IOP Publishing Mach. Learn.: Sci. Technol. 6 (2025) 030503 https://doi.org/10.1088/2632-2153/adf68b

MACHINE LEARNING
Science and Technology

CrossMark

OPEN ACCESS

RECEIVED
24 March 2025

REVISED
26 June 2025

ACCEPTED FOR PUBLICATION
31 July 2025

PUBLISHED
19 August 2025

BENCHMARK

Discriminative versus generative approaches to simulation-based inference

Benjamin Sluijter[1,2], Sascha Diefenbacher[2,*], Wahid Bhimji[3] and Benjamin Nachman[2,4,5]

[1] Leiden Institute of Physics, Universiteit Leiden, Leiden, RA 2300, The Netherlands
[2] Physics Division, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, United States of America
[3] National Energy Research Scientific Computing Center, Berkeley Lab, Berkeley, CA 94720, United States of America
[4] Fundamental Physics Directorate, SLAC National Accelerator Laboratory, Menlo Park, CA 94025, United States of America
[5] Department of Particle Physics and Astrophysics, Stanford University, Stanford, CA 94305, United States of America
[*] Author to whom any correspondence should be addressed.

E-mail: sdiefenbacher@lbl.gov, benjaminsluijter@gmail.com, wbhimji@lbl.gov and bpnachman@lbl.gov

Keywords: machine learning, uncertainty quantification, likelihood estimation, particle physics, high energy physics

Abstract

Most of the fundamental, emergent, and phenomenological parameters of particle and nuclear physics are determined through parametric template fits. Simulations are used to populate histograms which are then matched to data. This approach is inherently lossy, since histograms are binned and low-dimensional. Deep learning has enabled unbinned and high-dimensional parameter estimation through neural likelihood(-ratio) estimation. We compare two approaches for neural simulation-based inference (NSBI): one based on discriminative learning (classification) and one based on generative modeling. These two approaches are directly evaluated on the same datasets, with a similar level of hyperparameter optimization in both cases. In addition to a Gaussian dataset, we study NSBI using a Higgs boson dataset from the FAIR Universe Challenge. We find that both the direct likelihood and likelihood ratio estimation are able to effectively extract parameters with reasonable uncertainties. For the numerical examples and within the set of hyperparameters studied, we found that the likelihood ratio method is more accurate and/or precise. Both methods have a significant spread from the network training and would require ensembling or other mitigation strategies in practice.

1. Introduction

At the level of reconstructed particle properties, collider physics events are high dimensional. There can be many outgoing particles, each with a four-vector and other properties like electric charge. Traditionally, these complex data are analyzed by first compressing the many-dimensional phase space into a small number of high-level features and then discretized. Histograms filled with simulated data are then compared with experimental data. The parameters of the simulation that produce the best match are declared the fitted values. While highly successful, this approach under-utilizes the available information.

Modern machine learning (ML) provides an alternative approach that can process the full phase space holistically. Instead of using high-level features and histograms, neural simulation-based inference (NSBI) allows for users to perform a likelihood-based analysis without an explicit form of the likelihood [1]. There are many methods for performing NSBI. Two well-studied approaches are discriminative (classifier-based) methods and generative methods. Discriminative methods use ML tools to approximate likelihood ratios $p(x|\theta)/p(x|\theta_0)$ for features x and parameters θ. The parameter θ_0 is a constant, so maximizing the ratio is equivalent to maximizing the numerator. In contrast, generative models use ML techniques to directly approximate $p(x|\theta)$. Many approaches actually approximate $p(\theta|x)$, but we will take a frequentist approach and focus on likelihoods. There is an analogous Bayesian interpretation in terms of posteriors. We note that while NSBI can be used for a number of applications (e.g. unfolding [2–4]), we focus on the well-studied case of parameter estimation.

NSBI tools for parameter estimation (henceforth, just NSBI) have been extensively studied phenomenologically and there are a growing number of experimental measurements using these methods in collider physics [5–7] and related areas [8, 9]. Each study focuses on discriminative or generative approaches to NSBI. We are not aware of a direct comparison between these two approaches[6].

The goal of this paper is to directly compare discriminative and generative NSBI techniques on a common dataset. To benchmark this comparison, we consider the problem of Higgs boson characterization at the LHC. This is the focus of the community-wide FAIR Universe HiggsML uncertainty challenge [11], which builds on the successful HiggsML Challenge [12] by integrating uncertainty quantification. While we suspect that the optimal NSBI method is application-specific, the Higgs boson characterization case is of inherent interest and it is similar to a number of related problems so lessons learned may be useful more generally.

This paper is organized as follows. Section 2 introduces NSBI for parameter estimation. The datasets we use for comparisons are described in section 3 and the metrics used for evaluating each method are detailed in section 4. Numerical results are presented in section 5 and the paper ends with conclusions and outlook in section 6.

2. Methods

In this work, we will compare two complementary approaches to NSBI, one that uses a classifier ML model and one that leverages generative ML models. In the context of frequentist inference, the goal of any NSBI method is to implicitly of explicitly approximate the likelihood function

$$p\left(x|\mu,z\right), \tag{1}$$

where x is the set of measured data, μ is the parameter(s) of scientific interest, and z is a set of nuisance parameter(s) that affect the likelihood function, but are themselves not of interest. The goal is to maximize the likelihood over μ and z, marginalize over z, and report confidence intervals around μ given by the shape of the likelihood around the maximum.

We will focus on a common setting for particle physics: the data are a mixture model of a signal process and a background process. The parameter of interest is the fraction of signal in data. Systematic uncertainties are controlled by nuisance parameters that affect both the signal and background probability densities.

2.1. Direct likelihood estimation (DLE)

We start with directly approximating the probability density $p(x|\mu,z)$. One ML tool well-suited for this task is the normalizing flow [13]. Normalizing flows are invertible functions that implement the change of variables formula. Typically, a normalizing flow starts with a standard normal random variable and then maps it to the data space. The map is composed of a series of invertible transformations with a computationally tractable Jacobian so that one can compute the probability density of the composition. The normalizing flow is optimized by maximizing the probability density in the data space.

When a normalizing flow is trained to learn the parameters given the data, the approach is called *neural posterior estimation*. In frequentist analysis, we instead learn the probability density of the data given the parameters and then consider the result as a function of the parameters for fixed data. This can be achieved with neural posterior estimation using a uniform prior or by training a conditional normalizing flow. We use the latter approach.

