MACHINE LEARNING
Science and Technology

PURPOSE-LED PUBLISHING™ Part of the MACHINE LEARNING Series

PAPER • **OPEN ACCESS**

Simulation-based inference with deep ensembles: evaluating calibration uncertainty and detecting model misspecification

To cite this article: James Alvey *et al* 2026 *Mach. Learn.: Sci. Technol.* **7** 015008

View the article online for updates and enhancements.

You may also like

- Calibrated and uncertain? Evaluating uncertainty estimates in binary classification models
Aurora Grefsrud, Trygve Buanes and Nello Blaser
- Amortized simulation-based frequentist inference for tractable and intractable likelihoods
Ali Al Kadhim, Harrison B Prosper and Olivia F Prosper
- Model Dispersion with prism: An Alternative to MCMC for Rapid Analysis of Models
Ellert van der Velden, Alan R. Duffy, Darren Croton et al.

This content was downloaded from IP address 91.201.160.78 on 20/05/2026 at 09:07

IOP Publishing Mach. Learn.: Sci. Technol. 7 (2026) 015008 https://doi.org/10.1088/2632-2153/ae3103

MACHINE LEARNING
Science and Technology

Check for updates

PAPER

Simulation-based inference with deep ensembles: evaluating calibration uncertainty and detecting model misspecification

James Alvey[1,2,*], Carlo R Contaldi[3] and Mauro Pieroni[4,5]

[1] Kavli Institute for Cosmology Cambridge, Madingley Road, Cambridge CB3 0HA, United Kingdom
[2] Institute of Astronomy, University of Cambridge, Madingley Road, Cambridge CB3 0HA, United Kingdom
[3] Department of Physics, Imperial College London, London SW7 2AZ United Kingdom
[4] Theoretical Physics Department, CERN, 1211 Geneva 23, Switzerland
[5] Instituto de Estructura de la Materia (IEM), CSIC, Serrano 121, 28006 Madrid, Spain
[*] Author to whom any correspondence should be addressed.

E-mail: jbga2@cam.ac.uk, c.contaldi@imperial.ac.uk and mauro.pieroni@csic.es

Keywords: simulation-based inference, uncertainty quantification, deep ensembles, model diagnostics, Bayesian inference, model misspecification

OPEN ACCESS

RECEIVED
10 September 2025

REVISED
5 November 2025

ACCEPTED FOR PUBLICATION
24 December 2025

PUBLISHED
12 January 2026

Original content from this work may be used under the terms of the Creative Commons Attribution 4.0 licence.

Any further distribution of this work must maintain attribution to the author(s) and the title of the work, journal citation and DOI.

Abstract

Simulation-based inference (SBI) offers a principled and flexible framework for conducting Bayesian inference in any situation where forward simulations are feasible. However, validating the accuracy and reliability of the inferred posteriors remains a persistent challenge. In this work, we point out a simple diagnostic approach rooted in ensemble learning methods to assess the internal consistency of SBI outputs that does not require access to the true posterior. By training multiple neural estimators under identical conditions and evaluating their pairwise Kullback–Leibler (KL) divergences, we define a consistency criterion that quantifies agreement across the ensemble. We highlight two core use cases for this framework: (a) for generating a robust estimate of the systematic uncertainty in parameter reconstruction associated with the training procedure, and (b) for detecting possible model misspecification when using trained estimators on real data. We also demonstrate the relationship between significant KL divergences and issues such as insufficient convergence due to, e.g. too low a simulation budget, or intrinsic variance in the training process. Overall, this ensemble-based diagnostic framework provides a lightweight, scalable, and model-agnostic tool for enhancing the trustworthiness of SBI in scientific applications.

1. Introduction

Simulation-based inference (SBI) has emerged as a powerful framework to perform Bayesian inference in diverse scientific domains (see, e.g. [1] for a review). In the broadest terms, the range of SBI algorithms, such as neural posterior estimation (NPE) [2–4], neural ratio estimation (NRE) [5, 6], or neural likelihood estimation (NLE) [7] (as well as their sequential analogues, see, e.g. [8–10]), all leverage machine learning (ML) techniques to approximate various statistical quantities relevant for Bayesian inference. All these approaches are based on the basic ability to sample parameters θ from some prior $p(\theta)$ and subsequently generate simulated data $x \sim p(x|\theta)$.

Despite the increasing sophistication and adoption of SBI methods, a fundamental challenge persists: how do we assess the quality and reliability of the inferred posterior distribution $q(\theta|x_{\rm obs}) \approx p(\theta|x_{\rm obs})$ for a specific observed dataset $x_{\rm obs}$? Since the true posterior is (at least in the most well-motivated use cases) often unknown and/or analytically intractable, direct comparison is rarely possible. Without robust validation methods, it remains challenging to determine whether an SBI result is accurate, whether the algorithm has converged sufficiently, or whether the resulting posterior approximation accurately captures the true uncertainty about the parameters. This lack of ground-truth comparison hinders the confident application and interpretation of SBI results in scientific discovery.

Consequently, there is a clear and pressing need for a suite of reliable diagnostics and validation techniques specifically tailored for SBI outputs. Several valuable approaches have already been developed

© 2026 The Author(s). Published by IOP Publishing Ltd

to address this need. Simulation-based calibration (SBC) [11–13] provides a necessary condition for correctness by checking whether, on average, the ranks of the true parameters under the inferred posteriors are uniformly distributed over many simulated datasets. Probability–probability (PP) plots are often used to assess this calibration. More recently, techniques like tests of accuracy with random points (TARPs) [14] have been introduced to quantify the concentration and calibration of posteriors by measuring the coverage probability of credible regions. Standard Bayesian techniques, such as posterior predictive checks, can also be adapted. Broadly, this involves comparing simulations generated from the synthetic model to the observed data to check for model misspecification or goodness-of-fit. A variety of methods have been developed to address this challenge, including approaches that learn statistics to distinguish between data from different models [15–20], adaptations of Bayesian model comparison and evidence estimation [21–23], deviations in different SBI algorithms [24–26], and diagnostics tailored to specific scientific applications [27, 28]. In general, while these methods are extremely valuable, they often assess average performance over many simulations (like SBC), or rely on visual inspection (like PP-plots and some PPCs), and may not fully characterise the uncertainty or potential biases associated with the posterior approximation for a single, specific observed dataset $x_{\rm obs}$.

In this paper, we propose fully leveraging ensemble learning principles [29–31] to assess the quality of an SBI posterior approximation. Our core idea is conceptually extremely straightforward: if an SBI algorithm, trained on a finite set of simulations, were capable of reliably converging to the exact, true posterior distribution, then multiple instances of the same algorithm, trained independently on the same simulation budget or subsets thereof, should produce posterior estimates that agree perfectly with each other. Qualitatively, then, significant disagreement between an ensemble of trained estimators would signal potential issues such as high variance due to limited simulation data, poor convergence of the training process (e.g. neural network optimisation), instability in the approximation architecture itself, or data mismodelling when analysing real data.

The main idea of this work is to perform quality checks by computing and comparing discrepancy measures between the posterior estimates produced by each member of the ensemble. In particular, we highlight how the Kullback–Leibler (KL) divergence [32] can be efficiently computed in an SBI context and provides a robust and interpretable quantity for comparison. This concept draws a clear analogy with diagnostic techniques used in Markov Chain Monte Carlo (MCMC) methods. In MCMC, multiple chains are often run in parallel from dispersed starting points. The convergence of these chains is assessed using diagnostics such as the Gelman-Rubin statistic ($\hat{R}$) [33], which compares the variance within individual chains to the variance between chains. High $\hat{R}$ values track differences between the chains and indicate a lack of convergence, suggesting the samples do not yet represent the target posterior. Similarly, we propose below using the consistency (or lack thereof) across an ensemble of SBI estimators as one diagnostic for the reliability of the resulting posterior approximation.

We note that, of course, ensemble learning itself is a well-established paradigm in ML, where multiple individual models are trained and their predictions combined or compared to achieve better performance, robustness, and uncertainty quantification than any single model could provide (see, e.g. [34] for a review). Common techniques in this regard include bagging, boosting, and stacking. By training an ensemble of SBI posterior estimators, we can not only assess convergence and stability but also potentially obtain more robust posterior estimates by combining the ensemble members, for instance, through mixture models or averaging, and gain insights into the approximation uncertainty. Ensemble methods have been explored before in the SBI context. For example, in [7], ensembles of likelihood estimators were used in an active learning framework to optimally generate new simulations with the acquisition protocol aimed at reducing likelihood uncertainty across the ensemble. In addition, [13] explored the coverage properties of ensembles, and [24] demonstrated the behaviour of different SBI algorithms under various levels of model misspecification. More closely related to the current work, [35] proposed the general principle of using deep ensemble disagreement to diagnose sensitivities arising from both insufficient training convergence and model misspecification. Our work provides a thorough validation and in-depth analysis of this diagnostic approach using a KL-divergence-based disagreement metric.

The rest of the paper is structured as follows. In section 2 we establish the theoretical background for our approach and discuss the interpretation of the KL divergence in terms of systematic uncertainty estimation for the posteriors. Section 3 then demonstrates the practical application of these methods for assessing the convergence and reliability of SBI posteriors. We will demonstrate how ensembles can be used for generating a robust estimate of the systematic uncertainty in parameter reconstruction associated with the training procedure. Moreover, we show how the difference between the various networks in the ensemble can be used as a diagnostic tool to identify model misspecification. Finally, in section 4, we summarise our results and conclude.

