Importance nested sampling with normalising flows

IOP Publishing Mach. Learn.: Sci. Technol. 4 (2023) 035011 https://doi.org/10.1088/2632-2153/acd5aa

MACHINE LEARNING
Science and Technology

OPEN ACCESS

RECEIVED
22 February 2023

REVISED
4 May 2023

ACCEPTED FOR PUBLICATION
15 May 2023

PUBLISHED
25 July 2023

PAPER

Importance nested sampling with normalising flows

Michael J Williams*, John Veitch and Chris Messenger

SUPA, School of Physics and Astronomy, University of Glasgow, Glasgow, G12 8QQ, United Kingdom
* Author to whom any correspondence should be addressed.

E-mail: m.williams.4@research.gla.ac.uk

Keywords: Bayesian inference, nested sampling, machine learning, normalising flows, gravitational waves

Abstract

We present an improved version of the nested sampling algorithm `nessai` in which the core algorithm is modified to use importance weights. In the modified algorithm, samples are drawn from a mixture of normalising flows and the requirement for samples to be independently and identically distributed (i.i.d.) according to the prior is relaxed. Furthermore, it allows for samples to be added in any order, independently of a likelihood constraint, and for the evidence to be updated with batches of samples. We call the modified algorithm `i-nessai`. We first validate `i-nessai` using analytic likelihoods with known Bayesian evidences and show that the evidence estimates are unbiased in up to 32 dimensions. We compare `i-nessai` to standard `nessai` for the analytic likelihoods and the Rosenbrock likelihood, the results show that `i-nessai` is consistent with `nessai` whilst producing more precise evidence estimates. We then test `i-nessai` on 64 simulated gravitational-wave signals from binary black hole coalescence and show that it produces unbiased estimates of the parameters. We compare our results to those obtained using standard `nessai` and `dynesty` and find that `i-nessai` requires 2.68 and 13.3 times fewer likelihood evaluations to converge, respectively. We also test `i-nessai` of an 80 s simulated binary neutron star signal using a reduced-order-quadrature basis and find that, on average, it converges in 24 min, whilst only requiring 1.01×10^6 likelihood evaluations compared to 1.42×10^6 for `nessai` and 4.30×10^7 for `dynesty`. These results demonstrate that `i-nessai` is consistent with `nessai` and `dynesty` whilst also being more efficient.

1. Introduction

John Skilling proposed nested sampling in [1, 2] and it has since seen widespread use in astronomical data analysis, including but not limited to the analyses of gravitational waves [3, 4], asteroseismology [5] and cosmology [6].

Nested sampling is a Monte Carlo algorithm that approximates the Bayesian evidence

$$Z \equiv p(d|H) = \int p(d|\boldsymbol{\theta}, H) \mathrm{d}\boldsymbol{\theta}, \tag{1}$$

for some observed data d with an assumed model H over the parameters $\boldsymbol{\theta}$ where $\mathcal{L}(\boldsymbol{\theta}) \equiv p(d|\boldsymbol{\theta}, H)$ is the likelihood. This is usually considered in the context of Bayes' theorem

$$p(\boldsymbol{\theta}|d, H) = \frac{p(d|\boldsymbol{\theta}, H)p(\boldsymbol{\theta}|H)}{p(d|H)}, \tag{2}$$

where $\pi(\boldsymbol{\theta}) \equiv p(\boldsymbol{\theta}|H)$ is the prior and $p(\boldsymbol{\theta}|d, H)$ is the posterior. Samples from the latter are a by-product of approximating the evidence.

When implementing nested sampling, the main challenge is drawing new points from the likelihood-constrained prior at a given iteration. There are different approaches to this such as using Markov Chain Monte Carlo (MCMC), slice sampling or sampling from bounding distributions [7]. There have also

been efforts to incorporate machine learning into nested sampling for approximating the likelihood [8], in the proposal process [9, 10] and for sampling from arbitrary priors [11].

In Williams *et al* [10], we proposed `nessai`, a nested sampling algorithm that uses normalising flows to approximate the likelihood-constrained prior at different iterations. We showed that this approach could speed up convergence and allowed for natural parallelisation of the likelihood. However, we noted that a significant portion of compute time was being spent performing rejection sampling to ensure points were distributed according to the prior, and this, alongside the inherently serial nature of nested sampling, set a lower limit on how fast the algorithm could be.

In this work, we present a modified nested sampling algorithm based on importance sampling that addresses the aforementioned bottlenecks. In particular, this modified algorithm:

- incorporates normalising flows in a similar fashion to Williams *et al* [10],
- removes the requirement for samples to be independently and identically distributed (i.i.d.) and distributed according to the prior,
- allows samples to be added in any order independent of a likelihood constraint,
- allows the evidence to be updated for batches of samples.

Taken together, these changes improve the efficiency of the algorithm, reducing the number of required likelihood evaluations by up to an order of magnitude over our previous version, and greatly increasing the scalability of the algorithm.

This is especially relevant in the context of gravitational-wave data analysis, where nested sampling is the de facto analysis algorithm [3, 4]. As of the last LIGO-Virgo-KAGRA [12–14] observing run, there are 90 confirmed detected compact binaries [15–17] and this number is expected to increase by a factor of $\sim$3.3 in the fourth observing run [18]. This presents a significant computational challenge since typical analyses take of order days to weeks. Furthermore, a subset of these analyses are currently only possible at great computational cost [19, 20]. The algorithm we present brings the possibility of tackling these challenging analyses and dramatically reduces the wall-time required to complete an analysis.

This paper is structured as follows: in section 2 we present background theory on nested sampling and various alternative formulations that this work builds upon. We then describe a simplified version of our modified algorithm and validate it in section 3. This is followed by a description of the complete method and algorithm in section 4. Finally, we present results in section 6 and discuss them in section 7.

2. Background

2.1. Nested sampling

Nested sampling [1, 2] is a stochastic sampling algorithm where the Bayesian evidence ($p(d|H)$ or Z) is rewritten as a one-dimensional integral in terms of the prior volume X

$$Z = \int_0^1 \mathcal{L}(X)\mathrm{d}X, \tag{3}$$

where $\mathcal{L}(X)$ is the likelihood at a given prior volume X. If the likelihood $\mathcal{L}(X)$ is a well-behaved function, then this formulation allows for the evidence to be approximated using an ordered sequence of decreasing prior volumes X_i such that

$$Z \approx \hat{Z} = \sum_{i=1}^{N} \mathcal{L}_i w_i, \tag{4}$$

where $\mathcal{L}_i = \mathcal{L}(X_i)$ is the likelihood at X_i and the weights w_i are, for example, given by $w_i = (1/2)(X_i - X_{i+1})$. The prior volume at a given iteration X_i is computed in terms of the previous prior volume X_{i-1}, the number of points within the likelihood-constrained prior N_{live} and the shrinkage factor t_i which is a random variable in $(0, 1)$ with probability density function $P(t) = N_{\mathrm{live}} t^{N_{\mathrm{live}}-1}$. The mean and standard deviation of $\log t$ are therefore

$$\mu[\log t] = -\frac{1}{N_{\mathrm{live}}}, \qquad \sigma[\log t] = \frac{1}{N_{\mathrm{live}}}. \tag{5}$$

Since each draw of $\log t_i$ is independent, the prior volume at a given iteration i is approximately $X_i \approx \exp(-i/N_{\mathrm{live}})$. We can express this as a recursive relationship in terms of t_i where

$$X_i = t_i X_{i-1}. \tag{6}$$

The overall nested sampling algorithm can then be summarised as follows:

1. Draw N_{live} points $\{\theta_i\}_{i=1}^{N_{\text{live}}} \sim \pi(\theta)$ and compute the likelihood $\mathcal{L}_i = \mathcal{L}(\theta_i)$ of each point,
2. Choose the point θ^* with the lowest likelihood $\mathcal{L}^* \equiv \mathcal{L}(\theta^*)$,
3. Draw new points $\hat{\theta}$ until $\mathcal{L}(\hat{\theta}) > \mathcal{L}^*$,
4. Replace θ^* with the new point $\hat{\theta}$ and add θ^* to the *nested samples*,
5. Update the evidence estimate via equation (4),
6. Repeat steps 2–5 until a stopping criterion is met.

The algorithm returns a set of nested samples, with corresponding prior volumes and likelihoods, and an evidence estimate with a corresponding error. The stopping criterion is typically related to the fractional change in the evidence between iterations [7].

Given a completed nested sampling run, posterior samples can be drawn by computing the posterior weights for each nested sample

$$p_i = \frac{\mathcal{L}_i w_i}{\hat{Z}}, \tag{7}$$

and then, for example, rejection sampling can be used to obtain samples from the posterior distribution.

This formulation has been extended and modified in various works, such as to allow for a varying number of live points [21], to use different proposal methods [6, 10, 22], or even using different definitions of the weights w_i in equation (4) [23–25], which is the focus of this work.

As mentioned previously, the main challenge when implementing a nested sampling algorithm is drawing live points that are i.i.d according to the prior and satisfy the likelihood constraint at the current iteration. There are various different approaches to this. In the original paper [2], Skilling proposes using MCMC over the prior and accepting only those points for which $\mathcal{L}(\theta) > \mathcal{L}^*$ until the correlation with the starting point (one of the existing samples) has been lost. This method requires a random walk that can adapt to the continuously shrinking likelihood-constrained prior and a method for determining the number of steps to take [7]. Further modifications are often needed to handle multi-modality and complex correlations between parameters, for example, as implemented in Veitch *et al* [3]. Similarly, slice sampling [26], where samples are drawn from a randomly oriented line within the likelihood-constrained prior, has also been used [6]. The challenge in this case is choosing the direction of the line and how to sample from it. Another approach is to sample from a bounding (or proposal) distribution that directly approximates or contains the likelihood-constrained prior, such as ellipsoids [22, 25] or mixtures of these to handle, for example, multi-modality. Finally, there are algorithms that use a mix of the aforementioned methods [27, 28].

One limitation of nested sampling is its inherently sequential nature. This is addressed in part by dynamic nested sampling [21] where an initial exploratory run is then retroactively improved upon by adding samples in regions of interest. However, the core algorithm is still sequential. Diffusive nested sampling [23] tackles this by using a multi-level exploration method which allows returning to lower likelihoods. We draw from this variant of nested sampling when developing our modified algorithm.

Machine learning has also been incorporated into nested sampling algorithms to address some of the limitations and accelerate inference. In Graff *et al* [8], the likelihood is approximated using a neural network which, for computationally expensive likelihoods, can reduce the overall computational cost. In Alsing and Handley [11], normalising flows are used to allow for arbitrary priors which could otherwise not be used, for example, when using a posterior distribution as the prior for subsequent inference. Normalising flows have also been applied specifically to the proposal process. The algorithm proposed in Moss [9] improves MCMC efficiency by transforming the sampling parameter space to a simpler space using a normalising flow and in Williams *et al* [10], we proposed `nessai` which uses normalising flows to directly approximate the likelihood-constrained prior and to avoid the need for MCMC, greatly improving sampling efficiency. We discuss `nessai` in detail in section 2.2.

2.2. `nessai`: Nested sampling with normalising flows

In Williams *et al* [10], to address the aforementioned challenged of proposing new live points from the likelihood-constrained prior, we introduced `nessai`, a nested sampling algorithm that incorporates normalising flows in the proposal process. We now review the core aspects of `nessai`.

Normalising flows are a family of parameterised invertible transforms that can be trained via an optimisation process to map from a simple distribution $p_{\mathcal{Z}}(z)$ in the latent space ($\mathcal{Z}$) to a complex distribution $p_{\mathcal{X}}(x)$ in the data space ($\mathcal{X}$). They were first proposed in [29, 30] and have since been applied to a range of problems including image synthesis, noise modelling, physics and simulation-based inference [31–33].

One property that distinguishes normalising flows from other generative models, such has variational autoencoders [34] and generative adversarial networks [35], is their construction allows for an explicit expression for the learnt distribution $p_{\mathcal{X}}(x)$

$$p_{\mathcal{X}}(x) = p_{\mathcal{Z}}(f(x)) \left| \det \left(\frac{\partial f(x)}{\partial x} \right) \right|, \tag{8}$$

where f is the normalising flow and $|\det(\partial f(x)/\partial x)|$ is the Jacobian determinant. The normalising flow f must be constructed such that the mapping is invertible and has a tractable Jacobian determinant. Depending on how the mapping is constructed, they fall into two main categories: *autoregressive flows* and *coupling flows*. The former have more expressive power at the cost of being more computational expensive to train and evaluate, whereas the opposite is true for the later [32]. In Williams *et al* [10] and in this work, we use coupling flows based on RealNVP [36]. For a complete review of normalising flows, see Kobyzev *et al* [31] and Papamakarios *et al* [32].

In `nessai`, at a given iteration, a normalising flow is trained using the current live points. The trained flow maps the live points from the sampling space $\mathcal{X}$ to samples in the latent space $\mathcal{Z}$. New samples are then drawn by sampling from a truncated latent distribution and applying the inverse mapping f^{-1}. Finally, rejection sampling is used to ensure that the samples are distributed according to the prior. The benefit of this approach is that all the samples are i.i.d. removing the need for MCMC sampling. Furthermore, since the points are drawn in parallel, the likelihood evaluation can also be parallelised, further reducing the time taken for the algorithm to converge.

However, we found that the rejection sampling step can be inefficient and lead to many samples being discarded. In particular, for the results we presented in Williams *et al* [10], this rejection sampling accounted for approximately 40% of the total sampling time and, unlike the likelihood evaluation, this time cannot be significantly reduced via parallelisation. Additionally, we found it was necessary to reparameterise certain parameters that would otherwise be difficult to sample or make the rejection sampling inefficient. For example, parameters with posterior distributions that rail against the prior bounds could be under-sampled when the latent space is truncated. Whilst reparameterising these problematic parameters does address these issues, it requires prior knowledge of the parameter space.

2.3. Alternative formulations of nested sampling

In this section, we highlight alternative formulations of nested sampling that will be built upon in this work.