It is challenging to learn the full likelihood directly because each event has so little information about the signal strength μ. Instead, we use the mixture model nature of the problem to train two normalizing flows—one for the signal and one for the background:

$$p\left(x|\mu,z\right) = \frac{\mu}{\mu+1}p_{\text{sig.}}\left(x|z\right) + \frac{1}{\mu+1}p_{\text{back}}\left(x|z\right), \tag{2}$$

where the signal and background probability densities are trained to be conditional on the nuisance parameter z.

[6] While this manuscript was being finalized, [10] appeared on the arXiv. It similarly compares these approaches, but uses different comparison metrics and features a different approach for integrating the signal-rate into the overall likelihood calculations.

2.2. Likelihood ratio estimation (LRE)

An alternative approach to approximating $p(x|\mu,z)$ is to estimate the ratio $p(x|\mu,z)/p(x|\mu_0,z_0)$ for fixed values μ_0, z_0. The reason for considering the ratio is that this converts the problem from density estimation to classification. Training a classifier to distinguish two samples is usually set up to learn the probability of the first sample. If the two samples are indexed with (μ_1,z_1) and (μ_0,z_0), this means that we can extract the likelihood ratio from the classifier $f(x) \approx \Pr(\mu_0,z_0|x)$:

$$\begin{aligned}\frac{p\left(x|\mu_1,z_1\right)}{p\left(x|\mu_0,z_0\right)} &= \frac{\Pr\left(\mu_1,z_1|x\right)p\left(x\right)/\Pr\left(\mu_1,z_1\right)}{\Pr\left(\mu_0,z_0|x\right)p\left(x\right)/\Pr\left(\mu_0,z_0\right)} \\ &\approx \frac{f(x)}{1-f(x)}\frac{\Pr\left(\mu_0,z_0\right)}{\Pr\left(\mu_1,z_1\right)}, \end{aligned} \tag{3}$$

where $p(\cdot)$ denotes a probability density and $\Pr(\cdot)$ represents a probability mass. When there are equal numbers of the two samples in training f, then $\Pr(\mu_1,z_1) = \Pr(\mu_0,z_0)$ and the likelihood ratio is approximated by $f(x)/(1-f(x))$, a fact that has been well-known in particle physics for many years [14]. Similar to the previous section, we can use the mixture model nature of the problem to break out the μ dependence:

$$\frac{p\left(x|\mu,z\right)}{p\left(x|\mu_0,z_0\right)} = \frac{\mu}{\mu+1}\frac{p_{\text{sig.}}\left(x|z\right)}{p\left(x|\mu_0,z_0\right)} + \frac{1}{\mu+1}\frac{p_{\text{back.}}\left(x|z\right)}{p\left(x|\mu_0,z_0\right)}. \tag{4}$$

The continuous likelihood ratios are approximated using parameterized classifiers [14, 15]. Instead of training a classifier with input x to distinguish samples drawn from two discrete set of parameters, a classifier is trained on (x,z) to distinguish one sample where each event has a z drawn from a distribution $p(z)$ while the other sample is generated with a fixed $z = z_0$, but then the network is presented (x,z) where z is randomly drawn also from $p(z)$. This gives

$$\frac{f(x,z)}{1-f(x,z)} \approx \frac{p\left(x,z\right)}{p\left(x|z_0\right)p\left(z\right)} = \frac{p\left(x|z\right)}{p\left(x|z_0\right)}. \tag{5}$$

which is what we need for the two likelihood ratio terms in equation (4).

2.3. Implementation

All normalizing flows and classifiers are parameterized as neural networks and implemented in PyTorch [16]. The signal and background classifier models have an identical architecture, consisting of a three-node input layer, three hidden layers with 120 nodes, and a final one-node output layer. Each layer has a standard ReLU activation function, except for the output layer, which uses a Sigmoid activation. Both flow models are conditional flows with two input and output dimensions. The conditional input consists of one dimension, which gets expanded into 32 dimensions via a fully connected embedding layer. Each model consists of four autoregressive masked piecewise rational quadratic spline [17] blocks. For each block, the spline parameters are learned by a fully connected network with two 64-node layers and ReLU activation functions. The network architecture hyperparameters for both model classes were selected based on reasonable baselines from the literature [18] and were not extensively optimized. Initial tests varying the hyperparameters of the models found little to no sensitivity to these variations, leading us to settle on the baseline. The training hyperparameters were chosen similarly, with the exception of the learning rate, which was found to have a noticeable impact on the model performance, and, as a result, was optimized using a logarithmic scan.

2.4. Inference

As all events are independent, the likelihood for a full dataset is the product of likelihoods across events [19]. We improve the numerical stability by taking the logarithm of the likelihood or likelihood ratio so that the product across events is instead a sum over events.

For the inference step, we freeze the neural network weights and only vary μ and z. We could use the differentiability of neural networks to directly optimize for μ, z with the network weights fixed. For this paper, we do a simple grid search, first profiling over z, and then performing a fourth-order polynomial to interpolate between grid points. This works well in the low-dimensional examples that follow, but the automatic differentiation approach is likely required when more parameters are included. Confidence intervals are constructed by identifying where the log likelihood decreases by 0.5 from the maximum.

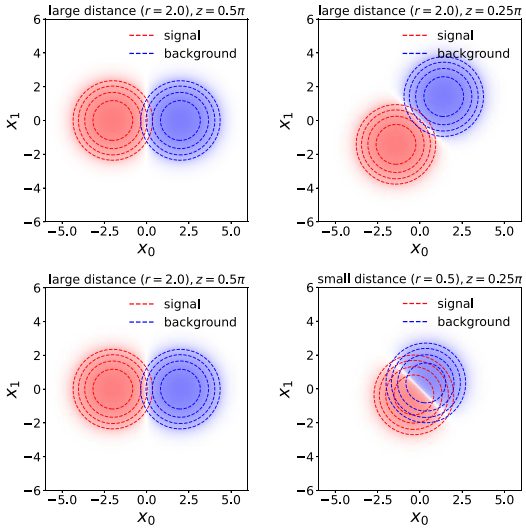

Figure 1. Visualization of the Gaussian data for rotation values $z = 0.5$ (left-hand side) and $z = 0.25\pi$ (right-hand side), as well as for the large distance case (upper panels) and the small distance case (lower panels).

3. Data sets

We compare DLE and LRE using two datasets—one based on Gaussians and one built from a collider physics example.

