

Zięba, Dominika Zuzanna (2026) *Variational inference and normalising flows for gravitational wave parameter estimation.* MSc(R) thesis.

https://theses.gla.ac.uk/86304/

Copyright and moral rights for this work are retained by the author

A copy can be downloaded for personal non-commercial research or study, without prior permission or charge

This work cannot be reproduced or quoted extensively from without first obtaining permission from the author

The content must not be changed in any way or sold commercially in any format or medium without the formal permission of the author

When referring to this work, full bibliographic details including the author, title, awarding institution and date of the thesis must be given

Enlighten: Theses
https://theses.gla.ac.uk/
research-enlighten@glasgow.ac.uk

Variational Inference and Normalising Flows for Gravitational Wave Parameter Estimation

Dominika Zuzanna Zięba

Submitted in fulfilment of the requirements for the
Degree of Master of Science (By Research)

School of Physics and Astronomy
College of Science and Engineering
University of Glasgow



September 2026

ABSTRACT

This thesis explores the possibility of the application of Variational Inference with Normalising Flows to the gravitational wave parameter estimation problem, providing a potential machine-learning alternative to the currently used stochastic sampling methods. This approach translates Bayesian inference into an optimisation problem, where a variational distribution parametrised by a neural network is matched to the target posterior by minimising a similarity measure between the two. If successful, it offers fast training utilising parallel GPU likelihood evaluations, low-cost post-training posterior sampling and lower bound evidence estimation. The variational inference algorithm is tested on a binary black hole gravitational wave signal injected into the ground-based interferometer network. Successful results are achieved only on constrained subspaces of the full posterior, with the main obstacle identified to be mode collapse fuelled by the reverse Kullback-Leibler divergence training objective. Different strategies of mitigating the mode collapse problem are investigated, with best improvements seen from posterior temperature annealing.

The thesis further investigates Bernstein polynomial normalising flows as an alternative to spline-based transforms, for use both within the variational inference algorithm, as well as other normalising flow applications across gravitational wave data analysis. While showing promising results on simple targets, the Bernstein flow architecture performs comparably to spline transforms when used within the variational inference algorithm applied to the gravitational wave problem. In the course of the investigation several useful properties of the Bernstein transform flows are observed — a circular transform configuration allowing for defining densities on periodic domains, as well as a density upper bound providing a lower limit on the Bernstein transform degree necessary to match a target with a given maximum density.

DECLARATION

This thesis is my own composition except where indicated in the text. No part of this thesis has been submitted elsewhere for any other degree or qualification.

ACKNOWLEDGEMENTS

I would like to thank my supervisor Dr John Veitch for all his support and guidance.

List of Figures

1 Example time domain gravitational waveform. 17
2 Normalising flow schematic. 28
3 KL divergence between Gaussian distributions. 38
4 KL divergence and multimodal distributions. 39
5 Sine-Gaussian signal model. 51
6 Variational inference applied to a sine-Gaussian posterior. 52
7 Failed variational inference of a CBC posterior in 8 dimensions. 54
8 Relationship between annealing schedule and reverse KL divergence. . . . 60
9 Variational inference with a coupling flow architecture trained on a temperature annealed BBH posterior in 5 dimensions. 63
10 Variational inference with MAF flow architecture trained on a temperature annealed BBH posterior in 5 dimensions.. 64
11 Training loss: coupling vs MAF flow architecture. 65
12 Weights & Biases hyperparameter sweep results.. 68
13 Mode collapse during training. 70
14 NVI training on a bimodal Gaussian mixture in 5 dimensions. 74
15 NVI training result on the BBH 5-dimensional posterior subspace. 75
16 Bernstein polynomial basis. 78
17 Bernstein polynomial transform. 82
18 Bernstein flow and neural spline flow training results on a 5-dimensional correlated Gaussian target. 84
19 Comparison between the Bernstein polynomial flow and the neural spline flow performance on the 5-dimensional BBH problem. 86
20 Theoretical maximum density deficit. 90
21 Measured density deficit. 91
22 Density deficit and mode location. 94
23 Circular Bernstein transform. 96
24 RCPM flow on a sine-Gaussian skymap. 102

List of Tables

1 Flow and training hyperparameters (coupling & MAF) 62
2 Sweep 1: Results 67
3 Sweep 2: Results 69

Contents

I. Preface 8

II. Gravitational Waves 11

2.1. Gravitational waves and their sources 11

2.2. Detections and Scientific Achievements 11

2.3. Gravitational Wave Detectors 13
2.3.1. Current ground-based detector network 13
2.3.2. Future detectors 13
2.3.3. Pulsar Timing Arrays 14

2.4. Compact binary coalescence signals 15
2.4.1. Newtonian approximation - chirp signals 15
2.4.2. CBC waveform models 17
2.4.3. Signal model in a detector 18

2.5. Detection 18

2.6. Bayesian Inference 19
2.6.1. Gravitational Wave Model Likelihood 20
2.6.2. Model Selection and Significance of the Evidence 21
2.6.3. Parameter Estimation 22
2.6.3.1. Markov Chain Monte Carlo Methods 23
2.6.3.2. Nested Sampling 24

III. Normalising Flows for Bayesian Inference 26

3.1. Normalising Flows 26
3.1.1. Overview 26
3.1.2. Density Estimation: Forward KL Training 28
3.1.2.1. Use of normalising flows as density estimators in GW data analysis 29

3.1.3. Variational Inference: Reverse KL Training 30
3.1.3.1. Variational inference and Evidence Lower Bound 32
3.1.3.2. Variational Inference within GW analysis 33
3.1.4. Simulation Based Inference (SBI) 34
3.1.4.1. SBI for GW inference 35

3.2. Kullback-Leibler Divergence 36
3.2.1. Definition and Properties 36
3.2.2. Example: KL divergence between two univariate Gaussian distributions 37
3.2.3. Forward vs Reverse KL divergence optimisation 38

3.3. Normalising Flow Architectures 40
3.3.1. Coupling flows 40
3.3.2. Masked Autoregressive Flows 41

3.4. Normalising Flow Bijectors 42
3.4.0.1. Affine Bijectors 42
3.4.0.2. Neural Spline Flows 43

3.5. Neural networks and computational tools 44
3.5.1. MLP 44
3.5.2. Training a network - backpropagation and stochastic optimisation 45
3.5.3. ResNet 45

3.6. JAX and GPU acceleration 46

3.7. Normalising flows on Riemannian Manifolds 46

IV. Variational Inference 47

4.1. Goal and related work 47

4.2. VI+NF in the context of Gravitational Wave Parameter Estimation 48

4.3. A toy example: a sine-Gaussian waveform likelihood 50
4.3.1. Sine-Gaussian signal model 50
4.3.2. Normalising Flow and training set up 51
4.3.3. Results 52

4.3.4. VI + NF for a full GW likelihood of a CBC signal 53
4.3.5. Initial results: problem with mode locking 53

4.4. Preventing Mode-Collapse in Variational Inference 55
4.4.1. Reparametrisation 56
4.4.2. Uniform mixing 57
4.4.3. Temperature annealing 58
4.4.3.1. Temperature annealing schedules and distribution information content 58
4.4.4. Learning Rate Annealing 60
4.4.5. Denser sampling 60
4.4.6. Preventing Mode collapse in Variational Inference: Results 61
4.4.6.1. 5-dimensional subspace 61
4.4.6.2. Hyperparameter sweeps: 6-dimensional subspace 62
4.4.6.3. Sweep 1 66
4.4.6.4. Sweep 2 67
4.4.6.5. Conclusion 69

4.5. Nested Variational Inference (NVI) 71
4.5.1. Results 73
4.5.1.1. Gaussian mixture 73
4.5.1.2. BBH posterior 74
4.5.2. Other variants and possible forward directions 75
4.5.3. Issues with the implementation 76

V. Bernstein Polynomial Flows 77

5.1. Bernstein-type Polynomials 77
5.1.1. Invertible Bernstein-type Polynomials 78
5.1.2. Range of a Bernstein polynomial 80
5.1.2.1. Bernstein-type polynomial as a normalising flow bijector 81

5.2. Implementation 83

5.3. Bernstein bijector performance 83

5.4. Bounds of the Bernstein Polynomial Transform 85
5.4.1. Bernstein Transform bounds: experimental example 88

5.4.2. Bound dependence on the domain argument 92

5.5. Circular Bernstein transform 94
5.5.1. Circular Bernstein Transform condition 94
5.5.2. Bernstein flow on periodic domains 95

VI. Conclusion 97

A. Riemannian Convex Potential Map flows (RCPMs) 99
A.1. Riemann Convex Potential Map flow for the sine-Gaussian sky map 101

B. Continuous Normalising Flows 102
B.1. Riemannian Continuous Normalising Flows (RCNFs) 104

Part I.
Preface

Since the Nobel prize winning detection of a gravitational wave signal from a binary black hole merger in September 2015 [1], the LIGO-Virgo-Kagra collaboration has continued to detect gravitational wave signals during its first four observing runs [2; 3; 4; 5], opening many novel domains of astrophysics, such as population analysis of the compact binary sources [6; 7; 8; 9], strong-field regime tests of Einstein's theory of General Relativity [10; 11], or independent cosmological inference of the Hubble constant [12; 13]. As the search for new signals continues, we are entering a time of next generation ground-based (3G; [14; 15]) and spaced-based (LISA; [16]) gravitational wave detectors, calling for new parameter estimation methods.

Non-axisymmetric acceleration of masses leads to emission of gravitational waves (GWs). These waves propagate through spacetime as disturbances of the metric, and as they pass through the Earth, they are recorded as laser interferometer strain. Only the most energetic events produce waves of detectable strain amplitudes — so far the only detected signals come from coalescing compact objects, such as black holes or neutron stars. These compact binary coalescence (CBC) signals are transients that stay within the detector bandwidth for only a few seconds, and, despite their extreme origin, upon arrival on Earth their amplitudes are comparable to the instrumental noise. Detecting, as well as analysing these faint signals poses a great statistical challenge.

The waveforms of the CBC signals are predicted by General Relativity and depend on the properties of the binary source, such as the masses, spins and positions of the two compact objects. Since these source parameters are a priori unknown, the data analysis is divided into two stages: searching and parameter estimation. The first stage of the data analysis focuses on searching for astrophysical signals by cross-correlating the recorded interferometer strain with a data bank of model waveforms in a process called matched filtering [17]. This allows us to isolate stretches of data which consist of potential signals — once the signals are located, the analysis can move to estimating the source parameters of the signals found by the search.

Parameter estimation is typically the most computationally costly stage of the GW analysis, and aims to extract all of the information on the signal source parameters from the interferometer data. This is performed within the statistical framework of Bayesian inference, which condenses the available information to the posterior probability distribution of the source parameters [18; 19]. This distribution contains the information about the most likely set of source parameters that led to the signal,

and more importantly the marginal probability distributions on each of the individual parameters — the goal of parameter estimation is to extract this information.

Marginalising the GW posterior distribution is problematic due to the high dimensionality of the parameter space (a CBC signal model has at least 15 free parameters) — typically this has been performed through stochastic sampling of the parameter space, using methods such as Markov Chain Monte Carlo (MCMC) [20] or Nested Sampling [21].

Stochastic sampling methods, even though well suited to analysing the data from the current network of ground-based interferometers, do not extend well to the analysis of the kinds of data we expect to see in the future generation detectors (3G, LISA). The upcoming detectors, due to their increased bandwidth and sensitivity, are expected to routinely detect signals of very long duration — hours (3G) or even years (LISA), as opposed to seconds (current detectors), and will further suffer from confusion noise arising from multiple detectable signals overlapping in time. Performing GW parameter estimation with stochastic methods such as MCMC requires large numbers ($\mathcal{O}(10^6)$) of GW likelihood evaluations, involving computationally costly waveform models, and, due to the inherently sequential nature of these methods, they are generally not easily parallelised and scale poorly to longer signals and larger datasets — analysing these future GW data sets requires innovative new methods to be developed.

An alternative to stochastic sampling is provided by *Variational Inference* (VI; [22]), where the sampling task is translated into an optimisation problem of fitting an easy-to-sample-from parametric distribution to approximate the target posterior. Until recently the use of VI in GW analysis was unfeasible due to the lack of suitable parametric distributions that would be expressive enough to capture the complex correlation structure present in a typical GW posterior — this however has changed with the introduction of *normalising flows*. A normalising flow (NF; [23; 24]) is a machine learning method of constructing an expressive parametric probability distribution by transforming a simple base distribution through a series of invertible mappings. While finding a single parametric transform that would be expressive enough to construct a distribution suitable for the GW problem would be difficult, NFs circumvent this problem and achieve high complexity by composing layers of simple transforms parametrised by a trainable neural network, optimised so that the transformed distribution best approximates the target posterior, according to a chosen metric. Effectively all of the computational cost lies in optimising the network parameters — once a flow is trained one can obtain an approximate posterior sample at the cost of passing a sample from the base distribution through the learnt transform, easily evaluate the target posterior at any point, as well as obtain an evidence

approximation, which is crucial for model selection and unavailable from standard MCMC sampling. NFs when applied in conjunction with VI have great potential of providing an efficient alternative to stochastic methods for GW parameter estimation, allowing for efficient use of GPUs and parallel likelihood evaluations during training.

Most of the development of normalising flows has been so far limited to scenarios where the parameter space is an N-dimensional hypercube, i.e. a subset of the Euclidean space $\mathbb{R}^n$, but the method can be, in principle, translated into cases where the parameter space is a manifold of any prescribed geometry. This happens to be the case with the GW parameter space — the parameter space is a product space of spaces with different topologies, involving for example a spherical component for the parameters describing the sky location of the source and a rotation group for the orientation of the orbit of the source. One could build a normalising flow specific to this geometry by requiring the trainable transforms to be diffeomorphisms of the underlying space — such a flow would be naturally well suited to the geometry of the parameter space and would have the potential of reducing correlations within the parameter space resulting from projecting the curved parameter space into flat space, improving the overall efficiency of parameter estimation.

This work focuses on developing a normalising flow based variational inference algorithm for future detector parameter estimation of GW events. Chapter II. provides an introduction to the field of gravitational wave science, including an overview of GW sources, the current state of science, GW detector network, CBC signals and GW data analysis. Chapter III. is an introduction to normalising flows and variational inference in the context of Bayesian inference for GW parameter estimation. The introduction chapters are followed by two science chapters, Chapter IV. on application of variational inference with normalising flows to the CBC parameter estimation problem, and Chapter V. investigating the properties of Bernstein polynomial normalising flows as a potential alternative to spline based transforms for use in the context of GW data analysis.

Part II.
Gravitational Waves

2.1. GRAVITATIONAL WAVES AND THEIR SOURCES

The Linearised Einstein Field Equations (EFE) of the theory of General Relativity (GR) admit plane wave solutions describing a metric perturbation propagating at the speed of light on top of a background flat Minkowski spacetime. This solution, observed by Einstein in 1916 [25], describes Gravitational Waves (GWs), emitted as a result of asymmetric acceleration of masses.

Only the most energetic sources produce signals of detectable amplitudes — in the context of the current network of ground-based observatories, the potential signals are split into two main categories: short-lived *transient* signals, as well as *continuous* signals. Transient signal sources include compact binary coalescences (CBCs) — these are merging pairs of compact objects, such as binary black holes (BBHs; [1]), binary neutron star systems (BNSs; [26]) and black hole neutron star binaries (NSBHs; [27]), all of which produce signals of understood morphology and are now routinely detected [2; 3; 4; 5], as well as unmodelled coherent burst signals [28; 29; 30] which are predicted to accompany events such as core-collapse supernovae [31], magnetar magnetosphere emission [32] and other exotic sources, but are yet to be detected. Continuous gravitational wave signals are expected to be generated by sources such as isolated rotating neutron stars with asymmetric mass distribution [33], resulting in periodic signals of simple morphology. Further continuous sources include persistent stochastic gravitational backgrounds [34] arising as a superposition of many overlapping unresolved signals.

2.2. DETECTIONS AND SCIENTIFIC ACHIEVEMENTS

As of January 2026, the The Laser Interferometer Gravitational-Wave Observatory (LIGO; [35]) Scientific Collaboration, the Virgo Collaboration [36] and the KAGRA Collaboration [37] have completed four gravitational wave observing runs (O1 - O4), and published catalogues [2; 3; 4; 5] of the detected events up to the end of the first part of the fourth observing run (O4a), which concluded in January 2024. The combined catalogue covering O1-O4a observations contains 218 gravitational wave transient candidates whose origin has been determined as astrophysical with a probability larger or equal to 50% [5]. All of the detected signals are produced by CBCs and include predominantly BBH signals, but also a small number of NSBH candidates and BNS mergers.

The gravitational wave transient observations have revolutionised modern astronomy, with impact on astrophysical population studies, general relativity and cosmology. The most impactful individual events include the first GW detection (GW150914; [1]) in September 2015 of a signal from a 36-29 $M_\odot$ BBH coalescence, which confirmed Einstein's GR prediction of asymmetric acceleration of masses leading to emission of GWs, and the first detected BNS event with an accompanying electromagnetic counterpart signal (GW170817; [26]) observed in August 2017 which confirmed the GR prediction of the GW propagation speed and marked the beginning of multimessenger astronomy.

The growing number of detections have allowed for modelling of the mass and spin distributions of stellar mass black hole binaries, providing insights into their population characteristics, merger rates, and formation channels [6; 7; 8; 9]. The mass distribution has been found to show no evidence of redshift dependence within the redshift limits of detected signals ($z \lessapprox 1$), and is described by a broad power law with support in both the lower ($3 - 5\,M_\odot$) and upper ($\gtrapprox 45\,M_\odot$) mass gaps and is characterised by yet unexplained overdensities at $\sim 10\,M_\odot$ and $\sim 35\,M_\odot$, posing challenges to current understanding of stellar physics [9]. BBH merger rate has been found to increase with redshift, supporting the understanding of the cosmic star formation density [8; 9]. Black hole spins are found to have non-extremal magnitudes and be more likely aligned with the orbital angular momentum, pointing to isolated binary formation, while at the same time significant support for misaligned spins points to a contribution from a dynamical formation channel to the overall binary population [8; 9].

The GW detections have provided a probe of gravity in the strong field regime, providing support to the Einstein's theory of General Relativity (GR) [10; 11]. Waveform consistency tests have found no statistically significant inconsistencies between the inspiral-merger-ringdown phases of the analysed signals. Parametrised tests introducing extra signal parameters capturing potential deviations from GR have found all tested deviations to be consistent with zero, within statistical uncertainty [11]. Polarisation tests searching for extra polarisations beyond the plus and cross predicted by GR have so far found no support for polarisations beyond the two tensor modes [11]. Gravitational Wave speed propagation measurements have put an upper bound on the fractional deviation of GW speed form the speed of light of 10^{-15} [38], ruling out many modified theories of gravity. Detections have allowed for deducing an upper bound on the graviton mass of $m_g < 2.42 \times 10^{-23}$ eV/c^2, the tightest bound so far, consistent with the GR prediction of massless gravitons [11]. Ringdown tests have found merger ringdown mode frequencies to match GR predictions [11].

Gravitational wave science has allowed for an independent method [39; 13] of constraining the Hubble constant H_0, with the potential of resolving the inconsistency between the early universe cosmic microwave background (CMB) measurements (Planck; [40]) and local universe supernova results [41], the so-called *Hubble tension*. The luminosity distance information encoded in the gravitational wave signal amplitude, together with redshift information from the detected electromagnetic counterpart (GW170817; [26]) or from galaxy surveys (no counterpart events) allows to use CBC signals as standard sirens — this has lead to a GW measurement of the Hubble constant as $H_0 = 76.6^{+13}_{9.5}$ $\mathrm{km\,s^{-1}\,Mpc^1}$ (68% credible interval) [13], so far consistent with both Planck and supernova results.

2.3. GRAVITATIONAL WAVE DETECTORS

2.3.1. *Current ground-based detector network*

The current network of gravitational wave detectors includes five ground based observatories: Advanced LIGO (aLIGO; [35]) consisting of two 4 km observatories in Hanford (H1) and Livingston (L1) in USA, Advanced Virgo (AdVirgo; [36]) — a 3 km detector near Pisa, Italy, the KAGRA [37] detector in Japan — an underground cryogenic 3 km observatory, as well as a 600 m GEO600 [42] detector in Germany serving, due to its lower sensitivity, as site for testing technological advancements. All these detectors are variants of L-shaped Michelson interferometers [43] which detect gravitational waves in the high frequency band of 10 - 1000 Hz by measuring the relative change in the lengths L of detector arms (dimensionless detector strain, $2\Delta L/L$) resulting from a passing gravitational wave. The strain signal is obtained by splitting a coherent laser beam down the two interferometer arms, reflecting the split beams and detecting the interference pattern when the signals rejoin.

The detector sensitivity is conventionally quoted as a BNS inspiral range — a sky-averaged distance at which a 1.4-1.4 Solar mass BNS signal would result in an SNR 8 signal at the detector. The O4a observing run BNS sensitive distance was equal to ~ 160 Mpc for L1 and H1, a double improvement from the initial sensitivity of ~ 80 Mpc in O1 [44]. Virgo and KAGRA observatories remained in commissioning during O4a and were expected to rejoin the observation with sensitivities of ~ 50 Mpc in Virgo and ~ 5 Mpc in KAGRA [4].

2.3.2. *Future detectors*

There are a number of future gravitational wave detectors in various stages of design and planning. Further to improvements planned for the existing LIGO detectors, the ground-based network of detectors will be joined by the LIGO India [45] detector which currently remains under construction and is based on the aLIGO design, providing further improvements to the signal sky-localisation capabilities of the network.

The so-called *third generation* (3G) of ground-based detectors will include two new projects – the Einstein Telescope (ET; [14]) and Cosmic Explorer (CE; [15]), extending the sensitive frequency range of the current network to high redshift signals. The European ET detector may be a triangular configuration interferometer with cryogenic cooling and 10 km arms, sensitive to frequencies as low as $\sim 1\,\mathrm{Hz}$ and expected to detect CBC signals with redshifts $z > 10$. CE detector will be located in USA; it will be an L-shaped detector with 40 km arms, 10 times the length of the current LIGO detectors, sensitive to binary inspirals at redshifts $z > 20$ and with a wide bandwidth between 1 Hz and a few kHz.

Further to the ground-based observatories, there are space-based detectors under planning and development. The European Space Agency (ESA) led Laser Interferometer Space Antenna (LISA; [16]) observatory, currently in the design and prototyping stage and expected to launch in mid 2030s, is planned as a triangular configuration with test masses on solar orbits, providing 2.5 million km arm lengths, with sensitivity in the 0.1 mHz – 1 Hz bandwidth, capable of observing supermassive black hole mergers, extreme mass ratio inspirals (EMRIs) as well as galactic white dwarf binaries and potential early universe relics. Further to LISA, similar Chinese projects are under development, including TianQin [46] observatory with test masses in geocentric orbits and Taiji [47] project in heliocentric orbits. A second generation Japanese space-based observatory DECIGO (DECi-Hertz Interferometer Gravitational-wave Observatory; [48]) with a frequency bandwidth of 0.1 – 10 Hz, aimed at detecting early inspiral CBC events as well as GWs from primordial black holes, is also currently in a planning stage with a pathfinder mission B-DECIGO expected to launch in the 2030s [49].

These future detectors will provide high-precision tests of general relativity, new insights into the history of black hole and neutron star formation, and a new window into the early universe.

2.3.3. *Pulsar Timing Arrays*

In addition to the interferometer detectors, gravitational waves can be detected by Pulsar Timing Arrays (PTAs; [50]), ground-based detectors which measure gravitational waves via observation of time delays of radio frequency pulse arrivals from astrophysical arrays of pulsars. PTAs are sensitive in the $10^{-9} - 10^{-6}$ Hz band, expected to detect low frequency stochastic background and continuous signals from supermassive black hole binaries. Evidence of gravitational wave signals in the PTA data has been reported in recent years by a number of separate studies [51; 52; 53].

2.4. COMPACT BINARY COALESCENCE SIGNALS

Compact binary coalescences (CBCs) are systems of merging pairs of compact objects — these can be binary black holes (BBHs; [1]), binary neutron star systems (BNSs; [26]), black hole neutron star binaries (NSBHs; [27]). The two compact objects forming a binary are on an orbit around a common centre of mass, and the non-axisymmetric nature of the system leads to emission of gravitational waves. Gravitational waves carry energy away from the system, resulting in a decreasing orbital separation and an increasing orbital frequency. This so-called *inspiral* phase continues until the two objects collide in a *merger* stage, leading to a formation of a new compact object, which GR predicts to be a Kerr black hole [44]. The remnant object continues to emit gravitational waves until it settles into its new state — this last stage of the waveform is the *ringdown*.

The emitted signals can be written as a perturbation h_{ij} of the background spacetime metric, which in the Cartesian frame with z directed along the direction of propagation can be written as

$$h_{ij}(t,\theta) = \begin{pmatrix} h_+(t,\theta) & h_\times(t,\theta) & 0 \\ h_\times(t,\theta) & -h_+(t,\theta) & 0 \\ 0 & 0 & 0 \end{pmatrix}, \tag{2.4.1}$$

where h_+ and $h_\times$ are amplitudes of two waveform polarisations — the plus and the cross polarisation respectively, which together form the waveform [54]. The two polarisations are functions of a parameter vector θ, describing the source (*intrinsic* parameters) and its orientation with respect to the detector (*extrinsic* parameters). The set of parameters [19] required to characterise a quasi-circular binary signal is

$$\theta = (m_1, m_2, D_\mathrm{L}, \alpha, \delta, \iota, \psi, \phi_\mathrm{c}, t_\mathrm{c}, \boldsymbol{S}_1, \boldsymbol{S}_2), \tag{2.4.2}$$

the primary m_1 and secondary m_2 mass of the binary, luminosity distance to the source D_L, source position in the sky determined by its right ascension α and declination δ, the inclination angle ι between the normal to the orbital plane of the binary and the line of sight, polarisation angle ψ, orbital phase ϕ_c at a reference time t_c, usually representing the time of coalescence, and two 3-dimensional spin vectors $\boldsymbol{S}_1, \boldsymbol{S}_2$ (one per each mass in the binary), giving a total of 15 free parameters.

2.4.1. *Newtonian approximation - chirp signals*

In the early stages of the inspiral, when the magnitude of the velocity difference of the two binary components, v, is much smaller than the speed of light, $v \ll c$, the CBC waveforms can be approximated by Newtonian mechanics, where we have,

following [44],

$$h_+ = -2(1+\cos^2\iota)\frac{GM\eta}{c^2 r}\left(\frac{v}{c}\right)^2\cos 2\phi \tag{2.4.3}$$

and

$$h_\times = -4\cos\iota\frac{GM\eta}{c^2 r}\left(\frac{v}{c}\right)^2\sin 2\phi, \tag{2.4.4}$$

where $M = m_1+m_2$ is the total mass of the system, ι is the inclination, $\eta = m_1 m_2/M^2$, ϕ is the orbital phase, and r is the distance to the observer, $r = (1+z)^{-1}D_L$ assuming a flat expanding universe, and where z is the redshift of the source.

The frequency of the inspiral signal is twice the orbital frequency of the binary, $f = 2f_{\rm orbital}$, and its time evolution is governed by the chirp mass of the system $\mathcal{M} = (m_1 m_2)^{3/5}M^{-1/5}$. In the Newtonian approximation [55] we have

$$\frac{df}{dt}(f) = \frac{96}{5}\frac{1}{c^5}\pi^{8/3}(G\mathcal{M})^{5/3}f^{11/3} \quad . \tag{2.4.5}$$

From Equations 2.4.3-2.4.5 we see that the inspiral frequency and waveform amplitude increases as a function of time, resulting in the characteristic *chirp* signals. The binary coalesces when the orbital separation will be approximately equal to 3 Schwarzschild radii, $R = \frac{6GM}{c^2}$, $M = m_1 + m_2$. The frequency at which this occurs is referred to as the frequency of the last stable orbit $f_{\rm ISCO}$,

$$f_{\rm ISCO} = \frac{c^3}{6^{3/2}\pi MG} \approx 220\left(\frac{20M_\odot}{M}\right)\,\text{Hz}. \tag{2.4.6}$$

A binary black hole system with masses $m_1 = m_2 = 10\,M_\odot$ will merge at approximately $f_{\rm ISCO} \approx 220\,$Hz, while a BNS system with $m_1 = m_2 = 1.4\,M_\odot$, will merge at approximately 1500 Hz – a lower mass system merges at a higher frequency and is present in the detector bandwidth for longer, however the signal strain amplitude at the merger will be lower.

The strain amplitude h varies with the system orbital frequency f approximately as

$$h(f) \sim 1.4\times 10^{-22}\left(\frac{f}{100\text{Hz}}\right)^{2/3}\left(\frac{\mathcal{M}}{1.22M_\odot}\right)^{5/3}\left(\frac{10\text{Mpc}}{r}\right), \tag{2.4.7}$$

so the strain amplitude at the merger of a binary BBH sytem with component masses $m_1 = m_2 = 10M_\odot$ located at $r = 10\,$Mpc away from the detector is only about

$$h_{\rm merger} \approx 6\times 10^{-21}, \tag{2.4.8}$$

and still larger by an order of magnitude than the lower mass BNS system, producing a serious detection challenge in bringing the detector noise down to a comparably low level.

An example time domain BBH waveform is presented in Figure 1.

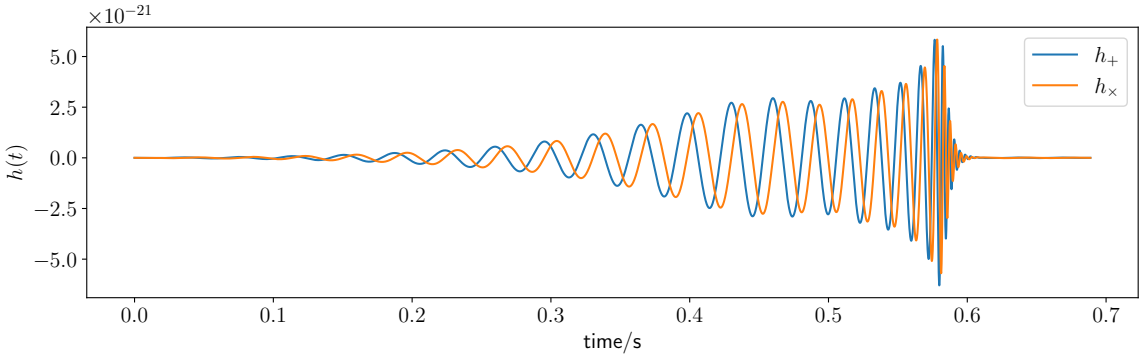


Figure 1: Example time domain gravitational waveform.

Plus and cross polarisations of an example time domain BBH waveform from a binary system with component masses $m_1 = m_2 = 35\,\mathrm{M}_\odot$ situated at luminosity distance $D_L = 100\,\mathrm{Mpc}$, and with inclination $\iota = \pi/4$. Generated using the IMRPhenomD approximant with minimum frequency $f_{\mathrm{min}} = 30\,\mathrm{Hz}$. The two polarisations are shifted in phase by a quarter of a cycle, the waveform frequency and signal amplitude increase throughout the inspiral, reaching the merger stage and subsequent ringdown of the merger object.

2.4.2. *CBC waveform models*

The Newtonian approximation to the inspiral describes only the leading quadrupole harmonic — the radiation emitted along the direction of the orbital angular momentum, and hence is only valid for non-precessing signals and in the weak field $v \ll c$. Obtaining a full relativistic description of all 3 stages of the waveform would require solving the full Einstein Field Equations (EFE) for the binary system — since the analytical solutions are not known, approximate waveform models are used. There are a number of approaches to obtaining the approximate models, as reviewed in [56]:

- Post Newtonian (PN) formalism, in which a Newtonian weak-field solution to the inspiral phase is expanded by introducing relativistic corrections in increasing expansion orders of v/c.
- Numerical relativity (NR) simulations, where full relativistic EFE are solved numerically, providing accurate approximation to the full inspiral-merger-ringdown signal.
- Phenomenological models, where rather than solving the equations of motion, the frequency domain signal amplitude and phase is directly described by fast to evaluate closed-form interpolants, informed by the PN formalism in the inspiral phase and NR in the merger-ringdown.
- Effective one-body (EOB; [57]) approach where the two-body dynamics of the system is mapped to a metric of a single test particle.

- Surrogate models, which extend NR simulations to regions of parameter space between NR simulated points.
- Gravitational self-force — a formalism where high mass ratio systems can be described as one body systems with a perturbation.

The LIGO-Virgo-KAGRA CBC data analysis [2; 3; 4; 5] uses predominantly phenomenological models from the IMRPHENOM [58] family, for their efficiency and parameter space coverage, as well as SEOBNR EOB [57] waveforms. Models currently used to analyse candidate signals assume non-eccentric quasi circular orbits.

2.4.3. *Signal model in a detector*

Upon arrival at the detector, the gravitational wave signal is a superposition of plus and cross polarisation of the waveform, each multiplied by the detector-specific antenna response functions F_+ and $F_\times$, resulting in signal of the form

$$h(t) = F_+(\alpha, \delta, \psi, t_c)h_+(t) + F_\times(\alpha, \delta, \psi, t_c)h_\times(t), \tag{2.4.9}$$

and where the antenna functions are determined by the spatial orientation of the interferometer arms with respect to the source [19].

The signals in different detectors will not only differ by the antenna response function, but also will be shifted in time with respect to one another due to differences in signal travel time between the source and the detector locations.

To account for the time delay and allow for coherent analysis of the signal in a network of detectors, the signal coalescence time is defined as the geocentric coalescence time, that is the time of coalescence which would be measured by a observatory located at the centre of the Earth, and signals in individual detectors are shifted by a time delay τ, accounting for the light travel time between the detector and the reference point at the Earth's centre [19].

In the frequency domain this time delay corresponds to a frequency modulation, i.e.

$$\tilde{h}_\tau(f) = \tilde{h}_0(f)e^{-2\pi i f\tau} \tag{2.4.10}$$

where $\tilde{h}_0(f)$ is the frequency domain signal in the reference frame of the centre of the Earth (Eq. 2.4.9) and $\tilde{h}_\tau(f)$ is the signal in the detector with time delay τ.

2.5. DETECTION

The detector strain data is a time series $d(t)$, which is modelled as a sum of detector-specific Gaussian coloured noise $n(t)$, and transient GW signal $h(t)$, which stays in the detector sensitive frequency band for a time duration of the order of tens of seconds, all sampled at a 4 kilohertz rate.

The signals are searched for in the interferometer strain data in order to identify short signal-contain, ing data segments for further analysis. The searches [4; 59**?**] are performed at two different latencies — a real-time low-latency *online* search for rapid public alert release and a later more sensitive *offline* search with refined noise statistics. There are two main search approaches — a modelled matched filtering search cross-correlating the interferometer data with a CBC waveform template bank, as well as an unmodelled search looking for coherent power excess within the interferometer network, without assumptions on signal morphology. In the latest observing runs 5 different search pipelines were used, including 4 template search pipelines, GstLAL [60], Multi-Band Template Analysis (MBTA) [61] and PyCBC [62], differing in template bank choices and ranking statistic computations, an online search only pipeline SPIIR [63], as well as an unmodelled cWB [64] coherent burst search pipeline.

Matched templates and unmodelled signals with signal-to-noise ratio (SNR) above a set pipeline-dependent threshold produce signal triggers, and each trigger's significance is assessed by computing its false alarm rate (FAR) — probability of the trigger being explained by noise alone, given the estimated background noise statistics. The FAR statistic is then combined with the understanding of astrophysical compact binary populations and merger rates to produce a $p_{\rm astro}$, probability that the trigger is a signal of astrophysical origin — triggers with $p_{\rm astro} > 0.5$ are usually considered astrophysical in origin and contribute to the detected event catalogs, used for further astrophysical inference [4; 59].

2.6. BAYESIAN INFERENCE

The goal of the gravitational wave data analysis is to access the information contained within the interferometer strain data, that is to understand what is the most likely signal model and subsequently examine the probability density distribution of the signal parameters. This allows us first to say, for example, that a particular signal was more likely an NSBH rather than a BBH coalescence, and subsequently to determine that the neutron star mass was between some m_1 and m_2 with a probability of, say, 90%. These two analysis steps are *model selection* and *parameter estimation* respectively, and both are performed within the statistical framework of *Bayesian Inference*.