2.3.1. Diffusive nested sampling

Diffusive nested sampling [23] uses a multi-level exploration method where a mixture of constrained distributions is sampled from at each iteration using MCMC. The constrained distributions are added sequentially and each contains approximately e^{-1} of the prior volume of the previous. In contrast to standard nested sampling approaches, all the samples from the MCMC chain are kept and those that do not meet the current likelihood criteria are added to the previous level. The values for the prior volume X are estimated using the fraction of samples above the likelihood threshold compared to the total number of samples.

This variation of nested sampling avoids the strict likelihood constraint and utilises all the samples drawn at a given iteration but still requires that new points be sampled from the prior.

2.3.2. Importance nested sampling

Importance nested sampling was proposed in Cameron and Pettitt [24] and expanded upon in Feroz *et al* [25]. In this version of nested sampling, the evidence integral is approximated in terms of a *pseudo-importance sampling density* $Q(\theta)$

$$\hat{Z} = \frac{1}{N_{\text{Total}}} \sum_{i=1}^{N_{\text{Total}}} \frac{\mathcal{L}(\theta_i)\pi(\theta_i)}{Q(\theta_i)}, \tag{9}$$

where N_{Total} is the total number of nested samples. Posterior weights are then computed using

$$p_i = \frac{\mathcal{L}(\theta_i)\pi(\theta_i)}{N_{\text{Total}} Q(\theta_i)}, \tag{10}$$

and these can be used to obtain posterior samples via rejection sampling, or used directly in weighted histograms or kernel density estimates to approximate marginal distributions.

In standard importance sampling, the unbiased estimator for the variance of the evidence is given by

$$\sigma^2[\hat{Z}] = \frac{1}{N_{\text{Total}}(N_{\text{Total}}-1)} \sum_{i=1}^{N_{\text{Total}}} \left[\frac{\mathcal{L}(\theta_i)\pi(\theta_i)}{Q(\theta_i)} - \hat{Z} \right]^2, \tag{11}$$

however, this does not apply when using a pseudo-importance sampling density, which is the case in `multinest` [25].

In `multinest` [22, 25], one or more ellipsoidal distributions are used to construct an approximation of the current likelihood contour defined by $\mathcal{L}^*$. New points are then drawn from within this proposal distribution and their likelihood evaluated until $\mathcal{L}(\hat{\theta}) > \mathcal{L}^*$ and, similarly to diffusive nested sampling, all these points are used in the evidence summation and define the number of points within a level n_i. The pseudo-importance sampling density for each point is given by

$$Q(\theta) = \frac{1}{N_{\text{tot}}} \sum_{i=1}^{N_{\text{iter}}} \frac{n_i E_i(\theta)}{V_{\text{tot},i}}, \tag{12}$$

where $V_{\text{tot},i}$ is the volume of the bounding distribution, E_i is an indicator function that is 1 if the point lies within the ith ellipsoidal decomposition and 0 otherwise, N_{iter} is the number of iterations, where an iteration is an instance of the ellipsoidal decomposition and N_{tot} is the total number of points $N_{\text{tot}} = \sum_{i=1}^{N_{\text{iter}}} n_i$.

This formulation of the evidence removes the requirement that samples are distributed according to the likelihood-constrained prior so long as the exact distribution of nested samples $Q(\theta)$ can be written down. However, only a single point is removed and updated between each update of the ellipsoidal decomposition, therefore convergence will require computing the decomposition hundreds or thousands of times. This makes it ill-suited to use with normalising flows that are, in comparison, slow to train.

2.3.3. Nested sampling via sequential Monte Carlo (SMC)

SMC is a general extension of importance sampling where random samples with corresponding weights are drawn from a sequence of probability densities such that they converge towards a target density [37]. These algorithms are typically comprised of three main steps: *mutation* in which the samples are moved towards the target density via a Markov kernel, *correction* where the weights of the samples are updated, and *selection* where the samples are resampled according to their weights.

In Salomone *et al* [38], the authors draw parallels between nested sampling and SMC and show that nested sampling is a type of adaptive SMC algorithm where weights are assigned suboptimally. They also highlight several limitations of the standard nested sampling algorithm, including the assumption of independent samples. They propose a new class of SMC algorithms called nested sampling via SMC and demonstrate that it is equivalent to nested sampling but addresses the aforementioned limitations. This formulation bares similarities to the importance nested sampling [24, 25] but removes batches of live points at each iteration and includes the mutation and selection steps that are typical in SMC.

A downside of this formulation is that since the points are resampled at each iteration, some samples for which the likelihood has been evaluated are discarded and not used in the final evidence estimate or output. In this work, we aim to avoid this by not including the resampling step and instead directly using the weights of the samples when constructing the next level.

3. Core importance nested algorithm

In this section, we motivate and present the core importance nested sampling algorithm used in `nessai`. We extend the formulation of importance nested sampling described in section 2.3.2 to allow the use of normalising flows instead of ellipsoidal bounding distributions. We also draw on the design of diffusive nested sampling where the likelihood constraint is relaxed such that samples are not rejected based on their likelihood.

We start by considering the definition of the evidence from equation (9). In importance nested sampling, the aim is to construct an importance sampling density $Q(\theta)$, which we will call *meta-proposal*, from which samples can be drawn, and used to estimate the evidence. The error on this estimate is given by equation (11) and depends on the number of samples N_{tot} and $Q(\theta)$. If we consider a fixed number of samples, the meta-proposal that maximises the effective sample size (ESS) of the set of summands $\mathcal{L}(\theta_i)\pi(\theta_i)/Q(\theta_i)$, and therefore provides the most precise evidence estimate, will be $Q(\theta) \equiv \mathcal{L}(\theta)\pi(\theta)/Z$, i.e. when $Q(\theta)$ is equal to the target posterior. Since the evidence is unknown *a-priori*, the aim is to construct the meta-proposal such that $Q(\theta) \propto \mathcal{L}(\theta)\pi(\theta)$.

This formulation of nested sampling is closely related to variational inference [39], where the goal is to approximate a target probability density. In this case, the target density is $\mathcal{L}(\theta)\pi(\theta)$ and the approximate distribution is the meta-proposal $Q(\theta)$. The difference is in how the approximate distribution is obtained. In variational inference, the approximate distribution is optimised by minimising a variational objective, whereas in this algorithm the distribution is constructed by progressively sampling and adding proposal distributions.

We now consider how to construct the meta-proposal using normalising flows. An important difference between the ellipsoidal bounds used in `multinest` and normalising flows is the space over which they are defined. For a normalising flow, this depends on the domain of the latent distribution $p_{\mathcal{Z}}$. For the typical case of a n-dimensional Gaussian the mapping is defined such that $f : \mathbb{R}^n \to \mathbb{R}^n$, so the flow will have infinite support. We need the meta-proposal to have the same support as the prior, so we include an additional invertible transform that maps from $\mathbb{R}^n$ to a bounded space, such as the sigmoid $s(x) = [1 - \exp(-x)]^{-1}$. We denote the bounded space $\mathcal{X}$ and the unbounded space $\mathcal{X}'$.

Therefore, instead of considering a series of bounded distributions, we consider a set of N normalised proposal distributions (normalising flows) $\{q_1, \ldots, q_N\}$ all defined over the entire prior volume and with corresponding weights α_j defined such that $\sum_{j=1}^{N} \alpha_j = 1$. The overall proposal density as a function of θ is given by

$$Q(\theta) = \sum_{j=1}^{N} \alpha_j q_j(\theta). \tag{13}$$

In practice, in order to sample from $Q(\theta)$ we first draw a proposal $k \in \{1, \ldots, N\}$, drawn from a categorical distribution with category weights $\{\alpha_1, \ldots, \alpha_N\}$, then a sample is drawn from the sub-proposal $q_k(\theta)$.

With this formulation, we can compute an estimate of the evidence for a set of samples drawn from $Q(\theta)$ using equation (9) and, as noted in Feroz *et al* [25], we no longer require new samples that have monotonically increasing likelihood values. Furthermore, as described in Salomone *et al* [38], we do not require that new samples be i.i.d. or distributed according to the likelihood-constrained prior. This removes the need for the rejection sampling that was a bottleneck in the version of `nessai` we described in Williams *et al* [10].

We now outline a simplified importance nested sampling algorithm which we build upon in later sections. The main changes are to steps 2–5 of the standard nested sampling algorithm outlined in section 2. Instead of removing a point and finding a single replacement point, we construct a proposal distribution $q_j(\theta)$ based on the points sampled thus far and draw a set of N_j new points $\Theta_j = \{\theta_i\}_{i=1}^{N_j}$ which are added to the overall set of points $\{\Theta_1, \ldots, \Theta_{j-1}\}$. The meta-proposal $Q(\theta)$ is then updated to include $q_j(\theta)$ and the evidence is updated. The new importance nested sampling algorithm therefore consists of the following steps:

1. Draw N_{live} points $\{\theta_i\}_{i=1}^{N_{\text{live}}} \sim \pi(\theta)$ and compute the likelihood $\mathcal{L}_i = \mathcal{L}(\theta_i)$ of each point,
2. add the next proposal distribution $q_j(\theta)$,
3. draw N_j samples from $\Theta_j = \{\theta_i\}_{i=1}^{N_j} \sim q_j(\theta)$ and compute the corresponding likelihoods,
4. update the meta-proposal $Q(\theta)$ to include $q_j(\theta)$,
5. compute the evidence and the corresponding error via equations (9) and (11),
6. repeat steps 2–5 until a stopping criterion is met,
7. redraw independent samples from the final meta-proposal,
8. compute the final evidence and posterior weights using the independent samples and equations (9) and (10).

This includes an additional step not present in standard nested sampling: redrawing independent samples from the final meta-proposal. Since subsequent proposals are constructed using samples from the previous iterations, new samples are not i.i.d. and equations (9)–(11) do not strictly apply. However, once the meta-proposal is finalised, i.i.d. samples can be sampled and used to compute unbiased estimates of the evidence and posterior weights.

The design of the algorithm hinges on how the next proposal distribution is added, how the number of samples drawn from each proposal (N_j) is determined and how the weights in the meta-proposal $Q(\theta)$ are determined. Note that the first proposal distribution $q_0(\theta)$ will typically be the prior. We now apply this simplified algorithm to a toy example.

3.1. Toy example

In this toy example, we consider a simple problem with an analytic evidence and posterior distribution. We apply the algorithm described in section 3 but with some simplifications. This allows us to validate the core algorithm.

We use a 2-dimensional Gaussian likelihood with mean $\mu_{\mathcal{L}} = 0$ and standard deviation $\sigma_{\mathcal{L}} = 1$ and a Gaussian prior a with mean $\mu_\pi = 0$ and standard deviation $\sigma_\pi = 2$. The posterior distribution is therefore another Gaussian distribution with mean $\mu_{\text{Post}} = 0$ and standard deviation $\sigma_{\text{Post}} = \sqrt{1/[(1/\sigma_{\mathcal{L}}^2) + (1/\sigma_\pi^2)]}$. The evidence is given by a Gaussian distribution with mean μ_π and standard deviation $\sqrt{\sigma_{\mathcal{L}}^2 + \sigma_\pi^2}$ evaluated at $\mu_{\mathcal{L}}$, so $Z_{\text{Analytic}} = 0.03183$.

To make the comparison between the true and sampled posterior distributions easier, we express the posterior distribution in terms of the log-likelihood $p(\ln \mathcal{L})$. To do this, we note that the posterior distribution defined in terms of the radius squared is $p(r^2) = \chi_2^2(r^2)/\sigma_{\text{Post}}^2$ where χ_2^2 is a chi-squared distribution with two degrees of freedom. Then

$$p(\ln \mathcal{L}) = p(r^2) \left| \frac{\partial r^2}{\partial \ln \mathcal{L}} \right|, \tag{14}$$

where

$$r^2 = -2\sigma_{\mathcal{L}}^2 \left[\ln(2\pi\sigma_{\mathcal{L}}^2) + \ln \mathcal{L} \right], \tag{15}$$

which is defined on $[0, \infty)$ since the maximum possible value of the log-likelihood is $\ln \mathcal{L} = -\ln(2\pi\sigma_{\mathcal{L}}^2)$.

The four steps we must define for the simplified algorithm are: how to construct each proposal distribution, how to determine the number of samples to draw from each proposal, how to determine the weights for each proposal in the meta-proposal and a stopping criterion. For the proposals, instead of normalising flows, we use 2-dimensional Gaussian distributions $q_j(\theta)$ with mean zero and different standard deviations. We determine the standard deviation of each proposal by setting a likelihood threshold $\mathcal{L}_t$ such that 50% of the points from the previous iteration are discarded and then compute the standard deviation of the remaining points. We set the number of samples drawn from each proposal to constant $N_j = N_{\text{live}} = 500$ and set the weights for the meta-proposal α_j to be equal. This means that each proposal will contribute equally to the meta-proposal. Finally, instead of using a stopping criterion, we define a fixed number of proposal distributions (iterations) $N = 4$ where the first is the prior distribution $q_0(\theta) \equiv \pi(\theta)$. This is akin to fixing the number of iterations in a normal nested sampling algorithm. Once the final proposal has been added, we draw i.i.d. samples from the finalised meta-proposal and compute the final unbiased evidence estimate and posterior weights.

We present the results obtained with this algorithm in figure 1. This shows the samples and the 1-σ contours for each of the proposal distributions, along with the corresponding distribution of log-likelihoods. We compute two evidence estimates: one with the initial samples that are not i.i.d. $\hat{Z} = 0.031\,77 \pm 0.000\,42$ and the other with the final i.i.d. samples $\hat{Z} = 0.031\,91 \pm 0.000\,42$. We find that both are in agreement with the analytic value, $Z = 0.031\,83$, but, as we will see in section 6.1, the initial estimate will be biased, the bias is just very small in this simple example. This demonstrates that the underlying algorithm can reliably estimate the evidence. We also compute the posterior weights using equation (10) and plot the weighted histogram in log-likelihood space, which shows good agreement with the analytic expression from equation (14). Overall, these results demonstrate the principles of the proposed algorithm and that, for a simple toy example, it converges to the expected result.