3.1. Gaussian umple

Our preliminary tests are performed on a synthetic dataset consisting of two, two-dimensional Gaussians. One of the Gaussians is designated as 'signal', while the other is designated as 'background'. The distance between the Gaussian mean and the origin is given by the factor r. As a result, the separation between the two means is $2r$. We distinguish between two cases:

- the long-distance case with $r = 2.0$
- the short-distance case with $r = 0.5$.

Since the difficulty of classifying a given event as either signal or background is directly related to the overlap between the Gaussians, and therefore r, this enables us to test the model performance on both simple and challenging classification tasks. We additionally introduce a systematic nuisance parameter z, which corresponds to the rotation of the means of the Gaussians around the origin [20]. The full definition of the dataset is given by:

$$X_{\text{back.}} \sim (\mathcal{N}(\cos(z)\, r, 1), \quad \mathcal{N}(\sin(z)\, r, 1)) \tag{6}$$

$$X_{\text{sig.}} \sim (\mathcal{N}(-\cos(z)\, r, 1), \quad \mathcal{N}(-\sin(z)\, r, 1)) \tag{7}$$

where $\sim \mathcal{N}(\alpha, \beta)$ represents a random variable that is normally distributed with mean α and variance β. The Gaussian data are illustrated in figure 1 with four combinations of r and z values.

A training set for the Gaussian data consists of one million events, evenly split between signal and background. For each point, a nuisance value was uniformly sampled from $z \in [0, 0.5\pi]$. To facilitate the training of the classifier models, an additional training set of the same size was generated, with a fixed nuisance value of $z_0 = 0.25\pi$, to act as a reference for the likelihood ratio calculation. Additional validation sets were directly generated during the model training. A test set for evaluation consists of 9000 total data points with fixed μ and z values.

The main reason to study the Gaussian data is that the true likelihood function is known, so we can compare the learned likelihood (ratio) with the true values. For the physics case presented in the next section, the true likelihood (ratio) is not known.

3.2. Higgs boson example

The Higgs boson was the last particle of the standard model (SM) to be discovered [21, 22]. While its properties are highly constrained within the context of the SM, the Higgs boson also plays a central role in

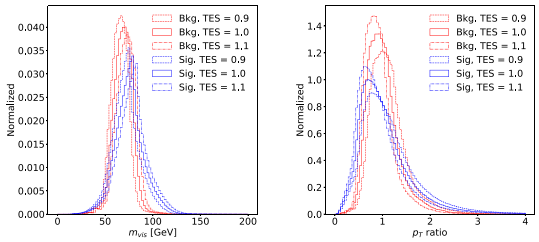

Figure 2. Visualization of the features used for the Higgs data. The left-hand plot shows the invariant mass of the hadronic τ and non-τ lepton, the right-hand plot shows the p_T ratio of the hadronic τ and non-τ lepton. The solid lines show the features for the nominal TES value, while the dashed and dash-dotted lines show them for smaller and larger TES values, respectively.

many theories of physics beyond the SM (BSM). In such BSM theories, the Higgs properties vary from the SM ones and so searching for deviations is a central task in collider physics. Many papers have been written on the use of ML for studying Higgs boson properties and a community data challenge on this subject attracted attention from researchers outside of particle physics [12]. Now that the existence of the Higgs boson is well-established, there is a need to focus on precision and so the FAIR Universe HiggsML Uncertainty project was started to extend the earlier data challenge by bringing in uncertainty quantification [11]. We use a version of the FAIR Universe HiggsML Uncertainty dataset in this paper, which is briefly described below.

The simulated data consist of semi-leptonic $H \to \tau\tau$ signal events, and a similarly semi-leptonic $Z \to \tau\tau$ background. The data were simulated using Pythia 8.2 [23], and Delphes 3.5.0 [24] for the detector simulation.

As both signal and background events feature a hadronically decaying τ in their final state, this dataset is highly sensitive to the energy calibration scale used for the τ jets (tau energy scale or TES). This makes the TES a well-suited nuisance parameter to test the ability of the ML models to handle a real systematic effect. The effects of the TES are not included in the initial simulation but are modeled ad-hoc during the model training and evaluation. To this end, we multiply the energy of the Tau jet by the TES and then recalculate any observables affected by this change. Both the classifier and flow models receive the TES as a conditional input.

To reduce the computational cost associated with the many pseudoexperiments we will perform later, we select two observables to use in the model training. Specifically, we chose the invariant mass of the hadronic tau and the non-τ lepton (m_{vis}), and the ratio of transverse momenta (p_T) between the hadronic tau and the non-τ lepton. Both of these observables are different for signal and background and are highly sensitive to the TES, making them ideal candidates to test the systematic effects. Histograms of the two features are illustrated in figure 2.

Unlike the Gaussian data, the Higgs data requires dedicated simulation software to generate data points. Therefore, we are limited to the existing simulation data and cannot simply generate more points on demand. As a result, we allocate the available data in the following way:

- train set: 7 million events total, divided into 2.9 million background and 4.1 million signal events,
- validation set: 400 000 events total, split evenly between signal and background,
- test sets: 6.02 million events total, with 6 million background and 22 500 signal events.

The class split of the test set emulates the small signal-to-background ratio found in realistic physics data as all events have unit weight, while the more even split in the training set is in line with the common practice to over-sample the signal case in particle physics simulation. The class imbalance in the training set only affects equation (3) as a constant factor, and therefore does not affect the log likelihood optimization.

4. Evaluation

4.1. Uncertainty quantification performance

Unlike standard classification tasks, we are not interested in the raw performance of a given model. Instead, we want confidence intervals to be as small as possible while also correctly representing the spread expected if we re-ran the experiment many times. In order to emulate multiple experiments, each time we would evaluate a model on a test set, we instead evaluate the model multiple times on bootstrapped re-samplings of the test set [25]. For N bootstrap samplings, this results in N confidence intervals. Since each confidence

interval represents 1σ, we expect that about 68.2% of the intervals will contain the true value. Therefore, we calculate the fraction of bootstrap evaluations for which the known μ and z values of the test set lie within the predicted interval. We refer to this percentage as the *coverage*. Since we also want precision, a second metric is the average size of the 1σ intervals, with smaller sizes representing better performance.