Within the Bayesian framework, the quantitative information about a gravitational wave signal model $M(\theta)$ and its parameters θ is described by the Bayes' Theorem,

$$p(\theta|d, M) = \frac{p(d|\theta, M)p(\theta|M)}{p(d|M)} \tag{2.6.1}$$

relating the *posterior probability* $p(\theta|d, M)$ to the data *likelihood* $p(d|\theta, M)$, *prior probability* $p(\theta|M)$ and the data model *evidence* $p(d|M)$, all conditioned on a chosen

signal model M. The posterior probability $p(\theta|d, M)$ describes the probability of the signal being correctly described by the set of parameters θ, given the observed experimental data d, represented by the detected interferometer strain — this probability is a priori unknown and is constructed from the remaining probabilities involved. The likelihood $p(d|\theta, M)$ describes the reverse probability, that is the probability of observing experimental data d assuming the model parameters θ describe the signal contained in the data — the likelihood essentially describes a probabilistic noise model for the experiment and can be written down based on assumptions about the gravitational wave interferometers. The prior $p(\theta|M)$ is the probability of observing a source with parameters θ regardless of the detected data, and follows from astrophysical assumptions about binary compact object populations and characteristics, as well as geometrical arguments about the source orientation with respect to the detectors. Finally, the evidence $p(d|M)$ describes the probability of observing the experimental data d given the model chosen signal model M — this probability is unknown prior to the analysis, but can be estimated as a normalising constant of the posterior, and by integrating the likelihood over the prior volume.

In the context of the introduced probability densities, the model selection stage of the analysis is concerned with comparing the evidences of different signal models to understand which is a more probable description of the recorded data, while parameter estimation is concerned with understanding the posterior distribution of the parameters of a particular assumed model. The two problems are parallel in terms of the conclusions drawn and ask separate, independent questions, however are closely related during the analysis, due to the relationship between the model posterior and model evidence captured within the Bayes' theorem. Posterior samples (Section 2.6.3) are sometimes by-products of methods of estimating the evidence, such as in nested sampling (Section 2.6.3.2), or the model evidence estimate can follow from a posterior estimation method, as in variational inference (Section 3.1.3.1).

2.6.1. *Gravitational Wave Model Likelihood*

Gravitational wave parameter estimation is typically performed in the frequency domain, where we assume that in each frequency bin k, the strain Fourier transform $\tilde{d}_k$ is equal to the sum of the GW signal $\tilde{h}_k(\theta)$ and interferometer noise noise $\tilde{n}_k$,

$$\tilde{d}_k = \tilde{h}_k(\theta) + \tilde{n}_k. \tag{2.6.2}$$

Assuming stationary and Gaussian noise in each detector, the likelihood of a GW signal is the Gaussian likelihood of the residual noise obtained when subtracting the signal model from the observed interferometer strain data in each frequency bin. The

likelihood of observing the data transform $\tilde{d}_k$ in the frequency bin k is then

$$p(\tilde{d}_k|\theta,\sigma_k) \propto \exp\frac{-(\tilde{d}_k-\tilde{h}_k(\theta))^2}{2\sigma_k^2}, \tag{2.6.3}$$

where σ_k^2 is the variance of the noise in that bin. Assuming the noise is independent in each frequency bin, the likelihood of the strain data is the product of the individual Gaussian likelihoods over the frequency bins.

When evaluated coherently in a network of detectors $\{D\}$, and assuming independence of noise between the detectors, the GW likelihood is a product of Gaussian likelihoods over the network, that is

$$p(\{\boldsymbol{d}^D\}|\theta,\{S^D(f)\}) \propto \prod_D p(\boldsymbol{d}^D|\theta,S^D(f)) = \prod_D \exp\left[-\frac{2}{T}\sum_{k>0}\frac{\left|\tilde{d}_k^D-\tilde{h}_k^D(\theta)\right|^2}{S^D(f_k)}\right], \tag{2.6.4}$$

where $\{\boldsymbol{d}^D\}$ denotes the set of data vectors detected in a detector network $\{D\}$, and where $\{S^D(f)\}$ is the corresponding set of one-sided interferometer noise power spectral densities (PSDs) and T is the signal duration [18].

Evaluating this Gaussian likelihood is the primary cost of performing parameter estimation, as every evaluation requires generating a new waveform model $h(\theta)$.

2.6.2. *Model Selection and Significance of the Evidence*

An important step of the signal data analysis is Bayesian model selection — a statistical method that allows for quantitative comparison between the probability of the 'noise only' versus the 'noise+signal' model, or two competing waveform models.

Let $\{M_i\}_{i=1}^N$ be a set of competing explanatory data models for the observed interferometer strain data d. The probability of the model M_i is, following the Bayes Theorem,

$$P(M_i|d) = \frac{P(M_i)P(d|M_i)}{P(d)}, \tag{2.6.5}$$

where $p(M_i)$ represents the prior probability of the model M_i, $p(d|M_i)$ is the likelihood of observing data d under model M_i, and $P(d)$ is the marginal likelihood of the data, marginalised over a complete set of data models M_i, i.e. $P(d) = \sum_i P(d|M_i)$, where $\sum_i P(M_i|d) = 1$.

Since we do not have access to a complete set of explanatory models, we cannot compute the marginal likelihood $p(d)$ as well as the model probability $P(M_i|d)$. What we can do however is compute ratios of probabilities between two models, allowing for model selection. Given two models, M_i and M_j, the ratio of probabilities of model

i to model j is given by

$$\frac{P(M_i|d)}{P(M_j|d)} = \frac{P(M_i)P(d|M_i)}{P(M_j)P(d|M_j)} = \frac{P(M_i)}{P(M_j)}B_{ij}, \tag{2.6.6}$$

where $B_{ij} = P(d|M_i)/P(d|M_j)$ is the so-called *Bayes factor*, or the likelihood ratio. Often we have no prior reason to prefer one model over the other, setting the prior odds $P(M_i)/P(M_j)$ to 1, so that the ratio of the model probabilities is given by the Bayes factor.

If our models M_i are usually parametric models representing the signal waveform model $h(\theta_{GW})$, where $\theta_{GW} \in \Theta_{GW}$ are waveform parameters describing the source and its relation to the observatory. The likelihood of the data d under model M_i parametrised by θ_{GW}, also known as the Bayesian evidence of model M_i, is given by the likelihood $p(d|\theta_{GW})$ marginalised over the parameter space Θ_{GW},

$$p(d|M_i) = \int_{\Theta_{GW}} p(d|\theta_{GW}, M_i)p(\theta_i|M_i)d\theta_{GW}. \tag{2.6.7}$$

Due to the functional form of the likelihood $p(d|\theta_{GW}, M_i)$, this evidence can not be computed analytically, and numerical integration methods are inefficient due to high dimensionality of the parameter space Θ_{GW}. Bayesian evidence is therefore not easily computed, while being crucial for model selection and hypothesis testing, and any data analysis methods that provide access to the evidence are particularly useful. These include the nested sampling algorithm, which stochastically computes the evidence integral, and as a byproduct of this computation produces posterior samples of the gravitational wave parametrs θ_{GW} — the standard algorithm for performing Bayesian inference on gravitational wave signals. Another such algorithm, if successful, would be a variational inference algorithm, which is the subject of this work — a byproduct of fitting a normalised variational distribution to the posterior distribution on the gravitational wave parameters is an estimate of the normalisation constant, the evidence, and in fact, in the specific case of reverse KL-divergence optimisation, an estimate of a lower bound on the evidence.

2.6.3. *Parameter Estimation*

Given the gravitational wave signal and noise model, it is straightforward to compute, up to a normalising constant (evidence), the posterior probability $p(\theta|d)$ for any choice of gravitational wave parameters $\theta = (\theta_1, \ldots, \theta_n)$. In order to understand the nature of the gravitational wave source one would like to understand the marginal distributions

$$p(\theta_i|d) = \int \cdots \int_U p(\theta|d)\, d\theta_1 \ldots d\theta_{i-1} d\theta_{i+1} \ldots d\theta_n, \tag{2.6.8}$$

in order to make statements about credible intervals for the individual parameters, or understand higher dimensional marginals to quantify the correlations between different parameters.

The naive approach would be to simply perform the integration to obtain the marginals of interest — however since analytic solutions to these integrals are not known, the integration would need to be performed numerically, the computational cost of which is prohibitive due to the high dimensionality of the problem.

The standard alternative to obtaining the marginal distributions via integration is provided by stochastic sampling techniques, where the aim is to obtain a finite sample $\{\theta\}_{i=1}^{N} \sim p(\theta|d)$ drawn from the target posterior. Once a posterior sample has been obtained, the marginal distributions of $p(\theta|d)$ can be approximated by the sample frequency densities, obtained by projecting the samples onto the axes of interest, obtaining one-dimensional sample histograms or 'corner plots' of 2-dimensional projections. The process of obtaining posterior samples is referred to as parameter estimation.

The stochastic sampling methods used within the field of gravitational waves fall within one of two larger classes of algorithms – Markov Chain Monte Carlo methods (MCMC) and Nested Sampling.

2.6.3.1. *Markov Chain Monte Carlo Methods*

MCMC methods constitute a class of stochastic sampling methods where a sample from a target probability distribution is obtained by performing a random walk in the probability space.

The simplest example, and a foundation of many of the more advanced MCMC sampling methods, is the Metropolis-Hastings algorithm, presented by W. K. Hastings in 1970 [65] as a generalisation of a sampling algorithm proposed by N. Metropolis in 1953 [66]. Given a sample θ drawn from a target probability density p, the Metropolis-Hastings algorithm proceeds by suggesting a transition to a state θ' drawn from a so-called *proposal distribution*. The transition is accepted with probability

$$p = \min\left(1, \frac{p(\theta')}{p(\theta)}\right), \tag{2.6.9}$$

so that transitions to higher probability states are always accepted, while transitions to lower probability states are accepted with probability equal to the ratio of the probabilities of the two states. Given a symmetric proposal distribution, the sequence of accepted state transitions is a discrete-time Markov chain representing a set of samples $\{\theta_i\}_{i=1}^{N} \sim p$.

In practice the algorithm is initialised at a random state $\theta_0 \in \Theta$ where $p(\theta_0) \neq 0$, and an initial 'burn-in' segment of the Markov chain is excluded from the final posterior sample, as not representative of the target.

Usually there is a high level of fine-tuning required to perform the sampling efficiently – one can easily see that a proposal distribution too narrow with respect to the target will lead to a high acceptance rate but slow and inefficient exploration of the sample space, while a very wide proposal can lead to low acceptance rates due to low target probability densities at the proposed states. MCMC methods tend to struggle with multimodal distributions, since a single Markov chain would require a sequence of unlikely transitions through the low probability density regions of the parameter space to traverse the distance separating isolated modes.

Different advanced methods exist that address these problems, ensamble methods which obtain a target sample from a union of multiple simultaneous Markov chains initiated in different regions of the target, parallel tempering where a main Markov chain can exchange states with chains exploring annealed target distributions, hence encouraging intermodal transitions, various adaptive methods where the proposal distribution is informed by the past chain of samples, as well as Hamiltonian methods where the Markov chains evolve in the probability space according to simulated Hamiltonian dynamics.

The MCMC sampler used by the LVK for CBC data analysis is an implementation of a parallel tempered MCMC algorithm included in the LALINFERENCE [19] library.

Despite the advancements in the MCMC methods, the use of Markov chains introduces a sequential element to their design, putting a limit to their efficiency.

2.6.3.2. *Nested Sampling*

Nested Sampling, an alternative stochastic sampling method introduced in 2004 by J. Skilling [21], unlike MCMC methods, is aimed at estimating the evidence integral, while a posterior sample is produced as an optional by-product.

The idea is to rewrite the n-dimensional evidence integral over the parameter space Θ,

$$Z \coloneqq p(d) = \int_{\theta \in \Theta} \underbrace{p(d|\theta)}_{\coloneqq \mathcal{L}(\theta)} p(\theta) d\theta \tag{2.6.10}$$

as a 1-dimensional integral over the *prior volume*. We define the prior volume at likelihood λ as the prior volume corresponding to the subset of the parameter space Θ, where the likelihood $\mathcal{L}(\theta)$ is larger than λ,

$$X(\lambda) \coloneqq \int_{\mathcal{L}(\theta) > \lambda} p(\theta) d\theta, \tag{2.6.11}$$

so that with increasing likelihood λ, $X(\lambda)$ shrinks from 1 at $\lambda_{\min}$ to 0 at $\lambda_{\max}$. We then view the likelihood as a 1-dimensional function of the prior volume X, $\mathcal{L}(X)$, defined via the mapping $\mathcal{L}(X(\lambda)) \mapsto \lambda$, which allows as to rewrite the evidence integral as

$$Z = \int_0^1 \mathcal{L}(X) dX, \tag{2.6.12}$$

where $dX = p(\theta)d\theta$ corresponds to a prior volume element in the 1-dimensional space of X.

The algorithm proceeds by sampling a set of N *live points* distributed according to the prior $p(\theta)$. At each iteration we reject the live point of lowest likelihood and replace it with a new live point of higher likelihood drawn from the prior. At step k of the algorithm the sequence of rejected samples $\theta_1, \ldots, \theta_k$ corresponds to a sequence of increasing likelihoods $\lambda_1 < \cdots < \lambda_k$, corresponding to a shrinking set of prior volumes $X_1 > \cdots > X_k$. These are then simply used to obtain the current lower bound of the evidence using a Riemann sum,

$$Z \geq \sum_i^k \lambda_i (X_{i-1} - X_i), \quad X_0 = 1. \tag{2.6.13}$$

Even though we do not know the exact values of volumes X_i, the key observation is that the shrinkage of the prior volume X can be described statistically and its expectation value depends only on the chosen fixed number of live points N, with $X_i \approx \exp(-i/N)$. The Riemann sum is evaluated using this approximation.

We can also compute an approximate upper bound of the evidence by associating the highest likelihood live point $\lambda_{\max}$ with the remaining prior volume X_k, so that $Z \lessapprox \sum_i^k \lambda_i (X_{i-1} - X_i) + \lambda_{\max} X_k$. The algorithm terminates at step k when the current evidence lower bound estimate of the evidence exceeds a set fraction of the approximate upper bound.

Posterior samples can be estimated by assigning importance weights $\{w_i\}_{i=1}^k$ to the sequence of rejected prior samples $\{\theta_i\}_{i=1}^k$, where

$$w_i = \frac{\lambda_i (X_{i-1} - X_i)}{Z}, \tag{2.6.14}$$

where Z is the nested sampling estimated of the evidence.

The nested sampling algorithm implementation used for the LVK data analysis is provided in the Dynesty [67] package.

A recent development in the nested sampling algorithms for gravitational wave applications is the Nessai [68] algorithm — a nested sampling scheme with a normalising flow based constrained prior estimation for live point sampling.

Part III.
Normalising Flows for Bayesian Inference

3.1. NORMALISING FLOWS

Bayesian inference of high-dimensional posterior distributions is typically performed using computationally costly stochastic sampling techniques. The idea behind the use of *normalising flows* (NFs) for Bayesian inference is to approximate a target posterior distribution with a tractable parametric distribution, effectively translating Bayesian inference into an optimisation problem.

3.1.1. *Overview*

Assume $\mathbf{x} \in X$ is a continuous random variable distributed according to an a priori unknown probability distribution $p_X(\mathbf{x})$. A *normalising flow* [23; 24] is a generative model of a flexible parametric distribution which can be used to approximate the density $p_X(\mathbf{x})$.

The idea is to introduce a *latent space* $Z \cong X$ and define a diffeomorphism (smooth function with a smooth inverse) f,

$$f \colon X \to Z, \tag{3.1.1}$$

such that the distribution p_Z of the variable $\mathbf{z} = f(\mathbf{x}) \sim p_Z$ is simple, i.e. easy to sample from and evaluate — such as an n-dimensional normal distribution, hence the *normalising* in the name.

Given such a diffeomorphism f, it follows that

$$p_X(\mathbf{x}) = p_Z(f(\mathbf{x})) \left| \frac{\partial f}{\partial \mathbf{x}} \right| \tag{3.1.2}$$

where $\left|\frac{\partial f}{\partial \mathbf{x}}\right| = |\mathbf{J}_f|$ is the absolute value of the Jacobian determinant of f.

In practice we are not looking for a diffeomorphism f that exactly satisfies Eq. 3.1.2, but rather a transform $f \colon X \to Z$ which normalises an approximate distribution $q_X(\mathbf{x}; \phi) \approx p_X(\mathbf{x})$,

$$p_X(\mathbf{x}) \approx q_X(\mathbf{x}; \phi) = p_Z(f(\mathbf{x}; \phi)) \left| \frac{\partial f}{\partial \mathbf{x}} \right|. \tag{3.1.3}$$

The diffeomorphism f is typically a composition of $K \in \mathbb{N}$ transforms $f = f_K \circ \cdots \circ f_1$ (Figure 2), and is parametrised by neural network (NN) parameters ϕ, which are

optimised to minimise a chosen divergence statistic between the flow distribution q_X and the target distribution p_X.

Once the optimal parametrisation of f is found, one can obtain an approximate target sample $\mathbf{x} \sim p_X$, by applying the inverse transform f^{-1} to a latent sample $\mathbf{z} \sim p_Z$,

$$\mathbf{z} \sim p_Z \implies \mathbf{x} = f^{-1}(\mathbf{z}) \sim q_X \approx p_X. \tag{3.1.4}$$

One can also evaluate an approximation of the target density p_X at arbitrary points $\mathbf{x} \in X$ using the forward transform f and its Jacobian determinant according to Eq. 3.1.3.

The choice of the divergence statistic is limited by whether the samples from the target p_X are available. A commonly used divergence statistic is the Kullback-Leibler (KL) divergence (Section 3.2), which measures the relative entropy between the two distributions.

In the case where one has access to samples from the target p_X, the training objective is usually the forward KL divergence,

$$\begin{aligned} D_{KL}(p_X \,||\, q_X) &= \int_X p_X(\mathbf{x}) \ln \frac{p_X(\mathbf{x})}{q_X(\mathbf{x};\phi)} d\mathbf{x} && (3.1.5)\\ &= -\int_X p_X(\mathbf{x}) \ln q_X(\mathbf{x};\phi) d\mathbf{x} + \int_X p_X(\mathbf{x}) \ln p_X(\mathbf{x}) d\mathbf{x} && (3.1.6)\\ &= -\mathop{\mathbb{E}}_{p_X} \left[\ln q_X(\mathbf{x};\phi)\right] + \text{const.} && (3.1.7)\\ &= -\mathop{\mathbb{E}}_{p_X} \left[\ln p_Z(f(\mathbf{x};\phi)) + \ln \left|\frac{\partial f}{\partial \mathbf{x}}\right|\right] + \text{const.} && (3.1.8) \end{aligned}$$

The term $\int_X p_X(\mathbf{x}) \ln p_X(\mathbf{x}) d\mathbf{x}$ is constant with respect to the trainable flow parameters ϕ. Distribution $q_X(., \phi)$ which minimises the term $-\mathop{\mathbb{E}}_{p_X} \left[\ln p_Z(f(\mathbf{x};\phi)) + \ln \left|\frac{\partial f}{\partial \mathbf{x}}\right|\right]$ will minimise the divergence $D_{KL}(p_X \,||\, q_X)$, therefore the constant term can be conveniently omitted from optimisation, allowing for non-differentiable target distributions p_X.

Otherwise, when target samples are unavailable, but the target probability can be evaluated at arbitrary points in the domain — as is the case when p_X is the GW posterior — the alternative is to use the reverse KL divergence, which can be

expressed as an expectation over the approximate distribution q_X,

$$D_{KL}(q_X \,||\, p_X) = \int_X q_X(\mathbf{x}) \ln \frac{q_X(\mathbf{x};\phi)}{p_X(\mathbf{x})} d\mathbf{x} \tag{3.1.9}$$

$$= \mathop{\mathbb{E}}_{q_X} \left[\ln \frac{q_X(\mathbf{x};\phi)}{p_X(\mathbf{x})} \right] \tag{3.1.10}$$

$$= \mathop{\mathbb{E}}_{q_X} \left[\ln p_Z(f(\mathbf{x};\phi)) + \ln \left| \frac{\partial f}{\partial \mathbf{x}} \right| - \ln p_X(\mathbf{x}) \right]. \tag{3.1.11}$$

Since the reparametrisation of the latent distribution through the invertible transform f preserves normalisation, as long as the latent distribution is normalised, the normalisation constant Z of the target can be estimated over the samples from the flow as

$$\hat{Z} = \mathop{\mathbb{E}}_{q_X(\mathbf{x})} \left[\frac{\tilde{p}_X(\mathbf{x})}{q_X(\mathbf{x})} \right], \tag{3.1.12}$$

where $\tilde{p}_X = Z p_X$ is the unnormalised target. This is particularly useful in the GW case where $p_X \equiv p(\theta_{GW}|d)$ is the signal parameter posterior and $\hat{Z}$ will provide an estimate of the Bayesian evidence of the signal model.

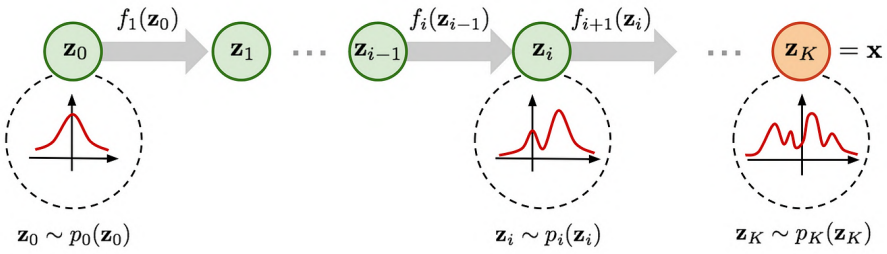


Figure 2: Normalising flow schematic.

Schematic [69] of a normalising flow transforming a simple base distribution p_0 into a complex distribution p_K, in a finite series of steps f_i. In the notation of this chapter p_0 is the latent distribution p_Z, and the final transformed distribution p_K is the normalising flow distribution q_X. The flow transform is defined in the generative direction $f : Z \to X$.

3.1.2. *Density Estimation: Forward KL Training*

Assume we have access to a set of N samples $\{\mathbf{x}_i\}$, $\mathbf{x}_i \sim p_X$, and we wish to approximate p_X with a distribution q_X obtained via a normalising flow, as defined in Eq. 3.1.3. The objective is to choose parameters ϕ which minimise the forward KL divergence between the target and the flow distribution. Given samples from p_X, we

can easily approximate expectation values over p_X as Monte Carlo integrals, so that the KL divergence becomes

$$D_{KL}(p_X \,||\, q_X) = -\mathop{\mathbb{E}}_{p_X}\left[\ln p_Z(f(\mathbf{x};\phi)) + \ln\left|\frac{\partial f}{\partial \mathbf{x}}\right|\right] + \text{const.} \tag{3.1.13}$$

$$\approx -\frac{1}{N}\sum_{i=1}^{N}\left[\ln p_Z(f(\mathbf{x}_i;\phi)) + \ln\left|\frac{\partial f(\mathbf{x}_i;\phi)}{\partial \mathbf{x}}\right|\right] + \text{const.}, \quad \mathbf{x}_i \sim p_X, \tag{3.1.14}$$

and hence an optimal approximate distribution q_X is obtained by minimizing the loss function of the form

$$\mathcal{L}(\phi) = -\frac{1}{N}\sum_{i=1}^{N}\left[\ln p_Z(f(\mathbf{x}_i;\phi)) + \ln\left|\frac{\partial f(\mathbf{x}_i;\phi)}{\partial \mathbf{x}}\right|\right], \quad \mathbf{x}_i \sim p_X \tag{3.1.15}$$

$$= -\frac{1}{N}\sum_{i=1}^{N}\ln q_X(\mathbf{x}_i;\phi), \quad \mathbf{x}_i \sim p_X. \tag{3.1.16}$$

This is often referred to as the *mean negative loglikelihood* loss, as it is simply the mean logarithm of the flow distribution q_X evaluated at the samples drawn from the target p_X.

The network parameters ϕ which minimise the loss $\mathcal{L}$ are normally optimised via means of a stochastic optimisation method, such as stochastic gradient descent (SGD) or its refinements.

3.1.2.1. *Use of normalising flows as density estimators in GW data analysis*

In the case of the GW posterior estimation, posterior samples are not available and hence forward KL divergence model training cannot be straightforwardly used for the GW signal model parameter posterior density estimation. There are numerous other use cases for normalising flow density estimation in the field of GW research. Normalising flows trained on target density samples have been successfully used in GW analyses which rely on posterior samples from the gravitational signal parameter space having been obtained prior to inference — conditional normalising flow density estimators have been used for population modeling of stellar mass [70; 71; 72; 73] and supermassive black hole binaries [74], as well as cosmological parameter inference [75].

Normalising flow density estimators have also been applied to aid stochastic sampling of the GW signal parameter posteriors — the nested sampling algorithm nessai [68] uses a normalising flow to approximate subsequent constrained prior densities, the importance nested sampler i-nessai [76] uses a hyperproposal model constructed as a

weighted sum of normalising flows, whereas the FLOWMC [77; 78] MCMC algorithm utilises a normalising flow to learn a global proposal distribution to aid intermodal jumps between local MCMC chains.

Another demonstrated use case is removal of noise glitches from data stretches containing GW signals, necessary for unbiased GW parameter recovery. In [79] a normalising flow trained on a catalog of detected glitch samples is used as an informative prior probability on glitch parameters, allowing for simultaneous stochastic Bayesian inference on a joint signal+glitch model.

3.1.3. *Variational Inference: Reverse KL Training*

In the case of the GW problem, samples from target distribution p_X given by the posterior distribution $p(\theta_{GW}|d)$ on the signal model parameters θ_{GW} are unavailable prior to inference, we are however capable of evaluating the density $p(\theta_{GW}|d)$ at arbitrary points θ_{GW} in the prior domain. In this case we can resort to training a normalising flow via reverse KL divergence — this scenario is referred to as *variational inference*, where we treat the normalising flow distribution q_X as a parametric variational distribution, which we fit to approximate the target density p_X, effectively translating Bayesian Inference of an unknown posterior distribution into an optimisation problem.

In this case, since we only have access to latent samples $\mathbf{z} \sim p_Z$, the trainable flow transform f is more naturally defined in the generative direction and is viewed as a reparametrisation of the latent space Z,

$$f(\,.\,;\phi)\colon Z \to X, \quad q_X(f(\mathbf{z};\phi)) = p_Z(\mathbf{z}) \left| \frac{\partial f(\mathbf{z})}{\partial \mathbf{z}} \right|^{-1}, \tag{3.1.17}$$

where $|\partial f/\partial \mathbf{z}|$ is the absolute value of the Jacobian determinant of the diffeomorphism f, and p_Z is the latent distribution of our choice on the space Z, commonly a Gaussian or a uniform distribution.

In order to train a normalising flow to minimise the reverse KL divergence between the flow distribution q_X and the target p_X, we express the expectation value in Eq. 3.1.10 as a Monte Carlo integral over the samples from the flow q_X,

$$D_{KL}(q_X \,||\, p_X) = \mathop{\mathbb{E}}_{q_X}\left[\ln \frac{q_X(\mathbf{x};\phi)}{p_X(\mathbf{x})}\right] \approx \frac{1}{N}\sum_{i=1}^{N} \ln \frac{q_X(\mathbf{x_i};\phi)}{p_X(\mathbf{x_i})} d\mathbf{x}, \quad \mathbf{x}_i \sim q_X \tag{3.1.18}$$

and define the training loss as

$$\mathcal{L}(\phi) \coloneqq \frac{1}{N}\sum_{i=1}^{N} \ln \frac{q_X(\mathbf{x_i};\phi)}{p_X(\mathbf{x_i})} d\mathbf{x}, \quad \mathbf{x}_i \sim q_X \tag{3.1.19}$$

$$= \frac{1}{N}\sum_{i=1} \ln \frac{q_X(f(\mathbf{z_i};\phi))}{p_X(f(\mathbf{z_i};\phi))} \quad \mathbf{z}_i \sim q_Z \tag{3.1.20}$$

$$= \frac{1}{N}\sum_{i=1} \left[\ln p_Z(\boldsymbol{z}_i) + \ln \left| \frac{\partial f(\boldsymbol{z}_i;\phi)}{\partial \mathbf{z}} \right|^{-1} - \ln p_X(f(\mathbf{z}_i;\phi)) \right] \quad \mathbf{z}_i \sim q_Z. \tag{3.1.21}$$

During training the loss is minimised via means of stochastic optimisation, and requires computations of the loss gradient as a function of the transform parameters ϕ, determined by the neural network. The gradient of the loss w.r.t. the parameters ϕ is

$$\nabla_\phi \mathcal{L} = \frac{1}{N}\sum_{i=1}^{N} \left[\nabla_\phi \ln \left| \frac{\partial f(\mathbf{z}_i,\phi)}{\partial \mathbf{z}} \right|^{-1} - (\nabla_{\mathbf{x}} \ln p_X)(f(\mathbf{z}_i;\phi)) \frac{\partial f(\mathbf{z}_i,\phi)}{\partial \phi} \right], \tag{3.1.22}$$

where $\frac{\partial f}{\partial \phi}$ is the Jacobian matrix of f as a function of parameters ϕ.

Computing the gradient in Eq. 3.1.22 requires evaluating the gradient $\nabla_{\mathbf{x}} \ln p_X = (\frac{1}{p_X}\frac{\partial p_X}{x_1}, \ldots, \frac{1}{p_X}\frac{\partial p_X}{x_d})^T$, which involves the partial derivatives of the target p_X with respect to the data space variables $\mathbf{x} = (x_1, \ldots, x_n)$. It follows that optimising the reverse KL training objective requires the target distribution p_X — in our case the gravitational wave posterior $p(\theta_{GW}|d)$ — to be differentiable with respect to the data variable $\mathbf{x}$ (θ_{GW}), unlike forward KL training. In practice these derivatives are computed using automatic differentiation, in our case enabled by implementing the GW posterior target distribution in the Jax deep learning framework. This has been made possible by the introduction of a Jax implementation of GW waveform models in the ripple project [80].

Note that in practice in the gravitational wave problem we only have access to unnormalised posterior evaluations of the target $p(\theta_{GW}|d) = p(d|\theta_{GW})p(\theta_{GW})/p(d)$, since the normalisation constant $Z = p(d)$, the Bayesian evidence, is unknown prior to inference. In the notation of this section this is equivalent to a scenario where the target satisfies $p_X = \tilde{p}_X/Z$, where $\tilde{p}_X$ not normalised. This is not of concern since the normalisation introduces a constant term into the KL divergence,

$$D_{KL}(q_X|p_X) = \mathop{\mathbb{E}}_{q_X}\left[\ln \frac{q_X(\mathbf{x};\phi)}{p_X(\mathbf{x})} \right] = \mathop{\mathbb{E}}_{q_X}\left[\ln \frac{q_X(\mathbf{x};\phi)}{\tilde{p}_X(\mathbf{x}) Z^{-1}} \right] \tag{3.1.23}$$

$$= \mathop{\mathbb{E}}_{q_X}\left[\ln \frac{q_X(\mathbf{x};\phi)}{\tilde{p}_X(\mathbf{x})} \right] + \underbrace{\mathop{\mathbb{E}}_{q_X}\left[\ln Z\right]}_{\text{const.}}. \tag{3.1.24}$$

Since the constant term drops when taking the loss gradient, the optimal solution $\check{q}$ satisfies

$$\check{q} \coloneqq \underset{q\in\mathcal{Q}}{\operatorname{argmin}} D_{KL}(q_X \,||\, p_X) = \underset{q\in\mathcal{Q}}{\operatorname{argmin}} \underset{q_X}{\mathbb{E}} \left[\ln \frac{q_X(\mathbf{x};\phi)}{\tilde{p}_X(\mathbf{x})}\right] \to p_X, \tag{3.1.25}$$

so that all evaluations of p_X in the training loss (Equation 3.1.19-3.1.21) can be safely performed up to a normalising constant, while still recovering the normalised target $\check{q} = p_X$ as the optimal solution of the problem. This conveniently allows for estimation of the normalising constant Z of the target when training completes (Equation 3.1.12). One should however further note that when resorting to unnormalised target evaluations, the true minimum of the loss is no longer at 0, but is shifted by $\ln Z$, making it more difficult to asses algorithm convergence, with Z unknown prior to training.

3.1.3.1. *Variational inference and Evidence Lower Bound*

In the context of Bayesian inference the target density p is usually a posterior distribution $p(x|y) = p(y|x)p(x)/p(y)$ of model parameters x under some observed data y. In this case, minimising the divergence $D_{KL}(q \,||\, p(x|y))$ is equivalent to maximising a lower bound on the Bayesian evidence $p(y)$, the probability associated with observing the data y under our data model — a quantity which is usually difficult to compute, while being essential for model selection, as is the case with GW inference.

The reverse KL-divergence satisfies

$$\begin{aligned}
D_{KL}(q(x) \,||\, p(x|y)) &= \underset{q(x)}{\mathbb{E}} \left[\ln \frac{q(x)}{p(x|y)}\right] && (3.1.26)\\
&= \underset{q(x)}{\mathbb{E}} \big[\ln q(x) - \ln p(x|y)\big] && (3.1.27)\\
&= \underset{q(x)}{\mathbb{E}} \big[\ln q(x) - \ln p(x,y) + \ln p(y)\big] && (3.1.28)\\
&= \underset{q(x)}{\mathbb{E}} \big[\ln q(x) - \ln p(x,y)\big] + \underset{q(x)}{\mathbb{E}} \big[\ln p(y)\big] && (3.1.29)\\
&= \underset{q(x)}{\mathbb{E}} \big[\ln q(x) - \ln p(x,y)\big] + \ln p(y), && (3.1.30)
\end{aligned}$$

where we have used that $p(x|y) = p(x,y)/p(y)$ and that the evidence $p(y)$ is a constant. Rearranging this for the Bayesian evidence, $p(y)$, we get

$$\begin{aligned}
\ln p(y) &= D_{KL}(q(x) \,||\, p(x|y)) - \underset{q(x)}{\mathbb{E}} \big[\ln q(x) - \ln p(x,y)\big] && (3.1.31)\\
&= D_{KL}(q(x) \,||\, p(x|y)) + \underset{q(x)}{\mathbb{E}} \big[\ln p(x,y) - \ln q(x)\big], && (3.1.32)
\end{aligned}$$

and since KL-divergence is non-negative, we can put a lower bound on the logarithm of the evidence,

$$\mathrm{ELBO}(q) \coloneqq \mathop{\mathbb{E}}_{q(x)}\big[\ln p(x,y) - \ln q(x)\big] \leq \ln p(y). \tag{3.1.33}$$

The true evidence is recovered in the limit when $q \to p$, whereas the further q is from the target posterior p, the lower and further from the truth is the bound on the evidence.

Minimizing the reverse KL-divergence is then equivalent to maximizing the evidence lower bound $\mathrm{ELBO}(q)$ over the family of approximate distributions $q \in \mathcal{Q}$. The optimal approximation $\check{q} \in \mathcal{Q}$ of the target posterior $p(x|y)$ w.r.t. the reverse KL-divergence can be written as

$$\check{q} \coloneqq \mathop{\mathrm{argmin}}_{q\in\mathcal{Q}} D_{KL}(q(.)\,||\,p(.|y)) = \mathop{\mathrm{argmax}}_{q\in\mathcal{Q}} \mathrm{ELBO}(q). \tag{3.1.34}$$

Rewriting the joint distribution $p(x,y)$ as the product of the likelihood and the prior, $p(x,y) = p(y|x)p(x)$, we have

$$\begin{aligned}
\mathrm{ELBO}(q) &= \mathop{\mathbb{E}}_{q(x)}\big[\ln p(x,y) - \ln q(x)\big] && (3.1.35)\\
&= \mathop{\mathbb{E}}_{q(x)}\big[\ln p(y|x) + \ln p(x) - \ln q(x)\big] && (3.1.36)\\
&= \mathop{\mathbb{E}}_{q(x)}\big[\ln p(y|x)\big] - \mathop{\mathbb{E}}_{q(x)}\left[\ln \frac{q(x)}{p(x)}\right] && (3.1.37)\\
&= \mathop{\mathbb{E}}_{q(x)}\big[\ln p(y|x)\big] - D_{KL}(q(x)\,||\,p(x)). && (3.1.38)
\end{aligned}$$

That is, $\mathrm{ELBO}(q)$ can be expressed as the difference between the expectation of the likelihood and the KL-divergence between q, the approximated posterior, and the prior, $p(x)$. Therefore maximizing $\mathrm{ELBO}(q)$ drives the approximated posterior close to the prior.

3.1.3.2. *Variational Inference within GW analysis*

At the time of writing this report, successful use of variational inference within the context of GW data analysis has been limited to population modeling. In [81] a block neural autoregressive flow model is used as a variational approximation of a GW population model posterior, providing an efficient tool for population model comparison and sampling. The approximated population posteriors were unimodal and lacked complex correlation structure, aiding the successful results. The discrepancies between the trained variational model posterior and the target were noted mainly in the distribution tails — variational models underestimated distribution tails. These

discrepancies were removed using smoothed importance sampling, correcting the variational model within its support — a method that can correct the model for small deviations from the target, however not for missed modes.

At the time there is no published work on applications of variational inference with normalising flows to full CBC posterior inference, other than this thesis.