4. Method

Having outlined the underlying algorithm, we now describe each of the steps in the complete algorithm in detail.

4.1. Constructing proposal distributions

With this formulation of nested sampling, the main design choice is how to construct the proposal distribution $q_j(\theta)$ at each iteration (step 2). This is akin to drawing new samples in standard nested sampling however, since we no longer require an ordered sequence of points with decreasing prior volume, new points no longer need strictly increasing likelihood values.

The new proposal $q_j(\theta)$ at each iteration is defined in terms of a likelihood threshold $\mathcal{L}_t$: of the current N_{live} points, M_j are discarded based on a likelihood threshold and the remaining $N_{\text{live}} - M_j$ points are used to

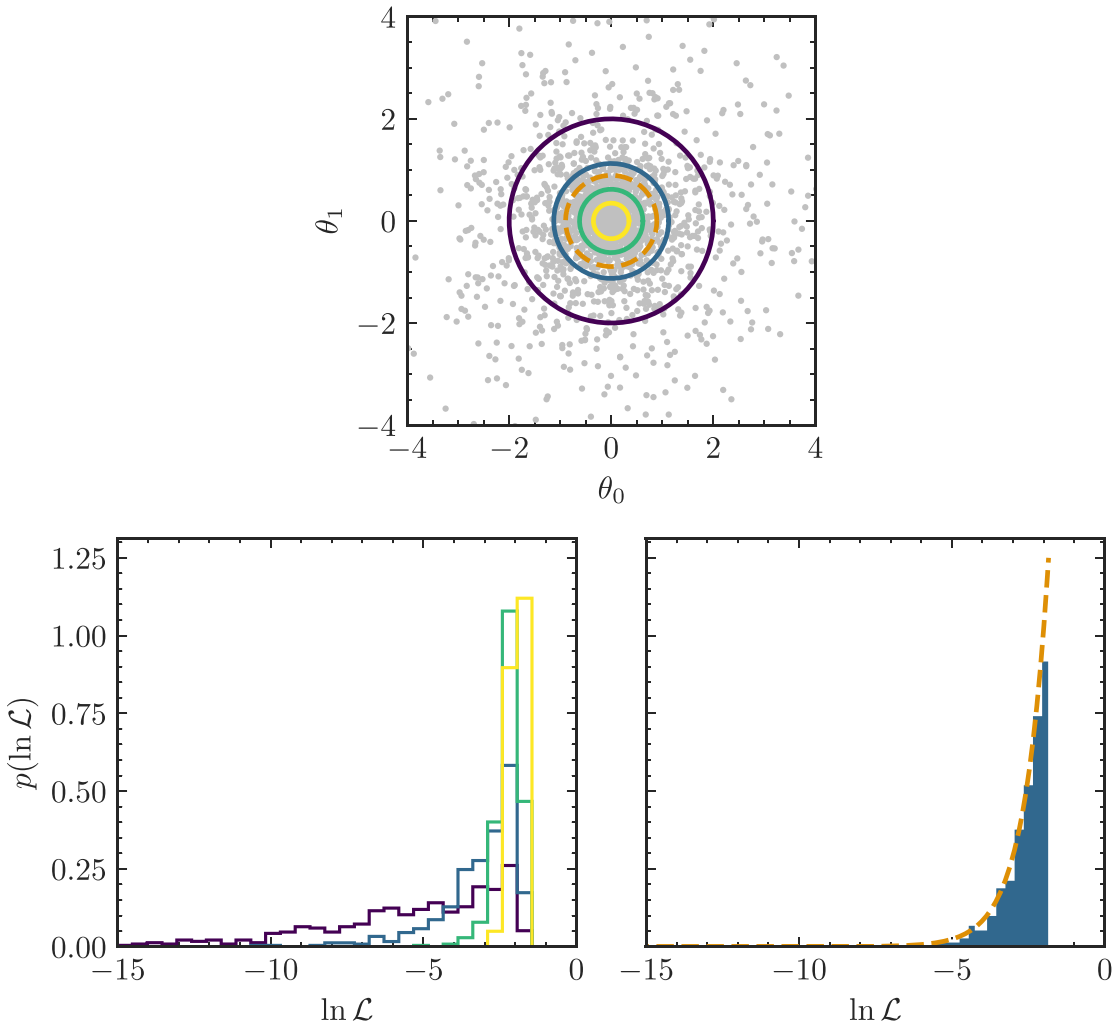

Figure 1. Results for the toy example described in section 3.1. Top: the final samples are shown in grey, the solid lines show the 1-σ contour for each proposal distribution starting with the prior, lighter colours indicate later iterations. The orange dashed line shows the 1-σ contour for the analytic posterior distribution. Bottom left: distribution of log-likelihoods for the final samples drawn from each proposal distribution. Bottom right: distribution of the log-likelihoods of the final samples weighted by their corresponding posterior weights. The orange dashed line indicates the analytic posterior distribution computed using equation (14).

construct the next proposal distribution $q_j(\theta)$. In our implementation, this is done by training a normalising flow. The result is a series of increasingly dense proposal distributions, which is equivalent to the distributions becoming narrower in the log-likelihood space. This is shown in figure 1.

We therefore require a method for determining the likelihood threshold $\mathcal{L}_t$ used to determine how many points will be discarded before constructing the next proposal distribution. We consider two methods, both of which use weights

$$w_i = \frac{\pi(\theta_i)}{Q(\theta_i)}, \tag{16}$$

which quantify the relative importance of each sample θ_j compared to the prior. Additionally, one could include the likelihood in the weights, however, we leave this for future work.

In the first method, the threshold $\mathcal{L}_t$ is determined using the $(1-\rho)$ quantile of the likelihood values of the samples from the previous iteration, where ρ is set by the user. To account for non-prior distributed samples used in our algorithm, we use a weighted quantile, where the weights are given by equation (16). This method is based on the standard method used in SMC [38] and diffusive nested sampling [23], but with the addition of the weighted quantile.

The second method we consider is closely related to the first but uses log-weights $\log w_i$ instead of w_i. We consider the normalised sum of $\log w_i$ for the set of N live points ordered by increasing likelihood

$$\lambda(M) = \frac{\sum_{m=1}^{M} \log w_m}{\sum_{i=1}^{N} \log w_i}, \tag{17}$$

where M is the number of live points to be discarded. We then determine the value of M at which $\lambda(M) \geqslant \rho$, for $\rho \in [0, 1]$ and set $\mathcal{L}_t \equiv \mathcal{L}(\theta_M)$. This is analogous to shrinking the log-prior volume by a factor ρ at each iteration whilst also accounting for the different weights of each sample. In practice, since the normalising flows have support over the entire prior volume, this results in increasing the entropy of $q_j(\theta)$. We therefore denote this as the *entropy-based* method to distinguish it from the quantile-based method.

For both methods, we employ a maximum number of live points that can be removed—this prevents the remaining live points being too few to robustly train the next normalising flow. This maximum together with the value of ρ will determine the total number of samples used in the algorithm. We also employ a minimum number of samples to ensure a minimum change in distribution of training data between subsequent proposals. We discuss the advantages and disadvantages of both methods in appendix B.

4.2. Training normalising flows with weights

As discussed in section 2.3.3, it is common practice in SMC to resample at each iteration prior to the mutation step. Different sampling methods can be used, but they all keep the total number of samples constant by including repeated samples. This works when the mutation step is a Markov kernel, but in this work we use a normalising flow to perform the equivalent of the mutation step and, when training a normalising flow duplicates in the training data, can be problematic. In extreme cases, where only a few samples are representative, the training data could contain tens of copies of the same sample, which will make training unstable.

Without a step that is equivalent to resampling, deficiencies in training can have a cumulative effect. For example, if the mapping learnt by the normalising flow $q_j(\theta)$ under-samples a region of the space compared to the target, then if another normalising flow $q_{j+1}(\theta)$ is trained with samples drawn using $q_j(\theta)$ then $q_{j+1}(\theta)$ will also under-sample the same region. To counteract this effect, we include weights in the approximation of (Kullback–Leilber divergence) used to train the normalising flow. We describe this in detail in appendix A. To train the jth flow, we use all samples from the current meta-proposal $Q_{j-1}(\theta)$ that satisfy the likelihood constraint $\mathcal{L}(\theta) > \mathcal{L}_t$ and then minimise

$$\text{Loss} = -\frac{1}{N} \sum_{i=1}^{N} w_i \log q_j(\theta_i), \tag{18}$$

where $q_j(\theta)$ is given by equation (8) and w_i are the weights for each sample. In principle these weights could include the likelihood, however in this work we use the weights given by equation (16) which are proportional to the ratio of the likelihood-constrained prior and the likelihood-constrained meta-proposal.

4.3. Drawing samples from the proposal distributions

At a given iteration j, once the normalising flow $q_j(\theta)$ has been trained (step 2), we sample from the flow (step 3) and evaluate the likelihood for each new sample. This involves sampling from the latent distribution $p_{\mathcal{Z}}(z)$ and then applying the inverse flow mapping f^{-1} to obtain samples in $\mathcal{X}'$. These samples must then be mapped backed to the original space $\mathcal{X}$, where the likelihood can be computed.

The number of samples drawn at a given iteration N_j should be determined by drawing from a multinomial distribution with N possible outcomes (the number of proposal distributions) and $N_{\text{Total}} = \sum_{j=1}^{N} N_j$ trials, however the weights for each outcome are not known prior to sampling. Instead, we set N_j and determine the weight for the current iteration α_j based on its value. We allow N_j to either be equal to the number of samples removed at that iteration (M_j) or kept constant ($N_j = N_{\text{live}}$). The former will maintain a fixed number of live points N_{live} throughout the run whereas the latter allows for N_{live} to vary. We discuss the consequences of this approximation in sections 4.4 and 4.7.

Similarly to diffusive nested sampling, all the samples are kept irrespective of their likelihood, which means that samples can 'leak' below the current likelihood threshold.

4.4. Updating the meta-proposal

Having drawn samples from the current proposal distribution, the meta-proposal $Q(\theta)$ must be updated. The overall form of $Q(\theta)$ will depend on the weights α_j that are assigned to each proposal. Whilst adding proposals, we approximate the weights as $\alpha_j \propto N_j$ and normalise them such that they sum to one. This approximation can be corrected for once the sampling has been terminated by fixing the weights to their values from sampling, recomputing N_j by sampling from a multinomial distribution with weights $\{\alpha_0, \ldots, \alpha_{N_j}\}$ and drawing new samples from each $q_j(\theta)$ according to N_j. However, in practice, we find error introduced by this approximation to be significantly smaller than the overall error of the estimated evidence.

4.5. Stopping criterion

We define the stopping criterion to be the ratio of the evidence between the live points and the current evidence

$$\text{Condition} = \frac{\hat{Z}_{\text{LP}}}{\hat{Z}}, \tag{19}$$

where $\hat{Z}_{\text{LP}}$ is computed using equation (9) and including only the live points in the sum. The algorithm will then terminate when the condition is less than a user-defined threshold τ.

This is more suitable than the fractional change in the evidence between iterations, that is used in standard nested sampling algorithms, because multiple points are removed simultaneously at each iteration, the number of points can vary between iterations and points can leak below the current $\mathcal{L}_t$, which all mean fractional change does not decrease smoothly and instead can fluctuate significantly between iterations.

4.6. Posterior samples

Similarly to SMC and `multinest`, our algorithm returns samples $\{\theta_i\}_{i=1}^{N_{\text{Total}}}$ and their corresponding posterior weights p_i given by equation (10). Different methods can then be employed to draw posterior samples. The standard approach in nested sampling is to use rejection sampling [10] or multinomial resampling [28] to resample the nested samples using the posterior weights. Alternatively, the weights can be used directly in weighted histograms or kernel density estimates.

When using multinomial resampling or the weights directly, the posterior samples are not statistically independent, so it is informative to compute Kish's ESS [40]

$$\text{ESS} = \frac{\left[\sum_{i=1}^{N} p_i\right]^2}{\sum_{i=1}^{N} p_i^2}, \tag{20}$$

where p_i is given by equation (10). This gives an indication of the effective number of posterior samples in the posterior and allows for comparing results obtained via different sampling methods. It can also be used to diagnose poorly converged runs, since a low ESS is an indication that the samples and their corresponding weights are a poor match for the true posterior distribution.

4.7. Post-processing

Once sampling is complete, we correct for the approximation of the meta-proposal $Q(\theta)$ discussed in section 4.4 by redrawing N_{Final} samples from the meta-proposal according the draws from the multinomial distribution. The number of samples can be equal to N_{Total} or can be increased or decreased depending on the desired output.

This has the additional benefit of allowing more samples to be drawn after sampling has completed and can be used to obtain more posterior samples or decrease the estimated error on the evidence.

4.8. Complete algorithm

We can now combine all these elements into a complete algorithm which is shown in algorithm 1. The algorithm incorporates normalising flows but no longer requires that samples drawn from them be i.i.d. according to the prior. Furthermore, samples are drawn and their likelihoods evaluated in batches and all the samples are kept irrespective of their likelihood. Finally, the evidence is a simple sum, so it can be updated for batches of samples. Thus, this algorithm meets all the criteria that were initially set out.

Algorithm 1. Overview of `i-nessai`.

Input: Likelihood $\mathcal{L}$, Prior π, Tolerance τ, Method for determining N_j, N_{Final}
Output: Evidence $\hat{Z}$, samples $\{\Theta_1, \ldots, \Theta_j\}$ and posterior weights W

$j \leftarrow 1$;
$\Theta_1 \leftarrow \{\theta_i \sim \pi\}_{i=1}^{N_1}$;
$N_{\text{Total}} \leftarrow N_1, q_1 \leftarrow \pi$;
while condition $\geqslant \tau$ **do**
  $j \leftarrow j + 1$;
  $q_j \leftarrow$ trained normalising flow;
  $N_j \leftarrow$ determined via specified method;
  $\Theta_j \leftarrow \{\theta_i \sim q_j\}_{i=1}^{N_j}$;
  $N_{\text{Total}} \leftarrow N_{\text{Total}} + N_j$;
  $\hat{Z} \leftarrow \frac{1}{N_{\text{Total}}} \sum_{i=1}^{N_{\text{tot}}} \frac{\mathcal{L}(\theta_i)\pi(\theta_i)}{Q(\theta_i)}$;
  $W \leftarrow \left\{ \frac{\mathcal{L}(\theta_i)\pi(\theta_i)}{N_{\text{Total}} Q(\theta_i)} \right\}_{i=1}^{N_{\text{tot}}}$;
end
Redraw N_{Final} samples from the final meta-proposal and compute the final evidence estimate and posterior weights.

