0
        QMLATB = 0
    END IF

    !Leftward lateral transport (to col - 1)
    IF (COL > 1) THEN
        BDL = GRID(ROW, COL) - GRID(ROW, COL - 1)
        IF (BDL > 0) THEN
            IF (H(ROW, COL - 1) > H_MIN .AND. &
                CV(ROW, COL - 1) > MIN_V) THEN
                IF (PMUD(ROW, COL) > 0.05) THEN
                    EL1 = 0
                    DO N = 1, NWS2
                        IF (TMX(ROW, COL - 1, N) > TECR(ROW, COL)) &
                            THEN
                            EM = LTFR * ME * (TMX(ROW, COL - 1, N) - &
                                TECR(ROW, COL)) * TSS / DMUD
                        ELSE
                            EM = 0
                        END IF
                        IF (PMUD(ROW, COL) > 0.15) THEN
                            ELN = EM * BDL / GRID_SIZE
                        ELSE
                            ELN = 10.0 * (PMUD(ROW, COL) - 0.05) * &
                                EM * BDL / GRID_SIZE
                        END IF
                        EL1 = EL1 + (ELN * P(N))
                    END DO
                ELSE
                    EL1 = 0
                END IF

                IF (PMUD(ROW, COL) < 0.15) THEN
                    QT = 0
                    DO N = 1, NWS2
                        HS = WAVEH(ROW, COL - 1, N)
                        TP = WAVET(ROW, COL - 1, N)
                        QTN = SAND_TR(CV(ROW, COL - 1), HS, TP, &
                            H(ROW, COL - 1), DSTR, D50, D90, DS, &
                            PMUD(ROW, COL), NULL)
                        QT = QT + (QTN * P(N))
```

```
                    END DO
                    IF (PMUD(ROW, COL) < 0.05) THEN
                        QMLATL = LTFR * QT * TSS * BDL / &
                            (GRID_SIZE ** 2)
                    ELSE
                        QMLATL = LTFR * 10.0 * &
                            (0.15 - PMUD(ROW, COL)) * QT * TSS * &
                            BDL / (GRID_SIZE ** 2)
                    END IF
                ELSE
                    QMLATL = 0
                END IF
                ELATL = EL1 + (PMUD(ROW, COL) * QMLATL)
                QMLATL = (1 - PMUD(ROW, COL)) * QMLATL
            ELSE
                ELATL = 0
                QMLATL = 0
            END IF
        ELSE
            BDL = 0
            ELATL = 0
            QMLATL = 0
        END IF
    ELSE
        BDL = 0
        ELATL = 0
        QMLATL = 0
    END IF

    !Rightward lateral transport (to col + 1)
    IF (COL < COLS) THEN
        BDR = GRID(ROW, COL) - GRID(ROW, COL + 1)
        IF (BDR > 0) THEN
            IF (H(ROW, COL + 1) > H_MIN .AND. &
                CV(ROW, COL + 1) > MIN_V) THEN
                IF (PMUD(ROW, COL) > 0.05) THEN
                    EL1 = 0
                    DO N = 1, NWS2
                        IF (TMX(ROW, COL + 1, N) > &
                            TECR(ROW, COL)) THEN
                            EM = LTFR * ME * (TMX(ROW, COL + 1, N) - &
                                TECR(ROW, COL)) * TSS / DMUD
                        ELSE
                            EM = 0
                        END IF
                        IF (PMUD(ROW, COL) > 0.15) THEN
                            ELN = EM * BDR / GRID_SIZE
                        ELSE
                            ELN = 10.0 * (PMUD(ROW, COL) - 0.05) * &
                                EM * BDR / GRID_SIZE
                        END IF
                        EL1 = EL1 + (ELN * P(N))
                    END DO
                ELSE
                    EL1 = 0
                END IF

                IF (PMUD(ROW, COL) < 0.15) THEN
                    QT = 0
                    DO N = 1, NWS2
                        HS = WAVEH(ROW, COL + 1, N)
                        TP = WAVET(ROW, COL + 1, N)
                        QTN = SAND_TR(CV(ROW, COL + 1), HS, TP, &
                            H(ROW, COL + 1), DSTR, D50, D90, DS, &
                            PMUD(ROW, COL), NULL)
```

```
                    QT = QT + (QTN * P(N))
                END DO
                IF (PMUD(ROW, COL) < 0.05) THEN
                    QMLATR = LTFR * QT * TSS * &
                        BDR / (GRID_SIZE ** 2)
                ELSE
                    QMLATR = LTFR * 10.0 * &
                        (0.15 - PMUD(ROW, COL)) * QT * TSS * &
                        BDR / (GRID_SIZE ** 2)
                END IF
            ELSE
                QMLATR = 0
            END IF
            ELATR = EL1 + (PMUD(ROW, COL) * QMLATR)
            QMLATR = (1 - PMUD(ROW, COL)) * QMLATR
        ELSE
            ELATR = 0
            QMLATR = 0
        END IF
    ELSE
        BDR = 0
        ELATR = 0
        QMLATR = 0
    END IF
ELSE
    BDR = 0
    ELATR = 0
    QMLATR = 0
END IF

!Total lateral transport of mud
ELATT = ELATF + ELATB + ELATL + ELATR
!Total lateral transport of sand
QMLATT = QMLATF + QMLATB + QMLATL + QMLATR

!Calculate maximum lateral transport
IF (GRID(ROW, COL) > FGRID(ROW, COL)) THEN
    MAXE = GRID(ROW, COL) - FGRID(ROW, COL)
ELSE
    MAXE = 0
END IF

!If total lateral transport exceeds maximum then
!adjust values so total is equal to maximum
LATT = ELATT + QMLATT
IF (LATT > MAXE) THEN
    IF (LATT > 1E-6) THEN
        LATR = MAXE / LATT
    ELSE
        LATR = 0
    END IF
    ELATF = ELATF * LATR
    ELATB = ELATB * LATR
    ELATL = ELATL * LATR
    ELATR = ELATR * LATR
    ELATT = ELATT * LATR
    QMLATF = QMLATF * LATR
    QMLATB = QMLATB * LATR
    QMLATL = QMLATL * LATR
    QMLATR = QMLATR * LATR
    QMLATT = QMLATT * LATR
END IF

!Total lateral transport depth limited to maximum bed level
!difference to neighbouring cell
```

```
BDMAX = MAX(BDF, BDB, BDL, BDR)
IF (LATT > BDMAX) THEN
    IF (LATT > 1E-6) THEN
        BDR = BDMAX / LATT
    ELSE
        BDR = 0
    END IF
    ELATF = ELATF * BDR
    ELATB = ELATB * BDR
    ELATL = ELATL * BDR
    ELATR = ELATR * BDR
    ELATT = ELATT * BDR
    QMLATF = ELATF * BDR
    QMLATB = ELATB * BDR
    QMLATL = ELATL * BDR
    QMLATR = ELATR * BDR
    QMLATT = ELATT * BDR
END IF

!Lateral sand transport limited total
!available sand in the active layer
IF (QMLATT > ALYR(ROW, COL, 1)) THEN
    IF (QMLATT > 1E-6) THEN
        QMR = ALYR(ROW, COL, 1) / QMLATT
        QMLATT = ALYR(ROW, COL, 1)
    ELSE
        QMR = 0
        QMLATT = 0
    END IF
    QMLATF = QMLATF * QMR
    QMLATB = QMLATB * QMR
    QMLATL = QMLATL * QMR
    QMLATR = QMLATR * QMR
END IF

!Lateral mud transport limited to total mud in the active layer
IF (ELATT > ALYR(ROW, COL, 2)) THEN
    IF (ELATT > 1E-6) THEN
        ELR = ALYR(ROW, COL, 2) / ELATT
        ELATT = ALYR(ROW, COL, 2)
    ELSE
        ELR = 0
        ELATT = 0
    END IF
    ELATF = ELATF * ELR
    ELATB = ELATB * ELR
    ELATL = ELATL * ELR
    ELATR = ELATR * ELR
END IF

!Active layer sand/mud reduced by
!lateral transported sediment depths
ALYR(ROW, COL, 1) = ALYR(ROW, COL, 1) - QMLATT
ALYR(ROW, COL, 2) = ALYR(ROW, COL, 2) - ELATT

!Active layer in neighbouring cells
!increased by transported depths
IF (ROW < ROWS) THEN
    ALYR1(ROW + 1, COL, 1) = ALYR1(ROW + 1, COL, 1) + QMLATF
    ALYR1(ROW + 1, COL, 2) = ALYR1(ROW + 1, COL, 2) + ELATF
END IF

IF (ROW > 1) THEN
    ALYR1(ROW - 1, COL, 1) = ALYR1(ROW - 1, COL, 1) + QMLATB
    ALYR1(ROW - 1, COL, 2) = ALYR1(ROW - 1, COL, 2) + ELATB
```

```
        END IF

        IF (COL > 1) THEN
            ALYR1(ROW, COL - 1, 1) = ALYR1(ROW, COL - 1, 1) + QMLATL
            ALYR1(ROW, COL - 1, 2) = ALYR1(ROW, COL - 1, 2) + ELATL
        END IF

        IF (COL < COLS) THEN
            ALYR1(ROW, COL + 1, 1) = ALYR1(ROW, COL + 1, 1) + QMLATR
            ALYR1(ROW, COL + 1, 2) = ALYR1(ROW, COL + 1, 2) + ELATR
        END IF

    END DO
END DO

ALYR = ALYR + ALYR1

!Calculate timestep for deposition/diffusion/advection



!Calculate suspended sediment deposition

DO ROW = 1, ROWS
    DO COL = 1, COLS

        BMS = BM_MAX * BM(ROW, COL)
        NST = 250 * (BMS ** 0.3032)
        HST = 0.069 * (BMS ** 0.1876)
        DST = 0.0006 * (BMS ** 0.3)
        IF (H(ROW, COL) > H_MIN) THEN

            !Deposition
            IF (BM(ROW, COL) > 1E-3) THEN
                !Marsh related sediment trapping
                SED_TRP = 0.224 * ((CV(ROW, COL) * DST / KV) ** 0.718) * &
                    ((8E-6 / DST) ** 2.08)
                FRSM = CV(ROW, COL) * SED_TRP * NST * DST * HST * TSS
                IF (FRSM > 1.0) FRSM = 1.0
                DEP1 = FRSM * MEAN_H(ROW, COL)
            ELSE
                DEP1 = 0
            END IF

            DEP2 = 0
            DO N = 1, NWS2
                IF (TMX(ROW, COL, N) < TD) THEN
                    !Whitehouse et. al.
                    !'Dynamics of Estuarine Muds', eqn 8.2
                    FRD = (1 - (TMX(ROW, COL, N) / TD)) * WS * &
                        TSS / MEAN_H(ROW, COL)
                    IF (FRD > 1.0) THEN
                        FRD = 1.0
                    END IF
                    DEPN = FRD *  MEAN_H(ROW, COL)
                ELSE
                    DEPN = 0
                END IF
                DEP2 = DEP2 + (DEPN  * P(N))
            END DO
            DTOT(ROW, COL) = DTOT(ROW, COL) + DEP1 + DEP2
            MUD_IN = MUD_IN + ((DEP1 + DEP2) * SC(ROW, COL) * CELL_AREA)
            ALYR(ROW, COL, 2) = ALYR(ROW, COL, 2) + &
                ((DEP1 + DEP2) * SC(ROW, COL))
```

```
        END IF

    END DO
END DO


!Marsh related deposition
!2 Organogenic sediment production
!Since values are very small, this is only added
!approximately once per month
IF (MT > 2628000) THEN
    DO ROW = 1, ROWS
        DO COL = 1, COLS

            IF (BM(ROW, COL) > 1E-3) THEN
                !Sediment generated is assumed to be of finest
                !available fraction size.  KB is max rate of production
                !in m/yr (typically in range 0.001-0.01)
                ALYR(ROW, COL, 2) = ALYR(ROW, COL, 2) + KB * &
                    BM(ROW, COL) / 12.0
                MUD_IN = MUD_IN + KB * BM(ROW, COL) * CELL_AREA / 12.0
            END IF
        END DO
    END DO
    MT = MT - 2628000
END IF
MT = MT + TSS


!Downstream boundary sand input
QTOT = 0
TMX_TOT = 0
QTTOT = 0
QTC = 0
DO COL = 1, COLS
    IF (H(ROWS, COL) > H_MIN) THEN
        QTOT = QTOT + QY(ROWS, COL)
        TMXTOTR(COL) = 0
        DO N = 1, NWS2
            TMX_TOT = TMX_TOT + (TMX(ROWS, COL, N) * P(N))
            TMXTOTR(COL) = TMXTOTR(COL) + (TMX(ROWS, COL, N) * P(N))
            HS = WAVEH(ROWS, COL, N)
            TP = WAVET(ROWS, COL, N)
            QTN = SAND_TR(CV(ROWS, COL), HS, TP, H(ROWS, COL), &
                          DSTR, D50, D90, DS, 0.0, BM(ROWS, COL))
            QTTOT = QTTOT + (QTN * P(N))
            QTC(COL) = QTC(COL) + (QTN * P(N))
        END DO
    END IF
END DO



DO COL = 1, COLS
    IF (H(ROWS, COL) > H_MIN .AND. CVY(ROWS, COL) > MIN_V) THEN
        IF (QTTOT > 1E-6) THEN
            ALYR(ROWS, COL, 1) = ALYR(ROWS, COL, 1) + &
                (DS_SAND_INPUT * TSS * QTC(COL) / (CELL_AREA * QTTOT))
            SAND_IN = SAND_IN + (DS_SAND_INPUT * TSS * QTC(COL) / QTTOT)
        ELSE
            ALYR(ROWS, COL, 1) = ALYR(ROWS, COL, 1) + (DS_SAND_INPUT * &
                TSS * TMXTOTR(COL) / (CELL_AREA * TMX_TOT))
            SAND_IN = SAND_IN + &
                (DS_SAND_INPUT * TSS * TMXTOTR(COL) / TMX_TOT)
        END IF
```

```
    END IF
END DO


!Update active layer and sublayers
DO ROW = 1, ROWS
    DO COL = 1, COLS

        ALYR_TOT = ALYR(ROW, COL, 1) + ALYR(ROW, COL, 2)
        DL = ALYR_TOT - ALYR_TH

        STH = GRID(ROW, COL) - ALYR_TH - BASE
        S_LYRS = STH / LYR_TH
        SUB_LYR_NR = INT(S_LYRS) + 1
        SUB_LYR_FR = GRID(ROW, COL) - ALYR_TH - BASE - &
            ((SUB_LYR_NR - 1.0) * LYR_TH)
        IF (SUB_LYR_FR < 0.0) SUB_LYR_FR = 0.0

        GRID(ROW, COL) = GRID(ROW, COL) + DL

        IF (DL > 0) THEN
            !Deposition
            IF (DL > LYR_TH - SUB_LYR_FR) THEN

                NL = INT((DL - (LYR_TH - SUB_LYR_FR)) / LYR_TH) + 1

                SED(ROW, COL, SUB_LYR_NR) = ((SED(ROW, COL, SUB_LYR_NR) &
                    * SUB_LYR_FR) + (ALYR(ROW, COL, 2) * &
                    (LYR_TH - SUB_LYR_FR) / ALYR_TOT)) / LYR_TH

                DO M = 1, NL
                    SED(ROW, COL, SUB_LYR_NR + M) = &
                        ALYR(ROW, COL, 2) / ALYR_TOT
                END DO

                ALYR(ROW, COL, 1) = ALYR(ROW, COL, 1) * &
                    (ALYR_TH / ALYR_TOT)
                ALYR(ROW, COL, 2) = ALYR_TH - ALYR(ROW, COL, 1)
            ELSE
               SED(ROW, COL, SUB_LYR_NR) = ((SED(ROW, COL, SUB_LYR_NR) * &
                    SUB_LYR_FR) + (ALYR(ROW, COL, 2) * DL / ALYR_TOT)) / &
                    (SUB_LYR_FR + DL)

               ALYR(ROW, COL, 1) = ALYR(ROW, COL, 1) * &
                    (ALYR_TH / ALYR_TOT)
               ALYR(ROW, COL, 2) = ALYR_TH - ALYR(ROW, COL, 1)
            END IF
        ELSE
            !Erosion
            IF (DL + SUB_LYR_FR < 0) THEN
                ALYR(ROW, COL, 2) = ALYR(ROW, COL, 2) + &
                    (SED(ROW, COL, SUB_LYR_NR) * SUB_LYR_FR) - &
                    (SED(ROW, COL, SUB_LYR_NR - 1) * (DL + SUB_LYR_FR))

                ALYR(ROW, COL, 1) = ALYR(ROW, COL, 1) + &
                    ((1 - SED(ROW, COL, SUB_LYR_NR)) * SUB_LYR_FR) - &
                  ((1 - SED(ROW, COL, SUB_LYR_NR - 1)) * &
                  (DL + SUB_LYR_FR))

            ELSE
                ALYR(ROW, COL, 2) = ALYR(ROW, COL, 2) - &
                    (SED(ROW, COL, SUB_LYR_NR) * DL)
                ALYR(ROW, COL, 1) = ALYR(ROW, COL, 1) - &
                    ((1 - SED(ROW, COL, SUB_LYR_NR)) * DL)
            END IF
```

```
        END IF
        IF (ALYR(ROW, COL, 1) < 0) ALYR(ROW, COL, 1) = 0
        IF (ALYR(ROW, COL, 2) < 0) ALYR(ROW, COL, 2) = 0
    END DO
END DO


END SUBROUTINE



REAL FUNCTION CALC_TMX(U, HS, TP, WCD, H)
!Function to calculate bed shear stress

REAL, INTENT(IN) :: U       !Depth averaged current velocity
REAL, INTENT(IN) :: HS      !Significant wave height
REAL, INTENT(IN) :: TP      !Peak spectral wave period
REAL, INTENT(IN) :: WCD     !Wave / current direction
REAL, INTENT(IN) :: H       !Water depth


REAL UST         !Shear velocity
REAL TC          !Bed shear stress due to current
REAL TW          !Bed shear stress due to waves
REAL WAVE_L      !Wave length
REAL UD, AD      !Peak wave orbital velocity and excursion
REAL HRMS        !Root-mean-square wave height
REAL TME         !Cycle-mean bed shear stress for smooth beds
                 !(Whitehouse et. al. 'Dynamics of Marine Muds' pg 55
REAL TCFSM       !Bed shear stress adjustment based on
                 !salt-marsh biomass present
REAL E1          !Calculated erosion rate (kg/m2/s)
REAL ME          !Erosion constant
                 !(Whitehouse et. al. Fig 16/Table 8, Pg.69)
REAL RW, N1, B1, FWS, TECR_SM
REAL T, T1, R, TMAX

INTEGER N

ME = 0.002
UST = 1
DO N = 1, 6
    !Whitehouse et. al. 'Dynamics of Marine Muds' Eqn 3.6
    !Taking velocity at 0.37h = depth averaged velocity
    UST = U / (5.5 + 2.5 * LOG(UST * 0.37 * H / KV))
END DO
TC = DWA * (UST ** 2)

!wavelength
!Using approximate solution (+-10%) from CEM pt. 2 Chap. 4
IF (TP > 1) THEN
    WAVE_L = (G * (TP ** 2) / (2 * PII)) * &
        ((TANH(4 * (PII ** 2) * H / ((TP ** 2) * G))) ** 0.5)
    HRMS = HS / 1.414
    UD = PII * HRMS / (TP * SINH(2 * PII * H / WAVE_L))
    AD = 0.5 * UD * TP / PII
    RW = UD * AD / KV
    IF (RW <= 5E5) THEN
        B1 = 2
        N1 = 0.5
    ELSE
        B1 = 0.0521
        N1 = 0.187
    END IF
```

```
    IF (RW > 1E-3) THEN
        FWS = B1 * (RW ** -N1)
        TW = 0.5 * DWA * FWS * (UD ** 2)
    ELSE
        TW = 0
    END IF
ELSE
    TW = 0
END IF

!Whitehouse et. al. Eqn 3.28
TME = TC * (1 + 9 * ((TW / (TC + TW)) ** 9))
!Eqn 3.26
TMAX = SQRT(((TME + TW * COS(WCD)) ** 2) + ((TW * SIN(WCD)) ** 2))

CALC_TMX = TMAX

END FUNCTION

END MODULE
```

MODULE SANDa

Subroutines	Purpose	Page nr.
VAN_RIJN_MF	Multi-fraction sediment transport calculation	354
CALC_ST_MF	Sediment transport calculation and bed updating	366
Functions		
TB_CRIT	Calculate critical bed shear stress	353
GETSIZE	Calculate d_{10}, d_{50} etc. for transport calculation	394

```
MODULE SANDa

USE SAND_MUD

IMPLICIT NONE

!Standard value for gravitational acceleration
REAL, PARAMETER :: G = 9.80665
!Density of sea water (at salinity = 35ppt, temperature = 10 degrees C)
REAL, PARAMETER :: DWA = 1027
!Kinematic viscosity of sea water (temperature/salinity as above)
REAL, PARAMETER :: KV = 1.36E-6
REAL, PARAMETER :: KPA = 0.4    !Von Karmans constant
REAL, PARAMETER     :: PII = 3.14159265358979

CONTAINS

REAL FUNCTION TB_CRIT(FI, DSED, FMIN, FMAX, PI, PCS, NF, BM)
!Function to calculate critical bed shear stress for
!multi-fraction sediment

!Array containing the fraction sizes (mean)
REAL, DIMENSION(:), INTENT(IN) :: FI
!Array containing the fraction sizes (min)
REAL, DIMENSION(:), INTENT(IN) :: FMIN
!Array containing the fraction sizes (max)
REAL, DIMENSION(:), INTENT(IN) :: FMAX
!Array containing the proportion of each fraction
REAL, DIMENSION(:), INTENT(IN) :: PI

REAL, INTENT(IN) :: DSED       !Particle density
REAL, INTENT(IN) :: BM         !Biomass as fraction of max biomass
REAL, INTENT(IN) :: PCS        !Clay fraction proportion

INTEGER, INTENT(IN) :: NF      !Number of sediment fractions

REAL KVEG       !Critical shear stress factor for salt marsh
REAL D50        !Median particle size
REAL SD         !Relative particle density
!Minimum sand particle size (smaller grain sizes treated as silt)
REAL DSAND
REAL DD50       !Dimensionless particle size parameter
REAL SPCRD50    !Sheilds parameter relating to median grain size
REAL TBCRD50    !Critical bed shear stress (before adjustments)
REAL CGEL, CGELS !Gelling concentration
!Crtical shear stress adjustment factors
REAL PHI_COHESIVE, PHI_PACKING, PHI_CLAY

KVEG = 5
```

```
DSAND = 0.000062

!Calculate d50 & D90 by linear interpolation between fractions
D50 = GETSIZE(FI, FMIN, FMAX, PI, NF + 1, 0.5)

!2 Compute non fraction dependant sediment characteristics
!Relative density of sediment
SD = DSED / DWA
!Dimensionless d50 particle diameter
DD50 = D50 * (((SD - 1) * G / (KV ** 2)) ** (1.0 / 3.0))

!Critical Shields parameter based on d50
!(Van Rijn, 2007a) and Soulsby (1997)
IF (DD50 < 4) THEN
    SPCRD50 = 0.115 / SQRT(DD50)
ELSE
    IF (DD50 < 10) THEN
        SPCRD50 = 0.14 / (DD50 ** 0.64)
    ELSE
        IF (DD50 < 20) THEN
            SPCRD50 = 0.04 / (DD50 ** 0.1)
        ELSE
            IF (DD50 < 150) THEN
                SPCRD50 = 0.013 * (DD50 ** 0.29)
            ELSE
                SPCRD50 = 0.055
            END IF
        END IF
    END IF
END IF

CGELS = 0.65
IF (D50 > DSAND) THEN
    CGEL = CGELS
ELSE
    CGEL = (D50 / DSAND) * CGELS
END IF
IF (CGEL < 0.05) CGEL = 0.05

!Critical bed shear stress based on d50, with
!adjustment based on biomass present
TBCRD50 = SPCRD50 * G * D50 * (DSED - DWA)
IF (D50 < DSAND) THEN
    PHI_COHESIVE = (DSAND / D50) ** 1.5
ELSE
    PHI_COHESIVE = 1
END IF

PHI_PACKING = CGEL / CGELS
IF (PHI_PACKING > 1) PHI_PACKING = 1

PHI_CLAY = (1 + PCS) ** 3
IF (PHI_CLAY > 2.0) PHI_CLAY = 2.0

TB_CRIT = PHI_CLAY * PHI_COHESIVE * PHI_PACKING * TBCRD50 * &
    (1 + (KVEG * BM))

END FUNCTION

SUBROUTINE VAN_RIJN_MF(FI, WS, FMIN, FMAX, PI, NF, DSED, H, HS, TP, &
    WAVE_DIR, V, QS, QB, BM, TBCRD50)
!Subroutine to calculate sediment transport for multiple sediment
!fractions, using a simplified version of Van Rijns TR2004 method

REAL, DIMENSION(:), INTENT(IN) :: FI    !Fraction sizes (mean)
```

```
REAL, DIMENSION(:), INTENT(IN) :: WS    !Fall velocities
REAL, DIMENSION(:), INTENT(IN) :: FMIN  !Fraction sizes (min)
REAL, DIMENSION(:), INTENT(IN) :: FMAX  !Fraction sizes (max)
REAL, DIMENSION(:), INTENT(IN) :: PI    !Fraction proportions
REAL, DIMENSION(:), INTENT(OUT):: QS    !Suspended sediment transport
REAL, DIMENSION(:), INTENT(OUT):: QB    !Bed load transport

REAL, INTENT(IN) :: DSED                !Sediment density
REAL, INTENT(IN) :: H                   !Water depth
REAL, INTENT(IN) :: HS                  !Significant wave height
REAL, INTENT(IN) :: TP                  !Peak spectral wave period
!Wave direction, relative to current direction (clockwise, radians)
REAL, INTENT(IN) :: WAVE_DIR
REAL, INTENT(IN) :: V              !Depth averaged flow velocity
REAL, INTENT(IN) :: BM             !Biomass as fraction of max biomass
!Critical bed shear stress based on median grain size
REAL, INTENT(OUT):: TBCRD50

INTEGER, INTENT(IN) :: NF               !Number of sediment fractions

REAL U                          !Absolute (unsigned) velocity
!Median grain size, d90 grain size, relative particle density
REAL D50, D90, SD
REAL KSC, KSW           !Current related / wave related bed roughness
REAL A                  !Reference level
REAL DD50               !Dimensionless d50 particle size
!Wavelength, near bed orbital excursion, near bed orbital velocity
REAL L, AD, UD
REAL KA                 !Apparent bed roughness
REAL FC, FC1, FA, FW    !Friction factors
REAL FDC, FDW, FDCW     !Fricton factors
!Efficiency factors (currents / waves), wave-current interaction factor
REAL EFC, EFW, ACW
!Bed shear stress due to currents / waves / combined waves & currents
REAL TC, TW, TCW
REAL CDU                !Current related shear velocity
!Shields parameter basedon d50, unadjusted critical bed shear stress
REAL SPCRD50, TBCRD500
REAL DSAND, FSILT       !Minimum sand grain size, silt factor
!Dimensionless particles size for individual sediment fraction
REAL DFI
REAL LI                 !Correction factor of effective grain shear stress
!Hiding-exposure correction factor, bed shear stress parameter
REAL EI, TI
REAL CA                 !Reference concentration
REAL BTA, ALPHA         !Beta factor / alpha factor
REAL MR                 !Max. ratio
REAL KVEG               !Salt marsh critical bed shear stress factor
REAL CW, WDU            !Wave related shear velcoity
REAL CGEL, CGELS        !Gelling concentration
REAL ALPHA1             !Floculation factor (PHI_FLOC) exponent
REAL ESC_MAX            !Sediment mixing coefficient currents for z > h/2
REAL GAMMA_BR           !Empirical wave breaking coefficient
REAL DELTA_W            !Wave boundary layer thickness
REAL DELTA_S     !Thickness of effective near-bed sediment mixing layer
REAL ESW_MAX            !Sediment mixing coefficient waves for z > h/2
REAL ESW_BED            !Sediment mixing coefficient waves, near bed
REAL BTA_C              !Beta factor currents
REAL BTA_W              !Beta factor waves
REAL Z, Z1, Z05         !Height above bed
REAL SC, SC1, SC2       !Sediment concentration
REAL DZ, DZ1, DCDZ !Change in z / rate of change of concentration with z
REAL UZ, UZ1            !Current velocity at height z
REAL PHI_FLOC, PHI_HS   !Floculation / hindered settling factors
REAL PHI_D, PHI_FS      !Damping coefficient / calibration factor
```

```
!Critical shear stress parameters for cohesion & bulk density effects
REAL PHI_COHESIVE, PHI_PACKING
REAL PHI_CLAY       !Critical shear stress parameter for clay proportion
REAL WSI            !Adjusted fall velocity for fraction i
!Sediment coefficient for currents/waves/combined currents and waves
REAL ESC, ESW, ESCW
!Near bed orital velocity at intervals through wave cycle
REAL VI1, VI2, VI3, VI4, VI5
!Bed shear stress at intervals through wave cycle
REAL TBCWI1, TBCWI2, TBCWI3, TBCWI4, TBCWI5
!Transport direction at intervals through wave cycle
REAL ST_DR1, ST_DR2, ST_DR3, ST_DR4, ST_DR5
!Transport parameter at intervals through wave cycle
REAL TI1, TI2, TI3, TI4, TI5
REAL QB1, QB2, QB3, QB4, QB5    !Bed load at intervals through wave cycle
REAL DGRAVEL, DSILT             !Particle size parameters
REAL SI                         !Current-wave mobility parameter
REAL UWC                        !Wave/current velocity parameter
REAL FCS, FFS !Ripple roughness factor, megaripple roughness factor
REAL PCS                        !Clay proportion in bed
!Ripple roughness, megaripple roughness, dune roughness
REAL KSCR, KSCMR, KSCD
REAL WDR, GAMMA1         !Wave direction, wave direction factor
REAL DELTA_V, DELTA_M    !Velocity distribution factors
REAL UBD                 !Current velocity at edge of wave boundary layer
REAL DC, DC1             !Sediment concentration increment
!Sediment concentration at half timestep, sediment concentration gradient
REAL SC05, DCDZ1
REAL DSDC_FR, CHZC

INTEGER N                       !Counter

!True if integration carried using defined
!changes in sediment concentration
!False for defined changes in elevation (z)
LOGICAL BYSC


KVEG = 5

U = ABS(V)


!Calculate d50 & D90 by linear interpolation between fractions
D50 = GETSIZE(FI, FMIN, FMAX, PI, NF + 1, 0.5)
D90 = GETSIZE(FI, FMIN, FMAX, PI, NF + 1, 0.9)

!2 Compute non fraction dependant sediment characteristics
SD = DSED / DWA   !Relative density of sediment
!Dimensionless d50 particle diameter
DD50 = D50 * (((SD - 1) * G / (KV ** 2)) ** (1.0 / 3.0))

!3 Compute the wavelength
!Using approximate solution (+-10%) from CEM pt. 2 Chap. 4
L = (G * (TP ** 2) / (2 * PII)) * &
    ((TANH(4 * (PII ** 2) * H / ((TP ** 2) * G))) ** 0.5)

!4 Compute the wave parameters
IF (H / L > 2.0) THEN
    AD = 0
    UD = 0
ELSE
    AD = HS / (2 * SINH(2 * PII * H / L))
    UD = PII * HS / (TP * SINH(2 * PII * H / L))
END IF
```

```
!Bed roughness (Van Rijn, 2007)
DGRAVEL = 0.002
DSILT = 0.000032
DSAND = 0.000062
UWC = (UD ** 2) + (U ** 2)
SI = UWC/((SD - 1) * G * D50)
IF (D50 <= 0.25 * DGRAVEL) THEN
    FCS = 1
ELSE
    FCS = (0.25 * DGRAVEL / D50) ** 1.2
END IF
IF (D50 >= 1.5 * DSAND) THEN
    FFS = 1
ELSE
    FFS = D50 / (1.5 * DSAND)
END IF

IF (D50 < DSILT) THEN
    KSCR = 20 * DSILT
    KSCMR = 0
    KSCD = 0
ELSE
    IF (SI <= 50) THEN
        KSCR = 150 * FCS * D50
    ELSE
        IF (SI > 250) THEN
            KSCR = 20 * FCS * D50
        ELSE
            KSCR = (182.5 - 0.652 * SI) * FCS * D50
        END IF
    END IF
    IF (D50 >= DSILT) THEN
        IF (SI <= 50) THEN
            KSCMR = 0.0002 * FFS * SI * H
        ELSE
            IF (SI <= 550) THEN
                KSCMR = (0.011 - 0.00002 * SI) * FFS * H
            ELSE
                IF (D50 >= 1.5 * DSAND) THEN
                    KSCMR = 0.02
                ELSE
                    KSCMR = 200 * D50
                END IF
            END IF
        END IF
    ELSE
        KSCMR = 0
    END IF
    IF (SI <= 100) THEN
        KSCD = 0.0004 * FFS * SI * H
    ELSE
        IF (SI <= 600) THEN
            KSCD = (0.048 - 0.00008 * SI) * FFS * H
        ELSE
            KSCD = 0
        END IF
    END IF
END IF

IF (WAVE_DIR < 0) THEN
    WDR = WAVE_DIR + PII
ELSE
    IF (WAVE_DIR > PII) THEN
        WDR = WAVE_DIR - PII
```

```
    ELSE
        WDR = WAVE_DIR
    END IF
END IF
GAMMA1 = 0.8 + WDR - (0.3 * (WDR ** 2))

!Current related bed roughness
KSC = SQRT((KSCR ** 2) + (KSCMR ** 2) + (KSCD ** 2))
IF (KSC > H) KSC = H
KSW = KSCR
IF (KSW > H) KSW = H !Wave related bed roughness

!5 Compute the apparent bed roughness
IF (U > 1E-6) THEN
    IF (GAMMA1 * UD / U > 2.5) THEN
        KA = 10 * KSC
    ELSE
        KA = KSC * EXP(GAMMA1 * UD / U)
    END IF
ELSE
    KA = 10 * KSC
END IF
IF (KA > 10) KA = 10 * KSC
IF (KA > H) KA = H

IF (SI > 250) THEN
    A = 20 * D50
ELSE
    A = 0.5 * KSCR !Reference level
END IF
IF (A < 0.01) A = 0.01
IF (A < KSC / 30) A = A + KSC / 30
IF (A > H) A = 0.99 * H

!6 Compute the friction factors

!Current
!Some limits need to be set for shallow depths
MR = 1.1
IF (12 * H / D90 < MR) THEN
    FDC = 8 * G / ((18 * LOG10(MR)) ** 2)
ELSE
    FDC = 8 * G / ((18 * LOG10(12 * H  / D90)) ** 2)
END IF

IF (12 * H / KSC < MR) THEN
    CHZC = 18 * LOG10(MR)
ELSE
    CHZC = 18 * LOG10(12 * H  / KSC)
END IF
FC = 8 * G / (CHZC ** 2)

IF (12 * H / KA < MR) THEN
    FA = 0.24 / ((LOG10(MR)) ** 2)
ELSE
    FA = 0.24 / ((LOG10(12 * H  / KA)) ** 2)
END IF

!Waves
IF (AD / KSW < 1.53) THEN
    !Very small value of AD can cause error
    FW = 0.3
ELSE
    FW = EXP( -6 + (5.2 / ((AD / KSW) ** 0.19)))
END IF
```

```
!7 Compute time averaged bed-shear stresses
EFC = FDC / FC      !Efficiency factor currents
IF (EFC > 1.0) EFC = 1.0

IF (DD50 >= 5.0) THEN
    EFW = 0.14
ELSE
    IF (DD50 <= 2.0) THEN
        EFW = 0.35
    ELSE
        EFW = 0.7 / DD50    !Efficiency factor waves
    END IF
ENDIF

!Wave boundary layer thickness (Van Rijn, 2007b, eqn 3b)
IF (AD < 1E-6) THEN
    DELTA_W = 0
ELSE
    DELTA_W = 0.36 * AD * ((AD / KSW) ** -0.25)
END IF

IF (DELTA_W / KSC < 0.03) THEN
    ACW = 1
ELSE
    IF (KA > 89 * DELTA_W) THEN
        ACW = 0.00001
    ELSE
        !Wave current interaction coefficient
        ACW = ((LOG(90 * DELTA_W / KA) / LOG(90 * DELTA_W / KSC)) ** 2) &
            * (((-1 + LOG(30 * H / KSC)) / (-1 + LOG(30 * H / KA))) ** 2)
    END IF
END IF
IF (ACW > 1) ACW = 1

TC = DWA * ACW * EFC * FC * (U ** 2) / 8.0  !Bed shear stress currents
TW = DWA * EFW * FW * (UD ** 2) / 4
TCW = TC + TW

!Current related shear velocity
CDU = SQRT(G) * ABS(U) / CHZC

!Wave related shear velocity
CW = SQRT(77.8 / FW)
WDU = SQRT(G) * UD / CW

!Critical Shields parameter based on d50 (Van Rijn, 2007a)
!and Soulsby (1997)
IF (DD50 < 4) THEN
    SPCRD50 = 0.115 / SQRT(DD50)
ELSE
    IF (DD50 < 10) THEN
        SPCRD50 = 0.14 / (DD50 ** 0.64)
    ELSE
        IF (DD50 < 20) THEN
            SPCRD50 = 0.04 / (DD50 ** 0.1)
        ELSE
            IF (DD50 < 150) THEN
                SPCRD50 = 0.013 * (DD50 ** 0.29)
            ELSE
                SPCRD50 = 0.055
            END IF
        END IF
    END IF
END IF
```

```
CGELS = 0.65
IF (D50 > DSAND) THEN
    CGEL = CGELS
ELSE
    CGEL = (D50 / DSAND) * CGELS
END IF
IF (CGEL < 0.05) CGEL = 0.05

!Critical bed shear stress based on d50,
!with adjustment based on biomass present
TBCRD500 = SPCRD50 * G * D50 * (DSED - DWA)
IF (D50 < DSAND) THEN
    PHI_COHESIVE = (DSAND / D50) ** 1.5
ELSE
    PHI_COHESIVE = 1
END IF

PHI_PACKING = CGEL / CGELS
IF (PHI_PACKING > 1) PHI_PACKING = 1

PCS = PI(1)
PHI_CLAY = (1 + PCS) ** 3
IF (PHI_CLAY > 2.0) PHI_CLAY = 2.0

TBCRD50 = PHI_CLAY * PHI_COHESIVE * PHI_PACKING * TBCRD500 * &
    (1 + (KVEG * BM))

IF (U < 1E-6) THEN
    ALPHA = 0
ELSE
    ALPHA = U / (U + UD)
END IF
BTA = 1
IF (AD < 1E-2) THEN
    FDW = 0
ELSE
    FDW = EXP(-6 + (5.2 / ((AD / D90) ** 0.19)))
END IF
FDCW = (ALPHA * BTA * FDC) + ((1 - ALPHA) * FDW)

ALPHA1 = (DSAND / D50) - 1
IF (D50 >= DSAND) THEN
    PHI_FS = 1
ELSE
    PHI_FS = D50 / (1.5 * DSAND)
END IF
IF (ALPHA1 < 0) ALPHA1 = 0
IF (ALPHA1 > 3) ALPHA1 = 3
ESC_MAX = 0.25 * KPA * H * CDU
IF (HS / H <= 0.4) THEN
    GAMMA_BR = 1
ELSE
    GAMMA_BR = 1 + SQRT((HS / H) - 0.4)
END IF


DELTA_S = 2 * GAMMA_BR * DELTA_W
IF (DELTA_S < 0.1) DELTA_S = 0.1
IF (DELTA_S > 0.5) DELTA_S = 0.5
ESW_MAX = 0.035 * GAMMA_BR * HS / TP
IF (ESW_MAX > 0.05) ESW_MAX = 0.05

IF (U > 1E-6) THEN
    !DEC$ PARALLEL
```

```
DO N = 2, NF + 1
    !2 Compute fraction dependant sediment characteristics
    DFI = FI(N) * ((((SD - 1.0) * G) / (KV ** 2)) ** (1.0 / 3.0))

    IF (FI(N) > DSAND) THEN
        FSILT = 1.0
    ELSE
        FSILT = DSAND / FI(N)
    END IF

    !Van Rijn, 2007c
    !Hiding - Exposure function, Egiazarof (1965)
    !If Di/D50 < 0.19 switch to power law (di/d50)^-1 (equal mobility)
    IF ((FI(N) / D50) > 0.19) THEN
        EI = ((LOG10(19.0) / LOG10(19.0 * FI(N) / D50))) ** 2
    ELSE
        EI = D50 / FI(N)
    END IF
    !LI is 'correction factor of effective grain shear stress'
    !(Van Rijn, 2007c)
    LI = (FI(N) / D50) ** 0.25


    !If percentage of fraction < 1% then treat as zero.
    IF (PI(N) < 0.01) THEN
        QB(N) = 0
        QS(N) = 0
    ELSE
        !Bed load (Van Rijn, 2007a,c)
        !Calculate instananeous velocities
        !using sine wave approximation
        DELTA_M = MAX(2 * DELTA_W, KSC)
        DELTA_V = U * LOG(30 * DELTA_M / KA) / (-1 + LOG(30 * H / KA))
        UBD = DELTA_V * LOG(30 * A / KSC) / LOG(30 * DELTA_M / KSC)
        VI1 = UBD
        VI2 = SQRT(((UBD + 0.707 * UD * COS(WAVE_DIR)) ** 2) + &
            ((0.707 * UD * SIN(WAVE_DIR)) ** 2))
        VI3 = SQRT(((UBD + UD * COS(WAVE_DIR)) ** 2) + &
            ((UD * SIN(WAVE_DIR)) ** 2))
        VI4 = SQRT(((UBD - UD * 0.707 * COS(WAVE_DIR)) ** 2) + &
            ((UD * 0.707 * SIN(WAVE_DIR)) ** 2))
        VI5 = SQRT(((UBD - UD * COS(WAVE_DIR)) ** 2) + &
            ((UD * SIN(WAVE_DIR)) ** 2))
        TBCWI1 = 0.5 * DWA * FDCW * (VI1 ** 2)
        TBCWI2 = 0.5 * DWA * FDCW * (VI2 ** 2)
        TBCWI3 = 0.5 * DWA * FDCW * (VI3 ** 2)
        TBCWI4 = 0.5 * DWA * FDCW * (VI4 ** 2)
        TBCWI5 = 0.5 * DWA * FDCW * (VI5 ** 2)

        ST_DR1 = 0
        IF (ABS(UBD + 0.707 * UD * COS(WAVE_DIR)) > 1E-6) THEN
            IF (UBD + 0.707 * UD * COS(WAVE_DIR) < 0) THEN
                ST_DR2 = ATAN(0.707 * UD * SIN(WAVE_DIR) / &
                    (UBD + 0.707 * UD * COS(WAVE_DIR))) + PII
            ELSE
                ST_DR2 = ATAN(0.707 * UD * SIN(WAVE_DIR) / &
                    (UBD + 0.707 * UD * COS(WAVE_DIR)))
            END IF
        ELSE
            ST_DR2 = PII / 2
        END IF
        IF (ABS(UBD + UD * COS(WAVE_DIR)) > 1E-6) THEN
            IF (UBD + UD * COS(WAVE_DIR) < 0) THEN
                ST_DR3 = ATAN(UD * SIN(WAVE_DIR) / &
                    (UBD + UD * COS(WAVE_DIR))) + PII
```

```
        ELSE
            ST_DR3 = ATAN(UD * SIN(WAVE_DIR) / &
                (UBD + UD * COS(WAVE_DIR)))
        END IF
    ELSE
        ST_DR3 = PII / 2
    END IF
    IF (ABS(UBD - 0.707 * UD * COS(WAVE_DIR)) > 1E-6) THEN
        IF (UBD - 0.707 * UD * COS(WAVE_DIR) < 0) THEN
            ST_DR4 = ATAN(-0.707 * UD * SIN(WAVE_DIR) / &
                (UBD - 0.707 * UD * COS(WAVE_DIR))) + PII
        ELSE
            ST_DR4 = ATAN(-0.707 * UD * SIN(WAVE_DIR) / &
                (UBD - 0.707 * UD * COS(WAVE_DIR)))
        END IF
    ELSE
        ST_DR4 = PII / 2
    END IF
    IF (ABS(UBD - UD * COS(WAVE_DIR)) > 1E-6) THEN
        IF (UBD - UD * COS(WAVE_DIR) < 0) THEN
            ST_DR5 = ATAN(-UD * SIN(WAVE_DIR) / &
                (UBD - UD * COS(WAVE_DIR))) + PII
        ELSE
            ST_DR5 = ATAN(-UD * SIN(WAVE_DIR) / &
                (UBD - UD * COS(WAVE_DIR)))
        END IF
    ELSE
        ST_DR5 = PII / 2
    END IF
    TI1 = LI * (TBCWI1 - (EI * (FI(N) / D50) * TBCRD50)) / &
        ((FI(N) / D50) * TBCRD50)
    IF (TI1 < 0) TI1 = 0
    TI2 = LI * (TBCWI2 - (EI * (FI(N) / D50) * TBCRD50)) / &
        ((FI(N) / D50) * TBCRD50)
    IF (TI2 < 0) TI2 = 0
    TI3 = LI * (TBCWI3 - (EI * (FI(N) / D50) * TBCRD50)) / &
        ((FI(N) / D50) * TBCRD50)
    IF (TI3 < 0) TI3 = 0
    TI4 = LI * (TBCWI4 - (EI * (FI(N) / D50) * TBCRD50)) / &
        ((FI(N) / D50) * TBCRD50)
    IF (TI4 < 0) TI4 = 0
    TI5 = LI * (TBCWI5 - (EI * (FI(N) / D50) * TBCRD50)) / &
        ((FI(N) / D50) * TBCRD50)
    IF (TI5 < 0) TI5 = 0
    QB1 = COS(ST_DR1) * 0.5 * FSILT * FI(N) * (DFI ** -0.3) * &
        (SQRT(TBCWI1 / DWA)) * TI1
    QB2 = COS(ST_DR2) * 0.5 * FSILT * FI(N) * (DFI ** -0.3) * &
        (SQRT(TBCWI2 / DWA)) * TI2
    QB3 = COS(ST_DR3) * 0.5 * FSILT * FI(N) * (DFI ** -0.3) * &
        (SQRT(TBCWI3 / DWA)) * TI3
    QB4 = COS(ST_DR4) * 0.5 * FSILT * FI(N) * (DFI ** -0.3) * &
        (SQRT(TBCWI4 / DWA)) * TI4
    QB5 = COS(ST_DR5) * 0.5 * FSILT * FI(N) * (DFI ** -0.3) * &
        (SQRT(TBCWI5 / DWA)) * TI5
    QB(N) = PI(N) * ((2 * QB1) + (2 * QB2) + &
        QB3 + (2 * QB4) + QB5) / 8