3.1.4. *Simulation Based Inference (SBI)*

Simulation Based Inference provides an alternative way of using normalising flows for Bayesian Inference, which bypasses the problem of lack of target training samples by resorting to simulation of prior parameters and simulated data pairs, used for training a conditional density estimator via forward KL divergence.

Assume we are interested in sampling the posterior distribution $p(\mathbf{x}|\mathbf{y}=\mathbf{y_0})$ of the variable $\mathbf{x}$ (typically model parameters) conditioned on a variable $\mathbf{y}$ (typically experimental data). From Bayes' theorem the posterior distribution satisfies

$$p(\mathbf{x}|\mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{x})p(\mathbf{x})}{p(\mathbf{y})}. \tag{3.1.39}$$

Conventional stochastic sampling methods such as MCMC as well as variational inference require frequent computationally costly evaluations of the likelihood $p(\mathbf{y}|\mathbf{x})$. One can bypass likelihood evaluations with the use of normalising flows, as long as we are able to produce samples from the joint distribution $p(\mathbf{x}, \mathbf{y})$. In literature this is referred to as *likelihood-free inference* or *simulation-based inference* [82; 83; 24].

The idea then is to approximate the posterior $p(\mathbf{x}|\mathbf{y})$ with a flow based conditional model distribution $q(\mathbf{x}|\mathbf{y};\phi)$,

$$p(\mathbf{x}|\mathbf{y}) \approx q(\mathbf{x}|\mathbf{y};\phi) = p_Z(f(\mathbf{x};\mathbf{y},\phi)) \left|\frac{\partial f(\mathbf{x};\mathbf{y},\phi)}{\partial \mathbf{x}}\right|. \tag{3.1.40}$$

The flow now depends on both the trainable neural network parameters ϕ, but also on the conditional variable $\mathbf{y}$; however for a fixed set of parameters $\phi = \phi_0$, $\mathbf{y} = \mathbf{y}_0$, the normalising transform $f(\mathbf{x};\mathbf{y}=\mathbf{y}_0,\phi=\phi_0)\colon X \to Z$ is still a diffeomorphism and hence is invertible and has a tractable Jacobian.

The optimal parameters of the neural network $\hat{\phi}$ are chosen so that they minimise the forward KL divergence between the target posterior $p(\mathbf{x}|\mathbf{y})$ and the approximation $q(\mathbf{x}|\mathbf{y};\phi)$ with respect to the data realisations $\mathbf{y}$, which is achieved by minimising the expectation value of the forward KL divergence with respect to the distribution of

the data, the evidence $p(\mathbf{y})$, following [83],

$$\hat{\phi} = \underset{\phi}{\operatorname{argmin}} \underset{p(\mathbf{y})}{\mathbb{E}} \left[D_{KL}(p(\mathbf{x}|\mathbf{y}) \,||\, q(\mathbf{x}|\mathbf{y};\phi)) \right] \tag{3.1.41}$$

$$= \underset{\phi}{\operatorname{argmin}} \underset{p(\mathbf{y})p(\mathbf{x}|\mathbf{y})}{\mathbb{E}} \left[-\ln q_X(\mathbf{x}|\mathbf{y};\phi) \right] \tag{3.1.42}$$

$$= \underset{\phi}{\operatorname{argmin}} \iint -\underbrace{p(\mathbf{x}|\mathbf{y})p(\mathbf{y})}_{p(\mathbf{x},\mathbf{y})} \ln q_X(\mathbf{x}|\mathbf{y};\phi) d\mathbf{x} d\mathbf{y} \tag{3.1.43}$$

$$= \underset{\phi}{\operatorname{argmin}} \underset{p(\mathbf{x},\mathbf{y})}{\mathbb{E}} \left[-\ln q_X(\mathbf{x}|\mathbf{y};\phi) \right]. \tag{3.1.44}$$

Assuming we can draw prior samples $\mathbf{x}_i \sim p(\mathbf{x}_i)$ and simulate the corresponding experimental data $\mathbf{y}_i \sim p(\mathbf{y}|\mathbf{x} = \mathbf{x}_i)$, the pairs $(\mathbf{x}_i, \mathbf{y}_i)$ will be samples from the joint distribution $p(\mathbf{x}, \mathbf{y}) = p(\mathbf{y}|\mathbf{x})p(\mathbf{x})$. Given a set of N such samples we can approximate the expectation as a Monte Carlo integral and hence define the loss function as

$$\mathcal{L}(\phi) = -\frac{1}{N} \sum_{i=1}^{N} \ln q_X(\mathbf{x}_i|\mathbf{y}_i;\phi) \tag{3.1.45}$$

$$= -\frac{1}{N} \sum_{i=1}^{N} \left[\ln p_Z(f(\mathbf{x_i};\mathbf{y}_i,\phi)) + \ln \left| \frac{\partial f(\mathbf{x}_i;\mathbf{y}_i,\phi)}{\partial \mathbf{x}} \right| \right]. \tag{3.1.46}$$

One can think of optimising this target as simultaneously training a continuous family of normalising flows parametrised by the data realisations $\mathbf{y}$, where the loss function KL divergence contribution from each such flow is weighted by the evidence $p(\mathbf{y})$, i.e. the probability of observing the data $\mathbf{y}$ under no assumptions on the model parameters.

3.1.4.1. *SBI for GW inference*

The SBI framework provides an alternative application of NFs to GW posterior estimation. Even though the target samples from the GW posterior $p(\theta|d)$ are unavailable, one can obtain samples (θ, d) from the joint distribution $p(\theta, d) = p(d|\theta)p(\theta)$ by sampling θ from the prior $p(\theta)$ and simulating corresponding interferometer data $d = h(\theta)+n$. Since it also holds that $p(\theta, d) = p(\theta|d)p(d)$, one can use the joint samples to train a conditional normalising flow $q(\theta|d;\phi)$ to approximate the target posterior $p(\theta|d)$ for all data realisations d. Unlike training an unconditional flow $q(\theta;\phi)$ to approximate $p(\theta|d)$ for a single detected data vector d within the framework of VI, SBI flow training is not limited by the use of divergence statistics evaluated on the samples from the flow.

The groundwork for use of SBI within the GW field has been laid by the Dingo [84] project, a likelihood-free posterior estimation algorithm where a conditional normalising flow is trained on samples from the parameter prior and the corresponding

simulated GW strain data — this enables so-called *amortised* inference, where once the conditional flow is trained, the parameter posterior for any new stretch of GW strain data can be produced without further training, and, in theory, only at the cost of evaluating the flow. So far DINGO has focused on parameter estimation for the current ground-based detectors, but it has been shown that a similar SBI model can been trained to analyse overlapping signals of the types we expect to see in the next generation detectors [85].

3.2. KULLBACK-LEIBLER DIVERGENCE

3.2.1. *Definition and Properties*

The statistical distance measure of the relative proximity of two probability distributions most widely used within the context of normalising flows is the Kullback-Leibler (KL) divergence. Given two probability distributions P and Q of a continuous random variable $x \in X$, with densities p and q respectively, the KL-divergence of Q relative to P is defined [86] as

$$D_{KL}(P \,||\, Q) = \int_X p(x) \ln \frac{p(x)}{q(x)} dx \tag{3.2.1}$$

$$= \mathop{\mathbb{E}}_{p(x)} \left[\ln \frac{p(x)}{q(x)} \right]. \tag{3.2.2}$$

The KL-divergence $D_{KL}(P \,||\, Q)$ can be expressed as the difference between the cross-entropy $H(P \,||\, Q)$ and the self entropy of P,

$$D_{KL}(P \,||\, Q) = -\int_X p(x) \ln q(x) + \int_X p(x) \ln p(x) = H(P \,||\, Q) - H(P \,||\, P), \tag{3.2.3}$$

which can be intuitively interpreted as the expected (under P) difference between the information encoded by P and Q. The units of KL-divergence as defined in Eq. 3.2.1 are nats, units of information.

In the context of normalising flows we always work with continuous random variables and their densities, so for the simplicity of notation let $D_{KL}(p \,||\, q) \coloneqq D_{KL}(P \,||\, Q)$.

It is important to note that the KL divergence is not a metric on the space of distributions, as it does not satisfy the triangle inequality nor is it symmetric.

Proposition 3.2.1 (Properties of KL-divergence)**.** *Let p and q be probability densities of the random variable $x \in X$. The KL-divergence is*

1. *non-symmetric, that is $D_{KL}(p \,||\, q) \neq D_{KL}(q \,||\, p)$,*
2. *non-negative, $D_{KL}(p \,||\, q) = 0 \iff p = q$, and $D_{KL}(p \,||\, q) > 0$ otherwise.*

Proof.

1. Clear from the definition.

2. Follows from the *Gibb's inequality*, stating that the entropy of a random variable is greater or equal to the cross-entropy between two random variables:

$$\int_X p(x) \ln p(x) dx \geq \int_X p(x) \ln q(x) dx. \tag{3.2.4}$$

   The KL-divergence is the difference between these terms.

□

Due to the non-symmetry of the KL divergence, the approximate distributions favoured by minimising forward or reverse KL divergence will be different – the same pair of distributions can have low forward and high reverse KL divergence, the opposite can also be true. This is important in the light of the discussions in the previous sections — depending on whether we are using forward or the reverse KL divergence as a training objective for a normalising flow, different approximate flow model distributions will be favoured by the optimisation.

3.2.2. *Example: KL divergence between two univariate Gaussian distributions*

In the case of two univariate Gaussian distributions,

$$p = \mathcal{N}(\mu_1, \sigma_1), \quad q = \mathcal{N}(\mu_2, \sigma_2), \tag{3.2.5}$$

the KL divergence $D_{KL}(p \,||\, q)$ can be easily derived as

$$D_{KL}(p \,||\, q) = \log \frac{\sigma_1}{\sigma_2} + \frac{\sigma^2 + (\mu_1 - \mu_2)^2}{2\sigma_2^2} - \frac{1}{2}. \tag{3.2.6}$$

We can see from Eq. 3.2.6 that the even in this simple case the KL divergence is not symmetric, unless the two Gaussian distribution have equal variance $\sigma_1 = \sigma_2$. In general $D_{KL}(p \,||\, q) \neq D_{KL}(q \,||\, p)$, and as we can see in Figure 3, forward KL divergence rewards the distribution wider than the target, whereas the reverse KL divergence prefers a distribution which does not exceed the target in the tails. This behaviour extends to more general distributions, and constitutes examples of *mode-covering* and *zero-forcing* respectively, as detailed in section 3.2.3.

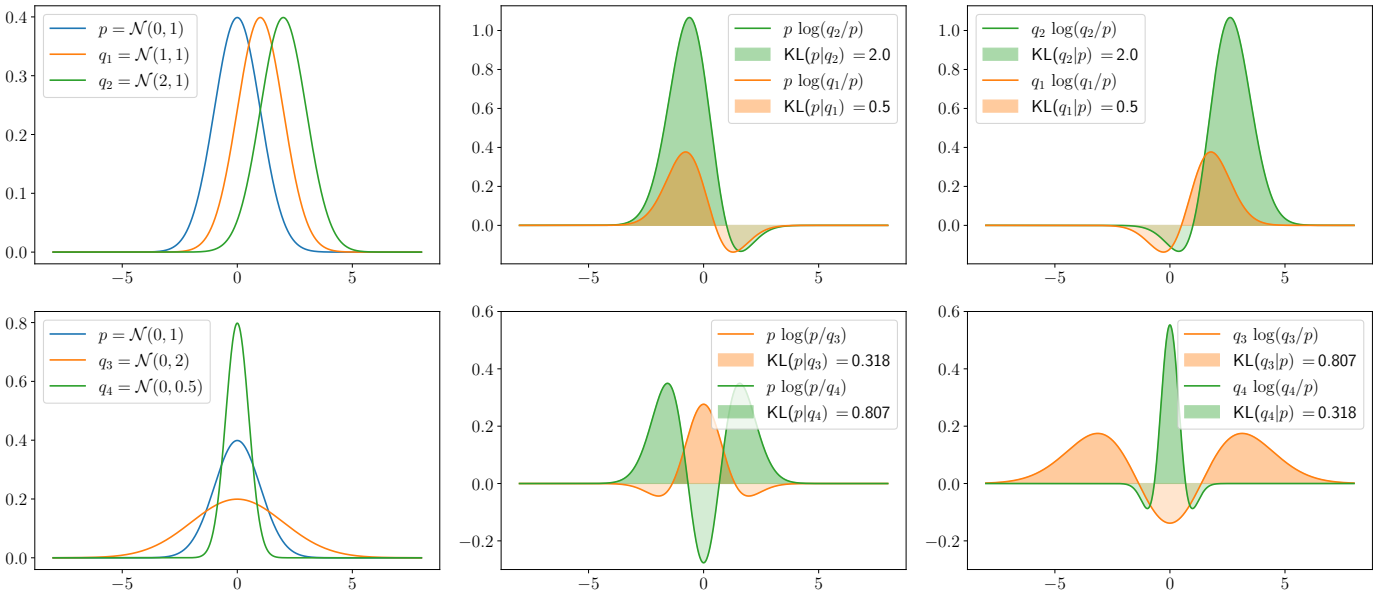


Figure 3: KL divergence between Gaussian distributions.

Forward and reverse KL divergences between pairs of univariate Gaussian distributions. The forward KL divergence is the integral of the curves in the centre panel, while the reverse KL divergence is the integral of the curves in the right panel. In the top row the distributions differ by a translation, which in this case results in a symmetry between the forward and reverse KL divergence. In the bottom row the variance of the distributions is varied, resulting in an asymmetry of KL divergence, that is in general $D_{KL}(p\,||\,q) \neq D_{KL}(q\,||\,p)$.

3.2.3. *Forward vs Reverse KL divergence optimisation*

Considering the two optimisation problems, associated to minimizing the forward and reverse KL divergence respectively,

$$\text{Forward KL:} \quad \hat{q} := \operatorname*{argmin}_{q\in\mathcal{Q}} D_{KL}(p\,||\,q) = \operatorname*{argmin}_{q\in\mathcal{Q}} \mathop{\mathbb{E}}_{p(x)}\left[\ln \frac{p(x)}{q(x)}\right] \tag{3.2.7}$$

$$\text{Reverse KL:} \quad \check{q} := \operatorname*{argmin}_{q\in\mathcal{Q}} D_{KL}(q\,||\,p) = \operatorname*{argmin}_{q\in\mathcal{Q}} \mathop{\mathbb{E}}_{q(x)}\left[\ln \frac{q(x)}{p(x)}\right] \tag{3.2.8}$$

we can establish some intuition about the properties of the approximations q of p favoured in the course of the optimisation in each case.

In the case of forward KL, we have that

$$q(x) \to 0 \implies \ln \frac{p(x)}{q(x)} \to \infty, \quad p(x) \neq 0. \tag{3.2.9}$$

hence the optimisation will disfavour configurations where $q(x)$ is small w.r.t. $p(x)$ – this is referred to as *zero avoiding*. Distributions $q \in \mathcal{Q}$ which overestimate p will be preferred; the approximate distribution q will be *mode-covering*. On the other hand,

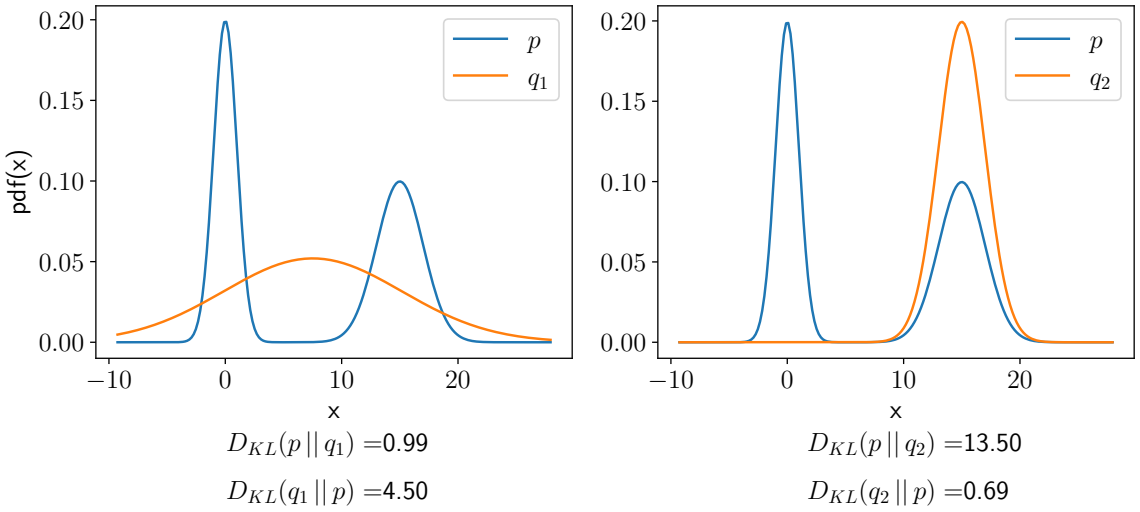


Figure 4: KL divergence and multimodal distributions.

Forward and reverse KL divergences for a bimodal Gaussian mixture target approximated by a unimodal Gaussian. Minimising forward KL divergence promotes mode-covering behaviour (left), whereas minimising reverse KL divergence promotes mode-seeking behaviour (right), given a model distribution not flexible enough to adequately match the target multimodality.

in the case of reverse KL,

$$p(x) \to 0 \implies \begin{cases} \ln \frac{q(x)}{p(x)} \to \infty, & q(x) > 0. \\ \ln \frac{q(x)}{p(x)} = 1, & q(x) = 0. \end{cases} \tag{3.2.10}$$

hence optimisation will reward configurations where $q(x) = 0$ whenever $p(x)$ is very small – this is referred to as *zero forcing*. Distributions $q \in \mathcal{Q}$ which underestimate p will be preferred – this might express itself by the q locking on one of the modes of p if p was multimodal; q will tend to be *mode-seeking*.

The above issues can affect both the sequence of approximate distributions q considered in the course of the optimisation process, as well as the optimal solution to which the algorithm converges. By definition of the KL divergence, for an unconstrained distribution q the true minima of both the reverse and forward divergence will result in an optimal solution $q = p$ (provided p is normalised, or $q = p/Z$ otherwise), however in practice q will be of limited expressiveness, constrained by its parametric functional form, so the optimal solution q in all but simple problems will be an approximation, which may be biased by the described behaviours.

Training with a mode-covering divergence is therefore preferable in the GW application, however direct application of the forward KL optimisation would require access to target posterior samples, which are unavailable prior to training. If reverse KL divergence is chosen, careful consideration must be given to the optimisation algo-

rithm as well as to the expressiveness of the variational model q, so that local minima suffering from mode collapse can be avoided during training.

3.3. NORMALISING FLOW ARCHITECTURES

3.3.1. *Coupling flows*

Coupling flows [23; 24] are a highly expressive type of flows with a simple design. The idea is to factorise the domain X of the diffeomorphism f as a product of two subspaces $X = X_A \times X_B$, so that every point $x \in X$ can be written as $x = (x_A, x_B)$ where $x_A \in X_A$ and $x_B \in X_B$, and condition the flow on this factorisation. The flow is built up of *coupling layers* — within each such layer x_B is left unchanged, i.e. transformed via an identity function, whereas x_A is transformed via a bijection h_ϕ, where ϕ is a function of x_B, effectively constructing $p(x_A \,|\, x_B)$.

More formally, each coupling transform $f\colon X = X_A \times X_B \to X$ is defined by a parametric family of bijections $h_\phi\colon X_A \to X_A$ parametrised by $\phi \in \Phi$, and an arbitrary function $\Theta\colon X_B \to \Phi$, called the *conditioner*. We then define the coupling transform f via the mapping

$$x = (x_A, x_B) \mapsto (h(x_A; \Theta(x_B)), x_B) = (y_A, y_B), \tag{3.3.1}$$

where $h(\,\cdot\,; \phi) \coloneqq h_\phi$.

The Jacobian of the coupling transform f is then, by construction, triangular with diagonal blocks given by the Jacobian of h_ϕ as a function on X_A and the identity matrix, so that the Jacobian determinant of f is simply the Jacobian determinant of h_ϕ,

$$\mathbf{J}_f = \begin{bmatrix} \mathbf{J}_h & \mathbf{A} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}, \quad \det(\mathbf{J}_f) = \det(\mathbf{J}_h) \times \det(\mathbf{I}) = \det(\mathbf{J}_h), \tag{3.3.2}$$

$$\mathbf{J}_h = \prod_{i=1}^{d} \frac{\partial h_\phi^i}{\partial x_i}, \quad x_A = (x_1, \ldots, x_d), \quad h_\phi = (h_\phi^1, \ldots, h_\phi^d), \tag{3.3.3}$$

which is easily computed provided the partial derivatives of h_ϕ are available. The *coupling function* h_ϕ is usually defined elementwise with one-dimensional *bijectors*, h_ϕ^i with easy to compute derivatives. The Jacobian determinant does not depend on the conditioner Θ, so the conditioner function can be arbitrarily complex and is usually a neural network. It is the complexity of the conditioner which grants coupling flows their expressive power.

Invertibility of the bijector h suffices to invert the coupling layer, and the inversion requires only a single pass through the conditioner network Θ. Given (y_A, y_B), we

can invert the mapping via

$$(y_A, y_B) \mapsto (h^{-1}(y_A; \Theta(y_B)), y_B) = (x_A, x_B). \tag{3.3.4}$$

The arbitrary factorisation $X = X_A \times X_B$ limits the expressiveness of the flow, so a coupling flow is normally constructed as a composition of a number of subsequent coupling layers, between which the dimensions of the data space X are permuted, allowing for an expression of increasingly complex dependencies and correlations.

3.3.2. *Masked Autoregressive Flows*

A commonly used alternative to the Coupling Flow is a Masked Autoregressive Flow (MAF) [87], where the idea is to fully factorise the D-dimensional data space $X = \prod_{i=1}^{D} X_i$ into a product of its 1-dimensional subspaces X_i, and build a flow q_X from the one-dimensional conditional densities,

$$q_Y(\mathbf{y}) = \prod_{i=1}^{D} q_{Y_i}(y_i \,|\, \mathbf{y}_{1:i-1}). \tag{3.3.5}$$

Each of the conditional densities is generated by a bijective *autoregressive transform* f_i parametrised by an arbitrary conditioner $\Theta_i \colon Y_{1:i-1} \to \Phi$,

$$x_1 \mapsto f_1(x_1; \phi_1) = y_1 \tag{3.3.6}$$
$$x_i \mapsto f_i(x_i; \Theta_i(\mathbf{y}_{1:i-1})) = y_i. \tag{3.3.7}$$

The generative $X \to Y$ direction requires D passes through the network, whereas the flow inversion, $Y \to X$, neeeded for density evaluation,

$$y_1 \mapsto f_1^{-1}(y_1; \phi_1) = x_1 \tag{3.3.8}$$
$$y_i \mapsto f_i^{-1}(y_i; \Theta_i(\mathbf{y}_{1:i-1})) = x_i, \tag{3.3.9}$$

can be performed in parallel. Autoregressive flows are implemented using masked conditioners, which given a data sample $\mathbf{y} \in X$ return the coefficients $\Theta_i(\mathbf{y}_{1:i-1})$ in a single network pass.

Depending on whether the flow is used for sampling or density evaluation, we can instead define the generative direction as the *inverse autoregressive flow*, with transforms conditioned directly on the domain X, that is

$$x_1 \mapsto f_1(x_1; \phi_1) = y_1 \tag{3.3.10}$$
$$x_i \mapsto f_i(x_i; \Theta_i(\mathbf{x}_{1:i-1})) = y_i. \tag{3.3.11}$$

In this case the generative direction used for sampling will require a single network pass, whereas density estimation requires D network passes.

By design, the overall transform $f\colon X \to X$, $f = (f_1, \dots f_D)$ will have a triangular Jacobian matrix with Jacobian determinant

$$\det(\mathbf{J}_f) = \prod_{i=1}^{D} \frac{\partial f^i_{\phi_i}}{\partial x_i}, \quad X = (x_1, \dots, x_D), \quad f = (f^1_{\phi_1}, \dots, f^D_{\phi_D}). \tag{3.3.12}$$

The autoregressive flow is by design more expressive than the coupling flow, as a single autoregressive transform allows for conditioning between any two data variables, however it suffers from a trade off in speed between the sampling and density estimation direction, as one will be D times slower than the other, where D is the dimensionality of the data space.

3.4. NORMALISING FLOW BIJECTORS

Theoretically, a normalising flow can be built using any parametric diffeomorphism f, however practically the transforms used in the field are built using elementwise 1-dimensional parametric bijections, $f = (f_1, \dots, f_D)$ treating the domain X as a subspace of $\mathbb{R}^n$, $X = \prod_{i=1}^{N} X_i$ with $X_i \cong R$, or in the bounded case $X_i \cong [a, b] \subset \mathbb{R}$. To be practically useful we need the 1-dimensional bijectors f_i to be computationally efficient to evaluate and invert, as well as be expressive enough to approximate, ideally arbitrarily complex monotonic transforms.

3.4.0.1. *Affine Bijectors*

One of the simplest 1-dimensional bijectors used within the context of normalising flows are affine transformations,

$$f_i(x_i) = \alpha_i x_i + \beta_i, \quad \alpha_i \coloneqq \exp(\hat{\alpha}_i), \quad x_i, \hat{\alpha}_i, \beta_i \in \mathbb{R} \tag{3.4.1}$$

consisting of a scaling by α_i and translation by β_i. These are invertible provided $\alpha_i \neq 0$, which is achieved by taking unconstrained parameters $\hat{\alpha}_i, \beta_i \in \mathbb{R}$ and defining $\alpha_i \coloneqq \exp(\hat{\alpha}_i)$, resulting in a positive scaling factor, and hence a monotonic affine transform.

Given $y_i = \alpha x_i + \beta$, the inverse transform is easily computed as

$$f_i^{-1}(y_i) = \frac{y_i - \beta_i}{\alpha_i} = x_i. \tag{3.4.2}$$

The coefficients α_i and β_i will be in general functions of the data variables $(x_1, x_2, \dots, x_{i-1}, x_{i+1}, \dots x_D)$ defined via a neural network. These dependencies will vary depending on the chosen flow architecture (e.g. coupling layers or MAF), but, as explained in Section 3.3, for ease of computation the flow is usually constructed so that the Jacobian of the D-dimensional element-wise affine transform $f = (f_1, \dots f_D)$

is triangular. The Jacobian determinant of f where each f_i is an affine transform is then simply

$$\det(\mathbf{J}_f) = \prod_{i=1}^{D} \frac{\partial f_i}{\partial x_i} = \prod_{i=1}^{D} \alpha_i. \tag{3.4.3}$$

The affine bijectors are used within normalising flow models such as NICE [88], and the affine coupling layer based RealNVP [89]. RealNVP flow models are utilised within the normalising-flow aided nested sampler NESSAI [68], developed for GW parameter estimation.

The obvious appeal of the affine transforms is their simplicity and computational efficiency due to exact inversion and fast derivative computation. The expressiveness of flows built using affine transforms is however limited, resulting in the need of composing many such transforms to represent more complex distributions [24].

3.4.0.2. *Neural Spline Flows*

The state of the art normalising flow bijectors are constructed using elementwise monotonic rational-quadratic splines as proposed in [90]. Flows constructed with these bijectors are referred to as Rational Quadratic Neural Spline Flows (RQ-NSF).

Given K monotonically increasing *knots* $\{(x^{(k)}, y^{(k)})\}_{k=0}^{K}$ and K-1 derivatives $\{\delta^{(k)}\}_{k=1}^{K-1}$, the transform is constructed as a monotonic piecewise rational-quadratic spline passing through the knots, and with the derivatives at the knots specified by the parameters δ.

In the k^{th} bin, we can express the rational quadratic transform as

$$s_k(x) = \frac{\alpha^{(k)}}{\beta^{(k)}}(\xi) = y^{(k)} + \frac{(y^{k+1} - y^{(k)})\left[m^{(k)}\xi^2 + \delta^{(k)}\xi(1-\xi)\right]}{m^{(k)} + \left[\delta^{(k+1)} + \delta^{(k)} - 2s^{(k)}\right]\xi(1-\xi)}, \tag{3.4.4}$$

$$\xi = \frac{x - x^{(k)}}{x^{(k+1)} - x^{(k)}}, \quad m^{(k)} = \frac{y^{(k+1} - y^{(k)}}{x^{(k+1)} - x^{(k)}}. \tag{3.4.5}$$

The domain of the spline is determined by the boundary knots $(x^{(0)}, y^{(0)}) = (-B, -B)$, $(x^{(K)}, y^{(K)}) = (B, B)$, for some $B \in \mathbb{R}$, and the spline can be extended linearly beyond the interval $[-B, B]$ by setting the transform on $(-\infty, -B) \cup (B, \infty)$ to the identity transform while requiring $\delta^{(0)} = \delta^{(K)} = 1$.

The full 1-dimensional transform can be then written as

$$f\colon \mathbb{R} \to \mathbb{R}, \quad f(x) = \begin{cases} s_k(x), & x \in [x^{(k)}, x^{(k+1)}]. \\ x, & \text{otherwise} \end{cases} \tag{3.4.6}$$

The transform is monotonic and continuously once-differentiable by construction. The derivative has a rational closed form expression and the transform can be analytically

inverted by determining the relevant bin and solving a quadratic equation. It is also universal, in the sense that theoretically it can approximate any monotonic function on the interval $[-B, B]$ with arbitrary precision, provided sufficient number of bins [90].

A Neural Spline Flow with N bins will have up to $3N+1$ free trainable parameters – N bin widths, N bin heights and $N+1$ derivatives at the knots.

One can easily obtain a rational quadratic spline diffeomorphism of the circle $S^1 = [0,1]/0 \sim 1$, the so-called *circular spline*, by setting $\delta^{(0)} = \delta^{(K)}$, and $(x^{(0)}, y^{(0)}) = (0,1)$, $(x^{(K)}, y^{(K)}) = (1,1)$, while not extending the transform beyond the $[0,1]$ domain. This would result in a transform naturally suited to modelling distributions on any periodic variables.

Even though the spline transform is differentiable by construction, it is only once-differentiable at the spline knots and the expressiveness of flow between the knots is limited by the expressiveness of a rational quadratic transform.

3.5. NEURAL NETWORKS AND COMPUTATIONAL TOOLS

3.5.1. *MLP*

The simplest neural network architecture is an fully-connected feedforward NN, also known as Multi-Layer Perceptron (MLP),

$$f_\theta \colon \mathbb{R}^{d_0} \to \mathbb{R}^{d_N}, \quad x_0 \mapsto x_1 \mapsto \cdots \mapsto x_N = y \tag{3.5.1}$$

$$x_n = \phi(W_n x_{n-1} + b_n), \quad \forall\, n \geq 1, \tag{3.5.2}$$

is a function mapping the input layer $x_0 = (x_{01}, \ldots x_{0d_0})$ to an output $x_N = (x_{N1}, \ldots x_{Nd_N})$ by passing through a sequence of hidden layers of dimensions $\{d_1, d_2, \ldots d_{N-1}\}$. The output x_n of layer n is obtained from the previous layer output x_{n-1} by first multiplying by a $d_n \times d_{n-1}$ weights matrix W_n, adding a bias vector $b_n = (b_{n1}, \ldots, b_{nd_n})$ and transforming the result by a chosen activation function ϕ. The trainable network parameters $\theta = (W_1, b_1, \ldots W_N, b_N)$ are the weight matrices and bias vectors. The activation function ϕ is the source of nonlinearity in the mapping.

It is common to represent a neural network as a graph, where each connection is assigned a scalar multiplicative weight, and where nodes represent a summation operation followed by an addition of a scalar bias and subsequent transformation by an activation function. Nodes of a neural network are referred to as neurons. Given enough layers and neurons, an MLP can approximate arbitrary functions.

3.5.2. *Training a network - backpropagation and stochastic optimisation*

The network parameters are optimised by minimising a chosen objective loss function of the network parameters, $L(\theta)$ – in the case of normalising flows the target objective is the KL divergence between the target distribution and its normalising flow approximation. The optimisation is usually performed stochastically, by means of stochastic gradient descent (SGD) or one of its refined variants such as the Adam optimiser [91]. In the standard SGD algorithm, given the current network parameters θ, the new parameters are obtained by taking a step in the direction of the negative gradient of the loss,

$$\theta_{\text{new}} = \theta - \alpha \frac{1}{k} \sum_{i=1}^{k} \nabla L_i(\theta) \approx \theta - \alpha \nabla L(\theta) \quad L_i(\theta) = L(\theta)|_{x=x^{(i)}} \,, \tag{3.5.3}$$

where α is referred to as the learning rate. The gradient of the loss function is obtained stochastically — as an average over a batch of examples $\{x^i\}_{i=1}^k$, and the individual gradients are obtained via the means of *backpropagation* (reverse-mode autodifferentiation). Given an input output pair $(x, y = f_\theta(x))$ obtained in a forward pass of x through the network, derivatives of L at θ w.r.t. the network weights and biases are obtained by propagating the derivatives along each network connection backwards through the network via the chain rule.

3.5.3. *ResNet*

Since gradients of the network weights are calculated using backpropagation, gradients of weights in deep layers of the network are computed using large numbers of gradient multiplications — this can lead to gradients vanishing when the subsequent multiplied gradients are small (many standard activation functions have gradients constrained to [0,1]) or in the opposite extreme gradients can grow exponentially (explode) when the multiplications in the chain are large. A residual neural network is a modification of the simple fully-connected MLP aimed at preventing vanishing and exploding gradients. Assume we have a stack of nonlinear layers and the desired mapping of this stack is $x \mapsto h(x)$. Instead of letting the stack learn the mapping h, the stack learns the residual mapping $f(x) = h(x) - x$. This is achieved by introducing identity *skip connections*, that add the stack input to the stack output, skipping the layers in between,

$$x \mapsto f(x) \mapsto f(x) + x = h(x). \tag{3.5.4}$$

The learnt residual mapping f represents a stack of fully connected layers and is referred to as a residual block. The identity skip connection omits the stack f and sums the input of the stack with the output. A residual network (ResNet) is a network obtained by stacking residual blocks.

3.6. JAX AND GPU ACCELERATION

Jax [92] is a framework for accelerating Python+Numpy based programs, particularly well-suited to machine learning algorithms due to its compatibility with automatic differentiation. It uses high-level tracing together with the XLA compiler for optimizing code for high-performance execution on CPUs and GPUs to create accelerated versions of pure-and-statically-composed (PSC) subroutines used within Python+Numpy programs. Tracing creates function graphs compatible with autodifferentiation, effectively allowing for taking derivatives of arbitrary Python+Numpy PSC routines, allowing for building accelerated neural networks and differentiable functions.

In the case of GW applications, introduction of Jax allows for construction of GPU-accelerated waveform models and likelihoods, which can both speed up existing search and parameter estimation methods, as well as enable posterior inference via variational inference. The reverse KL divergence optimisation used within the framework of variational inference requires taking posterior derivatives with respect to the GW signal parameters – these derivatives can be easily obtained via autodifferentiation of a Jax traced posterior; taking posterior derivatives is otherwise not easily achievable.

3.7. NORMALISING FLOWS ON RIEMANNIAN MANIFOLDS

So far we have been assuming, explicitly or not, that a normalising flow transform is usually defined on a Euclidean domain $\mathbb{R}^n$, $f : \mathbb{R}^n \to \mathbb{R}^n$. This need not be the case, as all of the definitions readily extend to the case where $f : X \to X$ is a diffeomorphism of a Riemannian manifold X of an arbitrary geometry. This means that normalising flows can be used to define parametric distributions on non-trivial domains — for example a spherical S^2 domain for the gravitational wave sky location parameter pair, a circular S^1 domain for any periodic parameter, or even a more complex domain such as the rotation group $SO(3)$ to describe the orientation of the binary orbit. Constructing such a flow for the gravitational wave problem would allow for the approximation of the posterior on its natural domain which stems from the geometry of the problem, rather than working in a Euclidean projection. This approach would naturally impose the expected symmetries within the parameter space and serve as an alternative to introducing the problem geometry via the priors.

Appendix A provides an introduction to a way of building normalising flows on Riemannian manifolds using results from optimal transport theory (Riemannian Convex Potential Map flows), together with an initial exploratory results of applying these to to a sine-Gaussian sky map posterior estimation.

Appendix B introduces the concept of Continuous Normalising Flows and its extension to the Riemannian manifold case, as another possible flow framework.

Part IV.
Variational Inference

4.1. GOAL AND RELATED WORK

The goal of this project was to investigate the applicability of the use of variational inference with normalising flows (VI+NF) as a machine learning (ML) alternative to the traditional stochastic sampling methods used for Bayesian inference of gravitational wave (GW) signal parameters.