4.9. Biases

In our algorithm, the proposal distributions (normalising flows) are trained and then sampled from, rather than being constructed post sampling. This means that, unlike in `multinest`, the meta-proposal distribution is an importance sampling density and equation (11) should give a reliable estimate of the evidence error. We verify this in section 6.1.

We also note that a different bias in the evidence arises from evaluating each normalising flow with samples that were also used to train it. This is necessary since the meta-proposal requires evaluating each normalising flow on every sample. This is a side effect of the small amount of training data available to each flow and difficulty in setting the hyperparameters for N different normalising flows prior to sampling. This bias is corrected for when the samples are redrawn as described in section 4.7 which we demonstrate in section 6.

5. Related work

As described in section 2, the proposed method draws from existing variations of nested sampling: the soft likelihood constraint from diffusive nested sampling [23], the formulation of importance nested sampling used in `multinest` [25] and the use of normalising flows as described in Williams *et al* [10] and Moss [9]. However, it also has parallels to standard importance sampling and the methods derived from it.

Considering the use of a sequence of normalising flows to approximate a target (or posterior) distribution, the most closely related works are nested variational inference [41], annealed flow transport Monte Carlo [42] and preconditioned Monte Carlo [43]. The first is a hybrid between variational inference and SMC where a series of parameterised distributions are simultaneously optimised using an annealed version of the target distribution. In the latter two works, the standard SMC algorithm is modified to include an additional step that uses a normalising flow. Additionally, in Karamanis *et al* [43] the authors apply their algorithm to gravitational-wave inference, however only a single simulated event is analysed rather than a set of events.

As with any stochastic sampling algorithm for Bayesian inference, this work can also be compared to simulation-based or likelihood-free inference [33] where the posterior distribution is approximated using repeated simulations of the data instead of evaluating the likelihood. This technique has been applied to data analysis in physics and astrophysics, including but not limited to gravitational-wave data analysis [44–47], cosmology [48, 49] and particle physics [50]. The approach used in these methods involves training on a dataset that is representative of the entire parameter space and then being able to perform inference for any given point in that space. This is the opposite to the approach employed in this work, where the algorithm is general purpose and is not trained for a specific task but instead is trained on the fly, removing the need for expensive initial training at the cost of being slower when performing inference.

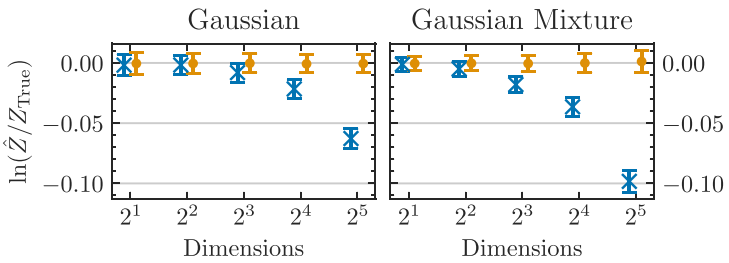

Figure 2. Mean estimated log-evidence before (blue cross) and after (orange dot) the resampling step described in section 4.7 for an n-dimensional Gaussian and Gaussian mixture. The error-bars show the mean estimated error for the log-evidence. The estimated evidence has been rescaled using the true value such that the distributions of log-evidences should be centred around zero. The number of samples drawn during the resampling step is set such that is equal to the number of samples accumulated during the initial sampling.

6. Results

We present results obtained using the algorithm described in section 4.8 on range of problems. We implement the algorithm in the `nessai` software package and it is available at [51]. To distinguish it from the version of `nessai` described in Williams *et al* [10], we will refer to it as `i-nessai`.

We run all our experiments using normalising flows based on RealNVP [36] as we find that more complex flows, such as neural spline flows [52], over-fit to the small amount of data available[1] and, compared to the other components of the algorithm, are too computationally expensive to justify using. Furthermore, `i-nessai` requires storing the normalising flow for each level so using a flow with more parameters can significantly increase the memory footprint of the algorithm.

We start with a series of tests using analytic likelihoods followed by a test using a more challenging likelihood and compare these results to those obtained with `nessai`. We then apply `i-nessai` to two different gravitational-wave analyses. Finally, we investigate parallelisation of the algorithm and how it scales with the number of live points.

For all experiments, we use the entropy-based method for constructing each proposal distribution described in section 4.1 with $\rho = 0.5$. We discuss this choice in appendix B. We also set the number of samples per flow to a constant $N_j = N_{\text{live}}$. Code to reproduce all the experiments is available at https://doi.org/10.5281/zenodo.8124198 [53]

6.1. Validation using analytic likelihoods

We start by validating `i-nessai` using likelihoods for which the evidence can be computed analytically in n dimensions. We choose to analyse the simple case of an n-dimensional Gaussian. For a more complex case, we employ the n-dimensional M-component Gaussian mixture likelihood described and used in Moss [9] and Higson *et al* [21]

$$\mathcal{L}_{\text{GM}}(\theta) = \sum_{m=1}^{M} W^{(m)} \left(2\pi\,\sigma^{(m)^2}\right)^{-n/2} \exp\left(\frac{-|\theta - \mu^{(m)}|^2}{2\sigma^{(m)^2}}\right), \tag{21}$$

where $\mu^{(m)}$ and $\sigma^{(m)}$ are the mean and standard deviation of each component in all dimensions and $\sum_{m=1}^{M} W^{(m)} = 1$. We use the same hyperparameters [9, 21]: $M = 4$, $W^{(m)} = \{0.4, 0.3, 0.2, 0.1\}$, $\mu_1^{(m)} = \{0, 0, 4, -4\}$, $\mu_2^{(m)} = \{4, -4, 0, 0\}$, $\mu_n^{(m)} = 0\ \forall\ n \in \{3, \ldots, n\}$ and $m \in \{1, \ldots, M\}$, and $\sigma^{(m)} = 1\ \forall\ m \in 1, \ldots, M$.

For both likelihoods, we consider $n = \{2, 4, 8, 16, 32\}$ and use uniform priors on $[-10, 10]^n$. The analytical log-evidence for both models is $\ln Z = -n \log 20$. We analyse each likelihood 50 times, including redrawing the samples as described in section 4.7, and examine the distribution of the log-evidence estimates and the corresponding estimated error. In figure 2, we include the result of the redrawing of the samples and recomputing the final log-evidence estimate. This shows that without redrawing the samples there is a bias in the estimated log-evidence, however this bias is small compared to the value of the log-evidence, for example, for the 32-dimensional Gaussian and Gaussian mixture the true log-evidence is −95.86 and the average biases are 0.6% and 0.9% respectively. After redrawing the samples, `i-nessai` reliably estimates the evidence

[1] A single instance of over-fitting across all the flows will not significantly impact results, however, if the flows consistently over-fit then the final result will be over-constrained.

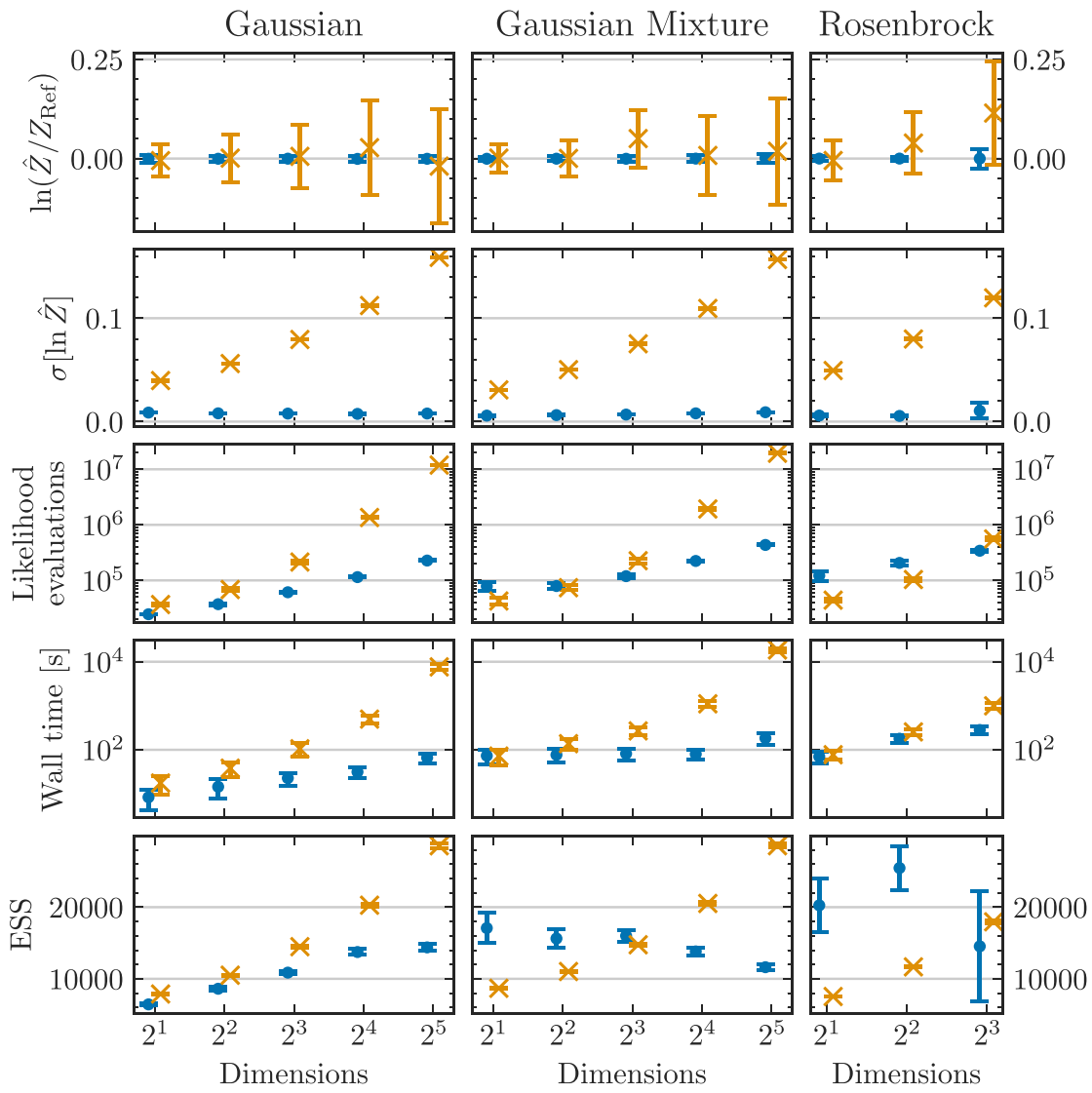

Figure 3. Comparison of results produced using `nessai` (orange) and `i-nessai` (blue) when applied to the n-dimensional Gaussian, Gaussian mixture and Rosenbrock likelihoods as described in sections 6.1 and 6.3. From top to bottom, results are shown for the final estimated log-evidence rescaled by a reference evidence (the true value for the Gaussian and Gaussian mixture and the mean value obtained with `i-nessai` for the Rosenbrock), the estimated log-evidence error, the total number of likelihood evaluations, the total wall time in seconds and the ESS of the posterior distribution. Results are averaged over 50 runs with different random seeds for both samplers and the error bars show the standard deviation.

for both models for all values of n. We also compare the distribution of the re-computed log-evidences alongside the expected distribution computed using equation (11) in appendix C and observe that the estimated log-evidence errors agree with the observed distributions.

6.2. Comparison with standard nested sampling

We now compare `i-nessai` with standard nested sampling, in particular the standard version of `nessai`. This allows us to verify the results obtained with `i-nessai`, compare the observed and estimated evidences and evidence errors, the number of likelihood evaluations, the wall time and ESS of the posterior distribution. We repeat the analyses described in section 6.1 using `nessai` and present the results for both likelihoods in figure 3.

Figure 3 shows that `i-nessai` produces estimates of the log-evidence for the Gaussian and Gaussian mixture that are consistent with `nessai` but have significantly lower variances and the corresponding estimates of the error are correspondingly smaller. We explore how the error on the log-evidence estimate scales in section 6.7. Furthermore, figure 3 shows that `i-nessai` requires a comparable number of likelihood evaluations in lower dimensions but more than an order of magnitude less in higher dimensions and a similar trend is seen with the wall time. However, this behaviour is highly dependent on the user-defined settings, which in these experiments were set based on the requirements for the

high-dimensional analyses. The ESS of the posterior distribution highlights a notable difference between the two samplers; with `nessai` the ESS increases as the number of dimensions increase for both likelihoods whereas with `i-nessai`, for the Gaussian Mixture likelihood, it decreases in higher dimensions but is still of order 10^4. Since in importance nested sampling the ESS depends on how well the meta-proposal approximates the likelihood times the prior, a lower ESS indicates a 'worse' approximation. In contrast, in standard nested sampling, and therefore `nessai`, the ESS does not depend on the convergence of the sampler and an under- or over-constrained result can still have a large ESS.