2. Ensemble learning for SBI

Building upon the need for robust validation methods in SBI, this section develops the theoretical underpinnings of using ensemble methods in this context. We begin by very briefly reviewing the core concepts of ensemble learning within the broader context of ML. Subsequently, we discuss how these principles can be applied generally across different SBI methodologies. We then introduce a quantitative metric, the KL divergence matrix, designed to measure the consistency among different posterior estimators within an ensemble trained for a specific SBI task. This provides a means to assess the reliability of the inference results for a given observation. Finally, we explain how this metric can be used to place a lower bound on the systematic errors induced by the training and inference processes in SBI.

2.1. Principles of ensemble learning

Ensemble learning constitutes a powerful paradigm in ML in which multiple individual models, often referred to as base learners or ensemble members, are constructed and their predictions are combined to produce a final result [36, 37]. The fundamental principle motivating this approach is that aggregating predictions from a diverse set of reasonably accurate models can lead to enhanced generalisation performance, increased robustness against noise or model misspecification, and more reliable uncertainty quantification compared to relying on any single constituent model.

Several techniques are common in ensemble learning to design the ensemble and aggregate the results. Bagging (or bootstrap aggregating) consists of training independent learners on different bootstrap samples (random samples drawn with replacement) of the original training dataset. Aggregating the predictions can lead to a reduced variance, which stabilises the predictions [38]. Boosting consists of training a set of models sequentially, such that each subsequent model focuses on correcting the errors made by the models trained earlier in the sequence [39, 40]. Boosting primarily aims to reduce the bias component of the error. Finally, stacking involves training multiple, often heterogeneous, base learners, with a separate meta-learner trained to optimally combine the predictions of these base learners [41].

Beyond improving aggregate predictive accuracy, ensembles inherently provide a mechanism for assessing model stability and uncertainty. Disagreement among ensemble members trained under nominally similar conditions (for example, using the same algorithm and data budget but different random seeds or bootstrap samples) can denote high variance in the learning process or sensitivity to specific training instances [42]. Although ensemble techniques have seen application within SBI, primarily aimed at improving the accuracy or robustness of the resulting posterior approximation [7, 13, 43], their systematic utilisation as a *diagnostic* tool to gauge the reliability and convergence of the inference procedure for a *single* specific observed dataset remains less explored. Addressing this problem is the central motivation for the present work.

2.2. SBI and ensemble applications

SBI addresses the fundamental challenge of performing Bayesian inference when the likelihood function $p(x|\theta)$, which quantifies the probability of observing data x given parameters θ, is intractable or computationally prohibitive to evaluate directly/sample efficiently [1].

SBI methods bypass this limitation by using a simulator, or forward model, capable of generating synthetic data $x \sim p(x|\theta)$ for any given parameter vector θ sampled from the prior distribution $p(\theta)$. The overarching goal of SBI methods is to approximate the posterior distribution $p(\theta|x_{\text{obs}})$ for a specific, observed dataset x_{obs}. A variety of SBI algorithms/strategies have been developed for this purpose, including: NPE, which directly targets the posterior distribution; NLE, which instead estimates the likelihood and subsequently samples it; and NRE, which finally constructs a neural estimator of the likelihood-to-evidence ratio that can also be sampled. Despite these diverse mechanisms, the unifying feature of modern SBI methods is the training of a statistical model using a finite set of simulations to approximate the posterior $q_\phi(\theta|x)$ or a related quantity. The principles of ensemble learning from section 2.1 can be naturally applied in this context. Instead of training a single estimator $q_\phi(\theta|x)$, we train an ensemble of M posterior estimators, $\{q_m(\theta|x)\}_{m=1}^{M}$. Each of these can be considered as an approximation to the target posterior for the given observation x_{obs}.

Our central hypothesis is that the *consistency* among these approximations can serve as a valuable diagnostic. If the SBI training process is stable and has converged reliably for observation x_{obs}, the posterior estimates $q_m(\theta|x_{\text{obs}})$ from different ensemble members should be measureably similar. Conversely, significant divergence among them signals potential problems, such as insufficient simulation budget, poor optimiser convergence, or inherent difficulties in approximating the posterior for that specific x_{obs} (e.g. due to mismodelling).

2.3. The KL divergence matrix for ensemble assessment

To quantify the (dis)agreement between these posterior approximations, we employ the KL divergence. The KL divergence between a probability distribution function (PDF) Q and another PDF P measures the information lost when Q is used to approximate P. It is defined as:

$$\mathrm{KL}\left(P||Q\right) = \int P(\theta) \log \frac{P(\theta)}{Q(\theta)} \mathrm{d}\theta. \tag{1}$$

The measure $\mathrm{KL}\left(P||Q\right) \geqslant 0$ with equality being approached if and only if $P = Q$ almost everywhere. From the definition, we also see that the KL is in general asymmetric, i.e. $\mathrm{KL}\left(P||Q\right) \neq \mathrm{KL}\left(Q||P\right)$ for arbitrary P and Q. In our context, we will compute the KL divergence between pairs of posterior approximations $q_i(\theta|x_{\rm obs})$ and $q_j(\theta|x_{\rm obs})$ from the ensemble.

To systematically capture pairwise disagreements within the ensemble of posteriors $\{q_m(\theta|x_{\rm obs})\}_{m=1}^{M}$ for a specific observation $x_{\rm obs}$, we propose to construct an $M \times M$ matrix, termed the 'KL matrix', $\mathbf{K}$. The elements of this matrix are defined to be the KL divergences between the respective posterior approximations:

$$K_{ij} = \mathrm{KL}\left(q_i\left(\theta|x_{\rm obs}\right)||q_j\left(\theta|x_{\rm obs}\right)\right). \tag{2}$$

By definition, the diagonal elements K_{ii} are zero. The off-diagonal elements K_{ij} $(i \neq j)$ quantify the divergence from q_j to q_i, representing the information lost when approximating q_i using q_j.

A practical advantage, particularly relevant for SBI methods based on neural density estimators for the posterior (such as NPE), is the feasibility of estimating these KL divergences using Monte Carlo sampling. Specifically, K_{ij} can be approximated by drawing $N_{\rm s}$ samples[6] $\theta^{(k)}$ from the distribution $q_i(\theta|x_{\rm obs})$ and averaging the log-ratio of the probability densities:

$$K_{ij} \approx \frac{1}{N_{\rm s}} \sum_{k=1}^{N_{\rm s}} \left[\log q_i\left(\theta^{(k)}|x_{\rm obs}\right) - \log q_j\left(\theta^{(k)}|x_{\rm obs}\right)\right]. \tag{3}$$

All this estimation requires is the ability to both sample from each $q_m(\theta|x_{\rm obs})$ and evaluate the log-densities $\log q_m(\theta|x_{\rm obs})$, capabilities readily available in many modern SBI implementations. While generating samples is simple and efficient in the NPE case, it can be more costly for alternative approaches like NRE or NLE. However, this does not mean that estimating the components K_{ij} is also computationally expensive. For NRE, each ensemble member learns the ratio $r_i(\theta, x) \approx p(x|\theta)/p(x)$, leading to a posterior estimate $q_i(\theta|x_0) \propto r_i(\theta, x_0)p(\theta)$. The log-posterior difference needed for the KL calculation in equation (3) simplifies to $\log r_i(\theta^{(k)}, x_{\rm obs}) - \log r_j(\theta^{(k)}, x_{\rm obs})$, as the prior $p(\theta)$ cancels. As such, we see that the primary cost for NRE is generating samples from each ensemble member with stochastic sampling, after which computing the KL matrix only requires evaluating the learned ratio. For NLE, each member learns the likelihood $\tilde{q}_i(x|\theta) \approx p(x|\theta)$. This means that the posterior approximation is given by $q_i(\theta|x) = \tilde{q}_i(x|\theta)p(\theta)/\tilde{p}_i(x)$, where $\tilde{p}_i(x)$ is the approximate evidence of the data x for each ensemble member i. In this case, the log-posterior difference involves both the log-likelihood difference and the log of the Bayes factor between members. To compute this in full, this requires *both* posterior samples from each estimator $\tilde{q}_i(x_{\rm obs}|\theta)$ and an estimate of the evidence for each member $\tilde{p}_i(x_{\rm obs})$. In general, this latter component may be challenging, however, it can be computed along with posterior sampling using techniques like nested sampling. Thus, while the KL framework is most efficiently applied in NPE, it remains conceptually and practically applicable to NRE and NLE, with the main cost being the stochastic sampling step already required to obtain posterior constraints.

The KL matrix $\mathbf{K}$ serves as a rich diagnostic tool, allowing for quantitative assessment of ensemble consistency:

- **Magnitude of off-diagonal elements:** small values across all K_{ij} $(i \neq j)$ indicate high consistency; the ensemble members have converged to similar posterior approximations. This suggests the inference process was stable for $x_{\rm obs}$. Conversely, large values signal significant disagreement, potentially arising from insufficient training data (simulations), poor convergence of the underlying optimisation for some members, multimodality in the optimisation landscape, or inherent instability in approximating the posterior at $x_{\rm obs}$.

[6] To be more precise, enough samples should be drawn such that the Monte Carlo error in equation (3) falls below a chosen threshold.

- **Structure and asymmetry:** examining the matrix structure can reveal patterns. For instance, if one estimator q_k consistently yields large K_{ik} or K_{kj} values when paired with others, it might be an outlier. Asymmetry ($K_{ij} \neq K_{ji}$) can indicate differences in the distributional shapes; for example, if q_j is significantly more diffuse than q_i, K_{ij} (approximating the broader q_j with the narrower q_i) might be substantially larger than K_{ji} (approximating the narrower q_i with the broader q_j).