For the Gaussian dataset, we introduce another metric because the true likelihood is known. The overlap metric[7] o is defined as

$$o = 1 - \frac{1}{2}\sum_{\theta\in\Theta} \left|\hat{p}_{\text{norm}}\left(\{x\}\,|\theta\right) - p_{\text{norm}}\left(\{x\}\,|\theta\right)\right|, \tag{8}$$

where Θ is a grid over either μ, z, or (μ, z) (if only one, the other is marginalized over), $\hat{p}$ is the estimated likelihood (ratio), p is the true likelihood, $p(\{x\}|\theta) = e^{-\text{LL}}$ for log likelihood (ratio) LL over all events up to a constant independent of θ, and $p_{\text{norm}}(\{x\}|\theta) = p(\{x\}|\theta)\left(\sum_{\theta'\in\Theta} p(\{x\}|\theta')\right)^{-1}$. The normalization allows the likelihood ratio methods to be directly comparable to the likelihood methods, since during the calculation of $p_{\text{ratio,norm}}$

$$p_{\text{ratio,norm}} = \frac{p\left(\{x\}\,|\theta\right)}{p\left(\{x\}\,|\theta_0\right)}\left(\sum_{\theta'\in\Theta}\frac{p\left(\{x\}\,|\theta'\right)}{p\left(\{x\}\,|\theta_0\right)}\right)^{-1} \tag{9}$$

$$= p\left(\{x\}\,|\theta\right)\left(\sum_{\theta'\in\Theta} p\left(\{x\}\,|\theta'\right)\right)^{-1}, \tag{10}$$

the denominator cancels, as it does not contain the variable θ over which the sum is performed. This gives us a metric between zero and one, which allows us to directly gauge how well the model-predicted likelihood agrees with the true likelihood of the Gaussian data set.

4.2. Evaluation process

Since the proposed coverage metric is only applicable in aggregate, we require the method to be evaluated on many evaluation sets to be able to quantify the spread of predicted intervals. To this end, we make use of data splitting and bootstrapping. While the natural signal-to-background ratio is much smaller, we consider $\mu \sim \mathcal{O}(10\%)$ in order to facilitate faster training to more readily perform the full battery of coverage tests. The 22 500 signal events in the test set are split into five disjoint sets. For each set, we have a total of 9000 events, which is approximately the maximum allowable for the highest signal fraction we scan (30%) and fixing the number of background events. From each set of 9000, we re-sample with replacement 1000 points (explained in section 4.3)). For each 9000 point set, we draw a total of 100 bootstrapped datasets. A given model is then evaluated on these bootstrapped sets, and the resulting confidence intervals are used to calculate the coverage metric. The computational bottleneck is from the optimization and confidence interval finding for a given dataset due to the fine scan in μ and z. It may be possible to accelerate this with gradient descent and approximate confidence intervals based on the Fisher matrix. The number of bootstrapped datasets is not limiting because we evaluate the network on all data points before (sub)sampling. Due to the averaging across disjoint sets and across parameter values (described below), we did not find a significant improvement in precision by creating more pseudodatasets.

Additionally, we require multiple 9000 point starting evaluation sets, generated with different values for μ and z in order to ensure the robustness of a given method under changes to these values. Therefore, we define a grid in μ, z with $\mu \in [0.1, 0.2, 0.3]$ for both datasets and $z \in [0.15\pi, 0.25\pi, 0.35\pi]$ or $z \in [0.93, 1.00, 1.07]$ for the Gaussian and Higgs data, respectively. For each of these 9 grid points, we create a test set, either by generating a new point in the Gaussian case, or by selecting an appropriate number of signal and background events from the set-aside test set in the Higgs case.

The entire bootstrapping process is repeated on each of the five disjoint sets, resulting in 45 evaluation sets. For each bootstrapped dataset, we train N models ($N = 30$ for the Gaussian and $N = 40$ for the Higgs). In summary: for each of the 45 evaluation sets and each of the N models, we calculate the relevant metrics over the 100 bootstrapped datasets and average for each of the N models. In practice, one could make the method more precise by ensembling the N models. We study their spread as a way to illustrate the robustness of the method to random network initializations. Even though this spread can be reduced through averaging,

[7] This metric was chosen as it is constrained between zero and one and linked to more common metrics like a KLD by a monotonic factor.

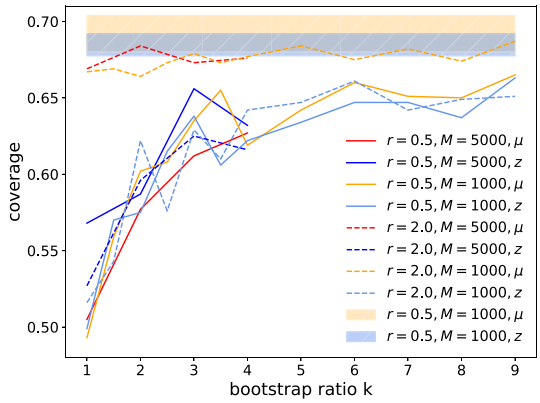

Figure 3. Coverage values as a function of the ratio k between the size of an evaluation set and the size of the bootstrapped subset (N). Inference is performed on either μ or z as indicated in the legend. Results are shown for both the small-distance ($r = 0.5$) and large-distance ($r = 2$) Gaussian datasets. Shaded regions represent results of the same test using completely resampled sets instead of bootstrapped sets. Therefore, shaded regions are independent of k.

it does add to the computational complexity of the setup. It may be also be possible in the future to mitigate this spread through more extensive hyperparameter optimization, including using more robust optimizers [26].

4.3. Bootstrapping evaluation

For a given set of parameters, the previous section described how confidence intervals are obtained using bootstrapped datasets. We use the simplest estimate of the 68% confidence interval from these estimates – $\hat{\theta}_{(84)} - \hat{\theta}_{(16)}$, where $\hat{\theta}$ is the predicted value of the parameter θ, and the subscript with parentheses denotes the order statistics from the 100 bootstrapped pseudodatasets. There are other, more accurate estimates [27] that it would be interesting to study in future work. All of these estimates have the property that they converge to the true confidence interval as the number of original samples grows to infinity. In the finite dataset limit, the confidence interval estimates can be biased. We found that one way to reduce the bias is to sample M events from kM events ($k > 1$) with replacement instead of sampling M from M. This section briefly explores how big to make k.

For this study, we use the Gaussian dataset, using the true Gaussian likelihoods for inference, to remove potential biases introduced from the ML modeling. The results are shown in figure 3. The coverage from 100 bootstrapped datasets is shown as a function of k, for $M \in \{1000, 5000\}$, $\theta \in \{\mu, z\}$, and for the small ($r = 0.5$) and large ($r = 2$) distance configurations from section 3.1.

