

Neural Methods for Amortised Parameter Inference

Andrew Zammit-Mangion¹, Matthew Sainsbury-Dale^{1,2}, and
Raphaël Huser²

¹School of Mathematics and Applied Statistics, University of Wollongong, Wollongong, New South Wales, Australia; email: azm@uow.edu.au

²Statistics Program, Computer, Electrical and Mathematical Sciences and Engineering Division, King Abdullah University of Science and Technology (KAUST), Thuwal, Saudi Arabia

Abstract

Simulation-based methods for making statistical inference have evolved dramatically over the past 50 years, keeping pace with technological advancements. The field is undergoing a new revolution as it embraces the representational capacity of neural networks, optimisation libraries, and graphics processing units for learning complex mappings between data and inferential targets. The resulting tools are amortised, in the sense that they allow inference to be made quickly through fast feedforward operations. In this article we review recent progress made in the context of point estimation, approximate Bayesian inference, the automatic construction of summary statistics, and likelihood approximation. The review also covers available software, and includes a simple illustration to showcase the wide array of tools available for amortised inference and the benefits they offer over state-of-the-art Markov chain Monte Carlo methods. The article concludes with an overview of relevant topics and an outlook on future research directions.

Keywords— Approximate Bayesian inference; Likelihood-Free Inference; Likelihood Approximation; Neural Networks; Simulation-Based Inference; Variational Bayes

1 INTRODUCTION

Statistical inference is the process of drawing conclusions on an underlying population based on an observational sample, and is the cornerstone of evidence-based decision making and scientific discovery. This process often relies on a parametric statistical model with unknown or uncertain parameters. Parameter inference consists in estimating, or finding a distribution over, these model parameters. When the underlying parametric model admits a likelihood function that is analytically and computationally tractable, a suite of likelihood-based methods are available that are well-suited for parameter inference. However, there are many cases where the likelihood function is either unavailable or computationally prohibitive (i.e., time consuming to evaluate), but where it is feasible to simulate from the model; this is the case, for example, with several geophysical models (e.g., Siahkoohi et al., 2023), epidemiological models (Fasiolo et al., 2016), cognitive neuroscience models (Fengler et al., 2021), agent-based models (Dyer et al., 2024), and some classes of statistical models like Markov random fields (e.g., Besag, 1986) or models for spatial extremes (e.g., Davison et al., 2012, Huser and Wadsworth, 2022). In such cases, one often resorts to simulation-based techniques to bypass evaluation of the likelihood function when making parameter inference. Simulation-based inference requires substantial computing capability, and thus only became a viable solution in the second half of the 20th century (see Hoel and Mitchell, 1971, Ross, 1971, for early examples). The field has evolved much since then, due to the dramatic increase in affordable computing power, and the increased ability to generate, store, and process large datasets.

Simulation-based inference: Any procedure that makes statistical inference from data by leveraging information from simulations of a generative model.

Several review papers have been published in recent years on simulation-based inference. The review paper of Cranmer et al. (2020) has a high-level and broad scope, and gives a succinct summary of several methods ranging from approximate Bayesian computation (ABC, Sisson et al., 2018) to density estimation (Diggle and Gratton, 1984) and more recent machine-learning approaches, including some of the neural-network based approaches we review here. The reviews of Blum et al. (2013) and Grazian and Fan (2020) focus on ABC methods while those of Drovandi (2018) and Drovandi and Frazier (2022) additionally review indirect inference (Gourieroux et al., 1993), and Bayesian synthetic likelihood (BSL, Wood, 2010) approaches to simulation-based inference.

This review paper differs from those available in the existing literature on simulation-based inference in two ways. First, it exclusively considers simulation-based methods that incorporate neural networks; the reason for this focus is that neural networks have become state-of-the-art in high-dimensional modelling due to their representational capacity (Hornik, 1989) and due to the increased availability of the software and hardware required to train them. Second, it focuses on neural simulation-based methods that are amortised, that is, that leverage a feed-forward relationship between the data and the inferential target to allow for fast inference. The Oxford Languages dictionary defines amortisation as “the action or process of gradually writing off the initial cost of an asset”. In the context of neural amortised inference, the initial cost involves training a neural network to learn a complex mapping used for inference (e.g., a point estimator, an approximate posterior distribution, an approximate likelihood function, etc.). Once the neural network is trained, parameter inferences can then be made several orders of magnitude faster than classical methods (e.g., Markov chain Monte Carlo (MCMC)). Hence, the initial cost of training the network is “amortised over time”, as the trained network is used over and over again for making inferences with new data (see the sidebar titled The Power of Amortisation). Amortised inference is also associated with the way humans operate, reusing demanding inferences made in the past to make quick decisions (Gershman and Goodman, 2014).