    !Suspended sediment transport
    !Compute the bed-shear stress parameter, T
    !(Multi-fraction method A)
    TI = LI * (TCW - (EI * (FI(N) / D50) * TBCRD50)) / &
        ((FI(N) / D50) * TBCRD50)
    IF (TI < 0) TI = 0

    !Reference concentration (Van Rijn, 2007c)
```

```
CA = 0.015 * FSILT * (FI(N) / A) * &
    (DFI ** (-0.3)) * (TI ** 1.5)
IF (CA > 0.05) CA = 0.05

QS(N) = 0
IF (CA > 1E-9) THEN
    Z = A
    SC = CA
    BYSC = .TRUE.
    !Integration
    DO WHILE (Z + 0.001 < H .AND. SC > 1E-15)

        !Fall velocity

        IF (2 * SC / CGEL < 0.00011) THEN
            PHI_FLOC = 1
        ELSE
            PHI_FLOC = (4 + LOG10(2 * SC / CGEL)) ** ALPHA1
        END IF
        IF (PHI_FLOC < 1) PHI_FLOC = 1
        IF (PHI_FLOC > 10) PHI_FLOC = 10

        PHI_HS = (1 - 0.65 * SC / CGEL) ** 5

        PHI_D = PHI_FS * (1 + ((SC / CGELS) ** 0.8) - &
            (2 * (SC / CGELS) ** 0.4))

        WSI = WS(N) * PHI_FLOC * PHI_HS

        !Beta factor currents
        IF (CDU < 1E-9) THEN
            BTA_C = 1.5
        ELSE
            IF ((WSI / CDU) < 0.01) THEN
                BTA_C = 1.0
            ELSE
                IF (WSI / CDU > 0.5) THEN
                    BTA_C = 1.5
                ELSE
                    BTA_C = 1.0 + (2.0 * ((WSI / CDU) ** 2))
                END IF
            END IF
        END IF

        !Beta factor waves
        IF (WDU < 1E-9) THEN
            BTA_W = 1.5
        ELSE
            IF ((WSI / WDU) < 0.01) THEN
                BTA_W = 1.0
            ELSE
                IF (WSI / WDU > 0.5) THEN
                    BTA_W = 1.5
                ELSE
                    BTA_W = 1.0 + (2.0 * ((WSI / WDU) ** 2))
                END IF
            END IF
        END IF

        IF (Z < H / 2) THEN
            EFC = 4 * (Z / H) * (1 - (Z / H)) * ESC_MAX
        ELSE
            EFC = ESC_MAX
        END IF
        ESC = PHI_D * BTA_C * EFC
```

```
ESW_BED = 0.018 * GAMMA_BR * BTA_W * DELTA_S * UD
IF (ESW_BED > ESW_MAX) ESW_BED = ESW_MAX

IF (Z < DELTA_S) THEN
    ESW = ESW_BED
ELSE
    IF (Z > H / 2) THEN
        ESW = ESW_MAX
    ELSE
        ESW = ESW_BED + (ESW_MAX - ESW_BED) * &
            (Z - DELTA_S) / ((0.5 * H) - DELTA_S)
    END IF
END IF
ESCW = SQRT((ESC ** 2) + (ESW ** 2))

DCDZ = -(SC * WSI) / ESCW

IF (BYSC .AND. DCDZ < -SC / (H - Z)) THEN
    IF (SC > 1E-6) THEN
        DC = -SC / 2
    ELSE
        DC = -SC
    END IF
    DZ = DC / DCDZ
    IF (DZ > H - Z) THEN
        DZ = H - Z
        DC = DZ * DCDZ
    END IF
ELSE
    BYSC = .FALSE.
    IF (Z + (H / 5) < H) THEN
        DZ = H / 5
    ELSE
        DZ = H - Z
    END IF
    DC = DZ * DCDZ
    IF (SC + DC < 0) THEN
        DC = -SC
        DZ = DC / DCDZ
    END IF
END IF
DC1 = DC / 2
DZ1 = DZ / 2

SC05 = SC + DC1
Z05 = Z + DZ1

!Fall velocity
IF (2 * SC05 / CGEL < 0.00011) THEN
    PHI_FLOC = 1
ELSE
    PHI_FLOC = (4 + LOG10(2 * SC05 / CGEL)) ** ALPHA1
END IF
IF (PHI_FLOC < 1) PHI_FLOC = 1
IF (PHI_FLOC > 10) PHI_FLOC = 10

PHI_HS = (1 - 0.65 * SC05 / CGEL) ** 5

PHI_D = PHI_FS * (1 + ((SC05 / CGELS) ** 0.8) - &
    (2 * (SC05 / CGELS) ** 0.4))

WSI = WS(N) * PHI_FLOC * PHI_HS

!Beta factor currents
IF (CDU < 1E-9) THEN
```

```
    BTA_C = 1.5
ELSE
    IF ((WSI / CDU) < 0.01) THEN
        BTA_C = 1.0
    ELSE
        IF (WSI / CDU > 0.5) THEN
            BTA_C = 1.5
        ELSE
            BTA_C = 1.0 + (2.0 * ((WSI / CDU) ** 2))
        END IF
    END IF
END IF

!Beta factor waves
IF (WDU < 1E-9) THEN
    BTA_W = 1.5
ELSE
    IF ((WSI / WDU) < 0.01) THEN
        BTA_W = 1.0
    ELSE
        IF (WSI / WDU > 0.5) THEN
            BTA_W = 1.5
        ELSE
            BTA_W = 1.0 + (2.0 * ((WSI / WDU) ** 2))
        END IF
    END IF
END IF


EFC = 4 * (Z05 / H) * (1 - (Z05 / H)) * ESC_MAX
ESC = PHI_D * BTA_C * EFC

IF (Z05 < DELTA_S) THEN
    ESW = ESW_BED
ELSE
    IF (Z05 > H / 2) THEN
        ESW = ESW_MAX
    ELSE
        ESW = ESW_BED + (ESW_MAX - ESW_BED) * &
            (Z05 - DELTA_S) / ((0.5 * H) - DELTA_S)
    END IF
END IF
ESCW = SQRT((ESC ** 2) + (ESW ** 2))

DCDZ1 = -(SC05 * WSI) / ESCW

IF (BYSC) THEN
    IF (DCDZ1 < - SC / (H - Z)) THEN
        DZ = DC / DCDZ1
    ELSE
        BYSC = .FALSE.
        DSDC_FR = -((SC / (H - Z)) + &
            DCDZ1) / (DCDZ - DCDZ1)
        DC = DC * (1 - DSDC_FR)
        DZ = -DC * H / CA
        IF (Z + DZ > H) THEN
            DZ = H - DZ
            DC = DZ * SC / H
        END IF
    END IF
ELSE
    DC = DCDZ1 * DZ
    IF (SC + DC < 0) THEN
        DC = -SC
        DZ = DC / DCDZ1
```

```
                    END IF
                END IF
                SC1 = SC + DC
                Z1 = Z + DZ

                SC2 = SC + (DZ * DCDZ)
                IF (SC2 < 0) SC2 = 0


                UZ = U * LOG(30 * Z / KSC) / (LOG(30 * H / KSC) - 1)
                UZ1 = U * LOG(30 * Z1 / KSC) / &
                    (LOG(30 * H / KSC) - 1)

                QS(N) = QS(N) + (PI(N) * ((SC + SC1) / 2) * DZ * &
                    (UZ + UZ1) / 2)

                Z = Z1
                SC = SC1

            END DO
        END IF
    END IF
  END DO
ELSE
  DO N = 2, NF + 1
    QS(N) = 0
    QB(N) = 0
  END DO
END IF

END SUBROUTINE

!----------------------------------------------------------------------

SUBROUTINE CALC_ST_MF(GRID, ALYR, SED, FGRID, ROWS, COLS, GRID_SIZE, &
    LYR_TH, ALYR_TH, NF, BASE, NR_LAYERS, FI, FMIN, FMAX, STYPES, &
    SED_IN, DSSC, DS, DMUD, PS, TSS, QX, QY, H, H_MIN, MEAN_H, &
    WAVEH, WAVET, WAVE_DIR, BM_MAX, BM, KB, LTFR, VX, VY, TSPRP, KD, &
    TD, TE, MF_SED_IN, MF_SED_OUT, DTOT, ETOT, MF_SSVOL, NWS2, P, SC)


!Array to to store x direction flow velocity at each cell
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:),   INTENT(INOUT) :: GRID
!Current sediment fractions in the active layer
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:,:), INTENT(INOUT) :: ALYR
!Sub-layer composition
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:,:,:),INTENT(INOUT) :: SED
!Mean water depth in each cell
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:), INTENT(IN)    :: MEAN_H
!Sediment input (mass balance)
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:),INTENT(INOUT) :: MF_SED_IN
!Sediment output (mass balance)
REAL (SELECTED_REAL_KIND(15,307)),DIMENSION(:),INTENT(INOUT) :: MF_SED_OUT
!Suspended sediment volume (mass balance)
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:), INTENT(INOUT) :: MF_SSVOL
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:),   INTENT(INOUT) :: DTOT
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:),   INTENT(INOUT) :: ETOT
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:),   INTENT(INOUT) :: SC

!Minimum bed levels (no erosion below fgrid)
REAL,  DIMENSION(:,:),     INTENT(IN)    :: FGRID
!X direction flow at each cell (right side)
REAL,   DIMENSION(:,:),     INTENT(IN)    :: QX
!Y direction flow at each cell
REAL,   DIMENSION(:,:),     INTENT(IN)    :: QY
```

```
!Sediment fraction size (median)
REAL,   DIMENSION(:),       INTENT(IN)    :: FI
!Sediment fraction size (min)
REAL,   DIMENSION(:),       INTENT(IN)    :: FMIN
!Sediment fraction size (max)
REAL,   DIMENSION(:),       INTENT(IN)    :: FMAX
!Water depth at each cell
REAL,  DIMENSION(:,:),     INTENT(IN)    :: H
!Biomass at each cell as fraction of max biomass
REAL,  DIMENSION(:,:),     INTENT(INOUT) :: BM
REAL,  DIMENSION(:,:,:),   INTENT(IN)    :: WAVEH      !Wave height
REAL,  DIMENSION(:,:,:),   INTENT(IN)    :: WAVET      !Wave period
REAL,  DIMENSION(:,:,:),   INTENT(IN)    :: WAVE_DIR   !Wave direction
!X direction flow velocity at each cell
REAL,   DIMENSION(:,:),     INTENT(OUT)   :: VX
!Y direction flow velocity at each cell
REAL,   DIMENSION(:,:),     INTENT(OUT)   :: VY
!Fraction proportions for each sediment type
REAL,   DIMENSION(:,:),     INTENT(IN)    :: STYPES
!Sediment input at upstream & downstream boundary
REAL,   DIMENSION(:,:),     INTENT(IN)    :: SED_IN
REAL,  DIMENSION(:),       INTENT(IN)    :: P

!Active layer thickness (m)
REAL (SELECTED_REAL_KIND(15,307)), INTENT(IN) :: ALYR_TH
!Thickness of sublayers (m)
REAL (SELECTED_REAL_KIND(15,307)), INTENT(IN) :: LYR_TH

REAL, INTENT(IN) :: DS       !Sediment density (~2650kg/m3)
REAL, INTENT(IN) :: DMUD     !Bulk density of mud
REAL, INTENT(IN) :: PS       !Sediment porosity (~0.4)
REAL, INTENT(IN) :: TSS      !Time step in seconds
REAL, INTENT(IN) :: GRID_SIZE !Grid cell size
REAL, INTENT(IN) :: BASE     !Lowest level for sublayers
REAL, INTENT(IN) :: BM_MAX   !Max biomass (marsh)
REAL, INTENT(IN) :: LTFR     !Lateral transport parameter
REAL, INTENT(IN) :: KB       !Organogenic sediment production (m/yr)
REAL, INTENT(IN) :: KD       !Diffusion coefficient
REAL, INTENT(IN) :: TD       !Critical bed shear stress for deposition
REAL, INTENT(IN) :: TE       !Critical bed shear stress for erosion of mud
REAL, INTENT(IN) :: H_MIN    !Minimum water depth
!Downstream boundary suspended sediment concentration
REAL, INTENT(IN) :: DSSC

INTEGER, INTENT(IN) :: ROWS         !Total number of rows
INTEGER, INTENT(IN) :: COLS         !Total number of columns
INTEGER, INTENT(IN) :: NF           !Number of sediment fractions
INTEGER, INTENT(IN) :: NR_LAYERS    !Number of sub-layers
!Nr. of time steps per tidal cycle or spring/neap period
INTEGER, INTENT(IN) :: TSPRP
INTEGER, INTENT(IN) :: NWS2

!Bed load in current direction
REAL,   DIMENSION(:),       ALLOCATABLE :: BLC
!Suspended load due to current
REAL,   DIMENSION(:),       ALLOCATABLE :: SLC
REAL,   DIMENSION(:),       ALLOCATABLE :: BLCM
REAL,   DIMENSION(:),       ALLOCATABLE :: SLCM

REAL  (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:,:), &
    ALLOCATABLE, SAVE :: SDEP     !Suspended sediment deposition layer
REAL  (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:,:), &
    ALLOCATABLE       :: SDEP1    !Suspended sediment deposition layer

REAL,   DIMENSION(:,:),     ALLOCATABLE :: CV       !Magnitude of velocity
```

```
!Unsigned (+ve) x component of velocity
REAL,    DIMENSION(:,:),      ALLOCATABLE :: CVX
!Unsigned (+ve) y component of velocity
REAL,    DIMENSION(:,:),      ALLOCATABLE :: CVY
!Proportion of each sediment fraction in bed
REAL,    DIMENSION(:),        ALLOCATABLE :: SPR

!Positive outflows at cell boundaries
!(set to zero if flow is into the cell)
!Positive flow across cell boundary (to row + 1)
REAL,    DIMENSION(:,:),      ALLOCATABLE :: QF
!Positive flow across cell boundary (to row - 1)
REAL,    DIMENSION(:,:),      ALLOCATABLE :: QB
!Positive flow across cell boundary (to col - 1)
REAL,    DIMENSION(:,:),      ALLOCATABLE :: QL
!Positive flow across cell boundary (to col + 1)
REAL,    DIMENSION(:,:),      ALLOCATABLE :: QR
REAL,    DIMENSION(:,:),      ALLOCATABLE :: QT   !Sum of boundary outflows

!Lateral transport sediment depth change
!Forward (to row + 1)
REAL (SELECTED_REAL_KIND(15,307)),   DIMENSION(:), ALLOCATABLE :: DLTF
!Backward (to row - 1)
REAL (SELECTED_REAL_KIND(15,307)),   DIMENSION(:), ALLOCATABLE :: DLTB
!Left (to col - 1)
REAL (SELECTED_REAL_KIND(15,307)),   DIMENSION(:), ALLOCATABLE :: DLTL
!Right (to col + 1)
REAL (SELECTED_REAL_KIND(15,307)),   DIMENSION(:), ALLOCATABLE :: DLTR
!Total (sum of DLTF, DLTB, DLTL, DLTR)
REAL (SELECTED_REAL_KIND(15,307)),   DIMENSION(:), ALLOCATABLE :: DLTT

!Depth change due to suspended sediment transport
!Forward (to row + 1)
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:), ALLOCATABLE :: SDLF
!Backward (to row - 1)
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:), ALLOCATABLE :: SDLB
!Left (to col - 1)
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:), ALLOCATABLE :: SDLL
!Right (to col + 1)
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:), ALLOCATABLE :: SDLR
!Total (sum of SDLF, SDLB, SDLL, SDLR)
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:), ALLOCATABLE :: SDLT

!Depth change due to bed load transport
!Forward (to row + 1)
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:), ALLOCATABLE :: BDLF
!Backward (to row - 1)
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:), ALLOCATABLE :: BDLB
!Left (to col - 1)
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:), ALLOCATABLE :: BDLL
!Right (to col + 1)
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:), ALLOCATABLE :: BDLR
!Total (sum of BDLF, BDLB, BDLL, BDLR)
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:), ALLOCATABLE :: BDLT

!Depth of each sediment fraction in the active layer (m)
REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:,:), ALLOCATABLE :: ALYR1

REAL (SELECTED_REAL_KIND(15,307)),   DIMENSION(:),   ALLOCATABLE :: MFTOT
REAL (SELECTED_REAL_KIND(15,307)),   DIMENSION(:),   ALLOCATABLE :: MFTOT1
REAL (SELECTED_REAL_KIND(15,307)),   DIMENSION(:),   ALLOCATABLE :: MFIN
REAL (SELECTED_REAL_KIND(15,307)),   DIMENSION(:),   ALLOCATABLE :: MFOUT

!Angle between wave and current directions
REAL,    DIMENSION(:,:,:),   ALLOCATABLE :: WC_DIR
```

```
!Critical bed shear stress
REAL,    DIMENSION(:,:),     ALLOCATABLE :: TBCR
!Maximum bed shear stress under combined currents and waves
REAL,    DIMENSION(:,:,:),   ALLOCATABLE :: TMX
!Bed levels at start of timestep
REAL,    DIMENSION(:,:),     ALLOCATABLE :: SGRID
!Cumulative sediment inputs (for time averaging)
REAL,    DIMENSION(:), SAVE, ALLOCATABLE :: DS_SED_IN
!Scaling factor for boundary sediment inputs
REAL,    DIMENSION(:), SAVE, ALLOCATABLE :: DSSF
!Particle fall velocity
REAL,    DIMENSION(:), SAVE, ALLOCATABLE :: WS

REAL,    DIMENSION(:),       ALLOCATABLE       :: QTC
REAL,    DIMENSION(:),       ALLOCATABLE       :: TMXTOTR
REAL,    DIMENSION(:),       ALLOCATABLE       :: PI1

!Depth of sediment transported by diffusion in 4 directions
!(equivalent depth of settled bed)
REAL (SELECTED_REAL_KIND(15,307)) DDF, DDB, DDL, DDR
!Fraction of suspended sediment transported by diffusion,
!fraction not transported
REAL (SELECTED_REAL_KIND(15,307)) FD, FD1
REAL (SELECTED_REAL_KIND(15,307)) HRF, HRB, HRL, HRR !Depth ratios
!Max. proportion of suspended sediment transported
!between 2 cells by diffusion in one timestep
REAL (SELECTED_REAL_KIND(15,307)) FDMAX
REAL (SELECTED_REAL_KIND(15,307)) EM1, EMM !Mud erosion depth
REAL (SELECTED_REAL_KIND(15,307)) DEP1, DEP2, DEPM

REAL (SELECTED_REAL_KIND(15,307)) MFDIFF
!Total amount of sediment in active layer, change in bed level
REAL (SELECTED_REAL_KIND(15,307)) ALYR_TOT, DL
!Fraction of sublayer to be updated
REAL (SELECTED_REAL_KIND(15,307)) SUB_LYR_FR
REAL (SELECTED_REAL_KIND(15,307)) DF, DSED

REAL, SAVE :: CELL_AREA !Plan area of one cell
REAL DLWH               !Depth limited wave height/H (breaker index)
REAL HS, TP             !Significant wave height/period
REAL MIN_HS, MIN_TP     !Minimum values for HS and TP (wave height/period)
REAL MIN_V, MAX_V       !Minimum/maximum velocity
!Suspended sediment transport (currents) in each of 4 grid directions
REAL SQTCF, SQTCB, SQTCL, SQTCR
!Bed load transport (currents) in each of 4 grid directions
REAL BQTCF, BQTCB, BQTCL, BQTCR
!Total amount (m) of bed material to be moved (suspended/bed load/total)
REAL SDL_TOT, BDL_TOT, DL_TOT
REAL BDL                !Bed level difference
REAL FR                 !Fraction of suspended sediment to be deposited
REAL MAXE         !Max erosion allowed where limited by minimum bed level
REAL SD           !Specific density of sediment
REAL QTOT         !Total flow in boundary row cells
!Lateral transport bed level change, total bed level change
REAL DLL, DLR, DLTTT
REAL CURR_DIR                                   !Current direction
REAL DUM1, DUM2, DUM3                           !Dummy subroutine argument
REAL NST, HST, DST, SED_TRP, FRD, FRSM, BMS     !Salt marsh parameters
REAL DSAND, P_MUD     !Minimum sand grain size, proportion of mud
!Total sediment transported to each of the 4 neighbouring cells
REAL DLF_TOT, DLB_TOT, DLL_TOT, DLR_TOT
REAL QTD                     !Calculated sand transport at boundary
!Sum of boundary cell shear stresses, depth of sand transport at boundary
REAL TMX_TOT, QM_SAND
REAL TDI             !Critical shear stress for deposition for fraction i
```

```
!Dimensionless particle size parameter, median particle size,
!d90 particle size
REAL DSTR, D50, D90
REAL QTTOT, QTM
REAL SEDINTOT, DSILT
REAL, SAVE :: MT   !Time from last organogenic sediment addition (seconds)

INTEGER ROW, COL   !Row/Column counters
INTEGER N, M       !Counters
INTEGER SUB_LYR_NR !Array index of sub-layer to be updated
INTEGER NL         !Number of sub-layers to be updated
INTEGER AA, RW, CL

INTEGER, SAVE :: NMUD     !Number of sediment fractions treated as mud
!Timestep number used for time-averaged advection calculation
INTEGER, SAVE :: TSN
INTEGER, SAVE :: TM       !Time step number
!Timestep number from start of model run (upto TRPRP)
INTEGER, SAVE :: TS_COUNT

LOGICAL :: FIRST_EXEC = .TRUE.

!True of boundary sediment input distributed by shear stress
LOGICAL, DIMENSION(:), ALLOCATABLE :: DTMX

ALLOCATE (SPR(NF + 1))
ALLOCATE (CV(ROWS, COLS))
ALLOCATE (CVX(ROWS, COLS))
ALLOCATE (CVY(ROWS, COLS))
ALLOCATE (BLCM(NF + 1))
ALLOCATE (SLCM(NF + 1))
ALLOCATE (BLC(NF + 1))
ALLOCATE (SLC(NF + 1))
ALLOCATE (QF(ROWS, COLS))
ALLOCATE (QB(ROWS, COLS))
ALLOCATE (QL(ROWS, COLS))
ALLOCATE (QR(ROWS, COLS))
ALLOCATE (QT(ROWS, COLS))
ALLOCATE (WC_DIR(ROWS, COLS, NWS2))
ALLOCATE (DLTF(NF + 1))
ALLOCATE (DLTB(NF + 1))
ALLOCATE (DLTL(NF + 1))
ALLOCATE (DLTR(NF + 1))
ALLOCATE (DLTT(NF + 1))
ALLOCATE (TBCR(ROWS, COLS))
ALLOCATE (TMX(ROWS, COLS, NWS2))
ALLOCATE (ALYR1(ROWS, COLS, NF + 1))
ALLOCATE (SDLF(NF + 1))
ALLOCATE (SDLB(NF + 1))
ALLOCATE (SDLL(NF + 1))
ALLOCATE (SDLR(NF + 1))
ALLOCATE (BDLF(NF + 1))
ALLOCATE (BDLB(NF + 1))
ALLOCATE (BDLL(NF + 1))
ALLOCATE (BDLR(NF + 1))
ALLOCATE (SDLT(NF + 1))
ALLOCATE (BDLT(NF + 1))
ALLOCATE (DTMX(NF))
ALLOCATE (SGRID(ROWS, COLS))
ALLOCATE (QTC(COLS))
ALLOCATE (TMXTOTR(COLS))
ALLOCATE (PI1(NF + 1))

ALLOCATE (MFTOT(NF + 1))
ALLOCATE (MFTOT1(NF + 1))
```

```
ALLOCATE (MFIN(NF + 1))
ALLOCATE (MFOUT(NF + 1))

DSAND = 0.000062
SD = DS / DWA    !Specific density
SGRID = GRID

IF (FIRST_EXEC) THEN
    FIRST_EXEC = .FALSE.
    MT = 0
    TS_COUNT = 1

    NMUD = 1

    ALLOCATE (WS(NF + 1))
    ALLOCATE (DS_SED_IN(NF - NMUD + 1))
    ALLOCATE (DSSF(NF - NMUD + 1))

    TSN = 1

    SDEP = 0
    CELL_AREA = GRID_SIZE * GRID_SIZE

    !Calculate particle fall velocities
    DO N = 2, NF + 1
        IF (FI(N) < 0.0001) THEN
            WS(N) = (1.0 / 18.0) * (SD - 1.0) * G * (FI(N) ** 2) / KV
        ELSE
            IF (FI(N) < 0.001) THEN
                WS(N) = 10.0 * (KV / FI(N)) * (((1.0 + ((0.01 * &
                    (SD - 1.0) * G * (FI(N) ** 3.0)) / &
                    (KV ** 2))) ** 0.5) - 1.0)
            ELSE
                WS(N) = 1.1 * (((SD - 1) * G * FI(N)) ** 0.5)
            END IF
        END IF
    END DO
    WS(1) = 1.3E-5
    DS_SED_IN = 0
    DSSF = 0
    DTMX = .FALSE.

END IF

ALLOCATE (SDEP1(ROWS, COLS, NMUD))

IF (TSN == TSPRP) THEN
    TSN = 1
ELSE
    TSN = TSN + 1
END IF

SDEP1 = 0
MIN_HS = 0.01
MIN_TP = 1
DLWH = 0.78
SDEP1 = 0
MAX_V = 2.0
MIN_V = 0.01

!Calculate velocity for each cell, based on positive
!outflow values at cell boundary
DO ROW = 1, ROWS
    DO COL = 1, COLS
        IF (H(ROW, COL) > H_MIN) THEN
```

```
        IF (COL == 1) THEN
            VX(ROW, COL) = QX(ROW, COL) / &
            (2.0 * H(ROW, COL) * GRID_SIZE)
        ELSE
            IF (QX(ROW, COL) > 0) THEN
                IF (QX(ROW, COL - 1) < 0) THEN
                    IF (ABS(QX(ROW, COL)) > &
                        ABS(QX(ROW, COL - 1))) THEN
                        VX(ROW, COL) = QX(ROW, COL) / &
                        (H(ROW, COL) * GRID_SIZE)
                    ELSE
                        VX(ROW, COL) = QX(ROW, COL - 1) / &
                        (H(ROW, COL) * GRID_SIZE)
                    END IF
                ELSE
                    VX(ROW, COL) = QX(ROW, COL) / &
                    (H(ROW, COL) * GRID_SIZE)
                END IF
            ELSE
                IF (QX(ROW, COL - 1) < 0) THEN
                    VX(ROW, COL) = QX(ROW, COL - 1) / &
                    (H(ROW, COL) * GRID_SIZE)
                ELSE
                    VX(ROW, COL) = 0
                END IF
            END IF
        END IF

        IF (ROW == 1) THEN
            VY(ROW, COL) = QY(ROW, COL) / (H(ROW, COL) * GRID_SIZE)
        ELSE
            IF (QY(ROW, COL) > 0) THEN
                IF (QY(ROW - 1, COL) < 0) THEN
                    IF (ABS(QY(ROW, COL)) > &
                        ABS(QY(ROW - 1, COL))) THEN
                        VY(ROW, COL) = QY(ROW, COL) / &
                            (H(ROW, COL) * GRID_SIZE)
                    ELSE
                        VY(ROW, COL) = QY(ROW - 1, COL) / &
                            (H(ROW, COL) * GRID_SIZE)
                    END IF
                ELSE
                    VY(ROW, COL) = QY(ROW, COL) / &
                        (H(ROW, COL) * GRID_SIZE)
                END IF
            ELSE
                IF (QY(ROW - 1, COL) < 0) THEN
                    VY(ROW, COL) = QY(ROW - 1, COL) / &
                        (H(ROW, COL) * GRID_SIZE)
                ELSE
                    VY(ROW, COL) = 0
                END IF
            END IF
        END IF

        CV(ROW, COL) = SQRT((VX(ROW, COL) ** 2) + (VY(ROW, COL) ** 2))
        CVX(ROW, COL) = ABS(VX(ROW, COL))
        CVY(ROW, COL) = ABS(VY(ROW, COL))
    ELSE
        VX(ROW, COL) = 0
        VY(ROW, COL) = 0
        CV(ROW, COL) = 0
        CVX(ROW, COL) = 0
        CVY(ROW, COL) = 0
    END IF
```

```
        IF (CV(ROW, COL) > MAX_V) THEN
            VX(ROW, COL) = MAX_V * VX(ROW, COL) / CV(ROW, COL)
            VY(ROW, COL) = MAX_V * VY(ROW, COL) / CV(ROW, COL)
            CV(ROW, COL) = MAX_V
            CVX(ROW, COL) = ABS(VX(ROW, COL))
            CVY(ROW, COL) = ABS(VY(ROW, COL))
        END IF
    END DO
END DO


!***********************************************************************
!******************** Sediment Transport Calculation *******************
!***********************************************************************
NMUD  = 1
ALYR1 = 0

DO ROW = 1, ROWS
    DO COL = 1, COLS

        DO N = 1, NF + 1
            SPR(N) = ALYR(ROW, COL, N) / ALYR_TH
        END DO


        IF (H(ROW, COL) > H_MIN .AND. CV(ROW, COL) > MIN_V) THEN

            !Calculate angle between wave and current direction
            IF (CVY(ROW, COL) < 1E-6) THEN
                IF (VX(ROW, COL) > 0) THEN
                    CURR_DIR = PII / 2
                ELSE
                    CURR_DIR = -PII / 2
                END IF
            ELSE
                IF (VY(ROW, COL) > 0) THEN
                    CURR_DIR = ATAN(VX(ROW, COL) / VY(ROW, COL))
                ELSE
                    CURR_DIR = ATAN(VX(ROW, COL) / VY(ROW, COL)) + &
                        (PII / 2)
                END IF
            END IF

            IF (SPR(1) < 0.3) THEN
                !If proportion of clay in active layer < 30%, use
                !Van Rijn TR2004 method to calculate sand/silt transport

                !Calculate sediment transport based on combined
                !composition of active layer bed load and suspended load

                SLC = 0
                BLC = 0
                EM1 = 0
                DO M = 1, NWS2
                    WC_DIR(ROW, COL, M) = WAVE_DIR(ROW, COL, M) - CURR_DIR
                    HS = WAVEH(ROW, COL, M)
                    IF (HS < MIN_HS) HS = MIN_HS
                    TP = WAVET(ROW, COL, M)
                    IF (TP < MIN_TP) TP = MIN_TP

                    CALL VAN_RIJN_MF(FI, WS, FMIN, FMAX, SPR(:), NF, DS, &
                            H(ROW, COL), HS, TP, WC_DIR(ROW, COL, M), &
                            CV(ROW, COL), SLCM, BLCM, BM(ROW, COL), &
                            TBCR(ROW, COL))
```

```
            SLC = SLC + (SLCM * P(M))
            BLC = BLC + (BLCM * P(M))


           !Mud erosion rate only used where clay proportion > 20%
           !Mud erosion procedure also used to calculate max bed
           !shear stress
            CALL MUD_ER(CV(ROW, COL), HS, TP, &
                WC_DIR(ROW, COL, M), H(ROW, COL), SPR(1), DMUD, &
                TBCR(ROW, COL), 0.0, 0.0, BM(ROW, COL), TSS, &
                EMM, TMX(ROW, COL, M), DUM1, .TRUE.)

            EM1 = EM1 + (EMM * P(M))

        END DO

        SLC(1) = 0
        BLC(1) = 0


        IF (SPR(1) > 0.2) THEN
            !For clay proportion between 20% and 30% use both
            !TR2004 and mud erosion methods With linear weighting
            !between 100% TR2004 at 20% clay and 100% mud erosion
            !at 30% clay
            P_MUD = 0
            DO N = 1, NMUD
                P_MUD = P_MUD + SPR(N)
            END DO
            EM1 = EM1 * 10.0 * (SPR(1) - 0.2)

            SLC = SLC * 10.0 * (0.3 - SPR(1))
            BLC = BLC * 10.0 * (0.3 - SPR(1))

            !Clay fraction assumed to be brought into suspension
            !due to the transport of other fractions
            DO N = 2, NF + 1
                SLC(1) = SLC(1) + (SPR(1) * (SLC(N) + BLC(N)))
            END DO

            !Clay and silt fractions suspended according to mud
            !erosion equation
            DO N = 1, NMUD
                SLC(N) = SLC(N) + ((SPR(N) / P_MUD) * EM1 * &
                    (1 - PS) * GRID_SIZE / TSS)
            END DO

        ELSE
            !Clay fraction assumed to be brought into suspension
            !only due to the transport of other fractions
            DO N = 2, NF + 1
                SLC(1) = SLC(1) + (SPR(1) * (SLC(N) + BLC(N)))
            END DO

        END IF

    ELSE
        !For clay proportion > 30% only mud erosion
        !equation is used
        !TB_CRIT function used to calculate critical bed shear
        !stress used the TR2004 method
        IF (SPR(1) < 0.999) THEN
            FR = (SPR(1) - 0.3) / 0.7
            TBCR(ROW, COL) = ((1.0 - FR) * TB_CRIT(FI, DS, FMIN, &
                FMAX, SPR, 0.3, NF, BM(ROW, COL))) + (FR * TE)
```

```
        ELSE
            TBCR(ROW, COL) = TE
        END IF

        EM1 = 0
        DO M = 1, NWS2

            WC_DIR(ROW, COL, M) = WAVE_DIR(ROW, COL, M) - CURR_DIR
            HS = WAVEH(ROW, COL, M)
            IF (HS < MIN_HS) HS = MIN_HS
            TP = WAVET(ROW, COL, M)
            IF (TP < MIN_TP) TP = MIN_TP

            CALL MUD_ER(CV(ROW, COL), HS, TP, &
                WC_DIR(ROW, COL, M), H(ROW, COL), SPR(1), DMUD, &
                TBCR(ROW, COL), 0.0, 0.0, BM(ROW, COL), TSS, &
                EMM, TMX(ROW, COL, M), DUM1, .TRUE.)

            EM1 = EM1 + (EMM * P(M))

        END DO



        !Calculate total mud proportion (clay + silt fractions)
        P_MUD = 0
        DO N = 1, NMUD
            P_MUD = P_MUD + SPR(N)
        END DO

        !Clay and silt fractions suspended according to mud
        !erosion equation
        DO N = 1, NMUD
            BLC(N) = 0
            SLC(N) = (SPR(N) / P_MUD) * EM1 * (1 - PS) * &
                GRID_SIZE / TSS
        END DO

        !Sand fraction transport set to zero
        IF (NMUD < NF + 1) THEN
            DO N = NMUD + 1, NF + 1
                BLC(N) = 0
                SLC(N) = 0
            END DO
        END IF

    END IF

    !Calculate boundary outflows to 4 neighbouring cells
    IF (QY(ROW, COL) < 0) THEN
        QB(ROW, COL) = -QY(ROW, COL)
    ELSE
        QB(ROW, COL) = 0
    END IF

    IF (ROW < ROWS) THEN
        IF (QY(ROW + 1, COL) > 0) THEN
            QF(ROW, COL) = QY(ROW + 1, COL)
        ELSE
            QF(ROW, COL) = 0
        END IF
    ELSE
        IF (QY(ROW, COL) > 0) THEN
            QF(ROW, COL) = QY(ROW, COL)
        ELSE
```

```
            QF(ROW, COL) = 0
        END IF
    END IF
    IF (COL > 1) THEN
        IF (QX(ROW, COL - 1) < 0) THEN
            QL(ROW, COL) = -QX(ROW, COL - 1)
        ELSE
            QL(ROW, COL) = 0
        END IF
    ELSE
        QL(ROW, COL) = 0
    END IF
    IF (QX(ROW, COL) > 0) THEN
        QR(ROW, COL) = QX(ROW, COL)
    ELSE
        QR(ROW, COL) = 0
    END IF
    QT(ROW, COL) = QB(ROW, COL) + QF(ROW, COL) + QL(ROW, COL) + &
        QR(ROW, COL)

    !Calculate sediment depths transported to 4 neighbouring cells
    DLF_TOT = 0
    DLB_TOT = 0
    DLL_TOT = 0
    DLR_TOT = 0
    DO N = 1, NF + 1

        !Determine transport due to currents in each direction
        IF (QT(ROW, COL) > 1E-6) THEN
            SQTCF = (QF(ROW, COL) / QT(ROW, COL)) * SLC(N)
            SQTCB = (QB(ROW, COL) / QT(ROW, COL)) * SLC(N)
            SQTCL = (QL(ROW, COL) / QT(ROW, COL)) * SLC(N)
            SQTCR = (QR(ROW, COL) / QT(ROW, COL)) * SLC(N)
            BQTCF = (QF(ROW, COL) / QT(ROW, COL)) * BLC(N)
            BQTCB = (QB(ROW, COL) / QT(ROW, COL)) * BLC(N)
            BQTCL = (QL(ROW, COL) / QT(ROW, COL)) * BLC(N)
            BQTCR = (QR(ROW, COL) / QT(ROW, COL)) * BLC(N)
        ELSE
            SQTCF = 0
            SQTCB = 0
            SQTCL = 0
            SQTCR = 0
            BQTCF = 0
            BQTCB = 0
            BQTCL = 0
            BQTCR = 0
        END IF

        SDLF(N) = SQTCF * TSS / ((1 - PS) * GRID_SIZE)
        SDLB(N) = SQTCB * TSS / ((1 - PS) * GRID_SIZE)
        SDLL(N) = SQTCL * TSS / ((1 - PS) * GRID_SIZE)
        SDLR(N) = SQTCR * TSS / ((1 - PS) * GRID_SIZE)
        BDLF(N) = BQTCF * TSS / ((1 - PS) * GRID_SIZE)
        BDLB(N) = BQTCB * TSS / ((1 - PS) * GRID_SIZE)
        BDLL(N) = BQTCL * TSS / ((1 - PS) * GRID_SIZE)
        BDLR(N) = BQTCR * TSS / ((1 - PS) * GRID_SIZE)

        IF (N > NMUD) THEN
            DLF_TOT = DLF_TOT + SDLF(N)
            DLB_TOT = DLB_TOT + SDLB(N)
            DLL_TOT = DLL_TOT + SDLL(N)
            DLR_TOT = DLR_TOT + SDLR(N)
        END IF
        DLF_TOT = DLF_TOT + BDLF(N)
        DLB_TOT = DLB_TOT + BDLB(N)
```

```
        DLL_TOT = DLL_TOT + BDLL(N)
        DLR_TOT = DLR_TOT + BDLR(N)

    END DO

    !Limit transported sediment depths based on water depth
    SDL_TOT = 0
    BDL_TOT = 0
    DO N = 1, NF + 1

        !Sediment moved to row + 1 limited to depth in that cell
        IF (ROW < ROWS) THEN
            IF (DLF_TOT > H(ROW + 1, COL)) THEN
                IF (H(ROW + 1, COL) > 1E-3) THEN
                    BDLF(N) = BDLF(N) * H(ROW + 1 , COL) / DLF_TOT
                    IF (N > NMUD) THEN
                        SDLF(N) = SDLF(N) * H(ROW + 1 , COL) / &
                            DLF_TOT
                    END IF
                ELSE
                    BDLF(N) = 0
                    IF (N > NMUD) THEN
                        SDLF(N) = 0
                    END IF
                END IF
            END IF
        END IF

        !Sediment moved to row - 1 limited to depth in that cell
        IF (ROW > 1) THEN
            IF (DLB_TOT > H(ROW - 1, COL)) THEN
                IF (H(ROW - 1, COL) > 1E-3) THEN
                    BDLB(N) = BDLB(N) * H(ROW - 1 , COL) / DLB_TOT
                    IF (N > NMUD) THEN
                        SDLB(N) = SDLB(N) * H(ROW - 1 , COL) / &
                            DLB_TOT
                    END IF
                ELSE
                    BDLB(N) = 0
                    IF (N > NMUD) THEN
                        SDLB(N) = 0
                    END IF
                END IF
            END IF
        END IF

        !Sediment moved to col - 1 limited to depth in that cell
        IF (COL > 1) THEN
            IF (DLL_TOT > H(ROW, COL - 1)) THEN
                IF (H(ROW, COL - 1) > 1E-3) THEN
                    BDLL(N) = BDLL(N) * H(ROW, COL - 1) / DLL_TOT
                    IF (N > NMUD) THEN
                        SDLL(N) = SDLL(N) * H(ROW, COL - 1) / &
                            DLL_TOT
                    END IF
                ELSE
                    BDLL(N) = 0
                    IF (N > NMUD) THEN
                        SDLL(N) = 0
                    END IF
                END IF
            END IF
        END IF

        !Sediment moved to col + 1 limited to depth in that cell
```

```
        IF (COL < COLS) THEN
            IF (DLR_TOT > H(ROW, COL + 1)) THEN
                IF (H(ROW, COL + 1) > 1E-3) THEN
                    BDLR(N) = BDLR(N) * H(ROW, COL + 1) / DLR_TOT
                    IF (N > NMUD) THEN
                        SDLR(N) = SDLR(N) * H(ROW, COL + 1) / &
                            DLR_TOT
                    END IF
                ELSE
                    BDLR(N) = 0
                    IF (N > NMUD) THEN
                        SDLR(N) = 0
                    END IF
                END IF
            END IF
        END IF

        SDLT(N) = SDLF(N) + SDLB(N) + SDLL(N) + SDLR(N)
        BDLT(N) = BDLF(N) + BDLB(N) + BDLL(N) + BDLR(N)
        SDL_TOT = SDL_TOT + SDLT(N)
        BDL_TOT = BDL_TOT + BDLT(N)

    END DO

    DL_TOT = SDL_TOT + BDL_TOT

    !Total erosion limited to total sediment thickness in cell
    !or active layer thickness
    IF (GRID(ROW, COL) > FGRID(ROW, COL)) THEN
        MAXE = GRID(ROW, COL) - FGRID(ROW, COL)
    ELSE
        MAXE = 0
    END IF

    IF (MAXE > ALYR_TH) THEN
        MAXE = ALYR_TH
    END IF

    IF (MAXE > 1E-4) THEN
        IF (DL_TOT > MAXE) THEN
            SDLF = SDLF * MAXE / DL_TOT
            SDLB = SDLB * MAXE / DL_TOT
            SDLL = SDLL * MAXE / DL_TOT
            SDLR = SDLR * MAXE / DL_TOT
            SDLT = SDLT * MAXE / DL_TOT
            BDLF = BDLF * MAXE / DL_TOT
            BDLB = BDLB * MAXE / DL_TOT
            BDLL = BDLL * MAXE / DL_TOT
            BDLR = BDLR * MAXE / DL_TOT
            BDLT = BDLT * MAXE / DL_TOT
        END IF
    ELSE
        SDLF = 0
        SDLB = 0
        SDLL = 0
        SDLR = 0
        SDLT = 0
        BDLF = 0
        BDLB = 0
        BDLL = 0
        BDLR = 0
        BDLT = 0
    END IF

    !Limit transported sediment of each fraction to current depth
```

```
!in active layer
DO N = 1, NF + 1
    IF (ALYR(ROW, COL, N) > 1E-4) THEN
        IF (SDLT(N) + BDLT(N) < ALYR(ROW, COL, N)) THEN
            ALYR1(ROW, COL, N) = ALYR1(ROW, COL, N) + &
                ALYR(ROW, COL, N) - (SDLT(N) + BDLT(N))
        ELSE
            SDLF(N) = SDLF(N) * ALYR(ROW, COL, N) / &
                (SDLT(N) + BDLT(N))
            SDLB(N) = SDLB(N) * ALYR(ROW, COL, N) / &
                (SDLT(N) + BDLT(N))
            SDLL(N) = SDLL(N) * ALYR(ROW, COL, N) / &
                (SDLT(N) + BDLT(N))
            SDLR(N) = SDLR(N) * ALYR(ROW, COL, N) / &
                (SDLT(N) + BDLT(N))
            BDLF(N) = BDLF(N) * ALYR(ROW, COL, N) / &
                (SDLT(N) + BDLT(N))
            BDLB(N) = BDLB(N) * ALYR(ROW, COL, N) / &
                (SDLT(N) + BDLT(N))
            BDLL(N) = BDLL(N) * ALYR(ROW, COL, N) / &
                (SDLT(N) + BDLT(N))
            BDLR(N) = BDLR(N) * ALYR(ROW, COL, N) / &
                (SDLT(N) + BDLT(N))
        END IF
    ELSE
        ALYR1(ROW, COL, N) = ALYR1(ROW, COL, N) + &
            ALYR(ROW, COL, N)
        SDLF(N) = 0
        SDLB(N) = 0
        SDLL(N) = 0
        SDLR(N) = 0
        BDLF(N) = 0
        BDLB(N) = 0
        BDLL(N) = 0
        BDLR(N) = 0
    END IF
END DO

!Add clay & silt suspended sediment transport added to
!erosion totals
DO N = 1, NMUD
    MF_SED_OUT(N) = MF_SED_OUT(N) + ((SDLB(N) + SDLF(N) + &
        SDLL(N) + SDLR(N)) * CELL_AREA)
    ETOT(ROW, COL) = ETOT(ROW, COL) + SDLB(N) + SDLF(N) + &
        SDLL(N) + SDLR(N)
END DO

!Move bed load transported sediment and suspended sand
!transported sediment to active layer in neighbouring cells
DO N = 1, NF + 1
    IF (ROW < ROWS) THEN
        ALYR1(ROW + 1, COL, N) = ALYR1(ROW + 1, COL, N) + &
            BDLF(N)
        IF (N > NMUD) THEN
            ALYR1(ROW + 1, COL, N) = &
                ALYR1(ROW + 1, COL, N) + SDLF(N)
        END IF
    ELSE
        MF_SED_OUT(N) = MF_SED_OUT(N) + (BDLF(N) * CELL_AREA)
        IF (N > NMUD) THEN
            MF_SED_OUT(N) = MF_SED_OUT(N) + &
                (SDLF(N) * CELL_AREA)
        END IF
    END IF
    IF (ROW > 1) THEN
```

```
                    ALYR1(ROW - 1, COL, N) = ALYR1(ROW - 1, COL, N) + &
                        BDLB(N)
                    IF (N > NMUD) THEN
                        ALYR1(ROW - 1, COL, N) = &
                            ALYR1(ROW - 1, COL, N) + SDLB(N)
                    END IF
                ELSE
                    ALYR1(ROW, COL, N) = ALYR1(ROW, COL, N) + BDLB(N)
                    IF (N > NMUD) THEN
                        ALYR1(ROW, COL, N) = ALYR1(ROW, COL, N) + SDLB(N)
                    END IF
                END IF
                IF (COL > 1) THEN
                    ALYR1(ROW, COL - 1, N) = ALYR1(ROW, COL - 1, N) + &
                        BDLL(N)
                    IF (N > NMUD) THEN
                        ALYR1(ROW, COL - 1, N) = &
                            ALYR1(ROW, COL - 1, N) + SDLL(N)
                    END IF
                ELSE
                    ALYR1(ROW, COL, N) = ALYR1(ROW, COL, N) + BDLL(N)
                    IF (N > NMUD) THEN
                        ALYR1(ROW, COL, N) = ALYR1(ROW, COL, N) + SDLL(N)
                    END IF
                END IF
                IF (COL < COLS) THEN
                    ALYR1(ROW, COL + 1, N) = ALYR1(ROW, COL + 1, N) + &
                        BDLR(N)
                    IF (N > NMUD) THEN
                        ALYR1(ROW, COL + 1, N) = &
                            ALYR1(ROW, COL + 1, N) + SDLR(N)
                    END IF
                ELSE
                    ALYR1(ROW, COL, N) = ALYR1(ROW, COL, N) + BDLR(N)
                    IF (N > NMUD) THEN
                        ALYR1(ROW, COL, N) = ALYR1(ROW, COL, N) + SDLR(N)
                    END IF
                END IF
            END DO !(FRACTIONS)

        ELSE !(if cv < min_v or h < min_h)

            !If water depth or velocity below minimum, only
            !calculate critical
            !bed shear stress, for use in lateral transport calcs
            IF (SPR(1) < 0.3) THEN
                TBCR(ROW, COL) = TB_CRIT(FI, DS, FMIN, FMAX, SPR, &
                    SPR(1), NF, BM(ROW, COL))
            ELSE
                IF (SPR(1) < 0.999) THEN
                    FR = (SPR(1) - 0.3) / 0.7
                    TBCR(ROW, COL) = ((1.0 - FR) * TB_CRIT(FI, DS, &
                        FMIN, FMAX, SPR, 0.3, NF, BM(ROW, COL))) + (FR * TE)
                ELSE
                    TBCR(ROW, COL) = TE
                END IF
            END IF

            ALYR1(ROW, COL,:) = ALYR1(ROW, COL,:) + ALYR(ROW, COL,:)
            TMX(ROW, COL,:) = 0

        END IF

    END DO !(COLS)
END DO    !(ROWS)
```

```
ALYR = 0
!Lateral sediment transport
DO ROW = 1, ROWS
    DO COL = 1, COLS

        ALYR_TOT = 0
        DO N = 1, NF + 1
            ALYR_TOT = ALYR_TOT + ALYR1(ROW, COL, N)
        END DO

        IF (ALYR_TOT > 1E-4) THEN
            !Calculate updated fraction proportions
            DO N = 1, NF + 1
                SPR(N) = ALYR1(ROW, COL, N) / ALYR_TOT
            END DO


            !Forward lateral transport (to row + 1)
            IF (ROW < ROWS) THEN

                !Calculate bed level difference (limited to water depth
                !in adjacent cell)
                BDL = GRID(ROW, COL) - GRID(ROW + 1, COL)
                IF (BDL > H(ROW + 1, COL)) THEN
                    BDL = H(ROW + 1, COL)
                END IF

                IF (BDL > 0.01 .AND. CV(ROW + 1, COL) > MIN_V) THEN

                    IF (SPR(1) < 0.3) THEN
                        BLC = 0
                        SLC = 0
                        DO M = 1, NWS2

                            HS = WAVEH(ROW + 1, COL, M)
                            IF (HS < MIN_HS) HS = MIN_HS
                            TP = WAVET(ROW + 1, COL, M)
                            IF (TP < MIN_TP) TP = MIN_TP

                            !Calculate transport rates using velocity and
                            !wave properties in the adjacent cell
                            CALL VAN_RIJN_MF(FI, WS, FMIN, FMAX, SPR(:), &
                                NF, DS, H(ROW + 1, COL), HS, TP, &
                                WC_DIR(ROW + 1, COL, M), &
                                CV(ROW + 1, COL), SLCM, BLCM, &
                                BM(ROW, COL), DUM1)

                            BLC = BLC + (BLCM * P(M))
                            SLC = SLC + (SLCM * P(M))

                        END DO

                        SLC(1) = 0
                        BLC(1) = 0

                        IF (SPR(1) > 0.2) THEN

                            !Calculate mud erosion rates using velocity
                            !and wave properties in the adjacent cell
                            DUM1 = 0.0
                            EM1 = 0
                            DO M = 1, NWS2
                                HS = WAVEH(ROW + 1, COL, M)
```

```
                    IF (HS < MIN_HS) HS = MIN_HS
                    TP = WAVET(ROW + 1, COL, M)
                    IF (TP < MIN_TP) TP = MIN_TP

                    CALL MUD_ER(CV(ROW + 1, COL), HS, TP, &
                        WC_DIR(ROW + 1, COL, M), &
                        H(ROW + 1, COL), SPR(1), DMUD, &
                        TBCR(ROW, COL), 0.0, 0.0, &
                        DUM1, TSS, EMM, DUM2, DUM3, .TRUE.)