The idea is very simple — we use a normalising flow (Section 3.1.1) as an expressive parametric variational family of probability distributions, which we optimise to approximate the full posterior distribution on the GW signal parameters. The optimisation of the normalising flow parameters is performed by minimising the reverse KL divergence between the variational distribution and the GW posterior target (variational inference, Section 3.1.3), with the KL divergence computation relying on GW likelihood evaluations. Once the flow is trained, we can obtain samples from the target at the cost of evaluating the flow model, evaluate an approximation of the posterior at any point in the domain without the need for likelihood evaluation, as well as obtain a lower bound on the Bayesian evidence of the signal model.

If successful, this approach would result in a purely machine learning based parameter estimation algorithm for GW inference, entirely bypassing the need for stochastic methods, much like the SBI (Section 3.1.4) algorithm Dingo [84]. Unlike Dingo, the VI+NF approach is not amortised (pre-trained), and as such would resemble the more traditional sampling algorithms where learning the posterior distribution begins after a detection is made and relies on likelihood evaluations. Dingo, due to its amortised nature, has great potential for rapid real-time parameter estimation on standard GW signals, yet by its very design is unlikely to extrapolate well to waveforms of morphologies deviating from the training set — this is contrary to the VI+NF approach, which, since the flow is not pre-trained, would allow flexibility over waveform model choice, and even network architecture design, depending on the morphology and complexity of the analysed signal. The one-time training cost of the VI+NF model would be substantially lower than the training of an SBI model, however in the VI case the cost would be paid per parameter estimation run.

So far there is only one published successful attempt of using VI+NF for Bayesian Inference within the context of gravitational wave data analysis, however not for signal parameter estimation, but for population modelling [81]. This encouraging result does not however guarantee the success of using VI+NF as a replacement of stochastic methods for signal parameter estimation — distributions on black hole population

parameters lack the complexity of the signal parameter posteriors, such as multi-modality, non-trivial parameter space geometry, and complex correlation structures, so the problem we are trying to solve is substantially more difficult.

The obvious drawbacks of the use of variational inference, instead of an SBI model like DINGO, is that the GW detection has to entirely precede model training, and that training requires likelihood evaluations, both of which are inherent to the VI parameter estimation method. At the same time reliance on likelihood evaluations is not entirely bypassed by DINGO, which in its current version requires likelihood-dependent post-training importance weighting of samples generated by the model, performed in order to reweigh the samples according to the true posterior. An extra importance sampling step might prove necessary for the VI+NF algorithm too, as a method of validating the posterior samples, as has been implemented in the VI application to GW population modelling [81], adding an extra layer of computation. Importance sampling will correct for the discrepancies between the ML model and the true posterior on the fraction of prior space contained in the support of the model — reverse KL trained flows are not guaranteed to be mode-covering, therefore the support of the target may not be fully contained in the support of the model posterior, potentially leading to scenarios where modes of the posterior are missed.

4.2. VI+NF IN THE CONTEXT OF GRAVITATIONAL WAVE PARAMETER ESTIMATION

To briefly recap the method, in the context of the gravitational wave problem — we wish to obtain samples $\{\theta_{GW}\}_{i=1}^{N}$ drawn from the posterior distribution $p(\theta_{GW}|d)$,

$$p(\theta_{GW}|d) = \frac{p(d|\theta_{GW})p(\theta_{GW})}{p(d)}, \tag{4.2.1}$$

where θ_{GW} are gravitational wave signal model parameters and d is the interferometer data segment containing the signal. We can evaluate $p(\theta_{GW}|d)$ up to the normalising constant $p(d)$ (Bayesian evidence).

To this end we introduce a variational distribution $q(\theta_{GW};\phi)$ (normalising flow, Section 3.1.1) and we optimise the parameters ϕ to minimise the reverse KL divergence (Section 3.2) between the target distribution $p(\theta_{GW}|d)$ and $q(\theta_{GW};\phi)$, thereby obtaining $\hat{q}$ that best approximates the target posterior (Section 3.1.3). This is conveniently equivalent to maximising the evidence lower bound (Section 3.1.3.1),

$$\hat{q} = \operatorname{argmax}\left(\mathrm{ELBO}(q)\right) = \mathop{\mathbb{E}}_{q(\theta_{GW};\phi)}\left[\ln \frac{p(d|\theta_{GW})p(\theta_{GW})}{q(\theta_{GW};\phi)}\right] \leq \ln p(\theta_{GW}), \tag{4.2.2}$$

so that the optimisation does not require an a priori knowledge of the evidence, and relies on evaluating the product $p(d|\theta_{GW})p(\theta_{GW})$ of the likelihood and the prior, over samples from the variational distribution $q(\theta_{GW};\phi)$.

By construction of the normalising flow, the variational distribution is defined as

$$q(\theta_{GW};\phi) \coloneqq q(f(z;\phi)) = p_Z(z)\left|\frac{\partial f(z;\phi)}{\partial z}\right|^{-1}, \quad f(\,.\,;\phi)\colon Z \to \Theta_{GW}, \tag{4.2.3}$$

where Z is the latent space of the flow and f is a bijective transform from Z to the gravitational wave parameter space Θ_{GW}, parametrised by a neural network with network parameters ϕ, and p_Z (latent distribution) is an easy to sample distribution (Gaussian or uniform) on Z.

Once the optimisation of ϕ is complete, approximate samples from the gravitational wave parameter posterior $p(\theta_{GW}|d)$ are given by samples from the variational normalising flow distribution $q(\theta_{GW};\phi)$, obtained via

$$\theta_{GW} \sim q(\theta_{GW};\phi) \quad \theta_{GW} = f(z;\phi), \quad z \sim p_Z, \tag{4.2.4}$$

therefore achieving our initial goal. These samples are obtained at the cost of sampling from the latent distribution p_Z and evaluating the transform $f(.;\phi)$, preceded by the one-time cost of optimising ϕ. With this approach, if successful, we will have fully bypassed the need for traditional stochastic methods to obtain posterior samples. Once the samples are drawn, these can be used to visualise the marginal distributions of the signal posterior using sample histograms or 'corner plots' of 2-dimensional sample projections, as well as for further scientific analyses, such as population modelling or cosmological parameter estimation.

On top of solving the sampling problem, once the flow q is trained, it provides an efficient approximation to the posterior $p(\theta_{GW}|d)$, which can be evaluated at any point in the domain Θ_{GW} as

$$\theta_{GW} \in \Theta_{GW}, \quad p(\theta_{GW}|d) \approx p_Z(z)\left|\frac{\partial f(z;\phi)}{\partial z}\right|^{-1} = q(\theta_{GW};\phi), \tag{4.2.5}$$

without needing to evaluate the computationally costly waveform model necessary for likelihood computation.

Lastly, the maximised ELBO training objective provides us with an estimate of a lower bound on the logarithm of the Bayesian evidence $\ln p(d)$,

$$\ln p(d) \geq \mathop{\mathbb{E}}_{q(\theta_{GW};\phi)}\left[\ln \frac{p(d|\theta_{GW})p(\theta_{GW})}{q(\theta_{GW};\phi)}\right] \tag{4.2.6}$$

$$\approx \frac{1}{N}\sum_{i=1}^{N} \ln \frac{p(d|\theta_{GW}^{(i)})p(\theta_{GW}^{(i)})}{q(\theta_{GW}^{(i)};\phi)}, \quad \theta_{GW}^{(i)} \sim q(\theta_{GW}^{(i)};\phi), \tag{4.2.7}$$

which can be used for model comparison. From this perspective VI+NF provides advantages similar to those of nested sampling, with the products of both parameter

estimation methods being target samples as well as an evidence estimate, unlike standard MCMC, which results in target samples but no evidence estimate.

In the case when the optimisation results in an exact match between the flow model and the target posterior, $q(\theta_{GW};\phi) = p(\theta_{GW}|d)$, the optimisation target exactly recovers the logarithm of the evidence,

$$\mathop{\mathbb{E}}_{q(\theta_{GW};\phi)}\left[\ln \frac{p(d|\theta_{GW})p(\theta_{GW})}{q(\theta_{GW};\phi)}\right] = \mathop{\mathbb{E}}_{p(\theta_{GW}|d)}\left[\ln p(d)\right] = \int_{\Theta_{GW}} p(\theta_{GW}|d)\underbrace{\ln p(d)}_{\text{const.}} = \ln p(d). \tag{4.2.8}$$

4.3. A TOY EXAMPLE: A SINE-GAUSSIAN WAVEFORM LIKELIHOOD

The first test of variational inference with normalising flows applied to the GW posterior estimation problem was to approximate a parameter posterior of a simple sine-Gaussian signal. Even though far from a full CBC waveform, it is a signal model that is frequently used by LIGO for ad-hoc modelling of GW burst signals [93], making it a good low-dimensional toy model test case for the applicability of the method.

4.3.1. *Sine-Gaussian signal model*

The sine-Gaussian signal in time domain (Figure 5) can be written as

$$h(t) = h_+(t) + ih_\times(t) = A\exp\left[\frac{-(t-t_0)^2}{2\tau^2} + 2\pi i f_0(t-t_0)\right], \tag{4.3.1}$$

$$h_+(t) = A\exp\left[\frac{-(t-t_0)^2}{\tau^2}\right]\cos\left[2\pi f_0(t-t_0)\right], \tag{4.3.2}$$

$$h_\times(t) = A\exp\left[\frac{-(t-t_0)^2}{\tau^2}\right]\sin\left[2\pi f_0(t-t_0)\right], \tag{4.3.3}$$

where A is the signal amplitude, f_0 is the frequency, t_0 is the time corresponding to peak amplitude, and the parameter τ determines the pulse duration. In the frequency domain this translates to

$$h_+(f) = C\times A\sqrt{\pi}\tau\times\exp\left[-\pi^2\tau^2(f-f_0)^2\right]\exp\left[-2\pi i(f-f_0)t_0\right] \tag{4.3.4}$$

$$h_-(f) = -iC\times A\sqrt{\pi}\tau\times\exp\left[-\pi^2\tau^2(f-f_0)^2\right]\exp\left[-2\pi i(f-f_0)t_0\right], \tag{4.3.5}$$

where C is a constant dependent on the sample time spacing.

A signal of this form was injected into a simulated 2-detector L1+H1 network, for simplicity with a zero noise realisation and a constant noise PSD across the frequency bins, with the noise power set to obtain a network SNR of the order 20. The parameter space of the model was, for the purposes of the method test, constrained to 4 dimensions, where the free parameters are the signal amplitude A, and the extrinsic

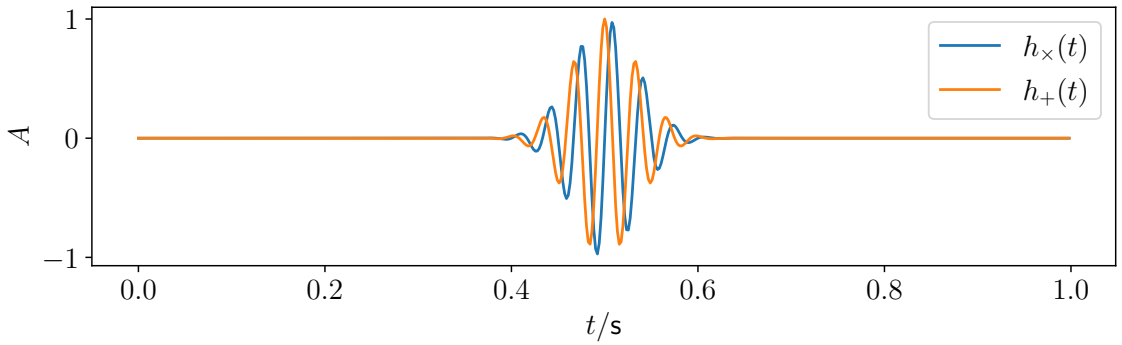


Figure 5: Sine-Gaussian signal model.

Time domain plus and cross polarisation of an example sine-Gaussian signal model with $f_0 = 30\,\text{Hz}$, $\tau^{-1} = 20\,\text{Hz}$, $t_0 = 0.5\,\text{s}$, in units of signal amplitude A.

sky location parameters which influence the antenna response patterns of the detectors — declination δ, right ascension α and signal polarisation ψ. The injection parameters were $(A, \alpha, \delta, \psi) = (0.5, 0.5, 0.5, 0.5)$. Priors were flat, with the exception of a cosine prior on δ. Injected segment duration was 1 s with sampling frequency 100 Hz, $f_0 = 20\,\text{Hz}$, $t_0 = 0.5s$, $\tau^{-1} = 10\,\text{Hz}$.

4.3.2. *Normalising Flow and training set up*

The normalising flow used was a coupling normalising flow (Section 3.3.1) with monotonic rational-quadratic spline transforms in each of the latent dimensions (C-RQS flow, Section 3.4.0.2) and a uniform latent distribution, implemented in JAX [92] (Section 3.6) and available within the DISTRAX package [94]. The flow consisted of 4 spline blocks with 6 knots per block, conditioned on a 2 layer MLP network (Section 3.5.1) with a width of 100 neurons, chosen based upon examination of the training performance, however without resorting to optimisation via a thorough hyperparameter sweep.

The optimisation aim was to maximise the evidence lower bound of the target, equivalent to minimising the reverse KL divergence between the flow distribution and the target posterior (Section 3.1.3); optimisation was performed stochastically (Section 3.5.2) with the Adam optimiser, using JAX for automatic differentiation for computing loss gradients and performance optimisation. The flow was trained for 5000 stochastic gradient descent updates of the NN parameters, where within each update the loss was calculated using 1000 flow samples. The optimiser learning rate used was 0.01, fixed throughout the training.

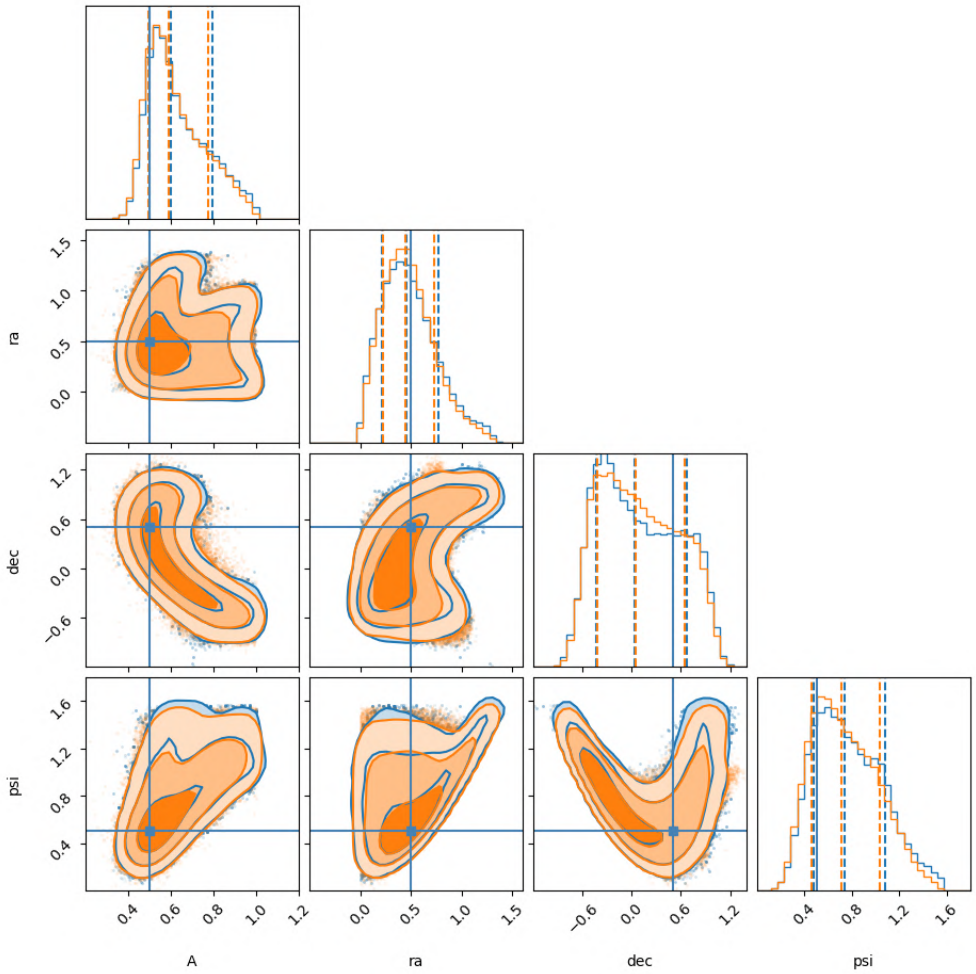


Figure 6: Variational inference applied to a sine-Gaussian posterior.

Comparison of the posterior distribution on the sine-Gaussian signal parameters inferred by variational inference with normalising flows (orange) and a standard MCMC algorithm emcee (blue).

4.3.3. *Results*

The best training results and their comparison against target MCMC samples obtained with the Emcee [95] sampler are shown in Figure 6. Even though the flow has successfully trained and achieved a good agreement with the MCMC posterior on the sine-Gaussian parameters, its robustness proved to be limited — success of the training varied largely between runs with different random initialisations and depended on fine tuning the optimiser learning rate — in the unsuccessful runs the flow locked on secondary features in the tails of the posterior. This lack of robustness is a first indication of the shortcomings of using VI with reverse KL divergence, where the

tendency for mode-seeking behaviour stems from the training objective (elaborated on in Section 3.2.3). Even for the runs where training would end successfully, the flow behaviour was to initially rapidly lock onto narrow support secondary features of the posterior, while retaining just enough support in proximity of the injection parameters, to eventually draw high likelihood samples from the flow tails and reject the secondary mode in favour of the true posterior mode.

4.3.4. *VI + NF for a full GW likelihood of a CBC signal*

Motivated by the posterior recovery variational in the sine-Gaussian waveform problem, variational inference with normalising flows was tested in the context of compact binary coalescence (CBC) gravitational wave signal parameter inference.

Training a flow via reverse KL divergence requires a differentiable target distribution (Section 3.1.3) — in the gravitational wave case the parameter posterior is a function of the likelihood which itself is a function of the GW waveform model, and hence applying VI to the GW parameter estimation relies on access to differentiable waveforms. These have recently been implemented in Jax in the Ripple package [80], which opens the avenue for testing VI+NFs on the full CBC problem.

4.3.5. *Initial results: problem with mode locking*

After successful implementation of a differentiable GW CBC likelihood based on Ripple frequency domain IMRPhenomD waveforms in Jax and, involving real detector noise PSDs, the VI algorithm was tested on a different configurations of coupling normalising flows with spline bijectors. The injected test signal was a 20-10 solar mass BBH in a two detector network (L1+H1) with a network SNR of ~ 20 and flat priors.

Initial tests showed good flow performance on simple low dimensional slices of the target (distance-inclination) yet failed to successfully learn more complex distribution projections even in low dimensions (chirp mass - mass ratio). Regardless of the flow configuration, when extended to higher dimensions, the flow suffered from mode-seeking behaviour, locking on the likelihood plateau corresponding to the 0 amplitude signal (high distance, inclination $\pi/2$), an example of which can be seen in Figure 7 — this plateau is a much broader feature of the likelihood surface than the mode corresponding to the signal injection and hence is easily picked up by the flow in the absence of high likelihood training samples.

The problem with the reverse KL training of the VI method seems to lie in the mode-seeking behaviour of the flow, which, as explained in Section 3.2.3 is intrinsic to the reverse KL training target. The flow tends to lock onto a secondary mode/feature which then leads to a situation where the training effectively halts due to lack of high likelihood samples informing the loss gradient — since these high likelihood samples

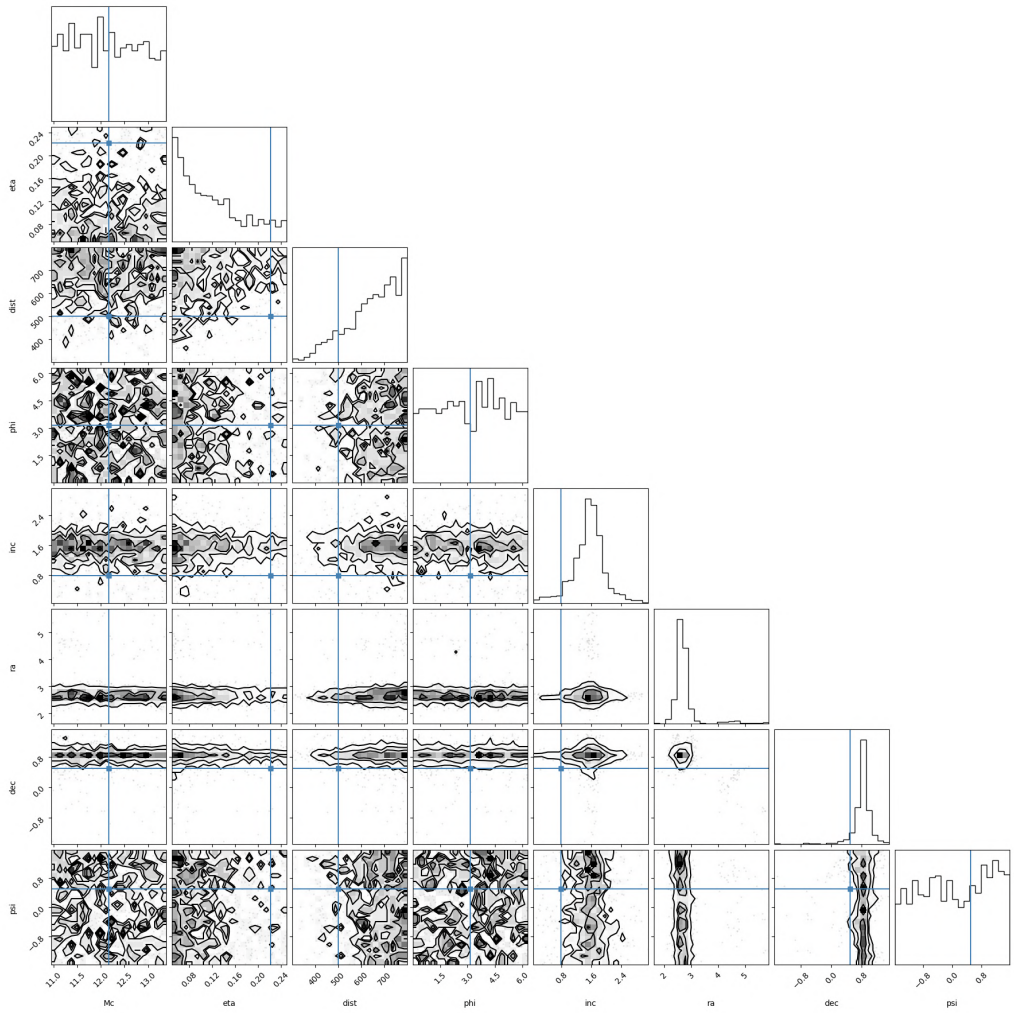


Figure 7: Failed variational inference of a CBC posterior in 8 dimensions.

Typical result of a failed reverse KL divergence VI+NFs training on a CBC posterior on a constrained domain. The flow locks on the 0 amplitude signal plateau corresponding to high distance and inclination $\iota = \pi/2$.

are in the tails of the flow distribution, and the training relies solely on the flow samples, these are rarely drawn.

Once the flow has locked onto an isolated distribution feature, covering another mode without losing the original feature would require the flow to assign non-zero probability to points in-between the two modes. If in the in-between modes region the target density is close to 0, covering both features can increase the reverse KL

loss, as, recalling from Section 3.2.3, we have

$$p(x) \to 0 \implies \ln \frac{q(x)}{p(x)} \to \infty \quad q(x) > 0, \quad KL(q||p) = \mathop{\mathbb{E}}_{q}\left[\ln \frac{q}{p}\right]. \tag{4.3.6}$$

In the case of unimodal targets this effect can be mitigated by fine-tuning the learning rate so that in the initial stages of training it would allow flow state updates corresponding to inter-modal jumps, allowing the flow to drop low-likelihood features without increasing the loss. The problem becomes unavoidable however for multimodal targets. As long as the target distribution is unimodal, or the distribution modes are connected, and provided the flow is expressive enough, the true loss minimum will correspond to the flow q equal to the target p, and therefore increasing the access to high likelihood samples should still lead to successful training. In the case of disconnected posterior modes however the lowest loss configuration can correspond to only one mode being covered by the flow. This could potentially be mitigated by introducing multimodal latent distributions, or possibly using a mixture of flows to model the target. This would however introduce the problem of needing to a priori know how many modes are present in the target, or training a more flexible flow model, with variable number of components.

4.4. PREVENTING MODE-COLLAPSE IN VARIATIONAL INFERENCE

The main issue preventing successful application of variational inference to the CBC posterior inference has been identified as mode collapse, and specifically, the flow locking to the 0-amplitude signal mode. A simple problem-specific fix could be reparametrisation of the posterior, so that this secondary mode is given a smaller support in the parameter space.

After consultation with Dr Paul Henderson[1] different problem-independent improvements were tested — these included:

- Mixing samples from the uniform distribution to stop the modes from collapsing at early stages of the training,
- Temperature annealing of the posterior,
- Learning rate scheduling,
- Use of wider MLPs (at least 100 neurons wide) and longer training runs of $\mathcal{O}(10^5)$ epochs.

All these were implemented, with details provided in the following sections.

[1] School of Computer Science, University of Glasgow

4.4.1. *Reparametrisation*

One of the problems we experienced when applying variational inference to the gravitational wave problem is the flow locking to the mode associated with the solution corresponding to no signal being present in the data, supported in the region of the parameter space resulting in a signal of low amplitude — that is the high distance, low mass, edge-on inclination ($i = \pi/2$). The gravitational wave posterior can be thought of as a 'slab and spike' distribution, where the slab corresponds to the wide plateau of the 'no signal' solution, and the spike to sharply peaked narrow mode corresponding to the real signal. Provided there is a signal in the data, these two modes will always be present, and hence without access to high likelihood samples at the beginning of the training, the flow will tend to lock onto the wider feature of the plateau, as it corresponds to a larger prior volume and hence is more easily randomly sampled.

The possible solution to this problem would be reparametrising the problem such that the 0-amplitude mode has less support in the sampling space. For example, instead of representing the target as a function of luminosity distance D_L, one can represent the target in terms of $\log D_L$. Since we are reparametrising along the distance dimension D only, we can represent GW parameter space as $\Theta = D \times \Theta/D$, where D is the luminosity distance dimension, and think of the reparametrisation as a product $\Phi \coloneqq \phi \times \mathrm{Id}$ of a one-dimensional distance transformation $\phi\colon D \to D$, and an identity transformation Id on the remaining dimensions, that is

$$\Phi\colon \Theta = D \times \Theta/D \to \Theta, \quad \Phi = \phi \times \mathrm{Id}, \tag{4.4.1}$$

$$\phi\colon D \to D, \quad \phi(D_L) = \log D_L. \tag{4.4.2}$$

Due to the reparametrisation, the target distribution p needs to be updated with the Jacobian determinant of the transform $|J_\Phi| = |J_\phi J_{\mathrm{Id}}| = \frac{d\phi}{dD_L} = 1/D_L$,

$$p(\phi(D_L), \mathrm{Id}(\boldsymbol{\theta})) = p(D_L, \boldsymbol{\theta}) \left|J_\Phi\right|^{-1}, \tag{4.4.3}$$

$$p(\log(D_L), \boldsymbol{\theta}') = p(D_L, \boldsymbol{\theta}') D_L, \tag{4.4.4}$$

$$\log p(\log D_L) = \log p(D_L) + \log D_L. \tag{4.4.5}$$

For example given a prior bound of [300,800] Mpc, the upper half of the prior [550,800], corresponds to 0.5 of the prior volume in the uniform latent space of the flow. The same interval in the log-space, [log(550), log(800)] corresponds to only 0.38 of the prior volume, a decrease of 25% of the prior volume. In the case of variational inference, loss is evaluated on samples from the flow, and moreover our flow is initialised as a uniform distribution in the data space (via an identity transform on the uniform latent space), this volume decrease corresponds to reducing the fraction of the training samples from the 0-amplitude plateau by 25%.

4.4.2. *Uniform mixing*

Another issue that variational inference suffers from is the limited access to high likelihood samples during training. If the mode of the target is not initially found by the flow, in subsequent training steps the flow is supplied only with low likelihood samples from the features already found, which can often be secondary features away from the mode, reinforcing the initial poor approximation. One would expect that the additional information coming from the gradient of the target distribution would help the flow exit low likelihood local loss minima, but in practice we found that the gravitational wave problem is too complex for the flow to find the main mode of the target with gradient information alone, and would benefit from a supply of high likelihood samples.

Since the target samples are unavailable during training, we cannot supply the flow with high likelihood samples directly, but we can instead ensure that at every training step the flow has a good probability of drawing a high likelihood sample. This is done by setting the variational distribution to be a mixture of the trained flow and a uniform distribution over the prior space. Even if a mode of the target is not yet seen by the flow, there is a chance that a sample from that feature will be drawn from the uniform component and introduced to training.

The mixture model is defined as

$$m(\theta) \coloneqq w\,\mathcal{U}_\Theta + (1-w)q(\theta), \quad \theta \in \Theta \tag{4.4.6}$$

where $\mathcal{U}_\Theta$ is the multivariate uniform distribution on the data space Θ, $q(\theta)$ is the normalising flow, and $w \in [0, 1)$ is a mixture weight chosen for the uniform distribution component.

To obtain a sample from the mixture one needs to first perform a Bernoulli trial to decide which mixture component to draw a sample from. We draw a sample from a Bernoulli distribution with success probability $p = w$,

$$f(k; p) = p^k(1-p)^{1-k}, \quad k \in \{0, 1\}, \tag{4.4.7}$$

and depending on the outcome $k \in \{0, 1\}$, we proceed to draw a sample from either the flow component, $(k = 0)$, or the uniform mixture component, $(k = 1)$. The variational model probability density associated with this sample is then evaluated according to Equation 4.4.6.

Since the uniform mixture component is constant over the domain Θ, $\mathcal{U}_\Theta(\theta) = C$, $\forall\,\theta \in \Theta$, the optimal normalising flow distribution q^* will be proportional to the target p minus a constant, so the flow distribution q^* will continue to learn to represent

the features of p

$$q^* = \operatorname*{argmin}_q D_{KL}(m \,||\, p) = \operatorname*{argmin}_q \mathop{\mathbb{E}}_{q(x)} \left[\ln \frac{w\,\mathcal{U}_\Theta + (1-w)q(\theta)}{p(\theta)} \right] \tag{4.4.8}$$

$$\iff wC + (1-w)q^*(\theta) = p(\theta) \quad \forall\, \theta \in \Theta \tag{4.4.9}$$

$$\implies q^*(\theta) \propto p(\theta) - \text{const.} \quad \forall\, \theta \in \Theta \tag{4.4.10}$$

During training, the mixture weight w is decreased over sampling time $t \in [0, 1]$, in a linear manner, $w(t) = (1-t)w_0$, so that at the end of training $w = w(1) = 0$.

4.4.3. *Temperature annealing*

A commonly used method of facilitating learning complicated target distributions is temperature annealing, successfully used for the GW CBC problem by MCMC samplers utilising parallel tempering — such as the LALINFERENCE [20] MCMC sampler used in the LVC observing runs O1-O3 [2; 3; 4]. The idea is to decrease the difficulty of the problem by reducing the target distribution towards the prior. To this end, we introduce a temperature parameter T, and define the target posterior at temperature $T \in [1, \infty]$ as

$$Z\, p(\theta|d; T) \coloneqq p(d|\theta)^{1/T} p(\theta). \tag{4.4.11}$$

When the temperature is high, $T \to \infty$, $p(d|\theta)^{1/T} \to 1$, so that the target is equal to the prior, and for $T = 1$ we return to the original target.

Introducing a decreasing annealing schedule $T\colon [0, 1] \to [1, \infty]$, with $T(0) = T_0$, $T(1) = 1$, $T(t) < T(t'), \forall\, t > t'$, we can start the training with a simpler problem ($t = 0$) and gradually introduce more information about the target until the end of training at $t = 1$.

4.4.3.1. *Temperature annealing schedules and distribution information content*

The simplest annealing schedule would be a linear schedule with $T(t) = T_0$ and $T(t) = 1$, given by

$$T(t) = t + (1-t)T_0 \tag{4.4.12}$$

Let's inspect how the training loss is altered by the training schedule. Assume we are not updating the network parameters, so the flow distribution q is constant with time. For simplicity the priors are flat, so that the annealed target is $p^{1/T(t)}$. The

training loss is the reverse KL divergence between the target and the flow,

$$L(t) = D_{KL}(q(x) \,||\, p^{1/T(t)}(x)) \tag{4.4.13}$$
$$= \mathop{\mathbb{E}}_{q}\left[\log \frac{q(x)}{p^{1/T(t)}(x)}\right] \tag{4.4.14}$$
$$= \mathop{\mathbb{E}}_{q}\left[\log q(x) - \frac{1}{T(t)} \log p(x)\right] \tag{4.4.15}$$
$$= \mathop{\mathbb{E}}_{q}\left[\log q(x)\right] + \frac{1}{T(t)} \mathop{\mathbb{E}}_{q}\left[-\log p(x)\right] \tag{4.4.16}$$
$$= A + \frac{B}{T(t)}, \tag{4.4.17}$$

where A and B are constant with time. In particular, in the case where the target p is a Gaussian likelihood, $\log p \leq 0$ over the whole of the domain, and hence the constant B, equal to the cross-entropy between q and p, is positive. We can see that decreasing the temperature T will increase the KL divergence between the target and the flow, and in particular if we choose a linear annealing schedule, the loss will vary with training time t as

$$L(t) = A + \frac{B}{t + (1-t)T_0}. \tag{4.4.18}$$

As shown in Figure 8, the linear annealing schedule introduces information slowly when the distribution is simple ($T \gg 1, t \ll 1$) and rapidly when we approach the true target ($T = 1, t = 1$). This points to the fact that there will be an interplay between the the annealing schedule and the learning rate used for performing gradient descent on the loss function in the network parameter space – ideally the learning rate and the annealing schedule should be chosen so that the learning rate can keep up with the growing loss and take advantage of the simplicity of the initial high temperature target (higher learning rate to begin with).

In order to avoid increasing the amount of target information content introduced in subsequent training steps, a temperature annealing schedule which increases the information difference linearly with time is preferable. This will be achieved by a temperature schedule chosen so that $1/T(t)$ increases linearly with time t, that is

$$\frac{1}{T(t)} = t + (1-t)\frac{1}{T_0} \implies L(t) = A + (t + (1-t)\frac{1}{T_0})B. \tag{4.4.19}$$

Figure 8 shows how the reverse KL divergence between a constant uniform flow and an example gravitational wave posterior changes with time t due to introducing the two different temperature annealing schedules. One could further adjust the annealing schedule $T(t)$ to obtain a custom information supply schedule.

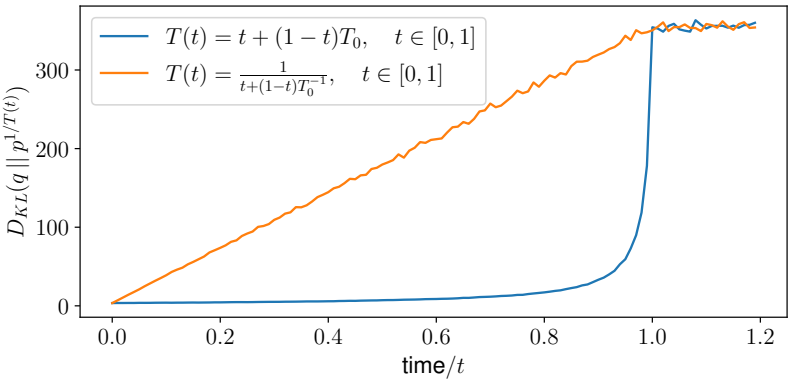


Figure 8: Relationship between annealing schedule and reverse KL divergence.

Variation in reverse KL divergence between a uniform variational distribution q and an annealed gravitational wave posterior $p^{1/T(t)}$ due to the introduction of different temperature annealing schedules $T(t)$. At $t > 1$ the annealing is switched off, that is $T = 1$ and p is the true target distribution – the only variation in the KL approximation is due to stochasticity.

4.4.4. *Learning Rate Annealing*

Improved training can be achieved by introducing a learning rate annealing schedule — the stochastic gradient descent step is decreased as the training progresses. This allows for an initially higher learning rate which prevents the algorithm from falling into local minima of the loss surface, and a small final learning rate that allows for precise exploration of the global minimum.

A cosine learning rate decay schedule [96] was chosen, where the learning rate $\alpha(t)$ is given by

$$\alpha(t) = \begin{cases} \epsilon\alpha_0 + (1-\epsilon)\alpha_0 \left[\frac{1}{2}\left(1 + \cos(\frac{\pi t}{T})\right]\right., & t < T \\ \epsilon\alpha_0, \quad t \geq T, \end{cases} \tag{4.4.20}$$

where α_0 is the initial learning rate, $\epsilon \in (0,1)$ is the decay coefficient . The final learning rate is attained at time T and is given by $\epsilon\alpha_0$.