Crucially, the KL matrix provides a way to benchmark the internal consistency of the ensemble, which acts as a proxy for a *lower bound* on the systematic error associated with the inference process. A large value in the KL matrix, specifically a large $\max_{i\neq j} K_{ij}$, indicates significant disagreement between at least two ensemble members. Since the true posterior $p(\theta|x_{\rm obs})$ is unique (assuming a well-defined problem), if two approximations q_i and q_j diverge significantly from each other (i.e. K_{ij} or K_{ji} is large), they cannot both be arbitrarily close to the true posterior. Thus, a large $\max_{i\neq j} K_{ij}$ serves as a quantitative indicator of ensemble instability or unreliability for the given observation $x_{\rm obs}$. It suggests that the inference procedure yields significantly different outcomes under slight perturbations (such as different initialisations or data samples), casting doubt on the trustworthiness of any single member or even an aggregated result as a faithful approximation of the true posterior. While low KL values do not guarantee accuracy (all members could converge to the same wrong answer), high KL values strongly signal a problem with the consistency of the inference for that specific $x_{\rm obs}$. This quantitative diagnostic forms the basis for the tools and analyses presented in the following sections.

2.4. Interpreting KL divergence as systematic uncertainty

A key aspect of our framework is interpreting the KL divergence as a lower bound on the systematic uncertainty of the posterior approximation. To understand what a given KL value implies about the disagreement between posteriors, it is helpful to consider the case of two one-dimensional, Gaussian distributions. For this case, it is simple to derive an analytical expression for the KL divergence,

$$\mathrm{KL}\left(\mathcal{N}_1||\mathcal{N}_2\right) = \log\left(\frac{\sigma_2}{\sigma_1}\right) + \frac{\sigma_1^2 + (\mu_1-\mu_2)^2}{2\sigma_2^2} - \frac{1}{2}\,, \tag{4}$$

where μ_1, μ_2, σ_1, and σ_2 are the means and standard deviations of the two distributions, respectively. Whether a given level of KL divergence is 'too large' depends entirely on the application context. If an analysis requires parameter constraints more precise than this systematic uncertainty, then such a KL value indicates that the ensemble has not achieved sufficient consistency. The translation given in this section is most accurate for approximately Gaussian posteriors; for highly non-Gaussian or multimodal distributions, the KL value serves as a general inconsistency metric which can be benchmarked across the ensemble.

To better understand the mapping between the KL divergence and σ-levels, let us proceed further by considering small perturbations between the two distributions. We parametrise this as $\sigma_2 = \sigma_1(1+\epsilon)$ and $\mu_2 = \mu_1 + \gamma\sigma_1$ such that ϵ, $\gamma \ll 1$ are perturbations in units of σ_1. With this parameterisation, we can obtain the perturbative result,

$$\mathrm{KL}\left(\mathcal{N}_1||\mathcal{N}_2\right) \simeq \frac{\gamma^2}{2} + \epsilon^2 - \gamma^2\epsilon\,, \tag{5}$$

at the third order in the mean and standard deviation differences. An example of the KL divergence dependence on γ and ϵ is shown in figure 1 for a simple case of two 1-d Gaussian distributions. From equation (5), we see that the KL divergence scales quadratically for small differences in both the mean and widths of the distributions. With large differences, however, the divergence becomes more complex and can induce correlated effects. This suggests that the KL divergence has a simple interpretation when the differences between distributions are small, which is the regime we aim to work in when training SBI models. For reference, in figure 1, we show a set of benchmark examples to provide some graphical intuition regarding typical numerical values of the KL divergence.

The generalisation of equation (4) to n-dimensional Gaussian distributions is

$$\mathrm{KL}\left(\mathcal{N}_1||\mathcal{N}_2\right) = \frac{1}{2}\left[(\vec{\mu}_2-\vec{\mu}_1)^\dagger\,\Sigma_2^{-1}\,(\vec{\mu}_2-\vec{\mu}_1) + \log\frac{|\Sigma_2|}{|\Sigma_1|} + \mathrm{Tr}\left(\Sigma_2^{-1}\Sigma_1\right) - n\right]\,, \tag{6}$$

where $\vec{\mu}_{1,2}$ and $\Sigma_{1,2}$ are the mean vectors and covariance matrices respectively. In this case, we can gain insight into the perturbative limit using a similar approach to the one-dimensional case. We can diagonalise the system about the eigenvector basis U of Σ_1 by assuming $\Sigma_1 = U\Lambda^2 U^\dagger$ with $\Lambda =$

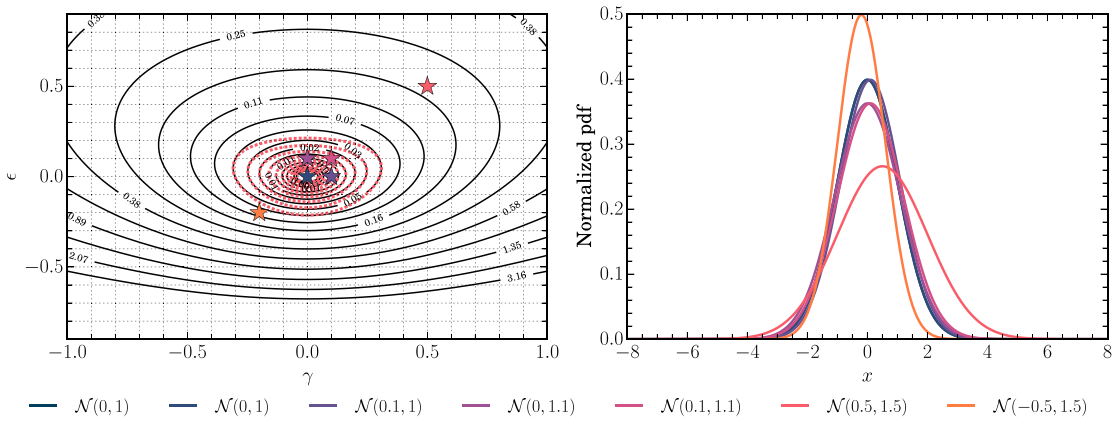


Figure 1. Interpreting the KL divergence: A set of examples illustrating quantitatively how the KL divergence depends on the differences in the means/variances of 1-d Normal distributions. Here, the base distribution $\mathcal{N}_1$ has $\mu_1 = 0$ and $\sigma_1 = 1$ such that $\mu_2 = \gamma$ and $\sigma_2 = 1 + \epsilon$ are the parameters of the contrasting distribution $\mathcal{N}_2$. *Left:* contour plot showing the value of the KL divergence for different values of ϵ and γ. Also shown (in red) is the quadratic approximation derived in equation (5), which we see matches well for small divergences. The coloured stars correspond to specific values reported in the legend. *Right:* 1d histograms showing the PDFs for the values of ϵ and γ corresponding to the stars in the left plot.

$\mathrm{diag}(\sigma_1, \ldots, \sigma_n)$ and defining perturbations $\Sigma_2 = U\Lambda'^2 U^\dagger$ and $U^\dagger(\vec{\mu}_2 - \vec{\mu}_1) = \Gamma\Lambda\vec{1}$ where $\Lambda' = \Lambda(\mathbb{I} + \mathcal{E})$, $\mathcal{E} = \mathrm{diag}(\epsilon_1, \ldots, \epsilon_n)$, and $\Gamma = \mathrm{diag}(\gamma_1, \ldots, \gamma_n)$. Then, in the diagonal basis, we recover

$$\mathrm{KL}\left(\mathcal{N}_1 \| \mathcal{N}_2\right) \simeq \sum_i^n \frac{\gamma_i^2}{2} + \epsilon_i^2 - \gamma_i^2 \epsilon_i \,, \tag{7}$$

which is indicative of how the KL divergence measure scales with dimensionality in the perturbative Gaussian limit. From this equation, it is manifest that the mismatch is expected to scale linearly with the number of dimensions. This can be useful to gauge the mismatch in higher-dimensional cases by comparing to the limiting perturbative Gaussian case. Guided by these considerations, we propose $\max_{ij}(\mathbf{K}/n) \ll 1$ as a heuristic necessary condition for consistency between models in the SBI context.

Another consideration is how the KL divergence is expected to scale as the size of the training set varies. In general, for an infinitely complex network capable of perfectly approximating the posterior, trained with a perfect strategy, we expect the divergence to endlessly scale with the inverse of the training set size (i.e. with more and more data, we expect the approximation to become better and better). However, for finite networks, an intrinsic variance due to imperfect training/network architecture will eventually set a floor (see the discussion around figure 2). To better understand the scaling behaviour of KL with the size of the training dataset n_{Train}, we can consider two models $\hat{P}_1(\theta)$ and $\hat{P}_2(\theta)$ trained to estimate the true PDF $P(\theta)$. We can assume that the offsets between estimators are small, such that $\hat{P}_i(\theta) = P(\theta) + \delta P_i(\theta)$, with $\delta P_i(\theta)$ denoting the residual with respect to $P(\theta)$. Using this setup, we can express the KL divergence between the two estimators as:

$$\mathrm{KL}\left(\hat{P}_1 || \hat{P}_2\right) \simeq \int \left[\delta P_1(\theta) - \delta P_2(\theta) + \frac{\delta P_1^2(\theta) + \delta P_2^2(\theta) - 2\delta P_1(\theta)\,\delta P_2(\theta)}{P(\theta)}\right] \mathrm{d}\theta. \tag{8}$$

We can proceed further by assuming the estimators to be unbiased, implying the expectation value (over the stochasticity due to initial random weights and steps involved in the training process) of the two models is $P(\theta)$, and $\langle \delta P_1(\theta)\rangle = \langle \delta P_2(\theta)\rangle = 0$. Then, the expectation value of equation (8) simplifies to

$$\langle \mathrm{KL}\left(\hat{P}_1 \| \hat{P}_2\right)\rangle \simeq \int \frac{\langle \delta P_1^2(\theta)\rangle + \langle \delta P_2^2(\theta)\rangle}{P(\theta)}\,\mathrm{d}\theta \propto \frac{1}{n_{\mathrm{Train}}}\,, \tag{9}$$

where we have assumed that, since $\hat{P}_i(\theta)$ are estimators of the true PDF, we can expect that, in the best-case scenario, the accuracy (i.e. the standard deviation) of such an estimate scales as $1/\sqrt{n_{\mathrm{Train}}}$. In such a case, the expectation value of $\mathrm{KL}(\hat{P}_1||\hat{P}_2)$ would scale as $1/n_{\mathrm{Train}}$. In the following sections, we will present an explicit example where we find good agreement with this analytical estimate.