There does not seem to be a noticeable difference in the behavior between subsets with size 1000 and subsets with size 5000, as the corresponding pairs of lines overlap within their fluctuations. As a result, cases with subsets-size 5000 and large values of $k > 4$ were left out, as testing them would have required significant computational cost without likely contributing additional insight. Only the large distance Gaussian case for inferring μ reaches the expected coverage of about 68% for all values of k. Meanwhile, all coverages for the small distance case, as well as all results involving z, require a comparatively large $k > 5$ to approach the expected coverage. For this reason, we settled on using a subset size of 1000 and $k = 9$ in our evaluation chain. A residual bias may still exist, which may be responsible for small variations about from 68% coverage in the final results.

As an additional check, we repeated these experiments without bootstrapping, using 100 independently sampled datasets instead of 100 bootstrapped sets. The results of this are shown as the shaded regions in figure 3. The size of the shaded region corresponds to the standard deviation we observed from repeating this experiment multiple times. We can see that all shaded regions group around a coverage of 68%. This is even the case for variable combinations ($[r = 0.5, M = 1000, \mu]$, $[r = 0.5, M = 1000, z]$) that do not achieve 68% when using bootstrapping. We can therefore conclude that the imperfect coverage is indeed a result of the bootstrapping and not of the confidence interval construction.

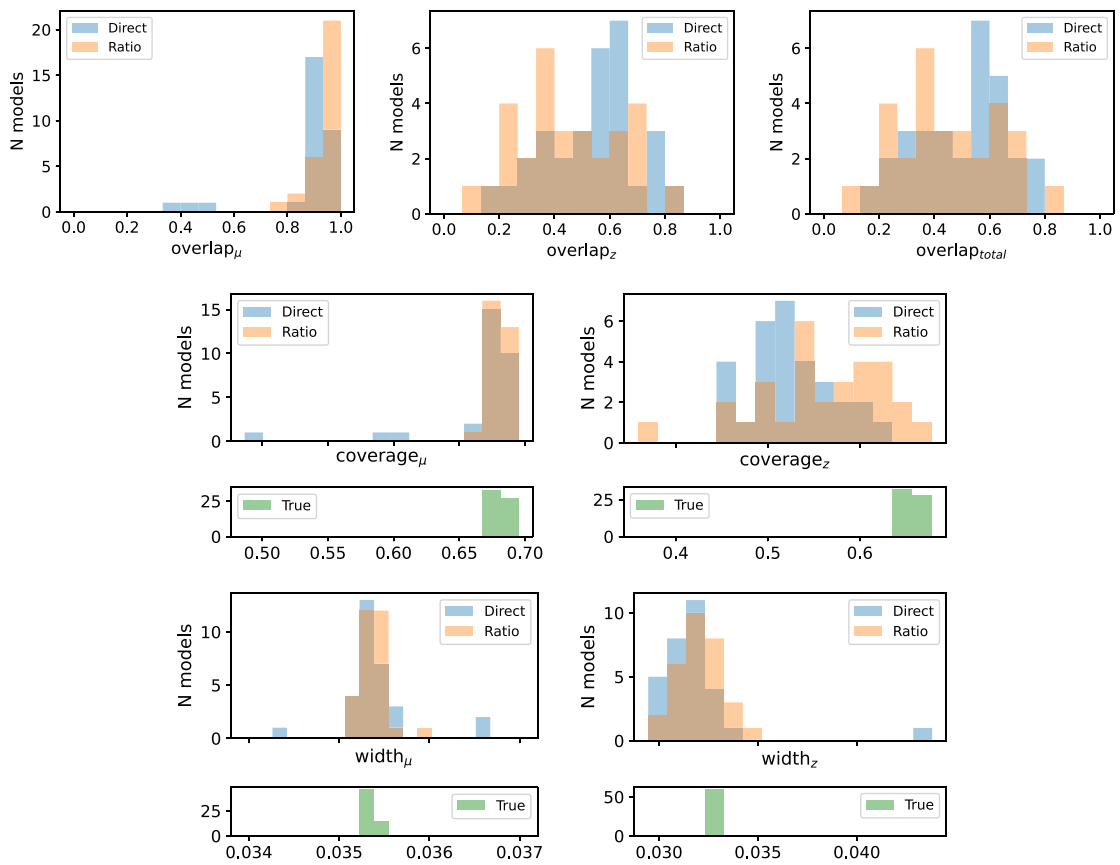

Figure 4. Gaussian long distance case results for the DLE (learning rate $= 3 \times 10^{-6}$) and LRE model (learning rate $= 3 \times 10^{-5}$). Histograms show the performance of 30 independently trained models of both the LRE model and the DLE model. For the coverage and mean width results, the plots are split into an upper section for the model results and a lower section for the ground truth model, indicating approximately what the histograms should look like if all 30 models were the perfect model.

5. Results

5.1. Gaussian example

For both the small and large distance Gaussian data sets, $N = 30$ DLE and LRE models were trained and evaluated as described in section 4. The results of this for the large distance case are shown in figure 4. The three top-row panels correspond to the overlap metrics between the model-predicted likelihoods and the true Gaussian likelihood. The overlap in the μ likelihood is shown in the right-hand panel, the overlap in z is shown in the center panel, and the left-hand panel shows the combined overlap in μ and z. The overlap for μ is narrowly peaked around 1.0 for both the DLE and LRE models, with only a small number of outliers for both models, indicating that both models perform adequately for determining μ. The z overlap is clustered around 0.5 and notably broader. This indicates that extracting the precise profile likelihood in z is a more challenging task compared to μ, which is in line with intuitive expectations for the large distance case. As a result, the combined overlap is largely dominated by the mismatches in z, and is therefore nearly identical to the z overlap. The center and lower rows show the coverage and width metrics, respectively, split into the results for μ in the left-hand column and the results for z in the right-hand column. Each plot shows the distribution of metric values for the N models in the top section and the equivalent results obtained using the true Gaussian likelihood in the bottom section. This allows us to determine if any deviation in the coverage can be linked to the bootstrapping effects investigated in section 4 or if it is caused directly by the model performance. The results of the coverage are in line with what was observed in the overlap; for μ, we see excellent coverage, in line with what is obtained using the true likelihood, while there are notable mismatches in z. Finally, the predicted interval widths of the models are in line with the true likelihood, indicating that for this dataset, both models perform very well in determining μ. It should be noted that the widths obtained by the models in z do not deviate significantly from the widths obtained from the truth.