4.4.5. *Denser sampling*

Another straightforward amendment that could circumvent the problem of lack of high likelihood training samples available to the flow in the early stages of training is increasing the number of training samples. In the limit as $N \to \infty$ the flow should be able to 'see' the whole target at every iteration and know the true gradient of the loss function, and, provided the flow is expressive enough and the learning rate schedule is chosen appropriately, the training should succeed.

In practice increasing the number of samples will reduce the stochastic variance in the loss and loss gradient estimates, and provide better initial access to high-likelihood samples, at the cost of increased computation time. In terms of the total training time, there will be an interplay between the per iteration computation time which increases with the number of sampled points, and the total number of training epochs needed for convergence.

4.4.6. *Preventing Mode collapse in Variational Inference: Results*

4.4.6.1. *5-dimensional subspace*

All of the described mode collapse prevention methods were implemented and initially tested on a 5-dimensional subspace of a zero-noise, SNR 20, 2-detector BBH injection target posterior distribution (with flat priors), where the estimated parameters were the extrinsic parameters $(D_L, \iota, \psi, \alpha, \delta)$. The target is bimodal in ι, so it provides a good test-case for the mode collapse.

Without resorting to extensive hyperparameter sweeps a training run with simple linear temperature annealing schedule, cosine learning rate annealing, and linearly annealed uniform mixing fraction led to successful training. Good results were achieved using both a 4-layer coupling flow and a masked autoregressive flow (MAF) with spline bijectors. The training hyperparameters used for both flows are presented in Table 1. Both trained flows found and retained both modes corresponding to the injected signal.

Beginning training at a high temperature allows the flow to learn the broad shape of the posterior first — at this stage the distribution is unimodal and the most prominent feature is the widened slab corresponding to the 0-amplitude signal. As the temperature is annealed the spike corresponding to the injection gradually emerges from the slab, allowing the flow to slowly increase the probability density in the injection region, as well as to keep both of the emerging modes. Flow samples at different stages of the annealing for both flows are presented in Figures 9 & 10.

Training was stopped arbitrarily at the end of the annealing process, when the target temperature reached $T = 1$ and uniform mixing fraction reached 0. Inspection of the in-training KL divergence (Figure 11) suggests that training should continue after temperature annealing and uniform mixing is phased out, allowing for a lower loss configuration to be reached.

A lower loss configuration within the fixed training time of 10000 updates was reached by the coupling flow. The evolution of the KL divergence between the flow and the temperature $T = 1$ target for both flow architectures is provided in Figure 11. The MAF flow exhibited higher KL variance during training, suggesting the coupling architecture provides more training stability.

Table 1: Flow and training hyperparameters (coupling & MAF)

Hyperparameter	Value
latent distribution	$\mathcal{U}_{[0,1]^5}$
# spline bijector bins	10
# coupling flow transforms	4
MLP layer width	128
# MLP hidden layers	4
training updates	10^4
flow samples per update	10^4
w_0 (initial uniform mixing fraction)	0.25
uniform mixing annealing schedule	linear
α_0 (initial learning rate)	5.0×10^{-4}
ϵ (learning rate decay coefficient)	10^{-2}
learning rate annealing schedule	cosine
T_0 (initial temperature)	100
temperature annealing schedule	linear

4.4.6.2. *Hyperparameter sweeps: 6-dimensional subspace*

Extending the 5-dimensional domain by the mass ratio η resulted in mode collapse to one of the ι modes when using the same flow and annealing configurations that successfully trained on the 5-dimensional subspace. This lead to the decision to investigate the impact of the annealing hyperparameters in more detail, to try to establish which of the implemented amendments to the VI training has the most effect on mode retention. Testing of the variational inference posterior estimation with different annealing schedules has been performed via two WEIGHTS & BIASES [97] hyperparameter sweeps.

The target distribution is a 6-dimensional subspace of a 0-noise BBH injection target posterior distribution in a 2-detector network, where the estimated parameters are the extrinsic parameters $(D_L, \iota, \psi, \alpha, \delta)$ and the mass ratio η. The remaining parameters were kept fixed at the injection parameters. Flat priors were used for simplicity.

The dimensionality of the problem was chosen based on the prior training outcomes — the flow tends to retain both modes on the 5-dimensional posterior with extrinsic parameters only, whereas the 7-dimensional posterior with both the mass ratio η and chirp mass $\mathcal{M}$ has so far led to only unsuccessful training with only the 0-amplitude mode found by the flow. This 6-dimensional problem is the transitionary case which has led to some successful training runs provided fine-tuning of the hyperparameters, however most runs suffer from mode collapse.

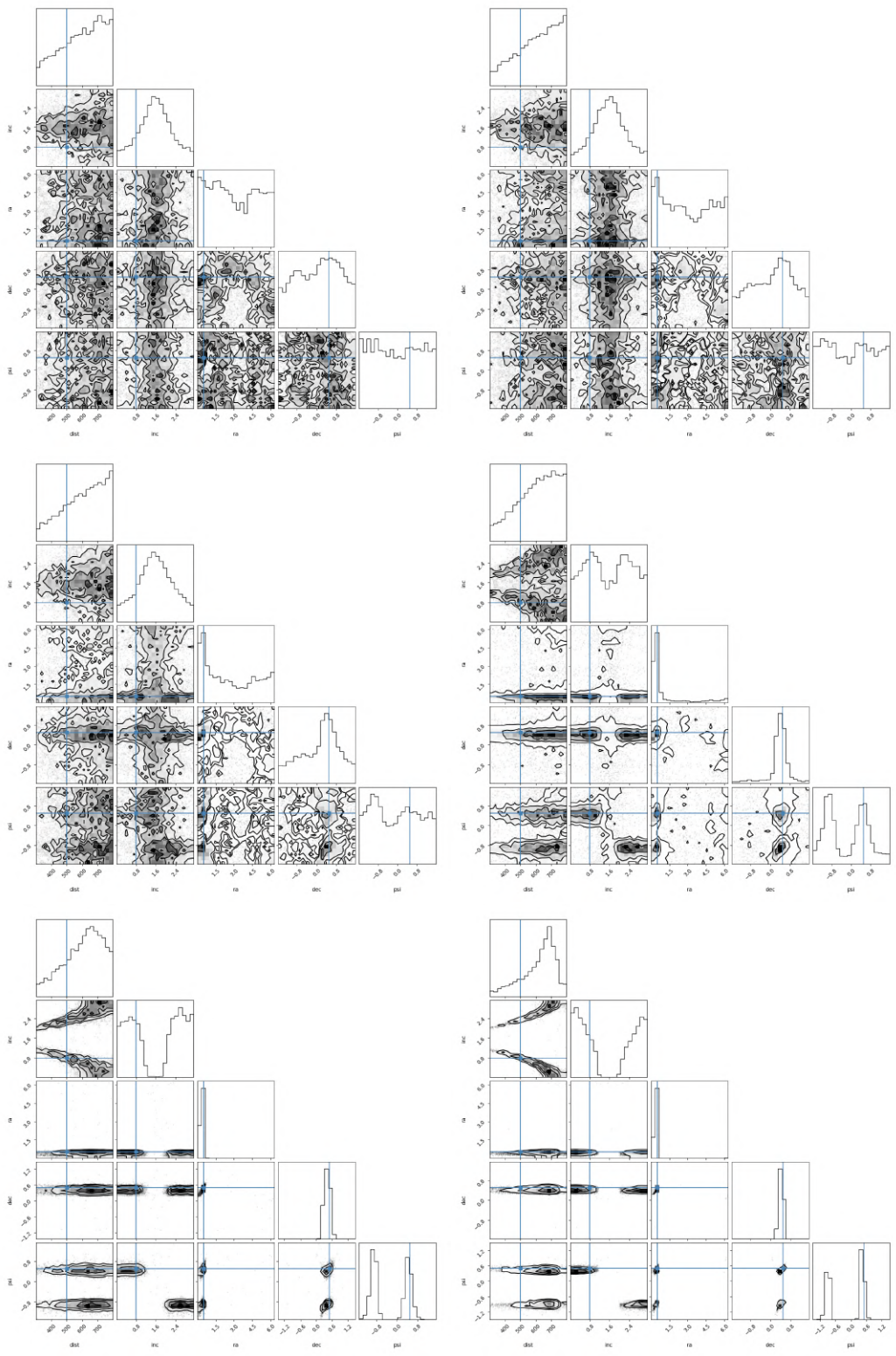


Figure 9: Variational inference with a coupling flow architecture trained on a temperature annealed BBH posterior in 5 dimensions.

Coupling flow trained on the temperature annealed BBH posterior target in 5 dimensions. Corner plots display samples from the trained flow at training epochs corresponding to annealing times $t = 1/10$, $3/10$, $5/10$, $7/10$, $9/10$ and 1 (end of training), $T(t) = t+(1-t)100$. Temperature annealing allows for recovery of both of the signal modes, however bias towards high distance is still present.

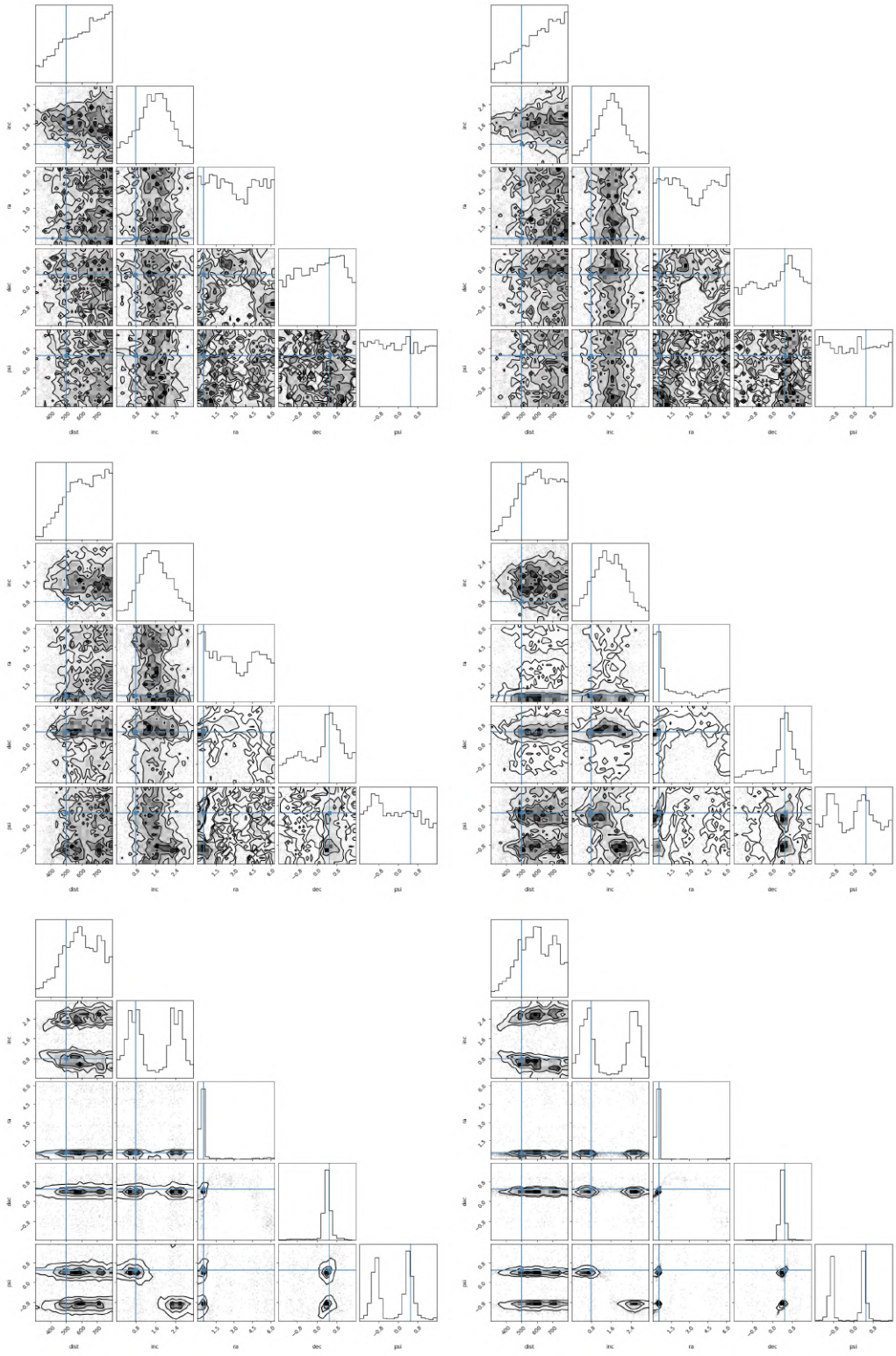


Figure 10: Variational inference with MAF flow architecture trained on a temperature annealed BBH posterior in 5 dimensions..

MAF flow trained on the temperature annealed BBH posterior target in 5 dimensions. Corner plots display samples from the trained flow at training at training epochs corresponding to annealing times $t = 1/10$, $3/10$, $5/10$, $7/10$, $9/10$ and 1 (end of training), $T(t) = t + (1 - t)100$. Both signal modes are recovered when $T = 1$.

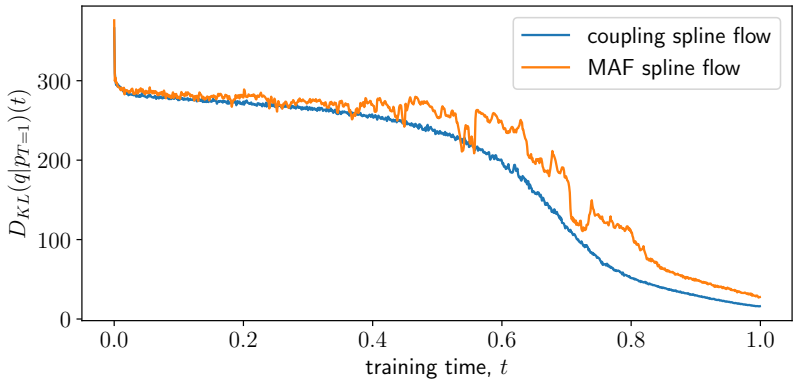


Figure 11: Training loss: coupling vs MAF flow architecture.

Reverse KL divergence between the flow q and the temperature $T = 1$ target p for the coupling and MAF architectures. MAF flow shows higher variance and higher loss at the end of annealing, suggesting the coupling architecture is more stable and hence preferable. The training pattern is otherwise similar.

The injected signal posterior is bimodal with two equal amplitude modes, with the hyperplane $\iota = \pi/2$ separating the parameter space into the two modes. Retention of both modes by the trained flow q was quantified by defining the following performance metric

$$\text{bimodality coefficient} = |\{\theta|\iota < \pi/2\}| - |\{\theta|\iota \geq \pi/2\}|, \quad \theta = (D_L, \iota, \psi, \alpha, \delta, \eta) \sim q, \tag{4.4.21}$$

i.e. the difference in the cardinalities of the sample sets (numbers of samples) belonging to the two distinct modes. A bimodality coefficient equal to 0 indicates perfect retention of both modes (desirable), whereas cardinality coefficient of N or $-N$, where N is the sample size, indicates collapse modes $i \geq \pi/2$ or $i < \pi/2$ respectively.

Two separate sweeps were performed, both of which were testing how the performance metrics of bimodality coefficient and training loss correlate with the following hyperparameters

1. Temperature annealing duration N_T,
2. Uniform mixing fraction annealing duration N_U,
3. Learning rate cosine annealing duration N_L
4. Initial learning rate α_0.

In the training runs included in Sweep 1 all of the annealing schedules start at the beginning of training and evolve simultaneously, whereas in Sweep 2 the annealing is

performed sequentially – first annealing the temperature, then the uniform mixture component fraction and lastly the learning rate.

The normalising flow architecture used for both sweeps was fixed — the flow used was a rational quadratic spline coupling flow with 4 transforms, 10 spline bins per transform, parametrised by an MLP conditioner network 128 neurons wide and 4 layers deep. The full list of hyperparameters is as shown in Table 1, unless explicitly stated otherwise. These parameters have provided optimal learning in the 5-dimensional problem case; the coupling flow was chosen over MAF as it showed superior training stability.

4.4.6.3. *Sweep 1*

The flow was trained by minimising the reverse KL divergence between the mixture distribution of the flow with the uniform and the target distribution. The training was performed for a fixed length of 10 000 training steps with 10 000 flow training samples per training step, each annealing schedule starts from iteration 1 onwards, for a random fraction of the fixed 10^4 training steps. The initial learning rate was randomly chosen as one of $\{5 \times 10^{-4}, 5 \times 10^{-5}\}$. A linear annealing schedule was used both for the temperature annealing and uniform mixing fraction annealing, with initial temperature fixed at $T_0 = 100$ and initial uniform mixing fraction $u_0 = 0.25$. The learning rate cosine annealing schedule decreases the learning rate by a factor of 10^{-2} over the number of annealing steps N_L.

The results of the sweep are presented in panels 1a-b of Figure 12 and Table 2. We can see in Figure 1a that the training loss is minimised when both modes are retained, so we can focus on the correlation between the tested parameters with the loss. The hyperparameter showing the greatest correlation with the training loss is the number of temperature annealing steps N_T – the correlation is negative, suggesting that prolonging temperature annealing, and therefore slower introduction of target information, decreases the final training loss and helps to retain both posterior modes. The number of uniform mixing and learning rate annealing steps N_U and N_L show a positive correlation with the training loss, suggesting the training benefits from a quick reduction in the uniform mixture component as well as quick decrease in the learning rate, possibly suggesting that the initial learning rate is too high. In Figure 1b we can see that all of the training runs which did not suffer from mode collapse have the initial learning rate α_0 equal to 10^{-5}, the lower of the two possible values. All of the runs retaining the modes have $N_T > 2500$, $N_U < 6000, N_L < 5000$. However – there is no clear region of the parameter space which would always lead to successful mode retention.

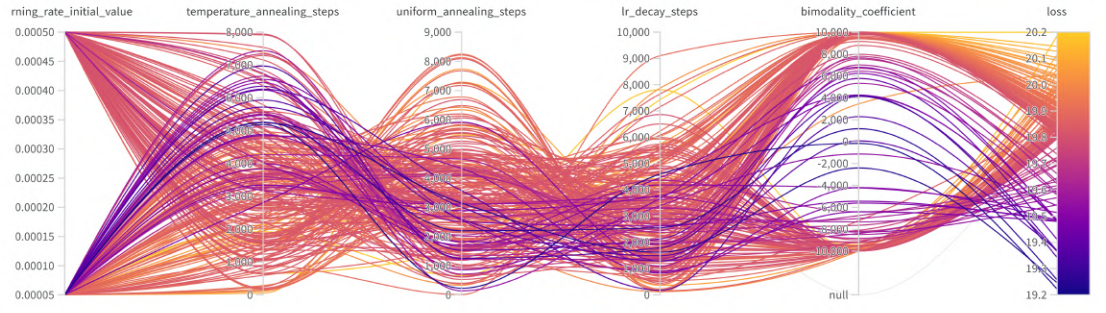
Table 2: Sweep 1: Results

Hyperparameter	Importance	Correlation with training loss
N_T	0.408	-0.436
N_U	0.152	0.261
N_L	0.157	0.194
α_0	0.149	0.061

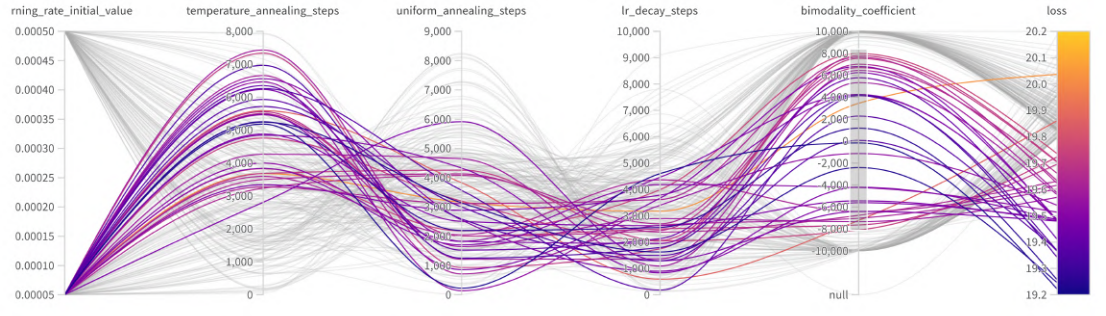
4.4.6.4. *Sweep 2*

In the second hyperparameter sweep the annealing steps are subsequent, starting with a linear temperature annealing schedule from $T_0 = 100$ to $T_1 = 1$ for the first N_T steps at constant uniform mixing fraction of $u_0 = 0.25$, and a constant initial learning rate uniformly sampled in the range $\{5 \times 10^{-5}, 5 \times 10^{-6}\}$; then proceeding with annealing the uniform mixture component to $u_1 = 0$ for N_U steps at a constant learning rate α_0, and finally lastly the learning rate at a cosine schedule for the last N_L steps. Each of the annealing durations is sampled uniformly in the range $[0, 10^4]$, with the total number of training iterations varying accordingly. The lower learning rate was suggested by the results of sweep 1.

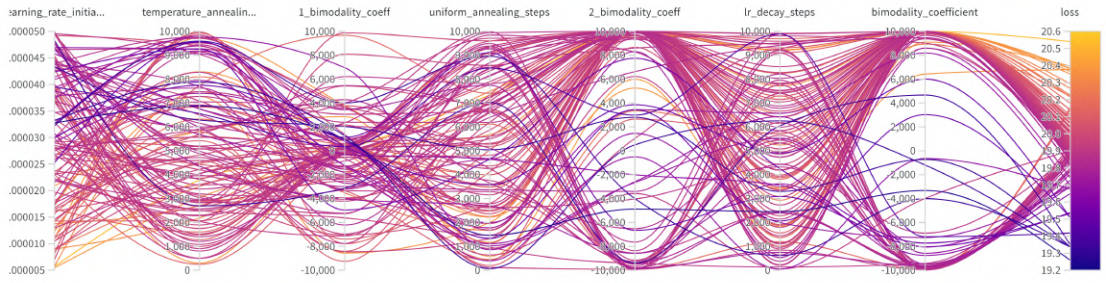
The results of the sweep are presented in panels 2a-b of Figure 12 and Table 3. This time the most significant correlation is exhibited by the initial learning rate α_0, where lowering the learning rate towards 5×10^{-6} increases the final training loss, indicating that the $\sim 10^{-5}$ is the preferred initial learning rate. It is beneficial to prolong the temperature annealing, further confirming the results of sweep 1. This time the length of uniform mixture annealing correlates slightly negatively with loss. This can be interpreted as a result of the sequential nature of this training, where increasing the number of annealing steps increases the overall number of training steps, which decreases the loss. The number of learning rate annealing steps has a slight positive correlation with the loss as in sweep 1, suggesting that rapid annealing benefits the overall training loss and mode retention. We can see in panel 2b of Figure 12 that mode retention is high prior to the uniform mixture annealing step. After the uniform mixture component is annealed the majority of runs fail to retain both of the posterior mode, and again, there is no defined region of the parameter space which consistently leads to successful mode retention.



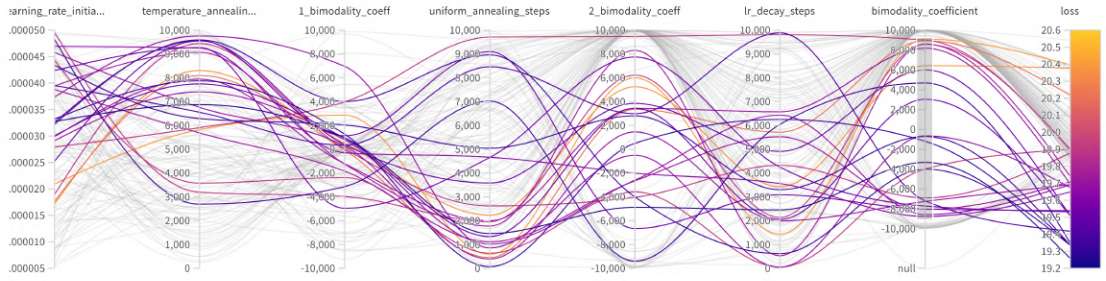
(1a)



(1b)



(2a)



(2b)

Figure 12: Weights & Biases hyperparameter sweep results..

Panels 1 & 2 correspond to sweeps 1 and 2 respectively, panels 'a' show the full sweep results, whereas panels 'b' include only the training runs in which both posterior modes are present in the final flow configuration.

Table 3: Sweep 2: Results

Hyperparameter	Importance	Correlation with training loss
α_0	0.422	-0.465
N_T	0.203	-0.299
$\mathrm{N}_T + N_U + N_L$	0.108	-0.205
N_U	0.098	-0.071
N_L	0.070	0.016

4.4.6.5. *Conclusion*

To the extent the VI+NF method was tested it has been lacking robustness. Temperature annealing generally increases mode retention, but based on the hyperparameter sweeps there are no regions of the tested hyperparameter space which guarantee successful training. There is scope for further investigation and systematic testing with a wider range of posteriors and flow configurations, but no clear direction emerges from the tests carried out so far.

In many of the training runs, despite large numbers of training samples used in each training step, modes collapse due to stochastic effects. Since reverse KL divergence $D_{KL}(q||p)$ does not penalise regions where $q = 0$ and $p > 0$, this cannot be fully prevented without changing the optimisation objective. Alternative mode covering training objectives exist, such as Rényi Divergence Variational Inference [98], allowing for use of general α-divergences which do not exhibit the mode-seeking behaviour. Use of a mode covering divergence that would prevent from mode collapse is where we see the most potential for increasing the robustness of VI+NF on multimodal targets, this is however left for further investigation.

An example of mode collapse is provided in Figure 13. Mode collapse cannot be fixed by importance sampling due to lack of flow support in the prior space corresponding to the lost mode. Lost modes will not be flagged by statistics such as effective sample size,

$$\mathrm{ESS} = \frac{\left(\sum_i^n w_i\right)^2}{\sum_i^n w_i^2} \quad w_i = \frac{p(x_i)}{q_i(x_i)} \tag{4.4.22}$$

— if the found mode is well matched by the flow, the effective sample size can be close to n suggesting a good approximation. Indication of lost modes can come from flow sample trace plots over the training duration, as it was in this case, or from jumps in loss value — however these indications will not be present whenever modes are missed by the flow entirely, at all stages of training.

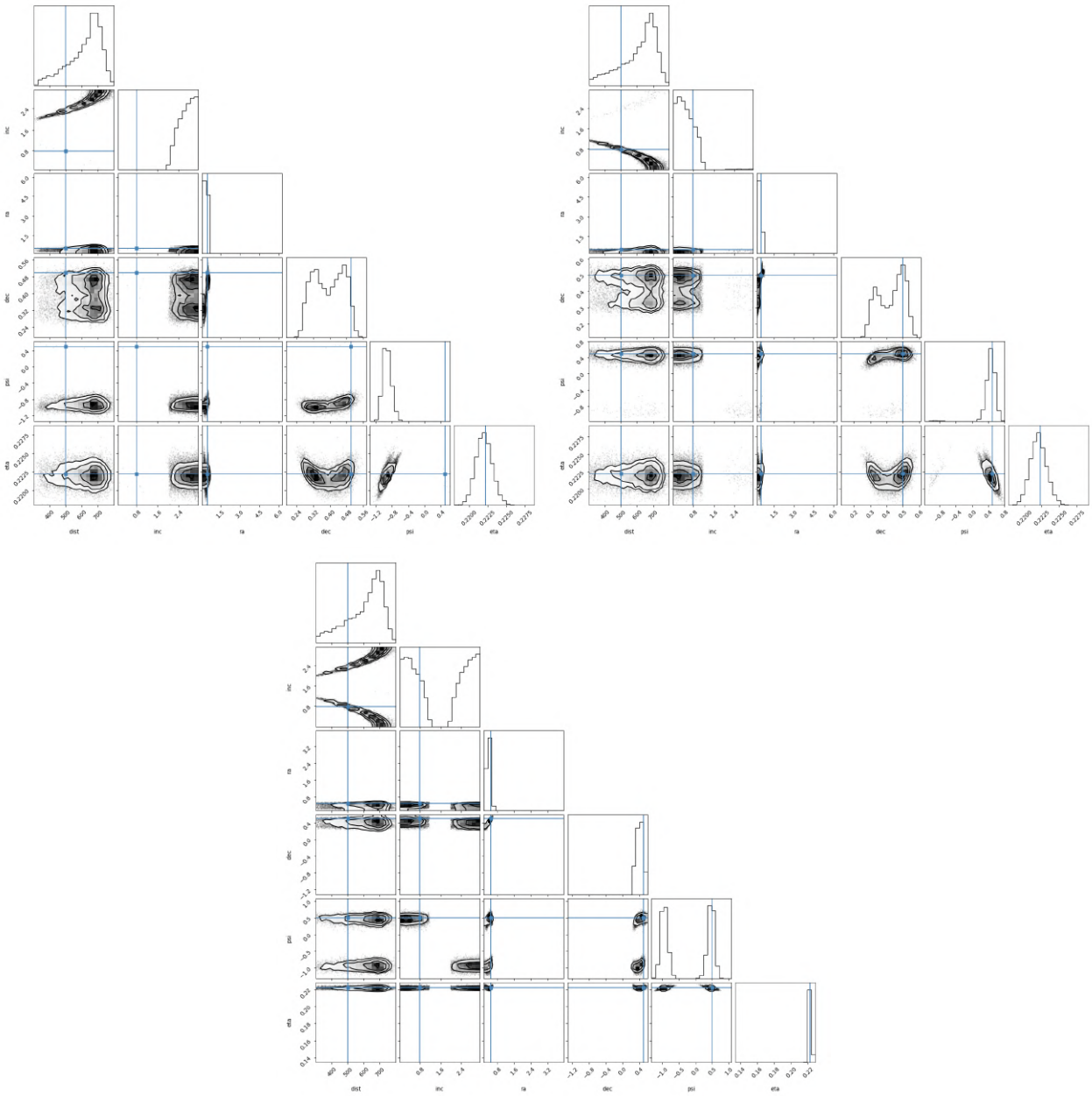


Figure 13: Mode collapse during training.

Example of the flow suffering from mode collapse on a 6-dimensional subspace of a posterior of a 2-detector BBH injection. At training epoch 4000 (top left), after annealing the posterior, one of the two inclination modes is found. At epoch 5000 (top right) the second mode is found, but the first one is almost entirely lost by the flow. In this example training proceeded to lose the first mode completely. Bottom: a different run where both modes are retained successfully, for reference.

4.5. NESTED VARIATIONAL INFERENCE (NVI)

Nested Variational Inference (NVI) [99] is an approach to Variational Inference in which the model is trained by learning a sequence of intermediate distributions that interpolate between the latent distribution and the target. This should be particularly advantageous in scenarios where successful training of the model heavily relies on the quality of proposal samples, such as training a flow with reverse KL divergence. In the early stages of training poor match between the model and the target distributions can be prohibitive of drawing high likelihood training samples; NVI addresses this problem by guiding the latent samples through a learnt annealing path towards the target.

Assume we want to learn a target density $p\colon X_N \to \mathbb{R}$ using NVI with normalising flows. We define a sequence of distributions

$$\{\gamma_k\colon X_k \to \mathbb{R}\}_{k=1}^{N}, \tag{4.5.1}$$

that interpolate between the initial proposal γ_1 and the target $p = \gamma_N$. These could be for example annealed versions of the target distribution at decreasing temperatures T. We want to simultaneously train a sequence of diffeomorphisms

$$\{f_k(\,\cdot\,;\theta_k)\colon X_k \to X_{k+1}\}_{k=1}^{N-1}, \tag{4.5.2}$$

such that the flow distributions $\{q_k\}$ are obtained as pushforwards of the intermediate targets $\{\gamma_{k-1}\}$ by the diffeomorphisms $\{f_{k-1}\}$, i.e.

$$q_k\colon X_k \to \mathbb{R}\}_{k=1}^{N}, \tag{4.5.3}$$

$$q_k(x_k) = \begin{cases} \pi_1(x_1), \quad \text{where } \pi_1 \text{ is the latent distribution of the flow } q_2 & \text{if } k = 1, \\ (f_1)_*(\pi_1)(x_2) \coloneqq \pi_1(x_1)|J_{f_1}(x_1)|^{-1}, \quad x_2 = f_1(x_1) & \text{if } k = 2, \\ (f_{k-1})_*(\gamma_{k-1})(x_k) \coloneqq \gamma_{k-1}(x_{k-1})|J_{f_{k-1}}(x_{k-1})|^{-1}, \quad x_k = f_{k-1}(x_{k-1}) & \text{for } k \geq 3, \end{cases} \tag{4.5.4}$$

are optimised to match the intermediate targets γ_k.

To achieve this, at each level of nesting we consider the pair of the unnormalised joint densities $\hat{\Gamma}_k, \check{\Gamma}_k$,

$$\hat{\Gamma}_k\colon X_k \times X_{k+1} \to \mathbb{R}, \quad \hat{\Gamma}(x_k, x_{k+1}) \coloneqq \gamma_k(x_k) q_{k+1}(x_{k+1}), \quad x_{k+1} = f_k(x_k), \tag{4.5.5}$$

$$\check{\Gamma}_k\colon X_k \times X_{k+1} \to \mathbb{R}, \quad \check{\Gamma}(x_k, x_{k+1}) \coloneqq \gamma_{k+1}(x_{k+1}) r_k(x_k), \quad x_k = f_k^{-1}(x_{k+1}), \tag{4.5.6}$$

and aim to minimise the KL divergence between them. Note that these are joint densities on the product space $X_k \times X_{k+1}$ — the *forward density* $\hat{\Gamma}_k$ is obtained

by taking the product of the intermediate target γ_k on the space X_k and the flow distribution q_{k+1} on the space X_{k+1}, where q_{k+1} is obtained by pushing γ_k forward to X_{k+1} through the forward diffeomorphism $f_k \colon X_k \to X_{k+1}$; similarly the *reverse density* $\check{\Gamma}_k$ is given by the product of the intermediate target γ_{k+1} on X_{k+1} and its pushforward to X_k by the inverse diffeomorphism f_k^{-1}, that is r_k, defined as

$$r_k(x_k) \coloneqq (f_k^{-1})_*(\gamma_{k+1})(x_k) = \gamma_{k+1}(x_{k+1})|J_{f_k}(x_k)|, \quad x_k = f_k^{-1}(x_{k+1}). \tag{4.5.7}$$

The overall optimisation objective is to parametrise the transforms $\{f_k\}$ to simultaneously minimise the divergence between the pairs of forward and reverse densities $\hat{\Gamma}_k, \check{\Gamma}_k$ at all levels of nesting — i.e. for all k. In the case when sampling from the target p is unavailable, this will be achieved by minimising a loss function $\mathcal{L}$ given by the sum of reverse KL divergences,

$$\mathcal{L} = D_{KL}(q_1||\gamma_1) + \sum_{k=2}^{N-1} D_{KL}(\hat{\Gamma}_k||\check{\Gamma}_k) \tag{4.5.8}$$

$$= \mathop{\mathbb{E}}_{x_1 \sim q_1}\left[\log \frac{q_1(x_1)}{\gamma_1(x_1)}\right] + \sum_{k=2}^{N-1} \mathop{\mathbb{E}}_{(x_k, x_{k+1}) \sim \hat{\Gamma}_k}\left[\log \frac{\hat{\Gamma}_k(x_k, x_{k+1})}{\check{\Gamma}_k(x_k, x_{k+1})}\right]. \tag{4.5.9}$$

The first term of this sum is minimised when the latent distribution of the flow, q_1, matches the initial intermediate target γ_1 — this can be achieved either through optimisation or more conveniently by choosing γ_1 to be equal to the latent distribution of the flow. Each term of the sum in the remaining term of the loss function expands as

$$\mathop{\mathbb{E}}_{(x_k, x_{k+1}) \sim \hat{\Gamma}_k}\left[\log \frac{\hat{\Gamma}_k(x_k, x_{k+1})}{\check{\Gamma}_k(x_k, x_{k+1})}\right] = \mathop{\mathbb{E}}_{\substack{x_k \sim \gamma_k \\ x_{k+1} \sim q_{k+1}}}\left[\log \frac{\gamma_k(x_k) q_{k+1}(x_{k+1})}{\gamma_{k+1}(x_{k+1}) r_k(x_k)}\right], \tag{4.5.10}$$

which is minimised when, at each stage of nesting, both the forward flow q_{k+1} matches the intermediate target γ_k and the reverse flow r_k matches γ_{k+1}. Note that in the case when $q_k = \gamma_k$ for all k, then automatically $r_k = \gamma_k$ for all k, which follows from the definition of r_k (by bijectivity of each of f_k). Also note that the intermediate targets γ_k need not be normalised — normalisation would add a constant term to the loss and hence would not affect optimisation.