6.3. Testing on more challenging likelihoods

To further test `i-nessai`, we consider the n-dimensional Rosenbrock likelihood [54] which has highly correlated parameters and is recognised as a challenging function to sample. We use the more involved variant [55, 56] where the log-likelihood is defined as

$$\ln \mathcal{L}_{\text{Rosenbrock}}(\theta) = -\sum_{i=1}^{n-1}[100(\theta_{i+1} - \theta_i^2)^2 + (1-\theta_i)^2], \tag{22}$$

with a uniform prior on $[-5,5]^n$. We test for $n = \{2,4,8\}$ and run `i-nessai` 50 times for each n. Above $n = 2$ there is no analytical solution for the log-evidence of the Rosenbrock likelihood, so we compare results to those obtained with `nessai`. We present these results in figure 3. We observe that `i-nessai` is consistent with `nessai` for $n = 2$ but for $n = \{4,8\}$ predicts a lower evidence than `nessai`, however the relative difference is less than 1%. The number of likelihood evaluations and wall times are comparable between both samplers but `i-nessai` has a larger ESS in $n = \{2,4\}$ and lower in $n = 8$. To better understand these differences, we inspect the results obtained with `nessai` and find that the insertion indices [10, 57] are consistent with the results being over-constrained (see appendix D). This corresponds to the log-evidence being marginally over-estimated which agrees with the differences in estimated log-evidence observed in figure 3.

6.4. Probability–probability (P–P) test with binary black hole signals

As a more practical test for `i-nessai`, we repeat the analysis used to validate `nessai` in Williams *et al* [10], where we used `bilby` [4] and `nessai` to analyse simulated signals from compact binary coalescence of binary black holes injected into 4 s of data sampled at 2048 Hz in a three-detector network. For this analysis, we use the same priors (described in appendix C of Williams *et al* [10]) and enable phase, distance and time marginalisation in the likelihood. This reduces the parameter space to 12 parameters. We analyse 64 injections simulated from the same priors and produce a P–P plot and corresponding p-values using `bilby`. This analysis includes the resampling step described in section 4.7 and we re-draw the same number of samples that were used in the initial sampling, doubling the number of likelihood evaluations. The P–P plot is presented in figure 4 with individual and combined p-values. The combined p-value is 0.3798 which demonstrates that `i-nessai` reliably recovers all 12 parameters. Furthermore, these results are obtained without introducing any of reparameterisations used in standard `nessai` [10] to handle, for example, angles and spin magnitudes.

In figure 5, we show the sampling time and the number of likelihood evaluations required to reach convergence. The median number of likelihood evaluations is 6.5×10^5 and the median wall time is 119 min. We also include results obtained using `nessai` and `dynesty`[28][2], which has been used extensively for gravitational-wave inference [15–17, 58]. P–P plots for both samplers are shown in figure E1. We observe that the median reduction in the number of likelihood evaluations are 2.68 and 13.3 for `nessai` and `dynesty` respectively. These equate to reductions in the total wall time of 4.2 times and 17.2 times.

6.5. Binary neutron star analysis with reduced order quadrature (ROQ) bases

We simulate the signal from a binary neutron star merger similar to GW190425 [59] at a distance of 45 Mpc using `IMRPhenomPv2_NRtidalv2` [60] and inject it into 80 s of simulated noise from a two-detector network with aLIGO noise spectral density sensitivity [12] sampled at 8192 Hz. The resulting signal has an optimal network SNR of 30.12.

To analyse the signal, we use `IMRPhenomPv2` [61–63] with a ROQ basis [64] to reduce the cost of evaluating the likelihood[3]. We also limit the analysis to assume aligned spins and use a low-spin prior a described in Abbott *et al* [59]. We run the analysis using `i-nessai`, `nessai` and `dynesty`. We repeat each

[2] We use `dynesty` version 1.0.1 with the custom random walk implementation included in `bilby` version 1.2.1 [4, 58].
[3] We use the ROQ data available at https://git.ligo.org/lscsoft/ROQ_data.

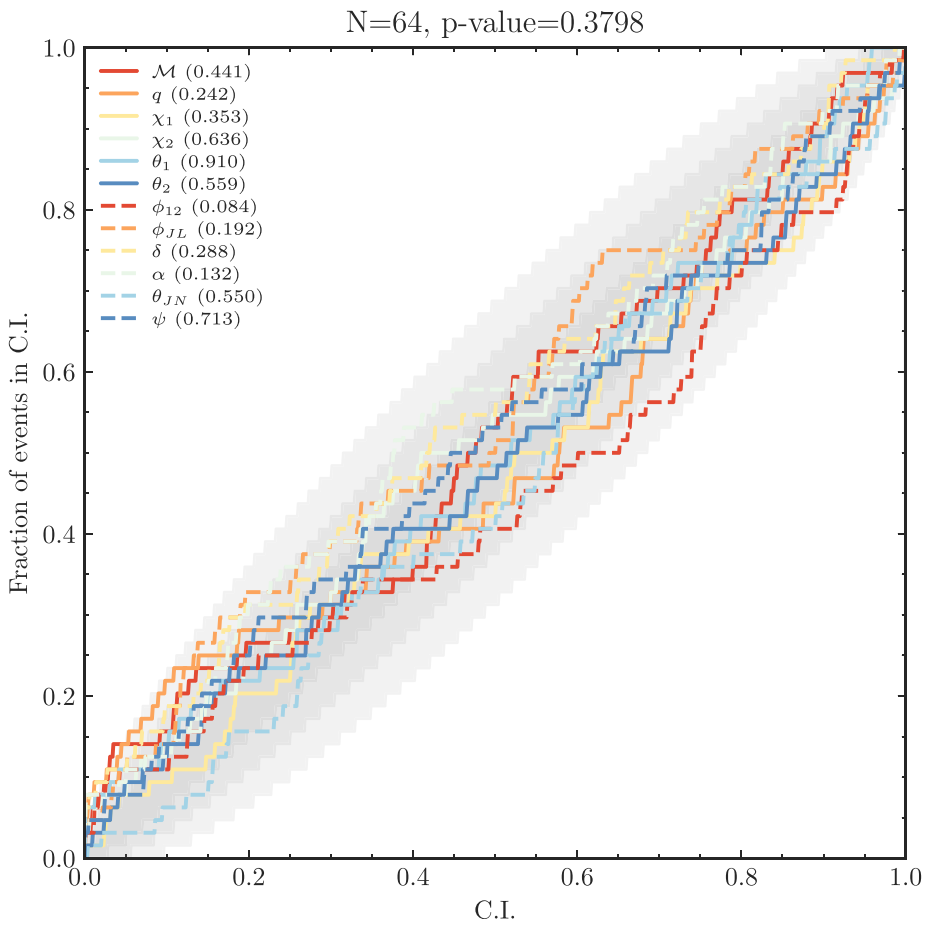

Figure 4. Probability–probability plot for 64 simulated signals from binary black hole coalescence analysed using `i-nessai`. The shaded regions indicated the 1-, 2- and 3-σ confidence intervals. Individual p-values are shown for each parameter and the combined p-value is also shown.

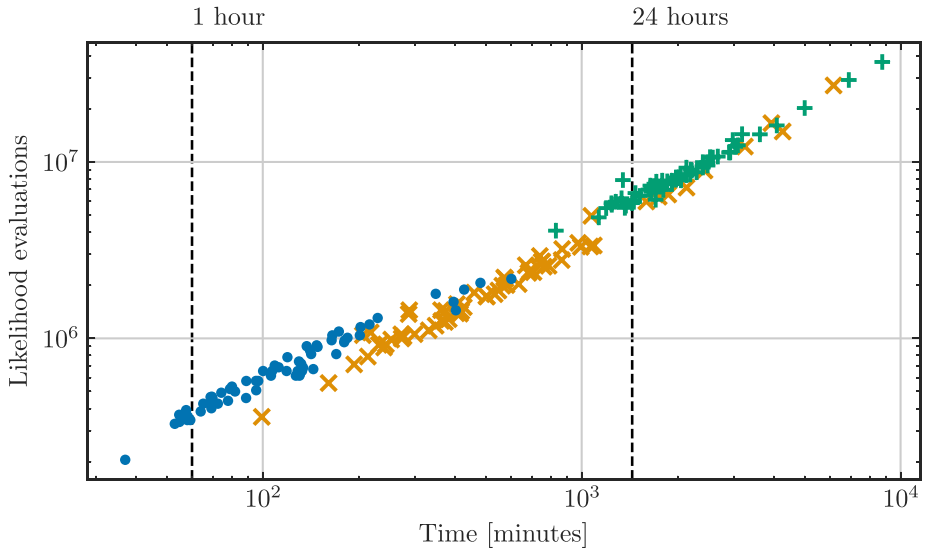

Figure 5. Total sampling time versus number of likelihood evaluations for `i-nessai` (blue dots), `nessai` (orange crosses) and `dynesty` (green plus signs) for the 64 binary black hole injections described in section 6.4.

analysis with four different random seeds and combine the posterior distributions for each seed into a single distribution. We use 16 cores for each analysis to decrease the overall wall time. The settings for `i-nessai` are tuned to ensure that the effective number of posterior samples are comparable to the other samplers.

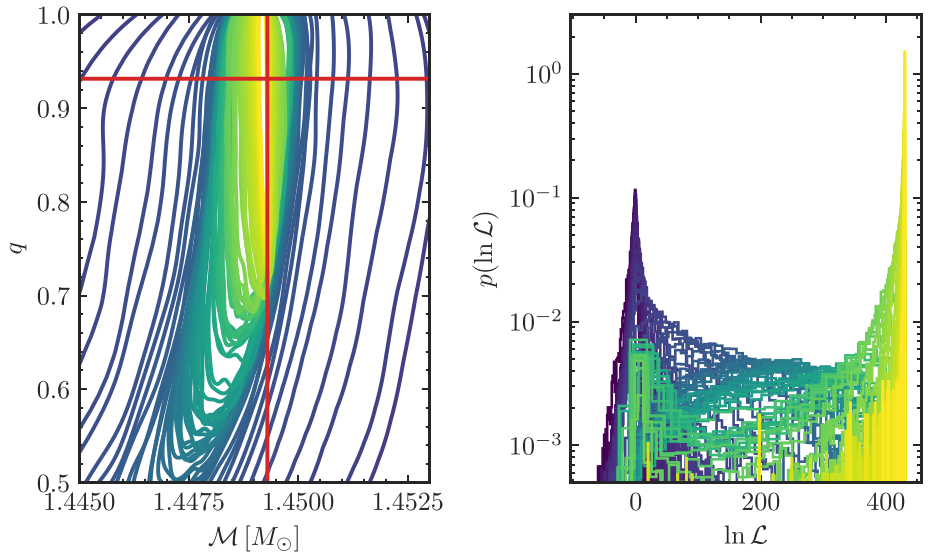

Figure 6. Evolution of the proposal distributions ($q_i(\theta)$) included in the meta-proposal when performing inference on the binary neutron star injection described in section 6.5. Brighter colours indicate later iterations in the algorithm. Left: the 90% contours for each of the proposal distributions in the chirp mass–mass ratio space. Only a small region of the parameter space around the highest likelihood is shown. The cross-hair indicates the injected value. Right: the distribution of log-likelihoods for each of the proposal distributions.

Table 1. Total likelihood evaluations, wall time in minutes and ESS of the posterior distribution for the binary neutron star analysis with ROQs as described in section 6.5 for `dynesty`, `nessai` and `i-nessai`. Results are averaged over four runs and the mean and standard deviations are quoted. All analyses were run with 16 cores.

	Wall time (min)	Likelihood evaluations	Effective sample size
`dynesty`	376.3 ± 8.1	$4.30 \times 10^7 \pm 7.12 \times 10^4$	$13\,098 \pm 131$
`nessai`	57.9 ± 8.9	$1.42 \times 10^6 \pm 1.74 \times 10^5$	$13\,036 \pm 45$
`i-nessai`	24.3 ± 3.0	$1.01 \times 10^6 \pm 8.99 \times 10^4$	$14\,625 \pm 3539$

In figure 6, we show how the meta-proposal evolves as more proposal distributions (normalising flows) are added over the course of sampling. This shows how the proposals converge around the parameters of the injected signal which correspond to the region with the highest log-likelihood.

To quantify the differences between the results, we compute the Jensen–Shannon (JS) divergence between the marginal posterior distributions for each parameter as described in Romero-Shaw *et al* [58]. We use the threshold described in Ashton and Talbot [65] to determine if the JS divergence indicate significant statistical differences between the results. We find that all the divergences are below the threshold, except for the in-plane spin χ_1, for which `i-nessai` and `nessai` agree but `dynesty` marginally disagrees with both. We include the complete set of JS divergences in appendix F and a corner plot comparing the distributions in figure G1 .

We also compare the total number of likelihood evaluations and wall time for each sampler in table 1. From these results we see that, on average, `i-nessai` requires 1.4 and 42.5 times fewer likelihood evaluations than `nessai` and `dynesty` respectively.

6.6. Parallelisation

As mentioned previously, the formulation of nested sampling used in this work does not have the same serial limitations of standard nested sampling. The algorithm we present is designed around drawing new samples and evaluating their likelihood in parallel. This leverages the inherently parallelised nature of the normalising flows. However, the process of training subsequent proposals to add to the meta-proposal is still a serial process.

In standard `nessai`, the costs of rejection sampling and training set an upper limit for the reduction in wall time that can be achieved by parallelising the likelihood evaluation. However, the total cost of training typically accounted for less than 8% of the total wall time [10]. In `i-nessai`, the rejection sampling step is no longer necessary, so the training is now the main limiting factor and the potential reduction in wall time is far greater. In figure 7, we present results showing how the wall time decreases for an increasing number of cores for one of the binary black holes injections used in section 6.4. This shows how initially the wall time is

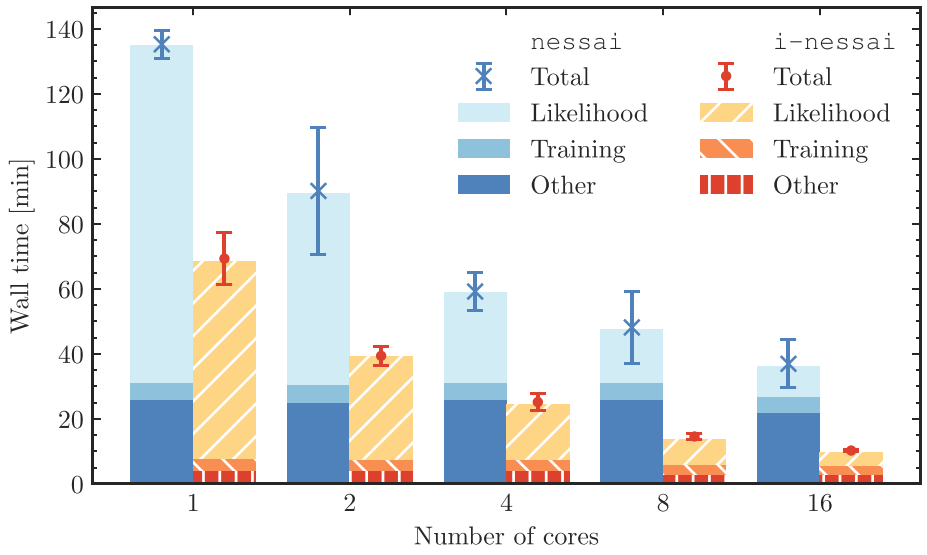

Figure 7. Comparison of the wall time spent training the normalising flows and evaluating the likelihood in `nessai` and `i-nessai` as a function of the number of cores. Results are shown for one of the binary black hole injections described in section 6.4 and are averaged over four runs.