                    EM1 = EM1 + (EMM * P(M))
                END DO

                P_MUD = 0
                DO N = 1, NMUD
                    P_MUD = P_MUD + SPR(N)
                END DO

                !Interpolation between mud type transport and
                !multifraction mode for pclay 20 - 30%
                EM1 = EM1 * 10.0 * (SPR(1) - 0.2)
                SLC = SLC * 10.0 * (0.3 - SPR(1))
                BLC = BLC * 10.0 * (0.3 - SPR(1))

                !Clay fraction assumed to be brought into
                !suspension due to the transport of other
                !fractions
                DO N = 2, NF + 1
                    SLC(1) = SLC(1) + (SPR(1) * &
                    (SLC(N) + BLC(N)))
                END DO
                !Clay and silt fractions suspended according
                !to mud erosion equation
                DO N = 1, NMUD
                    SLC(N) = SLC(N) + ((SPR(N) / P_MUD) * &
                        EM1 * (1 - PS) * GRID_SIZE / TSS)
                END DO
            ELSE
                !Clay fraction assumed to be brought into
                !suspension due to the transport of other
                !fractions
                DO N = 2, NF + 1
                    SLC(1) = SLC(1) + (SPR(1) * &
                        (SLC(N) + BLC(N)))
                END DO
            END IF
        ELSE

            !Calculate mud erosion rates using velocity and
            !wave properties in the adjacent cell
            DUM1 = 0.0

            DO M = 1, NWS2
                HS = WAVEH(ROW + 1, COL, M)
                IF (HS < MIN_HS) HS = MIN_HS
                TP = WAVET(ROW + 1, COL, M)
                IF (TP < MIN_TP) TP = MIN_TP

                CALL MUD_ER(CV(ROW + 1, COL), HS, TP, &
                    WC_DIR(ROW + 1, COL, M), &
                    H(ROW + 1, COL), SPR(1), DMUD, &
                    TBCR(ROW, COL), 0.0, 0.0, &
                    DUM1, TSS, EMM, DUM2, DUM3, .TRUE.)

                EM1 = EM1 + (EMM * P(M))
```

```
                END DO

                P_MUD = 0
                DO N = 1, NMUD
                    P_MUD = P_MUD + SPR(N)
                END DO

                !Clay and silt fractions suspended according to
                !mud erosion equation
                DO N = 1, NMUD
                    BLC(N) = 0
                    SLC(N) = (SPR(N) / P_MUD) * EM1 * (1 - PS) * &
                        GRID_SIZE / TSS
                END DO
                IF (NMUD < NF + 1) THEN
                    DO N = NMUD + 1, NF + 1
                        BLC(N) = 0
                        SLC(N) = 0
                    END DO
                END IF

            END IF

            !Calculate transported sediment depth for each
            !fraction
            DO N = 1, NF + 1
                DLTF(N) = LTFR * (BLC(N) + SLC(N)) * TSS * BDL / &
                    ((1 - PS) * GRID_SIZE * GRID_SIZE)
            END DO
        ELSE
            DLTF = 0
        END IF

    ELSE
        DLTF = 0
    END IF


    !Backward lateral transport (to row - 1)
    IF (ROW > 1) THEN

        !Calculate bed level difference
        !(limited to water depth in adjacent cell)
        BDL = GRID(ROW, COL) - GRID(ROW - 1, COL)
        IF (BDL > H(ROW - 1, COL)) THEN
            BDL = H(ROW - 1, COL)
        END IF

        IF (BDL > 0.01 .AND. CV(ROW - 1, COL) > MIN_V + 1E-6) THEN

            IF (SPR(1) < 0.3) THEN

                BLC = 0
                SLC = 0
                DO M = 1, NWS2
                    HS = WAVEH(ROW - 1, COL, M)
                    IF (HS < MIN_HS) HS = MIN_HS
                    TP = WAVET(ROW - 1, COL, M)
                    IF (TP < MIN_TP) TP = MIN_TP

                    !Calculate transport rates using velocity and
                    !wave properties in the adjacent cell
                    CALL VAN_RIJN_MF(FI, WS, FMIN, FMAX, SPR(:), &
                        NF, DS, H(ROW - 1, COL), HS, TP, &
```

```
                WC_DIR(ROW - 1, COL, M), &
                CV(ROW - 1, COL), SLCM, BLCM, &
                BM(ROW, COL), DUM1)

            BLC = BLC + (BLCM * P(M))
            SLC = SLC + (SLCM * P(M))

        END DO

        SLC(1) = 0
        BLC(1) = 0

        IF (SPR(1) > 0.2) THEN

            !Calculate mud erosion rates using velocity
            !and wave properties in the adjacent cell
            DUM1 = 0.0
            EM1 = 0
            DO M = 1, NWS2
                CALL MUD_ER(CV(ROW - 1, COL), HS, TP, &
                    WC_DIR(ROW - 1, COL, M), &
                    H(ROW - 1, COL), SPR(1), DMUD, &
                    TBCR(ROW, COL), 0.0, 0.0, &
                    DUM1, TSS, EMM, DUM2, DUM3, .TRUE.)

                EM1 = EM1 + (EMM * P(M))
            END DO


            P_MUD = 0
            DO N = 1, NMUD
                P_MUD = P_MUD + SPR(N)
            END DO

            !Interpolation between mud type transport and
            !multifraction mode for pclay 20 - 30%
            EM1 = EM1 * 10.0 * (SPR(1) - 0.2)
            SLC = SLC * 10.0 * (0.3 - SPR(1))
            BLC = BLC * 10.0 * (0.3 - SPR(1))

            !Clay fraction assumed to be brought into
            !suspension due to the transport of other
            !fractions
            DO N = 2, NF + 1
                SLC(1) = SLC(1) + (SPR(1) * &
                    (SLC(N) + BLC(N)))
            END DO
            !Clay and silt fractions suspended according
            !to mud erosion equation
            DO N = 1, NMUD
                SLC(N) = SLC(N) + ((SPR(N) / P_MUD) * &
                    EM1 * (1 - PS) * GRID_SIZE / TSS)
            END DO
        ELSE
            !Clay fraction assumed to be brought into
            !suspension due to the transport of other
            !fractions
            DO N = 2, NF + 1
                SLC(1) = SLC(1) + (SPR(1) * &
                    (SLC(N) + BLC(N)))
            END DO
        END IF
    ELSE
        !Calculate mud erosion rates using velocity and
        !wave properties in the adjacent cell
```

```
                DUM1 = 0.0
                EM1 = 0
                DO M = 1, NWS2
                    CALL MUD_ER(CV(ROW - 1, COL), HS, TP, &
                        WC_DIR(ROW - 1, COL, M), &
                        H(ROW - 1, COL), SPR(1), DMUD, &
                        TBCR(ROW, COL), 0.0, 0.0, &
                        DUM1, TSS, EMM, DUM2, DUM3, .TRUE.)
                    EM1 = EM1 + (EMM * P(M))
                END DO

                P_MUD = 0
                DO N = 1, NMUD
                    P_MUD = P_MUD + SPR(N)
                END DO

                !Clay and silt fractions suspended according to
                !mud erosion equation
                DO N = 1, NMUD
                    BLC(N) = 0
                    SLC(N) = (SPR(N) / P_MUD) * EM1 * (1 - PS) * &
                        GRID_SIZE / TSS
                END DO
                IF (NMUD < NF + 1) THEN
                    DO N = NMUD + 1, NF + 1
                        BLC(N) = 0
                        SLC(N) = 0
                    END DO
                END IF

            END IF

            !Calculate transported sediment depth for each fraction
            DO N = 1, NF + 1
                DLTB(N) = LTFR * (BLC(N) + SLC(N)) * TSS * BDL / &
                    ((1 - PS) * GRID_SIZE * GRID_SIZE)
            END DO
        ELSE
            DLTB = 0
        END IF

    ELSE
        DLTB = 0
    END IF

    !Leftward lateral transport (to col - 1)
    IF (COL > 1) THEN

        !Calculate bed level difference
        !(limited to water depth in adjacent cell)
        BDL = GRID(ROW, COL) - GRID(ROW, COL - 1)
        IF (BDL > H(ROW, COL - 1)) THEN
            BDL = H(ROW, COL - 1)
        END IF

        IF (BDL > 0.01 .AND. CV(ROW, COL - 1) > MIN_V + 1E-6) THEN

            IF (SPR(1) < 0.3) THEN
                BLC = 0
                SLC = 0
                DO M = 1, NWS2
                    HS = WAVEH(ROW, COL - 1, M)
                    IF (HS < MIN_HS) HS = MIN_HS
                    TP = WAVET(ROW, COL - 1, M)
                    IF (TP < MIN_TP) TP = MIN_TP
```

```
        !Calculate transport rates using velocity and
        !wave properties in the adjacent cell
        CALL VAN_RIJN_MF(FI, WS, FMIN, FMAX, SPR(:), &
            NF, DS, H(ROW, COL - 1), HS, TP, &
            WC_DIR(ROW, COL - 1, M), &
            CV(ROW, COL - 1), SLCM, BLCM, &
            BM(ROW, COL), DUM1)

        BLC = BLC + (BLCM * P(M))
        SLC = SLC + (SLCM * P(M))
    END DO

    SLC(1) = 0
    BLC(1) = 0

    IF (SPR(1) > 0.2) THEN

        !Calculate mud erosion rates using velocity
        !and wave properties in the adjacent cell
        DUM1 = 0
        EM1 = 0
        DO M = 1, NWS2
            CALL MUD_ER(CV(ROW, COL - 1), HS, TP, &
                WC_DIR(ROW, COL - 1, M), &
                H(ROW, COL - 1), SPR(1), DMUD, &
                TBCR(ROW, COL), 0.0, 0.0, &
                DUM1, TSS, EMM, DUM2, DUM3, .TRUE.)

            EM1 = EM1 + (EMM * P(M))
        END DO

        P_MUD = 0
        DO N = 1, NMUD
            P_MUD = P_MUD + SPR(N)
        END DO

        !Interpolation between mud type transport and
        !multifraction mode for pclay 20 - 30%
        EM1 = EM1 * 10.0 * (SPR(1) - 0.2)
        SLC = SLC * 10.0 * (0.3 - SPR(1))
        BLC = BLC * 10.0 * (0.3 - SPR(1))

        !Clay fraction assumed to be brought into
        !suspension due to the transport of other
        !fractions
        DO N = 2, NF + 1
            SLC(1) = SLC(1) + (SPR(1) * &
                (SLC(N) + BLC(N)))
        END DO
        !Clay and silt fractions suspended according
        !to mud erosion equation
        DO N = 1, NMUD
            SLC(N) = SLC(N) + ((SPR(N) / P_MUD) * &
                EM1 * (1 - PS) * GRID_SIZE / TSS)
        END DO
    ELSE
        !Clay fraction assumed to be brought into
        !suspension due to the transport of other
        !fractions
        DO N = 2, NF + 1
            SLC(1) = SLC(1) + (SPR(1) * &
                (SLC(N) + BLC(N)))
        END DO
    END IF
```

```
            ELSE

                !Calculate mud erosion rates using velocity and
                !wave properties in the adjacent cell
                DUM1 = 0
                EM1 = 0
                DO M = 1, NWS2
                    CALL MUD_ER(CV(ROW, COL - 1), HS, TP, &
                        WC_DIR(ROW, COL - 1, M), &
                        H(ROW, COL - 1), SPR(1), &
                        DMUD, TBCR(ROW, COL), 0.0, 0.0, &
                        DUM1, TSS, EMM, DUM2, DUM3, .TRUE.)

                    EM1 = EM1 + (EMM * P(M))
                END DO


                P_MUD = 0
                DO N = 1, NMUD
                    P_MUD = P_MUD + SPR(N)
                END DO

                !Clay and silt fractions suspended according to
                !mud erosion equation
                DO N = 1, NMUD
                    BLC(N) = 0
                    SLC(N) = (SPR(N) / P_MUD) * EM1 * &
                        (1 - PS) * GRID_SIZE / TSS
                END DO
                IF (NMUD < NF + 1) THEN
                    DO N = NMUD + 1, NF + 1
                        BLC(N) = 0
                        SLC(N) = 0
                    END DO
                END IF

            END IF

            !Calculate transported sediment depth for each
            !fraction
            DO N = 1, NF + 1
                DLTL(N) = LTFR * (BLC(N) + SLC(N)) * &
                    TSS * BDL / ((1 - PS) * GRID_SIZE * GRID_SIZE)
            END DO
        ELSE
            DLTL = 0
        END IF

    ELSE
        DLTL = 0
    END IF

    !Rightward lateral transport (to col + 1)
    IF (COL < COLS) THEN

        !Calculate bed level difference
        !(limited to water depth in adjacent cell)
        BDL = GRID(ROW, COL) - GRID(ROW, COL + 1)
        IF (BDL > H(ROW, COL + 1)) THEN
            BDL = H(ROW, COL + 1)
        END IF

        IF (BDL > 0.01 .AND. CV(ROW, COL + 1) > MIN_V + 1E-6) THEN

            IF (SPR(1) < 0.3) THEN
```

```
BLC = 0
SLC = 0
DO M = 1, NWS2
    HS = WAVEH(ROW, COL + 1, M)
    IF (HS < MIN_HS) HS = MIN_HS
    TP = WAVET(ROW, COL + 1, M)
    IF (TP < MIN_TP) TP = MIN_TP

    !Calculate transport rates using velocity and
    !wave properties in the adjacent cell
    CALL VAN_RIJN_MF(FI, WS, FMIN, FMAX, SPR(:), &
        NF, DS, H(ROW, COL + 1), HS, TP, &
        WC_DIR(ROW, COL + 1, M), &
        CV(ROW, COL + 1), SLCM, BLCM, &
        BM(ROW, COL), DUM1)

    BLC = BLC + (BLCM * P(M))
    SLC = SLC + (SLCM * P(M))
END DO

SLC(1) = 0
BLC(1) = 0

IF (SPR(1) > 0.2) THEN

    !Calculate mud erosion rates using velocity
    !and wave properties in the adjacent cell
    DUM1 = 0.0
    EM1 = 0
    DO M = 1, NWS2
        CALL MUD_ER(CV(ROW, COL + 1), HS, TP, &
            WC_DIR(ROW, COL + 1, M), &
            H(ROW, COL + 1), SPR(1), DMUD, &
            TBCR(ROW, COL), 0.0, 0.0, &
            DUM1, TSS, EMM, DUM2, DUM3, .TRUE.)

        EM1 = EM1 + (EMM * P(M))
    END DO

    P_MUD = 0
    DO N = 1, NMUD
        P_MUD = P_MUD + SPR(N)
    END DO

    !Interpolation between mud type transport and
    !multifraction mode for pclay 20 - 30%
    EM1 = EM1 * 10.0 * (SPR(1) - 0.2)
    SLC = SLC * 10.0 * (0.3 - SPR(1))
    BLC = BLC * 10.0 * (0.3 - SPR(1))

    !Clay fraction assumed to be brought into
    !suspension due to the transport of other
    !fractions
    DO N = 2, NF + 1
        SLC(1) = SLC(1) + (SPR(1) * &
            (SLC(N) + BLC(N)))
    END DO
    !Clay and silt fractions suspended according
    !to mud erosion equation
    DO N = 1, NMUD
        SLC(N) = SLC(N) + ((SPR(N) / P_MUD) * &
            EM1 * (1 - PS) * GRID_SIZE / TSS)
    END DO
ELSE
    !Clay fraction assumed to be brought into
```

```
                        !suspension due to the transport of other
                        !fractions
                        DO N = 2, NF + 1
                            SLC(1) = SLC(1) + (SPR(1) * &
                                (SLC(N) + BLC(N)))
                        END DO
                    END IF
                ELSE

                    !Calculate mud erosion rates using velocity and
                    !wave properties in the adjacent cell
                    DUM1 = 0.0
                    EM1 = 0
                    DO M = 1, NWS2
                        CALL MUD_ER(CV(ROW, COL + 1), HS, TP, &
                            WC_DIR(ROW, COL + 1, M), &
                            H(ROW, COL + 1), SPR(1), DMUD, &
                            TBCR(ROW, COL), 0.0, 0.0, &
                            DUM1, TSS, EMM, DUM2, DUM3, .TRUE.)

                        EM1 = EM1 + (EMM * P(M))
                    END DO

                    P_MUD = 0
                    DO N = 1, NMUD
                        P_MUD = P_MUD + SPR(N)
                    END DO

                    !Clay and silt fractions suspended according to
                    !mud erosion equation
                    DO N = 1, NMUD
                        BLC(N) = 0
                        SLC(N) = (SPR(N) / P_MUD) * EM1 * (1 - PS) * &
                            GRID_SIZE / TSS
                    END DO
                    IF (NMUD < NF + 1) THEN
                        DO N = NMUD + 1, NF + 1
                            BLC(N) = 0
                            SLC(N) = 0
                        END DO
                    END IF

                END IF

                !Calculate transported sediment depth for each
                !fraction
                DO N = 1, NF + 1
                    DLTR(N) = LTFR * (BLC(N) + SLC(N)) * TSS * BDL / &
                        ((1 - PS) * GRID_SIZE * GRID_SIZE)
                END DO
            ELSE
                DLTR = 0
            END IF

        ELSE
            DLTR = 0
        END IF
    ELSE
        !If active layer thickness reduced to 0.1mm set lateral
        !transport sediment depths to zero
        DLTF = 0
        DLTB = 0
        DLTL = 0
        DLTR = 0
    END IF
```

```
        DLTT = DLTF + DLTB + DLTL + DLTR
        DLTTT = 0
        DO N = 1, NF + 1
            DLTTT = DLTTT + DLTT(N)
        END DO

        !Calculate maximum erosion, limited by minimum bed level
        IF (GRID(ROW, COL) > FGRID(ROW, COL)) THEN
            MAXE = GRID(ROW, COL) - FGRID(ROW, COL)
        ELSE
            MAXE = 0
        END IF

        !Ensure total transport out of cell not greater than maximum
        IF (DLTTT - MAXE > 0) THEN
            DLTF = DLTF * MAXE / DLTTT
            DLTB = DLTB * MAXE / DLTTT
            DLTL = DLTL * MAXE / DLTTT
            DLTR = DLTR * MAXE / DLTTT
            DLTT = DLTT * MAXE / DLTTT
        END IF

        DO N = 1, NF + 1

            !Ensure transported depth not greater than current depth in
            !active layer for each fraction
            IF (DLTT(N) - ALYR1(ROW, COL, N) > 0) THEN
                DLTF(N) = DLTF(N) * ALYR1(ROW, COL, N) / DLTT(N)
                DLTB(N) = DLTB(N) * ALYR1(ROW, COL, N) / DLTT(N)
                DLTL(N) = DLTL(N) * ALYR1(ROW, COL, N) / DLTT(N)
                DLTR(N) = DLTR(N) * ALYR1(ROW, COL, N) / DLTT(N)
            ELSE
                !Depth of each fraction remaining in active layer (ALYR1)
                !(if less than maximum) added to ALYR
                ALYR(ROW, COL, N) = ALYR(ROW, COL, N) + &
                    ALYR1(ROW, COL, N) - DLTT(N)
            END IF

            !Forward lateral transport
            IF (ROW < ROWS) THEN
                ALYR(ROW + 1, COL, N) = ALYR(ROW + 1, COL, N) + DLTF(N)
            END IF

            !Backward lateral transport
            IF (ROW > 1) THEN
                ALYR(ROW - 1, COL, N) = ALYR(ROW - 1, COL, N) + DLTB(N)
            END IF

            !Leftward lateral transport
            IF (COL > 1) THEN
                ALYR(ROW, COL - 1, N) = ALYR(ROW, COL - 1, N) + DLTL(N)
            END IF

            !Rightward lateral transport
            IF (COL < COLS) THEN
                ALYR(ROW, COL + 1, N) = ALYR(ROW, COL + 1, N) + DLTR(N)
            END IF

        END DO

    END DO
END DO

FDMAX = 0.2
```

```
!Suspended sediment deposition
DO ROW = 1, ROWS
    DO COL = 1, COLS

        BMS = BM_MAX * BM(ROW, COL)
        NST = 250 * (BMS ** 0.3032)
        HST = 0.069 * (BMS ** 0.1876)
        DST = 0.0006 * (BMS ** 0.3)

        IF (H(ROW, COL) > H_MIN) THEN
            !Deposition
            TDI = TD
            IF (TDI > TE) TDI = TE
            IF (BM(ROW, COL) > 1E-3) THEN
                !Marsh related sediment trapping
                SED_TRP = 0.224 * ((CV(ROW, COL) * DST / KV) ** 0.718) * &
                    ((8E-6 / DST) ** 2.08)
                FRSM = CV(ROW, COL) * SED_TRP * NST * DST * HST * TSS
                IF (FRSM > 1.0) FRSM = 1.0
                DEP1 = FRSM * MEAN_H(ROW, COL)
            ELSE
                DEP1 = 0
            END IF

            DEP2 = 0
            DO M = 1, NWS2
                IF (TMX(ROW, COL, M) < TD) THEN
                    !Whitehouse et. al. 'Dynamics of Estuarine Muds',
                    !eqn 8.2
                    FRD = (1 - (TMX(ROW, COL, M) / TD)) * WS(1) * TSS / &
                        MEAN_H(ROW, COL)
                    IF (FRD > 1.0) THEN
                        FRD = 1.0
                    END IF
                    DEPM = FRD *  MEAN_H(ROW, COL)
                ELSE
                    DEPM = 0
                END IF
                DEP2 = DEP2 + (DEPM  * P(M))
            END DO
            DTOT(ROW, COL) = DTOT(ROW, COL) + DEP1 + DEP2
            MF_SED_IN(1) = MF_SED_IN(1) + ((DEP1 + DEP2) * SC(ROW, COL) &
                * CELL_AREA)
            ALYR(ROW, COL, 1) = ALYR(ROW, COL, 1) + ((DEP1 + DEP2) * &
                SC(ROW, COL))

        END IF

    END DO
END DO

!Marsh related deposition
!2 Organogenic sediment production
!Since values are very small, this is only added approximately
!once per month
IF (MT > 2628000) THEN
    DO ROW = 1, ROWS
        DO COL = 1, COLS
            IF (BM(ROW, COL) > 1E-3) THEN
                !Sediment generated is assumed to be of finest available
                !fraction size.  KB is max
                !rate of production in m/yr
                !(typically in range 0.001-0.01)
                ALYR(ROW, COL, 1) = ALYR(ROW, COL, 1) + &
```

```
                    KB * BM(ROW, COL) / 12.0
            END IF
        END DO
    END DO
    MT = MT - 2628000
END IF
MT = MT + TSS


TM = TM + 1

!Downstream boundary sand input
IF (NMUD < NF + 1) THEN
    QTOT = 0
    TMX_TOT = 0
    QTTOT = 0
    QTC = 0
    SEDINTOT = 0
    DO N = 2, NF + 1
        SEDINTOT = SEDINTOT + SED_IN(N, 2)
    END DO
    PI1(1) = 0
    DO N = 2, NF + 1
        PI1(N) = SED_IN(N, 2) / SEDINTOT
    END DO
    D50 = GETSIZE(FI, FMIN, FMAX, PI1, NF + 1, 0.5)
    D90 = GETSIZE(FI, FMIN, FMAX, PI1, NF + 1, 0.9)

    DO COL = 1, COLS
        IF (H(ROWS, COL) > H_MIN) THEN
            QTOT = QTOT + QY(ROWS, COL)
            TMXTOTR(COL) = 0
            DO M = 1, NWS2
                TMX_TOT = TMX_TOT + (TMX(ROWS, COL, M) * P(M))
                TMXTOTR(COL) = TMXTOTR(COL) + (TMX(ROWS, COL, M) * P(M))
                HS = WAVEH(ROWS, COL, M)
                IF (HS < MIN_HS) HS = MIN_HS
                TP = WAVET(ROWS, COL, M)
                IF (TP < MIN_TP) TP = MIN_TP
                QTM = SAND_TR(CV(ROWS, COL), HS, TP, H(ROWS, COL), &
                    DSTR, D50, D90, DS, 0.0, BM(ROWS, COL))
                QTTOT = QTTOT + (QTM * P(M))
                QTC(COL) = QTC(COL) + (QTM * P(M))
            END DO
        END IF
    END DO



    DO COL = 1, COLS
        IF (H(ROWS, COL) > H_MIN .AND. CVY(ROWS, COL) > MIN_V) THEN
            DO N = NMUD + 1, NF + 1

                IF (QTTOT > 1E-6) THEN
                    ALYR(ROWS, COL, N) = ALYR(ROWS, COL, N) + &
                        (SED_IN(N, 2) * TSS * QTC(COL) &
                        / (CELL_AREA * QTTOT))
                    MF_SED_IN(N) = MF_SED_IN(N) + (SED_IN(N, 2) * &
                        TSS * QTC(COL) / QTTOT)
                ELSE
                    ALYR(ROWS, COL, N) = ALYR(ROWS, COL, N) + &
                        (SED_IN(N, 2) * TSS * TMXTOTR(COL) / &
                        (CELL_AREA * TMX_TOT))
                    MF_SED_IN(N) = MF_SED_IN(N) + (SED_IN(N, 2) * &
                        TSS * TMXTOTR(COL) / TMX_TOT)
```

```
                END IF
            END DO

        END IF
    END DO
END IF


!Update active layer and sublayers
DO ROW = 1, ROWS
    DO COL = 1, COLS
IF (ROW == 202) THEN
AA = 1
END IF



        ALYR_TOT = 0
        DO N = 1, NF + 1
            ALYR_TOT = ALYR_TOT + ALYR(ROW, COL, N)
        END DO

        DL = ALYR_TOT - ALYR_TH


        SUB_LYR_NR = INT((GRID(ROW, COL) - ALYR_TH - BASE) / LYR_TH) + 1
        SUB_LYR_FR = GRID(ROW, COL) - ALYR_TH - BASE - &
            ((SUB_LYR_NR - 1.0) * LYR_TH)
        IF (SUB_LYR_FR < 0.0) SUB_LYR_FR = 0.0

        GRID(ROW, COL) = GRID(ROW, COL) + DL

        IF (DL > 0) THEN
            !Deposition
            IF (DL > LYR_TH - SUB_LYR_FR) THEN
                DO N = 1, NF + 1
                    SED(ROW, COL, SUB_LYR_NR, N) = &
                        ((SED(ROW, COL, SUB_LYR_NR, N) * SUB_LYR_FR) + &
                           (ALYR(ROW, COL, N) * (LYR_TH - SUB_LYR_FR) / &
                           ALYR_TOT)) / LYR_TH
                    NL = INT((DL - (LYR_TH - SUB_LYR_FR)) / LYR_TH) + 1
                    DO M = 1, NL
                        SED(ROW, COL, SUB_LYR_NR + M, N) = &
                            ALYR(ROW, COL, N) / ALYR_TOT
                    END DO
                    ALYR(ROW, COL, N) = ALYR(ROW, COL, N) * &
                        ALYR_TH / ALYR_TOT
                END DO
            ELSE
                DO N = 1, NF + 1
                    DF = ALYR(ROW, COL, N) * DL / ALYR_TOT
                    DSED = SED(ROW, COL, SUB_LYR_NR, N) * SUB_LYR_FR
                    SED(ROW, COL, SUB_LYR_NR, N) = (DF + DSED) / &
                        (SUB_LYR_FR + DL)
                    ALYR(ROW, COL, N) = ALYR(ROW, COL, N) * ALYR_TH / &
                        ALYR_TOT
                END DO
            END IF
        ELSE
            !Erosion
            IF (DL + SUB_LYR_FR < 0) THEN
                DO N = 1, NF + 1
                    ALYR(ROW, COL, N) = ALYR(ROW, COL, N) + &
                        (SED(ROW, COL, SUB_LYR_NR, N) * SUB_LYR_FR) - &
                        (SED(ROW, COL, SUB_LYR_NR - 1, N) * &
```

```
                            (DL + SUB_LYR_FR))
                END DO
            ELSE
                DO N = 1, NF + 1
                    ALYR(ROW, COL, N) = ALYR(ROW, COL, N) - &
                        (SED(ROW, COL, SUB_LYR_NR, N) * DL)
                END DO
            END IF

        END IF
    END DO
END DO


END SUBROUTINE

REAL FUNCTION GETSIZE(FI, FMIN, FMAX, PI, NFR, FR)
!Function to calculate representative grain size
!e.g. d50, d90 etc. based on the fraction sizes &
!proportions of each fraction

REAL, DIMENSION(:), INTENT(IN) :: PI
REAL, DIMENSION(:), INTENT(IN) :: FI
REAL, DIMENSION(:), INTENT(IN) :: FMIN
REAL, DIMENSION(:), INTENT(IN) :: FMAX

REAL, INTENT(IN) :: FR

INTEGER, INTENT(IN) :: NFR

REAL, DIMENSION(:), ALLOCATABLE :: PCUM

REAL FRA, FR1

INTEGER N

ALLOCATE (PCUM(NFR - 1))

!Reresentative particle size for sand/silt
!fractions (pclay excluded)
FRA = FR * (1 - PI(1))
PCUM(1) = PI(2)
GETSIZE = -1
N = 1
DO
    !Find smallest fraction for which fraction + all smaller fractions
    !make up greater proportion than specified (e.g. 50% for d50)
    IF ((PCUM(N) < FRA) .AND. (N < NFR - 1)) THEN
        N = N + 1
        PCUM(N) = PCUM(N - 1) + PI(N + 1)
    ELSE
        IF (N ==1) THEN
             FR1 = FRA / PCUM(N)
        ELSE
            FR1 = (FRA - PCUM(N - 1)) / (PCUM(N) - PCUM(N - 1))
        END IF
        GETSIZE = (FMIN(N + 1) ** (1 - FR1)) * (FMAX(N + 1) ** FR1)
        IF (GETSIZE < 1E-6) GETSIZE = 1E-6
    END IF

    IF (GETSIZE > -0.5) EXIT

END DO

END FUNCTION
```

```
END MODULE
```

Module MARSH

Subroutines	Purpose	Page nr.
CALC_MARSH	Calculate marsh biomass in each cell	396

```
MODULE MARSH

IMPLICIT NONE

CONTAINS

SUBROUTINE CALC_MARSH(ROWS, COLS, C, C_MIN, C_MAX, BM, GRID, WL1, WL2, &
    TSS, TIME, SUB_MIN, SUB_MAX, SM1, SM2)


REAL (SELECTED_REAL_KIND(15,307)), DIMENSION(:,:), INTENT(INOUT) :: GRID

REAL,  DIMENSION(:,:),            INTENT (IN) :: WL1
REAL,  DIMENSION(:,:),            INTENT (IN) :: WL2

REAL,  DIMENSION(:,:), INTENT(INOUT) :: C
REAL,   DIMENSION(:,:), INTENT(INOUT) :: BM

REAL, INTENT(IN) :: TIME
REAL, INTENT(IN) :: TSS
REAL, INTENT(IN) :: SUB_MIN
REAL, INTENT(IN) :: SUB_MAX
REAL, INTENT(IN) :: SM1
REAL, INTENT(IN) :: SM2
REAL, INTENT(IN) :: C_MIN
REAL, INTENT(IN) :: C_MAX

INTEGER, INTENT(IN) :: ROWS
INTEGER, INTENT(IN) :: COLS

REAL,    SAVE,    DIMENSION(:,:,:,:),    ALLOCATABLE :: PTS
REAL,    SAVE,    DIMENSION(:,:),        ALLOCATABLE :: SUB
REAL,    SAVE,    DIMENSION(:,:),        ALLOCATABLE :: SUB_TIME
REAL,    SAVE,    DIMENSION(:,:),        ALLOCATABLE :: BP


REAL RDAY, BP_MAX
REAL B_DIFF, E1, WLM

real aa

INTEGER IDAY
INTEGER SUB_PD
INTEGER ROW, COL

INTEGER :: NDAY = 0
INTEGER :: CDAY = 0
INTEGER :: LIST1 = 1
INTEGER :: LIST2 = 2

LOGICAL :: INIT = .TRUE.


BP_MAX = 1
```

```
SUB_PD = 30
E1 = 30

RDAY = TIME / 24.0
IDAY = INT(RDAY)

IF (INIT) THEN
    ALLOCATE (PTS(ROWS, COLS, 30, 2))
    ALLOCATE (SUB_TIME(ROWS, COLS))
    ALLOCATE (SUB(ROWS, COLS))
    ALLOCATE (BP(ROWS, COLS))
    PTS = 1
    SUB = 1
    SUB_TIME = 0
    INIT = .FALSE.
END IF

IF (IDAY > NDAY) THEN
    CDAY = CDAY + 1
    NDAY = NDAY + 1
    IF (CDAY > SUB_PD) THEN
        IF (LIST1 == 1) THEN
            LIST1 = 2
            LIST2 = 1
        ELSE
            LIST1 = 1
            LIST2 = 2
        END IF
        CDAY = 1
    END IF
    DO ROW = 1, ROWS
        DO COL = 1, COLS
            !SUB_TIME can slightly exceed 1, due to timestep not dividing
            !exactly into 24hrs
            IF (SUB_TIME(ROW, COL) > 1.0) THEN
                PTS(ROW, COL, CDAY, LIST1) = 1.0
            ELSE
                PTS(ROW, COL, CDAY, LIST1) = SUB_TIME(ROW, COL)
            END IF
            SUB(ROW, COL) = SUB(ROW, COL) + &
                (PTS(ROW, COL, CDAY, LIST1) / (SUB_PD))
            SUB(ROW, COL) = SUB(ROW, COL) - &
                (PTS(ROW, COL, CDAY, LIST2) / (SUB_PD))
            IF (SUB(ROW, COL) < 0) THEN
                SUB(ROW, COL) = 0
            ELSE
                IF (SUB(ROW, COL) > 1.0) THEN
                    SUB(ROW, COL) = 1.0
                END IF
            END IF

            B_DIFF = BP(ROW, COL) - BM(ROW, COL)
            BM(ROW, COL) = BM(ROW, COL) + (B_DIFF / E1)

        END DO
    END DO
    SUB_TIME = 0
END IF

DO ROW = 1, ROWS
    DO COL = 1, COLS
        WLM = (WL1(ROW, COL) + WL2(ROW, COL)) / 2.0
        IF (WLM > GRID(ROW, COL)) THEN
            SUB_TIME(ROW, COL)  = SUB_TIME(ROW, COL) + (TSS / 86400.0)
        END IF
```

```
    END DO
END DO

DO ROW = 1, ROWS
    DO COL = 1, COLS
        IF (SUB(ROW, COL) > SUB_MAX) THEN
            BP(ROW, COL) = 0
        ELSE
            IF (SUB(ROW, COL) > SM1) THEN
                BP(ROW, COL) = (SUB_MAX - SUB(ROW, COL)) / (SUB_MAX - SM1)
            ELSE
                IF (SUB(ROW, COL) > SM2) THEN
                    BP(ROW, COL) = 1.0
                ELSE
                    IF (SUB(ROW, COL) > SUB_MIN) THEN
                        BP(ROW, COL) = (SUB(ROW, COL) - SUB_MIN) / &
                            (SM2 - SUB_MIN)
                    ELSE
                        BP(ROW, COL) = 0
                    END IF
                END IF
            END IF
        END IF

        IF (BP(ROW, COL) > 1.0) BP(ROW, COL) = 1.0
        C(ROW, COL) = C_MAX - (BM(ROW, COL) * (C_MAX - C_MIN))

    END DO
END DO


END SUBROUTINE

END MODULE
```

MODULE ISIS

Subroutines	Purpose	Page nr.
CALC_TIDE	Calculate tidal water levels at model boundary	399
WRITE_ISIS	Write ISIS model (dat file)	400
RUN_ISIS	Run ISIS model	408
READ_ISIS	Read ISIS results file (csv output)	410
READ_LINE	Read one line from ISIS csv output file	412
READ_SS	Read ISIS steady-state results from ISIS zzs file	413
Functions		
CHECK_ISIS	Check ISIS diagnostic file to determine whether ISIS model run completed successfully	414

```
MODULE ISIS

IMPLICIT NONE

REAL, PARAMETER     :: PI = 3.14159265358979
REAL, PARAMETER :: LUNAR_MONTH    = 29.53059    !Synodic lunar month in days

CONTAINS

SUBROUTINE CALC_TIDE (TIDE, TDR_SP, TDR_NP, HTIDE, DBP)
!Subroutine to calculate tidal water levels at the downstream boundary

!Array comprising 2 colums: time & water level
REAL, DIMENSION (:,:), INTENT (INOUT)   :: TIDE
REAL, INTENT (IN)                                   :: TDR_SP    !Spring tidal
range
REAL, INTENT (IN)                                   :: TDR_NP    !Neap tidal range
REAL, INTENT (IN)                                   :: HTIDE     !Maximum water
level

INTEGER, INTENT (IN) :: DBP  !Number of water levels to calculate

REAL                         :: TIME = 0 !Time in hours

REAL TD_PERIOD             !Period of one tidal cycle in hours
REAL TD_RANGE       !Instantaneous tidal range
REAL NS_RANGE       !Differnece between spring and neap tidal ranges
REAL MID_TIDE       !Mean sea level
REAL LM_SOL         !Lunar month relative to sun in days

INTEGER       :: COUNT = 1 !Counter

LM_SOL = LUNAR_MONTH * (365.0 / 364.0)
TD_PERIOD = 12 / (1 - (1 / LM_SOL))

NS_RANGE = TDR_SP - TDR_NP
MID_TIDE = HTIDE - (TDR_SP / 2)

DO COUNT = 1, DBP
  !Spring neap period set to 29 tidal cycles rather than 1/2 lunar month,
  !as needs to be integer multple of tidal period
  TD_RANGE = TDR_SP - (0.5 * NS_RANGE) + (0.5 * NS_RANGE * &
    SIN (2 * PI * ((TIME / (TD_PERIOD * 29)) + 0.25)))
```

```
  TIDE (COUNT, 1) = TIME
  TIDE (COUNT, 2) = MID_TIDE + (0.5 * TD_RANGE * &
    SIN (2 * PI * TIME / TD_PERIOD))
  TIME = TIME + 0.2
END DO

END SUBROUTINE

SUBROUTINE WRITE_ISIS (CLG_BED, C, MAXC, NG_SIZE, ROW_CH, XSEC_CH, &
    XSEC_SP, ROWS, MAX_COLS, FILENAME, TIDE, DBP, INFLOW, ND_DSBDY, &
    INTERPOLATE, WALL_HT, MIN_WL)
!Subroutine to write an ISIS .dat file for a 1D model based on regular
!cross sections through the modelled estuary

REAL (SELECTED_REAL_KIND(15,307)), DIMENSION (:,:), INTENT (IN)    :: CLG_BED
REAL, DIMENSION (:,:), INTENT (IN)      :: TIDE
REAL, DIMENSION (:,:), INTENT (IN)      :: C
REAL, DIMENSION (:),   INTENT (IN)  :: ROW_CH
REAL, DIMENSION (:),   INTENT(OUT)  :: XSEC_CH