It is clear to see that when the above sum of divergences is minimised, $q_k = \gamma_k$ for all k, and one can obtain a sample from the target $q_N = \gamma_N = p$ by sampling from the latent distribution q_1 and flowing this sample through the composition of all of the diffeomorphisms $\{f_k\}$, i.e.,

$$x = f_N \circ f_{N-1} \circ \cdots \circ f_1(x_1) \sim p, \quad x_1 \sim q_1. \tag{4.5.11}$$

Each of the diffeomorphisms f_k would usually be a composition of a number of diffeomorphisms itself, e.g. a fixed number M of trainable spline blocks. One can therefore view performing nested variational inference with flows as training a single discrete flow obtained via the diffeomorphism $f = f_N \circ f_{N-1} \circ \cdots \circ f_1$, with the difference being that every M blocks we demand the intermediate target to be some γ_k. So instead of letting the flow find its own annealing path between the latent and the target distribution, we lay for it a sequence of road signs — the intermediate distributions γ_k, through which we force the annealing path to go.

In real scenarios, such as the gravitational wave problem, we will not only be unable to sample from the target $p = \gamma_N$, but also the intermediate annealed targets γ_k, and hence we need to resort to importance weighting to evaluate the loss. Given a sample $x_k \sim q_k$, we can weigh it according to γ_k, the weight being

$$w = \frac{\gamma_k(x_k)}{q_k(x_k)}. \tag{4.5.12}$$

We can then approximate the expectations in the loss as

$$\mathop{\mathbb{E}}_{\substack{x_k \sim \gamma_k \\ x_{k+1} \sim q_{k+1}}} \left[\log \frac{\gamma_k(x_k) q_{k+1}(x_{k+1})}{\gamma_{k+1}(x_{k+1}) r_k(x_k)} \right] \approx \frac{1}{S} \sum_{i=1}^{S} w_i \log \frac{\gamma_k(x_k^{(i)}) q_{k+1}(x_{k+1}^{(i)})}{\gamma_{k+1}(x_{k+1}^{(i)}) r_k(x_k^{(i)})}, \tag{4.5.13}$$

$$\{x_k^{(i)} \sim q_k,\, x_{k+1}^{(i)} \sim q_{k+1}\}_{i=1}^{S}. \tag{4.5.14}$$

4.5.1. *Results*

4.5.1.1. *Gaussian mixture*

The NVI algorithm was implemented by extending the JAX normalising flow VI implementation (DISTRAX) to accommodate for more than one normalising flow being trained at a time. Each of the trained flows is implemented as a separate neural network state ϕ_k of the same flow network, ensuring identical flow architecture for all kernels. For a training scheme with N normalising flows, the loss function $\mathcal{L}(\phi_1, \ldots, \phi_N)$ is then a function of the N network parameter vectors, so stochastic gradient descent is performed by taking a separate vector gradient with respect to each of ϕ_i.

The initial test of the implementation was performed using NVI with a single intermediate distribution to learn a 5D bimodal Gaussian target. The intermediate distribution is annealed at $T = 5$. The same coupling flow architecture is used for both of the trained flows, the flow consisted of 4 transforms, 12 spline bins, parametrised by a 2 layer deep MLP with 200 neurons width. The training result is presented in Figure 14. Both flows successfully train, however there is a lot of training instability, which we can see by examining the loss variances over the training duration.

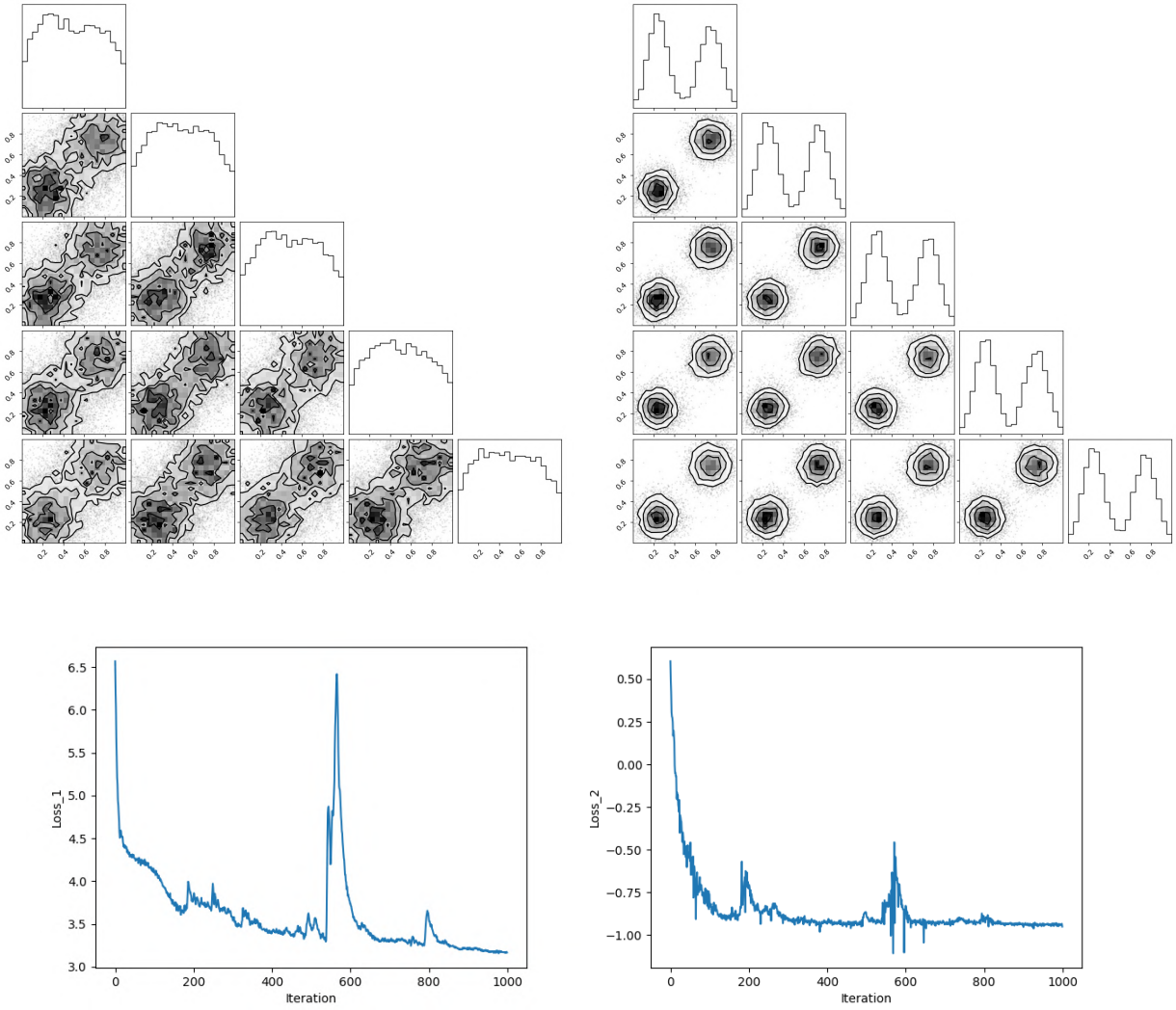


Figure 14: NVI training on a bimodal Gaussian mixture in 5 dimensions.

Two flows are trained simultaneously, flow 1 learns $p^{1/5}$, whereas flow 2 learns the full target p. Top panel shows samples from flow 1 and 2 at the end of training, while the bottom panel shows the corresponding training losses.

4.5.1.2. *BBH posterior*

The gravitational wave problem test was performed on the 5-dimensional subspace of the BBH signal analysed earlier in the chapter. Different temperature configurations of an NVI scheme with one intermediate distribution were tested (T=10, 20, 30, 50, 100), neither leading to retention of both modes in the second flow. In each of these configurations one of the target modes was (roughly) located, rather than the flow locking onto a secondary tail feature, which is an improvement over regular VI without

annealing or intermediate targets. An example training result with intermediate distribution at $T = 30$ is shown in Figure 15.

The natural direction was to extend the scheme by incorporating more intermediate targets, decreasing the information content increase between subsequent distributions. After adding the second intermediate distribution the scheme became highly unstable, with all but the initial flow (mapping between the flat prior and the first intermediate target) collapsing entirely, potentially due to the high variance importance weights in the loss components. No flow configuration was found that would prevent that.

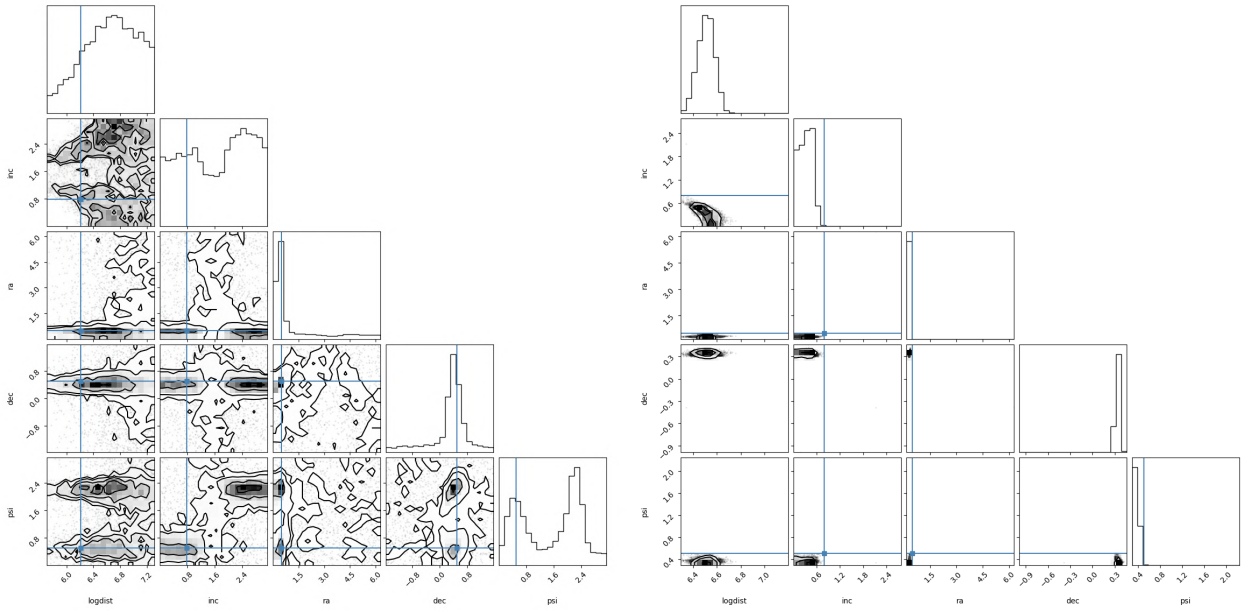


Figure 15: NVI training result on the BBH 5-dimensional posterior subspace.

Two flows are trained simultaneously, flow 1 learns $p^{1/30}$, whereas flow 2 learns the full target p. Top panel shows samples from flow 1 and 2 at the end of training.

4.5.2. *Other variants and possible forward directions*

In the NVI algorithm the flows are defined to be the pushforwards of the intermediate distributions, samples from which are unavailable in most applications, and in particular in the case these represent the annealed gravitational wave posterior. In that case, as described above, evaluating the loss requires resorting to importance sampling, which in our scenario increases the variance in the Monte Carlo estimates of the calculated KL divergences, leading to instability and poor training performance when more intermediate targets are introduced.

There is however a different variant of this training scheme – Annealed Variational Objective (AVO) [100] variant, which uses the flows as extensions of the preceding

forward kernels, rather than extensions of the intermediate distributions, and hence does not require an importance sampling step, as expectation values are taken only over flow distributions. This variant could improve the stability of the training scheme when samples from intermediate targets are unavailable, as it is the case in the GW case.

Concurrent work by Arbel et al. [101] has further explored the normalising flow setting of annealed variational inference and has found that including additional Metropolis Hastings steps was crucial for the performance of their samplers in more difficult settings. This could be a necessary step for an NVI implementation in the GW case, due to the complexity of the distribution.

Yet another approach is optimising the annealing path as well as the flow kernels, rather than fixing the inverse temperatures β_k, as explored in [99]. The learnt β_k tend to equalize the KL divergence between the consecutive annealed targets, which is not guaranteed by taking a linearly sampled inverse temperatures β_k. This route could potentially be promising.

4.5.3. *Issues with the implementation*

Issues with the NVI implementation experienced in the application to the GW problem were caused by serveral factors — the necessity of importance sampling for evaluating the intermediate KL divergences had negative implications on the robustness of the method, leading to high variance loss and therefore poor training. Due to the shape of the GW posterior distributions a high temperature is necessary to obtain a unimodal target, so in order to obtain an optimal annealing path many intermediate distributions are expected to be needed, reducing the efficiency of the method. In cases that were tested few intermediate targets were used (1-2), chosen to achieve unimodality at the intermediate step, rather than a uniform distribution of inverse temperatures $\{\beta_k\}$. These not-optimal annealing paths did not lead to retention of both modes in the final target. There is scope for further experimentation with both NVI with more intermediate targets, possibly in the AVO case to eliminate the need for importance sampling, or with the use of extra MCMC steps.

Part V.
Bernstein Polynomial Flows

Normalising flows with Bernstein-type polynomial bijectors have been argued [102] to constitute a class of universal flows providing improved robustness to perturbation in the data space and increased numerical stability with increasing transform complexity. Practical implementation advantages are improved interpretability and suitability to model compactly supported targets via easy range control.

Rational quadratic spline as well as affine flows have been found to struggle on some subspaces of the CBC posteriors (NESSAI struggles on distributions with sharp edges, due to the non-trivial space topology both NESSAI and DINGO require phase marginalisation [68; 84]), so there is potential for improvement that could be brought by Bernstein bijectors both to the variational inference method described in previous chapters, as well as to existing PE algorithms utilising normalising flows, such as the flow-aided samplers NESSAI and I-NESSAI.

There are currently no easily available open source implementations of Bernstein bijectors in JAX and PYTORCH — the goal is to provide such implementations, make them available within the GW data analysis community, and test the potential for improvements these could bring to the different normalising flow use cases within GW parameter estimation.

5.1. BERNSTEIN-TYPE POLYNOMIALS

A Bernstein polynomial normalising flow (BPF) is a normalising flow (Section 3.1.1) where the bijectors are constructed from element-wise Bernstein-type polynomial transforms.

Bernstein-type polynomials are defined as linear combinations of Bernstein basis polynomials. Formally, we have the following definitions [103]:

Definition 5.1.1 (Bernstein basis polynomial)**.** Let $n \in \mathbb{N}$. The k^{th} Bernstein basis polynomial of degree n is defined as

$$b_{k,n}\colon [0,1] \to \mathbb{R}, \quad x \mapsto \binom{n}{k} x^k (1-x)^{n-k}, \quad 0 \leq k \leq n \tag{5.1.1}$$

Definition 5.1.2 (Bernstein-type polynomial)**.** Let $n \in \mathbb{N}$. A Bernstein polynomial of degree n, $B_n^{\boldsymbol{\alpha}}$, is a linear combination of degree n Bernstein basis polynomials, that is

$$B_n^{\boldsymbol{\alpha}}\colon [0,1] \to \mathbb{R}, \quad x \mapsto \sum_{k=0}^{n} \alpha_k \binom{n}{k} x^k (1-x)^{n-k}, \quad \boldsymbol{\alpha} = (\alpha_0, \ldots, \alpha_n) \in \mathbb{R}^{n+1} \tag{5.1.2}$$

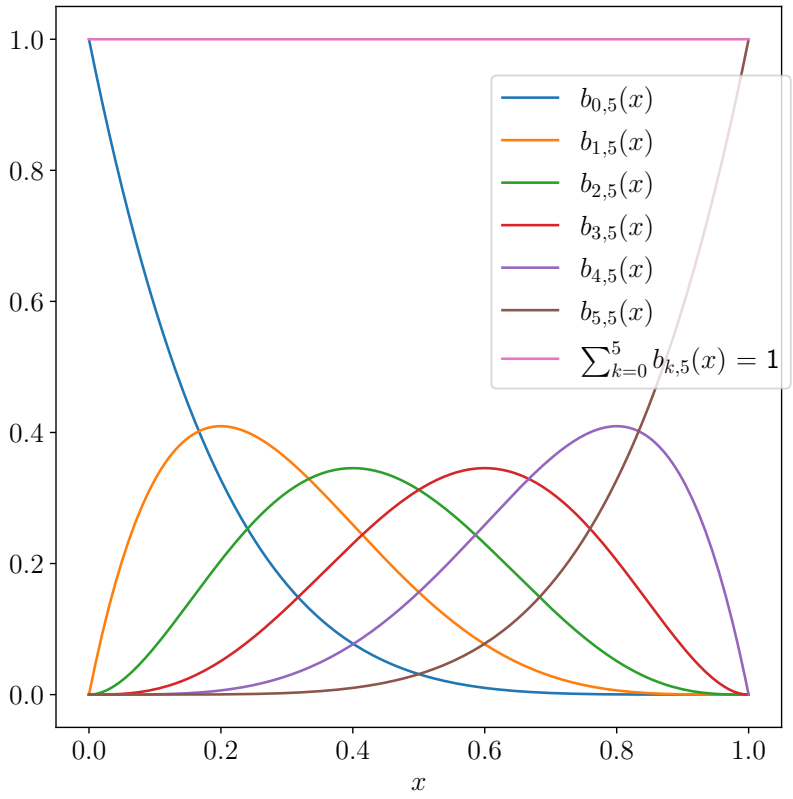


Figure 16: Bernstein polynomial basis.

Degree 5 Bernstein basis polynomials. Bernstein basis polynomials form a partition of unity, $\sum_{k=0}^{n} b_{k,n}(x) = 1 \; \forall \, x \in [0, 1]$.

Bernstein basis polynomials of degree n provide a basis for the vector space Π_n of polynomials of degree at most n with real coefficients on $[0, 1]$. Bernstein polynomials were introduced by S. Bernstein in a proof [104] of the Weierstrass approximation theorem, stating that any continuous function $f \in C_{[0,1]}$ on the unit interval can be uniformly approximated by polynomials, showing that the Bernstein polynomial

$$B_n(f)(x) = \sum_{k=0}^{n} f\left(\frac{k}{n}\right) b_{k,n}(x) \tag{5.1.3}$$

uniformly converges to f as $n \to \infty$. This universal approximation property of Bernstein polynomials makes them particularly appealing for application to normalising flows, where we need the transforms to approximate arbitrarily complex diffeomorphisms of the latent and data space.

5.1.1. *Invertible Bernstein-type Polynomials*

In the context of normalising flows Bernstein polynomials are not used to approximate the PDF itself, but to approximate an invertible reparametrisation of the latent

space — hence in order to construct a normalising flow from element-wise Bernstein-type polynomial transforms, we need to ensure invertibility of the Bernstein-type polynomials. If this requirement is not immediately obvious one can think of the simplest case of a 1-dimensional problem with a uniform latent distribution — in this case the normalising flow bijector approximates the CDF of the target PDF, and hence is necessarily strictly monotonic, or equivalently, invertible.

For a general coefficient vector $\boldsymbol{\alpha} \in \mathbb{R}^{n+1}$, the Bernstein-type polynomial $B_n^{\boldsymbol{\alpha}}$ need not be bijective, however we can constrain $\boldsymbol{\alpha}$ to ensure bijectivity, and hence inveribility of the transform.

Proposition 5.1.3 (Strict monotonicity of Bernstein-type polynomials [102])**.** *Let $n \in \mathbb{N}$ and $\alpha_0 < \alpha_1 < \dots \alpha_n$ be a strictly increasing sequence of real values $\alpha_k \in \mathbb{R}$, $\forall\, k$. Then the Bernstein polynomial $B_n^{\boldsymbol{\alpha}}$ of degree n corresponding to the basis coefficient vector $\boldsymbol{\alpha} = (\alpha_0, \alpha_1, \dots \alpha_n)$ is strictly increasing on the entirety of its domain.*

Since strict monotonicity implies invertability, choosing a strictly increasing sequence of coefficients α_k ensures invertability of the Bernstein polynomial. It should be noted here that Proposition 5.1.3 is only a one sided implication, that is the monotonicity of the α coefficients is a sufficient but not a necessary condition for Bernstein polynomial invertability.

To prove that the Bernstein-type polynomial with strictly increasing basis coefficients is monotonically increasing is equivalent to showing that the derivative $\mathrm{d}B_n^{\boldsymbol{\alpha}}/\mathrm{d}x$ is positive over the domain $[0, 1]$. The following propositions hold, with Proposition 5.1.5 providing the proof of the monotonicity claim.

Proposition 5.1.4 (Bernstein basis polynomial derivative)**.** *The derivative of a Bernstein basis polynomial $b_{k,n}$ can be expressed in terms of Bernstein basis polynomials of lower degree,*

$$b'_{k,n}(x) = n(b_{k-1,n-1} - b_{k,n-1}), \quad x \in [0,1] \quad k \geq 1 \tag{5.1.4}$$

$$b'_{k,n}(x) = -n\, b_{k,n-1}, \quad k = 0 \tag{5.1.5}$$

Proposition 5.1.5 (Bernstein-type polynomial derivative)**.** *The derivative of a Bernstein-type polynomial of degree n is a Bernstein polynomial of degree $n - 1$.*

For a Bernstein polynomial $B_n^{\boldsymbol{\alpha}}$, we have

$$\frac{dB_n^{\boldsymbol{\alpha}}}{dx} = \sum_{k=0}^{n} \alpha_k b'_{k,n} \tag{5.1.6}$$

$$= -n\alpha_0\, b_{0,n-1} + \sum_{k=1}^{n} \alpha_k n(b_{k-1,n-1} - b_{k,n-1}) \tag{5.1.7}$$

$$= n \sum_{k=1}^{n} (\alpha_k - \alpha_{k-1}) b_{k-1,n-1} \tag{5.1.8}$$

$$= n \sum_{k=0}^{n-1} (\alpha_{k+1} - \alpha_k) b_{k,n-1}. \tag{5.1.9}$$

In particular, if $\alpha_k > \alpha_{k-1},\ \forall\, k$, then the Bernstein polynomial has a strictly positive derivative, so that it is monotonically increasing and hence invertible.

Analytic inversion of the monotonic Bernstein-type polynomials is not possible for polynomial degrees higher or equal to 5, so inverting the Bernstein-type polynomials needs to be performed numerically, unlike in the case of rational quadratic splines or affine transforms, where explicit inversion is possible. Given $y \in [\alpha_0, \alpha_n]$, inverting a monotonic Bernstein-type polynomial $B_n^{\boldsymbol{\alpha}}$ requires finding the unique $x \in [0, 1]$ such that

$$y = B_n^{\boldsymbol{\alpha}}(x) = \sum_{k=0}^{n} \alpha_k \binom{n}{k} x^k (1-x)^{n-k} \iff \sum_{k=0}^{n} (\alpha_k - y) \binom{n}{k} x^k (1-x)^{n-k} = 0, \tag{5.1.10}$$

which can be done by either means of interpolation, or with an efficient root finding algorithm such as bisection search. Numerical inversion is an obvious drawback of the use of Bernstein-type polynomials as normalising flow bijectors, since the computation of the inverse will be needed for either sampling from the flow (for a flow defined in the normalising direction $f\colon X \to Z$) or density evaluation at arbitrary points in the data space (for a flow defined in the generative direction $f\colon Z \to X$).

5.1.2. *Range of a Bernstein polynomial*

When applying a normalising flow to a real-life problem the target distribution will often be compactly supported, for example on a high dimensional hypercube. Bernstein-type polynomial transforms naturally lend themselves to transforming compactly supported targets, as their range is easily controlled by the choice of basis coefficients $\boldsymbol{\alpha}$.

Proposition 5.1.6 (Range of a monotonic Bernstein-type Polynomial)**.** *Let $\boldsymbol{\alpha} \in R^{n+1}$ be a coefficient vector with strictly increasing entries, as in the previous proposition. Then the range of $B_n^{\boldsymbol{\alpha}}$ is $[\alpha_0, \alpha_n]$.*

Proof. Since $\{\alpha_k\}_{k=0}^n$ is a strictly increasing sequence, $B_n^{\boldsymbol{\alpha}}$ is strictly increasing. The range of $B_n^{\boldsymbol{\alpha}}$ is given by its values at the domain endpoints, $[B_n^{\boldsymbol{\alpha}}(0), B_n^{\boldsymbol{\alpha}}(1)] = [\alpha_0, \alpha_n]$. □

In the case where the target distribution is supported on an unbounded space, with data variables $x \in \mathbb{R}$, using a Bernstein polynomial is still possible, however only in composition with a pair of bijective transforms, $g\colon \mathbb{R} \to [0,1]$, $h\colon [0,1] \to \mathbb{R}$, which map the data variable into the domain of the Bernstein polynomial and back into the data space, obtaining a transform with unbounded domain and range of the form

$$h \circ B_n^{\boldsymbol{\alpha}} \circ g\colon \mathbb{R} \to \mathbb{R}. \tag{5.1.11}$$

A simple choice for g and h is the sigmoid-logit transform pair,

$$x \in \mathbb{R}, \quad g(x) \equiv \sigma(x) = \frac{1}{1+e^{-x}} \in [0,1], \tag{5.1.12}$$

$$x \in [0,1], \quad h(x) \equiv \sigma^{-1}(x) = \ln \frac{x}{1-x} \in \mathbb{R}. \tag{5.1.13}$$

When using the two extra transforms, the overall transform derivative will involve derivatives of the transforms g and h. According to the chain rule, the derivative of the composition $h \circ B_n^{\boldsymbol{\alpha}} \circ g$ can be computed as $h'(B_n^{\boldsymbol{\alpha}} \circ g(x))(B_n^{\boldsymbol{\alpha}})'(g(x))g'(x)$. Note that the derivatives of g and h do not cancel out even if g and h are a function-inverse pair, as they are evaluated at different points of the domain.

5.1.2.1. *Bernstein-type polynomial as a normalising flow bijector*

So far we have considered 1-dimensional Bernstein polynomials $B_n^{\boldsymbol{\alpha}}\colon [0,1] \to [\alpha_0, \alpha_n]$, while in order to construct a flow we generally need to transform a d-dimensional latent/data space input. This is achieved by applying a set of d 1-dimensional transforms elementwise, as is the case with the neural spline flows.

Definition 5.1.7 (d-dimensional Bernstein-type polynomial transform)**.** Given a vector $\boldsymbol{x} = (x_1, \ldots, x_d) \in [0,1]^d$, and a set of coefficient vectors $\{\boldsymbol{\alpha}^i\}_{i=0}^d$, $\boldsymbol{\alpha}^i \in \mathbb{R}^{n+1}$, a d-dimensional Bernstein transform of degree n, $B\colon [0,1]^d \to [\alpha_0^{(1)}, \alpha_n^{(1)}] \times \cdots \times [\alpha_0^{(d)}, \alpha_n^{(d)}]$, is defined by transforming x elementwise with Bernstein polynomials $B_n^{\boldsymbol{\alpha}^i}$, that is,

$$B(\boldsymbol{x}) = (B_n^{\boldsymbol{\alpha}^1}(x_1), B_n^{\boldsymbol{\alpha}^2}(x_2), \ldots, B_d^{\boldsymbol{\alpha}^d}(x_d)) \in \mathbb{R}^d \tag{5.1.14}$$

In order to construct a Bernstein polynomial normalising flow, we require the full d-dimensional Bernstein polynomial transform to be invertible. By 5.1.3, this is ensured by choosing $\boldsymbol{\alpha}^i \in \mathbb{R}^{n+1}\, \forall\, i$ to be coefficient vectors with strictly increasing entries. In practice, the parameters $\{\boldsymbol{\alpha}^i = (\alpha_0^i, \alpha_1^i, \ldots \alpha_n^i)\}_{i=0}^d$ are determined by the outputs of a

neural network, which will generally not fulfill the monotonicity constraints without further manipulation. Given an unconstrained NN output parameter vector $\boldsymbol{\gamma} = (\gamma_1, \gamma_2, \ldots, \gamma_n) \in \mathbb{R}^n - \{\mathbf{0}\}$, we can construct a strictly increasing sequence of $0 = \alpha_0 \leq \alpha_1 \leq \ldots \alpha_n = 1$, by setting, for example,

$$\alpha_0 = 0, \tag{5.1.15}$$

$$\alpha_k = \frac{\sum_{i=0}^{k} |\gamma_i|}{\sum_{i=0}^{n} |\gamma_i|}. \tag{5.1.16}$$

In order to compute the density of a normalising flow it is essential to compute the Jacobian determinant of the Bernstein polynomial transform. In general any of the transforms $B_n^{\boldsymbol{\alpha}^i}$ can depend on any of $x_1, \ldots, x_n$ — the dependencies will depend on the flow architecture (coupling flow, autoregressive flow), but will usually be chosen so that the Jacobian of the Bernstein transform is upper or lower diagonal. The Jacobian determinant is then computed as a product of the diagonal entries,

$$\det(\mathbf{J}_B) = \prod_{i=1}^{d} \frac{\mathrm{d}B_n^{\boldsymbol{\alpha}^i}}{\mathrm{d}x_i}, \quad \frac{\mathrm{d}B_n^{\boldsymbol{\alpha}^i}}{\mathrm{d}x_i} = \sum_{k=0}^{n} \alpha_k^i b'_{k,n}(x_i), \quad \boldsymbol{\alpha}^i = (\alpha_0^i \ldots, \alpha_n^i), \tag{5.1.17}$$

where $b'_{k,n}$ are the Bernstein basis derivatives as given in Prop 5.1.4.

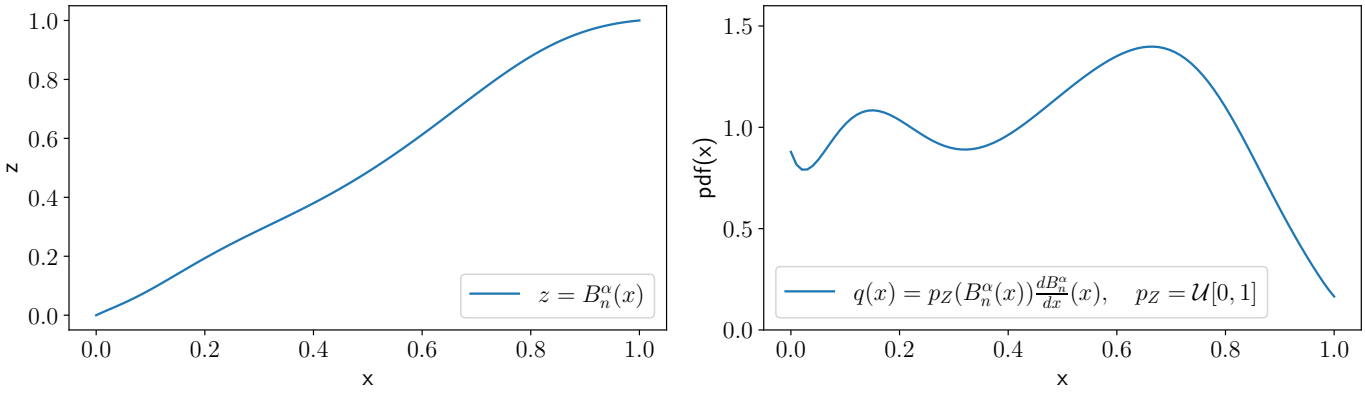


Figure 17: Bernstein polynomial transform.

Example of a randomly initiated degree 15 Bernstein transform $B_n^{\boldsymbol{\alpha}}\colon X \to Z$ and the resulting density obtained by transforming a uniform latent distribution on the latent domain $Z = [0, 1]$.

5.2. IMPLEMENTATION

Two implementations of the Bernstein polynomials bijectors were written, one in PyTorch as a transform class in the glasflow [105] Python module, for compatibility with the nessai and i-nessai algorithms which use the glasflow affine and spline transforms by default, as well as a jax implementation as a distrax bijector class for compatibility with the variational inference algorithm developed for the purposes of this work. Both these implementations are equivalent and follow the same practical solutions — operating on logarithms of the basis polynomials to avoid numerical under and overflows, transforming an unconstrained network output into a sequence of strictly increasing coefficients according to Equations 5.1.15-5.1.16 and inverting the polynomials using either linear interpolation or a bisection search, according to user preference. The pseudocode with the outline of the implementation is presented below.

Algorithm 1 Bernstein bijector of degree n on $[a, b] \subset \mathbb{R}$ (forward and inverse)

Require: scalar input $x \in [0, 1]$, NN output $\boldsymbol{\gamma} = (\gamma_0, \dots \gamma_{n-1}) \in \mathbb{R}^n$
  $\alpha_0 \leftarrow a$ ▷ $a = \alpha_0 < \alpha_1 < \cdots < \alpha_n = b$
  $\alpha_k \leftarrow a + \frac{\sum_{i=0}^{k} |\gamma_i| + \epsilon}{\sum_{i=0}^{n-1} |\gamma_i| + \epsilon}(b - a)$. ▷ $0 < \epsilon \ll 1$
  if forward transform **then**
    $y \leftarrow \exp \log \sum_{k=0}^{n} \alpha_k \exp \log b_{k,n}(x)$
    $\beta_k = n(\alpha_{k+1} - \alpha_k)$
    $\log dy/dx = \log \sum_{k=0}^{n-1} \beta_k \exp \log b_{k,n-1}(x)$
  else if inverse transform **then**
    if linear interpolation **then**
      $\boldsymbol{x} = a + (b - a)(0, \frac{1}{r}, \frac{2}{r}, \dots, \frac{r-1}{r}, 1), \quad r \in \mathbb{N}$
      $\boldsymbol{y} =$ forward transform of $\boldsymbol{x}$
      $f :=$ linear interpolation of the inverse mapping $\boldsymbol{y} \mapsto \boldsymbol{x}$
      $y \leftarrow f(x)$
    else if bisection search **then**
      $g(y') :=$ $x -$ forward transform of y'
      $y \leftarrow y'$ s.t. $g(y') \approx 0$ found by a bisection search
    end if
    $\log dy/dx = -\log \sum_{k=0}^{n-1} \beta_k \exp \log b_{k,n-1}(x)$
  end if
  Return $y, \log dy/dx$

5.3. BERNSTEIN BIJECTOR PERFORMANCE

The implemented Bernstein flows were tested on simple target distributions to validate the implementation. Figure 18 shows a comparison between the training of Bernstein polynomial flows and neural spline flows (both in the glasflow implementation) to estimate a density of correlated Gaussian targets. Even in this simple

example the Bernstein flow achieves lower KL divergence for both the flows with uniform and Gaussian latent distribution, and in the case of the uniform latent distribution is able to achieve a good approximation in fewer iterations than the neural spline flow for all choices of transform complexity. Most importantly successful training validates the implementation.

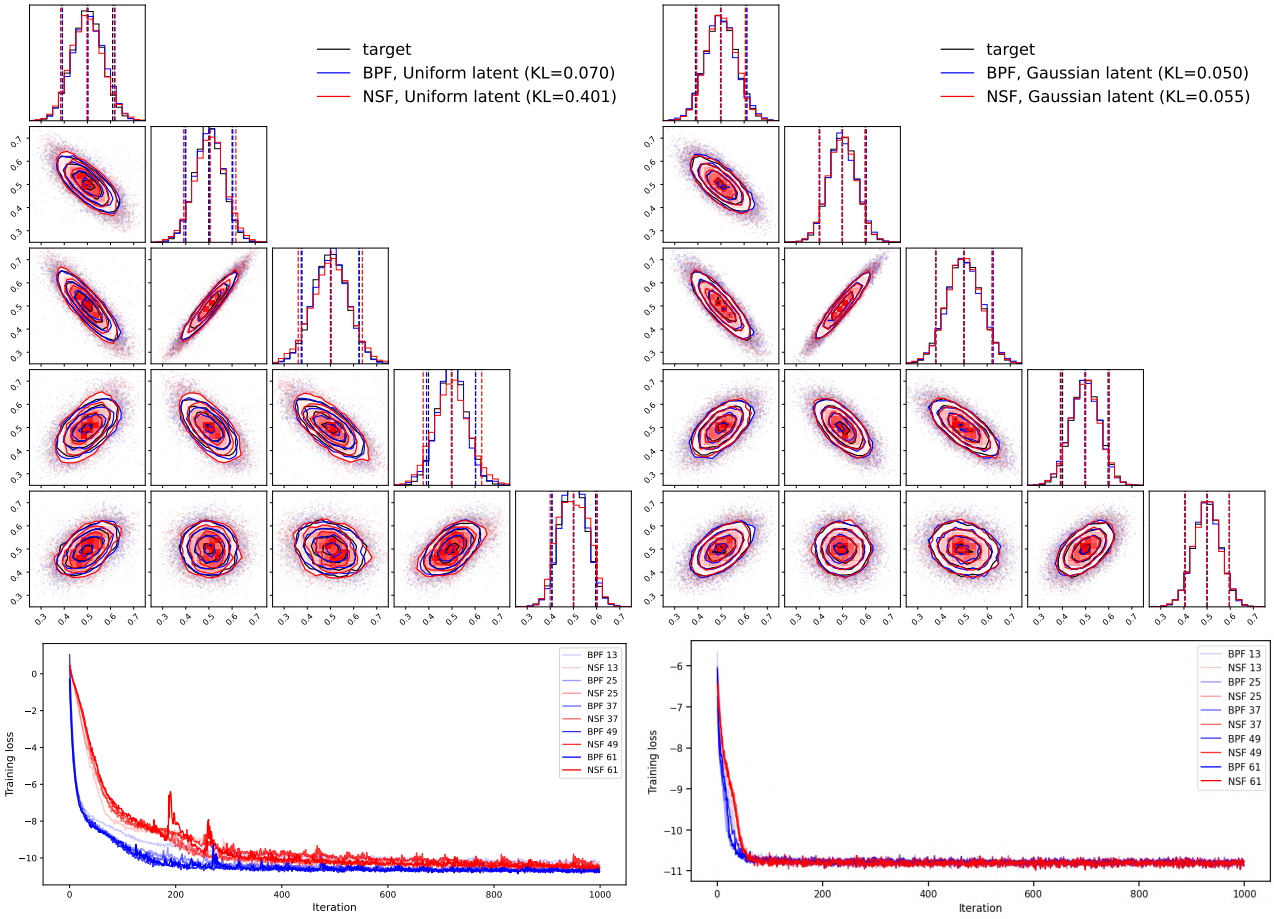


Figure 18: Bernstein flow and neural spline flow training results on a 5-dimensional correlated Gaussian target.