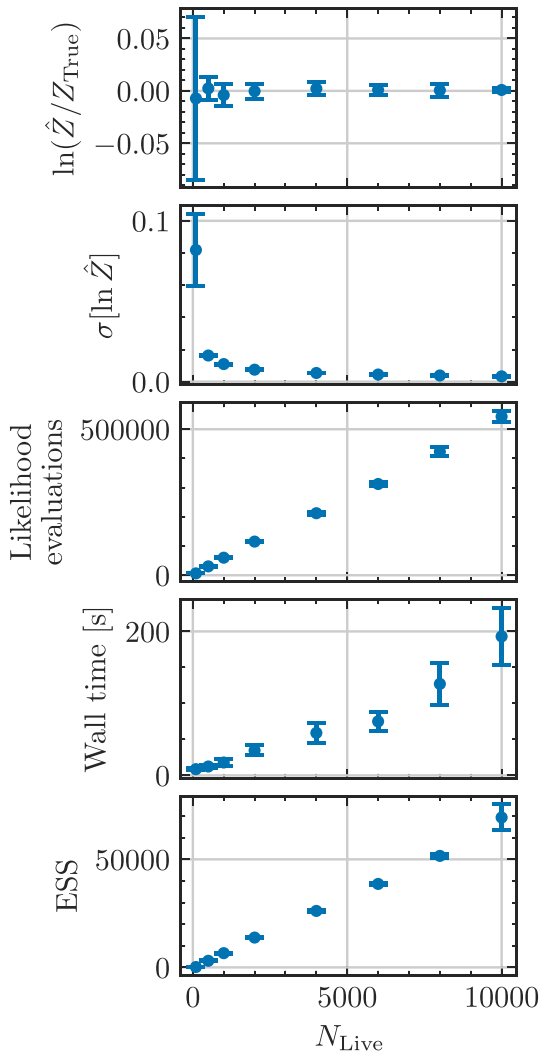

Figure 8. Scaling of `i-nessai` as a function of the number of live points N_{live} for an 16-dimensional Gaussian likelihood, as described in section 6.1. Results are averaged over 10 runs and the error-bars show the observed standard deviation. From top to bottom the results show the mean estimated log-evidence rescaled by the true value, the mean estimated standard deviation for the log-evidence, the total number of likelihood evaluations, the total wall time and the effective sample size (ESS) of the posterior distribution as defined in equation (20).

dominated by the cost of evaluating the likelihood but as more cores are added the inherent cost of sampling, which includes training the flows and drawing new samples, becomes the dominant cost. However, in this example, it only accounts for 13% of the total wall time when running on a single core.

6.7. Algorithm scaling

In `i-nessai` the number of live points has a different function to that in a typical nested sampler since, in combination with the method used to determine new levels, it will determine how many points are removed at an iteration and how many remain to train the normalising flow. We previously noted that, for `nessai`, 2000 points were needed for reliable results [10]. We now test `i-nessai` with different values of N_{live} and set the number of samples per flow $N_j = N_{\text{live}}$.

We evaluate the scaling of `i-nessai` as a function of N_{live} and present the results in figure 8 for a 16-dimensional Gaussian likelihood sampled with $N_{\text{live}} = \{100, 500, 1000, 2000, 4000, 6000, 8000, 10\,000\}$. The estimated log-evidence is consistent with the true value for all values of N_{live} and both the observed and estimated standard deviations decrease as N_{live} increases, which is consistent with equations (9) and (11). We observe that the number of likelihood evaluations scales approximately linearly with the number of live points. This contrasts with the wall time which, for a 100 times increase in the number of live points, only increases by ∼22 times. This is the result of using a likelihood that has a low computational cost, so the cost of running the sampler is dominated by the operations related to the normalising flow: training, drawing new samples and computing the meta-proposal probability as given by equation (13). In practice, most likelihoods will have a higher computational cost and the wall time will scale approximately linearly with N_{live}.

7. Discussion and conclusions

In this work, we present an importance sampling-based nested sampling algorithm, `i-nessai`, that builds on existing work [23, 25, 38] to incorporate normalising flows and overcome the main bottlenecks in `nessai` described in Williams *et al* [10]. The resulting algorithm is a hybrid between standard nested sampling and SMC, where normalising flows are successively trained and added to an overall meta-proposal that describes the distribution of samples.

We demonstrate that `i-nessai` reliably estimates the log-evidence and associated error for Gaussian and Gaussian mixture likelihoods in up to 32 dimensions. When we compare these results to those obtained with standard `nessai`, we observe that `i-nessai` converges significantly faster and requires fewer overall likelihood evaluations. Furthermore, the observed variance in the estimated log-evidence is consistently less than for `nessai`. This demonstrates that `i-nessai` produces consistent evidence estimates at a fraction of computational cost while also being more precise.

We perform inference on 64 simulated gravitational-wave signals from binary black hole coalescence using `i-nessai` and show that it passes a P–P test (figure 4) which indicates that it produces unbiased estimates of the system parameters. Furthermore, these results are obtained without introducing problem specific reparameterisations. Similarly to the analytic likelihoods, we compare these results to those obtained with `nessai` and `dynesty` and observe a median reduction in the number of likelihood evaluations of 2.68 and 13.3 times respectively, which equates to a 4.2 and 17.2 times reduction in the total wall time.

To further demonstrate the advantages of `i-nessai` compared to standard samplers, we perform inference on a simulated GW190425-like binary neutron star merger using ROQ bases [64] and aligned low-spin priors. The inference completes in just 24 min, 2.4 and 15.5 times faster than `nessai` and `dynesty` respectively, while also producing consistent posterior distributions and only requiring 1.01×10^6 likelihood evaluations compared to 1.42×10^6 and 4.30×10^7 respectively.

We also show how the likelihood evaluation can be parallelised in `i-nessai` and find that, once of the cost of evaluating the likelihood becomes negligible, training the normalising flows and drawing new samples are the main limiting factors. This is in contrast to `nessai`, where performing rejection sampling is the main limiting factor, accounting for approximately 40% of the time when running on a single core. In `i-nessai` training and drawing new samples account for significantly less of the total time. It therefore has improved scaling with respect to the number of cores compared to `nessai`, as shown in figure 7.

A downside of this approach when compared to `nessai` is that the order statistics-based tests proposed in Fowlie *et al* [57] and included in `nessai` are no longer applicable since we no longer require points be distributed according to the likelihood-constrained prior. It is therefore harder to identify under- or over-constraining in `i-nessai`. The ESS (equation (20)) can be used to diagnose issues during sampling, however it is not always a reliable diagnostic.

In future work we will consider alternative methods for constructing the meta-proposal which do not rely on discard samples, for example using only the weights in equation (18) and we will explore optimising the meta-proposal weights after sampling. We will also explore applications of `i-nessai` more complete gravitational-wave analyses like those described in [15–17] which included calibration uncertainties and waveforms with higher-order modes. Another possible application to explore is model comparison; typically, if we want to obtain a posterior distribution for a different prior than that used for the sampling, the existing posterior samples must be re-weighted using an alternative prior. However, the formulation of the nested sampling in this work would allow for the prior to be changed post-sampling and the evidence recomputed by updating equation (4), so long as the new prior does not extend the boundaries of the prior using during the initial sampling.

In summary, we have introduced an importance nested sampling algorithm, `i-nessai`, that leverages normalising flows and addresses the bottlenecks in `nessai` [10]. We have demonstrated that `i-nessai` produces results that are consistent with standard nested sampling for a range of problems, whilst requiring up to an order-of-magnitude fewer likelihood evaluations and having improved scalability. Similarly to `nessai`, `i-nessai` is a drop-in replacement for existing samplers, meaning it can easily be used to accelerate existing analyses.

Data availability statement

The data that support the findings of this study are openly available at the following URL/DOI: https://doi.org/10.5281/zenodo.8124198 [53].

Acknowledgments

The authors thank Jordan McGinn and Federico Stachurski for insightful discussions about training normalising flows with weights and Greg Ashton for providing the code for computing Jensen-Shannon divergences. The authors also thank the members of the Data Analysis Group of the Institute of Gravitational Research and the LVK Parameter Estimation group for helpful discussions. The authors thank the two anonymous referees for their suggestions, which helped improve the manuscript. The authors gratefully acknowledge the Science and Technology Facilities Council of the United Kingdom. M J W is supported by the Science and Technology Facilities Council [2285031]. J V and C M are supported by the Science and Technology Research Council [ST/V005634/1]. M J W and C M are also supported by the European Cooperation in Science and Technology (COST) action [CA17137]. The authors are grateful for computational resources provided by Cardiff University and the LIGO Laboratory, and funded by the STFC Grant [ST/I006285/1] supporting UK Involvement in the Operation of Advanced LIGO and the National Science Foundation Grants PHY-0757058 and PHY-0823459 respectively.

Software: `nessai` is implemented in `Python` and uses `NumPy` [66], `SciPy` [67], `pandas` [68, 69], `nessai-models` [70], `nflows` [71], `glasflow` [72], `PyTorch` [73], `matplotlib` [74] and `seaborn` [75]. Gravitational-wave injections were generated and analysed using `LALSuite` [76], `bilby` and `bilby_pipe` [4]. The analysis also made use of `statsmodels` [77]. Figures were prepared using `matplotlib` [74], `seaborn` [75], and `corner` [78].

Appendix A. Weighted Kullback–Leilber divergence

The KL divergence of two distributions $p(x)$ and $q(x)$ is defined as

$$\mathrm{KL}(p,q) = \int p(x) \log \left[\frac{p(x)}{q(x)}\right] \mathrm{d}x. \tag{A.1}$$

If we consider the case of minimising the KL divergence between two distributions $p(x)$ and $q(x)$ where $p(x)$ is fixed, then

$$\begin{aligned}\mathrm{KL}(p,q) = &-\int p(x)\log q(x)\mathrm{d}x + \int p(x)\log p(x)\mathrm{d}x,\\ &-\int p(x)\log q(x)\mathrm{d}x + \text{constant}.\end{aligned} \tag{A.2}$$

The constant term is independent of $q(x)$ so we only need to compute the first term when minimising the KL divergence. Using a Monte Carlo approximation of the integral with samples x drawn from $r(x)$ this becomes

$$\mathrm{KL}(p,q) \approx \widehat{\mathrm{KL}}(p,q) = -\frac{1}{N}\sum_{i=1}^{N}\frac{p(x_i)}{r(x_i)}\log q(x_i) + \text{constant}. \tag{A.3}$$

If $r \equiv p$ and this reduces to

$$\widehat{\mathrm{KL}}(p,q) = -\frac{1}{N}\sum_{i=1}^{N}\log q(x_i) + \text{constant}, \tag{A.4}$$

and we can ignore the constant when optimising $q(x)$. However, if $r \not\equiv p$ and both $p(x)$ and $r(x)$ are tractable, then we can define

$$\widehat{\mathrm{KL}}(p,q) = -\frac{1}{N}\sum_{i=1}^{N} w_i \log q(x_i) + \text{constant}, \tag{A.5}$$

where $w_i \equiv p(x_i)/r(x_i)$. This allows for the KL divergence to be minimised using samples that are not from the target distribution.

Appendix B. Methods for constructing the next proposal distribution

We test the quantile-based method and the entropy-based methods for constructing the next proposal distribution described in section 4.1 and consider the stability and number of iterations required to converge. We find that the quantile-based method for determining the next level is sensitive to outliers in the meta-proposal $Q(\theta)$. This leads to large changes in the number of discarded samples M_j between iterations which in turn can make the algorithm unstable. In contrast, the entropy-based approach is far more stable and leads to smoother variations in the number of discarded samples which we attribute to the use of the log-weights. Additionally, we find that the entropy-based method converges quicker than the quantile-based because the prior volume shrinks faster. As such, we use the entropy-based method for all our experiments.

Appendix C. Validating the variance estimator

We validate the unbiased estimator for the variance of the evidence from equation (11) for `i-nessai` using the Gaussian and Gaussian mixture likelihoods described in section 6.1. We use the results from the analyses described in section 6.1 and produce probability–probability (P–P) plots comparing the observed distribution of evidences and a Gaussian distribution with the mean equal to the true evidence and the standard deviation estimated using equation (11) averaged over the 50 runs per dimensions. The results are presented in figure C1 and show good agreement between the estimated and observed distributions.