!INTEGER, DIMENSION (:), INTENT (IN)    :: COLS
INTEGER, DIMENSION(:), ALLOCATABLE :: COLS

REAL, OPTIONAL, INTENT (IN) :: MIN_WL

REAL, INTENT (IN)                           :: INFLOW   !Fluvial inflow
REAL, INTENT (IN)                       :: NG_SIZE  !Grid cell size
!Additional points added to ends of sections at height WALL_HT
REAL, INTENT (IN)                   :: WALL_HT
REAL, INTENT (IN)                   :: MAXC     !Max Chezy value

CHARACTER (200), INTENT (IN)            :: FILENAME !ISIS dat filename

INTEGER, INTENT (IN) :: ROWS     !Number of rows
INTEGER, INTENT (IN) :: MAX_COLS !Number of columns
INTEGER, INTENT (IN) :: DBP      !Number of points in downstream boundary
!Number of grid rows per ISIS cross section (i.e. cross section spacing)
INTEGER, INTENT (IN) :: XSEC_SP

!True if normal depth boundary to be applied instead of tidal boundary
LOGICAL, INTENT(IN)                 :: ND_DSBDY
!True if interpolates to be added between cross sections
LOGICAL, INTENT(IN)                 :: INTERPOLATE

REAL XCH            !Distance along cross section - left to right
REAL DTN        !Distance to next cross section
REAL DSBDY_WL
REAL MANNING

INTEGER ROW             !Current row
INTEGER COL             !Current column
INTEGER FCOL, LCOL  !First and last columns for current row
INTEGER XSECS     !Number of cross sections
INTEGER STRLEN
INTEGER NODE
INTEGER PU_XSEC     !Penultimate cross section row number
INTEGER COUNT

CHARACTER (200) TCS_FN     !Filename for tcs file (control file csv output)
CHARACTER (10) TMP_STR     !Temporary string holder
CHARACTER (36) TMP_STR1    !Temporary string holder

!COLS originally to give number of columns in each row.
!All rows have same number of
!columns in current version
```

```
ALLOCATE (COLS(ROWS))
COLS = MAX_COLS

!Set cross section spacing
IF ((ROWS - 1) - (INT((ROWS - 1) / XSEC_SP) * XSEC_SP) > 0) THEN
  XSECS = INT((ROWS - 1) / XSEC_SP) + 2
ELSE
  XSECS = INT((ROWS - 1) / XSEC_SP) + 1
END IF
PU_XSEC = ((XSECS - 2) * XSEC_SP) + 1

!Create filename for TCS file (control file for ISIS CSV output)
STRLEN = LEN_TRIM(FILENAME)
TCS_FN = FILENAME(1:STRLEN - 3) // "tcs"

!Open output file (ISIS DAT file)
OPEN (1, FILE = FILENAME)  !Open ISIS dat file for output
!Open tsc file
OPEN (2, FILE = TCS_FN)

!Write header information to tcs file
WRITE (2, 40) "[Data]"
WRITE (2, 40) "OutputOption=0"
WRITE (2, 40) "DataItem=2"
WRITE (2, 40) "ColumnPerNode=2"
WRITE (2, 40) "OutputTimeUnits=1"
WRITE (2, 40) "MaxOverOutputInterval=0"
WRITE (2, 40) "[Times]"
WRITE (2, 40) "FirstOutputTimeID=-1"
WRITE (2, 40) "LastOutputTimeID=-1"
WRITE (2, 40) "OutputInterval=3"
WRITE (2, 40) "[Nodes]"
WRITE (2, 40, ADVANCE = 'NO') "Count="
WRITE (TMP_STR, 10) XSECS
WRITE (2, 40) TRIM(ADJUSTL(TMP_STR))

!Write header information to ISIS file
WRITE (1, 40) ""
WRITE (1, 40) "#REVISION#1"
IF (INTERPOLATE) THEN
    WRITE (1, 10, ADVANCE = 'NO') (2 * XSECS) - 2
ELSE
    WRITE (1, 10, ADVANCE = 'NO') XSECS
END IF
WRITE (1, 40) "     0.750     0.900     0.100     0.001        12"
WRITE (1, 40) &
 "    10.000     0.010     0.010     0.700     0.100     0.700     0.000"
WRITE (1, 40) "RAD FILE"
WRITE (1, 40) ""
WRITE (1, 40) "END GENERAL"

!Add upstream flow boundary
WRITE (1, 40) "QTBDY"
WRITE (1, 40) "S1"
WRITE (1, 40, ADVANCE = 'NO') "         2     0.000     0.000     "
WRITE (1, 40) "HOURS    EXTEND    LINEAR               0.000"
WRITE (1, 20, ADVANCE = 'NO') INFLOW
WRITE (1, 40) "     0.000"
WRITE (1, 20, ADVANCE = 'NO') INFLOW
WRITE (1, 20) DBP * 0.2

!Write information for 1st river section
!Node list for tcs file
WRITE (2, 40) "Node1=S1"
```

```
WRITE (1, 40) "RIVER"              !River section header
WRITE (1, 40) "SECTION"
WRITE (1, 40) "S1"             !ISIS node label
IF (INTERPOLATE) THEN
    !set DTN
    WRITE (1, 20, ADVANCE = 'NO') (ROW_CH(XSEC_SP + 1) - ROW_CH(1)) / 2.0
ELSE
    WRITE (1, 20, ADVANCE = 'NO') ROW_CH(XSEC_SP + 1) - ROW_CH(1)
END IF
!Slope term set to default value as not used
WRITE (1, 40) "              0.0001"
COUNT = 0
FCOL = 1 + ((MAX_COLS - COLS(1)) / 2)
LCOL = FCOL + COLS(1) - 1
DO COL = FCOL, LCOL
  IF (CLG_BED(1, COL) < 9999) THEN
    COUNT = COUNT + 1
  END IF
END DO
WRITE (1, 10) COUNT + 2     !Number of cross section points
XSEC_CH(1) = ROW_CH(1)
MANNING = 1 / C(1, 1)
WRITE (1, 20, ADVANCE = 'NO') 0                !Cross chainage
WRITE (1, 20, ADVANCE = 'NO') WALL_HT     !Level
WRITE (1, 20, ADVANCE = 'NO') MANNING     !Manning's n
WRITE (1, 20, ADVANCE = 'NO') 1.000             !RPL (relative path length)
WRITE (1, 40, ADVANCE = 'NO') "           "
WRITE (1, 30, ADVANCE = 'NO') 0.0         !Zeros entered in GIS easting
WRITE (1, 30) 0.0                                    !and northings columns
!Set initial value of cross chainage to half one grid square
!i.e. middle of 1st column
XCH = NG_SIZE / 2
DO COL = FCOL, LCOL
  IF (CLG_BED(1, COL) < 9999) THEN
    !Mannings n approximated to 1/C
    IF (COL < LCOL) THEN
        IF (MAXC - C(1, COL + 1) < 1.0) THEN !Write cross section data
            MANNING = 1.0 / MAXC
        ELSE
            MANNING = 1.0 / C(1, COL)
        END IF
    ELSE
        MANNING = 1.0 / C(1, COL)
    END IF
    WRITE (1, 20, ADVANCE = 'NO') XCH                          !Cross chainage
    WRITE (1, 20, ADVANCE = 'NO') CLG_BED (1, COL)     !Level
    WRITE (1, 20, ADVANCE = 'NO') MANNING                      !Manning's n
    IF (MAXC - C(1, COL) > 1.0) THEN
        IF (COL > FCOL .AND. COL < LCOL) THEN
            IF (MAXC - C(1, COL - 1) < 1.0 .OR. MAXC - &
                C(1, COL + 1) < 1.0) THEN
                WRITE (1, 40, ADVANCE = 'NO') "*"
            ELSE
                WRITE (1, 40, ADVANCE = 'NO') " "
            END IF
        ELSE
            WRITE (1, 40, ADVANCE = 'NO') " "
        END IF
    ELSE
        WRITE (1, 40, ADVANCE = 'NO') " "
    END IF
    !RPL (relative path length)
    WRITE (1, 50, ADVANCE = 'NO') 1.000
    WRITE (1, 40, ADVANCE = 'NO') "           "
    !Zeros entered in GIS easting
```

```
    WRITE (1, 30, ADVANCE = 'NO') 0.0
    !and northings columns
    WRITE (1, 30) 0.0
  END IF
  !Increment cross chainage one grid square. i.e. next cross section point
  XCH = XCH + NG_SIZE
END DO
WRITE (1, 20, ADVANCE = 'NO') XCH             !Cross chainage
WRITE (1, 20, ADVANCE = 'NO') WALL_HT       !Level
WRITE (1, 20, ADVANCE = 'NO') MANNING         !Manning's n
WRITE (1, 20, ADVANCE = 'NO') 1.000                 !RPL (relative path length)
WRITE (1, 40, ADVANCE = 'NO') "          "
WRITE (1, 30, ADVANCE = 'NO') 0.0             !Zeros entered in GIS easting
WRITE (1, 30) 0.0                                       !and northings
columns

IF (INTERPOLATE) THEN
    WRITE (1, 40) "INTERPOLATE"
    WRITE (1, 40) "S1_i"
    WRITE (1, 20) (ROW_CH(XSEC_SP + 1) - ROW_CH(1)) / 2.0
END IF

!Write information for river sections
NODE = 2
DO ROW =  XSEC_SP + 1, PU_XSEC, XSEC_SP

  !Node list for tcs file
  WRITE (TMP_STR, 10) NODE
  WRITE (2, 40, ADVANCE = 'NO') "Node" // TRIM(ADJUSTL(TMP_STR)) // "=S"
  WRITE (TMP_STR, 10) NODE
  WRITE (2, 40) TRIM(ADJUSTL(TMP_STR))


  WRITE (1, 40) "RIVER"                             !River section header
  WRITE (1, 40) "SECTION"
  WRITE (TMP_STR, 10) NODE                    !ISIS node label
  WRITE (1, 40) "S" // TRIM(ADJUSTL(TMP_STR)) !ISIS node label
  IF (ROW < PU_XSEC) THEN
    IF (INTERPOLATE) THEN
        WRITE (1, 20, ADVANCE = 'NO') (ROW_CH(ROW + XSEC_SP) - &
            ROW_CH(ROW)) / 2.0   !set DTN
    ELSE
        WRITE (1, 20, ADVANCE = 'NO') ROW_CH(ROW + XSEC_SP) - &
            ROW_CH(ROW)           !set DTN
    END IF
  ELSE
    WRITE (1, 20, ADVANCE = 'NO') ROW_CH(ROWS) - ROW_CH(PU_XSEC)
  END IF
  !Slope term set to default value as not used
  WRITE (1, 40) "              0.0001"
  COUNT = 0
  FCOL = 1 + ((MAX_COLS - COLS(ROW)) / 2)
  LCOL = FCOL + COLS(ROW) - 1
  DO COL = FCOL, LCOL
    IF (CLG_BED(ROW, COL) < 9999) THEN
      COUNT = COUNT + 1
    END IF
  END DO
  WRITE (1, 10) COUNT + 2  !Number of cross section points
  XSEC_CH(NODE) = ROW_CH(ROW)

  WRITE (1, 20, ADVANCE = 'NO') 0             !Cross chainage
  WRITE (1, 20, ADVANCE = 'NO') WALL_HT     !Level
  WRITE (1, 20, ADVANCE = 'NO') MANNING       !Manning's n
  WRITE (1, 20, ADVANCE = 'NO') 1.000         !RPL (relative path length)
```

```
  WRITE (1, 40, ADVANCE = 'NO') "          "
  WRITE (1, 30, ADVANCE = 'NO') 0.0                    !Zeros enetered in GIS
easting
  WRITE (1, 30) 0.0                                        !and northings
columns

  !Set initial value of cross chainage to half one grid square
  !i.e. middle of 1st column
  XCH = NG_SIZE / 2
  DO COL = FCOL, LCOL
    IF (COL < LCOL) THEN
        IF (MAXC - C(ROW, COL + 1) < 1.0) THEN
            MANNING = 1.0 / MAXC
        ELSE
            MANNING = 1.0 / C(ROW, COL)
        END IF
    ELSE
        MANNING = 1.0 / C(ROW, COL)
    END IF
    IF (CLG_BED(ROW, COL) < 9999) THEN  !Write cross section data
      WRITE (1, 20, ADVANCE = 'NO') XCH !Cross chainage
      WRITE (1, 20, ADVANCE = 'NO') CLG_BED(ROW, COL) !Level
      WRITE (1, 20, ADVANCE = 'NO') MANNING                   !Manning's n
      IF (MAXC - C(ROW, COL) > 1.0) THEN
        IF (COL > FCOL .AND. COL < LCOL) THEN
            IF (MAXC - C(ROW, COL - 1) < 1.0 .OR. MAXC - &
                C(ROW, COL + 1) < 1.0) THEN
                    WRITE (1, 40, ADVANCE = 'NO') "*"
            ELSE
                WRITE (1, 40, ADVANCE = 'NO') " "
            END IF
        ELSE
            WRITE (1, 40, ADVANCE = 'NO') " "
        END IF
      ELSE
        WRITE (1, 40, ADVANCE = 'NO') " "
      END IF
      !RPL (relative path length)
      WRITE (1, 50, ADVANCE = 'NO') 1.000
      WRITE (1, 40, ADVANCE = 'NO') "          "
      !Zeros enetered in GIS easting
      WRITE (1, 30, ADVANCE = 'NO') 0.0
      !and northings columns
      WRITE (1, 30) 0.0
       END IF
       !Increment cross chainage one grid square.
       !i.e. next cross section point
    XCH = XCH + NG_SIZE
  END DO
  WRITE (1, 20, ADVANCE = 'NO') XCH             !Cross chainage
  WRITE (1, 20, ADVANCE = 'NO') WALL_HT  !Level
  WRITE (1, 20, ADVANCE = 'NO') MANNING  !Manning's n
  WRITE (1, 20, ADVANCE = 'NO') 1.000    !RPL (relative path length)
  WRITE (1, 40, ADVANCE = 'NO') "          "
  WRITE (1, 30, ADVANCE = 'NO') 0.0             !Zeros enetered in GIS easting
  WRITE (1, 30) 0.0


  IF (INTERPOLATE .AND. ROW < PU_XSEC) THEN
    WRITE (1, 40) "INTERPOLATE"
    WRITE (TMP_STR, 10) NODE                              !ISIS node label
    WRITE (1, 40) "S" // TRIM(ADJUSTL(TMP_STR)) // "_i"  !ISIS node label
    WRITE (1, 20) (ROW_CH(XSEC_SP + 1) - ROW_CH(1)) / 2.0
  END IF
```

```
  NODE = NODE + 1

END DO



!Write information for LAST river section
!Node list for tcs file
WRITE (TMP_STR, 10) NODE
WRITE (2, 40, ADVANCE = 'NO') "Node" // TRIM(ADJUSTL(TMP_STR)) // "=S"
WRITE (TMP_STR, 10) NODE
WRITE (2, 40) TRIM(ADJUSTL(TMP_STR))

WRITE (1, 40) "RIVER"                                !River section header
WRITE (1, 40) "SECTION"
WRITE (TMP_STR, 10) NODE                      !ISIS node label
WRITE (1, 40) "S" // TRIM(ADJUSTL(TMP_STR))       !ISIS node label
WRITE (1, 20, ADVANCE = 'NO') 0                !set DTN
!Slope term set to default value as not used
WRITE (1, 40) "              0.0001"
COUNT = 0
FCOL = 1 + ((MAX_COLS - COLS(ROWS)) / 2)
LCOL = FCOL + COLS(ROWS) - 1
DO COL = FCOL, LCOL
  IF (CLG_BED(ROWS, COL) < 9999) THEN
    COUNT = COUNT + 1
  END IF
END DO
WRITE (1, 10) COUNT + 2           !Number of cross section points
XSEC_CH(NODE) = ROW_CH(ROWS)

MANNING  = 1 / C(ROWS, 1)
WRITE (1, 20, ADVANCE = 'NO') 0                !Cross chainage
WRITE (1, 20, ADVANCE = 'NO') WALL_HT    !Level
WRITE (1, 20, ADVANCE = 'NO') MANNING    !Manning's n
WRITE (1, 20, ADVANCE = 'NO') 1.000            !RPL (relative path length)
WRITE (1, 40, ADVANCE = 'NO') "          "
WRITE (1, 30, ADVANCE = 'NO') 0.0        !Zeros entered in GIS easting
WRITE (1, 30) 0.0                                    !and northings columns
!Set initial value of cross chainage to half one grid square
!i.e. middle of 1st column
XCH = NG_SIZE / 2
DO COL = FCOL, LCOL                       !Write cross section data
  IF (CLG_BED(ROWS, COL) < 9999) THEN
    IF (COL < LCOL) THEN
        IF (MAXC - C(ROWS, COL + 1) < 1.0) THEN
            MANNING = 1.0 / MAXC
        ELSE
            MANNING  = 1.0 / C(ROWS, COL)
        END IF
    ELSE
        MANNING  = 1.0 / C(ROWS, COL)
    END IF
    WRITE (1, 20, ADVANCE = 'NO') XCH    !Cross chainage
    WRITE (1, 20, ADVANCE = 'NO') CLG_BED (ROWS, COL) !Level
    WRITE (1, 20, ADVANCE = 'NO') MANNING                    !Manning's n
    IF (MAXC - C(ROWS, COL) > 1.0) THEN
        IF (COL > FCOL .AND. COL < LCOL) THEN
            IF (MAXC - C(ROWS, COL - 1) < 1.0 .OR. MAXC - &
                C(ROW, COL + 1) < 1.0) THEN
                    WRITE (1, 40, ADVANCE = 'NO') "*"
            ELSE
                WRITE (1, 40, ADVANCE = 'NO') " "
            END IF
        ELSE
```

```
            WRITE (1, 40, ADVANCE = 'NO') " "
        END IF
    ELSE
        WRITE (1, 40, ADVANCE = 'NO') " "
    END IF
    WRITE (1, 50, ADVANCE = 'NO') 1.000      !RPL (relative path length)
    WRITE (1, 40, ADVANCE = 'NO') "          "
    WRITE (1, 30, ADVANCE = 'NO') 0.0           !Zeros entered in GIS easting
    WRITE (1, 30) 0.0                                    !and northings
columns
  END IF
  !Increment cross chainage one grid square. i.e. next cross section point
  XCH = XCH + NG_SIZE
END DO
WRITE (1, 20, ADVANCE = 'NO') XCH        !Cross chainage
WRITE (1, 20, ADVANCE = 'NO') WALL_HT    !Level
WRITE (1, 20, ADVANCE = 'NO') MANNING    !Manning's n
WRITE (1, 20, ADVANCE = 'NO') 1.000            !RPL (relative path length)
WRITE (1, 40, ADVANCE = 'NO') "          "
WRITE (1, 30, ADVANCE = 'NO') 0.0        !Zeros entered in GIS easting
WRITE (1, 30) 0.0

CLOSE (2)

!Write downstream HT boundary
IF (ND_DSBDY == .TRUE.) THEN
    WRITE (1, 40) "NCDBDY #revision#1"
    WRITE (TMP_STR, 10) NODE
    WRITE (TMP_STR1, 40) "S" // TRIM( ADJUSTL (TMP_STR))
    WRITE (1, 40) TMP_STR1 // "S1"
    WRITE (1, 40) "CRITICAL"
!    WRITE (1, 40) "AUTO            BED"
    WRITE (1, 40) "         0"
    WRITE (1, 40) &
        "         1     HOURS    EXTENDNOOVERRIDE     0.000LINEAR"
    WRITE (1, 40) "     0.000     0.000"
ELSE
    WRITE (1, 40) "HTBDY"
    WRITE (TMP_STR, 10) NODE
    WRITE (1, 40) "S" // TRIM( ADJUSTL (TMP_STR))
    WRITE (1, 10, ADVANCE = 'NO') DBP
    WRITE (1, 40) "               HOURS    EXTEND    LINEAR"
    DO ROW = 1, DBP
      IF (MIN_WL > TIDE(ROW, 2)) THEN
        DSBDY_WL = MIN_WL
      ELSE
        DSBDY_WL = TIDE(ROW, 2)
      END IF
      !Write data to downstream boundary
      WRITE (1, 20, ADVANCE = 'NO') DSBDY_WL
      WRITE (1, 20) TIDE (ROW, 1)
    END DO
END IF

!Header for initial conditions
WRITE (1, 40) "INITIAL CONDITIONS"
WRITE (1, 40, ADVANCE = 'NO') " label   ?      flow     stage froude no  "
WRITE (1, 40) "velocity     umode    ustate         z"

!Write initial conditions to file
!1ST Xsec
WRITE (1, 40, ADVANCE = 'NO') "S1              0.000"
!Set initial level at all xsecs to initial level in DSBDY
WRITE (1, 20, ADVANCE = 'NO') TIDE(1, 2)
WRITE (1, 40) "     0.000     0.000     0.000     0.000     0.000"
```

```
IF (INTERPOLATE) THEN
  WRITE (1, 40, ADVANCE = 'NO') "S1_i             0.000"
  WRITE (1, 40, ADVANCE = 'NO') "      0.000"
  !Set initial level at all xsecs to initial level in DSBDY
  WRITE (1, 20, ADVANCE = 'NO') TIDE(1, 2)
  WRITE (1, 40) "     0.000     0.000     0.000     0.000     0.000"
END IF

NODE = 2
DO ROW = XSEC_SP + 1, PU_XSEC, XSEC_SP
  WRITE (TMP_STR, 10) NODE
  WRITE (1, 40, ADVANCE = 'NO') "S" // ADJUSTL (TMP_STR)
  WRITE (1, 40, ADVANCE = 'NO') "      0.000"
  !Set initial level at all xsecs to initial level in DSBDY
  WRITE (1, 20, ADVANCE = 'NO') TIDE(1, 2)
  WRITE (1, 40) "     0.000     0.000     0.000     0.000     0.000"
  IF (INTERPOLATE .AND. ROW < PU_XSEC) THEN
      WRITE (TMP_STR, 10) NODE
      WRITE (TMP_STR, 40) TRIM(ADJUSTL (TMP_STR)) // "_i"
      WRITE (1, 40, ADVANCE = 'NO') "S" // ADJUSTL (TMP_STR)
      WRITE (1, 40, ADVANCE = 'NO') "      0.000"
      !Set initial level at all xsecs to initial level in DSBDY
      WRITE (1, 20, ADVANCE = 'NO') TIDE(1, 2)
      WRITE (1, 40) "     0.000     0.000     0.000     0.000     0.000"
  END IF
  NODE = NODE + 1
END DO

!Last Xsec
WRITE (TMP_STR, 10) NODE
WRITE (1, 40, ADVANCE = 'NO') "S" // ADJUSTL (TMP_STR)
WRITE (1, 40, ADVANCE = 'NO') "      0.000"
!Set initial level at all xsecs to initial level in DSBDY
WRITE (1, 20, ADVANCE = 'NO') TIDE(1, 2)
WRITE (1, 40) "     0.000     0.000     0.000     0.000     0.000"


!Write dummy GIS data to file
WRITE (1, 40) "GISINFO"
!1ST xsec
WRITE (1, 40, ADVANCE = 'NO') "RIVER SECTION "
WRITE (1, 40, ADVANCE = 'NO') "S1"
WRITE (1, 40) " 0 0 0 0 0"

NODE = 2
DO ROW = XSEC_SP + 1, PU_XSEC, XSEC_SP
  WRITE (1, 40, ADVANCE = 'NO') "RIVER SECTION "
  WRITE (TMP_STR, 10) NODE
  WRITE (1, 40, ADVANCE = 'NO') "S" // TRIM (ADJUSTL (TMP_STR))
  WRITE (1, 40) " 0 0 0 0 0"
  NODE = NODE + 1
END DO

!Last xsec
WRITE (1, 40, ADVANCE = 'NO') "RIVER SECTION "
WRITE (TMP_STR, 10) NODE
WRITE (1, 40, ADVANCE = 'NO') "S" // TRIM (ADJUSTL (TMP_STR))
WRITE (1, 40) " 0 0 0 0 0"

CLOSE (1)

10 FORMAT (I10)
20 FORMAT (F10.3)
30 FORMAT (F10.2)
40 FORMAT (A)
```

```
50 FORMAT (F9.3)


END SUBROUTINE

!-----------------------------------------------------------------------

SUBROUTINE RUN_ISIS (F, FILENAME, RUN_TIME, SS)
!Subroutine to run the ISIS model

REAL,         INTENT(IN) :: RUN_TIME           !Period for which model to be run

INTEGER,      INTENT(IN) :: F                          !File number

LOGICAL,     INTENT(IN) :: SS

INTEGER STRLEN

CHARACTER (200), INTENT(IN) :: FILENAME !Filename for ISIS dat file
CHARACTER (200) RUN_FN           !Filename for ISIS ief file
CHARACTER (200) IC_FN            !Filename for ISIS steady state run results
CHARACTER (200) RESULTS_FN !Filename for ISIS results file
CHARACTER (200) TCS_FN     !tcs file filename (contains csv output format)
CHARACTER (10) TMP_STR     !Temporary string holder

STRLEN = LEN_TRIM(FILENAME)
RUN_FN = FILENAME(1:STRLEN - 3) // "ief"
IC_FN = FILENAME(1:STRLEN - 3) // "zzs"
RESULTS_FN = FILENAME(1:STRLEN - 3) // "zzn"
TCS_FN = FILENAME(1:STRLEN - 3) // "tcs"

!Write temporary batch file
OPEN (F, FILE = "C:\TEMP\TMP.BAT")

WRITE (F, 10, ADVANCE = 'NO') 'C:\isis\bin\isisf32.exe -sd "'
WRITE (F, 10, ADVANCE = 'NO') RUN_FN
WRITE (F, 10) '"'

CLOSE (F)


!ISIS STEADY DIRECT RUN TO GET INITIAL CONDITIONS
!ORCALCULATE MINIMUM WATER LEVELS

OPEN (F, FILE = RUN_FN)
!Write the ISIS .ief file (ISIS run settings)

WRITE (F, 10) "[ISIS Event Header]"
WRITE (F, 10, ADVANCE = 'NO') "Datafile="
WRITE (F, 10) FILENAME
WRITE (F, 10) "[ISIS Event Details]"
WRITE (F, 10) "RunType=Steady"
WRITE (F, 10, ADVANCE = 'NO') "InitialConditions="
WRITE (F, 10) IC_FN
WRITE (F, 10) "Start=0"
WRITE (F, 10) "Transcritical=0"
WRITE (F, 10) "Slot=1"
WRITE (F, 10) "SuppressWindowsOutput=1"
WRITE (F, 10) "RefineBridgeSecProps=0"
WRITE (F, 10) "MathRules=1"
WRITE (F, 10) "SolveDHEqualsZeroAtStart=1"
WRITE (F, 10) "PSTruePerimeter=1"
WRITE (F, 10) "RulesAtTimeZero=1"
WRITE (F, 10) "RulesOnFirstIteration=0"
WRITE (F, 10) "ResetTimesAfterPos=1"
```

```
WRITE (F, 10) "UseFPSModularLimit=1"
WRITE (F, 10) "OutputConvergencePlotBMP=0"

CLOSE (F)


CALL SYSTEM ("C:\TEMP\TMP.BAT")


!--------------------------------ISIS UNSTEADY RUN-----------------------
IF (.NOT. SS) THEN

    OPEN (F, FILE = RUN_FN)
    !Write new .ief file

    WRITE (F, 10) "[ISIS Event Header]"
    WRITE (F, 10, ADVANCE = 'NO') "Datafile="
    WRITE (F, 10) FILENAME
    WRITE (F, 10) "[ISIS Event Details]"
    WRITE (F, 10) "RunType=Unsteady"
    WRITE (F, 10, ADVANCE = 'NO') "InitialConditions="
    WRITE (F, 10) IC_FN
    WRITE (F, 10) "Start=0"
    WRITE (F, 10, ADVANCE = 'NO') "Finish="
    WRITE (TMP_STR, 20) RUN_TIME
    WRITE (F, 10) TRIM(ADJUSTL(TMP_STR))
    WRITE (F, 10) "SaveInterval=50"
    WRITE (F, 10) "SuppressWindowsOutput=1"
    WRITE (F, 10) "InitialTimestep=20"
    WRITE (F, 10) "MinimumTimestep=1"
    WRITE (F, 10) "SaveInterval=300"
    WRITE (F, 10) "AdaptiveTimestep=1"
    WRITE (F, 10) "RefineBridgeSecProps=0"
    WRITE (F, 10) "MathRules=1"
    WRITE (F, 10) "SolveDHEqualsZeroAtStart=1"
    WRITE (F, 10) "PSTruePerimeter=1"
    WRITE (F, 10) "RulesAtTimeZero=1"
    WRITE (F, 10) "RulesOnFirstIteration=0"
    WRITE (F, 10) "ResetTimesAfterPos=1"
    WRITE (F, 10) "UseFPSModularLimit=1"
    WRITE (F, 10) "OutputGXYErrors=0"
    WRITE (F, 10) "OutputConvergencePlotBMP=0"

    CLOSE (F)

    IF (.NOT. CHECK_ISIS(F, FILENAME)) THEN

        !Run unsteady model run
        CALL SYSTEM ("C:\TEMP\TMP.BAT")

        OPEN (F, FILE = "C:\TEMP\TMP.BAT")

        !Write csv output
        WRITE (F, 10, ADVANCE = 'NO') &
            'C:\isis\bin\Tabularcsv.exe -silent -tcs "'
        WRITE (F, 10, ADVANCE = 'NO') TRIM(TCS_FN)
        WRITE (F, 10, ADVANCE = 'NO') '" "'
        WRITE (F, 10, ADVANCE = 'NO') TRIM(RESULTS_FN)
        WRITE (F, 10) '"'

        CLOSE (F)

        CALL SYSTEM ("C:\TEMP\TMP.BAT")
    END IF
```

```
END IF

10 FORMAT (A)
20 FORMAT (F10.0)

END SUBROUTINE


SUBROUTINE READ_ISIS (F, FILENAME, WARMUP_TIME, TSPRP, ROWS, NODE_SP, &
    WL, RP, XSEC_CH, ROW_CH)
!Subroutine to read the ISIS csv results file

REAL, INTENT(IN)                  :: WARMUP_TIME      !Warm up time for ISIS run

!ISIS results array: water levels at each row
REAL,  DIMENSION(:,:), INTENT(OUT)       :: WL

REAL,    DIMENSION(:),    INTENT(IN) :: ROW_CH
REAL,    DIMENSION(:),    INTENT(IN) :: XSEC_CH

INTEGER, INTENT(IN)               :: TSPRP            !Time steps per repeat
period
INTEGER, INTENT(IN)               :: F                !File identifier
!Number of rows in routing model grid
INTEGER, INTENT(IN)               :: ROWS
!Spacing of ISIS nodes, in grid rows
INTEGER, INTENT(IN)               :: NODE_SP

!Repetition period from file true if 1
!tidal cycle / false if 1 spring/neap cycle
CHARACTER (5),  INTENT(IN) :: RP
CHARACTER (200), INTENT(IN)       :: FILENAME         !Filename


REAL TIDAL_PERIOD   !Tidal period in hours
REAL RESULTS_INT    !Results interval in hours
REAL INC
REAL CH1, CH2

!Period of results required (1 tidal cycle or spring/neap cycle)
REAL RESULTS_PERIOD
!Lunar month relative to sun (slightly longer because of movement
!of earth around sun)
REAL LM_SOL

!Array to store to times of the ISIS results
REAL,  DIMENSION(100000)    :: R_TIMES
REAL,    DIMENSION(100000)    :: R_LEVELS1
REAL,    DIMENSION(100000)    :: R_LEVELS2

INTEGER N, M, A            !Counters
INTEGER IO                 !IOSTAT variable
INTEGER NODE1, NODE2
INTEGER NR_NODES    !Number of river sections in ISIS model
INTEGER LAST_NODE
INTEGER PU_NODE

LOGICAL LAST_ROW

CHARACTER CHr

LM_SOL = LUNAR_MONTH * (365.0 / 364.0)
TIDAL_PERIOD = 12 / (1 - (1 / LM_SOL))

IF ((ROWS - 1) - (INT((ROWS - 1) / NODE_SP) * NODE_SP) > 0) THEN
```

```
  NR_NODES = INT((ROWS - 1) / NODE_SP) + 2
ELSE
  NR_NODES = INT((ROWS - 1) / NODE_SP) + 1
END IF
PU_NODE = ((NR_NODES - 2) * NODE_SP) + 1

RESULTS_PERIOD = TIDAL_PERIOD
IF (RP /= "True") THEN
    !Note that spring/neap cycle is set to 29 tidal cycles instead
    !of half a lunar month since it needs to be an integer number
    !of tidal cycles
  RESULTS_PERIOD = RESULTS_PERIOD * 29
END IF
RESULTS_INT = RESULTS_PERIOD / TSPRP

!Open ISIS results file
OPEN (F, FILE = FILENAME)

!Skip 1st 5 lines
DO N = 1, 5
  READ (F, 10) CHR
END DO

!Read 6th line upto 1st comma
CHR = "A"
IO = 0
DO WHILE ((CHR /= ',') .AND. (IO >= 0))
  READ (F, 10, IOSTAT = IO, ADVANCE = 'NO') CHR
END DO


!Read times of results from ISIS CSV file
IO = 0
N = 1
DO WHILE (IO == 0)
  READ (F, 20, IOSTAT = IO, ADVANCE = 'NO') R_TIMES (N)
  N = N + 1
END DO

CALL READ_LINE(F, NODE1, WARMUP_TIME, RESULTS_INT, &
    R_TIMES, TSPRP, R_LEVELS1)
CH1 = XSEC_CH(NODE1)

A = 1
LAST_ROW = .FALSE.
DO WHILE (.NOT. LAST_ROW)

  CALL READ_LINE(F, NODE2, WARMUP_TIME, RESULTS_INT, &
    R_TIMES, TSPRP, R_LEVELS2)
  CH2 = XSEC_CH(NODE2)

  DO WHILE ((ROW_CH(A) <= (CH2 + 0.00001)) .AND. (.NOT. LAST_ROW))
    DO M = 1, TSPRP
      WL(A, M) = R_LEVELS1(M) + ((R_LEVELS2(M) - R_LEVELS1(M))* &
        ((ROW_CH(A) - CH1) / (CH2 - CH1)))
    END DO
    IF (A < ROWS) THEN
      A = A + 1
    ELSE
      LAST_ROW = .TRUE.
    END IF
  END DO

  IF (.NOT. LAST_ROW) THEN
```

```
    CALL READ_LINE(F, NODE1, WARMUP_TIME, RESULTS_INT, &
        R_TIMES, TSPRP, R_LEVELS1)
    CH1 = XSEC_CH(NODE1)

    DO WHILE ((ROW_CH(A) <= (CH1 + 0.00001)) .AND. (.NOT. LAST_ROW))
      DO M = 1, TSPRP
        WL(A, M) = R_LEVELS2(M) + ((R_LEVELS1(M) - R_LEVELS2(M))* &
            ((ROW_CH(A) - CH2) / (CH1 - CH2)))
      END DO
      IF (A < ROWS) THEN
        A = A + 1
      ELSE
        LAST_ROW = .TRUE.
      END IF
    END DO

  END IF
END DO


CLOSE (F)

10 FORMAT (A1)
20 FORMAT (F8.0)

CONTAINS !-----------------------------------------------------------------

SUBROUTINE READ_LINE (F, NODE, WARMUP_TIME, RESULTS_INT, &
    R_TIMES, TSPRP, N_LEVELS)
!Subroutine to read one line from csv output

10 FORMAT (A1)
20 FORMAT (F8.0)
30 FORMAT (I10)

REAL, DIMENSION(:), INTENT(IN)  :: R_TIMES
REAL, DIMENSION(:), INTENT(OUT) :: N_LEVELS

REAL, INTENT(IN) :: RESULTS_INT
REAL, INTENT(IN) :: WARMUP_TIME

INTEGER, INTENT(OUT) :: NODE
INTEGER, INTENT(IN)  :: F         !File identifier
!Timesteps per repeat period (1 tidal cycle or 1 spring/neap cycle)
INTEGER, INTENT(IN)  :: TSPRP

REAL R_LEVELS(100000)      !Array to store levels from ISIS results file

REAL RTIME                        !Time of result to be derived from ISIS file

INTEGER N, M, IO

CHARACTER CHR

!Skip 1st character of next line ('S' from node label)
READ (F, 10, ADVANCE = 'NO') CHR

!Determine chainage from ISIS node label
READ (F, 30, ADVANCE = 'NO') NODE

!Read 1st line of level results
IO = 0
N = 1
M = 1
DO WHILE (IO == 0)
```

```
  READ (F, 20, IOSTAT = IO, ADVANCE = 'NO') R_LEVELS (N)
  N = N + 1
END DO

!Interpolate ISIS results to determine results at required intervals
!for 1st line of results
M = 1



DO N = 0, TSPRP - 1
  RTIME = WARMUP_TIME + (N * RESULTS_INT)
  !Find 1st value in ISIS result times greater than required
  DO WHILE ((R_TIMES(M) <= RTIME) .AND. (M < 10000))
    M = M + 1                                                          !result
time
  END DO

  !If m = 1 then first ISIS result is the one required
  !(also trying to interpolate would cause an error)
  IF (M == 1) THEN
    N_LEVELS(N + 1) = R_LEVELS(M)
  ELSE
    N_LEVELS(N + 1) = R_LEVELS(M) - ((R_LEVELS(M) - R_LEVELS(M - 1)) * &
        ((R_TIMES(M) - RTIME) / (R_TIMES(M) - R_TIMES(M - 1))))
  END IF
END DO

END SUBROUTINE READ_LINE

END SUBROUTINE READ_ISIS
!-----------------------------------------------------------------------

SUBROUTINE READ_SS(F, FILENAME, ROWS, NODE_SP, WL_SS)
!Subroutine to read results from an ISIS steady-state model run

REAL,  DIMENSION(:), INTENT(OUT)  :: WL_SS      !ISIS results array

INTEGER, INTENT(IN) :: F                !File identifier
INTEGER, INTENT(IN) :: ROWS                    !Number of rows in routing model
grid
INTEGER, INTENT(IN) :: NODE_SP          !Spacing of ISIS nodes, in grid rows

CHARACTER (200), INTENT(IN)       :: FILENAME         !Filename

REAL,   DIMENSION(:),     ALLOCATABLE :: WL_SS_NODE

REAL ROW_CH

INTEGER N
INTEGER NR_NODES
INTEGER ROW, CH1, CH2, N1, N2, N1R, N2R

CHARACTER CHR
CHARACTER (LEN = 24) DUMMY_STR

IF ((ROWS - 1) - (INT((ROWS - 1) / NODE_SP) * NODE_SP) > 0) THEN
  NR_NODES = INT((ROWS - 1) / NODE_SP) + 2
ELSE
  NR_NODES = INT((ROWS - 1) / NODE_SP) + 1
END IF


ALLOCATE (WL_SS_NODE(NR_NODES))
```

```
!Open ISIS results file
OPEN (F, FILE = FILENAME)

!Skip 1st 3 lines
DO N = 1, 3
  READ (F, 10) CHR
END DO

DO N = 1, NR_NODES
    READ (F, 20, ADVANCE = 'NO') DUMMY_STR
    READ (F, 30) WL_SS_NODE(N)
END DO

CLOSE (F)


ROW_CH = 0.5
CH1 = 0
CH2 = NODE_SP + 1
N1R = 1
N2R = NODE_SP + 1
N1 = 1
N2 = 2
ROW = 0
DO WHILE (ROW <  ROWS)
    ROW = ROW + 1
    IF (ROW == N1R) THEN
        WL_SS(ROW) = WL_SS_NODE(N1)
    ELSE
        IF (ROW < N2R) THEN
            WL_SS(ROW) = (WL_SS_NODE(N1) * (N2R - ROW) / NODE_SP) + &
                (WL_SS_NODE(N2) * (ROW - N1R) / NODE_SP)
        ELSE
            N1R = N2R
            N2R = N2R + NODE_SP
            N1 = N1 + 1
            N2 = N2 + 1
            ROW = ROW - 1
            IF (N2R > ROWS) THEN
                N2R = ROWS
            END IF
        END IF
    END IF
END DO


10 FORMAT (A1)
20 FORMAT (A)
30 FORMAT (F10.0)

END SUBROUTINE



FUNCTION CHECK_ISIS(F, FILENAME)
!Function to check ISIS zzd file to determine
!whether model run was successful

CHARACTER (200), INTENT (IN)            :: FILENAME
INTEGER, INTENT(IN) :: F

CHARACTER (LEN=200) :: FILENAME_ZZD
CHARACTER (LEN=200) :: LINE
```

```
LOGICAL :: CHECK_ISIS

INTEGER STLEN, IO

STLEN = LEN_TRIM(FILENAME)
FILENAME_ZZD = FILENAME(1:STLEN - 3) // "zzd"

CHECK_ISIS = .TRUE.