3. KL divergence applied to ensemble training

In this section, we restrict ourselves to the case of NPE and explore some applications of the KL matrix as a diagnostic tool for SBI[7]. In section 3.1, we begin by reviewing the main ideas behind NPE. Then, in section 3.2, we discuss the use of the KL matrix to assess the convergence among different networks in an ensemble.

Finally, in section 3.3, we explore the usage of the KL matrix to identify model misspecification.

3.1. Basics of NPE

As discussed in section 2.2, the core idea of NPE is to approximate the posterior distribution $p(\theta|x)$ using a neural density estimator, such as a normalising flow. A standard choice of loss function to train this type of model is

$$\mathcal{L}(\phi) = -\mathbb{E}_{p(\theta,x)}\left[\log q_\phi(\theta|x)\right] = -\int p(\theta,x)\log q_\phi(\theta|x)\,\mathrm{d}\theta\,\mathrm{d}x, \tag{10}$$

where ϕ are the parameters of the network and $q_\phi(\theta|x)$ is the approximate trained posterior. In practice, the integral in the second line of equation (10) is approximated using a finite dataset of N simulated pairs $(\theta_i, x_i) \sim p(\theta,x)$ via a Monte Carlo average:

$$\mathcal{L}(\phi) \approx -\frac{1}{N}\sum_{i=1}^{N}\log q_\phi(\theta_i|x_i)\ . \tag{11}$$

It is easy to show that, up to factors that do not depend on the network parameters, such a loss function is proportional to the KL divergence between the true posterior $p(\theta|x)$ and the learned approximation $q_\phi(\theta|x)$, averaged over the marginal data distribution $p(x) = \int p(x|\theta)p(\theta)\mathrm{d}\theta$:

$$\begin{aligned}\mathcal{L}(\phi) &= -\int p(x)\,p(\theta|x)\log q_\phi(\theta|x)\,\mathrm{d}\theta\,\mathrm{d}x \\ &= \int p(x)\left(\int p(\theta|x)\log\frac{p(\theta|x)}{q_\phi(\theta|x)}\,\mathrm{d}\theta\right)\mathrm{d}x + \text{const.} \\ &= \mathbb{E}_{p(x)}\left[\mathrm{KL}\left(p(\theta|x)\|q_\phi(\theta|x)\right)\right] + \text{const.},\end{aligned} \tag{12}$$

so that minimising the loss function is equivalent to minimising the KL divergence between $p(\theta|x)$ and $q_\phi(\theta|x)$. This connection highlights that NPE seeks an approximation q_ϕ that is close to the true posterior $p(\theta|x)$ in an information-theoretic sense, on average across possible datasets x drawn from the model. Moreover, it gives additional motivation for our choice of the KL divergence as a diagnostic tool for SBI. In particular, given two learned approximants $q_i(\theta|x)$ and $q_j(\theta|x)$ of $p(\theta|x)$, it is possible to construct counterexamples where the KL divergences between p and q_i, as well as p and q_j are small, while the KL divergence between q_i, q_j is larger. Thus, computing the KL divergence matrix for the networks in the ensemble defines a test that provides *independent* information compared to the loss function.

As discussed above, a particular advantage of NPE in this context is the relative ease of sampling and evaluating the log-density of the approximate posterior. This makes estimating the KL divergences across ensembles especially fast and efficient. However, we emphasise that there are no fundamental barriers to evaluating KL for the other SBI paradigms mentioned in section 2.2.

3.2. Assessing ensemble convergence and estimating systematic errors

For the first illustrative example demonstrating the application of the KL matrix as a diagnostic test, we consider training an ensemble of identical networks with initial random weights for a specific inference task. For this purpose, we choose a simple case study where the simulator corresponds to a two-dimensional Gaussian mixture model from the `sbibm` library [4]. In this context we can investigate a set of features exhibited by the ensemble such as: (a) the evolution of the KL matrix as a function of the number of training epochs, (b) the evolution of the KL matrix as a function of the training dataset size, and (c) the comparison between these behaviours and the standard loss function in equation (10). Before presenting the results, we note that all the relevant technical details regarding the network architectures, training settings, prior ranges, etc are provided in appendix.

[7] For simplicity, for all the examples considered in this work, we have only considered the upper triangular part of the **K** matrix, as K_{ij} and K_{ji} provide correlated information. The asymmetry ($K_{ij} \neq K_{ji}$) can provide additional diagnostic information about the relative shapes of the posteriors, but we do not explore it in this work as we found it to be a subdominant effect in our experiments.

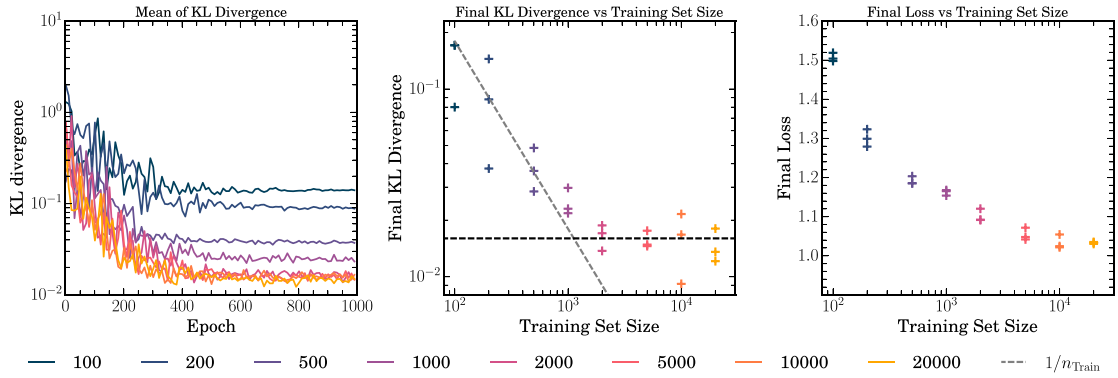


Figure 2. KL matrix as a training diagnostic. *Left:* the mean pairwise KL divergence, monitored during the training of single ensembles. Each curve represents one ensemble trained with a specific dataset size (see legend), demonstrating how the KL matrix can be used to track convergence and determine when further training is unlikely to improve consistency. *Middle:* value of the KL divergences in the ensemble (averaged over the last 30 epochs) at the end of the training as a function of training dataset size. In dashed grey, we show the expected $1/n_{\text{Train}}$ behaviour before saturating the intrinsic variance floor. *Right:* value of the loss function at the end of the training process as a function of the training dataset size.

The main results for this subsection are presented in figures 2 and 3. They demonstrate a number of core properties that justify our statement that the KL matrix can be used as a diagnostic tool for ensembles of posterior approximations. A key use case is monitoring the KL matrix for a single ensemble during training. As shown in the left panel of figure 2, when the KL divergence stabilises, we can conclude that further training will not improve ensemble consistency, and the asymptotic KL value provides an estimate of the systematic uncertainty for that training configuration. To estimate the systematic uncertainty for a single, fixed training setup, only one ensemble of multiple members is required.

Firstly, if we look at the left panel of figure 2, we observe the behaviour of the average KL divergence in the KL matrix as a function of the training epoch for a number of different training set sizes. Specifically, we see that for all training set sizes, the KL divergences initially decrease as the networks learn a better approximation to the true posterior. Eventually, once the information contained in the training dataset has been fully utilised and the networks have converged, an asymptotic value of the KL divergences is reached. The key insight here is that this asymptotic value can be used to directly estimate the systematic error associated with the specific training configuration (i.e. training set size, training dynamics, etc) via the interpretation discussed in section 2.4. In our plots, for visual clarity, we show the average of the off-diagonal KL matrix elements. However, the primary diagnostic for assessing convergence or flagging outliers is the maximum pairwise KL divergence, $\max_{i\neq j} K_{ij}$, as this value provides the lower bound on the approximation error.

If we then look at the middle panel of figure 2, we see a more detailed picture of the asymptotic values reached in the KL matrices for each experiment. These demonstrate empirically a number of key ideas pointed out in section 2. This panel also demonstrates how the KL matrix can be used to diagnose the source of disagreement. We observe two distinct regimes: (i) a data-limited regime at small n_{Train}, where the KL divergence decreases as $1/n_{\text{Train}}$ (in line with equation (9)), indicating that disagreement is driven by insufficient training data; and (ii) an optimisation-limited regime at large n_{Train}, where the KL divergence plateaus. This floor reflects intrinsic variance from the training dynamics (e.g. random initialisation, stochastic gradient descent, architectural instabilities), showing that simply adding more data will not improve consistency. This saturation value can be used to compare different training strategies (e.g. learning rate scheduling, hyperparameter choices, etc), with lower values indicating more stable performance, and to estimate the systematic error associated with the training dynamics itself. We emphasise that this information cannot be obtained when training only a single network, but must be bootstrapped via a trained ensemble.

The rightmost panel of figure 2 illustrates the long-term values of the validation loss for the different training datasets. The main takeaway here is simply that whilst the loss function shows similar convergence as the dataset size increases, the behaviour is subtly different from the KL matrices. This justifies our statement that the diagnostic tests we suggest here provide *complementary* information to the loss function itself.