Left panel corresponds to flows with uniform latent distribution, while the right panel to flows with a Gaussian latent distribution. Flows are trained on samples from the target, as density estimators. Top panel shows an example training run where both the Bernstein flow and the neural spline flow have 5 transforms and 46 trainable parameters per transform (15 spline bins, equivalent to Bernstein degree 46). Bottom panel shows the comparison between the training loss for training runs with increasing number of transform trainable parameters, where higher colour intensity represents a larger parameter set. The interpolated inversion of the Bernstein flows was used.

Upon testing the JAX Bernstein polynomial flow for the variational inference GW problem (described in detail in Section 4.4.6.1), it has been concluded that the flow performs comparably to the neural spline flow and within the scope of the conducted tests no improvement was seen with respect to mode retention. The results of the training on the 5-dimensional BBH problem are presented in Figure 19. For equal flow complexity (equal number of transforms and trainable bijector parameters), the Bernstein flow achieves a final loss that is higher than that of the spline flow, with the Bernstein flow training at a slower rate specifically in the final stages of training, where the target annealing temperature approaches $T = 1$, reaching full problem complexity. Both flows show similar loss variance, suggesting comparable training stability. A Bernstein flow with double the number of transforms as the spline flow reduces the loss diference between the Bernstein and spline flows to a negligible level, at the cost of decreased training speed.

Since in the GW posterior case there is no significant difference in the performance between the different flow bijectors, it can be concluded that the problems with applying the variational inference algorithm to the gravitational wave posterior do not stem from the bijector choice, and improvements should be sought within changes to the overall training scheme rather than flow architecture.

The JAX implementation of the Bernstein bijectors provides comparable training speeds to the DISTRAX rational quadratic splines, when tested with the VI algorithm, with Bernstein bijectors being about 1.5 times slower in training. The GLASFLOW implementation of the Bernstein bijector is 3-5 times slower than the corresponding rational quadratic spline transforms. The difference suggests that the GLASFLOW implementation could be further optimised. The comparison is performed between flows with equal number of trainable parameters.

5.4. BOUNDS OF THE BERNSTEIN POLYNOMIAL TRANSFORM

Bernstein polynomial derivatives are bounded by a function of the Bernstein degree and Bernstein basis coefficients, imposing bounds on the Bernstein transform Jacobian determinant, which enters the computation of the normalising flow density. Understanding these bounds in relation to the properties of the target density could inform the decision on the Bernstein polynomial degree and the number of transform blocks necessary to obtain a flow just expressive enough to represent a chosen target density.

To derive these bounds, we first note that Bernstein basis polynomials form a partition of unity, that is

$$\sum_{k=0}^{n} b_{k,n} = 1, \quad \forall\, n \in \mathbb{N}. \tag{5.4.1}$$

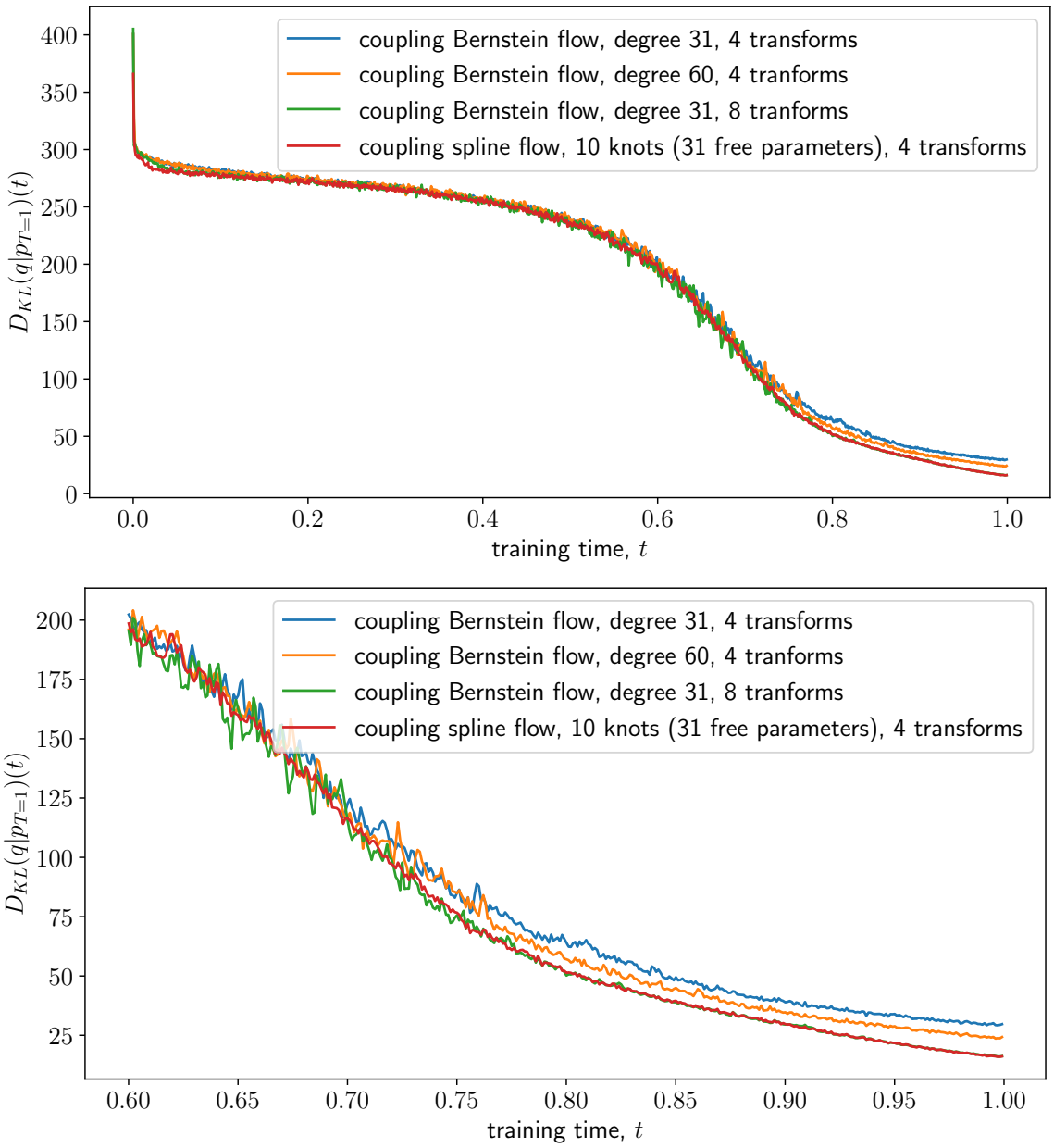


Figure 19: Comparison between the Bernstein polynomial flow and the neural spline flow performance on the 5-dimensional BBH problem.

Top panel shows the KL divergence between the flows and the temperature $T = 1$ target for the full training runs, whereas the bottom panel shows the last 40% of the training steps, where the performance of the different flows can be seen to diverge.

We can use this property to calculate an upper and lower bound on the Bernstein polynomial derivative. Using this property, together with the non-negativity of the Bernstein basis polynomials, the following inequalities hold

$$\frac{\mathrm{d}B_n^{\boldsymbol{\alpha}}}{\mathrm{d}x} = n\sum_{k=0}^{n-1}(\alpha_{k+1}-\alpha_k)b_{k,n-1} \leq n\max(\alpha_{k+1}-\alpha_k)\sum_{k=0}^{n-1}b_{k,n-1} = n\max(\alpha_{k+1}-\alpha_k), \tag{5.4.2}$$

$$\frac{\mathrm{d}B_n^{\boldsymbol{\alpha}}}{\mathrm{d}x} = n\sum_{k=0}^{n-1}(\alpha_{k+1}-\alpha_k)b_{k,n-1} \geq n\min(\alpha_{k+1}-\alpha_k)\sum_{k=0}^{n-1}b_{k,n-1} = n\min(\alpha_{k+1}-\alpha_k). \tag{5.4.3}$$

In particular, for a Bernstein transform with with range $[0,1]$, i.e. with $\alpha_0 = 0$ and $\alpha_n = 1$, it holds that $\max(\alpha_{k+1}-\alpha_k) \leq 1$ and $\min(\alpha_{k+1}-\alpha_k) \geq 0$, and therefore

$$(\alpha_0 = 0) \wedge (\alpha_n = 1) \implies 0 \leq \frac{\mathrm{d}B_n^{\boldsymbol{\alpha}}}{\mathrm{d}x} \leq n \tag{5.4.4}$$

This upper bound will govern the expressivity of the transform. Assume we use a one-dimensional Bernstein transform as a diffeomorphism between latent space Z and a data space X, with $Z = X = [0,1]$.

If the transform is defined in the normalising direction, that is from the data to the latent space,

$$B_n^{\boldsymbol{\alpha}} \colon X \to Z = [0,1], \tag{5.4.5}$$

the probability density q_X of a resulting single-block normalising flow will be bounded by

$$q_X(x;\boldsymbol{\alpha}) = p_Z(B_n^{\boldsymbol{\alpha}}(x))\left|\frac{\mathrm{d}B_n^{\boldsymbol{\alpha}}}{\mathrm{d}x}\right| \leq n\, p_Z(B_n^{\boldsymbol{\alpha}}(x)). \tag{5.4.6}$$

If, alternatively, the transform is defined with the latent space Z as its domain,

$$B_n^{\boldsymbol{\alpha}} \colon Z \to X = [0,1], \tag{5.4.7}$$

$$q_X(x;\boldsymbol{\alpha}) = p_Z((B_n^{\boldsymbol{\alpha}})^{-1}(x))\left|\frac{\mathrm{d}B_n^{\boldsymbol{\alpha}}}{\mathrm{d}z}\right|^{-1} \geq \frac{1}{n}\, p_Z((B_n^{\boldsymbol{\alpha}})^{-1}(x)). \tag{5.4.8}$$

These bounds impose constraints on how closely we can approximate the target with the flow. For example, if the flow is defined in the normalising direction, and the probability density of the target at x exceeds $n\, p_Z(B_n^{\boldsymbol{\alpha}}(x))$, the target will not be matched by the flow at that point for any choice of flow parameters $\boldsymbol{\alpha}$ — we therefore know we need to increase n to obtain an expressive enough flow to match the target density. These bounds can be easily extended to a flow composed of a finite number of Bernstein transform blocks — since the derivative factor is multiplicative

under composition, the factor n would in that case be replaced by n^m, where m is the number of blocks. In the most general multi-dimensional case, where the probability density changes by the Jacobian determinant of the transform, the factor n will be replaced with n^{md}, where d is the dimensionality of the domain and m is the number of degree n Bernstein blocks which form the flow transform. So for the most general degree n d-dimensional m-block Bernstein transform $B = B_m \circ \cdots \circ B_2 \circ B_1$, where $B_i = (B_n^{\boldsymbol{\alpha}_i^1} B_n^{\boldsymbol{\alpha}_i^2}, \ldots, B_d^{\boldsymbol{\alpha}_i^d})$, we have

$$B\colon X \to Z \implies q_X(x;\boldsymbol{\alpha}) \leq n^{md}\, p_Z(B_n^{\boldsymbol{\alpha}}(x)), \tag{5.4.9}$$

$$B\colon Z \to X \implies q_X(x;\boldsymbol{\alpha}) \geq \frac{1}{n^{md}}\, p_Z((B_n^{\boldsymbol{\alpha}})^{-1}(x)). \tag{5.4.10}$$

Since the dimensionality of the problem is fixed, the bounds can be relaxed or tightened by changing the Bernstein degree n, or the number m of composed Bernstein transforms that form the flow — in an ideal situation these could be chosen so that the upper bound on the flow density is just above the maximal target density, resulting in a choice of a 'smallest' possible flow for a particular target, speeding up the training and avoiding unnecessary overfitting. This of course makes the naive assumption that a flow that can match the maximum density is expressive enough to match the target shape over the entirety of its domain — this is not expected to generally be true, but the maximum density bound criterion could still be used as a guide to inform the choice of flow hyperparameters.

Another important caveat to note is that this is a theoretical bound which will hold in practice, but need not be attained at any point in the domain, that is there is no guarantee that there exists a point x in the domain with $q(x) = n^{md} p_Z(B(x))$. In fact these bounds in most cases will not be tight, and there will exist a lower upper bound and a greater lower bound — understanding of these tighter bounds would further inform the choice of the number of transform blocks and Bernstein bijector degree.

5.4.1. *Bernstein Transform bounds: experimental example*

Since the Bernstein polynomial derivative is bounded by a function of the polynomial degree and coefficients, the density of a flow obtained by transforming a bounded latent distribution via a Bernstein polynomial transform is bounded. This limits the expressiveness of a Bernstein polynomial flow, but could be used as a guide for choosing a lower bound for the number of Bernstein transform blocks and polynomial degrees needed to build a normalising flow capable of approximating a chosen target distribution, provided the maximum of the normalised density of the target is known. While we can not in general prove that a Bernstein flow with a certain degree and transform number will be expressive enough for a given problem, we can say with

certainty that a flow with these parameters less than a specific n and m will not be able to match the target at the point of maximum density.

Assume we want to train a Bernstein polynomial flow q_X to approximate a target distribution p_X, and the Bernstein polynomial flow has a uniform latent distribution $p_Z = 1$ on the domain $Z = [0, 1]^d$, and is defined in the 'normalising' direction, $X \to Z$. According to Equation 5.4.9, the density of the flow distribution q_X will be bounded from above according to

$$q_X(x) \leq n^{dm}, \tag{5.4.11}$$

where m is the number of composed transform blocks, each transform is of degree n, and d is the dimensionality of the problem.

Let, as an example, the target $p_X = \mathcal{N}(\boldsymbol{\mu}, \Sigma)$ be a bivariate Gaussian distribution with mean $\boldsymbol{\mu} = (0.5, 0.5)$ and a diagonal covariance matrix $\Sigma = (\sigma^2, \sigma^2)$, $\sigma = 0.01$, we can easily compute the maximum density of p_X as

$$p_X(\boldsymbol{\mu}) = \frac{1}{\sqrt{\det(\Sigma)\,(2\pi)^2}} \approx 1592. \tag{5.4.12}$$

Therefore if we were to choose the flow hyperparameters n and m such that

$$n^{2m} < p(\boldsymbol{\mu}) \approx 1592, \tag{5.4.13}$$

the Bernstein flow will **not** be able to match the target at the point of maximum density. The upper bound on the flow density results in a lower bound on the density deficit of q_X at $\boldsymbol{\mu}$,

$$\text{'Maximum density deficit'} \coloneqq \max(p_X(\boldsymbol{\mu}) - q(\boldsymbol{\mu}), 0) \geq 1592 - n^{2m}. \tag{5.4.14}$$

This theoretical maximum density deficit lower bound for this Gaussian target is plotted in Figure 20. We can see that there is a boundary in the $n - m$ space at which the density deficit is just equal to 0 — this gives the lower bound on the combination of parameters n and m resulting in flows that have the potential to meet the target at the point of maximum density (however it is not guaranteed they will).

It should be noted that any target distribution could be reparametrised from the physical variables of the problem in a way that would reduce the maximum density and therefore reduce the necessary number of transforms and Bernstein degree, and therefore the derived limit is parametrisation dependent. In the above example a simple scaling of the variables resulting in a larger σ would decrease the maximum density.

In order to put the behaviour of the theoretical bound on the flow density to a test, Bernstein bijector normalising flows with different choices of transform number m and

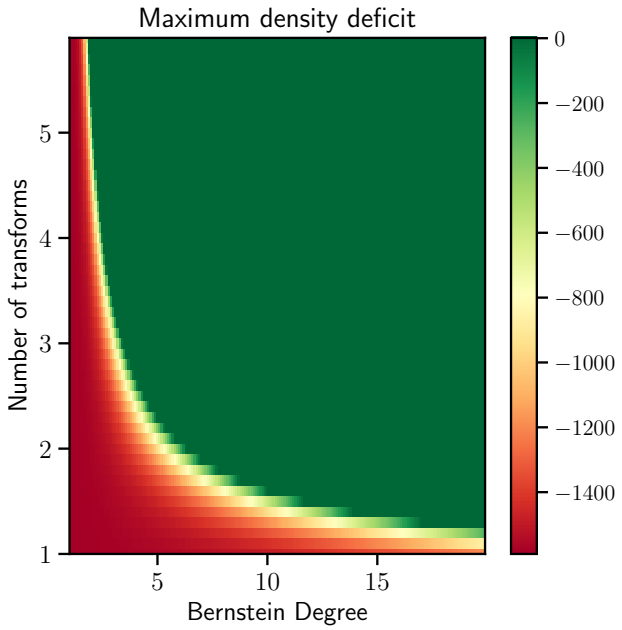


Figure 20: Theoretical maximum density deficit.

Theoretical maximum density deficit $\min(p(\mu) - n^{2m}, 0)$, where $p = \mathcal{N}(\mu, \Sigma)$, with $\boldsymbol{\mu} = (0.5, 0.5)$, $\Sigma = (\sigma^2, \sigma^2)$, $\sigma = 0.01$ is a bivariate normal distribution. The red/yellow region corresponds to flows whose density upper bound is below the maximum density of the target (positive density deficit), while the green region corresponds to cases where the upper bound on the flow density exceeds the maximum density of the target (density deficit of 0). Flow configurations falling into the red/yellow subspace, are not expressive enough to match the target. The flow configurations that have potential of being expressive enough for the problem fall into the green subspace.

Bernstein degree n have been trained to approximate the same bivariate Gaussian target. We expect flows in the red/yellow region of the $n - m$ space in Figure 20 to exhibit a density deficit, and therefore poor training, while a subset of the green region is expected to train well, achieving low KL divergence. Since the derived theoretical flow density upper bound is not the least upper bound, the expectation is that the boundary between the flows that do and do not exhibit density deficit will shift diagonally across the plot to higher values of m and n.

Training was performed using the flow as density estimators, that is via minimising the forward KL divergence $D_{KL}(p||q)$ over samples drawn from the target Gaussian. Training was performed with a fixed network size. The results of training are presented in Figure 21.

What we can observe is consistent with the prediction — flow configurations predicted to exhibit density deficit do exhibit one, confirming the validity of the flow density upper bound. The boundary between valid and invalid flows is however shifted

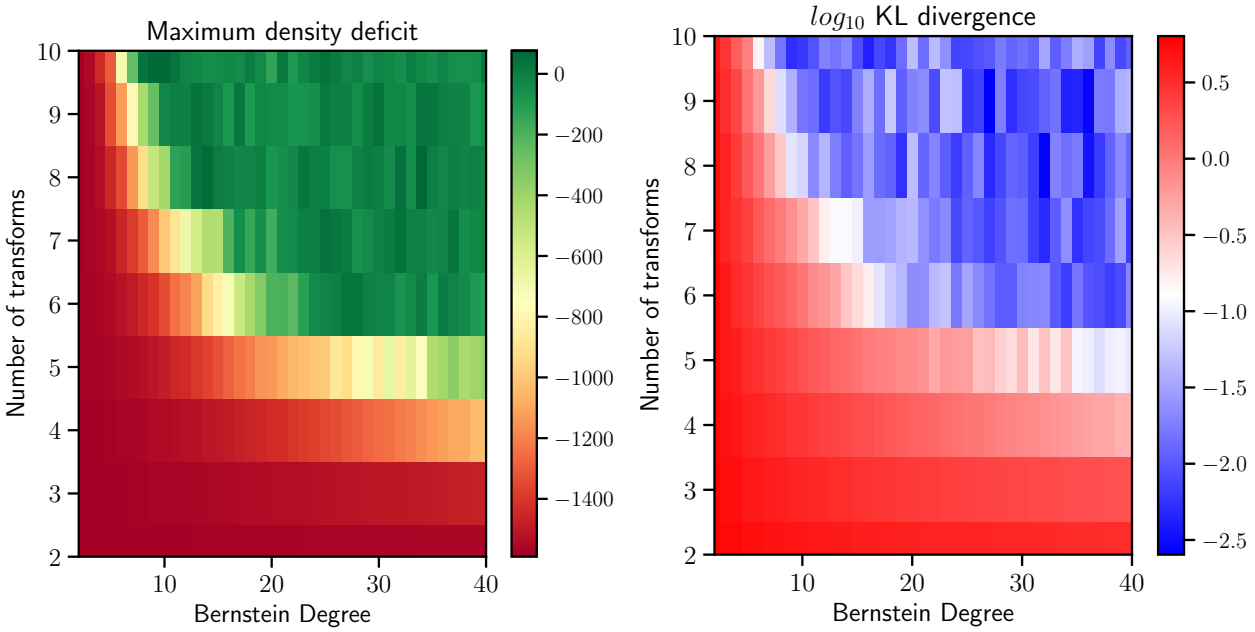


Figure 21: Measured density deficit.

Density difference $p(\mu) - q(\mu)$ between the target Gaussian p and the trained Bernstein polynomial flow q at the point of maximum target density (left), and the corresponding forward KL divergence $D_{KL}(p||q)$, for different choices of Bernstein degree n and transform number m. Note that the axes are different than those in the previous figure — in practice n and m can only take integer values and a coupling normalising flow needs at least 2 transforms in order to transform all of the variables.

significantly from the theoretical best-case scenario — we need higher than anticipated numbers of transforms and polynomial degrees to match the target, suggesting that the least upper bound on the flow density as a function of n and m is lower than predicted. What is reassuring is that the boundary between the two regions follows the predicted curve, therefore the general form of the functional relationship predicted is correct. What is also reassuring is that the KL divergence behaviour matches the density deficit — suggesting that flows with appropriate maximum density bounds are sufficiently expressive to approximate the target on the entirety of the domain. This might however not be a sufficient criterion in the case of more complex target distributions.

To further validate this result, training was performed with a larger network — this change had no influence on the maximum density deficit of the trained flows, further suggesting that the deficit can be explained with an upper bound, rather than being correlated with training performance.

5.4.2. *Bound dependence on the domain argument*

The theoretical bound on the flow density $q(x)$ which was derived in Section 5.4 is independent of x — the bound is a consequence of the partition of unity property of the Bernstein basis (Equation 5.4.1) which holds identically over the entirety of the Bernstein polynomial domain $[0, 1]$. This does not imply however that the true least upper bound on $q(x)$ is not a function of x, i.e. $q(x) \leq M(x)$, where $M(x)$ is a supremum of the density q at x.

We can notice that at the boundary $x = 0$,

$$\frac{\mathrm{d}B_n^{\boldsymbol{\alpha}}}{\mathrm{d}x}(0) = n \sum_{k=0}^{n-1} (\alpha_{k+1} - \alpha_k) b_{k,n-1}(0) = n(\alpha_1 - \alpha_0), \tag{5.4.15}$$

and also that in general we have

$$\sum_{i=0}^{n-1} (\alpha_{k+1} - \alpha_k) = 1, \quad \alpha_{k+1} - \alpha_k \geq \epsilon > 0, \tag{5.4.16}$$

since $0 = \alpha_0 < \alpha_1 < \cdots < \alpha_n = 1$ is an increasing sequence, so that the difference $\alpha_1 - \alpha_0$ can maximally be equal to

$$\alpha_1 - \alpha_0 = 1 - \sum_{i=1}^{n-1} (\alpha_{k+1} - \alpha_k) \leq 1 - (n-1)\epsilon, \tag{5.4.17}$$

where $\epsilon > 0$. In the implementation $\epsilon \sim 10^{-7}$ is a fixed positive constant which determines the minimal difference between consecutive α_k basis coefficients, necessary for ensuring an increasing sequence and therefore a bijective transform. Effectively then, up to a numerical correction we have

$$\max \left(\frac{\mathrm{d}B_n^{\boldsymbol{\alpha}}}{\mathrm{d}x}(0) \right) \approx n, \tag{5.4.18}$$

so the theoretical maximal density bound should be attained at $x = 0$ (and $x = 1$ by the same argument), up to a controlled numerical correction depending on the chosen ϵ. For a randomly chosen $x \in (0, 1)$ however there seems to be no simple way to derive an x-dependent bound.

To verify the hypothesis in practice, a test was performed where the Bernstein flow is trained, but this time not to minimise the divergence from the target distribution, but to minimise the squared difference between the flow density $q(x)$ at a chosen $x \in [0, 1]$ and a fixed density $p(x) = 1592$, chosen as the maximum density of the Gaussian from the previous test example. This should also be a better test of the

theoretical upper bound, as it directly optimises the transform parameters to reach a target density at a single point in the domain, without needing to optimise the full divergence which requires fitting distribution tails as well as the mode. The chosen points are $x = (0.5, 0.5)$, which is the mode location from the previous example, and the boundary point $x = (0, 0)$. By the above argument we expect the flow density bound $q(x) = n^{md}$, where m is the number of transforms and d is the dimensionality $d = 2$, to be attained at $x = (0, 0)$, up to a small error. The results of the test are presented in Figure 22.

We can see that in the case $x = (0, 0)$ the theoretical bound is, as predicted, almost attained, whereas in the case $x = (0.5, 0.5)$ there exists a lower upper bound on the density, confirming that the density bound is dependent on the domain argument. An interesting observation is that in the latter case the density deficit obtained by minimising the density difference directly (as opposed to minimising the KL between the flow and the target) is comparable to the density deficit obtained by minimising the KL divergence (Figure 21), suggesting that the optimal density does minimise the maximum density deficit, therefore validating the assumption that choosing the flow hyperparameters based on the maximum density deficit criterion could provide a flow that is just flexible enough to optimise the KL without overfitting.

The density bound domain dependence problem could be eliminated by introducing an extra trainable affine transform layer that performs shift on the domain of the form $x \mapsto (x + b) \bmod 1$— this would however require continuity of the density at the boundary $q(0) = q(1)$, effectively turning q to a density on a periodic domain (explored in Section 5.5), which may introduce its own limitations.

Another avenue is to further explore the relationship between the upper bound on $q(x)$ and x — this has however proven to be beyond the scope of this project. A possible direction that has been considered is looking at the upper bounds on the Bernstein approximation errors, as fitting a Bernstein flow with uniform density to a target is effectively finding a Bernstein polynomial approximation of the target density. The problem with that route of reasoning is that the optimal Bernstein flow may not coincide with the optimal Bernstein polynomial functional approximation of the density — this is because by our flow construction the flow density is necessarily a Bernstein polynomial with positive basis coefficients, limiting the choice of Bernstein transforms. This is a convenient sufficient condition for obtaining a valid density $q(x)$ which is everywhere non-negative, but is not a necessary condition. Since the optimal flow and the Bernstein polynomial approximation may not coincide, arguments about the latter seem not directly applicable in our case. This also poses a question of universality of the Bernstein flow, in the context of constrained α parameters. These

problems are left here unanswered, left for the reader who decides to carry on the research on this topic.

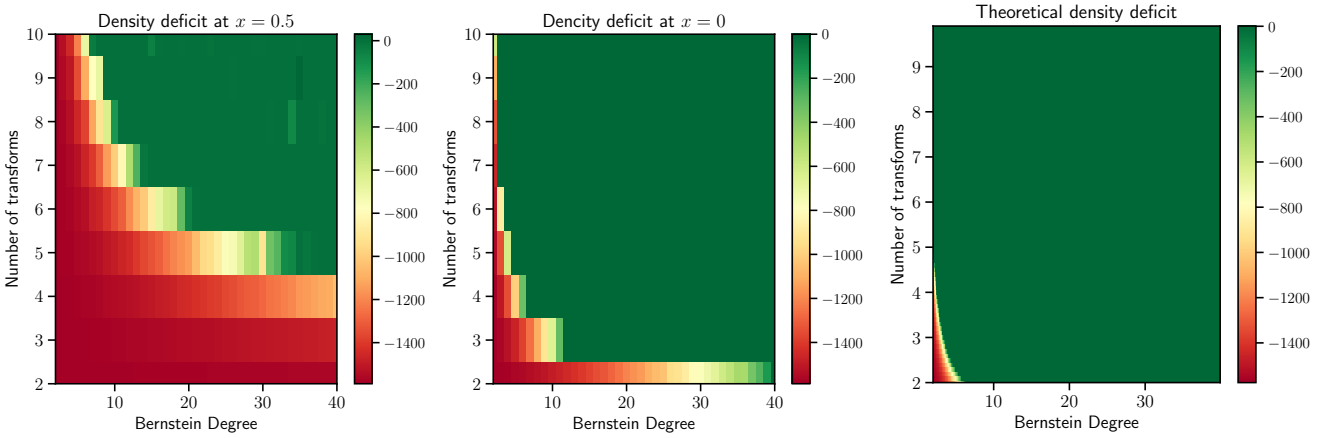


Figure 22: Density deficit and mode location.

Density difference $\max(p(x)) - q(x)$ between the target maximum density $\max(p(x)) = 1592$ and the density of the Bernstein polynomial flow q at x, trained to minimise the absolute value of this scalar density difference. In the left panel $x = (0.5, 0.5)$, in the middle panel $x = (0, 0)$, while the right panel shows the theoretical difference $\max(p(x)) - n^{md}$, between the target maximum density and the flow density theoretical upper bound.

5.5. CIRCULAR BERNSTEIN TRANSFORM

Analogically to the circular splines we can obtain a circular Bernstein transform, which will be a diffeomorphism of the circle $S^1 \cong [0, 1]/(0 \sim 1)$. This is particularly useful for expressing probability densities of periodic parameters, as a circular transform will impose the periodicity on the resulting probability distribution.

5.5.1. *Circular Bernstein Transform condition*

For the Bernstein transform to be a diffeomorphism of the circle S^1,

$$B_n^{\boldsymbol{\alpha}} \colon [0, 1]/(0 \sim 1) \to [0, 1]/(0 \sim 1) \cong S^1, \tag{5.5.1}$$

we need the following conditions to be satisfied

1. $B_n^{\boldsymbol{\alpha}} \equiv \sum_{k=0}^{n} \alpha_k b_{k,n}$ to be a smooth bijection on $[0, 1]$,
2. $B_n^{\boldsymbol{\alpha}}(0) \sim B_n^{\boldsymbol{\alpha}}(1)$,
3. $\frac{\mathrm{d}B_n^{\boldsymbol{\alpha}}}{\mathrm{d}x}(0) = \frac{\mathrm{d}B_n^{\boldsymbol{\alpha}}}{\mathrm{d}x}(1)$.

Condition 1 as proven earlier is satisfied provided that $\alpha_k < \alpha_{k+1} \forall\, k$, that is the Bernstein basis coefficients form a strictly increasing sequence.

Condition 2, regarding endpoint equivalence, is easily achieved by setting $B_n^{\boldsymbol{\alpha}}(0) = \alpha_0 = 0$ and $B_n^{\boldsymbol{\alpha}}(1) = \alpha_n = 1$.

For condition 3, we first notice that

$$b_{k,n}(0) = \binom{n}{k} 0^k (1-0)^{n-k} = \begin{cases} 0, \ \forall\, k > 0 \\ 1, \ k = 0, \end{cases} \tag{5.5.2}$$

so that

$$\frac{\mathrm{d}B_n^{\boldsymbol{\alpha}}}{\mathrm{d}x}(0) = n \sum_{k=0}^{n-1} (\alpha_{k+1} - \alpha_k) b_{k,n-1}(0) = n(\alpha_1 - \alpha_0). \tag{5.5.3}$$

Similarily, at the other endpoint,

$$b_{k,n}(1) = \binom{n}{k} 1^k (1-1)^{n-k} = \begin{cases} 0, \ 0 \leq k < n \\ 1, \ k = n, \end{cases} \tag{5.5.4}$$

and hence

$$\frac{\mathrm{d}B_n^{\boldsymbol{\alpha}}}{\mathrm{d}x}(1) = n \sum_{k=0}^{n-1} (\alpha_{k+1} - \alpha_k) b_{k,n-1}(1) = n(\alpha_n - \alpha_{n-1}). \tag{5.5.5}$$

Since we have already set $\alpha_0 = 0$ and $\alpha_1 = 1$, the equality of the endpoint derivatives requires

$$\frac{\mathrm{d}B_n^{\boldsymbol{\alpha}}}{\mathrm{d}x}(0) = \frac{\mathrm{d}B_n^{\boldsymbol{\alpha}}}{\mathrm{d}x}(1) \implies \alpha_1 = 1 - \alpha_{n-1}, \tag{5.5.6}$$

which can be easily achieved, with the cost of losing one degree of freedom.

The circular Bernstein polynomial condition (3) is therefore $\alpha_1 = 1 - \alpha_{n-1}$.

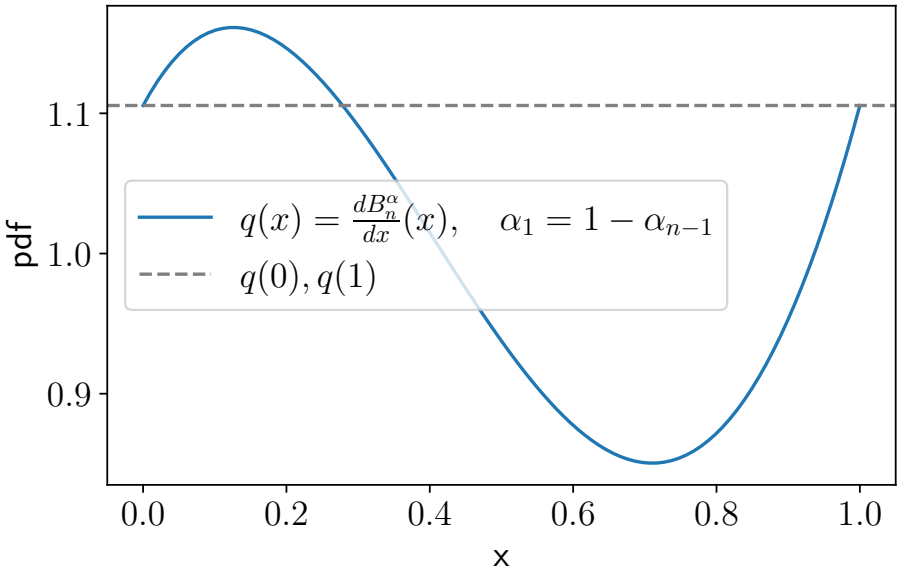
5.5.2. *Bernstein flow on periodic domains*

In order to construct a normalising flow density on a periodic domain it is sufficient to use a bijector which is a diffeomorphism of a circle, and a periodic base density on the bounded domain $[0, 1]$, that is with $p_Z(0) = p_Z(1)$. A flow constructed this way is guaranteed to result in a density q preserving continuity at the domain endpoint, $q(0) = q(1)$. By choosing, for example, the latent distribution to be uniform on $[0, 1]$, $p_Z = 1$, and a 1-dimensional Bernstein bijector with $\alpha_1 = 1 - \alpha_{n-1}$, we obtain a flow density on the circle $[0, 1]/(0 \sim 1) \cong S^1$ with

$$q(x) = \frac{\mathrm{d}B_n^{\boldsymbol{\alpha}}}{\mathrm{d}x}(x), \quad q(0) = q(1). \tag{5.5.7}$$

Example of a periodic density obtained via applying a circular Bernstein transform to a uniform latent space is shown in Figure 23. The circular condition removes just one degree of freedom — a n-degree circular Bernstein with fixed endpoints has n-3 unconstrained α_i coefficients.