OPEN (F, FILE = FILENAME_ZZD)

DO
    READ (F, 10, IOSTAT = IO) LINE

    IF (LINE(1:23) == " successful solution in" .OR. &
        LINE(1:13) == "run completed") THEN
        CHECK_ISIS = .FALSE.
    END IF

    IF (CHECK_ISIS == .FALSE. .OR. IO /= 0) EXIT

END DO

CLOSE (F)

10 FORMAT (A)

END FUNCTION


END MODULE ISIS
```

Module CLG

Subroutines	Purpose	Page nr.
SETUP_CLG	Calculate interpolated grid sections and dimensions	416
CREATE_CLG	Calculate interpolated grid properties and weightings needed to calculate interpolated bed levels from model grid cells.	418
GET_CLG_WTGS	Calculate weightings needed to interpolate calculated water levels onto model grid cells	423

```
MODULE CLG1

IMPLICIT NONE

REAL,    PARAMETER :: PI1 = 3.14159265358979

CONTAINS

SUBROUTINE SETUP_CLG(NG_ROWS, NG_COLS, GRID_SIZE, NG_SIZE, SETUP_PTS, &
    NR_PTS, DTN, LINE_START, ARC_ORIGIN, LINE_INC, LINE_LEN, ARC_INC, &
    ARC_LEN)
!Subroutine to define an interpolated grid of bed levels based on a
!cenreline made up of line and arc sections
!defined by a set of predefined points

!Predefined points giving centreline as points on source grid
REAL,  DIMENSION(:,:),           INTENT(IN)    :: SETUP_PTS
REAL,    DIMENSION(:,:),     INTENT(OUT) :: LINE_START
REAL,    DIMENSION(:,:),     INTENT(OUT) :: ARC_ORIGIN
!Length of grid 'squares' for arc, in radians (set so ARC_LENGTH is
!integer multiple and max length in m < GRID_SIZE)
REAL,  DIMENSION(:),       INTENT(OUT) :: ARC_INC
!Cell lengths for straight sections
REAL,    DIMENSION(:),       INTENT(OUT) :: LINE_INC
REAL,    DIMENSION(:),       INTENT(OUT) :: DTN

INTEGER,     DIMENSION(:),   INTENT(IN)  :: NG_COLS
INTEGER,     DIMENSION(:),   INTENT(OUT) :: ARC_LEN
INTEGER,     DIMENSION(:),   INTENT(OUT) :: LINE_LEN

REAL,    INTENT(IN) :: GRID_SIZE
REAL,    INTENT(IN) :: NG_SIZE

INTEGER,     INTENT(OUT) :: NG_ROWS
INTEGER,     INTENT(IN)  :: NR_PTS

REAL,    DIMENSION(:,:), ALLOCATABLE :: LINE_END

REAL NG_WIDTH
REAL DX, DY
REAL A1, A2, A3, A4
REAL ADIFF
REAL L
REAL OFFSET
REAL ARC_LENGTH
REAL LINE_LENGTH
```

```
INTEGER N

ALLOCATE (LINE_END(NR_PTS - 1, 2))



!Calculate the direction (angle in radians relative to source grid) of
!each line connecting the setup points
DO N = 1, NR_PTS - 1
  DX = SETUP_PTS(N + 1, 1) - SETUP_PTS(N, 1)
  DY = SETUP_PTS(N + 1, 2) - SETUP_PTS(N, 2)

  IF (ABS(DY) < 0.0000000001) THEN
    IF (DX > 0) THEN
      DTN(N) = PI1 / 2
    ELSE
      DTN(N) = 3 * PI1 / 2
    END IF
  ELSE
    DTN(N) = ATAN (DX / DY)
    IF (DY < 0) THEN
      DTN(N) = DTN(N) + PI1
    ELSE
      IF (DTN(N) < 0) THEN
        DTN(N) = DTN(N) + (2 * PI1)
      END IF
    END IF
  END IF
END DO

!Determine start and end points for straight
!line sections and arc lengths etc
LINE_START(1, 1) = SETUP_PTS(1, 1)
LINE_START(1, 2) = SETUP_PTS(1, 2)
LINE_END(NR_PTS - 1, 1) = SETUP_PTS(NR_PTS, 1)
LINE_END(NR_PTS - 1, 2) = SETUP_PTS(NR_PTS, 2)
NG_ROWS = 0
DO N = 1, NR_PTS - 2

  NG_WIDTH = NG_COLS(N) * NG_SIZE

  A1 = DTN(N)
  A2 = DTN(N + 1)

  ADIFF = A2 - A1

  IF (ADIFF > PI1) THEN
    ADIFF = ADIFF - (2 * PI1)
  ELSE
    IF (ADIFF < -PI1) THEN
      ADIFF = ADIFF + (2 * PI1)
    END IF
  END IF

  IF (ADIFF > 0) THEN !Clockwise curve
    IF (A1 - A2 + PI1 < 2 * PI1) THEN
      A4 = (A1 - A2 + PI1) / 2
    ELSE
      A4 = (A1 - A2 - PI1) / 2
    END IF
    A3 = A4 + A2
  ELSE !Anticlockwise curve
    IF  (A2 - A1 + PI1 < 2 * PI1) THEN
      A4 = (A2 - A1 + PI1) / 2
    ELSE
```

```
      A4 = (A2 - A1 - PI1) / 2
    END IF
    A3 = A4 + A1 + PI1
  END IF

  L = (NG_WIDTH / 2) / (SIN(A4))
  OFFSET = L * COS(A4)

  LINE_START(N + 1, 1) = SETUP_PTS(N + 1, 1) + (OFFSET * SIN(A2))
  LINE_START(N + 1, 2) = SETUP_PTS(N + 1, 2) + (OFFSET * COS(A2))
  LINE_END(N, 1) = SETUP_PTS(N + 1, 1) - (OFFSET * SIN(A1))
  LINE_END(N, 2) = SETUP_PTS(N + 1, 2) - (OFFSET * COS(A1))

  LINE_LENGTH  = (((LINE_END(N, 1) - LINE_START(N, 1)) ** 2) + &
    ((LINE_END(N, 2) - LINE_START(N, 2)) ** 2)) ** 0.5
  LINE_LEN(N) = INT((LINE_LENGTH / NG_SIZE) + 0.999)
  LINE_INC(N) = LINE_LENGTH / LINE_LEN(N)

  DX = L * SIN(A3)
  DY = L * COS(A3)

  ARC_ORIGIN(N, 1) = SETUP_PTS(N + 1, 1) + DX
  ARC_ORIGIN(N, 2) = SETUP_PTS(N + 1, 2) + DY

  ARC_LENGTH = PI1 - (A4 * 2)
  ARC_LEN(N) = ABS(INT((ARC_LENGTH * NG_WIDTH / GRID_SIZE) + 0.999))
  ARC_INC(N) = ABS(ARC_LENGTH / ARC_LEN(N))

  NG_ROWS = NG_ROWS + ARC_LEN(N) + LINE_LEN(N)

END DO
LINE_LENGTH  = (((LINE_END(NR_PTS - 1, 1) - &
    LINE_START(NR_PTS - 1, 1)) ** 2) + ((LINE_END(NR_PTS - 1, 2) - &
    LINE_START(NR_PTS - 1, 2)) ** 2)) ** 0.5
LINE_LEN(NR_PTS - 1) = INT((LINE_LENGTH / NG_SIZE) + 0.999)
LINE_INC(NR_PTS - 1) = LINE_LENGTH / LINE_LEN(NR_PTS - 1)

NG_ROWS = NG_ROWS + LINE_LEN(NR_PTS - 1)

END SUBROUTINE

!-------------------------------------------------------------------------

SUBROUTINE CREATE_CLG(ROWS, COLS, NG_ROWS, NG_COLS, NG_COLS_MAX, &
    ROW_WTH, GRID_LENGTH, GRID_WIDTH, GRID_SIZE, NR_PTS, LINE_START, &
    ARC_ORIGIN, LINE_INC, LINE_LEN, ARC_INC, ARC_LEN, ROW_DTN, ROW_CH, &
    NG_SIZE, NG_LOC, NG_AREA, DTN, CELL_LEN, WT_XCOORDS, WT_YCOORDS, BLWT)
!Subroutine to calculate interpolated grid properties and weightings
!to calculate interpolated bed levels

IMPLICIT NONE

!Array to store area of curvilinear grid cells
REAL,  DIMENSION(:,:),           INTENT(OUT) :: NG_AREA
!Array to store location of curvilinear grid cell on source grid
REAL,  DIMENSION(:,:,:),   INTENT(OUT) :: NG_LOC
!Length of grid 'squares' for arc, in radians (set so ARC_LENGTH
!is integer multiple and max length in m < GRID_SIZE)
REAL,  DIMENSION(:),       INTENT(IN)  :: ARC_INC
!Cell lengths for straight sections
REAL,   DIMENSION(:),       INTENT(IN)  :: LINE_INC
!Array to store coordinates of the start of each straight line section
REAL,   DIMENSION(:,:),     INTENT(IN)  :: LINE_START
REAL,   DIMENSION(:,:),     INTENT(IN)  :: ARC_ORIGIN
!Array to store the direction (angle in radians) of each staight line
```

```
!section of curvilinear grid
REAL,    DIMENSION(:),       INTENT(IN)  :: DTN
REAL,    DIMENSION(:,:),     INTENT(OUT) :: CELL_LEN
!Weightings for calculating CLG bed levels from source grid
REAL,    DIMENSION(:,:,:),   INTENT(OUT) :: BLWT
REAL,    DIMENSION(:),       INTENT(OUT) :: ROW_DTN
REAL,    DIMENSION(:),       INTENT(OUT) :: ROW_CH

!Line length in grid cells
INTEGER,     DIMENSION(:),     INTENT(IN)  :: LINE_LEN
!Length of circular arc sections of grid, in grid elements
INTEGER,     DIMENSION(:),     INTENT(IN)  :: ARC_LEN
!Length of each CLG section of grid cells
INTEGER,    DIMENSION(:),       INTENT(IN)  :: NG_COLS
!X Coordinate (row/col in source grid) for bed level weightings
INTEGER,    DIMENSION(:,:,:),   INTENT(OUT) :: WT_XCOORDS
!Y Coordinate (row/col in source grid) for bed level weightings
INTEGER,    DIMENSION(:,:,:),   INTENT(OUT) :: WT_YCOORDS
!Width of each CLG row in grid cells
INTEGER,    DIMENSION(:),       INTENT(OUT) :: ROW_WTH

REAL,    INTENT(IN) :: NG_SIZE       !Width of new grid cells in metres
REAL,    INTENT(IN) :: GRID_WIDTH
REAL,    INTENT(IN) :: GRID_LENGTH
REAL,    INTENT(IN) :: GRID_SIZE

INTEGER,    INTENT(IN) :: ROWS
INTEGER,    INTENT(IN) :: COLS
INTEGER,    INTENT(IN) :: NG_ROWS
INTEGER,    INTENT(IN) :: NG_COLS_MAX
INTEGER,    INTENT(IN) :: NR_PTS            !Number of setup points

!Length of circular arc sections of grid, in radians
REAL,  DIMENSION(:),       ALLOCATABLE :: ARC_LENGTH
!Lenght of straight sections in metres
REAL,   DIMENSION(:),       ALLOCATABLE :: LINE_LENGTH


REAL A1, A2, A3, A4
 !Angle of line connecting arc origin start of
 !arc / to centre of current cell
REAL A5, A6
REAL ADIFF
REAL DX, DY
REAL L1, L2
REAL L, AN
REAL RADIUS                 !distance from arc origin to centre of cell
REAL R1, R2                 !Cell outside/inside radius from origin
REAL OFFSET                 !Offset from setup point to line start/end
REAL LINE_X, LINE_Y         !Coordinates of CLG centreline at current row
REAL RW, CL, X, Y
REAL A, B, C
REAL CH
REAL NG_WIDTH

INTEGER ROW, COL
INTEGER ROW_OS              !Row offset (for centering grid data in array)
INTEGER L_ROW
INTEGER ARC_ROW
INTEGER N                               !Counter


NG_WIDTH = NG_COLS_MAX * NG_SIZE

!Determine coordinates of each curvilinear grid cell
```

```
ROW = 1
CH = 0
DO N = 1, NR_PTS - 2

  !Straight section
  DO L_ROW = 1, LINE_LEN(N)

    ROW_WTH(ROW) = NG_COLS(N)
    ROW_OS = (NG_COLS_MAX - NG_COLS(N)) / 2
    CH = CH + (LINE_INC(N) / 2)
    ROW_CH(ROW) = CH
    ROW_DTN(ROW) = DTN(N)
    LINE_X = LINE_START(N, 1) + &
        ((L_ROW - 0.5) * LINE_INC(N)) * SIN(DTN(N))
    LINE_Y = LINE_START(N, 2) + &
        ((L_ROW - 0.5) * LINE_INC(N)) * COS(DTN(N))

    DO COL = 1 + ROW_OS , NG_COLS_MAX - ROW_OS
      NG_AREA(ROW, COL) = NG_SIZE * LINE_INC(N)
      CELL_LEN(ROW, COL) = LINE_INC(N)
      NG_LOC(ROW, COL, 1) = LINE_X + (((COL - ROW_OS) - &
        (NG_COLS(N) / 2.0) - 0.5) * NG_SIZE * COS(DTN(N)))
      NG_LOC(ROW, COL, 2) = LINE_Y - (((COL - ROW_OS) - &
        (NG_COLS(N) / 2.0) - 0.5) * NG_SIZE * SIN(DTN(N)))
    END DO
    CH = CH + (LINE_INC(N) / 2)
    ROW = ROW + 1
  END DO

  !Arc section
  A1 = DTN(N)
  A2 = DTN(N + 1)

  ADIFF = A2 - A1

  !Adjust ADIFF so it is in the range +or- 180 degrees (PI1 radians)
  IF (ADIFF < -PI1) THEN
    ADIFF = ADIFF + (2 * PI1)
  ELSE
    IF (ADIFF > PI1) THEN
      ADIFF = ADIFF - (2 * PI1)
    END IF
  END IF

   IF (ADIFF > 0) THEN !Clockwise curve
    IF (A1 - A2 + PI1 < 2 * PI1) THEN
      A4 = (A1 - A2 + PI1) / 2
    ELSE
      A4 = (A1 - A2 - PI1) / 2
    END IF
    A3 = A4 + A2
  ELSE !Anticlockwise curve
    IF  (A2 - A1 + PI1 < 2 * PI1) THEN
      A4 = (A2 - A1 + PI1) / 2
    ELSE
      A4 = (A2 - A1 - PI1) / 2
    END IF
    A3 = A4 + A1 + PI1
  END IF

  IF (ADIFF < 0) THEN
    AN = A1 + (PI1 / 2)
  ELSE
    AN = A1 + (3 * PI1 / 2)
  END IF
```

```
  IF (AN > (2 * PI1)) THEN
    AN = AN - (2 * PI1)
  END IF

  DO ARC_ROW = 1, ARC_LEN(N)

    ROW_WTH(ROW) = NG_COLS(N)

    IF (ADIFF < 0) THEN
      A5 = AN - (ARC_INC(N) * (ARC_ROW - 0.5))
    ELSE
      A5 = AN + (ARC_INC(N) * (ARC_ROW - 0.5))
    END IF

    CH = CH + (ARC_INC(N) * NG_WIDTH / 4)
    ROW_CH(ROW) = CH

    IF (A1 > A2) THEN
      ROW_DTN(ROW) = A1 - ((ARC_ROW - 0.5) * ARC_INC(N))
    ELSE
      ROW_DTN(ROW) = A1 + ((ARC_ROW - 0.5) * ARC_INC(N))
    END IF

    DO COL = 1 + ROW_OS, NG_COLS_MAX - ROW_OS


      IF (ADIFF < 0) THEN
        RADIUS = NG_SIZE * (COL - ROW_OS - 0.5)
      ELSE
        !RADIUS is distance from arc origin to centre of
        !curvilinear grid cell
        RADIUS = NG_SIZE * (NG_COLS(N) - COL + ROW_OS + 0.5)
      END IF

      R1 = RADIUS + (NG_SIZE / 2)
      R2 = RADIUS - (NG_SIZE / 2)
      IF (ADIFF < 0) THEN
        CELL_LEN(ROW, COL) = R2 * ARC_INC(N)
      ELSE
        CELL_LEN(ROW, COL) = R1 * ARC_INC(N)
      END IF
      NG_AREA(ROW, COL) = ((R1 ** 2) * ARC_INC(N) / 2) - &
        ((R2 ** 2) * ARC_INC(N) / 2)
      NG_LOC(ROW, COL, 1) = ARC_ORIGIN(N, 1) + (RADIUS * SIN(A5))
      NG_LOC(ROW, COL, 2) = ARC_ORIGIN(N, 2) + (RADIUS * COS(A5))

    END DO

    CH = CH + (ARC_INC(N) * NG_WIDTH / 4)
    ROW = ROW + 1
  END DO

END DO

!Final straight section
ROW_OS = (NG_COLS_MAX - NG_COLS(NR_PTS - 1)) / 2
DO L_ROW = 1, LINE_LEN(NR_PTS - 1)

  ROW_WTH(ROW) = NG_COLS(NR_PTS - 1)
  CH = CH + (LINE_INC(NR_PTS - 1) / 2)
  ROW_CH(ROW) = CH
  ROW_DTN(ROW) = DTN(NR_PTS - 1)
  LINE_X = LINE_START(NR_PTS - 1, 1) + ((L_ROW - 0.5) * &
    LINE_INC(NR_PTS - 1)) * SIN(DTN(NR_PTS - 1))
  LINE_Y = LINE_START(NR_PTS - 1, 2) + ((L_ROW - 0.5) * &
```

```
    LINE_INC(NR_PTS - 1)) * COS(DTN(NR_PTS - 1))

  DO COL = 1 + ROW_OS, NG_COLS_MAX - ROW_OS
    NG_AREA(ROW, COL) = NG_SIZE * LINE_INC(NR_PTS - 1)
    CELL_LEN(ROW, COL) = LINE_INC(NR_PTS - 1)
    NG_LOC(ROW, COL, 1) = LINE_X + (((COL - ROW_OS) - &
        (NG_COLS(NR_PTS - 1) / 2.0) - 0.5) * NG_SIZE * COS(DTN(N)))
    NG_LOC(ROW, COL, 2) = LINE_Y - (((COL - ROW_OS) - &
        (NG_COLS(NR_PTS - 1) / 2.0) - 0.5) * NG_SIZE * SIN(DTN(N)))
  END DO

  CH = CH + (LINE_INC(NR_PTS - 1) / 2)
  ROW = ROW + 1
END DO

!Determine source grid cells and weightings required for calculting
!CLG bed levels
DO ROW = 1, NG_ROWS
  ROW_OS = (NG_COLS_MAX - ROW_WTH(ROW)) / 2
  DO COL = 1 + ROW_OS, NG_COLS_MAX - ROW_OS
    IF ((NG_LOC(ROW, COL, 1) >= 0) .AND. (NG_LOC(ROW, COL, 1) <= &
        GRID_WIDTH) .AND. (NG_LOC(ROW, COL, 2) >= 0) .AND. &
        (NG_LOC(ROW, COL, 2) <= GRID_LENGTH)) THEN

      RW = INT((NG_LOC(ROW, COL, 2) / GRID_SIZE) + 0.5)
      CL = INT((NG_LOC(ROW, COL, 1) / GRID_SIZE) + 0.5)
      X = NG_LOC(ROW, COL, 1) - ((CL - 0.5) * GRID_SIZE)
      Y = NG_LOC(ROW, COL, 2) - ((RW - 0.5) * GRID_SIZE)

      IF (CL == 0) THEN
        CL = 1
        X = X - GRID_SIZE
      ELSE
        IF (CL == COLS) THEN
          CL = COLS - 1
          X = X + GRID_SIZE
        END IF
      END IF

      IF (RW == 0) THEN
        RW = 1
        Y = Y - GRID_SIZE
      ELSE
        IF (RW == ROWS) THEN
          RW = ROWS - 1
          Y = Y + GRID_SIZE
        END IF
      END IF


      IF (Y >= X) THEN
        IF (ABS(GRID_SIZE - ABS(X)) < 0.00001) THEN
          !Avoid potential div by zero error when X = GRID_SIZE
          !(this should never happen anyway)
          WT_XCOORDS (ROW, COL, 1) = RW
          WT_YCOORDS (ROW, COL, 1) = CL
          BLWT(ROW, COL, 1) = 0
          WT_XCOORDS(ROW, COL, 2) = RW + 1
          WT_YCOORDS(ROW, COL, 2) = CL
          BLWT(ROW, COL, 2) = 0
          WT_XCOORDS(ROW, COL, 3) = RW + 1
          WT_YCOORDS(ROW, COL, 3) = CL + 1
          BLWT(ROW, COL, 3) = 1
        ELSE
          A = 1 - (Y / GRID_SIZE)
```

```
          B = (Y - X) / GRID_SIZE
          C = X / GRID_SIZE
          WT_XCOORDS (ROW, COL, 1) = RW
          WT_YCOORDS (ROW, COL, 1) = CL
          BLWT(ROW, COL, 1) = A
          WT_XCOORDS(ROW, COL, 2) = RW + 1
          WT_YCOORDS(ROW, COL, 2) = CL
          BLWT(ROW, COL, 2) = B
          WT_XCOORDS(ROW, COL, 3) = RW + 1
          WT_YCOORDS(ROW, COL, 3) = CL + 1
          BLWT(ROW, COL, 3) = C
        END IF
      ELSE
        IF (ABS(X) < 0.00001) THEN
          !Avoid potential div by zero error when X = 0
          !(this might happen if location is exact multiple of grid size)
          WT_XCOORDS (ROW, COL, 1) = RW
          WT_YCOORDS (ROW, COL, 1) = CL
          BLWT(ROW, COL, 1) = 1
          WT_XCOORDS(ROW, COL, 2) = RW
          WT_YCOORDS(ROW, COL, 2) = CL + 1
          BLWT(ROW, COL, 2) = 0
          WT_XCOORDS(ROW, COL, 3) = RW + 1
          WT_YCOORDS(ROW, COL, 3) = CL + 1
          BLWT(ROW, COL, 3) = 0
        ELSE
          A = 1 - (X / GRID_SIZE)
          B = ((X - Y) / GRID_SIZE)
          C = Y / GRID_SIZE
          WT_XCOORDS (ROW, COL, 1) = RW
          WT_YCOORDS (ROW, COL, 1) = CL
          BLWT(ROW, COL, 1) = A
          WT_XCOORDS(ROW, COL, 2) = RW
          WT_YCOORDS(ROW, COL, 2) = CL + 1
          BLWT(ROW, COL, 2) = B
          WT_XCOORDS(ROW, COL, 3) = RW + 1
          WT_YCOORDS(ROW, COL, 3) = CL + 1
          BLWT(ROW, COL, 3) = C
        END IF
      END IF
    ELSE
      WT_XCOORDS (ROW, COL, 1) = -9999
      WT_YCOORDS (ROW, COL, 1) = -9999
      BLWT(ROW, COL, 1) = -9999
      WT_XCOORDS(ROW, COL, 2) = -9999
      WT_YCOORDS(ROW, COL, 2) = -9999
      BLWT(ROW, COL, 2) = -9999
      WT_XCOORDS(ROW, COL, 3) = -9999
      WT_YCOORDS(ROW, COL, 3) = -9999
      BLWT(ROW, COL, 3) = -9999
    END IF
  END DO
END DO

END SUBROUTINE

!-------------------------------------------------------------------------

SUBROUTINE GET_CLG_WTGS (LINE_START, DTN, GRID_SIZE, NG_SIZE, NG_LOC, &
    LINE_INC, ARC_INC, LINE_LEN, ARC_LEN, ARC_ORIGIN, NR_PTS, ROWS, &
    COLS, NG_ROWS, NG_COLS, ROW_WTH, XWT, YWT, CLG_WT)
!Subroutine to calculate weightings needed to interpolate water level
!results back to the CA grid cells

REAL,   DIMENSION(:,:),     INTENT(IN)  :: LINE_START
```

```
REAL,    DIMENSION(:),        INTENT(IN)  :: DTN
REAL,    DIMENSION(:),        INTENT(IN)  :: LINE_INC
REAL,    DIMENSION(:),        INTENT(IN)  :: ARC_INC
REAL,    DIMENSION(:,:,:),    INTENT(OUT) :: CLG_WT
REAL,    DIMENSION(:,:,:),    INTENT(IN)  :: NG_LOC
REAL,    DIMENSION(:,:),      INTENT(IN)  :: ARC_ORIGIN

INTEGER,     DIMENSION(:),        INTENT(IN)  :: LINE_LEN
INTEGER,     DIMENSION(:),        INTENT(IN)  :: ARC_LEN
INTEGER,     DIMENSION(:),        INTENT(IN)  :: NG_COLS
INTEGER,     DIMENSION(:),        INTENT(IN)  :: ROW_WTH
INTEGER,     DIMENSION(:,:,:),    INTENT(OUT) :: XWT
INTEGER,     DIMENSION(:,:,:),    INTENT(OUT) :: YWT

REAL,    INTENT(IN) :: NG_SIZE
REAL,    INTENT(IN) :: GRID_SIZE

INTEGER,     INTENT(IN) :: NR_PTS
INTEGER,     INTENT(IN) :: ROWS
INTEGER,     INTENT(IN) :: COLS
INTEGER,     INTENT(IN) :: NG_ROWS

REAL X, Y
REAL RX1, RY1
REAL RX2, RY2
REAL D1, D2
REAL A1, A2, A3, A4, A5, A6, A7
REAL L, W
REAL NG_ROW, NG_COL
REAL PX, PY
REAL DX, DY
REAL X1, X2, X3
REAL Y1, Y2, Y3
REAL K1, K2, K3, K4, K5
REAL K6, K7, K8, K9, K10
REAL K11, K12, K13

INTEGER ROW, COL
INTEGER NG_COLS_MAX
INTEGER FCL, LCL
INTEGER N, M, F
INTEGER RW, CL
INTEGER P2R, P2C

NG_COLS_MAX = 0
DO N = 1, NR_PTS - 1
  IF (NG_COLS(N) > NG_COLS_MAX) THEN
    NG_COLS_MAX = NG_COLS(N)
  END IF
END DO

DO ROW = 1, ROWS
  DO COL = 1, COLS
    F = 0
    N = 0
    X = (COL - 0.5) * GRID_SIZE
    Y = (ROW - 0.5) * GRID_SIZE
    !Determine which line or arc section contains the current point
    DO WHILE ((F == 0) .AND. (N < NR_PTS - 1))
      NG_ROW = 0
      N = N + 1
      RX1 = X - LINE_START(N, 1)
      RY1 = Y - LINE_START(N, 2)
      A1 = DTN(N)
      IF (RY1 > 1E-15) THEN
```

```
  A2 = ATAN(RX1 / RY1)
ELSE
  IF (RY1 < -1E-15) THEN
      A2 = ATAN(RX1 / RY1) + PI1
  ELSE
      IF (RX1 > 0) THEN
          A2 = 0.5 * PI1
      ELSE
          A2 = -0.5 * PI1
      END IF
  END IF
END IF
D1 = ((RX1**2) + (RY1**2))**0.5
L = D1 * COS(A2 - A1)
W = D1 * SIN(A2 - A1)
!L is distance along the current line from 'LINE_START'
!W is the perpendicular distance from the centreline
!If L < 0 then the point might fall in the previous arc section
IF (L < 0) THEN
  !If n = 1 then there is no previous arc
  IF (N > 1) THEN
    RX2 = X - ARC_ORIGIN(N - 1, 1)
    RY2 = Y - ARC_ORIGIN(N - 1, 2)
    D2 = ((RX2**2) + (RY2**2))**0.5
    A4 = DTN(N) - DTN(N - 1)      !Bend angle

    IF (ABS(RY2 > 1E-15)) THEN
      IF (RX2 >= 0) THEN
          !Angle of current point about arc origin
          A5 = ATAN(RX2 / RY2)
      ELSE
          A5 = ATAN(RX2 / RY2) +  (2 * PI1)
      END IF
    ELSE
          IF (RY2 < -1E-15) THEN

              A5 = ATAN(RX2 / RY2) + PI1

          ELSE
              IF (RX2 >= 0) THEN
                  A5 = 0.5 * PI1
              ELSE
                  A5 = 1.5 * PI1
              END IF
      END IF
    END IF

    IF (A4 > PI1) THEN
      A4 = A4 - (2 * PI1)
    ELSE
      IF (A4 < - PI1) THEN
        A4 = A4 + (2 * PI1)
      END IF
    END IF

    IF (A4 > 0) THEN !Right hand bend
      !Angle of start of arc about arc origin
      A6 = DTN(N - 1) - (PI1 / 2)
      IF (A6 < 0) THEN
        A6 = A6 + (2 * PI1)
      END IF
      A7 = A5 - A6    !Angle of point relative to start of arc
      IF (A7 < 0) THEN
        A7 = A7 + (2 * PI1)
      END IF
```

```
          IF (((ARC_INC(N - 1) * ARC_LEN(N - 1)) - A7 > -1E-3) .AND. &
              (D2 < NG_COLS(N - 1) * NG_SIZE)) THEN
              !Point is in arc section

            F = 1
            NG_ROW = A7 / ARC_INC(N - 1)
            NG_COL = NG_COLS(N - 1) - (D2 / NG_SIZE)
          END IF
        ELSE              !Left hand bend
          A5 = (2 * PI1) - A5
          !Angle of start of arc about arc origin
          A6 = (3 * PI1 / 2) - DTN(N - 1)
          IF (A6 < 0) THEN
            A6 = A6 + (2 * PI1)
          END IF
          A7 = A5 - A6    !Angle of point relative to start of arc
          IF (A7 < 0) THEN
            A7 = A7 + (2 * PI1)
          END IF
          IF (((ARC_INC(N - 1) * ARC_LEN(N - 1)) - A7 > -1E-3) .AND. &
              (D2 < NG_COLS(N - 1) * NG_SIZE)) THEN
              !Point is in arc section
            F = 1
            NG_ROW = A7 / ARC_INC(N - 1)
            NG_COL = D2 / NG_SIZE
          END IF
        END IF

        IF (F ==1) THEN
          !Add up the cells in all complete line and arc sections upto
          !the current arc, for axial distance in cells
          DO M = 1, N - 1
            NG_ROW = NG_ROW + LINE_LEN(M)
          END DO
          IF (N > 2) THEN
            DO M = 1, N - 2
              NG_ROW = NG_ROW + ARC_LEN(M)
            END DO
          END IF
        END IF
      END IF

    ELSE  !Check to see if point falls within straight line section
      IF ((L < LINE_INC(N) * LINE_LEN(N)) .AND. &
          (ABS(W) < ROW_WTH(ROW) * NG_SIZE / 2.0)) THEN
        F = 1
        IF (N > 1) THEN
          DO M = 1, N - 1
            NG_ROW = NG_ROW + LINE_LEN(M) + ARC_LEN(M)
          END DO
        END IF
        NG_ROW = NG_ROW + (L / LINE_INC(N))
        NG_COL = (W / NG_SIZE) + (NG_COLS_MAX / 2.0)
      END IF
    END IF

  END DO

  IF (F == 0) THEN
    NG_ROW = -2
    NG_COL = -2
  END IF

  RW = INT(NG_ROW + 0.5)
```

```
IF (RW >= 0) THEN
  IF (RW == 0) THEN
    RW = 1
  ELSE
    IF (RW >= NG_ROWS) THEN
      RW = NG_ROWS - 1
    END IF
  END IF

  FCL = ((NG_COLS_MAX - ROW_WTH(RW)) / 2) + 1
  LCL = (NG_COLS_MAX + ROW_WTH(RW)) / 2
ELSE
  FCL = 1
  LCL = NG_COLS_MAX
END IF

CL = INT(NG_COL + 0.5)
IF (CL >= 0) THEN
  IF (CL < FCL) THEN
    CL = FCL
  ELSE
    IF (CL >= LCL) THEN
      CL = LCL - 1
    END IF
  END IF
END IF


IF ((RW > 0) .AND. (CL > 0)) THEN
  PX = (COL - 0.5) * GRID_SIZE
  PY = (ROW - 0.5) * GRID_SIZE
  X1 = NG_LOC(RW, CL, 1)
  Y1 = NG_LOC(RW, CL, 2)
  X3 = NG_LOC(RW + 1, CL + 1, 1)
  Y3 = NG_LOC(RW + 1, CL + 1, 2)
  DX = X3 - X1
  DY = Y3 - Y1
  IF (DY < 0) THEN
    A1 = ATAN(DX / DY) + PI1
  ELSE
    A1 = ATAN(DX / DY)
  END IF
  DX = PX - X1
  DY = PY - Y1
  IF (DY < 0) THEN
    A2 = ATAN(DX / DY) + PI1
  ELSE
    A2 = ATAN(DX / DY)
  END IF
  IF (A2 > A1) THEN
    P2R = RW
    P2C = CL + 1
  ELSE
    P2R = RW + 1
    P2C = CL
  END IF

  X2 = NG_LOC(P2R, P2C, 1)
  Y2 = NG_LOC(P2R, P2C, 2)

  K1 = Y3 - Y2
  K2 = Y1 - Y3
  K3 = Y2 - Y1
  K4 = X2 - X3
```

```
      K5 = X3 - X1
      K6 = X1 - X2
      K7 = (X1 * (Y2 - Y3)) + (X2 * (Y3 - Y1)) + (X3 * (Y1 - Y2))
      K8 = (X3 * Y2) - (X2 * Y3)
      K9 = (X1 * Y3) - (X3 * Y1)
      K10 = (X2 * Y1) - (X1 * Y2)
      K11 = - ((K1 * PX) + (K4 * PY) + K8) / K7
      K12 = - ((K2 * PX) + (K5 * PY) + K9) / K7
      K13 = - ((K3 * PX) + (K6 * PY) + K10) / K7
      XWT(ROW, COL, 1) = CL
      XWT(ROW, COL, 2) = P2C
      XWT(ROW, COL, 3) = CL + 1
      YWT(ROW, COL, 1) = RW
      YWT(ROW, COL, 2) = P2R
      YWT(ROW, COL, 3) = RW + 1
      CLG_WT(ROW, COL, 1) = K11
      CLG_WT(ROW, COL, 2) = K12
      CLG_WT(ROW, COL, 3) = 1 - (K11 + K12)
    ELSE
      XWT(ROW, COL, 1) = -1
      XWT(ROW, COL, 2) = -1
      XWT(ROW, COL, 3) = -1
      YWT(ROW, COL, 1) = -1
      YWT(ROW, COL, 2) = -1
      YWT(ROW, COL, 3) = -1
      CLG_WT(ROW, COL, 1) = 0
      CLG_WT(ROW, COL, 2) = 0
      CLG_WT(ROW, COL, 3) = 0
    END IF
  END DO
END DO

END SUBROUTINE

END MODULE
```

Appendix B

EXAMPLE INPUT FILES

Example Input Files

Example Input files for the baseline simulation described in Chapter 5. The sediment file is for the multi-fraction simulation in Chapter 5.

File	Purpose	Page nr.
Baseline.crl	Contains general model settings and links to other input files and output locations	431
InitBed.asc	ESRI grid file containing intial bathymetry	434
MinBed.asc	ESRI ASCII grid file containing minimum bed levels	439
InitSed.asc	ESRI ASCII grid file containing initial sediment types	444
InitSed.sed	Contains fraction sizes and initial sediment types for multi-fraction simulations	449

Baseline.crl

```
0050.0000                                                                    #Run time in years
00020                                                                        #Number of timesteps per tidal cycle
"C:\Users\iwben_000\Documents\Model\Test_Dec14\Baseline1\InitBed1.asc"       #Initial bed levels file
"C:\Users\iwben_000\Documents\Model\Test_Dec14\Baseline1\MinBed1.asc"        #Minimum bed levels file
False                                                                        #True if ISIS model to be used to calculate water levels
12.50                                                                        #ISIS model warmup time
1                                                                            #ISIS model cross section spacing
False                                                                        #True if interpolates to be added between ISIS cross sections
""                                                                           #ISIS DAT file
""                                                                           #ISIS Steady State DAT file
False                                                                        #True for simple tide, false for spring/neap cycle
25.00 04.00 02.00                                                            #Spring High tide level, spring and neap tidal ranges
True                                                                         #True if wave model to be used
000.00 05.00 05.00 00010000                                                  #Prevailing wind direction/speed/standard dev/bdy fetch
True                                                                         #True if single fraction model to be used
00100 00150 00200                                                            #D10, D50, D90
""                                                                           #Sediment data file
"C:\Users\iwben_000\Documents\Model\Test_Dec14\Baseline1\InitSed1.asc"       #Initial sediment types file
0.00005 01                                                                   #Downstream boundary type, time averaged input (cu.m/s) and type
0.000100                                                                     #Downstream sediment volumetric concentration and type
False                                                                        #True if salt marsh model to be used
""                                                                           #Salt marsh initial conditions file
2000.0 0.000 0.000 0.000 0.010                                               #Max biomass, min submergence (Smin), SM2, SM1, Smax
01.00 0.0001                                                                 #xp, s_min (slope based routing parameters)
True                                                                         #True if bed level results to be produced
00010000.00                                                                  #Bed level results interval (hours)
True                                                                         #True if sediment results to be produced
00010000.00                                                                  #Sediment results interval (hours)
False                                                                        #True if velocity results to be produced
00010000.00 00000001 00000020                                                #Output frq(hrs), 2nd int.(timesteps) & nr of results sets
False                                                                        #True if velocity time-series to be produced
00000000 00000000                                                            #Time-series output row and column numbers
"C:\Users\iwben_000\Documents\Model\Test_Dec14\Sensitivity_Jan15\Baseline1"   #Results directory
060                                 #Nr. Of inputs in list (below)
0000000.000000                      #Not used
0000000.000000                      #Not used
```

```
0002650.000000                  #Particle density (sand)
0000000.400000                  #Porosity
0000050.000000                  #Global roughness value (Chezy)
0000000.050000                  #Min depth (smaller depths treated as zero) (m)
0000002.000000                  #Maximum velocity for sediment transport calculations
0000000.200000                  #Maximum bed level change in one time step (single fraction version)
0000001.000000                  #Scaling factor for lateral sediment transport
0000000.000000                  #Not used
0000000.002000                  #Organogenic sediment production from saltmarsh (m/yr)
0000000.120000                  #Shear stress for erosion of mud (N/m2)
0000000.060000                  #Shear stress for deposition of mud (N/m2)
0001200.000000                  #Mud bulk density
0000000.000000                  #Single fraction min level for sublayers
0000050.000000                  #Single fraction max level for sublayers
0000020.000000                  #Single fraction percentage mud for sediment type 1
0000100.000000                  #Single fraction percentage mud for sediment type 2
0000050.000000                  #Single fraction percentage mud for sediment type 3
0000010.000000                  #Single fraction percentage mud for sediment type 4
0000090.000000                  #Single fraction percentage mud for sediment type 5
0000030.000000                  #Single fraction percentage mud for sediment type 6
0000000.200000                  #Active layer thickness for single fraction + mud model
0000000.200000                  #Sub-layer thickness for single fraction + mud model
0000010.000000                  #Minimum Chezy C value
0000050.000000                  #Maximum Chezy C value
0000500.000000                  #Diffusion coefficient
0000000.000000                  #Minimum sand input at downstream boundary as percentage of calculated transport at boundary
0100000.000000                  #Maximum sand input at downstream boundary as percentage of calculated transport at boundary
0000000.000000                  #Minimum sand input at upstream boundary as percentage of calculated transport at boundary
0000101.000000                  #Maximum sand input at upstream boundary as percentage of calculated transport at boundary
0000002.000000                  #Shear stress factor at 20% mud
0000000.000000                  #1 if grid interpolation to be used in water level calcs (otherwise 0)
0000000.000000                  #Number of centreline definition points (for grid interpolation)
0000000.000000                  #Centreline point 1: x coordinate
0000000.000000                  #Centreline point 1: y coordinate
0000000.000000                  #Centreline point 2: x coordinate
0000000.000000                  #Centreline point 2: y coordinate
0000000.000000                  #Centreline point 3: x coordinate
0000000.000000                  #Centreline point 3: y coordinate
0000000.000000                  #Centreline point 3: x coordinate
```

```
0000000.000000                  #Centreline point 3: y coordinate
0000000.000000                  #Centreline point 3: x coordinate
0000000.000000                  #Centreline point 3: y coordinate
0000000.000000                  #Centreline width section 1
0000000.000000                  #Centreline width section 2
0000000.000000                  #Centreline width section 3
0000000.000000                  #Centreline width section 4
0000001.000000                  #Number of inflow points
0000001.000000                  #Row number for first inflow point
0000011.000000                  #Column number for first inflow point
0000010.000000                  #Inflow at inflow point 1
0000001.000000                  #1 if wave height results to be generated with bathymetry output
0000000.010000                  #Maximum slope used to calculate depths in slope based routing model
0010000.000000                  #Maximum number of iterations for flow calculation
0000000.150000                  #Max biomass point 1
0000000.010000                  #Max biomass point 2
0000000.000000                  #1 if percentage mud in active layer to be loaded from file instead of generic sediment type, otherwise zero
0000000.000000                  #Lateral reflection of waves (proportion of wave energy reflected)
0000000.000000                  #Sea level rise in mm/yr
```

Initial Bed Levels – ESRI ASCII Grid Format

```
ncols         21
nrows         250
xllcorner     0
yllcorner     0
cellsize      50
nodata_value  -9999
  25.500  24.750  24.000  23.250  22.500  21.750  21.000  20.250  19.500  18.750  18.000  18.750  19.500  20.250  21.000  21.750  22.500  23.250  24.000  24.750  25.500
  25.600  24.780  24.030  23.280  22.530  21.780  21.030  20.280  19.530  18.780  18.030  18.780  19.530  20.280  21.030  21.780  22.530  23.280  24.030  24.780  25.600
  25.700  24.810  24.060  23.310  22.560  21.810  21.060  20.310  19.560  18.810  18.060  18.810  19.560  20.310  21.060  21.810  22.560  23.310  24.060  24.810  25.700
  25.800  24.840  24.090  23.340  22.590  21.840  21.090  20.340  19.590  18.840  18.090  18.840  19.590  20.340  21.090  21.840  22.590  23.340  24.090  24.840  25.800
  25.900  24.870  24.120  23.370  22.620  21.870  21.120  20.370  19.620  18.870  18.120  18.870  19.620  20.370  21.120  21.870  22.620  23.370  24.120  24.870  25.900
  26.000  24.900  24.150  23.400  22.650  21.900  21.150  20.400  19.650  18.900  18.150  18.900  19.650  20.400  21.150  21.900  22.650  23.400  24.150  24.900  26.000
  26.000  24.930  24.180  23.430  22.680  21.930  21.180  20.430  19.680  18.930  18.180  18.930  19.680  20.430  21.180  21.930  22.680  23.430  24.180  24.930  26.000
  26.000  24.960  24.210  23.460  22.710  21.960  21.210  20.460  19.710  18.960  18.210  18.960  19.710  20.460  21.210  21.960  22.710  23.460  24.210  24.960  26.000
  26.000  24.990  24.240  23.490  22.740  21.990  21.240  20.490  19.740  18.990  18.240  18.990  19.740  20.490  21.240  21.990  22.740  23.490  24.240  24.990  26.000
  26.000  25.020  24.270  23.520  22.770  22.020  21.270  20.520  19.770  19.020  18.270  19.020  19.770  20.520  21.270  22.020  22.770  23.520  24.270  25.020  26.000
  26.000  25.050  24.300  23.550  22.800  22.050  21.300  20.550  19.800  19.050  18.300  19.050  19.800  20.550  21.300  22.050  22.800  23.550  24.300  25.050  26.000
  26.000  25.080  24.330  23.580  22.830  22.080  21.330  20.580  19.830  19.080  18.330  19.080  19.830  20.580  21.330  22.080  22.830  23.580  24.330  25.080  26.000
  26.000  25.110  24.360  23.610  22.860  22.110  21.360  20.610  19.860  19.110  18.360  19.110  19.860  20.610  21.360  22.110  22.860  23.610  24.360  25.110  26.000
  26.000  25.140  24.390  23.640  22.890  22.140  21.390  20.640  19.890  19.140  18.390  19.140  19.890  20.640  21.390  22.140  22.890  23.640  24.390  25.140  26.000
  26.000  25.170  24.420  23.670  22.920  22.170  21.420  20.670  19.920  19.170  18.420  19.170  19.920  20.670  21.420  22.170  22.920  23.670  24.420  25.170  26.000
  26.000  25.200  24.450  23.700  22.950  22.200  21.450  20.700  19.950  19.200  18.450  19.200  19.950  20.700  21.450  22.200  22.950  23.700  24.450  25.200  26.000
  26.000  25.230  24.480  23.730  22.980  22.230  21.480  20.730  19.980  19.230  18.480  19.230  19.980  20.730  21.480  22.230  22.980  23.730  24.480  25.230  26.000
  26.000  25.260  24.510  23.760  23.010  22.260  21.510  20.760  20.010  19.260  18.510  19.260  20.010  20.760  21.510  22.260  23.010  23.760  24.510  25.260  26.000
  26.000  25.290  24.540  23.790  23.040  22.290  21.540  20.790  20.040  19.290  18.540  19.290  20.040  20.790  21.540  22.290  23.040  23.790  24.540  25.290  26.000
  26.000  25.320  24.570  23.820  23.070  22.320  21.570  20.820  20.070  19.320  18.570  19.320  20.070  20.820  21.570  22.320  23.070  23.820  24.570  25.320  26.000
  26.000  25.350  24.600  23.850  23.100  22.350  21.600  20.850  20.100  19.350  18.600  19.350  20.100  20.850  21.600  22.350  23.100  23.850  24.600  25.350  26.000
  26.000  25.380  24.630  23.880  23.130  22.380  21.630  20.880  20.130  19.380  18.630  19.380  20.130  20.880  21.630  22.380  23.130  23.880  24.630  25.380  26.000
  26.000  25.410  24.660  23.910  23.160  22.410  21.660  20.910  20.160  19.410  18.660  19.410  20.160  20.910  21.660  22.410  23.160  23.910  24.660  25.410  26.000
  26.000  25.440  24.690  23.940  23.190  22.440  21.690  20.940  20.190  19.440  18.690  19.440  20.190  20.940  21.690  22.440  23.190  23.940  24.690  25.440  26.000
  26.000  25.470  24.720  23.970  23.220  22.470  21.720  20.970  20.220  19.470  18.720  19.470  20.220  20.970  21.720  22.470  23.220  23.970  24.720  25.470  26.000
  26.000  25.500  24.750  24.000  23.250  22.500  21.750  21.000  20.250  19.500  18.750  19.500  20.250  21.000  21.750  22.500  23.250  24.000  24.750  25.500  26.000
  26.000  25.600  24.780  24.030  23.280  22.530  21.780  21.030  20.280  19.530  18.780  19.530  20.280  21.030  21.780  22.530  23.280  24.030  24.780  25.600  26.000
  26.000  25.700  24.810  24.060  23.310  22.560  21.810  21.060  20.310  19.560  18.810  19.560  20.310  21.060  21.810  22.560  23.310  24.060  24.810  25.700  26.000
  26.000  25.800  24.840  24.090  23.340  22.590  21.840  21.090  20.340  19.590  18.840  19.590  20.340  21.090  21.840  22.590  23.340  24.090  24.840  25.800  26.000
  26.000  25.900  24.870  24.120  23.370  22.620  21.870  21.120  20.370  19.620  18.870  19.620  20.370  21.120  21.870  22.620  23.370  24.120  24.870  25.900  26.000
  26.000  26.000  24.900  24.150  23.400  22.650  21.900  21.150  20.400  19.650  18.900  19.650  20.400  21.150  21.900  22.650  23.400  24.150  24.900  26.000  26.000
  26.000  26.000  24.930  24.180  23.430  22.680  21.930  21.180  20.430  19.680  18.930  19.680  20.430  21.180  21.930  22.680  23.430  24.180  24.930  26.000  26.000
  26.000  26.000  24.960  24.210  23.460  22.710  21.960  21.210  20.460  19.710  18.960  19.710  20.460  21.210  21.960  22.710  23.460  24.210  24.960  26.000  26.000
  26.000  26.000  24.990  24.240  23.490  22.740  21.990  21.240  20.490  19.740  18.990  19.740  20.490  21.240  21.990  22.740  23.490  24.240  24.990  26.000  26.000
  26.000  26.000  25.020  24.270  23.520  22.770  22.020  21.270  20.520  19.770  19.020  19.770  20.520  21.270  22.020  22.770  23.520  24.270  25.020  26.000  26.000
  26.000  26.000  25.050  24.300  23.550  22.800  22.050  21.300  20.550  19.800  19.050  19.800  20.550  21.300  22.050  22.800  23.550  24.300  25.050  26.000  26.000
  26.000  26.000  25.080  24.330  23.580  22.830  22.080  21.330  20.580  19.830  19.080  19.830  20.580  21.330  22.080  22.830  23.580  24.330  25.080  26.000  26.000
  26.000  26.000  25.110  24.360  23.610  22.860  22.110  21.360  20.610  19.860  19.110  19.860  20.610  21.360  22.110  22.860  23.610  24.360  25.110  26.000  26.000
  26.000  26.000  25.140  24.390  23.640  22.890  22.140  21.390  20.640  19.890  19.140  19.890  20.640  21.390  22.140  22.890  23.640  24.390  25.140  26.000  26.000
  26.000  26.000  25.170  24.420  23.670  22.920  22.170  21.420  20.670  19.920  19.170  19.920  20.670  21.420  22.170  22.920  23.670  24.420  25.170  26.000  26.000
  26.000  26.000  25.200  24.450  23.700  22.950  22.200  21.450  20.700  19.950  19.200  19.950  20.700  21.450  22.200  22.950  23.700  24.450  25.200  26.000  26.000
  26.000  26.000  25.230  24.480  23.730  22.980  22.230  21.480  20.730  19.980  19.230  19.980  20.730  21.480  22.230  22.980  23.730  24.480  25.230  26.000  26.000
  26.000  26.000  25.260  24.510  23.760  23.010  22.260  21.510  20.760  20.010  19.260  20.010  20.760  21.510  22.260  23.010  23.760  24.510  25.260  26.000  26.000
  26.000  26.000  25.290  24.540  23.790  23.040  22.290  21.540  20.790  20.040  19.290  20.040  20.790  21.540  22.290  23.040  23.790  24.540  25.290  26.000  26.000
```