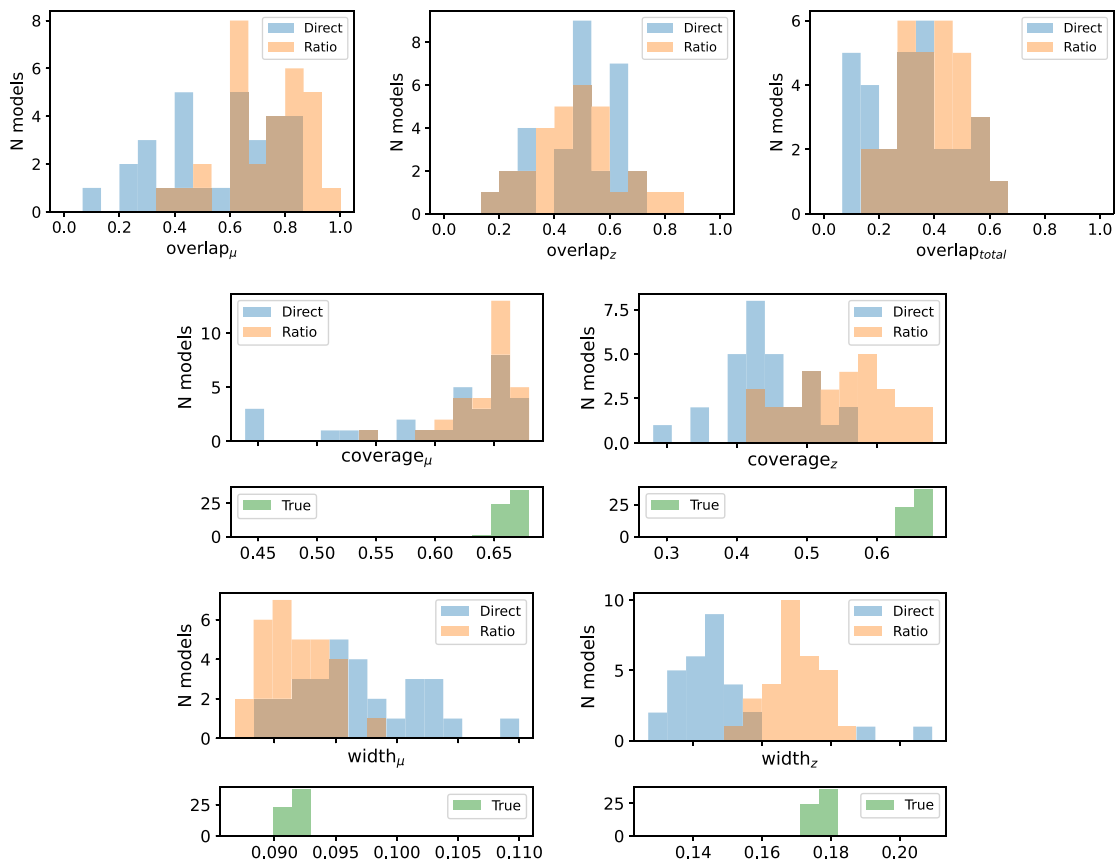

Figure 5. Gaussian small distance case results for the DLE (learning rate $= 10^{-5}$) and LRE model (learning rate $= 3 \times 10^{-5}$). Histograms show the performance of 30 independently trained models of both the LRE model and the DLE model. For the coverage and mean width results, the plots are split into an upper section for the model results and a lower section for the ground truth model, indicating approximately what the histograms should look like if all 30 models were the perfect model.

This indicates that the imperfect coverage result of the ML approaches is likely not caused by underestimating the uncertainty but by failing to capture the correct predictions for z.

Figure 5 shows the same evaluation performed on the small distance Gaussian dataset. Notably, the small distance between the two Gaussian peaks makes the task of classifying signal and background significantly more challenging than in the large distance case. This effect can clearly be seen in both the μ overlap and μ coverage, both of which show significantly larger deviations from the correct values than what was observed in the small distance case. The smaller distance between peaks also makes it more challenging to determine their relative rotation z, which is visible in the z coverage. Here, we also see a notable difference between the DLE model and LRE model. While both models have imperfect coverage, the DLE model performs notably worse, having an average coverage of only around 0.45. Looking at the width, we can further see that the DLE model predicts a z width significantly below the width derived from the true likelihood. This indicates that the DLE model systematically underestimates the uncertainty, resulting in the incorrect coverage. The performance discrepancy between the DLE and LRE models in determining z is challenging to explain, but can likely be traced down to the DLE model having to learn z implicitly via the conditional input in an otherwise generative training. Importantly, this difference between the z performance of the DLE and LRE models is not apparent from the overlap and can only be detected with the coverage metric.

Lastly, we can also examine the contours of the estimated likelihood in comparison to the true values for various combinations of μ and z as shown in figure 6. There is good agreement between the likelihood contours for the large distance case, while the small distance data set shows notable differences between model prediction and truth. This is in line with the observations from the previous figures. Overall, both the DLE model and the LRE model perform comparably on the large distance case. For the more challenging small distance Gaussians, the LRE model has the same coverage but better width compared to the DLE model in μ, and displays a better coverage than the DLE model in z. While only a single set of examples (with a single set of hyperparameters), this leads to the hypothesis that the LRE approach is more sensitive than the DLE approach. This hypothesis will be tested in the collider physics example in the next section.

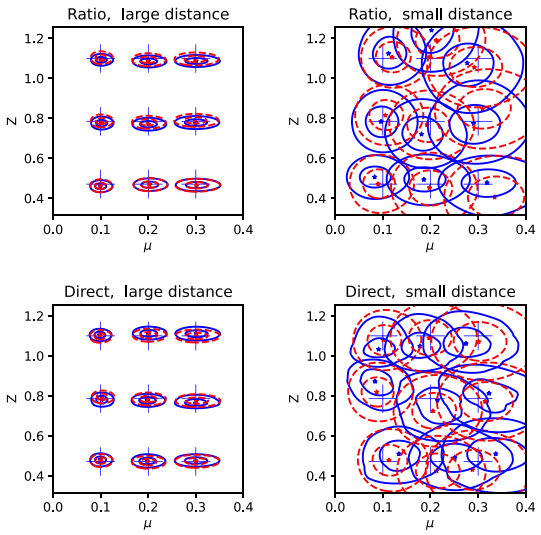

Figure 6. Examples of contours in the Gaussian case. The upper row shows results for the LRE approach, while the lower row shows results for the DLE approach. The results in the left column were obtained using the large-distance Gaussian data, and the results using the right column were trained using the small-distance Gaussian data. Metric averages over the test sets in each figure can be found in the appendix in table 1.