Finally, for reference, in figure 3, we show samples drawn from the three networks in the ensemble at the end of the training process, after conditioning to the same reference observation. The different

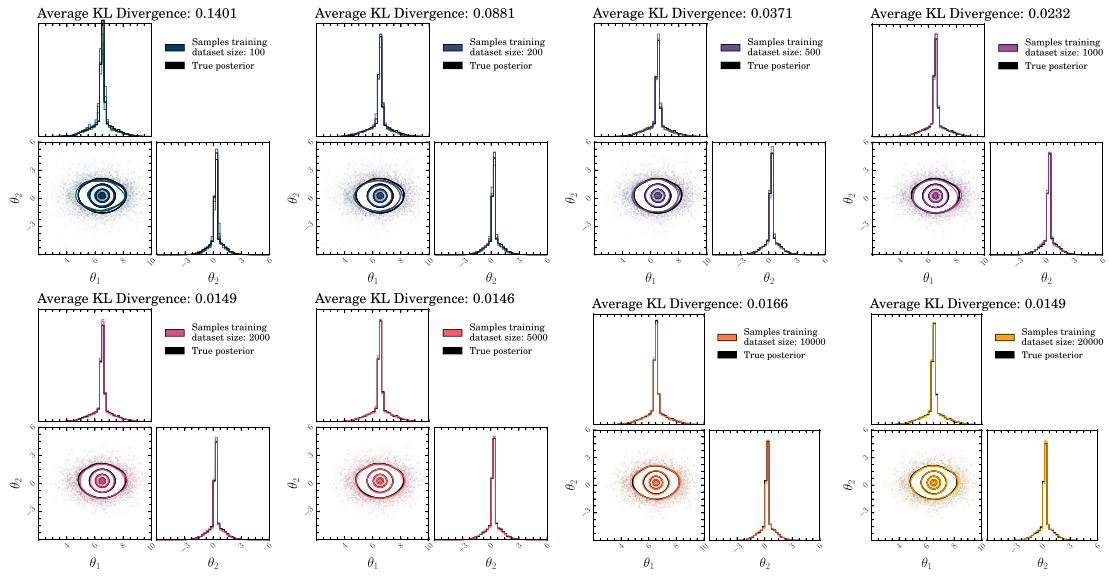


Figure 3. Evolution of the posteriors. True posterior distribution (black) compared to samples drawn from the different networks in the ensembles. The different panels (top left to bottom right) correspond to different training dataset sizes (colour code as in figure 2). Each panel shows the results from a single ensemble trained with a specific dataset size; multiple ensembles are only needed to compare how systematic uncertainties vary with training conditions, not to estimate the uncertainty for a single setup.

panels correspond to different training dataset sizes, which are chosen to match figure 2. In all the panels, we show the true posterior in black. Once again, we can appreciate that, while all panels show qualitatively good agreement between the true posterior and the approximants, the KL matrix provides a quantitative measure to assess the accuracy and consistency of the approximation. We demonstrate the convergence of both the average, and maximum/minimum KL divergences across the ensemble in the appendix. While visual inspection of these 2D marginals is feasible, the KL matrix is particularly powerful in high-dimensional problems. The KL divergence captures the full multivariate disagreement in a single scalar per ensemble member pair, providing a quantitative and automated diagnostic that can be integrated into training pipelines and systematic uncertainty budgets.

As a final comment, we stress that while the current work only presents results for the two-dimensional Gaussian mixture model, we have followed the same procedure for other typical benchmark examples from the `sbibm` package [4], reaching similar conclusions.

3.3. Testing model misspecification

In the previous section, we demonstrated the utility of the KL matrix as a diagnostic tool for tracking the evolution and stability of network training. In this section, we will further develop this idea and explore the possibility of using the KL matrix to detect model misspecification. 'Misspecification' in our framework refers to the scenario where the simulator used for training does not accurately represent the data-generating process for the real observed data. This mismatch is revealed only when we apply the trained estimators to actual observations, as during training the model is correctly specified by definition. The underlying hypothesis is simple: given the flexibility of neural network models, we have no guarantees on the extrapolation behaviour of a trained model beyond data in the training set. In particular, for an ensemble of models, we cannot expect that if network predictions agree (as quantified by low values in the KL matrix, for example) within the training sample that they will each extrapolate in an identical manner. This is broadly a statement about the large degeneracy that exists in the parameter space of large neural networks when optimising a scalar loss function. In general, this is presented as a negative feature, however, here we point out that one can actually leverage this fact to detect model misspecification.

A crucial prerequisite for this test is to distinguish between training-induced variance and misspecification-induced variance. We define two quantities: (i) the training-time KL divergence, $\mathrm{KL}_{\mathrm{train}}$, which is measured on simulated test data from the same distribution as the training data, and quantifies the baseline disagreement due to the training process. (ii) The inference-time KL divergence, $\mathrm{KL}_{\mathrm{obs}}$, measured on the actual observed data. Our misspecification diagnostic is based on the difference $\Delta = \mathrm{KL}_{\mathrm{obs}} - \mathrm{KL}_{\mathrm{train}}$. A significant positive value of Δ suggests the observation is out-of-distribution, as it triggers enhanced disagreement among ensemble members compared to their baseline. Therefore, the

test is most powerful when the ensemble has achieved a low, stable baseline $\mathrm{KL}_{\mathrm{train}}$, otherwise any signal from misspecification could be lost in the noise of training variance.

To test this hypothesis, we will train an ensemble of networks until the KL divergence among the networks in the ensemble is below a certain threshold value. Then, we will condition the density estimators on some observations that do not match the model in the simulator and compare the KL matrix on this new ensemble of posteriors to the values obtained during training. To systematically evaluate the performance of the KL divergence as a detector for model misspecification, we introduce a parameter to control the level of discrepancy between training and observed data. Following this procedure, we can keep track of the changes in the KL divergence between trained networks as we increase the level of model misspecification.

The specific example that we will study in this section is as follows: Each data realisation consists of $n = 10$ data points (in the following, we will label them with $k \in [1, n]$) drawn from a normal distribution with mean

$$\mu_k = \theta + a\mathcal{B}_k \,, \tag{13}$$

and standard deviation $\sigma_k = \sigma = 0.01$. The parameter θ is drawn from a uniform prior $\theta \sim \mathrm{Uniform}(-1, 1)$ and gives a constant contribution to the mean of the dataset and

$$\mathcal{B}_k \equiv \left(1.01 - \left|\frac{2\,(k-1)}{9} - 1\right|^{0.25}\right) \Big/ \bar{\mathcal{B}}_k \,, \tag{14}$$

with $\bar{\mathcal{B}}_k \simeq 0.9326$, is a symmetric bump centred at $k = n/2$, with the denominator chosen to have $\max_{k\in[1,\ldots,10]}(\mathcal{B}_k) = 1$. The contribution of this bump to μ_k is then controlled by the parameter a. Notice that for $a = 0$, there is no k dependence in the dataset; i.e. the samples for different values of k are independent realisations drawn from the same statistical distribution. For this very simple model, given d_k, we can determine the maximum likelihood estimator (and its variance) for the parameters analytically. In particular, in the absence of the bump, the solution is given by $\bar{\theta} = \sum_k d_k/n$ and $\sigma_{\bar{\theta}} = \sigma/\sqrt{n}$.

For the example presented in this section, we fix $N_{\mathrm{d}} = 10000$ as the size of our training dataset[8] and fix $a = 0$ for all examples used during the training process. The ensemble used for this example consists of $N_{Net} = 5$ networks. After fixing the network architecture and training strategy, we train and monitor the evolution of the KL divergence among all networks in the ensemble. At the end of our run, the maximal value of the KL divergence is $\sim 7 \times 10^{-2}$, corresponding to a maximal mismatch of $\sim .37\sigma$ in the peak position (see section 2.3 for the conversion from KL values to σ-levels). As already mentioned, for all datasets we used during the training process, we fix $a = 0$, meaning we only train our networks on examples that do not contain the contribution due to the bump. In other words, if the observed data consisted solely of θ, the candidate model would be correctly specified. This establishes a clear reference point against which the effects of misspecification can be assessed. Specifically, we can compare the observed values in the KL matrix for a given (potentially misspecified) observation and compare them to 7×10^{-2}, or look for posteriors that differ by more than $\sim .37\sigma$.

In figure 4, we compare the results obtained conditioning the networks of the ensemble on an observation matching data used for the training process (left plot), i.e. $a = 0$, and an observation that is misspecified (right plot) with $a = 3 \times 10^{-2}$. The two plots in the first row show the data realisation with and without adding the bump component. It is evident that with the addition of the bump, the data deviates from the simulation model (which assumes a flat mean across the bins) used during the training process. In the bottom row, we show the posteriors obtained from the different networks in the ensemble after conditioning on the corresponding observation. Specifically, on the left hand side, which excludes the bump, all posteriors agree well (i.e. to the level expected from the KL matrix analysis) with each other and with the true posterior (shown in black). On the other hand, on the right hand side, we see a much larger spread among the posteriors given by the different networks. To be more quantitative, after conditioning on an observation that lies outside the training set, we find a KL divergence among the networks of the ensemble that significantly exceeds the threshold value used during the training process. We highlight this visually by showing the orange vertical contours, which denote the $\pm 0.37\sigma$

[8] To be more accurate, as explained in appendix, we draw N_{d} values for θ, which we keep fixed during the training process and, every time we show a pair (θ, d_k) to the networks in the ensemble, we draw an independent realisation of d_k given θ. Following this procedure, we effectively show many instances of the data corresponding to the same parameters, which allows the networks to better learn the statistical properties of the data.