```
26.000  26.000  25.320  24.570  23.820  23.070  22.320  21.570  20.820  20.070  19.320  20.070  20.820  21.570  22.320  23.070  23.820  24.570  25.320  26.000  26.000
26.000  26.000  25.350  24.600  23.850  23.100  22.350  21.600  20.850  20.100  19.350  20.100  20.850  21.600  22.350  23.100  23.850  24.600  25.350  26.000  26.000
26.000  26.000  25.380  24.630  23.880  23.130  22.380  21.630  20.880  20.130  19.380  20.130  20.880  21.630  22.380  23.130  23.880  24.630  25.380  26.000  26.000
26.000  26.000  25.410  24.660  23.910  23.160  22.410  21.660  20.910  20.160  19.410  20.160  20.910  21.660  22.410  23.160  23.910  24.660  25.410  26.000  26.000
26.000  26.000  25.440  24.690  23.940  23.190  22.440  21.690  20.940  20.190  19.440  20.190  20.940  21.690  22.440  23.190  23.940  24.690  25.440  26.000  26.000
26.000  26.000  25.470  24.720  23.970  23.220  22.470  21.720  20.970  20.220  19.470  20.220  20.970  21.720  22.470  23.220  23.970  24.720  25.470  26.000  26.000
26.000  26.000  25.500  24.750  24.000  23.250  22.500  21.750  21.000  20.250  19.500  20.250  21.000  21.750  22.500  23.250  24.000  24.750  25.500  26.000  26.000
26.000  26.000  25.600  24.780  24.030  23.280  22.530  21.780  21.030  20.280  19.530  20.280  21.030  21.780  22.530  23.280  24.030  24.780  25.600  26.000  26.000
26.000  26.000  25.700  24.810  24.060  23.310  22.560  21.810  21.060  20.310  19.560  20.310  21.060  21.810  22.560  23.310  24.060  24.810  25.700  26.000  26.000
26.000  26.000  25.800  24.840  24.090  23.340  22.590  21.840  21.090  20.340  19.590  20.340  21.090  21.840  22.590  23.340  24.090  24.840  25.800  26.000  26.000
26.000  26.000  25.900  24.870  24.120  23.370  22.620  21.870  21.120  20.370  19.620  20.370  21.120  21.870  22.620  23.370  24.120  24.870  25.900  26.000  26.000
26.000  26.000  26.000  24.900  24.150  23.400  22.650  21.900  21.150  20.400  19.650  20.400  21.150  21.900  22.650  23.400  24.150  24.900  26.000  26.000  26.000
26.000  26.000  26.000  24.930  24.180  23.430  22.680  21.930  21.180  20.430  19.680  20.430  21.180  21.930  22.680  23.430  24.180  24.930  26.000  26.000  26.000
26.000  26.000  26.000  24.960  24.210  23.460  22.710  21.960  21.210  20.460  19.710  20.460  21.210  21.960  22.710  23.460  24.210  24.960  26.000  26.000  26.000
26.000  26.000  26.000  24.990  24.240  23.490  22.740  21.990  21.240  20.490  19.740  20.490  21.240  21.990  22.740  23.490  24.240  24.990  26.000  26.000  26.000
26.000  26.000  26.000  25.020  24.270  23.520  22.770  22.020  21.270  20.520  19.770  20.520  21.270  22.020  22.770  23.520  24.270  25.020  26.000  26.000  26.000
26.000  26.000  26.000  25.050  24.300  23.550  22.800  22.050  21.300  20.550  19.800  20.550  21.300  22.050  22.800  23.550  24.300  25.050  26.000  26.000  26.000
26.000  26.000  26.000  25.080  24.330  23.580  22.830  22.080  21.330  20.580  19.830  20.580  21.330  22.080  22.830  23.580  24.330  25.080  26.000  26.000  26.000
26.000  26.000  26.000  25.110  24.360  23.610  22.860  22.110  21.360  20.610  19.860  20.610  21.360  22.110  22.860  23.610  24.360  25.110  26.000  26.000  26.000
26.000  26.000  26.000  25.140  24.390  23.640  22.890  22.140  21.390  20.640  19.890  20.640  21.390  22.140  22.890  23.640  24.390  25.140  26.000  26.000  26.000
26.000  26.000  26.000  25.170  24.420  23.670  22.920  22.170  21.420  20.670  19.920  20.670  21.420  22.170  22.920  23.670  24.420  25.170  26.000  26.000  26.000
26.000  26.000  26.000  25.200  24.450  23.700  22.950  22.200  21.450  20.700  19.950  20.700  21.450  22.200  22.950  23.700  24.450  25.200  26.000  26.000  26.000
26.000  26.000  26.000  25.230  24.480  23.730  22.980  22.230  21.480  20.730  19.980  20.730  21.480  22.230  22.980  23.730  24.480  25.230  26.000  26.000  26.000
26.000  26.000  26.000  25.260  24.510  23.760  23.010  22.260  21.510  20.760  20.010  20.760  21.510  22.260  23.010  23.760  24.510  25.260  26.000  26.000  26.000
26.000  26.000  26.000  25.290  24.540  23.790  23.040  22.290  21.540  20.790  20.040  20.790  21.540  22.290  23.040  23.790  24.540  25.290  26.000  26.000  26.000
26.000  26.000  26.000  25.320  24.570  23.820  23.070  22.320  21.570  20.820  20.070  20.820  21.570  22.320  23.070  23.820  24.570  25.320  26.000  26.000  26.000
26.000  26.000  26.000  25.350  24.600  23.850  23.100  22.350  21.600  20.850  20.100  20.850  21.600  22.350  23.100  23.850  24.600  25.350  26.000  26.000  26.000
26.000  26.000  26.000  25.380  24.630  23.880  23.130  22.380  21.630  20.880  20.130  20.880  21.630  22.380  23.130  23.880  24.630  25.380  26.000  26.000  26.000
26.000  26.000  26.000  25.410  24.660  23.910  23.160  22.410  21.660  20.910  20.160  20.910  21.660  22.410  23.160  23.910  24.660  25.410  26.000  26.000  26.000
26.000  26.000  26.000  25.440  24.690  23.940  23.190  22.440  21.690  20.940  20.190  20.940  21.690  22.440  23.190  23.940  24.690  25.440  26.000  26.000  26.000
26.000  26.000  26.000  25.470  24.720  23.970  23.220  22.470  21.720  20.970  20.220  20.970  21.720  22.470  23.220  23.970  24.720  25.470  26.000  26.000  26.000
26.000  26.000  26.000  25.500  24.750  24.000  23.250  22.500  21.750  21.000  20.250  21.000  21.750  22.500  23.250  24.000  24.750  25.500  26.000  26.000  26.000
26.000  26.000  26.000  25.600  24.780  24.030  23.280  22.530  21.780  21.030  20.280  21.030  21.780  22.530  23.280  24.030  24.780  25.600  26.000  26.000  26.000
26.000  26.000  26.000  25.700  24.810  24.060  23.310  22.560  21.810  21.060  20.310  21.060  21.810  22.560  23.310  24.060  24.810  25.700  26.000  26.000  26.000
26.000  26.000  26.000  25.800  24.840  24.090  23.340  22.590  21.840  21.090  20.340  21.090  21.840  22.590  23.340  24.090  24.840  25.800  26.000  26.000  26.000
26.000  26.000  26.000  25.900  24.870  24.120  23.370  22.620  21.870  21.120  20.370  21.120  21.870  22.620  23.370  24.120  24.870  25.900  26.000  26.000  26.000
26.000  26.000  26.000  26.000  24.900  24.150  23.400  22.650  21.900  21.150  20.400  21.150  21.900  22.650  23.400  24.150  24.900  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  24.930  24.180  23.430  22.680  21.930  21.180  20.430  21.180  21.930  22.680  23.430  24.180  24.930  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  24.960  24.210  23.460  22.710  21.960  21.210  20.460  21.210  21.960  22.710  23.460  24.210  24.960  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  24.990  24.240  23.490  22.740  21.990  21.240  20.490  21.240  21.990  22.740  23.490  24.240  24.990  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  25.020  24.270  23.520  22.770  22.020  21.270  20.520  21.270  22.020  22.770  23.520  24.270  25.020  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  25.050  24.300  23.550  22.800  22.050  21.300  20.550  21.300  22.050  22.800  23.550  24.300  25.050  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  25.080  24.330  23.580  22.830  22.080  21.330  20.580  21.330  22.080  22.830  23.580  24.330  25.080  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  25.110  24.360  23.610  22.860  22.110  21.360  20.610  21.360  22.110  22.860  23.610  24.360  25.110  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  25.140  24.390  23.640  22.890  22.140  21.390  20.640  21.390  22.140  22.890  23.640  24.390  25.140  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  25.170  24.420  23.670  22.920  22.170  21.420  20.670  21.420  22.170  22.920  23.670  24.420  25.170  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  25.200  24.450  23.700  22.950  22.200  21.450  20.700  21.450  22.200  22.950  23.700  24.450  25.200  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  25.230  24.480  23.730  22.980  22.230  21.480  20.730  21.480  22.230  22.980  23.730  24.480  25.230  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  25.260  24.510  23.760  23.010  22.260  21.510  20.760  21.510  22.260  23.010  23.760  24.510  25.260  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  25.290  24.540  23.790  23.040  22.290  21.540  20.790  21.540  22.290  23.040  23.790  24.540  25.290  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  25.320  24.570  23.820  23.070  22.320  21.570  20.820  21.570  22.320  23.070  23.820  24.570  25.320  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  25.350  24.600  23.850  23.100  22.350  21.600  20.850  21.600  22.350  23.100  23.850  24.600  25.350  26.000  26.000  26.000  26.000
```

```
26.000  26.000  26.000  26.000  25.380  24.630  23.880  23.130  22.380  21.630  20.880  21.630  22.380  23.130  23.880  24.630  25.380  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  25.410  24.660  23.910  23.160  22.410  21.660  20.910  21.660  22.410  23.160  23.910  24.660  25.410  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  25.440  24.690  23.940  23.190  22.440  21.690  20.940  21.690  22.440  23.190  23.940  24.690  25.440  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  25.470  24.720  23.970  23.220  22.470  21.720  20.970  21.720  22.470  23.220  23.970  24.720  25.470  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  25.500  24.750  24.000  23.250  22.500  21.750  21.000  21.750  22.500  23.250  24.000  24.750  25.500  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  25.600  24.780  24.030  23.280  22.530  21.780  21.030  21.780  22.530  23.280  24.030  24.780  25.600  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  25.700  24.810  24.060  23.310  22.560  21.810  21.060  21.810  22.560  23.310  24.060  24.810  25.700  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  25.800  24.840  24.090  23.340  22.590  21.840  21.090  21.840  22.590  23.340  24.090  24.840  25.800  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  25.900  24.870  24.120  23.370  22.620  21.870  21.120  21.870  22.620  23.370  24.120  24.870  25.900  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  24.900  24.150  23.400  22.650  21.900  21.150  21.900  22.650  23.400  24.150  24.900  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  24.930  24.180  23.430  22.680  21.930  21.180  21.930  22.680  23.430  24.180  24.930  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  24.960  24.210  23.460  22.710  21.960  21.210  21.960  22.710  23.460  24.210  24.960  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  24.990  24.240  23.490  22.740  21.990  21.240  21.990  22.740  23.490  24.240  24.990  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  25.020  24.270  23.520  22.770  22.020  21.270  22.020  22.770  23.520  24.270  25.020  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  25.050  24.300  23.550  22.800  22.050  21.300  22.050  22.800  23.550  24.300  25.050  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  25.080  24.330  23.580  22.830  22.080  21.330  22.080  22.830  23.580  24.330  25.080  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  25.110  24.360  23.610  22.860  22.110  21.360  22.110  22.860  23.610  24.360  25.110  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  25.140  24.390  23.640  22.890  22.140  21.390  22.140  22.890  23.640  24.390  25.140  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  25.170  24.420  23.670  22.920  22.170  21.420  22.170  22.920  23.670  24.420  25.170  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  25.200  24.450  23.700  22.950  22.200  21.450  22.200  22.950  23.700  24.450  25.200  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  25.230  24.480  23.730  22.980  22.230  21.480  22.230  22.980  23.730  24.480  25.230  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  25.260  24.510  23.760  23.010  22.260  21.510  22.260  23.010  23.760  24.510  25.260  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  25.290  24.540  23.790  23.040  22.290  21.540  22.290  23.040  23.790  24.540  25.290  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  25.320  24.570  23.820  23.070  22.320  21.570  22.320  23.070  23.820  24.570  25.320  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  25.350  24.600  23.850  23.100  22.350  21.600  22.350  23.100  23.850  24.600  25.350  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  25.380  24.630  23.880  23.130  22.380  21.630  22.380  23.130  23.880  24.630  25.380  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  25.410  24.660  23.910  23.160  22.410  21.660  22.410  23.160  23.910  24.660  25.410  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  25.440  24.690  23.940  23.190  22.440  21.690  22.440  23.190  23.940  24.690  25.440  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  25.470  24.720  23.970  23.220  22.470  21.720  22.470  23.220  23.970  24.720  25.470  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  25.500  24.750  24.000  23.250  22.500  21.750  22.500  23.250  24.000  24.750  25.500  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  25.600  24.780  24.030  23.280  22.530  21.780  22.530  23.280  24.030  24.780  25.600  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  25.700  24.810  24.060  23.310  22.560  21.810  22.560  23.310  24.060  24.810  25.700  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  25.800  24.840  24.090  23.340  22.590  21.840  22.590  23.340  24.090  24.840  25.800  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  25.900  24.870  24.120  23.370  22.620  21.870  22.620  23.370  24.120  24.870  25.900  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  24.900  24.150  23.400  22.650  21.900  22.650  23.400  24.150  24.900  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  24.930  24.180  23.430  22.680  21.930  22.680  23.430  24.180  24.930  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  24.960  24.210  23.460  22.710  21.960  22.710  23.460  24.210  24.960  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  24.990  24.240  23.490  22.740  21.990  22.740  23.490  24.240  24.990  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  25.020  24.270  23.520  22.770  22.020  22.770  23.520  24.270  25.020  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  25.050  24.300  23.550  22.800  22.050  22.800  23.550  24.300  25.050  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  25.080  24.330  23.580  22.830  22.080  22.830  23.580  24.330  25.080  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  25.110  24.360  23.610  22.860  22.110  22.860  23.610  24.360  25.110  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  25.140  24.390  23.640  22.890  22.140  22.890  23.640  24.390  25.140  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  25.170  24.420  23.670  22.920  22.170  22.920  23.670  24.420  25.170  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  25.200  24.450  23.700  22.950  22.200  22.950  23.700  24.450  25.200  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  25.230  24.480  23.730  22.980  22.230  22.980  23.730  24.480  25.230  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  25.260  24.510  23.760  23.010  22.260  23.010  23.760  24.510  25.260  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  25.290  24.540  23.790  23.040  22.290  23.040  23.790  24.540  25.290  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  25.320  24.570  23.820  23.070  22.320  23.070  23.820  24.570  25.320  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  25.350  24.600  23.850  23.100  22.350  23.100  23.850  24.600  25.350  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  25.380  24.630  23.880  23.130  22.380  23.130  23.880  24.630  25.380  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  25.410  24.660  23.910  23.160  22.410  23.160  23.910  24.660  25.410  26.000  26.000  26.000  26.000  26.000  26.000
```

```
26.000  26.000  26.000  26.000  26.000  26.000  25.440  24.690  23.940  23.190  22.440  23.190  23.940  24.690  25.440  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  25.470  24.720  23.970  23.220  22.470  23.220  23.970  24.720  25.470  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  25.500  24.750  24.000  23.250  22.500  23.250  24.000  24.750  25.500  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  25.600  24.780  24.030  23.280  22.530  23.280  24.030  24.780  25.600  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  25.700  24.810  24.060  23.310  22.560  23.310  24.060  24.810  25.700  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  25.800  24.840  24.090  23.340  22.590  23.340  24.090  24.840  25.800  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  25.900  24.870  24.120  23.370  22.620  23.370  24.120  24.870  25.900  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.900  24.150  23.400  22.650  23.400  24.150  24.900  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.930  24.180  23.430  22.680  23.430  24.180  24.930  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.960  24.210  23.460  22.710  23.460  24.210  24.960  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.990  24.240  23.490  22.740  23.490  24.240  24.990  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.020  24.270  23.520  22.770  23.520  24.270  25.020  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.050  24.300  23.550  22.800  23.550  24.300  25.050  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.080  24.330  23.580  22.830  23.580  24.330  25.080  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.110  24.360  23.610  22.860  23.610  24.360  25.110  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.140  24.390  23.640  22.890  23.640  24.390  25.140  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.170  24.420  23.670  22.920  23.670  24.420  25.170  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.200  24.450  23.700  22.950  23.700  24.450  25.200  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.230  24.480  23.730  22.980  23.730  24.480  25.230  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.260  24.510  23.760  23.010  23.760  24.510  25.260  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.290  24.540  23.790  23.040  23.790  24.540  25.290  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.320  24.570  23.820  23.070  23.820  24.570  25.320  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.350  24.600  23.850  23.100  23.850  24.600  25.350  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.380  24.630  23.880  23.130  23.880  24.630  25.380  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.410  24.660  23.910  23.160  23.910  24.660  25.410  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.440  24.690  23.940  23.190  23.940  24.690  25.440  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.470  24.720  23.970  23.220  23.970  24.720  25.470  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.500  24.750  24.000  23.250  24.000  24.750  25.500  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.600  24.780  24.030  23.280  24.030  24.780  25.600  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.700  24.810  24.060  23.310  24.060  24.810  25.700  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.800  24.840  24.090  23.340  24.090  24.840  25.800  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.900  24.870  24.120  23.370  24.120  24.870  25.900  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.900  24.150  23.400  24.150  24.900  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.930  24.180  23.430  24.180  24.930  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.960  24.210  23.460  24.210  24.960  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.990  24.240  23.490  24.240  24.990  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.020  24.270  23.520  24.270  25.020  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.050  24.300  23.550  24.300  25.050  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.080  24.330  23.580  24.330  25.080  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.110  24.360  23.610  24.360  25.110  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.140  24.390  23.640  24.390  25.140  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.170  24.420  23.670  24.420  25.170  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.200  24.450  23.700  24.450  25.200  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.230  24.480  23.730  24.480  25.230  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.260  24.510  23.760  24.510  25.260  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.290  24.540  23.790  24.540  25.290  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.320  24.570  23.820  24.570  25.320  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.350  24.600  23.850  24.600  25.350  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.380  24.630  23.880  24.630  25.380  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.410  24.660  23.910  24.660  25.410  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.440  24.690  23.940  24.690  25.440  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.470  24.720  23.970  24.720  25.470  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
```

```
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.500  24.750  24.000  24.750  25.500  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.600  24.780  24.030  24.780  25.600  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.700  24.810  24.060  24.810  25.700  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.800  24.840  24.090  24.840  25.800  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.900  24.870  24.120  24.870  25.900  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.900  24.150  24.900  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.930  24.180  24.930  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.960  24.210  24.960  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.990  24.240  24.990  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.020  24.270  25.020  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.050  24.300  25.050  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.080  24.330  25.080  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.110  24.360  25.110  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.140  24.390  25.140  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.170  24.420  25.170  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.200  24.450  25.200  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.230  24.480  25.230  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.260  24.510  25.260  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.290  24.540  25.290  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.320  24.570  25.320  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.350  24.600  25.350  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.380  24.630  25.380  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.410  24.660  25.410  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.440  24.690  25.440  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.470  24.720  25.470  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.500  24.750  25.500  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.600  24.780  25.600  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.700  24.810  25.700  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.800  24.840  25.800  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.900  24.870  25.900  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.900  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.930  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.960  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.990  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.020  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.050  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.080  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.110  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.140  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.170  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.200  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.230  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.260  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.290  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.320  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.350  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.380  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.410  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.440  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.470  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
```

Minimum Bed Levels – ESRI ASCII Grid Format

```
ncols         21
nrows         250
xllcorner     0
yllcorner     0
cellsize      50
nodata_value  -9999
  25.500  23.000  20.500  18.000  15.500  13.000  10.500   8.000   5.500   3.000   0.500   3.000   5.500   8.000  10.500  13.000  15.500  18.000  20.500  23.000  25.500
  25.600  23.100  20.600  18.100  15.600  13.100  10.600   8.100   5.600   3.100   0.600   3.100   5.600   8.100  10.600  13.100  15.600  18.100  20.600  23.100  25.600
  25.700  23.200  20.700  18.200  15.700  13.200  10.700   8.200   5.700   3.200   0.700   3.200   5.700   8.200  10.700  13.200  15.700  18.200  20.700  23.200  25.700
  25.800  23.300  20.800  18.300  15.800  13.300  10.800   8.300   5.800   3.300   0.800   3.300   5.800   8.300  10.800  13.300  15.800  18.300  20.800  23.300  25.800
  25.900  23.400  20.900  18.400  15.900  13.400  10.900   8.400   5.900   3.400   0.900   3.400   5.900   8.400  10.900  13.400  15.900  18.400  20.900  23.400  25.900
  26.000  23.500  21.000  18.500  16.000  13.500  11.000   8.500   6.000   3.500   1.000   3.500   6.000   8.500  11.000  13.500  16.000  18.500  21.000  23.500  26.000
  26.000  23.600  21.100  18.600  16.100  13.600  11.100   8.600   6.100   3.600   1.100   3.600   6.100   8.600  11.100  13.600  16.100  18.600  21.100  23.600  26.000
  26.000  23.700  21.200  18.700  16.200  13.700  11.200   8.700   6.200   3.700   1.200   3.700   6.200   8.700  11.200  13.700  16.200  18.700  21.200  23.700  26.000
  26.000  23.800  21.300  18.800  16.300  13.800  11.300   8.800   6.300   3.800   1.300   3.800   6.300   8.800  11.300  13.800  16.300  18.800  21.300  23.800  26.000
  26.000  23.900  21.400  18.900  16.400  13.900  11.400   8.900   6.400   3.900   1.400   3.900   6.400   8.900  11.400  13.900  16.400  18.900  21.400  23.900  26.000
  26.000  24.000  21.500  19.000  16.500  14.000  11.500   9.000   6.500   4.000   1.500   4.000   6.500   9.000  11.500  14.000  16.500  19.000  21.500  24.000  26.000
  26.000  24.100  21.600  19.100  16.600  14.100  11.600   9.100   6.600   4.100   1.600   4.100   6.600   9.100  11.600  14.100  16.600  19.100  21.600  24.100  26.000
  26.000  24.200  21.700  19.200  16.700  14.200  11.700   9.200   6.700   4.200   1.700   4.200   6.700   9.200  11.700  14.200  16.700  19.200  21.700  24.200  26.000
  26.000  24.300  21.800  19.300  16.800  14.300  11.800   9.300   6.800   4.300   1.800   4.300   6.800   9.300  11.800  14.300  16.800  19.300  21.800  24.300  26.000
  26.000  24.400  21.900  19.400  16.900  14.400  11.900   9.400   6.900   4.400   1.900   4.400   6.900   9.400  11.900  14.400  16.900  19.400  21.900  24.400  26.000
  26.000  24.500  22.000  19.500  17.000  14.500  12.000   9.500   7.000   4.500   2.000   4.500   7.000   9.500  12.000  14.500  17.000  19.500  22.000  24.500  26.000
  26.000  24.600  22.100  19.600  17.100  14.600  12.100   9.600   7.100   4.600   2.100   4.600   7.100   9.600  12.100  14.600  17.100  19.600  22.100  24.600  26.000
  26.000  24.700  22.200  19.700  17.200  14.700  12.200   9.700   7.200   4.700   2.200   4.700   7.200   9.700  12.200  14.700  17.200  19.700  22.200  24.700  26.000
  26.000  24.800  22.300  19.800  17.300  14.800  12.300   9.800   7.300   4.800   2.300   4.800   7.300   9.800  12.300  14.800  17.300  19.800  22.300  24.800  26.000
  26.000  24.900  22.400  19.900  17.400  14.900  12.400   9.900   7.400   4.900   2.400   4.900   7.400   9.900  12.400  14.900  17.400  19.900  22.400  24.900  26.000
  26.000  25.000  22.500  20.000  17.500  15.000  12.500  10.000   7.500   5.000   2.500   5.000   7.500  10.000  12.500  15.000  17.500  20.000  22.500  25.000  26.000
  26.000  25.100  22.600  20.100  17.600  15.100  12.600  10.100   7.600   5.100   2.600   5.100   7.600  10.100  12.600  15.100  17.600  20.100  22.600  25.100  26.000
  26.000  25.200  22.700  20.200  17.700  15.200  12.700  10.200   7.700   5.200   2.700   5.200   7.700  10.200  12.700  15.200  17.700  20.200  22.700  25.200  26.000
  26.000  25.300  22.800  20.300  17.800  15.300  12.800  10.300   7.800   5.300   2.800   5.300   7.800  10.300  12.800  15.300  17.800  20.300  22.800  25.300  26.000
  26.000  25.400  22.900  20.400  17.900  15.400  12.900  10.400   7.900   5.400   2.900   5.400   7.900  10.400  12.900  15.400  17.900  20.400  22.900  25.400  26.000
  26.000  25.500  23.000  20.500  18.000  15.500  13.000  10.500   8.000   5.500   3.000   5.500   8.000  10.500  13.000  15.500  18.000  20.500  23.000  25.500  26.000
  26.000  25.600  23.100  20.600  18.100  15.600  13.100  10.600   8.100   5.600   3.100   5.600   8.100  10.600  13.100  15.600  18.100  20.600  23.100  25.600  26.000
  26.000  25.700  23.200  20.700  18.200  15.700  13.200  10.700   8.200   5.700   3.200   5.700   8.200  10.700  13.200  15.700  18.200  20.700  23.200  25.700  26.000
  26.000  25.800  23.300  20.800  18.300  15.800  13.300  10.800   8.300   5.800   3.300   5.800   8.300  10.800  13.300  15.800  18.300  20.800  23.300  25.800  26.000
  26.000  25.900  23.400  20.900  18.400  15.900  13.400  10.900   8.400   5.900   3.400   5.900   8.400  10.900  13.400  15.900  18.400  20.900  23.400  25.900  26.000
  26.000  26.000  23.500  21.000  18.500  16.000  13.500  11.000   8.500   6.000   3.500   6.000   8.500  11.000  13.500  16.000  18.500  21.000  23.500  26.000  26.000
  26.000  26.000  23.600  21.100  18.600  16.100  13.600  11.100   8.600   6.100   3.600   6.100   8.600  11.100  13.600  16.100  18.600  21.100  23.600  26.000  26.000
  26.000  26.000  23.700  21.200  18.700  16.200  13.700  11.200   8.700   6.200   3.700   6.200   8.700  11.200  13.700  16.200  18.700  21.200  23.700  26.000  26.000
  26.000  26.000  23.800  21.300  18.800  16.300  13.800  11.300   8.800   6.300   3.800   6.300   8.800  11.300  13.800  16.300  18.800  21.300  23.800  26.000  26.000
  26.000  26.000  23.900  21.400  18.900  16.400  13.900  11.400   8.900   6.400   3.900   6.400   8.900  11.400  13.900  16.400  18.900  21.400  23.900  26.000  26.000
  26.000  26.000  24.000  21.500  19.000  16.500  14.000  11.500   9.000   6.500   4.000   6.500   9.000  11.500  14.000  16.500  19.000  21.500  24.000  26.000  26.000
  26.000  26.000  24.100  21.600  19.100  16.600  14.100  11.600   9.100   6.600   4.100   6.600   9.100  11.600  14.100  16.600  19.100  21.600  24.100  26.000  26.000
  26.000  26.000  24.200  21.700  19.200  16.700  14.200  11.700   9.200   6.700   4.200   6.700   9.200  11.700  14.200  16.700  19.200  21.700  24.200  26.000  26.000
  26.000  26.000  24.300  21.800  19.300  16.800  14.300  11.800   9.300   6.800   4.300   6.800   9.300  11.800  14.300  16.800  19.300  21.800  24.300  26.000  26.000
  26.000  26.000  24.400  21.900  19.400  16.900  14.400  11.900   9.400   6.900   4.400   6.900   9.400  11.900  14.400  16.900  19.400  21.900  24.400  26.000  26.000
  26.000  26.000  24.500  22.000  19.500  17.000  14.500  12.000   9.500   7.000   4.500   7.000   9.500  12.000  14.500  17.000  19.500  22.000  24.500  26.000  26.000
  26.000  26.000  24.600  22.100  19.600  17.100  14.600  12.100   9.600   7.100   4.600   7.100   9.600  12.100  14.600  17.100  19.600  22.100  24.600  26.000  26.000
  26.000  26.000  24.700  22.200  19.700  17.200  14.700  12.200   9.700   7.200   4.700   7.200   9.700  12.200  14.700  17.200  19.700  22.200  24.700  26.000  26.000
```

```
26.000  26.000  24.800  22.300  19.800  17.300  14.800  12.300   9.800   7.300   4.800   7.300   9.800  12.300  14.800  17.300  19.800  22.300  24.800  26.000  26.000
26.000  26.000  24.900  22.400  19.900  17.400  14.900  12.400   9.900   7.400   4.900   7.400   9.900  12.400  14.900  17.400  19.900  22.400  24.900  26.000  26.000
26.000  26.000  25.000  22.500  20.000  17.500  15.000  12.500  10.000   7.500   5.000   7.500  10.000  12.500  15.000  17.500  20.000  22.500  25.000  26.000  26.000
26.000  26.000  25.100  22.600  20.100  17.600  15.100  12.600  10.100   7.600   5.100   7.600  10.100  12.600  15.100  17.600  20.100  22.600  25.100  26.000  26.000
26.000  26.000  25.200  22.700  20.200  17.700  15.200  12.700  10.200   7.700   5.200   7.700  10.200  12.700  15.200  17.700  20.200  22.700  25.200  26.000  26.000
26.000  26.000  25.300  22.800  20.300  17.800  15.300  12.800  10.300   7.800   5.300   7.800  10.300  12.800  15.300  17.800  20.300  22.800  25.300  26.000  26.000
26.000  26.000  25.400  22.900  20.400  17.900  15.400  12.900  10.400   7.900   5.400   7.900  10.400  12.900  15.400  17.900  20.400  22.900  25.400  26.000  26.000
26.000  26.000  25.500  23.000  20.500  18.000  15.500  13.000  10.500   8.000   5.500   8.000  10.500  13.000  15.500  18.000  20.500  23.000  25.500  26.000  26.000
26.000  26.000  25.600  23.100  20.600  18.100  15.600  13.100  10.600   8.100   5.600   8.100  10.600  13.100  15.600  18.100  20.600  23.100  25.600  26.000  26.000
26.000  26.000  25.700  23.200  20.700  18.200  15.700  13.200  10.700   8.200   5.700   8.200  10.700  13.200  15.700  18.200  20.700  23.200  25.700  26.000  26.000
26.000  26.000  25.800  23.300  20.800  18.300  15.800  13.300  10.800   8.300   5.800   8.300  10.800  13.300  15.800  18.300  20.800  23.300  25.800  26.000  26.000
26.000  26.000  25.900  23.400  20.900  18.400  15.900  13.400  10.900   8.400   5.900   8.400  10.900  13.400  15.900  18.400  20.900  23.400  25.900  26.000  26.000
26.000  26.000  26.000  23.500  21.000  18.500  16.000  13.500  11.000   8.500   6.000   8.500  11.000  13.500  16.000  18.500  21.000  23.500  26.000  26.000  26.000
26.000  26.000  26.000  23.600  21.100  18.600  16.100  13.600  11.100   8.600   6.100   8.600  11.100  13.600  16.100  18.600  21.100  23.600  26.000  26.000  26.000
26.000  26.000  26.000  23.700  21.200  18.700  16.200  13.700  11.200   8.700   6.200   8.700  11.200  13.700  16.200  18.700  21.200  23.700  26.000  26.000  26.000
26.000  26.000  26.000  23.800  21.300  18.800  16.300  13.800  11.300   8.800   6.300   8.800  11.300  13.800  16.300  18.800  21.300  23.800  26.000  26.000  26.000
26.000  26.000  26.000  23.900  21.400  18.900  16.400  13.900  11.400   8.900   6.400   8.900  11.400  13.900  16.400  18.900  21.400  23.900  26.000  26.000  26.000
26.000  26.000  26.000  24.000  21.500  19.000  16.500  14.000  11.500   9.000   6.500   9.000  11.500  14.000  16.500  19.000  21.500  24.000  26.000  26.000  26.000
26.000  26.000  26.000  24.100  21.600  19.100  16.600  14.100  11.600   9.100   6.600   9.100  11.600  14.100  16.600  19.100  21.600  24.100  26.000  26.000  26.000
26.000  26.000  26.000  24.200  21.700  19.200  16.700  14.200  11.700   9.200   6.700   9.200  11.700  14.200  16.700  19.200  21.700  24.200  26.000  26.000  26.000
26.000  26.000  26.000  24.300  21.800  19.300  16.800  14.300  11.800   9.300   6.800   9.300  11.800  14.300  16.800  19.300  21.800  24.300  26.000  26.000  26.000
26.000  26.000  26.000  24.400  21.900  19.400  16.900  14.400  11.900   9.400   6.900   9.400  11.900  14.400  16.900  19.400  21.900  24.400  26.000  26.000  26.000
26.000  26.000  26.000  24.500  22.000  19.500  17.000  14.500  12.000   9.500   7.000   9.500  12.000  14.500  17.000  19.500  22.000  24.500  26.000  26.000  26.000
26.000  26.000  26.000  24.600  22.100  19.600  17.100  14.600  12.100   9.600   7.100   9.600  12.100  14.600  17.100  19.600  22.100  24.600  26.000  26.000  26.000
26.000  26.000  26.000  24.700  22.200  19.700  17.200  14.700  12.200   9.700   7.200   9.700  12.200  14.700  17.200  19.700  22.200  24.700  26.000  26.000  26.000
26.000  26.000  26.000  24.800  22.300  19.800  17.300  14.800  12.300   9.800   7.300   9.800  12.300  14.800  17.300  19.800  22.300  24.800  26.000  26.000  26.000
26.000  26.000  26.000  24.900  22.400  19.900  17.400  14.900  12.400   9.900   7.400   9.900  12.400  14.900  17.400  19.900  22.400  24.900  26.000  26.000  26.000
26.000  26.000  26.000  25.000  22.500  20.000  17.500  15.000  12.500  10.000   7.500  10.000  12.500  15.000  17.500  20.000  22.500  25.000  26.000  26.000  26.000
26.000  26.000  26.000  25.100  22.600  20.100  17.600  15.100  12.600  10.100   7.600  10.100  12.600  15.100  17.600  20.100  22.600  25.100  26.000  26.000  26.000
26.000  26.000  26.000  25.200  22.700  20.200  17.700  15.200  12.700  10.200   7.700  10.200  12.700  15.200  17.700  20.200  22.700  25.200  26.000  26.000  26.000
26.000  26.000  26.000  25.300  22.800  20.300  17.800  15.300  12.800  10.300   7.800  10.300  12.800  15.300  17.800  20.300  22.800  25.300  26.000  26.000  26.000
26.000  26.000  26.000  25.400  22.900  20.400  17.900  15.400  12.900  10.400   7.900  10.400  12.900  15.400  17.900  20.400  22.900  25.400  26.000  26.000  26.000
26.000  26.000  26.000  25.500  23.000  20.500  18.000  15.500  13.000  10.500   8.000  10.500  13.000  15.500  18.000  20.500  23.000  25.500  26.000  26.000  26.000
26.000  26.000  26.000  25.600  23.100  20.600  18.100  15.600  13.100  10.600   8.100  10.600  13.100  15.600  18.100  20.600  23.100  25.600  26.000  26.000  26.000
26.000  26.000  26.000  25.700  23.200  20.700  18.200  15.700  13.200  10.700   8.200  10.700  13.200  15.700  18.200  20.700  23.200  25.700  26.000  26.000  26.000
26.000  26.000  26.000  25.800  23.300  20.800  18.300  15.800  13.300  10.800   8.300  10.800  13.300  15.800  18.300  20.800  23.300  25.800  26.000  26.000  26.000
26.000  26.000  26.000  25.900  23.400  20.900  18.400  15.900  13.400  10.900   8.400  10.900  13.400  15.900  18.400  20.900  23.400  25.900  26.000  26.000  26.000
26.000  26.000  26.000  26.000  23.500  21.000  18.500  16.000  13.500  11.000   8.500  11.000  13.500  16.000  18.500  21.000  23.500  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  23.600  21.100  18.600  16.100  13.600  11.100   8.600  11.100  13.600  16.100  18.600  21.100  23.600  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  23.700  21.200  18.700  16.200  13.700  11.200   8.700  11.200  13.700  16.200  18.700  21.200  23.700  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  23.800  21.300  18.800  16.300  13.800  11.300   8.800  11.300  13.800  16.300  18.800  21.300  23.800  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  23.900  21.400  18.900  16.400  13.900  11.400   8.900  11.400  13.900  16.400  18.900  21.400  23.900  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  24.000  21.500  19.000  16.500  14.000  11.500   9.000  11.500  14.000  16.500  19.000  21.500  24.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  24.100  21.600  19.100  16.600  14.100  11.600   9.100  11.600  14.100  16.600  19.100  21.600  24.100  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  24.200  21.700  19.200  16.700  14.200  11.700   9.200  11.700  14.200  16.700  19.200  21.700  24.200  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  24.300  21.800  19.300  16.800  14.300  11.800   9.300  11.800  14.300  16.800  19.300  21.800  24.300  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  24.400  21.900  19.400  16.900  14.400  11.900   9.400  11.900  14.400  16.900  19.400  21.900  24.400  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  24.500  22.000  19.500  17.000  14.500  12.000   9.500  12.000  14.500  17.000  19.500  22.000  24.500  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  24.600  22.100  19.600  17.100  14.600  12.100   9.600  12.100  14.600  17.100  19.600  22.100  24.600  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  24.700  22.200  19.700  17.200  14.700  12.200   9.700  12.200  14.700  17.200  19.700  22.200  24.700  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  24.800  22.300  19.800  17.300  14.800  12.300   9.800  12.300  14.800  17.300  19.800  22.300  24.800  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  24.900  22.400  19.900  17.400  14.900  12.400   9.900  12.400  14.900  17.400  19.900  22.400  24.900  26.000  26.000  26.000  26.000
```

```
26.000  26.000  26.000  26.000  25.000  22.500  20.000  17.500  15.000  12.500  10.000  12.500  15.000  17.500  20.000  22.500  25.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  25.100  22.600  20.100  17.600  15.100  12.600  10.100  12.600  15.100  17.600  20.100  22.600  25.100  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  25.200  22.700  20.200  17.700  15.200  12.700  10.200  12.700  15.200  17.700  20.200  22.700  25.200  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  25.300  22.800  20.300  17.800  15.300  12.800  10.300  12.800  15.300  17.800  20.300  22.800  25.300  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  25.400  22.900  20.400  17.900  15.400  12.900  10.400  12.900  15.400  17.900  20.400  22.900  25.400  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  25.500  23.000  20.500  18.000  15.500  13.000  10.500  13.000  15.500  18.000  20.500  23.000  25.500  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  25.600  23.100  20.600  18.100  15.600  13.100  10.600  13.100  15.600  18.100  20.600  23.100  25.600  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  25.700  23.200  20.700  18.200  15.700  13.200  10.700  13.200  15.700  18.200  20.700  23.200  25.700  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  25.800  23.300  20.800  18.300  15.800  13.300  10.800  13.300  15.800  18.300  20.800  23.300  25.800  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  25.900  23.400  20.900  18.400  15.900  13.400  10.900  13.400  15.900  18.400  20.900  23.400  25.900  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  23.500  21.000  18.500  16.000  13.500  11.000  13.500  16.000  18.500  21.000  23.500  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  23.600  21.100  18.600  16.100  13.600  11.100  13.600  16.100  18.600  21.100  23.600  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  23.700  21.200  18.700  16.200  13.700  11.200  13.700  16.200  18.700  21.200  23.700  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  23.800  21.300  18.800  16.300  13.800  11.300  13.800  16.300  18.800  21.300  23.800  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  23.900  21.400  18.900  16.400  13.900  11.400  13.900  16.400  18.900  21.400  23.900  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  24.000  21.500  19.000  16.500  14.000  11.500  14.000  16.500  19.000  21.500  24.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  24.100  21.600  19.100  16.600  14.100  11.600  14.100  16.600  19.100  21.600  24.100  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  24.200  21.700  19.200  16.700  14.200  11.700  14.200  16.700  19.200  21.700  24.200  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  24.300  21.800  19.300  16.800  14.300  11.800  14.300  16.800  19.300  21.800  24.300  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  24.400  21.900  19.400  16.900  14.400  11.900  14.400  16.900  19.400  21.900  24.400  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  24.500  22.000  19.500  17.000  14.500  12.000  14.500  17.000  19.500  22.000  24.500  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  24.600  22.100  19.600  17.100  14.600  12.100  14.600  17.100  19.600  22.100  24.600  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  24.700  22.200  19.700  17.200  14.700  12.200  14.700  17.200  19.700  22.200  24.700  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  24.800  22.300  19.800  17.300  14.800  12.300  14.800  17.300  19.800  22.300  24.800  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  24.900  22.400  19.900  17.400  14.900  12.400  14.900  17.400  19.900  22.400  24.900  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  25.000  22.500  20.000  17.500  15.000  12.500  15.000  17.500  20.000  22.500  25.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  25.100  22.600  20.100  17.600  15.100  12.600  15.100  17.600  20.100  22.600  25.100  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  25.200  22.700  20.200  17.700  15.200  12.700  15.200  17.700  20.200  22.700  25.200  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  25.300  22.800  20.300  17.800  15.300  12.800  15.300  17.800  20.300  22.800  25.300  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  25.400  22.900  20.400  17.900  15.400  12.900  15.400  17.900  20.400  22.900  25.400  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  25.500  23.000  20.500  18.000  15.500  13.000  15.500  18.000  20.500  23.000  25.500  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  25.600  23.100  20.600  18.100  15.600  13.100  15.600  18.100  20.600  23.100  25.600  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  25.700  23.200  20.700  18.200  15.700  13.200  15.700  18.200  20.700  23.200  25.700  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  25.800  23.300  20.800  18.300  15.800  13.300  15.800  18.300  20.800  23.300  25.800  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  25.900  23.400  20.900  18.400  15.900  13.400  15.900  18.400  20.900  23.400  25.900  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  23.500  21.000  18.500  16.000  13.500  16.000  18.500  21.000  23.500  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  23.600  21.100  18.600  16.100  13.600  16.100  18.600  21.100  23.600  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  23.700  21.200  18.700  16.200  13.700  16.200  18.700  21.200  23.700  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  23.800  21.300  18.800  16.300  13.800  16.300  18.800  21.300  23.800  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  23.900  21.400  18.900  16.400  13.900  16.400  18.900  21.400  23.900  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  24.000  21.500  19.000  16.500  14.000  16.500  19.000  21.500  24.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  24.100  21.600  19.100  16.600  14.100  16.600  19.100  21.600  24.100  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  24.200  21.700  19.200  16.700  14.200  16.700  19.200  21.700  24.200  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  24.300  21.800  19.300  16.800  14.300  16.800  19.300  21.800  24.300  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  24.400  21.900  19.400  16.900  14.400  16.900  19.400  21.900  24.400  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  24.500  22.000  19.500  17.000  14.500  17.000  19.500  22.000  24.500  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  24.600  22.100  19.600  17.100  14.600  17.100  19.600  22.100  24.600  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  24.700  22.200  19.700  17.200  14.700  17.200  19.700  22.200  24.700  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  24.800  22.300  19.800  17.300  14.800  17.300  19.800  22.300  24.800  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  24.900  22.400  19.900  17.400  14.900  17.400  19.900  22.400  24.900  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  25.000  22.500  20.000  17.500  15.000  17.500  20.000  22.500  25.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  25.100  22.600  20.100  17.600  15.100  17.600  20.100  22.600  25.100  26.000  26.000  26.000  26.000  26.000  26.000
```

```
26.000  26.000  26.000  26.000  26.000  26.000  25.200  22.700  20.200  17.700  15.200  17.700  20.200  22.700  25.200  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  25.300  22.800  20.300  17.800  15.300  17.800  20.300  22.800  25.300  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  25.400  22.900  20.400  17.900  15.400  17.900  20.400  22.900  25.400  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  25.500  23.000  20.500  18.000  15.500  18.000  20.500  23.000  25.500  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  25.600  23.100  20.600  18.100  15.600  18.100  20.600  23.100  25.600  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  25.700  23.200  20.700  18.200  15.700  18.200  20.700  23.200  25.700  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  25.800  23.300  20.800  18.300  15.800  18.300  20.800  23.300  25.800  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  25.900  23.400  20.900  18.400  15.900  18.400  20.900  23.400  25.900  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  23.500  21.000  18.500  16.000  18.500  21.000  23.500  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  23.600  21.100  18.600  16.100  18.600  21.100  23.600  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  23.700  21.200  18.700  16.200  18.700  21.200  23.700  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  23.800  21.300  18.800  16.300  18.800  21.300  23.800  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  23.900  21.400  18.900  16.400  18.900  21.400  23.900  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.000  21.500  19.000  16.500  19.000  21.500  24.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.100  21.600  19.100  16.600  19.100  21.600  24.100  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.200  21.700  19.200  16.700  19.200  21.700  24.200  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.300  21.800  19.300  16.800  19.300  21.800  24.300  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.400  21.900  19.400  16.900  19.400  21.900  24.400  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.500  22.000  19.500  17.000  19.500  22.000  24.500  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.600  22.100  19.600  17.100  19.600  22.100  24.600  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.700  22.200  19.700  17.200  19.700  22.200  24.700  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.800  22.300  19.800  17.300  19.800  22.300  24.800  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.900  22.400  19.900  17.400  19.900  22.400  24.900  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.000  22.500  20.000  17.500  20.000  22.500  25.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.100  22.600  20.100  17.600  20.100  22.600  25.100  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.200  22.700  20.200  17.700  20.200  22.700  25.200  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.300  22.800  20.300  17.800  20.300  22.800  25.300  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.400  22.900  20.400  17.900  20.400  22.900  25.400  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.500  23.000  20.500  18.000  20.500  23.000  25.500  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.600  23.100  20.600  18.100  20.600  23.100  25.600  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.700  23.200  20.700  18.200  20.700  23.200  25.700  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.800  23.300  20.800  18.300  20.800  23.300  25.800  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.900  23.400  20.900  18.400  20.900  23.400  25.900  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  23.500  21.000  18.500  21.000  23.500  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  23.600  21.100  18.600  21.100  23.600  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  23.700  21.200  18.700  21.200  23.700  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  23.800  21.300  18.800  21.300  23.800  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  23.900  21.400  18.900  21.400  23.900  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.000  21.500  19.000  21.500  24.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.100  21.600  19.100  21.600  24.100  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.200  21.700  19.200  21.700  24.200  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.300  21.800  19.300  21.800  24.300  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.400  21.900  19.400  21.900  24.400  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.500  22.000  19.500  22.000  24.500  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.600  22.100  19.600  22.100  24.600  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.700  22.200  19.700  22.200  24.700  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.800  22.300  19.800  22.300  24.800  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.900  22.400  19.900  22.400  24.900  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.000  22.500  20.000  22.500  25.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.100  22.600  20.100  22.600  25.100  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.200  22.700  20.200  22.700  25.200  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.300  22.800  20.300  22.800  25.300  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
```