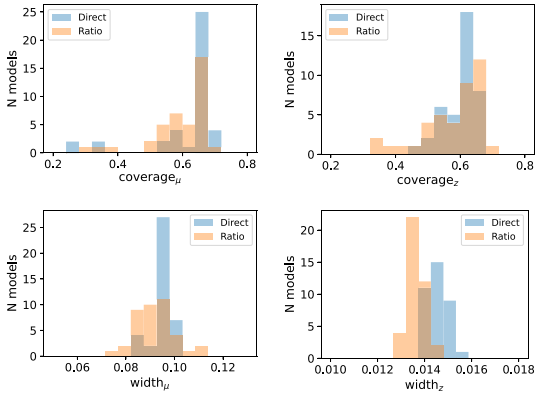

Figure 7. Higgs case results. Histograms show the average performance of 40 independently trained models of both the LRE model and DLE model.

5.2. Higgs example

We perform an identical analysis with the Higgs data, as was done with the Gaussian data, except that the number of trained models was increased to $N = 40$. Figure 7 shows the same results that were discussed in the Gaussian case, except without any overlap metrics or the true likelihood results, as these would require access to the underlying likelihood of the Higgs data, which is not available. Nevertheless, the coverage metric results shown in the upper row show that both models have a μ coverage peaked around 0.675, and a z coverage peaked around 0.625, which correspond to the correct coverage for μ, and a slightly suboptimal coverage for z. It should be noted, however, that without access to true likelihood results, it is not possible to disentangle whether the z coverage results are purely caused by poor model performance, or by an effect of the bootstrapping. Further, the width plots on the bottom row allow us to differentiate the performance of LRE and DLE models, where the DLE model has an, on average, larger width than the LRE model in both μ and z, even though the classifier displays a larger spread in μ width for different model trainings. These findings are consistent with the hypothesis from the Gaussian case, and it would be interesting to see how other examples compare with these two.

Figure 8 shows the contours for individual models. Since we do not have a true contour as a benchmark, we instead illustrate two examples. *Positive example* shows models with good coverage scores, while *Negative Example* shows the contours of models with less optimal coverages. From this, we can see that there appears to be a correlation between the coverage score and the ability of a model to determine the correct values for μ

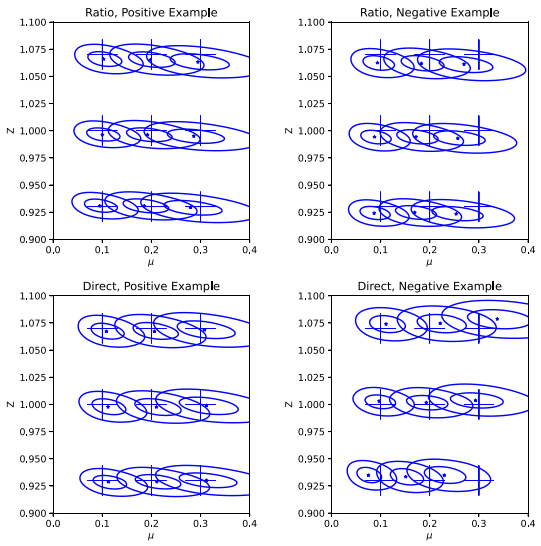

Figure 8. Examples of contours for models trained on the Higgs data. *Positive example* shows well-performing models, while *negative example* shows models with lower performance. Metric averages over the test sets in each figure can be found in the appendix in table 2.

and z. This further demonstrates the usefulness of the coverage as a performance metric for uncertainty-aware models.

6. Conclusions and outlook

We have directly compared direct LRE and LRE for neural simulation-based parameter inference. Our motivation for this study was the growing number of studies focusing on only one of these two approaches. We found that both methods were able to effectively extract parameters with uncertainties. On the specific examples we studied and with the hyperparameters selected, the likelihood-ratio method was more accurate as it received better coverage results, indicating a more accurate prediction of the measured quantity. The ratio approach further appeared to be more precise in the examined examples, as indicated by the smaller width, in cases with correct coverage. and/or precise. However, we found that the results are quite sensitive to the setup, including the data, network architecture/training, and inference evaluation protocol. Additionally, the differences between the two approaches became less pronounced for the realistic physics data. Extensive scans are computationally expensive, making even the basic task of hyperparameter optimization challenging. While these issues make the comparison between approaches difficult, they also highlight the overall difficulty of optimizing an NSBI setup. It will be interesting to see if the trends observed in this paper persist with higher-dimensional feature and parameter spaces and with advanced regularization techniques including ensembling, gradient-based parameter inference, and calibration [14].

Data availability statement

The data that support the findings of this study are openly available at the following URL/DOI: https://www.codabench.org/competitions/2164/.

Acknowledgments

BN and SD are supported by the U.S. Department of Energy (DOE), Office of Science under contract DE-AC02-05CH11231. BS is supported by the Leiden University International Study Fund. We thank Dorothea Samtleben for detailed feedback on the study and for many useful discussions with the FAIR Universe team, including Ragansu Chakkappai, Po-Wen Chang, Yuan-Tang Chou, Jordan Dudley, Steven Farrell, Aishik Ghosh, Isabelle Guyon, Chris Harris, Shih-Chieh Hsu, Elham E Khoda, David Rousseau, Ihsan Ullah, and Yulei Zhang.

Code and data

The software for this paper can be found on Github. A slightly updated version of the Higgs dataset is available at this Codabench competition, and the most recent version of the uncertainty-aware Higgs classification challenge data set can be found here.

Appendix. Numerical results of example contours

In this appendix, we present the coverage, mean width, and overlap values for the models that correspond to the contours shown in figures 6 and 8. The reported values are averaged over the 9 test sets shown in the contour plots.

Table 1. Coverage, mean width and overlap values of the models on the Gaussian data, for which the contours were shown in figure 6. The table is separated into the large distance case and the small distance case. Further, the table is subdivided into the LRE and DLE approaches.