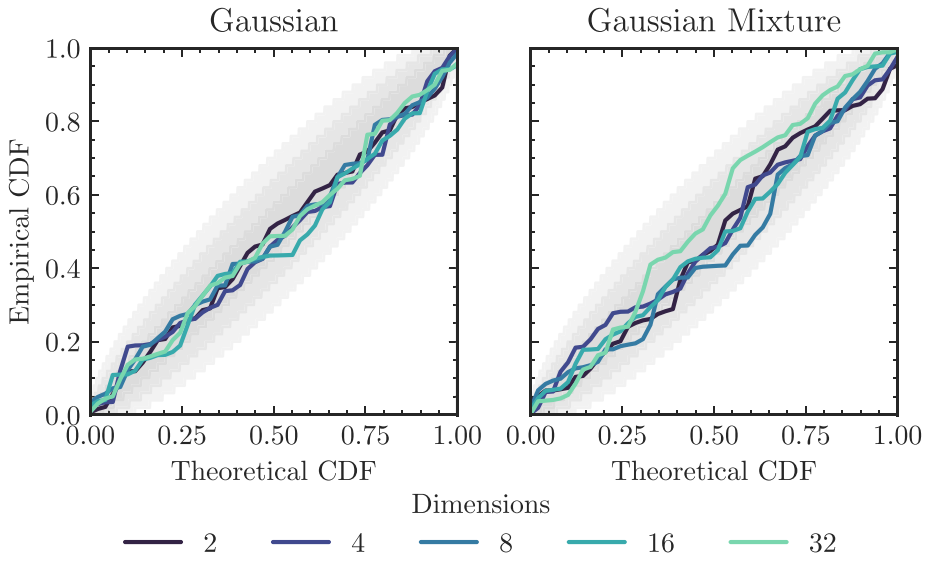

Figure C1. Probability–probability (P–P) plots for the estimated evidences for the Gaussian and Gaussian mixture models described in section 6.1 for $n = \{2, 4, 8, 16, 32\}$. The theoretical distribution is assumed to be a Gaussian centred around the true evidence with the standard deviation given by the estimated standard deviation equation (11) averaged over 50 analyses per dimension. The 1-, 2- and 3-σ confidence intervals are indicated by the shaded regions.

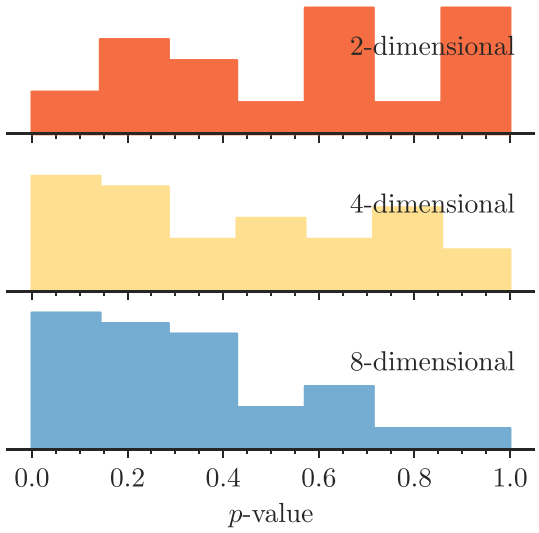

Figure D1. Distribution of the p-values for the insertion indices [57] when analysing the Rosenbrock likelihood 50 times using `nessai` with $n = \{2, 4, 8\}$.

Appendix D. Insertion indices test for the Rosenbrock likelihood

In section 6.3, we analyse the Rosenbrock likelihood for $n = \{2, 4, 8\}$ using `nessai` and `i-nessai` and find that the estimated log-evidence disagreed as shown in figure 3. In Fowlie *et al* [57], the authors proposed using order-statistics to check the convergence of nested sampling runs. This involves computing an insertion index for each new sample according to where it is inserted into the current ordered set of live points. If new samples are distributed according to the prior, then the overall distribution of the insertion indices should be uniform. This can be quantified by computing a p-value for the overall distribution using the Kolmogorov–Smirnov statistic [79] for discrete distributions [80]. We compute p-values for each analysis and presented the results in figure D1. If the results are unbiased then the distribution of p-values should be uniform on $[0, 1]$, however we observe that for $n > 2$ the distributions are not uniform, indicating problems during sampling. This agrees with the observation that for $n = \{4, 8\}$, with the settings used, `nessai` over-estimates the log-evidence.

Appendix E. Probability–probability plots for other samplers

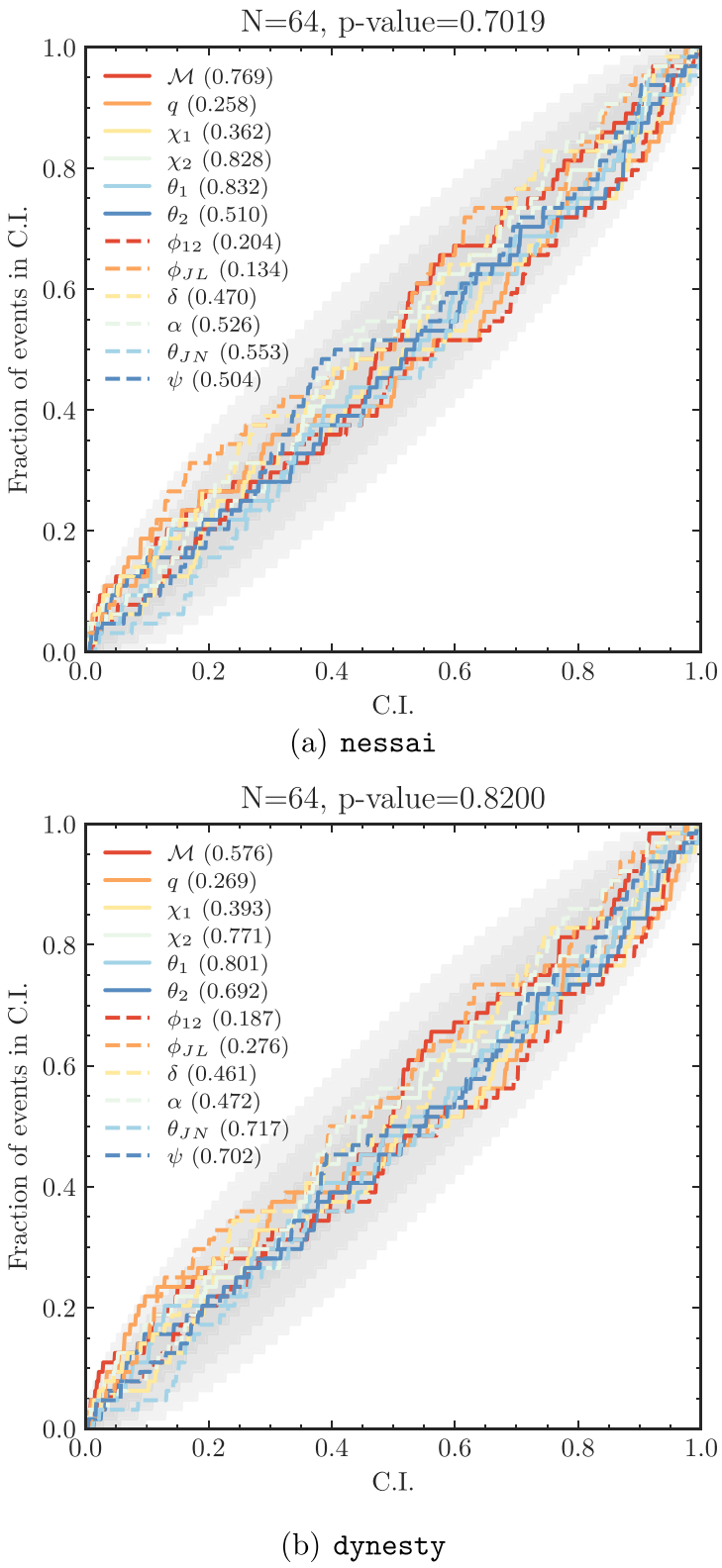

Figure E1. Probability–probability (P–P) plot showing the confidence interval versus the fraction of the events within that confidence interval for the posterior distributions obtained using `nessai` and `dynesty` for 64 simulated compact binary coalescence signals produced with `bilby` and `bilby_pipe`. The 1-, 2- and 3-σ confidence intervals are indicated by the shaded regions and p-values are shown for each of the parameters and the combined p-value is also shown.

Table F1. Jensen–Shannon divergences in units of 1×10^{-3} nats for the marginal posterior distributions between `nessai`, `i-nessai` and `dynesty`. Values shown are the mean and the 1-σ quantiles computed over 100 different realisations of 5000 samples.

	`dynesty-nessai`	`dynesty-i-nessai`	`nessai-i-nessai`
$\mathcal{M}$	$0.61^{0.20}_{-0.20}$	$0.69^{0.22}_{-0.19}$	$0.53^{0.21}_{-0.13}$
q	$0.52^{0.29}_{-0.16}$	$0.36^{0.22}_{-0.11}$	$0.30^{0.18}_{-0.08}$
χ_1	$2.24^{0.78}_{-0.55}$	$2.61^{0.77}_{-0.59}$	$0.53^{0.27}_{-0.17}$
χ_2	$1.68^{0.60}_{-0.46}$	$1.93^{0.47}_{-0.54}$	$0.73^{0.22}_{-0.22}$
δ	$1.37^{0.29}_{-0.28}$	$1.47^{0.34}_{-0.28}$	$1.59^{0.38}_{-0.31}$
α	$1.04^{0.22}_{-0.25}$	$1.15^{0.25}_{-0.27}$	$1.37^{0.30}_{-0.28}$
θ_{JN}	$0.71^{0.22}_{-0.17}$	$0.74^{0.21}_{-0.21}$	$0.79^{0.26}_{-0.23}$
ψ	$0.18^{0.13}_{-0.06}$	$0.21^{0.15}_{-0.09}$	$0.19^{0.10}_{-0.09}$
t_{c}	$1.29^{0.42}_{-0.25}$	$1.56^{0.31}_{-0.39}$	$1.57^{0.39}_{-0.33}$

Appendix F. Jensen–Shannon (JS) divergence for comparing marginal posterior distributions

We compute the JS divergence between the marginal posterior distributions obtained in section 6.5 as described in Romero-Shaw *et al* [58]. We use bootstrapping to compare 100 different realisations of 5000 samples from each posterior and quote the mean JS divergence and standard deviation in table F1. Following Ashton and Talbot [65], for 5000 posterior samples, the JS divergence threshold is 2×10^{-3} nats. The divergences between `i-nessai` and `nessai` agree for all the parameters, whereas for `dynesty` there is marginal disagreement in the posteriors for the aligned spin χ_1. However, since `nessai` and `i-nessai` are in agreement, we do not investigate this further in this work.

Appendix G. Binary neutron star corner plot

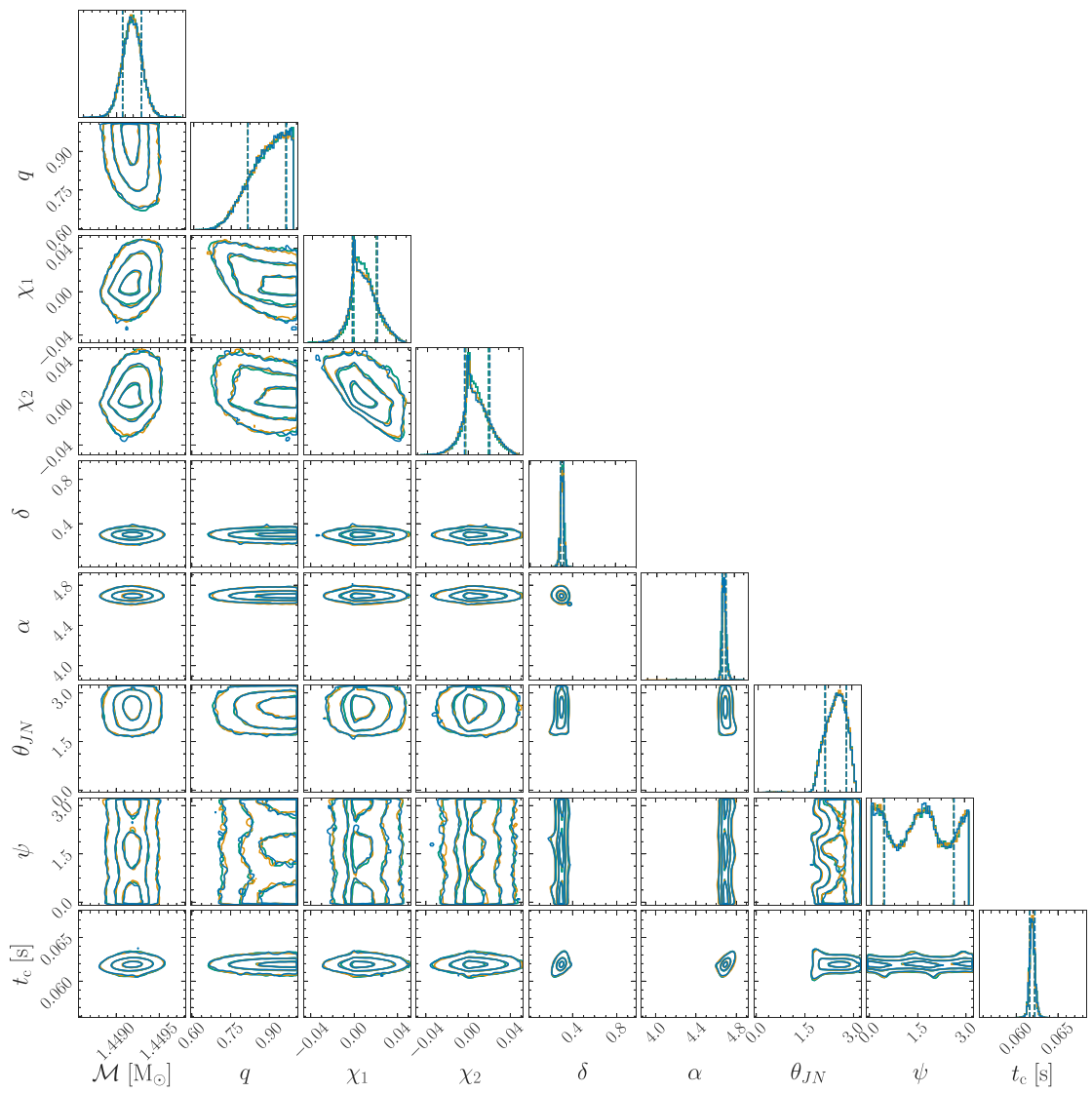

Figure G1. Posterior distributions for the GW190425-like injection described in section 6.5. Results are shown for `dynesty` in green, `nessai` in orange and `i-nessai` in blue. The 1-σ confidence intervals for each parameter are shown in the marginal histograms.