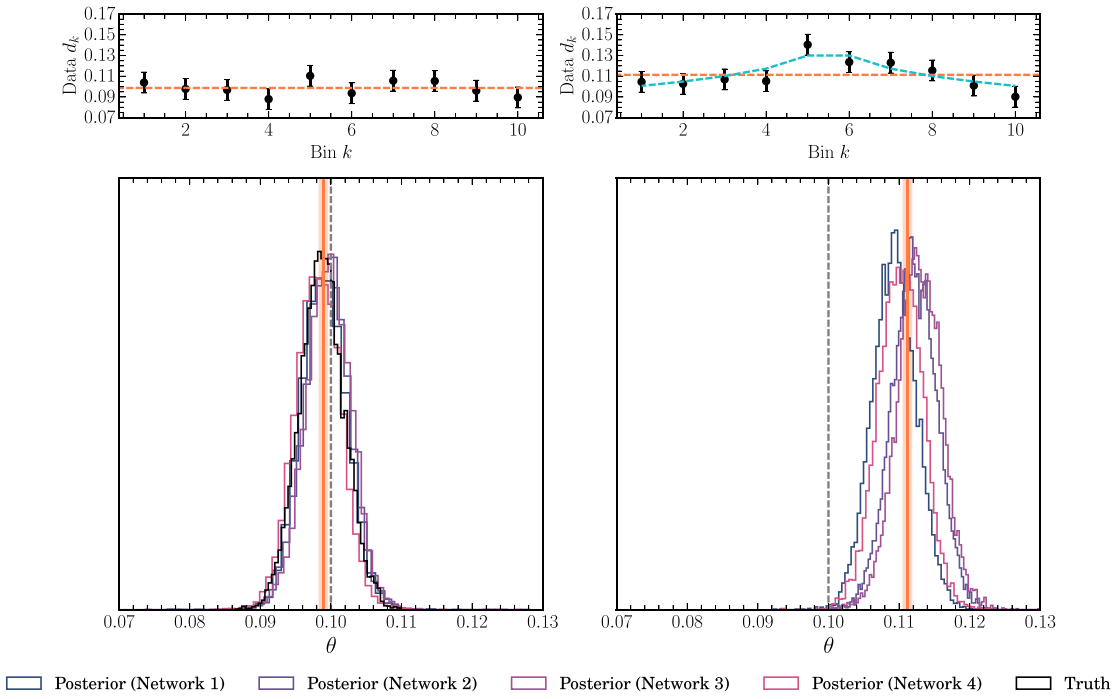


Figure 4. Detecting Model Misspecification. *Left*: Posteriors (bottom plot) obtained from networks conditioned to an observation (top plot) compatible with the data used during the training. In the top plot, the data realisation is shown in black, and the dashed orange line represents the input theoretical model μ_k. The orange vertical band in the bottom plot shows the maximal expected spread among estimates given the final value of the KL matrix ($\sim 7 \times 10^{-2}$) during training. As expected, given the small KL at the end of the training, all posteriors agree. For reference, we show the true posterior in black. The dashed grey line represents the injected value of θ and the orange vertical line the mean value inferred from the data. *Right:* Posteriors (bottom plot) from the same networks conditioned to an observation (top plot) that is *not* compatible with the data used during the training. It is manifest that, once conditioned to an observation that does not match the model used in the training process, the spread in the posteriors increases significantly, which we can easily detect using the KL matrix. The orange solid and grey dashed lines indicate the same quantities as in the left-hand figure.

variance expected from the in-sample training result. We can see that the posteriors in the misspecified model disagree to a level that is not captured by this error.

To better explore this behaviour, in the right plot of figure 5, we show how the KL divergences among the networks of the ensemble change as we condition on observations with increasingly large values of a in the range $a \in [10^{-3}, 2]$. For reference, in the left plots, we show the relative size of the bump for a number of benchmark cases corresponding to the vertical lines in the right-hand figure. With these results, we can see clearly that the deviation in the values of the KL matrix starts to become visible when a becomes larger than the threshold value $a_{\text{Thresh}} \sim 1.3 \times 10^{-2}$. Interestingly, this corresponds to deviations in the data at about the 1-2σ level. This suggests that our proposed misspecification test is highly sensitive. For values of a well below a_{Thresh}, we see that the values of the KL divergence remain consistent with the value achieved during the training process. However, for a above a_{Thresh}, the KL divergence starts to increase significantly. Thus, we can conclude that model misspecification can induce a significant increase in the KL divergence with respect to the value reached during training. In general, this provides a brand new test for detecting model misspecification that directly harnesses the power of ensemble learning.

Classical approaches to model misspecification detection include goodness-of-fit tests based on χ^2 statistics comparing observed and expected data distributions, posterior predictive checks, and Bayesian model comparison via evidence ratios. In our specific example with the bump model, a χ^2 test on the residuals between the data and the best-fit flat model would indeed detect the misspecification at comparable sensitivity levels ($\approx 1 - 2\sigma$). While classical tests like the χ^2 statistic are computationally cheaper and should be a standard first check, our ensemble-based approach offers several complementary advantages: (i) it does not require explicit specification of a null hypothesis or alternative model, (ii) it provides a continuous diagnostic that can be monitored during analysis rather than requiring a separate validation step, and (iii) it naturally accounts for parameter uncertainty through the ensemble disagreement rather than relying on point estimates. Recent ML-enhanced approaches include using

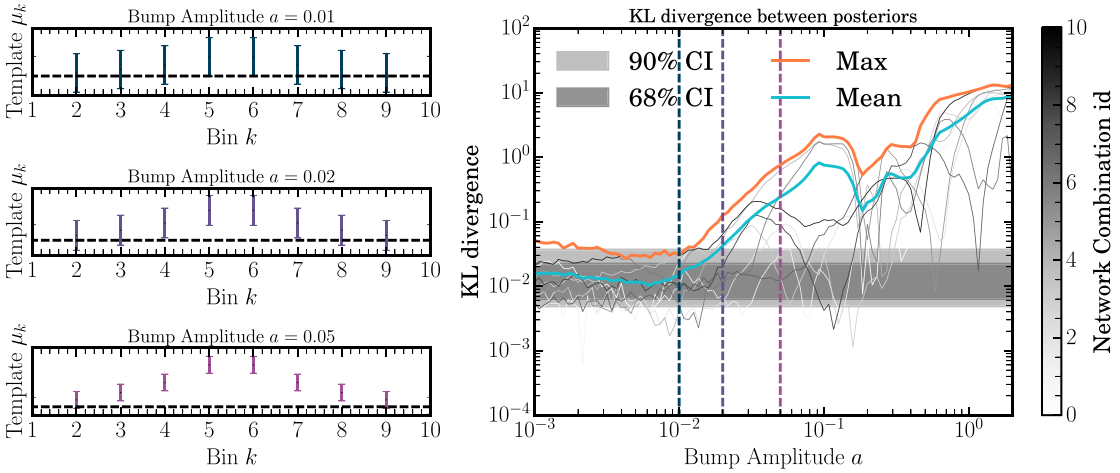


Figure 5. Tracking the amount of model misspecification. *Left:* Three examples of the injection templates $\mu_k = \theta + a\mathcal{B}_k$, corresponding to the dashed vertical lines in the right-hand figure. The error bars correspond to the 1σ noise levels in each bin, and the horizontal dashed lines indicate the equivalent flat template with $a = 0$. *Right:* The elements of the KL matrix as a function of the amplitude of the bump. The grey bands are the percentiles computed among networks in the regime where the bump contribution is negligible compared to the flat component. The remaining curves indicate the behaviour of the KL divergence associated with each network combination. The mean (light blue) and maximum (orange) curves are specifically highlighted to be used as a diagnostic for detecting model misspecification.

classifier-based test statistics [16], where a neural network is trained to distinguish between data generated under different hypotheses. Our KL-based approach is conceptually related but operates in posterior space rather than data space, making it naturally integrated with the inference procedure itself. A systematic comparison of these approaches across various misspecification scenarios would be valuable future work.

4. Outlook and conclusions

In this work, we have explored the usage of ensemble learning as a tool to diagnose convergence, stability, and model fidelity in SBI. The core idea is very simple: if an SBI algorithm reliably approximates the true posterior distribution, then independently trained models on the same simulation budget should yield consistent results. In other words, since the true posterior is unique, a small discrepancy among ensemble members is a necessary, but not sufficient, condition to ensure that all posterior estimates can be sufficiently close to the true posterior. Therefore, disagreement among ensemble members, as quantified by the KL divergence in this work, becomes an accessible and informative signal of failure modes such as undertraining, optimiser variance, or model misspecification.

We have shown that the KL divergence between posterior approximations within an ensemble provides a principled, quantitative diagnostic without requiring access to the true posterior. The KL matrix constructed from pairwise divergences among ensemble members encodes information about convergence properties, with elevated divergences signalling internal inconsistency and potential unreliability of the inferred posterior for a specific observation. In particular, we demonstrated that as the size of the training dataset increases, the KL divergences among the ensemble members decrease until they reach a floor set by the intrinsic variance of the training procedure and the network architecture, which offers a useful lower bound on the systematic errors intrinsic to the training process.

Moreover, we demonstrated in section 3 that ensemble disagreement, as measured by the KL divergence, can serve as a practical detector for model misspecification. Conditioning trained estimators on out-of-distribution observations results in a measurable increase in divergence between ensemble members, underscoring the diagnostic power of this approach beyond mere convergence assessment. Our experiments with controlled perturbations to observed data confirmed that the KL divergence can track systematic deviations beyond the information provided by the loss function. It would be interesting in future work to perform a systematic comparison of the sensitivity of this KL-based test against classical techniques and other ML-enhanced tools. Another interesting direction would be to deliberately train on misspecified simulations (e.g. with known simplifications) and use the KL diagnostic to assess when and where these simplifications become problematic. It would also be interesting to explore further ways to enhance ensemble learning for SBI. For example, by drawing inspiration from parallel tempering, training models with different learning rates, allowing occasional information exchange via the KL matrix

could improve robustness and generalisation, as well as achieving a better balance between exploration and exploitation. Furthermore, in analogy with reversible jump MCMC (RJ-MCMC), dynamically adapting model architectures within an ensemble would allow models to 'jump' between different complexities based on performance, potentially leading to more efficient use of computational resources. Beyond leveraging ensembles for consistency checks, combining outputs from diverse models also offers a natural path to more stable posterior estimates by mitigating variance across architectures.