We can similarly obtain a flow on the n-torus, $T^N \cong \underbrace{S^1 \times S^1 \cdots \times S^1}_{\text{N times}}$, product space of N circles, by using a d-dimensional Bernstein transform such that the circular condition $\alpha_1 = 1 - \alpha_{n-1}$ is satisfied for each elementwise transform — this can be achieved by using either of the coupling or autoregressive architectures. If only some of the parameters are periodic, that is the domain is of the form $S^1 \times \cdots \times S^1 \times X$, then one can build an autoregressive flow in which the periodic subspaces are selectively transformed by circular Bernstein transforms — the coupling architecture permutes the dimensions and splits the parameters in half, so it would not be convenient if different dimensions are transformed by different types of bijectors.



$$\alpha = (0.00, 0.22, 0.49, 0.65, 0.78, 1.00), \quad n = 5$$

$$\alpha_1 = 1 - \alpha_{n-1} \implies \frac{dB_n^\alpha}{dx}(0) = \frac{dB_n^\alpha}{dx}(1) \iff q(0) = q(1)$$

Figure 23: Circular Bernstein transform.

Example of a density obtained by transforming a uniform latent distribution using a circular Bernstein transform. The density is continuous on the periodic domain $[0, 1]/(0 \sim 1) \cong S^1$.

Part VI.
Conclusion

This work presented the results of an initial exploration into the possibility of the use of variational inference and normalising flows for the purpose of gravitational wave parameter estimation.

Variational inference posterior estimation with spline-based normalising flow transforms has been tested on posterior distributions of BBH signals injected into the ground-based interferometer network. The algorithm was able to successfully recover the target posterior only on constrained lower-dimensional subspaces of the parameter space. The main observed problem of this approach is the intrinsic tendency of the algorithm to converge onto unimodal normalising flow configurations, approximating either one of many posterior modes, or a secondary mode away from the high likelihood region of the parameter space. This tendency is understood to be a direct consequence of the reverse KL divergence training objective, which rewards mode collapse over poor mode approximation, incentivising the network to force the flow density towards zero in the high likelihood regions of the posterior which are not readily matched by the flow distribution.

Different approaches have been investigated to circumvent the mode collapse, including different flow architectures (coupling and MAF), domain reparametrisation, temperature annealing of the posterior, training the flow on a mixture distribution of the posterior with a gradually annealed uniform distribution, as well as nested variational inference algorithm formulation, where a sequence of transforms is trained simultaneously to interpolate between a series of annealed distributions. The main improvements on mode collapse prevention have been seen from the introduction of temperature annealing, allowing successful training on a ι-bimodal 6-dimensional subspace of a BBH posterior. This initially promising result however proved to not generalise well to different choices of training hyperparameters, pointing to the conclusion that the reverse KL-trained variational inference algorithm lacks the necessary robustness for reliable GW parameter estimation, where posterior mode collapse is not acceptable.

While unsuccessful on the full BBH problem in the extent to which it was tested within the scope of this work, the variational inference algorithm with a normalising flow based variational distribution remains a viable potential alternative to traditional methods of stochastic posterior sampling, which could provide a closed form posterior approximation benefiting from fast GPU training utilising parallel likelihood evaluations, low-cost post-training sampling, evidence lower bound estimation,

as well as the prospect of incorporating the non-trivial parameter space topology into the flow architecture, ensuring the posterior approximation is performed on the natural domain of the gravitational wave parameters. Concluding from the results of this work, research on producing such an algorithm should focus on exploring alternative training objectives, specifically objectives which do not promote mode collapse, such as is the case of the reverse KL divergence minimisation.

The latter chapter of this work has explored the possibility of using Bernstein Polynomial transforms as an alternative to the rational quadratic splines in the construction of normalising flows, both as an alternative architecture for the variational inference application, as well as for use in other normalising flow based algorithms utilised within the field. When trained as density estimators on samples from simple targets, Bernstein polynomial normalising flows converged to the target distribution in fewer gradient descent steps than their neural spline based equivalents, in particular when using a uniform latent distribution. In the variational inference BBH scenario however, Bernstein polynomials show comparable performance to the spline-based flows, without improvements to mode retention.

In the course of the investigation of Bernstein Polynomial flows, it has been observed that probability densities expressed as Bernstein polynomial normalising flows are bounded, with bounds related to the degree of the transform. If the maximal density of the target distribution is known, the upper bound on the Bernstein flow density leads to a lower bound on the Bernstein polynomial transform degree necessary for constructing a flow capable of matching the target, therefore informing the choice of hyperparameters. Since the density bound is not tight, it is not however possible to ensure that choosing the lower bound polynomial degree will result in a flow of sufficient complexity. There is scope for further research that could determine an optimal Bernstein degree choice for a GW posterior with a known maximum posterior density, narrowing the hyperparameter choice prior to training.

The investigation of the Bernstein flow bounds has led to an observation that similar to rational quadratic splines, Bernstein transforms admit a circular form, that is they can have their coefficients restricted to extend from a diffeomorphism on the interval $[0, 1]$ to a diffeomorphism on the circle S^1, by ensuring continuity and smoothness at the interval endpoints. Bernstein polynomial flows can therefore be readily used to approximate distributions on periodic domains, such as the distributions on the angular parameters of a GW signal.

APPENDIX

A. RIEMANNIAN CONVEX POTENTIAL MAP FLOWS (RCPMS)

Cohen et al. [106] propose a way of building flows on Riemannian manifolds using results from optimal transport theory.

Riemannian optimal transport theory is concerned with finding optimal ways of pushing forward probability measures on Riemannian manifolds. Given a Riemannian manifold $\mathcal{M}$ and two probability measures $\mu, \nu \in P(\mathcal{M})$ on $\mathcal{M}$, ν is a *pushforward* of μ by a *transport map* $f\colon \mathcal{M} \to \mathcal{M}$ denoted $\nu = f_{\#}\mu$ if $\nu = \mu \circ f^{-1}$. Given a transport map $f\colon \mathcal{M} \to \mathcal{M}$, we can associate with it a *transport cost* of the pushforward of the base measure μ

$$C(t) \coloneqq \int_{\mathcal{M}} \tfrac{1}{2} d(x, f(x))^2 d\mu(x) \tag{A1}$$

where $d : \mathcal{M} \times \mathcal{M} \to \mathbb{R}$ is the intrinsic Riemannian distance on $\mathcal{M}$. Intuitively, the further the points are transported by f the higher the cost, it also costs more to move sets with higher probability mass associated to them, whereas it costs nothing to transport sets of measure 0. An *optimal transport map* between two measures is a transport map which minimises the transport cost. Optimal transport cost is often used as an alternative to KL-divergence when defining a distance between two probability distributions.

Given two probability measures, there exists a unique optimal transport map between them, and moreover it takes the form

$$f(x) = \exp_x(\nabla\phi(x)), \tag{A2}$$

where $\phi\colon \mathcal{M} \to \mathbb{R}$ is a *c-convex* scalar potential on $\mathcal{M}$, and $\exp_x\colon T_x\mathcal{M} \to \mathcal{M}$ is the exponential map of the manifold at the point $x \in \mathcal{M}$. This map is indeed a map from $\mathcal{M}$ to $\mathcal{M}$, and given a point $x \in \mathcal{M}$ its image is obtained by first evaluating the scalar potential $\phi(x)$, taking its Riemannian gradient $\nabla\phi(x) \in T_x\mathcal{M}$, which is a vector in the tangent space to $\mathcal{M}$ at x, and lastly evaluating the exponential map $\exp_x(\nabla\phi(x))$, which is the endpoint $\gamma(1) \in \mathcal{M}$ of the unique geodesic $\gamma\colon [0,1] \to \mathcal{M}$ satisfying $\gamma(0) = x$ and $\gamma'(0) = \nabla\phi(x)$.

The idea is now the following — if we can find a way to parametrise a generic c-convex potential ϕ with trainable parameters we could build a flow whose building blocks would have the form of optimal transport maps, as given in Eq. A2.

If a potential ϕ is *c-convex* then $-\phi(x)$ is *c-concave* and we will define the latter. A potential ϕ is *c-concave* if there exists $\psi\colon \mathcal{M} \to \mathbb{R} \cup \{\pm\infty\}$ such that

$$\phi(x) = \inf_{y \in \mathcal{M}} \{\tfrac{1}{2} d(x, y)^2 + \psi(y)\}. \tag{A3}$$

A simple example of a c-concave function is obtained by taking a discrete set of manifold points $x_i \in \mathcal{M}$ and associated values $\alpha_i \in \mathbb{R}$, i.e. a set of pairs $(x_i, \alpha_i)_{i \in I} \in \mathcal{M} \times \mathbb{R}$ for some finite index set I, and defining

$$\psi(x) \coloneqq \begin{cases} \alpha_i & \text{if } x = x_i \\ \infty & \text{otherwise.} \end{cases} \tag{A4}$$

Then by construction the potential ϕ defined by

$$\phi(x) \coloneqq \min_{i \in I} \{ \tfrac{1}{2} d(x, x_i)^2 + \alpha_i \} \tag{A5}$$

is c-concave by construction. Cohen et al. refer to a potential defined in the above way as a *discrete c-concave potential*. A discrete c-concave potential will satisfy $\phi(x) = \alpha_i$ whenever $x = x_i$ and otherwise $\phi(x)$ will evaluate to α_i plus half the distance squared from x to the closest x_i. This potential is continuous, however it is not globally smooth – it has $|I|$ smooth neighbourhoods (each around one of x_i) which together cover all of $\mathcal{M}$.

The crucial observation that Cohen et al. make is that the discrete c-concave functions are dense in the set of all c-concave functions and hence given a large enough parameter set I, any c-concave function can be approximated by a discrete c-concave function arbitrarily well. This motivates the use of discrete c-concave potentials as universal building blocks of optimal transport map flows.

The proposed flow, which the authors name RCPM (Rimeannian Convex Potential Map) consists of $N \in \mathbb{N}$ blocks, each of which is of the form

$$f_n \colon \mathcal{M} \to \mathcal{M}, \quad f_n(x) = \exp_x(-\nabla \phi_n(x)) \tag{A6}$$

$$\phi_n \colon \mathcal{M} \to \mathbb{R}, \quad \phi_n(x) = \min_{i \in I} \{ \tfrac{1}{2} d(x, x_i)^2 + \alpha_i \}, \tag{A7}$$

so that for each $n \in \{1, 2, \ldots, N\}$ the block f_n is an optimal transport map. The overall flow $f \colon \mathcal{M} \to \mathcal{M}$ is a composition of all the blocks, i.e.

$$f = f_1 \circ f_2 \circ \cdots \circ f_N. \tag{A8}$$

The trainable parameters of an RCPM flow consist of the points $x_i \in \mathcal{M}$ and the potential values $\alpha_i \in \mathbb{R}$ for each of the blocks. This adds up to a total of $|I| \times N$ trainable parameters for an N block flow. Expressiveness of the flow can be further increased by composing multiple c-convex potentials (noting that this is restricted to manifolds where c-convexity is preserved under function composition). Like any other flow the RCPM can be trained to approximate a target posterior by minimizing a KL objective.

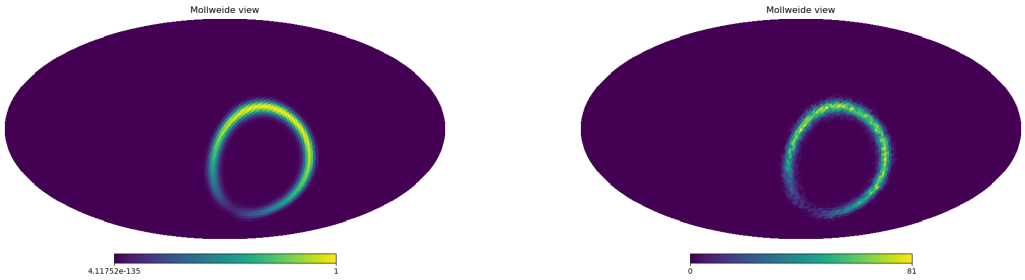
The main issue with RCPM flows as defined above is their discontinuity — this is cause by the lack of smoothness present in the discrete c-convex potential maps, which in turn results in discontinuous gradients. Authors of the paper address this issue via exchanging the min operation in the definition of the potential maps for a softmin. By doing so, the discrete c-convex potentials are smoothed, which restores continuity of the flow. The authors cannot prove that the softmin operation preserves c-convexity, yet if it did, then, according to the authors, the resulting RCPM flow would be a diffeomorphism. They claim that, in the case of a 2-sphere, smoothing the potentials with a softmin does indeed lead to a diffeomorphic flow. They obtain this result numerically.

A.1. *Riemann Convex Potential Map flow for the sine-Gaussian sky map*

Motivated by the successful application of the VI algorithm to the sine-Gaussian test case, the focus moved to investigating the existing methods of constructing normalising flows on parameter spaces with non-Euclidean geometry. Flows like this could be implemented to enhance inference over the dimensions in the parameter space exhibiting non-trivial topology, such as the sky location parameters (spherical geometry) or orientation of the source orbit (rotation group SO(3)).

In order to impose the parameter space geometry on the flow, one needs to define the invertible flow transforms to be diffeomorphisms (smooth bijections) of the underlying space. A particularly interesting solution is provided by the Riemann Convex Potential Map (RCPM) flows [106], where a diffeomorphism of the underlying manifold is constructed by defining a trainable scalar potential, taking its Riemannian gradient and locally evaluating the exponential map — that is flowing the manifold points along geodesics according to directions and distances specified by the gradient vector field of the trainable potential.

An implementation of the RCPM flows was tested on a 2-dimensional spherical test case of the sky location distribution (right ascension, declination) of a sine-Gaussian waveform injected into a two detector network. In this case the sky location posterior is a narrow ring corresponding to sky locations resulting in constant time delay between the arrival time of the signal at the two detectors. This distribution is notoriously difficult to learn by standard stochastic sampling methods due to the complicated correlation structure present, partly resulting from the non-trivial geometry of the parameter space. The method exhibited fast training and convergence, leading to a very good result (Figure 24). What has to be noted however is that the use of RCPM flows is limited, since the flow architecture does not allow for explicit inverse transform evaluation and hence allows only for sampling, but not for the flow probability density evaluations at arbitrary points of the parameter space.



(Target skymap) (Approximated Skymap)

Figure 24: RCPM flow on a sine-Gaussian skymap.

Target skymap of a sine-Gaussian signal (left) and an approximate skymap obtained by the RCPM flow.

B. CONTINUOUS NORMALISING FLOWS

A standard non-continuous normalising flow $f\colon X \to X$ is a composition of multiple bijective transformations, i.e.

$$f = f_N \circ f_{N-1} \circ \ldots \circ f_1, \quad f_t\colon X \to X \tag{B1}$$

Given any point $z_0 \in X$ we can interpret the set of points $\{z_0, z_t \coloneqq f_t(z_{t-1})\}_{t\in\{1,\ldots,N\}}$ as a discrete trajectory that z_0 takes through the space X under the flow f as time passes from $t = 0$ to $t = N$. The idea behind *Continuous Normalising Flows* [107] is to extend this to the continuous time limit and define the flow via specifying its infinitesimal dynamics.

We assume that time is running continuously from 0 to 1. Given an initial point $z_0 \in X$ we can consider its trajectory as a continuous path in X starting at z_0,

$$z\colon [0,1] \to X, \quad z(0) = z_0, \tag{B2}$$

so that $z(t)$ can be interpreted as a position of the particle z at time $t \in I$. Now given a time-dependent vector field v_θ on X, i.e. a function of the form

$$v_\theta\colon [0,1] \times X \to TX, \tag{B3}$$

where TX is the tangent bundle of X, we can interpret this vector field as the *velocity field* of the particle z by setting

$$\frac{\mathrm{d}z(t)}{\mathrm{d}t} \coloneqq v_\theta(t, z(t)) \quad \in T_{z(t)}X. \tag{B4}$$

From here we can define a time-continuous family of flows $f\colon [0,1] \times X \to X$ as the solution to the ODE given by Equation B4. Given a point $z_0 \in X$ and a time

$t \in [0, 1]$, the value of the flow will be the position of the particle z at time t, subject to the initial constraint $z(0) = z_0$,

$$f(t, z_0) \coloneqq z(t) = z_0 + \int_{t=0}^{t} v_\theta(t, z(t))dt. \tag{B5}$$

In particular, given an initial position $z_0 \in X$, we solve for the final position z_1 via evaluating

$$z_1 \coloneqq z(1) = z_0 + \int_{t=0}^{t=1} v_\theta(t, z(t))dt. \tag{B6}$$

Similarly, given a sample $z_1 \in X$ we can reverse the flow to obtain the corresponding initial position z_0 by integrating backwards in time,

$$z_0 = z_1 + \int_{t=1}^{t=0} v_\theta(t, z(t))dt = z_1 - \int_{t=0}^{t=1} v_\theta(t, z(t))dt. \tag{B7}$$

CNFs are therefore always easily invertible and evaluating the inverse and the forwards flow are associated with the same computational cost – the cost of integrating the ODE between times 0 and 1.

The computational advantage of CNFs is the density update computation. If $z(t)$ is distributed according to $p(z(t))$ and if v_θ meets certain conditions (i.e. is uniformly Lipschitz continuous in z and continuous in time t, then it holds that

$$\frac{\mathrm{d} \log p(z(t))}{\mathrm{d}t} = -\mathrm{Tr}\left[J_{v_\theta(t,\cdot)}(z(t))\right], \tag{B8}$$

where Tr denotes the trace operator (for proof see Appendix in [107]). Then if $z_0 \sim p(z_0)$, we can find the distribution of z_1 by integrating B8, i.e.

$$\log p(z_1) = \log p(z_0) - \int_{t=0}^{t=1} \mathrm{Tr}\left[J_{v_\theta(t,\cdot)}(z(t))\right] \mathrm{d}t. \tag{B9}$$

The trace operation is of lower computational complexity than the determinant operation required for computing density updates in discrete normalising flows.

In practice, during training, given an initial sample $z_0 \in X$, one can simultaneously integrate for a final sample z_1 and its log probability via solving the system of ODEs

$$\begin{bmatrix} z_1 \\ \log p(z_1) \end{bmatrix} = \begin{bmatrix} z_0 \\ \log p(z_0) \end{bmatrix} + \int_{t=0}^{t=1} \begin{bmatrix} v_\theta(t, z(t)) \\ -\mathrm{Tr}\left[J_{v_\theta(t,\cdot)}(z(t))\right] \end{bmatrix} \mathrm{d}t. \tag{B10}$$

The velocity field v_θ is defined via a neural network parametrised by parameters θ, the network architecture ensures that the density update property is satisfied and that the resulting flow family is indeed a family of diffeomorphisms of X. The ODE integration is then performed using a numerical solver.

B.1. *Riemannian Continuous Normalising Flows (RCNFs)*

Riemannian Continuous Normalising Flows (RCNFs) [108] is a type of a Continuous Normalising Flow CNF) generalised to the Riemannian manifold setting; in this flow model the flow is defined as a solution to an ODE describing the time evolution of manifold-valued particles.

Let $\mathcal{M}$ be a Riemannian manifold corresponding to the domain of the flow. What the authors describe as a *manifold-valued particle* is a function

$$z\colon \mathbb{R} \to \mathcal{M}, \tag{B11}$$

so that $z(t)$ can be interpreted as a position of the particle z at time $t \in \mathbb{R}$. Now given a time-variable vector field f_θ on the manifold $\mathcal{M}$, i.e. a function of the form

$$f_\theta\colon \mathcal{M} \times \mathbb{R} \to T\mathcal{M}, \tag{B12}$$

where $T\mathcal{M}$ is the tangent bundle on $\mathcal{M}$, we can interpret this vector field as the *velocity field* of the particle z by setting

$$\frac{\mathrm{d}z(t)}{\mathrm{d}t} \coloneqq f_\theta(z(t), t) \quad \in T_{z(t)}\mathcal{M}. \tag{B13}$$

From here a continuous flow $\phi\colon \mathcal{M} \times \mathbb{R} \to \mathcal{M}$ can be defined as the solution to the ODE given by Equation B13. Given a point $z_0 \in \mathcal{M}$ and a time $t \in \mathbb{R}$, the value of the flow will be the position of the particle z at time t, subject to the initial constraint $z(0) = z_0$,

$$\phi(z_0, t) \coloneqq z(t). \tag{B14}$$

As long as the time-dependent velocity field f_θ is C^1 (once differentiable with a continuous derivative) and bounded, the resulting vector flow will be bijective and a C^1-diffeomorphism.

The velocity field f_θ is defined via a neural network which takes as input manifold-valued particles (positions and times), and outputs their velocities. Differentiability and boundedness of the output is guaranteed by the design of the network layers. From this neural network parametrized velocity-field, the flow is obtained by numerically integrating the ODE in Equation B13 to obtain the position of the particles as a function of time and evolving it in time for a desired time t. The flow can be trained by the standard methods such as negative log-likelihood or reverse KL minimisation w.r.t. the NN parameters.

Information about the manifold geometry is incorporated into the NN design and Jacobian estimation.

REFERENCES

[1]B. P. Abbott et al. Observation of Gravitational Waves from a Binary Black Hole Merger. *Phys. Rev. Lett.*, 116:061102, February 2016.

[2]B. P. Abbott et al. GWTC-1: A Gravitational-Wave Transient Catalog of Compact Binary Mergers Observed by LIGO and Virgo during the First and Second Observing Runs. *Physical Review X*, 9(3), September 2019.

[3]B. P. Abbott et al. GWTC-2: Compact Binary Coalescences Observed by LIGO and Virgo during the First Half of the Third Observing Run. *Physical Review X*, 11(2), June 2021.

[4]B. P. Abbott et al. GWTC-3: Compact Binary Coalescences Observed by LIGO and Virgo during the Second Part of the Third Observing Run. *Physical Review X*, 13(4), June 2021.

[5]The LIGO Scientific Collaboration and the Virgo Collaboration and the KAGRA Collaboration. GWTC-4.0: Updating the Gravitational-Wave Transient Catalog with Observations from the First Part of the Fourth LIGO-Virgo-KAGRA Observing Run. *arXiv e-prints*, August 2025.

[6]B. P. Abbott et al. Binary Black Hole Population Properties Inferred from the First and Second Observing Runs of Advanced LIGO and Advanced Virgo. *The Astrophysical Journal Letters*, 882(2):L24, September 2019.

[7]R. Abbott et al. Population Properties of Compact Objects from the Second LIGO–Virgo Gravitational-Wave Transient Catalog. *The Astrophysical Journal Letters*, 913(1):L7, May 2021.

[8]The LIGO Scientific Collaboration and the Virgo Collaboration and the KAGRA Collaboration. The population of merging compact binaries inferred using gravitational waves through GWTC-3, 2022.

[9]The LIGO Scientific Collaboration and the Virgo Collaboration and the KAGRA Collaboration. GWTC-4.0: Population Properties of Merging Compact Binaries, 2025.

[10]The LIGO Scientific Collaboration and the Virgo Collaboration and the KAGRA Collaboration. Tests of general relativity with binary black holes from the second LIGO-Virgo gravitational-wave transient catalog. *Physical Review D*, 103(12), June 2021.

[11]The LIGO Scientific Collaboration and the Virgo Collaboration and the KAGRA Collaboration. Tests of General Relativity with GWTC-3, 2021.

[12]The LIGO Scientific Collaboration and the Virgo Collaboration and the KAGRA Collaboration. A Gravitational-wave Measurement of the Hubble Constant Following the Second Observing Run of Advanced LIGO and Virgo. *The Astrophysical Journal*, 909(2):218, March 2021.

[13]The LIGO Scientific Collaboration and the Virgo Collaboration and the KAGRA Collaboration. GWTC-4.0: Constraints on the Cosmic Expansion Rate and Modified Gravitational-wave Propagation, 2025.

[14]M. Punturo et al. The Einstein Telescope: A third-generation gravitational wave observatory. *Classical and Quantum Gravity*, 27, October 2010.

[15]D. Reitze et al. Cosmic Explorer: The U.S. Contribution to Gravitational-Wave Astronomy beyond LIGO, 2019.

[16]P. Amaro-Seoane et al. Laser Interferometer Space Antenna, 2017.

[17]K. Cannon. Toward early-warning detection of gravitational waves from compact binary coalescence. *The Astrophysical Journal*, 748(2):136, March 2012.

[18]J. Veitch and A. Vecchio. Bayesian coherent analysis of in-spiral gravitational wave signals with a detector network. *Phys. Rev. D*, 81:062003, March 2010.
[19]J. Veitch et al. Parameter estimation for compact binaries with ground-based gravitational-wave observations using the LALInference software library. *Phys. Rev. D*, 91:042003, February 2015.
[20]LIGO Scientific Collaboration and Virgo Collaboration and KAGRA Collaboration. LIGO Algorithm Library - LALSuite. Free software (GPL), 2018.
[21]J. Skilling. Nested Sampling. *AIP Conference Proceedings*, 735(1):395–405, November 2004.
[22]D. J. Rezende and S. Mohamed. Variational Inference with Normalizing Flows, 2016.
[23]I. Kobyzev et al. Normalizing Flows: An Introduction and Review of Current Methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(11):3964–3979, November 2021.
[24]G. Papamakarios et al. Normalizing Flows for Probabilistic Modeling and Inference, 2021.
[25]A. Einstein. *Prussian Academy Proceedings*, 1:688–696, 1916.
[26]B. P. Abbott et al. GW170817: Observation of Gravitational Waves from a Binary Neutron Star Inspiral. *Phys. Rev. Lett.*, 119:161101, October 2017.
[27]The LIGO Scientific Collaboration, the Virgo Collaboration, and the KAGRA Collaboration. Observation of Gravitational Waves from the Coalescence of a 2.5–4.5 M Compact Object and a Neutron Star. *The Astrophysical Journal Letters*, 970(2):L34, July 2024.
[28]R. Abbott et al. All-sky search for short gravitational-wave bursts in the third Advanced LIGO and Advanced Virgo run. *Phys. Rev. D*, 104:122004, December 2021.
[29]R. Abbott et al. All-sky search for long-duration gravitational-wave bursts in the third Advanced LIGO and Advanced Virgo run. *Physical Review D*, 104(10), November 2021.
[30]LIGO Scientific Collaboration and Virgo Collaboration and KAGRA Collaboration. All-sky search for short gravitational-wave bursts in the first part of the fourth LIGO-Virgo-KAGRA observing run, 2025.
[31]H. Janka. Explosion Mechanisms of Core-Collapse Supernovae. *Annual Review of Nuclear and Particle Science*, 62(1):407–451, November 2012.
[32]C. Kouvaris. Gravitational waves from magnetars, 2025.
[33]R. Abbott et al. All-sky search for continuous gravitational waves from isolated neutron stars using Advanced LIGO and Advanced Virgo O3 data. *Phys. Rev. D*, 106:102008, November 2022.
[34]N. Christensen. Stochastic gravitational wave backgrounds. *Reports on Progress in Physics*, 82(1):016903, November 2018.
[35]The LIGO Scientific Collaboration. Advanced LIGO. *Classical and Quantum Gravity*, 32(7):074001, March 2015.
[36]F. Acernese et al. Advanced Virgo: a second-generation interferometric gravitational wave detector. *Classical and Quantum Gravity*, 32(2):024001, December 2014.
[37]T. Akutsu et al. Overview of KAGRA: Detector design and construction history, 2020.
[38]B. P. Abbott et al. Gravitational Waves and Gamma-Rays from a Binary Neutron Star Merger: GW170817 and GRB 170817A. *The Astrophysical Journal Letters*, 848(2):L13, October 2017.
[39]R. Abbott et al. Constraints on the Cosmic Expansion History from GWTC–3. *The Astrophysical Journal*, 949(2):76, June 2023.

[40]N. Aghanim et al. Planck2018 results: Vi. cosmological parameters. *Astronomy amp; Astrophysics*, 641:A6, September 2020.

[41]L. Galbany et al. An updated measurement of the Hubble constant from near-infrared observations of Type Ia supernovae. *Astronomy amp; Astrophysics*, 679:A95, November 2023.

[42]B Willke et al. The GEO 600 gravitational wave detector. *Classical and Quantum Gravity*, 19(7):1377, March 2002.

[43]J. Vinet et al. Optimization of long-baseline optical interferometers for gravitational-wave detection. *Phys. Rev. D*, 38:433–447, July 1988.

[44]The LIGO Scientific Collaboration and the Virgo Collaboration and the KAGRA Collaboration. GWTC-4.0: An Introduction to Version 4.0 of the Gravitational-Wave Transient Catalog, 2025.

[45]T. Souradeep et al. LIGO-India – a unique adventure in Indian science. *Current Science*, 113:672, 2017.

[46]J. Luo et al. TianQin: a space-borne gravitational wave detector. *Classical and Quantum Gravity*, 33(3):035010, January 2016.

[47]W. Hu and Y. Wu. The Taiji Program in Space for gravitational wave physics and the nature of gravity. *National Science Review*, 4(5):685–686, October 2017.

[48]S. Kawamura et al. Space gravitational-wave antennas DECIGO and B-DECIGO. *International Journal of Modern Physics D*, 28(12):1845001, January 2019.

[49]S. Kawamura et al. Space gravitational wave antenna DECIGO and B-DECIGO. *The Sixteenth Marcel Grossmann Meeting*, pages 3254–3260, 2023.

[50]P.W. Joris et al. Status report on global pulsar-timing-array efforts to detect gravitational waves. *Results in Physics*, 61:107719, 2024.

[51]G. Agazie et al. The NANOGrav 15-year Data Set: Evidence for a Gravitational-Wave Background. June 2023.

[52]H. Xu et al. Searching for the Nano-Hertz Stochastic Gravitational Wave Background with the Chinese Pulsar Timing Array Data Release I. *Research in Astronomy and Astrophysics*, 23(7):075024, June 2023.

[53]EPTA Collaboration and InPTA Collaboration. The second data release from the European Pulsar Timing Array - V. Search for continuous gravitational wave signals. *AA*, 690:A118, 2024.

[54]M. Isi. Parametrizing gravitational-wave polarizations. *Classical and Quantum Gravity*, 40(20):203001, 2023.

[55]B. P. Abbott et al. The basic physics of the binary black hole merger GW150914. *Annalen der Physik*, 529(1–2), October 2016.

[56]P. Schmidt. Gravitational Waves From Binary Black Hole Mergers: Modeling and Observations. *Frontiers in Astronomy and Space Sciences*, Volume 7 - 2020, 2020.

[57]A. Buonanno and T. Damour. Effective one-body approach to general relativistic two-body dynamics. *Phys. Rev. D*, 59:084006, March 1999.

[58]P. Ajith et al. A phenomenological template family for black-hole coalescence waveforms. *Classical and Quantum Gravity*, 24(19):S689, September 2007.

[59]The LIGO Scientific Collaboration and the Virgo Collaboration and the KAGRA Collaboration. GWTC-4.0: Methods for Identifying and Characterizing Gravitational-wave Transients, 2025.

[60]P. Joshi et al. New Methods for Offline GstLAL Analyses, 2025.

[61]C. Alléné et al. The MBTA pipeline for detecting compact binary coalescences in the fourth LIGO-Virgo-KAGRA observing run. *Classical and Quantum Gravity*, 42(10):105009, May 2025.
[62]A. Nitz et al. Detecting Binary Compact-object Mergers with Gravitational Waves: Understanding and Improving the Sensitivity of the PyCBC Search. *The Astrophysical Journal*, 849(2):118, November 2017.
[63]Q. Chu et al. SPIIR online coherent pipeline to search for gravitational waves from compact binary coalescences. *Phys. Rev. D*, 105:024023, January 2022.
[64]S. Klimenko et al. Method for detection and reconstruction of gravitational wave transients with networks of advanced detectors. *Phys. Rev. D*, 93:042004, February 2016.
[65]W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, April 1970.
[66]N. Metropolis et al. Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953.
[67]J. S Speagle. Dynesty: a dynamic nested sampling package for estimating Bayesian posteriors and evidences. *Monthly Notices of the Royal Astronomical Society*, 493(3):3132–3158, February 2020.
[68]M. J. Williams, J. Veitch, and C. Messenger. Nested sampling with normalizing flows for gravitational-wave inference. *Phys. Rev. D*, 103:103006, May 2021.
[69]L. Weng. Flow-based deep generative models, October 2018.
[70]K. Wong et al. Gravitational-wave population inference with deep flow-based generative network. *Phys. Rev. D*, 101:123005, June 2020.
[71]K. Wong et al. Joint constraints on the field-cluster mixing fraction, common envelope efficiency, and globular cluster radii from a population of binary hole mergers via deep learning. *Phys. Rev. D*, 103:083021, Apr 2021.
[72]K. Wong et al. Constraining the primordial black hole scenario with bayesian inference and machine learning: The GWTC-2 gravitational wave catalog. *Phys. Rev. D*, 103:023026, January 2021.
[73]S. Colloms et al. Exploring the evolution of gravitational-wave emitters with efficient emulation: Constraining the origins of binary black holes using normalising flows, 2025.
[74]N. Laal et al. Deep neural emulation of the supermassive black-hole binary population, 2024.
[75]F. Stachurski, C. Messenger, and M. Hendry. Cosmological inference using gravitational waves and normalising flows, 2023.
[76]M. J. Williams, J. Veitch, and C. Messenger. Importance nested sampling with normalising flows. *Machine Learning: Science and Technology*, 4(3):035011, July 2023.
[77]K. Wong, M. Gabrié, and D. Foreman-Mackey. flowMC: Normalizing-flow enhanced sampling package for probabilistic inference in Jax, 2022.
[78]K. Wong, M. Isi, and T. Edwards. Fast gravitational wave parameter estimation without compromises, 2023.
[79]A. Malz and J. Veitch. Joint inference for gravitational wave signals and glitches using a data-informed glitch model, 2025.
[80]T. D. P. Edwards et al. ripple: Differentiable and Hardware-Accelerated Waveforms for Gravitational Wave Data Analysis, 2023.
[81]M. Mould, N. Wolfe, and S. Vitale. Rapid inference and comparison of gravitational-wave population models with neural variational posteriors. *Physical Review D*, 111(12), June 2025.

[82]L. Pacchiardi and R. Dutta. Likelihood-Free Inference with Generative Neural Networks via Scoring Rule Minimization, 2022.
[83]S. T. Radev et al. BayesFlow: Learning complex stochastic models with invertible neural networks, 2020.
[84]M. Dax et al. Real-Time Gravitational Wave Science with Neural Posterior Estimation. *Physical Review Letters*, 127(24), December 2021.
[85]J. Langendorff et al. Normalizing Flows as an Avenue to Studying Overlapping Gravitational Wave Signals. *Physical Review Letters*, 130(17), April 2023.
[86]D. MacKay. *Information Theory, Inference, and Learning Algorithms*. Copyright Cambridge University Press, 2003.
[87]G. Papamakarios, T. Pavlakou, and I. Murray. Masked Autoregressive Flow for Density Estimation, 2018.
[88]L. Dinh, D. Krueger, and Y. Bengio. NICE: Non-linear Independent Components Estimation, 2015.
[89]L. Dinh, J. Sohl-Dickstein, and S. Bengio. Density estimation using Real NVP, 2017.
[90]C. Durkan, A. Bekasov, I. Murray, and G. Papamakarios. Neural Spline Flows, 2019.
[91]Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.
[92]R. Frostig, M. Johnson, and C. Leary. Compiling machine learning programs via high-level tracing. 2018.
[93]B. P. Abbott et al. First upper limits from LIGO on gravitational wave bursts. *Physical Review D*, 69(10), May 2004.
[94]DeepMind and I. Babuschkin et al. The DeepMind JAX Ecosystem, 2020.
[95]D. Foreman-Mackey et al. emcee: The MCMC Hammer. *Publications of the Astronomical Society of the Pacific*, 125(925):306–312, March 2013.
[96]I. Loshchilov and F. Hutter. SGDR: Stochastic Gradient Descent with Warm Restarts, 2017.
[97]L. Biewald. Experiment Tracking with Weights and Biases, 2020. Software available from wandb.com.
[98]Y. Li and R. E. Turner. Rényi Divergence Variational Inference, 2016.
[99]H. Zimmermann et al. Nested Variational Inference, 2021.
[100]C. Huang et al. Improving Explorability in Variational Inference with Annealed Variational Objectives. *CoRR*, abs/1809.01818, 2018.
[101]M. Arbel, A. Matthews, and A. Doucet. Annealed Flow Transport Monte Carlo, 2021.
[102]S. Ramasinghe et al. Robust normalizing flows using bernstein-type polynomials, 2022.
[103]G. Lorentz. *Bernstein Polynomials*. AMS Chelsea Publishing Series. Chelsea Publishing Company, 1986.
[104]S. Bernstein. Démonstration du théorème de weierstrass fondée sur le calcul des probabilités. *Communications of the Kharkov Mathematical Society*, 1912.
[105]M. J. Williams et al. uofgravity/glasflow: v0.4.1, October 2024.
[106]S. Cohen, B. Amos, and Y. Lipman. Riemannian convex potential maps. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 2028–2038. PMLR, July 2021.
[107]R. Chen et al. Neural Ordinary Differential Equations, 2019.
[108]E. Mathieu and M. Nickel. Riemannian Continuous Normalizing Flows, 2020.