ORCID iDs

Michael J Williams https://orcid.org/0000-0003-2198-2974
John Veitch https://orcid.org/0000-0002-6508-0713
Chris Messenger https://orcid.org/0000-0001-7488-5022

References

[1] Skilling J 2004 Nested Sampling *Bayesian Inference and Maximum Entropy Methods in Science and Engineering: 24th Int. Workshop on Bayesian Inference and Maximum Entropy Methods in Science and Engineering* (*American Institute of Physics Conf. Series*) vol 735, ed R Fischer, R Preuss and U V Toussaint pp 395–405
[2] Skilling J 2006 Nested sampling for general Bayesian computation *Bayesian Anal.* **1** 833–59
[3] Veitch J *et al* 2015 Parameter estimation for compact binaries with ground-based gravitational-wave observations using the LALInference software library *Phys. Rev.* D **91** 042003
[4] Ashton G *et al* 2019 BILBY: a user-friendly Bayesian inference library for gravitational-wave astronomy *Astrophys. J. Suppl.* **241** 27
[5] Corsaro E and De Ridder J 2015 DIAMONDS: a new Bayesian nested sampling tool *European Physical Journal Web of Conferences* (*European Physical Journal Web of Conferences*) vol 101 p 06019

[6] Handley W J, Hobson M P and Lasenby A N 2015 Polychord: nested sampling for cosmology *Mon. Not. R. Astron. Soc.* **450** L61–L65
[7] Buchner J 2023 Nested sampling methods *Stat. Surv.* **17** 169–215
[8] Graff P, Feroz F, Hobson M P and Lasenby A 2012 BAMBI: blind accelerated multimodal Bayesian inference *Mon. Not. R. Astron. Soc.* **421** 169–80
[9] Moss A 2020 Accelerated Bayesian inference using deep learning *Mon. Not. R. Astron. Soc.* **496** 328–38
[10] Williams M J, Veitch J and Messenger C 2021 Nested sampling with normalizing flows for gravitational-wave inference *Phys. Rev.* D **103** 103006
[11] Alsing J and Handley W 2021 Nested sampling with any prior you like *Mon. Not. R. Astron. Soc.* **505** L95–L99
[12] Aasi J *et al* 2015 Advanced LIGO *Class. Quantum Grav.* **32** 074001
[13] Acernese F *et al* 2015 Advanced Virgo: a second-generation interferometric gravitational wave detector *Class. Quantum Grav.* **32** 024001
[14] Akutsu T *et al* 2021 Overview of KAGRA: calibration, detector characterization, physical environmental monitors and the geophysics interferometer *Prog. Theor. Exp. Phys.* **2021** 05A102
[15] Abbott R *et al* 2021 GWTC-2: compact binary coalescences observed by LIGO and Virgo during the first half of the third observing run *Phys. Rev.* X **11** 021053
[16] Abbott R *et al* 2021 GWTC-2.1: deep extended catalog of compact binary coalescences observed by LIGO and Virgo during the first half of the third observing run (arXiv:2108.01045 [gr-qc])
[17] Abbott R *et al* 2021 GWTC-3: compact binary coalescences observed by LIGO and Virgo during the second part of the third observing run (arXiv:2111.03606 [gr-qc])
[18] Abbott B P *et al* 2018 Prospects for observing and localizing gravitational-wave transients with advanced LIGO, advanced Virgo and KAGRA *Living Rev. Relativ.* **21** 3
[19] Smith R J E, Ashton G, Vajpeyi A and Talbot C 2020 Massively parallel Bayesian inference for transient gravitational-wave astronomy *Mon. Not. R. Astron. Soc.* **498** 4492–502
[20] Lange J, O'Shaughnessy R and Rizzo M 2018 Rapid and accurate parameter inference for coalescing, precessing compact binaries (arXiv:1805.10457)
[21] Higson E, Handley W, Hobson M and Lasenby A 2019 Dynamic nested sampling: an improved algorithm for parameter estimation and evidence calculation *Stat. Comput.* **29** 891–913
[22] Feroz F, Hobson M P and Bridges M 2009 MultiNest: an efficient and robust Bayesian inference tool for cosmology and particle physics *Mon. Not. R. Astron. Soc.* **398** 1601–14
[23] Brewer B J, Pártay L B and Csányi G 2009 Diffusive nested sampling (arXiv:0912.2380)
[24] Cameron E and Pettitt A 2013 Recursive pathways to marginal likelihood estimation with prior-sensitivity analysis (arXiv:1301.6450)
[25] Feroz F, Hobson M P, Cameron E and Pettitt A N 2019 Importance nested sampling and the MultiNest algorithm *Open J. Astrophys.* **2** 10
[26] Neal R M 2003 Slice sampling *Ann. Stat.* **31** 705–67
[27] Buchner J 2021 Ultranest - a robust, general purpose Bayesian inference engine *J. Open Source Softw.* **6** 3001
[28] Speagle J S 2020 DYNESTY: a dynamic nested sampling package for estimating Bayesian posteriors and evidences *Mon. Not. R. Astron. Soc.* **493** 3132–58
[29] Jimenez Rezende D and Mohamed S 2015 Variational inference with normalizing flows (arXiv:1505.05770)
[30] Dinh L, Krueger D and Bengio Y 2014 NICE: non-linear independent components estimation (arXiv:1410.8516)
[31] Kobyzev I, Prince S J D, and Brubaker M A 2019 Normalizing flows: an introduction and review of current methods (arXiv:1908.09257)
[32] Papamakarios G, Nalisnick E, Jimenez Rezende D, Mohamed S and Lakshminarayanan B 2019 Normalizing flows for probabilistic modeling and inference (arXiv:1912.02762)
[33] Cranmer K, Brehmer J and Louppe G 2020 The frontier of simulation-based inference *Proc. Nat. Acad. Sci.* **117** 30055–62
[34] Kingma D P and Welling M 2014 Auto-encoding variational Bayes *2nd Int. Conf. on Learning Representations, ICLR 2014 (Conf. Track Proc.) (Banff, AB, Canada, 14–16 April 2014)* ed Y Bengio and Y LeCun
[35] Goodfellow I J, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A C and Bengio Y 2020 Generative adversarial networks *Commun. ACM* **63** 139–44
[36] Dinh L, Sohl-Dickstein J and Bengio S 2016 Density estimation using real NVP (arXiv:1605.08803)
[37] Naesseth C A, Lindsten F and Schön T B 2019 Elements of sequential Monte Carlo (arXiv:1903.04797)
[38] Salomone R, South L F, Drovandi C C and Kroese D P 2018 Unbiased and consistent nested sampling via sequential Monte Carlo (arXiv:1805.03924)
[39] Blei D M, Kucukelbir A and McAuliffe J D 2017 Variational inference: a review for statisticians *J. Am. Stat. Assoc.* **112** 859–77
[40] Wiegand H 1968 Kish, l.: Survey sampling. Wiley, Inc., New York, London 1965, ix + 643 s., 31 abb., 56 table, preis 83 s *Biom. Z.* **10** 88–89
[41] Zimmermann H, Wu H, Esmaeili B and van de Meent J-W 2021 Nested variational inference (arXiv:2106.11302)
[42] Arbel M, Matthews A G D G and Doucet A 2021 Annealed flow transport Monte Carlo (arXiv:2102.07501)
[43] Karamanis M, Beutler F, Peacock J A, Nabergoj D and Seljak U 2022 Accelerating astronomical and cosmological inference with preconditioned Monte Carlo (arXiv:2207.05652)
[44] Gabbard H, Messenger C, Heng I S, Tonolini F and Murray-Smith R 2022 Bayesian parameter estimation using conditional variational autoencoders for gravitational-wave astronomy *Nature Phys.* **18** 112–7
[45] Chua A J K and Vallisneri M 2020 Learning Bayesian posteriors with neural networks for gravitational-wave inference *Phys. Rev. Lett.* **124** 041102
[46] Green S R, Simpson C and Gair J 2020 Gravitational-wave parameter estimation with autoregressive neural network flows *Phys. Rev.* D **102** 104057
[47] Dax M, Green S R, Gair J, Macke J H, Buonanno A and Schölkopf B 2021 Real-time gravitational wave science with neural posterior estimation *Phys. Rev. Lett.* **127** 241103
[48] Alsing J, Charnock T, Feeney S and Wandelt B 2019 Fast likelihood-free cosmology with neural density estimators and active learning *Mon. Not. R. Astron. Soc.* **488** 4440–58
[49] Jeffrey N, Alsing J and Lanusse F 2021 Likelihood-free inference with neural compression of DES SV weak lensing map statistics *Mon. Not. R. Astron. Soc.* **501** 954–69
[50] Brehmer J 2021 Simulation-based inference in particle physics *Nat. Rev. Phys.* **3** 305–305

[51] Williams M J 2021 nessai: Nested sampling with artificial intelligence (https://doi.org/10.5281/zenodo.4550693)
[52] Durkan C, Bekasov A, Murray I and Papamakarios G 2019 Neural spline flows *Advances in Neural Information Processing Systems 32: Annual Conf. on Neural Information Processing Systems 2019, NeurIPS 2019 (Vancouver, BC, Canada, 8–14 December 2019)* ed H M Wallach, H Larochelle, A Beygelzimer, F d'Alch'e-Buc, E B Fox and R Garnett pp 7509–20
[53] Williams M J 2023 mj-will/nessai-ins-paper (Zenodo) (https://doi.org/10.5281/zenodo.8124198)
[54] Rosenbrock H H 1960 An automatic method for finding the greatest or least value of a function *Comput. J.* **3** 175–84
[55] Goldberg D E and Holland J H 1988 Genetic algorithms and machine learning *Mach. Learn.* **3** 95–99
[56] Shang Y-W and Qiu Y-H 2006 A note on the extended Rosenbrock function *Evol. Comput.* **14** 119–26
[57] Fowlie A, Handley W and Su L 2020 Nested sampling cross-checks using order statistics *Mon. Not. R. Astron. Soc.* **497** 5256–63
[58] Romero-Shaw I M *et al* 2020 Bayesian inference for compact binary coalescences with BILBY: validation and application to the first LIGO–Virgo gravitational-wave transient catalogue (arXiv:2006.00714)
[59] Abbott B P *et al* 2020 GW190425: observation of a compact binary coalescence with total mass $\sim 3.4 M_{\odot}$ *Astrophys. J. Lett.* **892** L3
[60] Dietrich T, Samajdar A, Khan S, Johnson-McDaniel N K, Dudi R and Tichy W 2019 Improving the NRTidal model for binary neutron star systems *Phys. Rev.* D **100** 044003
[61] Hannam M, Schmidt P, Bohé A, Haegel L, Husa S, Ohme F, Pratten G and Pürrer M 2014 Simple model of complete precessing black-hole-binary gravitational waveforms *Phys. Rev. Lett.* **113** 151101
[62] Husa S, Khan S, Hannam M, Pürrer M, Ohme F, Forteza X J and Bohé A 2016 Frequency-domain gravitational waves from nonprecessing black-hole binaries. I. New numerical waveforms and anatomy of the signal *Phys. Rev.* D **93** 044006
[63] Khan S, Husa S, Hannam M, Ohme F, Pürrer M, Forteza X J and Bohé A 2016 Frequency-domain gravitational waves from nonprecessing black-hole binaries. II. A phenomenological model for the advanced detector era *Phys. Rev.* D **93** 044007
[64] Smith R, Field S E, Blackburn K, Haster C-J, Pürrer M, Raymond V and Schmidt P 2016 Fast and accurate inference on gravitational waves from precessing compact binaries *Phys. Rev.* D **94** 044031
[65] Ashton G and Talbot C 2021 Bilby-MCMC: an MCMC sampler for gravitational-wave inference *Mon. Not. R. Astron. Soc.* **507** 2037–51
[66] van der Walt S, Colbert S C and Varoquaux G 2011 The NumPy array: a structure for efficient numerical computation *Comput. Sci. Eng.* **13** 22–30
[67] Virtanen P *et al* 2020 SciPy 1.0: fundamental algorithms for scientific computing in Python *Nat. Methods* **17** 261–72
[68] The Pandas development team 2020 pandas-dev/pandas: Pandas (https://doi.org/10.5281/zenodo.3509134)
[69] McKinney W 2010 Data structures for statistical computing in Python *Proc. 9th Python in Science Conf.*, ed S van der Walt and J Millman pp 56–61
[70] Williams M J 2022 mj-will/nessai-models: v0.1.0 (https://doi.org/10.5281/zenodo.7105560)
[71] Durkan C, Bekasov A, Murray I, and Papamakarios G 2020 nflows: normalizing flows in PyTorch (https://doi.org/10.5281/zenodo.4296287)
[72] Mcginn J, Stachurski F, John V, and Williams M J 2022 glasflow (https://doi.org/10.5281/zenodo.7108558)
[73] Paszke A *et al* 2019 Pytorch: an imperative style, high-performance deep learning library *Advances in Neural Information Processing Systems* vol 32, ed H Wallach, H Larochelle, A Beygelzimer, F d'Alché Buc, E Fox and R Garnett (Curran Associates, Inc.) pp 8024–35
[74] Hunter J D 2007 Matplotlib: a 2D graphics environment *Comput. Sci. Eng.* **9** 90–95
[75] Waskom M and (the seaborn development team) 2020 mwaskom/seaborn (https://doi.org/10.5281/zenodo.592845)
[76] LIGO Scientific Collaboration 2018 LIGO Algorithm Library - LALSuite, free software (GPL)
[77] Seabold S and Perktold J 2010 statsmodels: econometric and statistical modeling with Python *9th Python in Science Conf.*
[78] Foreman-Mackey D 2016 corner.py: Scatterplot matrices in Python *J. Open Source Softw.* **1** 24
[79] Smirnov N 1948 Table for estimating the goodness of fit of empirical distributions *Ann. Math. Stat.* **19** 279–81
[80] Arnold T B and Emerson J W 2011 Nonparametric goodness-of-fit tests for discrete null distributions *R J.* **3** 34–39