Overall, our results suggest that ensemble training, combined with internal divergence metrics such as the KL matrix, provides a scalable, flexible, and inference-agnostic framework for improving the transparency and reliability of SBI pipelines. This paradigm is particularly well suited for scientific settings where inference is performed once, but scrutiny must be rigorous. Future work will explore generalisations to other divergence measures, combinations with SBC methods, and applications to high-dimensional real-world inference tasks.

Data availability statement

The code to reproduce all results in this manuscript is publicly available on GitHub at https://github.com/james-alvey-42/sbi-ensemble-diagnostics, and the datasets generated and analysed during the current study are available in the Zenodo repository, https://zenodo.org/records/17522623.

Acknowledgments

We thank Jonas Elias El Gammal, Michele Mancarella, Michael Williams, and Christoph Weniger for their comments on a nearly final version of this draft. M P would like to thank Lorenzo Cevolani for very useful comments on a draft of this work. M P and J A acknowledge the hospitality of Imperial College London, which provided office space during some parts of this project. J A is supported by a fellowship from the Kavli Foundation. The work of MP is supported by the Comunidad de Madrid under the Programa de Atracción de Talento Investigador with number 2024-T1TEC-3134.

Appendix. Further details and tests

In this appendix, we provide further details on the setups used for this work, along with additional plots to clarify the training process and the behaviour of the ensembles.

All posterior density estimation performed in this work relies on the neural spline flow (NSF) architectures [44] implemented via the `build_nsf` function[9] in the `sbi` library [45]. Each NSF network models the conditional posterior distribution $p(\theta|x)$, where simulated observations x are provided as input and the network outputs a flexible approximation over parameters θ. Training data x is generated on the fly and loaded in mini-batches of size 64. A particular aspect of our training process that may be non-standard is 'noise resampling'. While the set of parameter values $\{\theta\}$ is kept fixed throughout training, the observations $\{x\}$ are dynamically regenerated by the simulator whenever data is requested. This approach effectively augments the training data with new noise realisations for each θ, enhancing robustness and reducing variance induced by fixed realisations.

For all our examples, we independently train an ensemble of networks, each with an input dimension matching the observation space of the specific simulation task. Each network is optimised using the `AdamW` optimiser with an initial learning rate of 10^{-2}. A `StepLR` scheduler reduces the learning rate by a factor of 0.9 every 10 epochs. As detailed in the main text, a central component of our ensemble approach is the use of the KL divergence to assess agreement among the ensemble members. Specifically, every 10 epochs, we estimate the pairwise KL divergence between the learned posterior distributions using 10 000 samples drawn from the estimated posteriors. Early stopping is triggered when the maximum absolute KL divergence across all pairs falls below a tolerance of 10^{-3}, indicating that the networks have converged to mutually consistent posterior approximations.

The Gaussian mixture model example used in section 3.2 follows the default configuration from the `sbibm` benchmark library [4] to ensure reproducibility. The specific model details are as follows:

[9] For all examples considered in this work, we stick to the default internal architecture parameters for components such as the number of coupling layers, hidden features, and spline configurations.

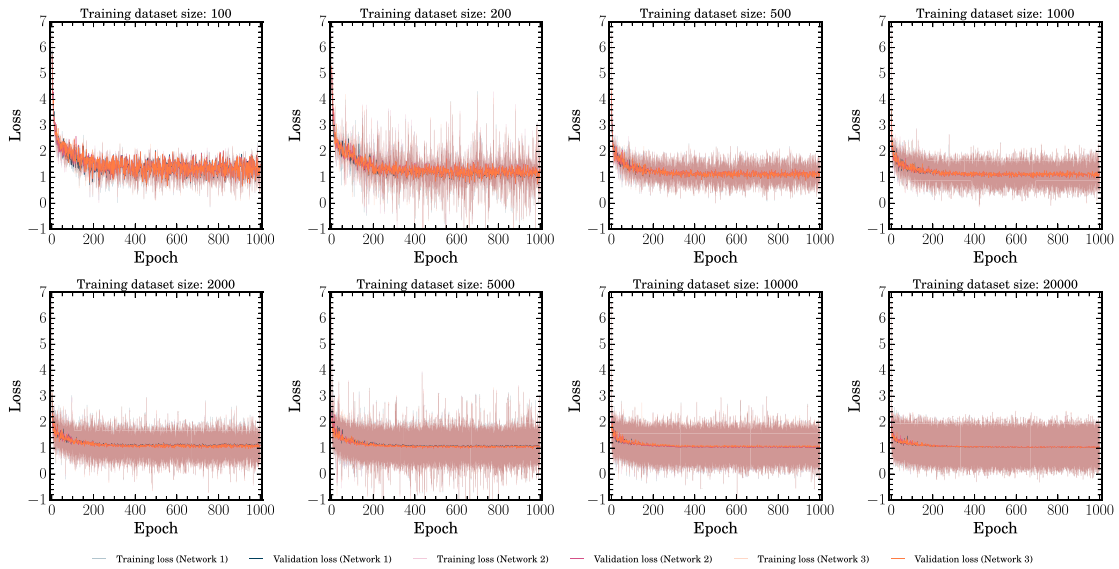


Figure 6. Training and Validation Losses. Training (light colour) versus validation loss (full colour) for the different dataset sizes (100, 200, 500, 1000 top row, left to right, and 2000, 5000, 10 000, 20 000 left to right bottom row) as a function of epochs for all the networks in the ensemble (red, green, blue) discussed in section 3.2.

- The prior distributions for both parameters are uniform: $\theta_1, \theta_2 \sim \text{Uniform}(-10, 10)$.
- The likelihood $p(x|\theta)$ is a mixture of two 2D Gaussian distributions with means at (θ_1, θ_2) and $(-\theta_1, -\theta_2)$, with equal weights of 0.5.
- The standard deviations of each mode are as specified in the `sbibm` benchmark library implementation [4].

We also provide further details as to the ensembles used in section 3.2. As explained in the main text, the ensembles consist of three networks that are trained in parallel using different training dataset sizes n_{Train}, ranging from 100 to 20000 across the various experiments. We train each network for up to 1000 epochs, fixing the validation set size to $n_{\text{Val}} = 2 \times n_{\text{Train}}$. In figure 6, we show the training/validation losses as functions of epochs. All panels, meaning all ensembles with different values of n_{Train}, exhibit a similar, and expected, behaviour. After a steep decrease in the initial epochs, the training and validation losses oscillate randomly around some asymptotic value. While for small n_{Train}, both training and validation loss exhibit quite a large variance around such asymptotic value, for larger n_{Train}, the validation loss variance is strongly reduced. This behaviour originates from our choice for n_{Val}: for larger n_{Train}, the validation dataset is larger, and the results are less subject to statistical fluctuations.

We conclude this appendix by providing further details on the models used in section 3.3. As already mentioned in the main text, for this example, parameters θ are uniformly sampled from $[-1, 1]$. The corresponding observations x are produced by a simulator that adds Gaussian noise ($\sigma = 0.01$) to θ across the 10 independent measurements. In practice, each observation corresponds to a 10-dimensional data vector modelling noisy measurements. The ensemble used for this example consists of five networks trained in parallel using the procedure described above in this appendix. The training dataset consists of 10 000 θ samples (the observations are resampled dynamically as discussed above), and the validation dataset has size 2000. In the left panel of figure 7, we show the training/validation losses for the five networks in the ensemble as a function of training epochs. As expected, after a steep initial decrease, the losses of the five networks flatten out and fluctuate randomly around an asymptotic value. In the right panel of the same figure, we show the evolution of the KL divergences among the different networks in the ensemble as functions of training epochs. Again, after an initial phase, they all tend to scatter around some nearly constant value. We notice that all KL divergences involving network ID 1 are significantly larger, indicating that there is less consistency between this particular model and the others in the ensemble. For reference, the maximum value of KL among all combinations at the end of the training process is $\simeq 7 \times 10^{-2}$. We use this number in the main text as the lower bound to which we compare detections of model misspecification.

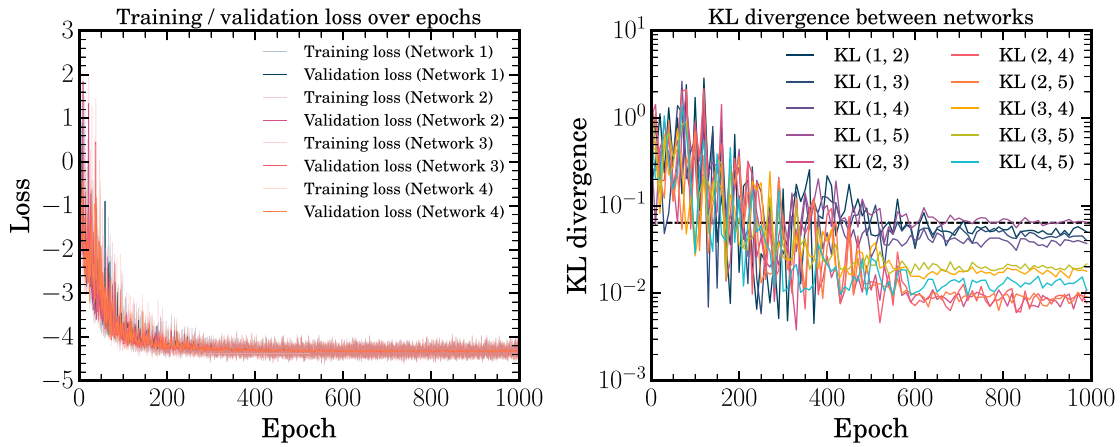


Figure 7. Training Losses and KL matrix. *Left:* Training and validation loss as functions of training epochs for the five networks in the ensemble trained in section 3.3 (see legend). *Right:* Evolution of the KL divergence between network combinations in the ensemble (see legend for various combinations) as a function of training epochs.