```
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.400  22.900  20.400  22.900  25.400  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.500  23.000  20.500  23.000  25.500  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.600  23.100  20.600  23.100  25.600  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.700  23.200  20.700  23.200  25.700  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.800  23.300  20.800  23.300  25.800  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.900  23.400  20.900  23.400  25.900  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  23.500  21.000  23.500  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  23.600  21.100  23.600  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  23.700  21.200  23.700  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  23.800  21.300  23.800  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  23.900  21.400  23.900  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.000  21.500  24.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.100  21.600  24.100  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.200  21.700  24.200  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.300  21.800  24.300  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.400  21.900  24.400  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.500  22.000  24.500  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.600  22.100  24.600  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.700  22.200  24.700  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.800  22.300  24.800  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.900  22.400  24.900  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.000  22.500  25.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.100  22.600  25.100  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.200  22.700  25.200  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.300  22.800  25.300  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.400  22.900  25.400  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.500  23.000  25.500  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.600  23.100  25.600  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.700  23.200  25.700  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.800  23.300  25.800  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.900  23.400  25.900  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  23.500  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  23.600  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  23.700  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  23.800  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  23.900  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.100  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.200  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.300  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.400  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.500  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.600  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.700  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.800  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  24.900  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.100  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.200  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.300  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  25.400  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000  26.000
```

Initial Sediment Types – ESRI ASCII Grid Format

```
ncols         21
nrows         250
xllcorner     0
yllcorner     0
cellsize      50
nodata_value  -9999
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
```

```
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
```

```
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000  3.000
```

```
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000   3.000
```

3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000
3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000	3.000

Sediment File (Multi-Fraction Sediment Option)

```
003                 #(Number of sediment fractions)
+000.00 +100.00     #(Min & Max Levels for sublayers)
0.200 0.200         #(Sub-layer thickness, Active layer thickness)
N  Min(m)    Max(m)
01 0.0000750 0.0001250
02 0.0001250 0.0001750
03 0.0001750 0.0002250
005                 #Number of initial sediment types - Proportion of each fraction listed for each type
01 0.300 0.400 0.300
02 0.000 0.000 0.000
03 0.150 0.200 0.150
04 0.000 0.000 0.000
05 0.000 0.000 0.000
```

Appendix C

EXAMPLE OUTPUT FILES

Example Output Files

Example output files for the baseline simulation described in Chapter 5. Output may also be generated in ESRI ASCII grid format giving the depth of mud in the active layer (or median grain size for multi-fraction simulations), wave heights, velocities (x and y components in separate grids) and depth of mud in the active layer.

File	Purpose	Page nr.
TS.txt	Time series output for first tidal cycle, at row 210, column 6 in the model lattice.	452
MASS_BALANCE.txt	Sediment mass balance file	453
GRID_430000.asc	Bathymetry output in ESRI ASCII grid format at a simulation time of 430,000 hours	471

TS.txt

Time series output for:

C:\Users\iwben_000\Documents\Model\Test_Dec14\Sensitivity_Jan15\crl_files_in_revised_format\baseline1.crl

ROW = 210

COL = 6

TIME	VX	VY	H
0.000	0.000	0.000	0.000
0.621	0.000	0.000	0.000
1.242	0.000	0.000	0.000
1.863	0.000	0.000	0.000
2.484	0.000	0.000	0.000
3.105	0.000	0.000	0.000
3.726	0.000	0.000	0.000
4.347	0.000	0.000	0.000
4.968	0.000	0.000	0.000
5.589	0.000	-0.090	0.223
6.210	-0.003	-0.145	0.672
6.831	-0.004	-0.155	1.084
7.452	-0.001	-0.126	1.395
8.073	-0.001	-0.069	1.575
8.694	0.001	0.003	1.606
9.315	0.002	0.079	1.484
9.936	0.004	0.143	1.222
10.556	0.009	0.174	0.845
11.177	0.013	0.153	0.389

MASS_BALANCE.txt

TIME	SAND IN	SAND OUT	MUD IN	MUD OUT	TOTAL SAND VOLUME	TOTAL MUD VOLUME	TOTAL CURRENT VOL.
0.000000	0.000000	0.000000	0.000000	0.000000	161126562.500000	161126562.500000	322253125.000000
1000.385254	180.180741	0.000000	5273.461183	181428.945165	161126742.680721	160950407.016018	322077149.696738
2000.149658	360.137743	3.370988	10604.666770	280111.880444	161126919.266829	160857055.286326	321983974.553155
3000.347900	540.206476	51.094918	16087.892986	359109.366591	161127051.611654	160783541.026395	321910592.638049
4000.536621	720.274904	132.634386	21647.994491	421504.218432	161127150.140612	160726706.276060	321853856.416672
5000.459961	900.231412	242.120964	27118.267699	480424.082605	161127220.610429	160673256.685095	321800477.295523
6000.420898	1080.187598	430.926869	32440.377352	532637.387662	161127211.760717	160626365.489692	321753577.250409
7000.381836	1260.142502	703.027395	37861.239984	579643.731203	161127119.614913	160584780.008782	321711899.623695
8000.342773	1440.092927	1005.003719	43387.279840	621067.209069	161126997.589113	160548882.570772	321675880.159884
9000.303711	1620.032567	1341.771312	48885.349477	658932.791545	161126840.761109	160516515.057933	321643355.819041
10000.264648	1799.977526	1750.350873	54240.287724	695800.774106	161126612.126259	160485002.013617	321611614.139876
11000.225586	1979.928343	2201.876520	59570.850248	730375.551440	161126340.551340	160455757.798808	321582098.350148
12000.186523	2159.879421	2697.921917	65037.675557	764237.662688	161126024.457173	160427362.512870	321553386.970042
13000.147461	2339.828759	3151.373985	70535.286585	794631.963320	161125750.954465	160402465.823264	321528216.777730
14000.108398	2519.776945	3757.562292	75956.226723	826531.900629	161125324.714321	160375986.826094	321501311.540415
15000.069336	2699.723157	4379.965052	81262.194236	855152.318429	161124882.257880	160352672.375807	321477554.633687
16000.030273	2879.669008	5098.052298	86671.647360	881158.760633	161124344.116523	160332075.386727	321456419.503250
17000.611328	3059.725239	5730.297357	92181.231863	903751.060749	161123891.927778	160314992.671114	321438884.598892
18000.572266	3239.665229	6496.379826	97643.412812	927699.690111	161123305.785289	160296506.222700	321419812.007990
19000.533203	3419.600361	7351.950181	103000.343995	951948.387743	161122630.150017	160277614.456253	321400244.606270
20000.494141	3599.534032	8223.382770	108331.734692	976209.482214	161121938.651069	160258684.752479	321380623.403549

21000.455078	3779.471250	9009.312645	113794.791465	998325.157058	161121332.658395	160242032.134407	321363364.792801
22000.416016	3959.405918	9716.260592	119247.703297	1018750.647299	161120805.644899	160227059.555998	321347865.200898
23000.376953	4139.335615	10583.940438	124623.311307	1040367.016438	161120117.894635	160210818.794871	321330936.689506
24000.337891	4319.263879	11474.722448	129869.304330	1061942.559413	161119407.041101	160194489.244918	321313896.286019
25000.298828	4499.193929	12351.379752	135231.382622	1082783.358947	161118710.313991	160179010.523674	321297720.837666
26000.259766	4679.124822	13068.117496	140631.271087	1101717.837165	161118173.507141	160165475.933923	321283649.441064
27000.220703	4859.056799	13900.254538	146007.854900	1121336.938318	161117521.301958	160151233.416582	321268754.718539
28000.181641	5038.984144	14835.839439	151233.502849	1141927.926039	161116765.644514	160135868.076810	321252633.721324
29000.142578	5218.910475	15791.596396	156486.688672	1162554.776387	161115989.813911	160120494.412285	321236484.226197
30000.103516	5398.841437	16662.871650	161823.152576	1181982.541558	161115298.469786	160106403.111018	321221701.580804
31000.064453	5578.771572	17442.436059	167182.528706	1200124.313586	161114698.835758	160093620.715121	321208319.550879
32000.025391	5758.695846	18421.700785	172433.899838	1220546.246031	161113899.495337	160078450.153808	321192349.649145
33000.605469	5938.731339	19413.094848	177622.643565	1239963.892898	161113088.136851	160064221.250668	321177309.387519
34000.566406	6118.658447	20389.559729	182886.646280	1258300.393914	161112291.599013	160051148.752368	321163440.351380
35000.527344	6298.584817	21149.477351	188240.677839	1274325.818559	161111711.607626	160040477.359281	321152188.966908
36000.488281	6478.511990	22087.791136	193526.339663	1291894.883028	161110953.221304	160028193.956635	321139147.177938
37000.449219	6658.434651	23086.547758	198714.302245	1310258.285725	161110134.387168	160015018.516521	321125152.903689
38000.410156	6838.356823	24116.286810	203905.824564	1328446.522743	161109284.570367	160002021.801820	321111306.372188
39000.371094	7018.284480	25012.407809	209241.686239	1344194.880934	161108568.377276	159991609.305305	321100177.682580
40000.332031	7198.211364	25890.231507	214554.933573	1359163.823601	161107870.480342	159981953.609973	321089824.090315
41000.292969	7378.131440	26933.822310	219807.141254	1375634.745528	161107006.809466	159970734.895727	321077741.705192
42000.253906	7558.051674	27988.587810	224951.512979	1392570.118612	161106131.964068	159958943.894367	321065075.858435
43000.214844	7737.976188	29005.627902	230249.715484	1408273.022635	161105294.848477	159948539.192849	321053834.041327
44000.175781	7917.902288	29834.630866	235569.473665	1422022.039155	161104645.771398	159940109.934511	321044755.705909
45000.136719	8097.827571	30843.012570	240877.514927	1437277.926500	161103817.315112	159930162.088428	321033979.403540

46000.097656	8277.751101	31917.351404	246024.460323	1453221.664106	161102922.900025	159919365.296219	321022288.196243
47000.058594	8457.674149	32999.007234	251260.489214	1468744.958308	161102021.167149	159909078.030906	321011099.198054
48000.019531	8637.602042	33943.868448	256574.252809	1482641.066773	161101256.233833	159900495.686037	321001751.919870
49000.601562	8817.642951	34864.730609	261929.342015	1495895.571645	161100515.412704	159892596.270371	320993111.683075
50000.562500	8997.565411	35973.270286	267143.220469	1510590.055933	161099586.795483	159883115.664537	320982702.460020
51000.523438	9177.487698	37085.632370	272331.340163	1525318.874390	161098654.355949	159873574.965774	320972229.321724
52000.484375	9357.412784	38154.598284	277616.721843	1539315.694177	161097765.314776	159864863.527667	320962628.842443
53000.445312	9537.339672	38993.954103	282994.134051	1551409.533254	161097105.885751	159858147.100799	320955252.986549
54000.406250	9717.265683	40081.053082	288282.857154	1565215.129162	161096198.712778	159849630.227994	320945828.940771
55000.367188	9897.178959	41213.342547	293478.868840	1579435.393149	161095246.336339	159840605.975693	320935852.312032
56000.328125	10077.056887	42349.162925	298711.984648	1593516.881421	161094290.393891	159831757.603229	320926047.997121
57000.289062	10256.931024	43279.274914	304091.704083	1605729.951179	161093540.156022	159824924.252908	320918464.408931
58000.250000	10436.791502	44281.060263	309431.810769	1618222.568503	161092718.231040	159817771.742269	320910489.973310
59000.210938	10616.615247	45430.094748	314696.016002	1631984.651150	161091749.020262	159809273.864854	320901022.885115
60000.171875	10796.472031	46590.715191	319878.976178	1646061.704765	161090768.256492	159800379.771415	320891148.027907
61000.132812	10976.283811	47679.334306	325238.673916	1659338.116515	161089859.449190	159792463.057403	320882322.506593
62000.093750	11156.087760	48597.034584	330607.651338	1671009.571571	161089121.552705	159786160.579770	320875282.132475
63000.054688	11335.876546	49744.279451	335952.421560	1683746.667505	161088154.096610	159778768.254058	320866922.350668
64000.015625	11515.685919	50940.517611	341133.115596	1696662.405363	161087137.667802	159771033.210236	320858170.878038
65000.597656	11695.547470	52136.009272	346443.593015	1709430.400891	161086122.037805	159763575.692128	320849697.729932
66000.109375	11875.718488	53133.276697	351820.401488	1720783.547589	161085304.941630	159757599.353904	320842904.295534
67000.570312	12056.587095	54205.097232	357241.646111	1732191.693574	161084413.989713	159751612.452544	320836026.442257
68000.414062	12237.316780	55439.714764	362567.310079	1744502.114839	161083360.101977	159744627.695245	320827987.797222
69000.257812	12417.942351	56688.803262	367821.677714	1756881.963997	161082291.639155	159737502.213723	320819793.852879
70000.101562	12598.463064	57920.440653	373192.905267	1768924.062998	161081240.522311	159730831.342276	320812071.864586

71000.562500	12779.065655	58921.527482	378650.717616	1779519.775036	161080420.037907	159725693.442586	320806113.480493
72000.406250	12959.565452	60084.654691	384083.448575	1790984.434359	161079437.410476	159719661.514221	320799098.924696
73000.250000	13140.204702	61370.927427	389408.258342	1803257.354711	161078331.776902	159712713.403635	320791045.180537
74000.093750	13320.804745	62670.678065	394693.958329	1815536.125665	161077212.626304	159705720.332669	320782932.958974
75000.554688	13501.419853	63928.491087	400120.019125	1827321.627672	161076135.428430	159699360.891460	320775496.319891
76000.398438	13681.873796	64952.930492	405588.800325	1837617.906090	161075291.442908	159694533.394242	320769824.837150
77000.242188	13862.292885	66207.673255	411025.353530	1848963.210614	161074217.119149	159688624.642925	320762841.762073
78000.085938	14042.706654	67546.307081	416345.019868	1860643.826052	161073058.899110	159682263.693824	320755322.592935
79000.546875	14223.110922	68893.116207	421677.899842	1872078.697789	161071892.494427	159676161.702061	320748054.196487
80000.390625	14403.355223	70156.529399	427125.596873	1882846.583318	161070809.325437	159670841.513564	320741650.839000
81000.234375	14583.575622	71220.642526	432613.891161	1892372.207167	161069925.432608	159666804.184002	320736729.616610
82000.078125	14763.900994	72551.521706	438053.943637	1903177.816988	161068774.878818	159661438.626657	320730213.505476
83000.539062	14944.261712	73937.850600	443380.065658	1914342.384325	161067568.910625	159655600.181342	320723169.091967
84000.382812	15124.376769	75333.936207	448737.271187	1925326.743418	161066352.939970	159649973.027779	320716325.967749
85000.226562	15304.450027	76597.154243	454213.344479	1935506.505578	161065269.795178	159645269.338909	320710539.134087
86000.070312	15484.487492	77735.804896	459716.549740	1944945.318881	161064311.181988	159641333.730867	320705644.912856
87000.531250	15664.844624	79154.805361	465158.477296	1955581.896473	161063072.538665	159636139.080830	320699211.619495
88000.375000	15845.002556	80627.857048	470483.375320	1966314.617336	161061779.645013	159630731.257991	320692510.903004
89000.218750	16025.060108	82144.387799	475877.772300	1976840.540826	161060443.171848	159625599.731481	320686042.903329
90000.062500	16205.107109	83516.398289	481380.560393	1986280.448165	161059251.208165	159621662.612234	320680913.820400
91000.523438	16385.265209	84949.425902	486898.356936	1995385.189862	161057998.338534	159618075.667081	320676074.005615
92000.367188	16565.549090	86866.822403	492336.546104	2005524.248735	161056261.225967	159613374.797376	320669636.023343
93000.210938	16745.789913	89193.089347	497674.864310	2015810.140973	161054115.199826	159608427.223343	320662542.423170
94000.054688	16925.877915	92092.329887	503111.051731	2025978.034005	161051396.047354	159603695.517732	320655091.565087
95000.515625	17106.119399	95171.214651	508642.533136	2035171.306464	161048497.404112	159600033.726678	320648531.130790

96000.359375	17286.253642	98949.645006	514175.470989	2044382.011641	161044899.108016	159596355.959355	320641255.067370
97000.203125	17466.591598	104119.198632	519615.119860	2054540.128525	161039909.892387	159591637.491341	320631547.383728
98000.046875	17646.799074	109973.749636	524981.831254	2064757.146657	161034235.548701	159586787.184604	320621022.733305
99000.507812	17827.030828	116143.181500	530468.885828	2074758.615374	161028246.348548	159582272.770461	320610519.119010
100000.351562	18007.144626	121436.488690	536037.588968	2083617.315715	161023133.155076	159578982.773261	320602115.928337
101000.195312	18187.249121	127248.220295	541595.826125	2092919.580604	161017501.527947	159575238.745528	320592740.273475
102000.039062	18367.591874	133883.517581	547047.450116	2103022.549851	161011046.573375	159570587.400273	320581633.973648
103000.500000	18547.936319	140543.236513	552459.071100	2113160.894493	161004567.198931	159565860.676617	320570427.875548
104000.343750	18728.068186	146981.494886	558002.616426	2122907.119127	160998309.072630	159561657.997308	320559967.069937
105000.187500	18908.145822	152209.826186	563613.913158	2131416.546205	160993260.818924	159558759.866961	320552020.685885
106000.031250	19088.382054	158095.944765	569199.630391	2140605.967860	160987554.936714	159555156.162539	320542711.099253
107000.492188	19268.761951	164495.056712	574668.472657	2150402.151788	160981336.204551	159550828.820879	320532165.025429
108000.335938	19449.001674	170806.861840	580132.602156	2160224.202397	160975204.639277	159546470.899770	320521675.539047
109000.179688	19629.203440	176742.705842	585732.415424	2169694.986036	160969448.997100	159542599.929396	320512048.926496
110000.023438	19809.375685	181559.038435	591382.338025	2177994.736990	160964812.836773	159539950.101045	320504762.937819
111000.484375	19989.807650	187246.300727	596990.179039	2187292.618197	160959306.006474	159536260.060851	320495566.067325
112000.328125	20169.906733	193215.981624	602477.941488	2197063.322483	160953516.424607	159531977.119015	320485493.543622
113000.171875	20349.985312	199096.430712	607993.990279	2206793.597133	160947816.053999	159527762.893156	320475578.947154
114000.015625	20529.977377	204510.046938	613642.081951	2215979.697063	160942582.429912	159524224.884897	320466807.314808
115000.476562	20710.069564	209055.946790	619328.282083	2224210.203355	160938216.622176	159521680.578738	320459897.200915
116000.320312	20890.336577	214552.690486	624952.995342	2233568.425221	160932900.145549	159517947.070130	320450847.215679
117000.164062	21070.380045	220196.606327	630460.946855	2243194.129174	160927436.273196	159513829.317689	320441265.590885
118000.007812	21250.457570	225769.359432	636030.976625	2252793.180842	160922043.597683	159509800.295792	320431843.893475
119000.468750	21430.636257	230696.412637	641724.034679	2261708.187597	160917296.723253	159506578.347091	320423875.070345
120000.312500	21610.631332	235074.432506	647437.807573	2269955.447774	160913098.698562	159504044.859807	320417143.558369

121000.156250	21790.765356	240348.383978	653076.377031	2279364.088654	160908004.881154	159500274.788386	320408279.669539
122000.000000	21970.820997	245654.134490	658608.830933	2288963.176855	160902879.186120	159496208.154086	320399087.340206
123000.460938	22150.907577	250858.133418	664233.023709	2298400.238739	160897855.273737	159492395.284979	320390250.558715
124000.304688	22330.829830	255286.684168	669965.544220	2306978.913470	160893606.645291	159489549.130758	320383155.776049
125000.148438	22510.749607	259542.398153	675708.148369	2315295.151816	160889530.851116	159486975.496560	320376506.347677
126000.609375	22691.145091	264556.466816	681360.749854	2324629.110245	160884697.178138	159483294.139617	320367991.317755
127000.453125	22871.316981	269563.659588	686925.446707	2334095.457265	160879870.157185	159479392.489451	320359262.646636
128000.296875	23051.452042	274451.133489	692602.471955	2343482.207048	160875162.818219	159475682.764915	320350845.583134
129000.140625	23231.696691	278472.229947	698375.387422	2351771.993904	160871321.966422	159473165.893527	320344487.859949
130000.601562	23412.106997	282683.394982	704143.951977	2360240.712311	160867291.211865	159470465.739675	320337756.951541
131000.445312	23592.628034	287467.301525	709805.437646	2369536.226710	160862687.826362	159466831.710945	320329519.537307
132000.171875	23770.850159	292216.094056	715330.387783	2378863.939176	160858117.255958	159463028.948617	320321146.204575
133000.171875	23948.708243	296573.801450	721023.565038	2387702.932871	160853937.406673	159459883.132176	320313820.538849
134000.171875	24126.448064	300135.293074	726775.918155	2395466.291225	160850553.654798	159457872.126940	320308425.781738
135000.171875	24304.384157	304648.894281	732427.112656	2404559.139260	160846217.989693	159454430.473405	320300648.463098
136000.171875	24482.293340	309195.742751	737965.210269	2413825.243535	160841849.050437	159450702.466745	320292551.517182
137000.171875	24660.098989	313507.618863	743662.480871	2422777.723689	160837714.979922	159447447.257190	320285162.237112
138000.171875	24837.841333	316881.728887	749443.847426	2430368.811718	160834518.612282	159445637.535715	320280156.147998
139000.171875	25015.613749	321156.742783	755145.922313	2439180.452873	160830421.370775	159442527.969447	320272949.340223
140000.171875	25193.326834	325555.085924	760706.707726	2448306.481969	160826200.740740	159438962.725765	320265163.466505
141000.171875	25370.812555	329808.209357	766404.102211	2457241.103889	160822125.103027	159435725.498330	320257850.601356
142000.171875	25548.234959	333089.629627	772209.360227	2464824.585095	160819021.105174	159433947.275140	320252968.380313
143000.171875	25725.831373	337114.992145	777957.936178	2473411.080547	160815173.339078	159431109.355639	320246282.694718
144000.171875	25903.555619	341369.980368	783548.087719	2482457.023085	160811096.075131	159427653.564641	320238749.639773
145000.171875	26081.097130	345537.718107	789241.383828	2491337.066810	160807105.878946	159424466.817025	320231572.695971

146000.171875	26258.600266	348800.048652	795066.791077	2498929.936741	160804021.051574	159422699.354344	320226720.405919
147000.171875	26436.255664	352557.330862	800859.483647	2507157.866797	160800441.424745	159420264.116858	320220705.541601
148000.171875	26614.220669	356674.778493	806487.924252	2515988.009914	160796501.942021	159417062.414345	320213564.356366
149000.171875	26792.137705	360739.224286	812177.278167	2524732.895342	160792615.413241	159414006.882832	320206622.296074
150000.171875	26970.010142	364035.090323	818019.360893	2532421.893140	160789497.419681	159412159.967760	320201657.387441
151000.171875	27147.751508	367521.771791	823851.216649	2540341.924105	160786188.479633	159410071.792550	320196260.272183
152000.171875	27325.690741	371510.994149	829523.241536	2549129.197728	160782377.196552	159406956.543815	320189333.740367
153000.171875	27503.438213	375450.372945	835208.289539	2557902.564950	160778615.565227	159403868.224596	320182483.789823
154000.171875	27681.134799	378798.539483	841062.006459	2565818.459542	160775445.095340	159401806.046925	320177251.142265
155000.171875	27858.823334	382020.234610	846928.537434	2573466.613973	160772401.088733	159400024.423469	320172425.512202
156000.171875	28036.721452	385883.193524	852647.975088	2582185.549888	160768716.028006	159397024.925207	320165740.953214
157000.171875	28214.595416	389725.895188	858331.292386	2590940.328932	160765051.200300	159393953.463461	320159004.663762
158000.171875	28392.379721	393133.193739	864192.206097	2599072.388210	160761821.686012	159391682.317894	320153504.003907
159000.171875	28570.137521	396118.811844	870089.541191	2606485.095133	160759013.825647	159390166.946065	320149180.771712
160000.171875	28747.944100	399848.517663	875858.752983	2615121.751181	160755461.926365	159387299.501808	320142761.428174
161000.171875	28925.950627	403584.655696	881545.836795	2623852.396841	160751903.794874	159384255.939960	320136159.734835
162000.171875	29103.679379	407026.383120	887409.979274	2632162.958618	160748639.796219	159381809.520662	320130449.316881
163000.171875	29281.388355	409836.553917	893335.041745	2639418.101861	160746007.334451	159380479.439891	320126486.774342
164000.171875	29459.131775	413430.807167	899153.469707	2647959.655125	160742590.824689	159377756.314588	320120347.139277
165000.171875	29637.041214	417070.638732	904851.563761	2656652.014828	160739128.902537	159374762.048940	320113890.951478
166000.171875	29814.638311	420526.442401	910714.884830	2665043.607461	160735850.695951	159372233.777375	320108084.473326
167000.171875	29992.146072	423222.117817	916664.043374	2672220.382412	160733332.528343	159371006.160969	320104338.689311
168000.171875	30169.815790	426664.076523	922529.427532	2680591.393204	160730068.239354	159368500.534335	320098568.773688
169000.171875	30347.748100	430210.258204	928246.333174	2689235.031112	160726699.989987	159365573.802068	320092273.792055
170000.171875	30525.407073	433650.017417	934105.029933	2697707.221666	160723437.889723	159362960.308273	320086398.197996

171000.171875	30703.053951	436295.157682	940074.379135	2704888.682918	160720970.396337	159361748.196224	320082718.592561
172000.171875	30880.768836	439561.962853	945983.643237	2713035.977943	160717881.306068	159359510.165300	320077391.471369
173000.171875	31058.778829	443016.344843	951727.851827	2721578.584118	160714604.934051	159356711.767716	320071316.701767
174000.171875	31236.492722	446410.055113	957579.236663	2729929.641584	160711388.937598	159354212.095086	320065601.032685
175000.171875	31414.166577	449064.216367	963566.084316	2737101.039111	160708912.450234	159353027.545213	320061939.995447
176000.171875	31591.870996	452136.450234	969517.786895	2744872.189235	160706017.920793	159351208.097667	320057226.018460
177000.171875	31769.797348	455497.458943	975298.464118	2753217.009717	160702834.838383	159348643.954408	320051478.792791
178000.171875	31947.393768	458816.050404	981141.381961	2761518.059486	160699693.843321	159346185.822482	320045879.665802
179000.171875	32124.916979	461505.222735	987140.896338	2768836.520795	160697182.194173	159344866.875549	320042049.069722
180000.171875	32302.436065	464357.220070	993129.208447	2776344.493924	160694507.715948	159343347.214530	320037854.930479
181000.171875	32480.285372	467622.813859	998950.780572	2784646.984075	160691419.971338	159340866.296504	320032286.267842
182000.171875	32657.927216	470864.541928	1004784.799385	2792936.601697	160688355.885094	159338410.697695	320026766.582789
183000.171875	32835.505394	473617.059301	1010792.631599	2800437.591408	160685780.945845	159336917.540198	320022698.486043
184000.171875	33013.067946	476265.240197	1016812.714791	2807670.634160	160683310.327552	159335704.580638	320019014.908189
185000.171875	33190.935388	479445.926173	1022679.958578	2815908.099950	160680307.509011	159333334.358635	320013641.867646
186000.171875	33368.635958	482613.654827	1028508.094720	2824175.008536	160677317.480956	159330895.586191	320008213.067147
187000.171875	33546.222077	485426.595710	1034519.322215	2831870.212295	160674682.126174	159329211.609928	320003893.736101
188000.171875	33723.804204	487887.347935	1040567.036817	2838894.038435	160672398.956099	159328235.498389	320000634.454488
189000.171875	33901.590192	490959.182405	1046481.069722	2847069.495021	160669504.907635	159325974.074708	319995478.982343
190000.171875	34079.428312	494015.213170	1052308.409181	2855327.546131	160666626.714992	159323543.363057	319990170.078050
191000.171875	34257.110175	496827.369228	1058318.424005	2863196.083652	160663992.240762	159321684.840361	319985677.081123
192000.171875	34434.774435	499106.294465	1064390.218523	2870082.207204	160661890.979818	159320870.511326	319982761.491144
193000.171875	34612.435583	502037.996285	1070350.691956	2878186.710976	160659136.939101	159318726.480987	319977863.420089
194000.171875	34790.357616	505008.687857	1076184.754695	2886437.391339	160656344.169591	159316309.863363	319972654.032954
195000.171875	34967.973314	507834.839814	1082189.485038	2894431.837644	160653695.633351	159314320.147401	319968015.780752

196000.171875	35145.553454	510033.324321	1088281.279600	2901256.208203	160651674.729018	159313587.571404	319965262.300423
197000.171875	35323.185728	512854.658635	1094285.576515	2909229.608747	160649031.027018	159311618.467775	319960649.494794
198000.171875	35501.084395	515757.084583	1100134.704489	2917461.776486	160646306.499693	159309235.428009	319955541.927702
199000.171875	35678.712197	518577.022708	1106129.899517	2925584.558511	160643664.189399	159307107.841013	319950772.030412
200000.171875	35856.306975	520739.097149	1112238.006648	2932502.967196	160641679.709708	159306297.539458	319947977.249167
201000.171875	36033.906970	523423.092008	1118283.559035	2940376.123386	160639173.314871	159304469.935656	319943643.250527
202000.171875	36211.825093	526258.762813	1124157.070999	2948641.755670	160636515.562172	159302077.815337	319938593.377509
203000.171875	36389.479900	529048.659118	1130140.094475	2956857.196854	160633903.320664	159299845.397628	319933748.718293
204000.171875	36567.046111	531226.361517	1136261.041158	2963974.443784	160631903.184446	159298849.097381	319930752.281829
205000.171875	36744.585575	533757.427149	1142344.437390	2971713.300002	160629549.658272	159297193.637395	319926743.295668
206000.171875	36922.465057	536530.483341	1148250.566229	2979978.684970	160626954.481541	159294834.381265	319921788.862805
207000.171875	37100.041589	539274.906540	1154220.272451	2988094.603386	160624387.634814	159292688.169071	319917075.803886
208000.171875	37277.558245	541499.068548	1160350.499880	2995207.797189	160622340.989454	159291705.202698	319914046.192152
209000.171875	37455.089641	543862.859424	1166468.498023	3002485.219200	160620154.729995	159290545.778830	319910700.508824
210000.171875	37632.948910	546570.048402	1172414.942304	3010517.794847	160617625.400301	159288459.647463	319906085.047764
211000.171875	37810.606716	549259.488318	1178373.026788	3018545.237178	160615113.618113	159286390.289617	319901503.907730
212000.171875	37988.144410	551543.960524	1184508.893777	3025822.013153	160613006.683591	159285249.380631	319898256.064222
213000.171875	38165.696888	553741.539988	1190656.500724	3032839.570898	160610986.656554	159284379.429833	319895366.086388
214000.171875	38343.535267	556381.732184	1196646.643811	3040815.951013	160608524.302739	159282393.192806	319890917.495544
215000.171875	38521.398622	559011.869111	1202595.343892	3048836.043798	160606072.029142	159280321.800100	319886393.829244
216000.171875	38699.185830	561351.690056	1208731.687633	3056308.702724	160603909.995376	159278985.484917	319882895.480294
217000.171875	38876.958852	563398.343779	1214904.553562	3063107.743435	160602041.114683	159278359.310134	319880400.424817
218000.171875	39054.901714	565964.922241	1220939.685362	3071002.767374	160599652.479072	159276499.417995	319876151.897067
219000.171875	39232.829009	568534.680567	1226884.390183	3078965.876227	160597260.648087	159274481.013963	319871741.662051
220000.171875	39410.622413	570913.222973	1233016.867942	3086566.197263	160595059.899110	159273013.170686	319868073.069796

221000.171875	39588.388391	572842.119623	1239210.736207	3093202.557089	160593308.768448	159272570.679125	319865879.447574
222000.171875	39766.185952	575326.289649	1245289.829921	3101011.158339	160591002.395980	159270841.171589	319861843.567569
223000.171875	39944.234393	577837.531370	1251237.512392	3108947.784744	160588669.202768	159268852.227656	319857521.430424
224000.171875	40121.994191	580231.919186	1257360.413467	3116640.335585	160586452.574759	159267282.577889	319853735.152647
225000.171875	40299.724743	582086.857072	1263571.436587	3123189.585536	160584775.367402	159266944.351057	319851719.718459
226000.171875	40477.479024	584474.680718	1269692.845422	3130839.709771	160582565.298021	159265415.635657	319847980.933678
227000.171875	40655.488234	586929.514484	1275652.997695	3138719.103740	160580288.473448	159263496.393961	319843784.867409
228000.171875	40833.246568	589316.808671	1281762.495213	3146457.891064	160578078.937575	159261867.104154	319839946.041729
229000.171875	41011.000026	591143.990330	1287986.746469	3153026.969443	160576429.509361	159261522.277033	319837951.786394
230000.171875	41188.803622	593418.032403	1294146.954651	3160487.403695	160574333.270872	159260222.050963	319834555.321835
231000.171875	41366.809490	595817.965319	1300128.803647	3168317.269880	160572111.343864	159258374.033774	319830485.377637
232000.171875	41544.617168	598179.553489	1306222.451359	3176067.993036	160569927.563374	159256716.958329	319826644.521704
233000.171875	41722.343876	600019.806805	1312456.641411	3182739.233549	160568265.036717	159256279.907869	319824544.944586
234000.171875	41900.079377	602163.463956	1318651.659506	3189983.058402	160566299.115074	159255231.101109	319821530.216184
235000.171875	42078.039814	604509.216272	1324663.883709	3197758.874808	160564131.323230	159253467.508908	319817598.832138
236000.171875	42255.891500	606832.193978	1330740.021877	3205493.903789	160561986.197169	159251808.618096	319813794.815265
237000.171875	42433.637258	608713.284623	1336979.658532	3212327.458583	160560282.852274	159251214.699955	319811497.552229
238000.171875	42611.386870	610715.962195	1343205.196852	3219320.806634	160558457.924323	159250446.890224	319808904.814547
239000.171875	42789.349625	613008.239521	1349253.930583	3227036.254847	160556343.609768	159248780.175744	319805123.785512
240000.171875	42967.216806	615286.153012	1355313.530954	3234754.015368	160554243.563428	159247122.015594	319801365.579021
241000.171875	43145.021843	617222.306638	1361554.317432	3241778.748974	160552485.214838	159246338.068465	319798823.283302
242000.171875	43322.817414	619085.469408	1367806.490403	3248547.968550	160550799.847624	159245821.021860	319796620.869484
243000.171875	43500.795333	621324.943940	1373896.053549	3256226.937239	160548738.351026	159244231.616318	319792969.967344
244000.171875	43678.749552	623556.757031	1379942.036396	3263939.192921	160546684.492197	159242565.343482	319789249.835679
245000.171875	43856.573971	625544.836785	1386179.391904	3271149.790210	160544874.236866	159241592.101701	319786466.338567

246000.171875	44034.420590	627282.781307	1392453.444166	3277714.537061	160543314.138917	159241301.407112	319784615.546029
247000.171875	44212.404547	629465.248813	1398585.280999	3285351.798824	160541309.655339	159239795.982183	319781105.637521
248000.171875	44390.413440	631650.636148	1404623.538153	3293063.397211	160539302.276973	159238122.640949	319777424.917921
249000.171875	44568.284127	633675.900545	1410853.080651	3300438.561525	160537454.883287	159236977.019132	319774431.902420
250000.171875	44746.125932	635315.729559	1417145.549221	3306874.132034	160535992.896074	159236833.917194	319772826.813268
251000.171875	44924.020664	637431.121591	1423319.146454	3314456.742981	160534055.398793	159235424.903480	319769480.302273
252000.171875	45102.109564	639568.192753	1429357.674950	3322177.977208	160532096.416537	159233742.197749	319765838.614286
253000.171875	45279.925749	641607.057641	1435574.351332	3329631.679890	160530235.367832	159232505.171449	319762740.539281
254000.171875	45457.693633	643182.531979	1441881.453177	3335970.367059	160528837.661379	159232473.586125	319761311.247504
255000.171875	45635.514778	645215.079283	1448095.717271	3343389.381926	160526982.935247	159231268.835353	319758251.770599
256000.171875	45813.560665	647302.174810	1454145.239602	3351021.047313	160525073.885604	159229686.692296	319754760.577900
257000.171875	45991.419043	649332.502670	1460345.996501	3358533.311447	160523221.416134	159228375.185062	319751596.601195
258000.171875	46169.215684	650882.893228	1466664.533867	3364921.584115	160521848.822217	159228305.449759	319750154.271977
259000.171875	46347.044862	652816.570442	1472916.335002	3372199.171437	160520092.974180	159227279.663571	319747372.637752
260000.171875	46525.064101	654853.832888	1478985.321923	3379816.951497	160518233.730946	159225730.870433	319743964.601378
261000.171875	46702.880890	656858.644709	1485166.983904	3387371.211110	160516406.735912	159224358.272801	319740765.008713
262000.171875	46880.618166	658418.406537	1491492.515469	3393891.499587	160515024.711358	159224163.515889	319739188.227248
263000.156250	47058.402647	660238.708110	1497777.526534	3400991.211922	160513382.194278	159223348.814619	319736731.008897
264000.156250	47236.385140	662227.477140	1503875.076869	3408606.307899	160511571.407776	159221831.268977	319733402.676754
265000.156250	47414.262558	664196.260592	1510037.231820	3416198.079306	160509780.501754	159220401.652520	319730182.154274
266000.156250	47592.032897	665789.347571	1516366.039371	3422925.738848	160508365.185129	159220002.800530	319728367.985659
267000.156250	47769.731651	667486.321267	1522680.299682	3429835.907840	160506845.910195	159219406.891848	319726252.802044
268000.156250	47947.605769	669426.363649	1528813.029548	3437470.959375	160505083.741918	159217904.570179	319722988.312098
269000.156250	48125.385109	671347.527345	1534955.937738	3445116.752034	160503340.357530	159216401.685709	319719742.043240
270000.156250	48303.116363	672970.568529	1541283.546207	3452073.448346	160501895.047609	159215772.597866	319717667.645475

271000.156250	48480.837708	674531.337954	1547621.931006	3458766.953881	160500511.999538	159215417.477131	319715929.476669
272000.156250	48658.750124	676408.130275	1553793.532490	3466343.365614	160498813.119635	159214012.666881	319712825.786517
273000.156250	48836.637627	678277.425040	1559920.488242	3473940.438048	160497121.712402	159212542.550200	319709664.262602
274000.156250	49014.417127	679944.946043	1566242.694202	3481069.069111	160495631.970914	159211736.125098	319707368.096011
275000.156250	49192.195160	681402.026497	1572602.109647	3487558.318655	160494352.668511	159211606.291000	319705958.959510
276000.156250	49370.121268	683234.986629	1578815.233266	3495098.185460	160492697.634519	159210279.547811	319702977.182330
277000.156250	49548.071092	685071.365696	1584932.620737	3502708.862539	160491039.205259	159208786.258204	319699825.463463
278000.156250	49725.862618	686776.109583	1591244.338795	3510000.504877	160489512.252886	159207806.333923	319697318.586809
279000.156250	49903.628588	688154.504220	1597620.068050	3516356.429862	160488311.624207	159207826.138194	319696137.762401
280000.156250	50081.515802	689937.108889	1603873.467376	3523843.787003	160486706.906776	159206592.180379	319693299.087155
281000.156250	50259.453158	691738.568781	1609989.327326	3531463.232041	160485083.384253	159205088.595291	319690171.979544
282000.156250	50437.292419	693460.198950	1616285.736989	3538898.497790	160483539.593341	159203949.739205	319687489.332545
283000.156250	50615.115594	694788.521134	1622674.007492	3545224.164333	160482389.094321	159204012.343165	319686401.437486
284000.156250	50792.920402	696507.668638	1628965.940520	3552633.513221	160480847.751628	159202894.927305	319683742.678933
285000.156250	50970.979571	698273.455428	1635089.599733	3560255.436721	160479260.024021	159201396.663018	319680656.687040
286000.156250	51148.860042	699994.861393	1641366.576517	3567781.366725	160477716.498521	159200147.709798	319677864.208319
287000.156250	51326.660914	701308.366580	1647763.970915	3574172.607051	160476580.794166	159200153.863871	319676734.658037
288000.156250	51504.489594	702951.797066	1654092.296311	3581463.885118	160475115.192341	159199190.911198	319674306.103539
289000.156250	51682.513006	704682.194876	1660234.058747	3589110.845200	160473562.817944	159197685.713551	319671248.531495
290000.156250	51860.394916	706383.145449	1666489.938391	3596701.025286	160472039.749260	159196351.413110	319668391.162371
291000.156250	52038.189945	707699.924897	1672892.417754	3603240.338186	160470900.764867	159196214.579573	319667115.344441
292000.156250	52215.962401	709232.530017	1679253.062442	3610356.023517	160469545.932196	159195459.538931	319665005.471126
293000.156250	52393.931624	710898.477045	1685422.235399	3617987.111829	160468057.954421	159193997.623576	319662055.577996
294000.156250	52571.789316	712543.359546	1691656.649001	3625588.963479	160466590.929606	159192630.185527	319659221.115133
295000.156250	52749.504907	713872.011351	1698061.191550	3632315.425396	160465439.993413	159192308.266159	319657748.259572

296000.156250	52927.206336	715287.800379	1704450.072511	3639211.197989	160464201.905839	159191801.374527	319656003.280366
297000.156250	53105.100590	716904.858516	1710654.116459	3646812.883776	160462762.741950	159190403.732689	319653166.474639
298000.156250	53282.919315	718511.478037	1716867.786876	3654416.515579	160461333.941175	159189013.771302	319650347.712478
299000.156250	53460.654608	719878.269294	1723270.005227	3661341.163974	160460144.885197	159188491.341259	319648636.226456
300000.156250	53638.391373	721194.136855	1729682.910506	3667999.987256	160459006.754390	159188245.423256	319647252.177645
301000.156250	53816.303942	722776.746670	1735925.592441	3675554.593306	160457602.057127	159186933.499141	319644535.556268
302000.156250	53994.240722	724355.744529	1742122.249710	3683145.864671	160456200.996036	159185538.885045	319641739.881081
303000.156250	54172.047722	725766.391116	1748517.890075	3690266.124928	160454968.156415	159184814.265153	319639782.421568
304000.156250	54349.830243	726998.217884	1754950.774564	3696734.501248	160453914.112191	159184778.773321	319638692.885512
305000.156250	54527.752581	728549.125091	1761234.430908	3704262.905103	160452541.127333	159183534.025810	319636075.153144
306000.156250	54705.717676	730103.275966	1767420.336126	3711865.787353	160451164.941538	159182117.048779	319633281.990317
307000.156250	54883.557888	731547.663353	1773804.160742	3719164.287431	160449898.394363	159181202.373317	319631100.767679
308000.156250	55061.389198	732712.783323	1780252.977896	3725506.686474	160448911.105692	159181308.791427	319630219.897119
309000.156250	55239.328739	734220.396022	1786576.703130	3732989.290111	160447581.432502	159180149.913024	319627731.345527
310000.156250	55417.393956	735740.798986	1792760.021978	3740589.161751	160446239.094747	159178733.360233	319624972.454980
311000.156250	55595.300161	737194.491938	1799126.709005	3748000.755066	160444963.307990	159177688.453945	319622651.761934
312000.156250	55773.169832	738298.010885	1805586.894950	3754292.645989	160444037.658698	159177856.748967	319621894.407664
313000.156250	55951.096822	739732.796668	1811948.762119	3761671.749989	160442780.799915	159176839.512136	319619620.312052
314000.156250	56129.168748	741203.370557	1818139.194022	3769243.614619	160441488.297951	159175458.079407	319616946.377358
315000.156250	56307.099612	742641.248541	1824484.900804	3776703.810080	160440228.350805	159174343.590729	319614571.941534
316000.156250	56484.969232	743736.176999	1830952.837555	3782993.052755	160439311.291981	159174522.284804	319613833.576785
317000.156250	56662.870460	745114.512843	1837350.274894	3790188.617734	160438110.857355	159173724.157164	319611835.014519
318000.156250	56840.939027	746568.316222	1843558.145091	3797715.666554	160436835.122524	159172404.978542	319609240.101065
319000.156250	57018.896442	748005.091375	1849881.578390	3805188.125691	160435576.304813	159171255.952703	319606832.257517
320000.156250	57196.766292	749123.053105	1856354.066372	3811623.451723	160434636.212944	159171293.114654	319605929.327598

321000.156250	57374.658721	750436.675381	1862783.486469	3818635.659579	160433500.483109	159170710.326895	319604210.810003
322000.156250	57552.704870	751873.004477	1869017.962640	3826145.336212	160432242.200145	159169435.126433	319601677.326578
323000.156250	57730.645001	753300.294037	1875318.291586	3833627.588469	160430992.850708	159168253.203122	319599246.053829
324000.156250	57908.463003	754457.722148	1881791.220031	3840256.645758	160430013.240593	159168097.074278	319598110.314871
325000.156250	58086.288895	755696.683605	1888248.238306	3847074.468113	160428952.105008	159167736.270198	319596688.375205
326000.156250	58264.281178	757114.263524	1894517.150623	3854593.322447	160427712.517378	159166486.328181	319594198.845559
327000.156250	58442.232709	758526.503428	1900795.678706	3862110.448907	160426478.229017	159165247.729804	319591725.958821
328000.156250	58620.058745	759730.875511	1907265.477804	3868965.945950	160425451.682991	159164862.031859	319590313.714851
329000.156250	58797.874835	760892.373322	1913745.774509	3875558.114725	160424468.001279	159164750.159788	319589218.161068
330000.156250	58975.844671	762290.740049	1920053.006698	3883044.166519	160423247.604398	159163571.340185	319586818.944582
331000.156250	59153.812897	763687.377784	1926313.126206	3890555.236306	160422028.934884	159162320.389905	319584349.324789
332000.156250	59331.633115	764937.286658	1932774.625926	3897607.834414	160420956.846232	159161729.291517	319582686.137749
333000.156250	59509.428537	766029.066102	1939273.788622	3904008.386727	160420042.862195	159161827.901900	319581870.764094
334000.156250	59687.357400	767405.701086	1945620.947303	3911453.788750	160418844.156053	159160729.658558	319579573.814611
335000.156250	59865.307264	768786.739447	1951869.321360	3918964.023861	160417641.067563	159159467.797505	319577108.865067
336000.156250	60043.037337	770073.014708	1958317.718077	3926177.144798	160416532.522375	159158703.073284	319575235.595659
337000.156250	60220.742024	771111.518734	1964832.230814	3932439.392717	160415671.723011	159158955.338102	319574627.061114
338000.156250	60398.582706	772460.752534	1971219.688362	3939818.824207	160414500.329908	159157963.364161	319572463.694068
339000.156250	60576.480983	773825.981312	1977465.260255	3947306.063163	160413312.999407	159156721.697097	319570034.696504
340000.156250	60754.200192	775135.944335	1983896.217676	3954613.623089	160412180.755577	159155845.094592	319568025.850169
341000.156250	60931.861615	776144.574614	1990422.149780	3960819.246320	160411349.786728	159156165.403465	319567515.190193
342000.156250	61109.614427	777453.834316	1996848.192564	3968107.755592	160410218.279861	159155302.936977	319565521.216837
343000.156250	61287.429449	778796.618038	2003101.119719	3975592.319293	160409053.311157	159154071.300432	319563124.611589
344000.156250	61465.102496	780107.093771	2009510.996052	3982975.525190	160407920.508466	159153097.970867	319561018.479333
345000.156250	61642.721744	781103.546533	2016044.634730	3989241.978016	160407101.674952	159153365.156719	319560466.831671

346000.156250	61820.435685	782347.855743	2022506.152967	3996410.717105	160406035.079692	159152657.935866	319558693.015558
347000.156250	61998.300725	783650.536034	2028775.383915	4003904.127342	160404910.264469	159151433.756578	319556344.021047
348000.156250	62176.010319	784944.529517	2035161.174566	4011335.948140	160403793.980572	159150387.726431	319554181.707003
349000.156250	62353.577559	785950.033957	2041697.606002	4017743.621335	160402966.043367	159150516.484670	319553482.528037
350000.156250	62531.229836	787128.913152	2048190.152982	4024724.708834	160401964.816424	159150027.944151	319551992.760576
351000.156250	62709.004420	788423.421017	2054484.558904	4032186.647380	160400848.083152	159148860.411527	319549708.494679
352000.156250	62886.719034	789708.377530	2060844.984755	4039625.363999	160399740.841244	159147782.120760	319547522.962004
353000.156250	63064.335750	790751.992644	2067380.041989	4046212.955603	160398874.842855	159147729.586390	319546604.429246
354000.156250	63242.017461	791871.298500	2073898.621143	4052972.648528	160397933.218682	159147488.472619	319545421.691301
355000.156250	63419.807906	793150.712123	2080225.613343	4060408.840714	160396831.595507	159146379.272633	319543210.868140
356000.156250	63597.543150	794425.488951	2086561.792849	4067846.769908	160395734.553898	159145277.522945	319541012.076843
357000.156250	63775.212245	795504.230609	2093091.399630	4074646.800101	160394833.481331	159145007.099532	319539840.580864
358000.156250	63952.879461	796542.021721	2099631.566449	4081193.936232	160393973.357407	159145000.130221	319538973.487628
359000.156250	64130.753076	797782.870961	2105996.029161	4088618.509762	160392910.381789	159143940.019403	319536850.401191
360000.156250	64308.601136	799022.550491	2112312.649894	4096076.294259	160391848.550317	159142798.855639	319534647.405955
361000.156250	64486.310920	800135.361562	2118832.660134	4103087.187088	160390913.449030	159142307.973050	319533221.422080
362000.156250	64664.074989	801108.016162	2125391.328605	4109449.003450	160390118.558508	159142504.825159	319532623.383667
363000.156250	64841.998978	802334.147095	2131795.757727	4116850.572170	160389070.351560	159141507.685561	319530578.037119
364000.156250	65019.995714	803566.708705	2138100.067139	4124317.422122	160388015.786683	159140345.145021	319528360.931704
365000.156250	65197.795907	804717.559928	2144606.054361	4131500.841685	160387042.735643	159139667.712680	319526710.448322
366000.156250	65375.566823	805646.221577	2151179.312038	4137744.340499	160386291.844917	159139997.471544	319526289.316461
367000.156250	65553.434584	806854.276826	2157623.194668	4145115.277822	160385261.657446	159139070.416850	319524332.074296
368000.156250	65731.418733	808071.925722	2163923.387763	4152598.566276	160384221.992701	159137887.321490	319522109.314192
369000.156250	65909.266157	809239.287970	2170410.113389	4159906.022334	160383232.477866	159137066.591059	319520299.068925
370000.156250	66087.067067	810140.427824	2176993.538911	4166108.851326	160382509.138932	159137447.187589	319519956.326522

371000.156250	66264.934540	811313.372139	2183475.031159	4173384.165119	160381514.062090	159136653.366044	319518167.428134
372000.156250	66442.939447	812512.021064	2189781.139117	4180847.293898	160380493.418075	159135496.345223	319515989.763297
373000.156250	66620.811601	813683.397492	2196245.015882	4188218.822055	160379499.913799	159134588.693831	319514088.607630
374000.156250	66798.611795	814575.683734	2202834.522052	4194476.651334	160378785.427750	159134920.370723	319513705.798473
375000.156250	66976.466627	815699.932599	2209350.660261	4201630.328854	160377839.033733	159134282.831412	319512121.865144
376000.156250	67154.453803	816884.996964	2215672.622309	4209100.154932	160376831.956532	159133134.967382	319509966.923914
377000.156250	67332.352350	818058.622273	2222111.641925	4216518.581533	160375836.229747	159132155.560397	319507991.790145
378000.156250	67510.153026	818972.909714	2228703.332461	4222916.382533	160375099.742982	159132349.449933	319507449.192915
379000.156250	67687.977068	820050.128463	2235250.126709	4229889.369958	160374200.348269	159131923.256755	319506123.605025
380000.156250	67865.963399	821227.031611	2241596.712698	4237332.736789	160373201.431452	159130826.475913	319504027.907365
381000.156250	68043.888584	822397.006681	2248009.395839	4244753.468602	160372209.381562	159129818.427241	319502027.808803
382000.156250	68221.690349	823346.710304	2254598.766788	4251328.150611	160371437.479700	159129833.116182	319501270.595881
383000.156250	68399.500019	824365.051511	2261171.396121	4258072.479705	160370596.948178	159129661.416422	319500258.364600
384000.156250	68577.468064	825529.831369	2267550.266146	4265483.876761	160369610.136377	159128628.889391	319498239.025767
385000.156250	68755.376268	826691.183838	2273938.240511	4272894.979415	160368626.692100	159127605.761102	319496232.453202
386000.156250	68933.136873	827682.835941	2280521.822949	4279669.903362	160367812.800612	159127414.419593	319495227.220206
387000.156250	69110.890026	828639.267851	2287115.822927	4286188.087281	160367034.121850	159127490.235652	319494524.357501
388000.156250	69288.798688	829789.547387	2293532.003071	4293568.326942	160366061.750987	159126526.176134	319492587.927121
389000.156250	69466.699456	830938.097086	2299899.564283	4300978.515268	160365091.102041	159125483.549020	319490574.651061
390000.156250	69644.461891	831966.697924	2306472.159134	4307952.443364	160364240.263628	159125082.215774	319489322.479402
391000.156250	69822.192627	832863.969331	2313083.719064	4314284.848128	160363520.722953	159125361.370941	319488882.093895
392000.156250	70000.059906	833994.498843	2319539.171499	4321646.167404	160362568.060708	159124455.504099	319487023.564807
393000.156250	70177.979162	835127.561601	2325893.364329	4329071.555715	160361612.917191	159123384.308619	319484997.225809
394000.156250	70355.734701	836183.497674	2332450.789549	4336210.820289	160360734.736643	159122802.469263	319483537.205907
395000.156250	70533.457470	837034.544550	2339076.112897	4342417.843440	160360061.412543	159123220.769463	319483282.182005

396000.156250	70711.291726	838140.783144	2345570.683718	4349741.560776	160359133.008210	159122391.622947	319481524.631157
397000.156250	70889.223320	839259.153695	2351920.403514	4357171.102875	160358192.569254	159121311.800645	319479504.369898
398000.156250	71067.003058	840332.562612	2358457.934641	4364433.063813	160357296.940069	159120587.370833	319477884.310902
399000.156250	71244.730267	841158.438742	2365093.196545	4370601.370161	160356648.791153	159121054.326390	319477703.117543
400000.156250	71422.541003	842235.377387	2371625.220036	4377838.573349	160355749.663258	159120349.146693	319476098.809950
401000.156250	71600.474155	843341.100944	2377980.298347	4385257.529172	160354821.872837	159119285.269181	319474107.142018
402000.156250	71778.279053	844422.197672	2384494.328184	4392581.516265	160353918.580998	159118475.311926	319472393.892923
403000.156250	71956.005864	845244.384501	2391135.049266	4398802.032701	160353274.120998	159118895.516572	319472169.637570
404000.156250	72133.804051	846280.366720	2397701.232081	4405910.262640	160352415.936957	159118353.469447	319470769.406404
405000.156250	72311.719463	847369.795029	2404071.307605	4413324.456842	160351504.424066	159117309.350769	319468813.774834
406000.156250	72489.540318	848445.693576	2410559.361143	4420699.585230	160350606.346387	159116422.275919	319467028.622306
407000.156250	72667.251898	849281.577935	2417201.234667	4427070.443293	160349948.173607	159116693.291379	319466641.464987
408000.156250	72844.990957	850266.009160	2423797.596768	4434016.249109	160349141.481451	159116343.847665	319465485.329116
409000.156250	73022.907987	851339.801867	2430192.029438	4441432.715594	160348245.605768	159115321.813850	319463567.419617
410000.156250	73200.746049	852406.612955	2436653.191304	4448830.727134	160347356.632715	159114384.964177	319461741.596892
411000.156250	73378.464787	853271.947560	2443292.341865	4455392.603998	160346669.016849	159114462.237874	319461131.254722
412000.156250	73556.187264	854200.026484	2449914.337229	4462120.057345	160345918.660413	159114356.779892	319460275.440305
413000.156250	73734.085828	855260.180010	2456340.537340	4469514.990826	160345036.405416	159113388.046521	319458424.451936
414000.156250	73911.904074	856316.475675	2462775.806516	4476913.289667	160344157.927979	159112425.016856	319456582.944835
415000.156250	74089.582606	857218.163005	2469408.109288	4483696.013930	160343433.919189	159112274.595364	319455708.514553
416000.156250	74267.281768	858088.173990	2476050.877924	4490226.357565	160342741.607355	159112387.020365	319455128.627721
417000.156250	74445.148228	859134.142697	2482513.780583	4497627.568060	160341873.505105	159111448.712529	319453322.217635
418000.156250	74623.007340	860178.750473	2488927.743773	4505058.394256	160341006.756443	159110431.849524	319451438.605966
419000.156250	74800.732224	861114.725724	2495548.267856	4512046.709099	160340248.506067	159110064.058763	319450312.564830
420000.156250	74978.425287	861931.405942	2502207.718899	4518347.058258	160339609.518891	159110423.160648	319450032.679538

421000.156250	75156.275832	862960.818654	2508709.479166	4525656.600671	160338757.956731	159109615.378500	319448373.335232
422000.156250	75334.184035	863992.803994	2515109.404522	4533016.485858	160337903.879579	159108655.418670	319446559.298249
423000.156250	75511.927420	864928.532375	2521713.701950	4540097.859634	160337145.894592	159108178.342322	319445324.236915
424000.156250	75689.638680	865669.699516	2528386.491787	4546249.429185	160336582.438728	159108699.562608	319445282.001336
425000.156250	75867.464634	866651.582194	2534927.188870	4553480.776449	160335778.382008	159108008.912427	319443787.294435
426000.156250	76045.383673	867648.909100	2541322.030295	4560817.689108	160334958.974168	159107066.841194	319442025.815361
427000.156250	76223.147965	868599.088256	2547905.592362	4568002.950003	160334186.559315	159106465.142365	319440651.701680
428000.156250	76400.860675	869336.183283	2554587.488228	4574119.411618	160333627.176991	159107030.576615	319440657.753607
429000.156250	76578.662456	870294.844140	2561165.090622	4581281.739910	160332846.317917	159106445.850717	319439292.168634
430000.156250	76756.584301	871282.585500	2567564.699532	4588623.264410	160332036.498426	159105503.935128	319437540.433554
431000.156250	76934.418437	872249.467728	2574123.973823	4595882.944239	160331247.450331	159104803.529589	319436050.979920
432000.156250	77112.167319	872985.914488	2580811.209864	4602062.998371	160330688.752460	159105310.711497	319435999.463957
433000.156250	77289.956720	873916.409595	2587422.881746	4609124.869399	160329936.046763	159104860.512351	319434796.559114
434000.156250	77467.859932	874896.033462	2593837.378919	4616487.777698	160329134.326111	159103912.101225	319433046.427336
435000.156250	77645.678357	875865.344652	2600370.117217	4623806.267331	160328342.833363	159103126.349890	319431469.183252
436000.156250	77823.393134	876619.029044	2607057.973449	4630132.929479	160327766.863749	159103487.543975	319431254.407724
437000.156250	78001.174682	877508.928064	2613699.719737	4637024.753311	160327054.746260	159103237.466430	319430292.212691
438000.156250	78179.056685	878479.894287	2620138.191804	4644372.729264	160326261.662045	159102327.962544	319428589.624589

GRID_430000.asc