	Large Distance		Small Distance	
Gaussian	LRE	DLE	LRE	DLE
Overlap_μ	0.94	0.95	0.69	0.55
Overlap_z	0.52	0.47	0.47	0.41
$\text{Overlap}_{\text{tot}}$	0.52	0.47	0.41	0.29
Coverage_μ	0.66	0.65	0.64	0.61
Coverage_z	0.60	0.52	0.63	0.50
Width_μ	0.036	0.035	0.089	0.096
Width_z	0.031	0.032	0.17	0.15

Table 2. Coverage and mean width values of the models on the Higgs data, for which the contours were shown in figure 8. The table is separated into the models that agree well with the truth (left) and the instances where the models agree less well (right). Further, the table is subdivided into the LRE and DLE approaches.

	High Agreement		Low Agreement	
Higgs data	LRE	DLE	LRE	DLE
Coverage_μ	0.67	0.68	0.60	0.56
Coverage_z	0.68	0.66	0.48	0.59
Width_μ	0.098	0.095	0.085	0.087
Width_z	0.014	0.014	0.013	0.014

ORCID iDs

Benjamin Sluijter 0009-0009-6830-9016
Sascha Diefenbacher 0000-0003-4308-6804
Wahid Bhimji 0000-0002-6213-8617
Benjamin Nachman 0000-0003-1024-0932

References

[1] Cranmer K, Brehmer J and Louppe G 2020 The frontier of simulation-based inference *Proc. Natl Acad. Sci.* **117** 30055–62
[2] Arratia M *et al* 2021 Presenting unbinned differential cross section results (arXiv:2109.13243)
[3] Butter A *et al* 2023 Machine learning and LHC event generation, *SciPost Phys.* **14** 079
[4] Huetsch N *et al* 2024 The landscape of unfolding with machine learning (arXiv:2404.18807)
[5] Aad G *et al* (ATLAS Collaboration) 2024 Measurement of off-shell Higgs boson production in the $H^* \to ZZ \to 4\ell$ decay channel using a neural simulation-based inference technique in 13 TeV *pp* collisions with the ATLAS detector (arXiv:2412.01548)
[6] Aad G *et al* (ATLAS Collaboration) 2024 An implementation of neural simulation-based inference for parameter estimation in ATLAS (arXiv:2412.01600)
[7] Chekhovsky V *et al* (CMS Collaboration) 2024 Constraints on standard model effective field theory for a Higgs boson produced in association with W or Z bosons in the H→ $b\bar{b}$ decay channel in proton-proton collisions at $\sqrt{s} = 13$ TeV (arXiv:2411.16907)
[8] Hermans J, Banik N, Weniger C, Bertone G and Louppe G 2021 Towards constraining warm dark matter with stellar streams through neural simulation-based inference *Mon. Not. R. Astron. Soc.* **507** 1999–2011
[9] Dax M, Green S R, Gair J, Pürrer M, Wildberger J, Macke J H, Buonanno A and Schölkopf B 2023 Neural importance sampling for rapid and reliable gravitational-wave inference *Phys. Rev. Lett.* **130** 171403

[10] Chatterjee A, Choudhury A, Mitra S, Mondal A and Mondal S 2025 Exploring the BSM parameter space with neural network aided simulation-based inference
[11] Bhimji W *et al* 2024 FAIR universe HiggsML uncertainty challenge competition (arXiv:2410.02867)
[12] Adam-Bourdarios C, Cowan G, Germain C, Guyon I, Kégl B and Rousseau D 2015 The Higgs boson machine learning challenge *Proc. NIPS 2014 Workshop on High-Energy Physics and Machine Learning (Montreal, Canada)* (*Proc. Machine Learning Research* vol 42), ed G Cowan, C Germain, I Guyon, B Kégl and D Rousseau (PMLR) pp 19–55
[13] Papamakarios G, Nalisnick E, Rezende D J, Mohamed S and Lakshminarayanan B 2021 Normalizing flows for probabilistic modeling and inference
[14] Cranmer K, Pavez J and Louppe G 2015 Approximating likelihood ratios with calibrated discriminative classifiers (arXiv:1506.02169)
[15] Baldi P, Cranmer K, Faucett T, Sadowski P and Whiteson D 2016 Parameterized neural networks for high-energy physics *Eur. Phys. J.* **C76** 235
[16] Paszke A *et al* 2019 Pytorch: an imperative style, high-performance deep learning library *Advances in Neural Information Processing Systems* (vol 32) (Curran Associates, Inc) pp 8024–35
[17] Durkan C, Bekasov A, Murray I and Papamakarios G 2019 Neural spline flows *Advances in Neural Information Processing Systems* vol 32
[18] Raine J A, Klein S, Sengupta D and Golling T 2022 CURTAINs for your sliding window: constructing unobserved regions by transforming adjacent intervals (arXiv:2203.09470)
[19] Nachman B and Thaler J 2021 Learning from many collider events at once *Phys. Rev. D* **103** 11 116013
[20] Ghosh A, Nachman B and Whiteson D 2021 Uncertainty-aware machine learning for high energy physics *Phys. Rev. D* **104** 056026
[21] Aad G *et al* (ATLAS Collaboration) 2012 Observation of a new particle in the search for the standard model Higgs boson with the ATLAS detector at the LHC *Phys. Lett. B* **716** 1–29
[22] Chatrchyan S *et al* (CMS Collaboration) 2012 Observation of a new boson at a mass of 125 GeV with the CMS experiment at the LHC *Phys. Lett. B* **716** 30–61
[23] Sjöstrand T, Ask S, Christiansen J R, Corke R, Desai N, Ilten P, Mrenna S, Prestel S, Rasmussen C O and Skands P Z 2015 An introduction to PYTHIA 8.2 *Comput. Phys. Commun.* **191** 159–77
[24] de Favereau J, Delaere C, Demin P, Giammanco A, Lemaitre V, Mertens A and Selvaggi M (DELPHES 3 Collaboration DELPHES 3) 2014 A modular framework for fast simulation of a generic collider experiment *J. High Energy Phys.* JHEP02(2014)057
[25] Efron B 1979 Bootstrap methods: another look at the jackknife *Ann. Stat.* **7** 1–26
[26] De Luca G B, Nachman B, Silverstein E and Zheng H 2025 Optimizers for stabilizing likelihood-free inference (arXiv:2501.18419)
[27] Efron B and Tibshirani R 1986 Bootstrap methods for standard errors, confidence intervals and other measures of statistical accuracy *Stat. Sci.* **1** 54–75