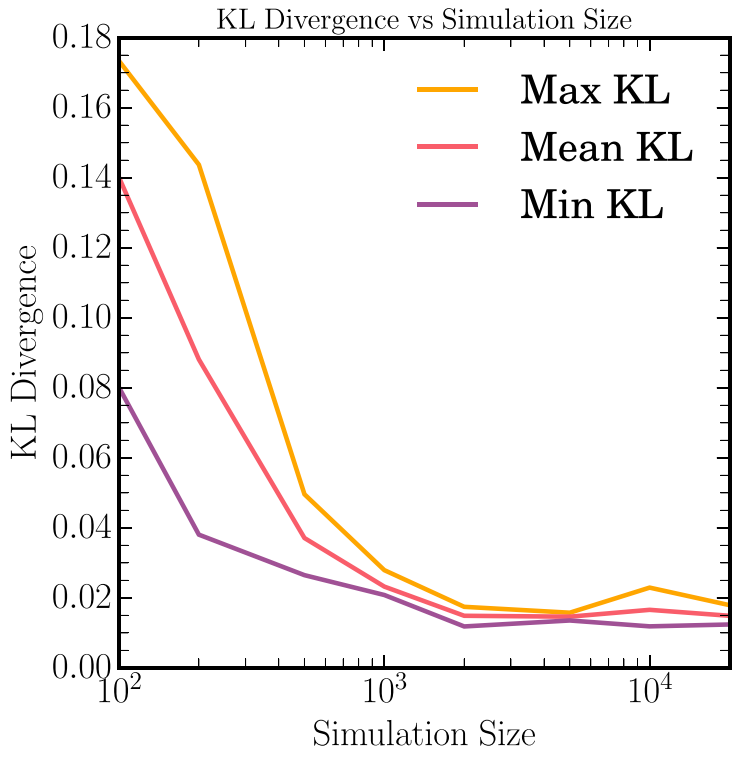


Figure 8. Evolution of maximum/average KL divergences. Behaviour of the maximum, minimum and average values of the KL divergences in the ensembles trained for the example discussed in section 3.2. We see that all three quantities decrease systematically as the dataset size increases until convergence is reached.

ORCID iDs

James Alvey 0000-0003-2020-0803
Carlo R Contaldi 0000-0001-7285-0707
Mauro Pieroni 0000-0003-0665-266X

References

[1] Cranmer K, Brehmer J and Louppe G 2020 *Proc. Natl Acad. Sci.* **117** 30055
[2] Greenberg D, Nonnenmacher M and Macke J 2019 *Int. Conf. on Machine Learning* (PMLR) pp 2404–14 (arXiv:1905.07488 [cs.LG])
[3] Papamakarios G and Murray I 2016 *Advances in Neural Information Processing Systems* vol 29 (arXiv:1605.06376 [stat.ML])

[4] Lueckmann J-M, Boelts J, Greenberg D, Goncalves P and Macke J 2021 *Int. Conf. on Artificial Intelligence and Statistics* (PMLR) pp 343–51 (arXiv:2101.04653 [stat.ML])
[5] Hermans J, Begy V and Louppe G 2020 *Int. Conf. on Machine Learning* (PMLR) pp 4239–48 (arXiv:1903.04057 [stat.ML])
[6] Rozet F and Louppe G 2021 Arbitrary marginal neural ratio estimation for simulation-based inference (arXiv:2110.00449 [cs.LG])
[7] Alsing J and Wandelt B 2019 *Mon. Not. Roy. Astron. Soc.* **488** 4440
[8] Papamakarios G, Sterratt D and Murray I 2019 *Proc. of the 22nd Int. Conf. on Artificial Intelligence and Statistics (AISTATS) (Proc. of Machine Learning Research)* vol 89 (PMLR) pp 837–48 (arXiv:1805.07226 [stat.ML])
[9] Miller B K, Cole A, Forré P, Louppe G and Weniger C 2021 *35th Conf. on Neural Information Processing Systems* (https://doi.org/10.5281/zenodo.5043706) (arXiv:2107.01214 [stat.ML])
[10] Deistler M, Goncalves P J and Macke J H 2022 *Advances in Neural Information Processing Systems* vol 35 p 23135 (arXiv:2210.04815 [stat.ML])
[11] Cook S R, Gelman A and Rubin D B 2006 *J. Comput. Graph. Stat.* **15** 675 (https://www.jstor.org/stable/27594203)
[12] Talts S, Betancourt M, Simpson D, Vehtari A and Gelman A 2018 (arXiv:1804.06788 [stat.ME])
[13] Hermans J, Delaunoy A, Rozet F, Wehenkel A, Begy V and Louppe G 2021 *TMLR (2021)* (arXiv:2110.06581 [stat.ML])
[14] Lemos P, Coogan A, Hezaveh Y and Perreault-Levasseur L 2023 *Int. Conf. on Machine Learning* (PMLR) pp 19256–73 (arXiv:2302.03026 [stat.ML])
[15] Anau Montel N, Alvey J and Weniger C 2025 *Phys. Rev.* D **111** 083013
[16] Cranmer K, Pavez J and Louppe G 2016 Approximating likelihood ratios with calibrated discriminative classifiers (arXiv:1506.02169 [stat.AP])
[17] Dalmasso N, Izbicki R and Lee A B 2020 (arXiv:2002.10399 [stat.ME])
[18] Heinrich L 2022 Learning optimal test statistics in the presence of nuisance parameters (arXiv:2203.13079 [physics, stat])
[19] Huang D, Bharti A, Souza A, Acerbi L and Kaski S 2023 *Advances in Neural Information Processing Systems* vol 36 p 7289 (available at: https://proceedings.neurips.cc/paper_files/paper/2023/hash/16c5b4102a6b6eb061e502ce6736ad8a-Abstract-Conference.html)
[20] Wehenkel A, Gamella J L, Sener O, Behrmann J, Sapiro G, Cuturi M and Jacobsen J-H 2024 Addressing misspecification in simulation-based inference through data-driven calibration (arXiv:2405.08719 [stat.ML])
[21] Jeffrey N and Wandelt B D 2024 *Mach. Learn.: Sci. Technol.* **5** 015008
[22] Gessey-Jones T and Handley W J 2024 *Phys. Rev.* D **109** 123541
[23] Mancini A S, Docherty M M, Price M A and McEwen J D 2023 *RAS Tech. Instrum.* **2** 710
[24] Cannon P, Ward D and Schmon S M 2022 (arXiv:2209.01845 [stat.ML])
[25] Schmitt M, Bürkner P-C, Köthe U and Radev S T 2024 Detecting model misspecification in amortized Bayesian inference with neural networks: an extended investigation (arXiv:2406.03154 [cs.LG])
[26] Kelly R P, Warne D J, Frazier D T, Nott D J, Gutmann M U and Drovandi C 2025 (arXiv:2503.12315 [stat.ME])
[27] von Wietersheim-Kramsta M, Lin K, Tessore N, Joachimi B, Loureiro A, Reischke R and Wright A H 2025 *Astron. Astrophys.* **694** A223
[28] Pierre S, Blancard B R-S, Hahn C and Eickenberg M 2025 (arXiv:2507.03086 [astro-ph.CO])
[29] Opitz D and Maclin R 1999 *J. Artif. Intell. Res.* **11** 169–98
[30] Polikar R 2006 *IEEE Circuits Syst. Mag.* **6** 21
[31] Rokach L 2010 *Artif. Intell. Rev.* **33** 1
[32] Kullback S and Leibler R A 1951 *Ann. Math. Stat.* **22** 79
[33] Gelman A and Rubin D B 1992 *Stat. Sci.* **7** 457
[34] Sagi O and Rokach L 2018 *Wiley Interdiscip. Rev.: Data Min. Knowl. Discovery* **8** e1249
[35] Elsemüller L, Olischläger H, Schmitt M, Bürkner P-C, Köthe U and Radev S T 2023 *TMLR (2024)* (https://doi.org/10.48550/arXiv.2310.11122) (arXiv:2310.11122 [stat.ML])
[36] Dietterich T G 2000 *Int. Workshop on Multiple Classifier Systems* (Springer) pp 1–15 10.1007/3-540-45014-9_1
[37] Hastie T, Tibshirani R, Friedman J and Franklin J 2005 *Math. Intelligencer* **27** 83
[38] Breiman L 1996 *Mach. Learn.* **24** 123
[39] Freund Y and Schapire R E 1997 *J. Comput. Syst. Sci.* **55** 119
[40] Friedman J H 2001 *Ann. Stat.* **7** 1189
[41] Wolpert D H 1992 *Neural Netw.* **5** 241
[42] Lakshminarayanan B, Pritzel A and Blundell C 2017 *Advances in Neural Information Processing Systems* vol 30 (arXiv:1612.01474 [stat.ML])
[43] Yao Y, Régaldo-Saint Blancard B and Domke J 2024 *Int. Conf. on Artificial Intelligence and Statistics* (PMLR) pp 4267–75 (arXiv:2310.17009 [stat.ME])
[44] Durkan C, Bekasov A, Murray I and Papamakarios G 2019 *Published at the 33rd Conf. on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada* (https://doi.org/10.48550/arXiv.1906.04032) (arXiv:1906.04032 [stat.ML])
[45] Boelts J, Deistler M, Gloeckler M and Tejero-Cantero A *et al* 2025 *J. Open Source Softw.* **10** 7754