```
ncols                21

nrows               250

xllcorner             0

yllcorner             0

cellsize         50.000

nodata_value      -9999

      25.500      24.724      23.891      22.988      21.745      21.341      20.048      20.084      18.800      19.116      18.596      19.082      18.800      20.068      20.058      21.357
21.777      22.996      23.894      24.726      25.500

      25.600      24.519      23.817      23.035      22.100      21.449      20.208      20.075      18.909      19.118      18.648      19.088      18.904      20.056      20.221      21.463
22.122      23.039      23.808      24.510      25.600

      25.700      24.540      23.670      23.008      22.205      21.468      20.259      20.006      18.918      19.094      18.659      19.067      18.915      19.989      20.270      21.482
22.227      23.001      23.658      24.529      25.700

      25.800      24.601      23.661      23.004      22.300      21.518      20.344      19.967      18.940      19.096      18.694      19.071      18.940      19.950      20.349      21.531
22.314      22.987      23.649      24.591      25.800

      25.900      24.636      23.668      22.989      22.346      21.565      20.401      19.940      18.953      19.108      18.736      19.085      18.957      19.923      20.411      21.578
22.352      22.979      23.656      24.629      25.900

      26.000      24.682      23.707      23.008      22.381      21.629      20.449      19.925      18.964      19.129      18.785      19.107      18.973      19.908      20.462      21.641
22.382      22.997      23.693      24.673      26.000

      26.000      24.721      23.742      23.016      22.416      21.686      20.495      19.914      18.971      19.152      18.836      19.132      18.987      19.898      20.509      21.699
22.418      23.007      23.726      24.710      26.000

      26.000      24.766      23.800      23.044      22.444      21.744      20.557      19.904      18.977      19.174      18.886      19.156      18.997      19.892      20.567      21.755
22.445      23.034      23.776      24.756      26.000

      26.000      24.805      23.829      23.054      22.448      21.779      20.612      19.895      18.982      19.200      18.934      19.181      18.999      19.889      20.627      21.787
22.448      23.043      23.817      24.802      26.000

      26.000      24.845      23.868      23.091      22.473      21.823      20.685      19.885      18.988      19.222      18.977      19.206      19.031      19.889      20.694      21.828
22.472      23.079      23.856      24.834      26.000

      26.000      24.873      23.902      23.115      22.477      21.846      20.760      19.875      18.996      19.241      19.018      19.227      19.047      19.889      20.766      21.850
22.476      23.101      23.889      24.865      26.000
```

```
      26.000      24.904      23.948      23.162      22.493      21.870      20.828      19.865      19.036      19.258      19.053      19.245      19.068      19.889      20.836      21.872
22.491      23.147      23.934      24.899      26.000

      26.000      24.934      24.002      23.211      22.508      21.890      20.900      19.852      19.096      19.270      19.080      19.258      19.099      19.885      20.904      21.890
22.507      23.190      23.984      24.929      26.000

      26.000      24.967      24.038      23.253      22.538      21.916      20.968      19.838      19.143      19.281      19.102      19.268      19.134      19.877      20.973      21.915
22.539      23.240      24.028      24.960      26.000

      26.000      24.995      24.058      23.278      22.555      21.925      21.016      19.828      19.186      19.294      19.126      19.281      19.169      19.869      21.026      21.925
22.556      23.265      24.047      24.989      26.000

      26.000      25.033      24.101      23.315      22.603      21.958      21.075      19.820      19.234      19.311      19.151      19.297      19.195      19.859      21.084      21.958
22.601      23.301      24.091      25.022      26.000

      26.000      25.063      24.127      23.350      22.625      21.973      21.107      19.815      19.289      19.327      19.175      19.314      19.246      19.850      21.115      21.976
22.621      23.333      24.117      25.052      26.000

      26.000      25.103      24.175      23.402      22.667      22.000      21.141      19.816      19.344      19.342      19.200      19.329      19.303      19.846      21.148      22.006
22.656      23.374      24.163      25.086      26.000

      26.000      25.138      24.212      23.437      22.696      22.006      21.181      19.822      19.389      19.357      19.223      19.344      19.357      19.847      21.184      22.015
22.679      23.426      24.203      25.118      26.000

      26.000      25.191      24.251      23.464      22.752      22.024      21.232      19.833      19.400      19.375      19.250      19.362      19.394      19.854      21.237      22.032
22.725      23.454      24.242      25.157      26.000

      26.000      25.218      24.283      23.473      22.806      22.022      21.242      19.852      19.410      19.408      19.285      19.391      19.400      19.868      21.249      22.029
22.764      23.464      24.275      25.187      26.000

      26.000      25.264      24.333      23.510      22.882      22.055      21.273      19.872      19.468      19.438      19.318      19.422      19.410      19.884      21.277      22.057
22.846      23.498      24.324      25.235      26.000

      26.000      25.286      24.373      23.528      22.938      22.081      21.287      19.890      19.517      19.456      19.343      19.441      19.456      19.901      21.292      22.080
22.920      23.515      24.365      25.261      26.000

      26.000      25.325      24.421      23.561      22.981      22.132      21.316      19.913      19.563      19.466      19.359      19.453      19.492      19.926      21.319      22.130
22.971      23.545      24.416      25.302      26.000

      26.000      25.400      24.450      23.586      23.012      22.173      21.360      19.944      19.600      19.469      19.367      19.458      19.521      19.961      21.360      22.171
23.006      23.569      24.445      25.400      26.000

      26.000      25.500      24.497      23.622      23.036      22.230      21.404      19.981      19.601      19.471      19.373      19.463      19.550      19.994      21.402      22.224
23.032      23.613      24.491      25.500      26.000

      26.000      25.600      24.526      23.628      23.035      22.267      21.413      20.011      19.604      19.482      19.386      19.475      19.587      20.025      21.412      22.264
23.032      23.621      24.520      25.600      26.000
```

```
      26.000      25.700      24.573      23.658      23.054      22.336      21.450      20.043      19.606      19.497      19.408      19.491      19.600      20.060      21.445      22.338
23.050      23.650      24.564      25.700      26.000

      26.000      25.800      24.614      23.671      23.054      22.391      21.469      20.077      19.608      19.515      19.432      19.510      19.601      20.100      21.467      22.397
23.051      23.664      24.604      25.800      26.000

      26.000      25.900      24.650      23.706      23.068      22.444      21.498      20.121      19.609      19.535      19.458      19.531      19.601      20.149      21.500      22.449
23.064      23.697      24.635      25.900      26.000

      26.000      26.000      24.693      23.733      23.071      22.483      21.540      20.172      19.607      19.555      19.483      19.550      19.600      20.192      21.538      22.486
23.067      23.722      24.674      26.000      26.000

      26.000      26.000      24.742      23.783      23.092      22.526      21.583      20.206      19.603      19.578      19.512      19.574      19.600      20.223      21.582      22.528
23.085      23.766      24.721      26.000      26.000

      26.000      26.000      24.779      23.815      23.089      22.540      21.616      20.236      19.600      19.610      19.543      19.605      19.620      20.250      21.614      22.540
23.084      23.804      24.762      26.000      26.000

      26.000      26.000      24.864      23.852      23.116      22.582      21.673      20.264      19.606      19.636      19.569      19.630      19.655      20.275      21.660      22.580
23.109      23.840      24.807      26.000      26.000

      26.000      26.000      24.876      23.882      23.126      22.599      21.707      20.300      19.646      19.651      19.586      19.644      19.687      20.307      21.687      22.597
23.117      23.866      24.839      26.000      26.000

      26.000      26.000      24.920      23.929      23.162      22.627      21.752      20.353      19.678      19.654      19.595      19.646      19.716      20.354      21.725      22.625
23.149      23.910      24.876      26.000      26.000

      26.000      26.000      24.941      23.988      23.202      22.634      21.823      20.398      19.709      19.649      19.599      19.643      19.744      20.396      21.790      22.633
23.178      23.964      24.904      26.000      26.000

      26.000      26.000      24.988      24.040      23.236      22.650      21.890      20.436      19.743      19.641      19.607      19.638      19.775      20.426      21.874      22.650
23.227      24.027      24.939      26.000      26.000

      26.000      26.000      25.005      24.066      23.250      22.648      21.917      20.482      19.785      19.634      19.622      19.631      19.800      20.458      21.911      22.650
23.244      24.054      24.966      26.000      26.000

      26.000      26.000      25.057      24.112      23.284      22.661      21.971      20.561      19.816      19.623      19.646      19.614      19.824      20.503      21.963      22.662
23.276      24.100      25.003      26.000      26.000

      26.000      26.000      25.078      24.137      23.304      22.657      21.991      20.656      19.837      19.606      19.674      19.601      19.839      20.576      21.986      22.659
23.298      24.126      25.031      26.000      26.000

      26.000      26.000      25.123      24.179      23.349      22.671      22.027      20.767      19.839      19.600      19.702      19.600      19.846      20.631      22.018      22.671
23.340      24.167      25.069      26.000      26.000

      26.000      26.000      25.149      24.207      23.407      22.674      22.044      20.907      19.817      19.627      19.715      19.626      19.833      20.701      22.036      22.674
23.385      24.200      25.099      26.000      26.000
```

```
      26.000      26.000      25.196      24.236      23.450      22.694      22.071      20.992      19.794      19.650      19.717      19.657      19.808      20.795      22.060      22.693
23.438      24.228      25.144      26.000      26.000

      26.000      26.000      25.216      24.258      23.478      22.693      22.071      21.021      19.785      19.670      19.722      19.687      19.796      20.852      22.062      22.696
23.469      24.248      25.167      26.000      26.000

      26.000      26.000      25.265      24.299      23.528      22.725      22.088      21.134      19.782      19.683      19.724      19.712      19.792      20.933      22.078      22.726
23.519      24.287      25.221      26.000      26.000

      26.000      26.000      25.286      24.331      23.560      22.747      22.083      21.193      19.788      19.691      19.726      19.734      19.798      20.990      22.076      22.747
23.552      24.317      25.245      26.000      26.000

      26.000      26.000      25.327      24.377      23.612      22.822      22.086      21.152      19.807      19.710      19.740      19.766      19.816      21.019      22.079      22.808
23.604      24.362      25.289      26.000      26.000

      26.000      26.000      25.352      24.417      23.627      22.907      22.110      21.246      19.826      19.726      19.752      19.794      19.835      21.071      22.096      22.870
23.621      24.410      25.317      26.000      26.000

      26.000      26.000      25.400      24.462      23.648      23.027      22.137      21.306      19.834      19.739      19.758      19.800      19.852      21.108      22.131      23.013
23.643      24.454      25.400      26.000      26.000

      26.000      26.000      25.500      24.495      23.649      23.059      22.170      21.260      19.857      19.771      19.773      19.810      19.879      21.127      22.165      23.055
23.645      24.490      25.500      26.000      26.000

      26.000      26.000      25.600      24.548      23.675      23.093      22.248      21.355      19.878      19.800      19.783      19.816      19.906      21.178      22.225      23.090
23.670      24.544      25.600      26.000      26.000

      26.000      26.000      25.700      24.590      23.686      23.099      22.277      21.398      19.898      19.800      19.800      19.822      19.934      21.216      22.252      23.098
23.681      24.580      25.700      26.000      26.000

      26.000      26.000      25.800      24.626      23.711      23.116      22.315      21.376      19.938      19.807      19.828      19.837      19.975      21.228      22.288      23.114
23.707      24.619      25.800      26.000      26.000

      26.000      26.000      25.900      24.658      23.735      23.122      22.361      21.451      19.980      19.837      19.851      19.843      20.005      21.253      22.326      23.121
23.729      24.646      25.900      26.000      26.000

      26.000      26.000      26.000      24.706      23.780      23.144      22.427      21.489      20.009      19.867      19.871      19.840      20.025      21.281      22.376      23.140
23.768      24.690      26.000      26.000      26.000

      26.000      26.000      26.000      24.743      23.808      23.146      22.472      21.473      20.039      19.909      19.898      19.835      20.045      21.304      22.440      23.140
23.800      24.732      26.000      26.000      26.000

      26.000      26.000      26.000      24.864      23.840      23.169      22.557      21.533      20.063      19.959      19.924      19.820      20.060      21.356      22.539      23.159
23.829      24.802      26.000      26.000      26.000

      26.000      26.000      26.000      24.872      23.856      23.190      22.600      21.595      20.081      20.001      19.950      19.804      20.074      21.410      22.592      23.170
23.846      24.842      26.000      26.000      26.000
```

```
     26.000      26.000      26.000      24.922      23.898      23.220      22.652      21.604      20.105      20.042      19.977      19.800      20.093      21.436      22.649      23.212
23.885      24.900      26.000      26.000      26.000

     26.000      26.000      26.000      24.940      23.941      23.229      22.676      21.675      20.124      20.072      20.009      19.826      20.110      21.486      22.676      23.223
23.926      24.917      26.000      26.000      26.000

     26.000      26.000      26.000      24.992      24.011      23.252      22.700      21.734      20.137      20.091      20.031      19.865      20.122      21.531      22.699      23.245
23.996      24.972      26.000      26.000      26.000

     26.000      26.000      26.000      25.003      24.044      23.251      22.691      21.726      20.158      20.108      20.053      19.906      20.140      21.557      22.690      23.245
24.034      24.984      26.000      26.000      26.000

     26.000      26.000      26.000      25.060      24.095      23.285      22.714      21.813      20.183      20.121      20.069      19.945      20.162      21.609      22.712      23.276
24.086      25.041      26.000      26.000      26.000

     26.000      26.000      26.000      25.076      24.131      23.294      22.707      21.867      20.204      20.128      20.081      19.983      20.190      21.649      22.704      23.286
24.124      25.058      26.000      26.000      26.000

     26.000      26.000      26.000      25.118      24.180      23.331      22.716      21.853      20.229      20.138      20.098      20.009      20.217      21.668      22.712      23.319
24.171      25.102      26.000      26.000      26.000

     26.000      26.000      26.000      25.140      24.211      23.360      22.726      21.933      20.252      20.149      20.116      20.042      20.243      21.697      22.719      23.350
24.207      25.127      26.000      26.000      26.000

     26.000      26.000      26.000      25.185      24.234      23.428      22.745      21.976      20.270      20.159      20.134      20.077      20.264      21.735      22.733      23.415
24.230      25.173      26.000      26.000      26.000

     26.000      26.000      26.000      25.202      24.248      23.451      22.753      21.951      20.288      20.168      20.154      20.122      20.285      21.761      22.724      23.440
24.244      25.191      26.000      26.000      26.000

     26.000      26.000      26.000      25.255      24.281      23.497      22.803      22.042      20.301      20.171      20.171      20.176      20.304      21.826      22.748      23.486
24.275      25.244      26.000      26.000      26.000

     26.000      26.000      26.000      25.272      24.301      23.534      22.806      22.088      20.312      20.165      20.189      20.222      20.326      21.883      22.760      23.521
24.294      25.263      26.000      26.000      26.000

     26.000      26.000      26.000      25.319      24.345      23.597      22.840      22.068      20.325      20.156      20.214      20.264      20.357      21.913      22.817      23.571
24.336      25.311      26.000      26.000      26.000

     26.000      26.000      26.000      25.343      24.390      23.632      22.869      22.154      20.343      20.139      20.240      20.300      20.394      21.962      22.848      23.624
24.376      25.335      26.000      26.000      26.000

     26.000      26.000      26.000      25.391      24.429      23.665      22.914      22.254      20.371      20.114      20.267      20.330      20.425      22.021      22.892      23.658
24.421      25.385      26.000      26.000      26.000

     26.000      26.000      26.000      25.411      24.459      23.670      22.943      22.196      20.400      20.086      20.296      20.359      20.457      22.025      22.923      23.664
24.453      25.406      26.000      26.000      26.000
```

```
     26.000      26.000      26.000      25.500      24.508      23.697      23.041      22.377      20.427      20.064      20.323      20.387      20.492      22.052      23.008      23.689
24.505      25.500      26.000      26.000      26.000

     26.000      26.000      26.000      25.600      24.546      23.706      23.093      22.446      20.445      20.049      20.340      20.411      20.545      22.087      23.070      23.699
24.545      25.600      26.000      26.000      26.000

     26.000      26.000      26.000      25.700      24.609      23.727      23.143      22.404      20.461      20.033      20.338      20.407      20.637      22.135      23.126      23.720
24.610      25.700      26.000      26.000      26.000

     26.000      26.000      26.000      25.800      24.644      23.750      23.168      22.538      20.517      19.981      20.301      20.340      20.836      22.214      23.153      23.743
24.640      25.800      26.000      26.000      26.000

     26.000      26.000      26.000      25.900      24.832      23.807      23.209      22.548      20.711      19.835      20.207      20.237      21.171      22.286      23.191      23.797
24.799      25.900      26.000      26.000      26.000

     26.000      26.000      26.000      26.000      24.804      23.815      23.209      22.479      21.077      19.698      20.095      20.131      21.399      22.319      23.202      23.808
24.718      26.000      26.000      26.000      26.000

     26.000      26.000      26.000      26.000      24.885      23.850      23.241      22.562      21.441      19.615      20.014      20.052      21.555      22.381      23.232      23.841
24.874      26.000      26.000      26.000      26.000

     26.000      26.000      26.000      26.000      24.878      23.861      23.234      22.592      21.639      19.614      19.983      20.013      21.658      22.451      23.229      23.852
24.864      26.000      26.000      26.000      26.000

     26.000      26.000      26.000      26.000      24.938      23.898      23.254      22.547      21.708      19.654      19.981      19.992      21.720      22.473      23.247      23.887
24.926      26.000      26.000      26.000      26.000

     26.000      26.000      26.000      26.000      24.945      23.928      23.264      22.637      21.773      19.707      19.995      19.976      21.772      22.546      23.257      23.916
24.930      26.000      26.000      26.000      26.000

     26.000      26.000      26.000      26.000      25.008      24.001      23.298      22.668      21.827      19.766      20.004      19.965      21.810      22.597      23.285      23.975
24.997      26.000      26.000      26.000      26.000

     26.000      26.000      26.000      26.000      25.010      24.024      23.293      22.653      21.876      19.824      20.011      19.960      21.856      22.634      23.283      24.013
24.997      26.000      26.000      26.000      26.000

     26.000      26.000      26.000      26.000      25.074      24.065      23.329      22.707      21.923      19.889      20.013      19.959      21.894      22.690      23.318      24.054
25.065      26.000      26.000      26.000      26.000

     26.000      26.000      26.000      26.000      25.081      24.097      23.344      22.768      21.986      19.948      20.022      19.954      21.951      22.721      23.329      24.086
25.071      26.000      26.000      26.000      26.000

     26.000      26.000      26.000      26.000      25.130      24.146      23.398      22.733      22.054      19.992      20.036      19.946      22.024      22.736      23.361      24.138
25.123      26.000      26.000      26.000      26.000

     26.000      26.000      26.000      26.000      25.140      24.203      23.424      22.920      22.096      20.027      20.059      19.940      22.083      22.775      23.410      24.197
25.131      26.000      26.000      26.000      26.000
```

```
      26.000      26.000      26.000      26.000      25.196      24.236      23.453      22.947      22.148      20.059      20.091      19.933      22.140      22.819      23.441      24.233
25.189      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      25.198      24.249      23.462      22.885      22.214      20.092      20.132      19.922      22.195      22.820      23.451      24.245
25.189      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      25.260      24.282      23.507      22.991      22.261      20.132      20.175      19.908      22.240      22.873      23.489      24.277
25.253      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      25.266      24.297      23.522      23.001      22.318      20.179      20.203      19.893      22.291      22.884      23.504      24.292
25.259      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      25.320      24.335      23.560      22.954      22.379      20.215      20.223      19.884      22.353      22.888      23.545      24.329
25.315      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      25.337      24.358      23.618      23.026      22.467      20.249      20.237      19.884      22.420      22.937      23.601      24.352
25.330      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      25.388      24.416      23.651      23.032      22.554      20.279      20.247      19.893      22.482      22.976      23.639      24.409
25.384      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      25.407      24.438      23.664      22.974      22.566      20.305      20.255      19.912      22.536      22.962      23.654      24.431
25.403      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      25.454      24.482      23.702      23.055      22.599      20.326      20.261      19.940      22.592      23.019      23.690      24.476
25.451      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      25.500      24.518      23.718      23.090      22.657      20.343      20.264      19.978      22.655      23.041      23.708      24.514
25.500      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      25.600      24.601      23.748      23.067      22.677      20.354      20.269      20.011      22.666      23.045      23.737      24.600
25.600      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      25.700      24.826      23.806      23.133      22.716      20.364      20.278      20.052      22.705      23.081      23.784      24.817
25.700      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      25.800      24.855      23.833      23.160      22.750      20.368      20.287      20.097      22.748      23.110      23.824      24.848
25.800      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      25.900      24.862      23.840      23.148      22.786      20.365      20.301      20.144      22.779      23.140      23.832      24.856
25.900      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      24.894      23.879      23.237      22.838      20.360      20.325      20.186      22.828      23.219      23.868      24.887
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      24.912      23.893      23.263      22.863      20.353      20.364      20.212      22.855      23.248      23.882      24.906
26.000      26.000      26.000      26.000      26.000
```

```
      26.000      26.000      26.000      26.000      26.000      24.935      23.926      23.250      22.859      20.346      20.399      20.232      22.853      23.251      23.913      24.928
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      24.974      23.976      23.296      22.869      20.345      20.431      20.255      22.876      23.281      23.953      24.968
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      25.004      24.024      23.337      22.883      20.345      20.457      20.283      22.897      23.308      24.012      24.999
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      25.029      24.032      23.320      22.899      20.345      20.475      20.319      22.909      23.306      24.025      25.024
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      25.066      24.075      23.421      22.931      20.345      20.485      20.366      22.941      23.354      24.062      25.062
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      25.095      24.095      23.458      22.966      20.350      20.491      20.400      22.984      23.412      24.082      25.090
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      25.120      24.129      23.432      23.007      20.364      20.497      20.430      23.011      23.420      24.118      25.115
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      25.157      24.197      23.513      23.042      20.390      20.502      20.455      23.046      23.450      24.170      25.153
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      25.185      24.222      23.560      23.072      20.412      20.505      20.477      23.075      23.477      24.216      25.180
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      25.210      24.243      23.506      23.092      20.434      20.511      20.498      23.085      23.470      24.238      25.206
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      25.249      24.286      23.602      23.135      20.457      20.517      20.523      23.139      23.513      24.279      25.246
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      25.278      24.306      23.632      23.173      20.478      20.522      20.553      23.183      23.535      24.301      25.274
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      25.304      24.339      23.613      23.185      20.496      20.526      20.587      23.202      23.549      24.335      25.300
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      25.344      24.405      23.663      23.230      20.517      20.532      20.611      23.257      23.624      24.393      25.342
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      25.372      24.425      23.703      23.271      20.545      20.539      20.632      23.302      23.655      24.418      25.370
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      25.400      24.437      23.677      23.287      20.578      20.548      20.645      23.308      23.654      24.430      25.396
26.000      26.000      26.000      26.000      26.000
```

```
      26.000      26.000      26.000      26.000      26.000      25.435      24.491      23.734      23.318      20.603      20.562      20.657      23.333      23.683      24.477      25.433
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      25.467      24.540      23.812      23.353      20.623      20.586      20.667      23.369      23.706      24.510      25.464
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      25.500      24.624      23.806      23.381      20.640      20.606      20.677      23.386      23.719      24.543      25.500
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      25.600      24.804      23.851      23.431      20.657      20.628      20.688      23.435      23.809      24.741      25.600
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      25.700      24.852      23.924      23.462      20.672      20.654      20.698      23.466      23.842      24.844      25.700
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      25.800      24.866      23.877      23.478      20.684      20.687      20.705      23.474      23.846      24.857      25.800
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      25.900      24.882      23.996      23.501      20.690      20.733      20.709      23.500      23.880      24.876      25.900
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      24.902      24.049      23.524      20.693      20.785      20.710      23.524      23.905      24.898      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      24.923      24.009      23.544      20.697      20.826      20.713      23.541      23.925      24.917      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      24.955      24.093      23.579      20.704      20.861      20.719      23.578      24.004      24.949      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      24.986      24.132      23.630      20.715      20.887      20.730      23.626      24.037      24.980      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      25.013      24.097      23.647      20.730      20.908      20.742      23.642      24.037      25.007      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      25.046      24.182      23.678      20.751      20.920      20.756      23.667      24.071      25.040      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      25.076      24.231      23.711      20.779      20.926      20.770      23.696      24.102      25.072      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      25.104      24.201      23.734      20.802      20.928      20.786      23.709      24.120      25.098      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      25.137      24.295      23.770      20.826      20.930      20.801      23.737      24.172      25.130      26.000
26.000      26.000      26.000      26.000      26.000
```

```
      26.000      26.000      26.000      26.000      26.000      26.000      25.168      24.383      23.817      20.849      20.933      20.815      23.765      24.223      25.162      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      25.196      24.357      23.854      20.873      20.939      20.827      23.803      24.231      25.189      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      25.228      24.447      23.891      20.899      20.945      20.838      23.823      24.259      25.221      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      25.259      24.510      23.923      20.933      20.954      20.847      23.834      24.296      25.253      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      25.288      24.480      23.951      20.981      20.967      20.855      23.845      24.319      25.281      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      25.319      24.571      23.984      21.022      20.990      20.862      23.857      24.364      25.313      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      25.349      24.658      24.009      21.063      21.009      20.868      23.870      24.418      25.345      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      25.379      24.630      24.019      21.030      21.058      20.876      23.886      24.431      25.374      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      25.410      24.718      24.012      20.947      21.260      20.885      23.917      24.462      25.405      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      25.440      24.773      24.009      20.826      22.266      20.869      23.974      24.499      25.436      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      25.470      24.785      24.027      20.843      22.471      20.885      23.991      24.532      25.466      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      25.500      24.815      24.047      20.860      22.568      20.893      24.005      24.674      25.500      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      25.600      24.835      24.072      20.878      22.609      20.903      24.021      24.834      25.600      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      25.700      24.856      24.109      20.895      22.471      20.914      24.042      24.856      25.700      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      25.800      24.870      24.137      20.916      20.926      20.920      24.065      24.870      25.800      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      25.900      24.890      24.388      22.853      20.370      22.811      24.388      24.890      25.900      26.000
26.000      26.000      26.000      26.000      26.000
```

```
      26.000      26.000      26.000      26.000      26.000      26.000      26.000      24.907      24.411      23.242      20.370      23.248      24.411      24.907      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      24.930      24.431      23.330      20.389      23.326      24.431      24.930      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      24.960      24.452      23.371      20.402      23.371      24.452      24.960      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      24.990      24.473      23.423      20.416      23.423      24.473      24.990      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.020      24.494      23.463      20.432      23.463      24.494      25.020      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.050      24.515      23.494      20.448      23.494      24.515      25.050      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.080      24.536      23.523      20.463      23.523      24.536      25.080      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.110      24.557      23.556      20.479      23.556      24.557      25.110      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.140      24.577      23.600      20.494      23.600      24.577      25.140      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.170      24.597      23.638      20.510      23.638      24.597      25.170      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.200      24.618      23.661      20.528      23.661      24.618      25.200      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.230      24.636      23.679      20.546      23.679      24.636      25.230      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.260      24.655      23.699      20.566      23.699      24.655      25.260      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.290      24.676      23.719      20.585      23.719      24.676      25.290      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.320      24.696      23.743      20.599      23.743      24.696      25.320      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.350      24.716      23.770      20.615      23.770      24.716      25.350      26.000      26.000
26.000      26.000      26.000      26.000      26.000
```

```
      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.380      24.738      23.809      20.630      23.809      24.738      25.380      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.410      24.759      23.840      20.648      23.840      24.759      25.410      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.440      24.777      23.862      20.665      23.862      24.777      25.440      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.470      24.796      23.882      20.683      23.882      24.796      25.470      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.500      24.815      23.898      20.701      23.898      24.815      25.500      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.600      24.834      23.914      20.720      23.914      24.834      25.600      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.700      24.851      23.932      20.739      23.932      24.851      25.700      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.800      24.869      23.951      20.758      23.951      24.869      25.800      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.900      24.892      23.971      20.777      23.970      24.892      25.900      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      24.907      23.997      20.796      23.997      24.907      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      24.930      24.016      20.811      24.016      24.930      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      24.960      24.032      20.827      24.032      24.960      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      24.990      24.045      20.841      24.045      24.990      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.020      24.057      20.856      24.057      25.020      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.050      24.069      20.869      24.069      25.050      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.080      24.083      20.882      24.083      25.080      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000
```

```
      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.110      24.097      20.895      24.097      25.110      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.140      24.110      20.906      24.110      25.140      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.170      24.117      20.915      24.117      25.170      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.200      24.121      20.922      24.121      25.200      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.230      24.126      20.930      24.126      25.230      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.260      24.131      20.935      24.131      25.260      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.290      24.142      20.940      24.141      25.290      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.320      24.160      20.946      24.160      25.320      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.350      24.184      20.951      24.184      25.350      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.380      24.202      20.957      24.202      25.380      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.410      24.215      20.963      24.215      25.410      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.440      24.245      20.969      24.245      25.440      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.470      24.276      20.975      24.276      25.470      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.500      24.294      20.981      24.294      25.500      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.600      24.312      20.987      24.312      25.600      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.700      24.367      20.993      24.367      25.700      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000
```

```
      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.800      24.404      20.999      24.404      25.800      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.900      24.443      21.005      24.443      25.900      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      24.475      21.011      24.475      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      23.600      21.100      23.600      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      23.700      21.200      23.700      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      23.800      21.300      23.800      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      23.900      21.400      23.900      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      24.000      21.500      24.000      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      24.100      21.600      24.100      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      24.200      21.700      24.200      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      24.300      21.800      24.300      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      24.400      21.900      24.400      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      24.500      22.000      24.500      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      24.600      22.100      24.600      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      24.700      22.200      24.700      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      24.800      22.300      24.800      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000
```

```
      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      24.900      22.400      24.900      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.000      22.500      25.000      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.100      22.600      25.100      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.200      22.700      25.200      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.300      22.800      25.300      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.400      22.900      25.400      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.500      23.000      25.500      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.600      23.100      25.600      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.700      23.200      25.700      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.800      23.300      25.800      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.900      23.400      25.900      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      23.500      26.000      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      23.600      26.000      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      23.700      26.000      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      23.800      26.000      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      23.900      26.000      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000
```

```
      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      24.000      26.000      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      24.100      26.000      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      24.200      26.000      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      24.300      26.000      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      24.400      26.000      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      24.500      26.000      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      24.600      26.000      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      24.700      26.000      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      24.800      26.000      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      24.900      26.000      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.000      26.000      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.100      26.000      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.200      26.000      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.300      26.000      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000

      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      26.000      25.400      26.000      26.000      26.000      26.000      26.000
26.000      26.000      26.000      26.000      26.000
```