Neural network: A flexible, highly parameterised nonlinear mapping, constructed using a composition of simple functions connected together in a network.

Training: The process of estimating (or “learning”) parameters in a neural network, typically by minimising an expected loss using a variant of stochastic gradient descent.

THE POWER OF AMORTISATION

Neural networks are ideally suited for situations in which a complex task needs to be done repeatedly, and where the import of the problem justifies a big initial outlay. In such settings, one expends a large cost to initially train the network with the view of reaping dividends over time with its repeated use, a strategy known as amortisation. The power of amortisation is perhaps most clearly exhibited in large language models like OpenAI’s ChatGPT. Training the BigScience Large Open-science Open-access Multilingual Language Model (BLOOM) required over a million hours of computing on several hundreds of state-of-the-art graphics processing units (GPUs), each costing many thousands of dollars, for nearly four months (Luccioni et al., 2023). The total estimated consumed energy for training was 433 MWh, which is roughly equivalent to what 72 Australian households would consume in an entire year. However, once trained, BLOOM could repeatedly produce polished text outputs (inferences) in response to inputs (data) on a single GPU with a total consumed energy that would be far less than that needed to boil water for a cup of tea.

Not all simulation-based inference methods are amortised, and may require re-optimisation or re-sampling every time new data are observed. This makes them unsuitable for situations where the same inferential task needs to be repeated many times on different datasets, as is often the case in operational settings. For example, NASA’s Orbiting-Carbon-Observatory-2 (OCO-2) satellite takes approximately 1,000,000 measurements of light spectra per day. For each spectrum, an estimate of carbon dioxide mole fraction is obtained by computing a Bayesian posterior mode via a forward physics

model (Cressie, 2018), requiring an exorbitant amount of computing power when done at scale. Neural networks offer a way forward to make Bayesian inference from the spectra quickly and accurately at a tiny fraction of the computational cost (David et al., 2021).

1.1 Paper outline

This review provides an introductory high-level roadmap to a suite of amortised neural inferential methods. It organises and categorises these methodologies to help navigate this relatively new and dynamic field of research. In Section 2 we discuss (Bayesian) point estimation and methods that yield approximate posterior distributions. We also discuss why amortisation is possible, by drawing on a result by Brown and Purves (1973). In Section 3 we present ways in which neural networks can be used to construct summary statistics from data, which often feature in amortised statistical inference methods. In Section 4, we review neural methods for amortised likelihood-function and likelihood-to-evidence-ratio approximation. In Section 5, we discuss available software for making amortised parameter inference, and we demonstrate their use on a simple model where asymptotically exact inference with MCMC is possible. In Section 6, we outline additional topics that are related to the main topic of the review, and in Section 7 we conclude. The review paper also contains Supplementary Material that includes background material on neural network architectures and normalising flows often used in the context of amortised inference; an additional illustration; and a brief discussion of related topics.

1.2 Notation convention

Throughout the review we consider data $\mathbf{Z} \in \mathcal{Z}$ from a sample space $\mathcal{Z} \subseteq \mathbb{R}^n$, and some model parameter vector of interest $\boldsymbol{\theta} \in \Theta$, where $\Theta \subseteq \mathbb{R}^d$ is the parameter space. Unless specified otherwise, we assume that all measures admit densities with respect to the Lebesgue measure, in which case we use $p(\cdot)$ to denote an arbitrary density function. To simplify notation, we let the argument of $p(\cdot)$ indicate both the random variable the density is associated with and its input argument. In particular, $p(\boldsymbol{\theta})$ denotes the prior distribution of the parameters, $p(\mathbf{Z})$ is the marginal likelihood (also known as the model evidence), and $p(\boldsymbol{\theta} | \mathbf{Z}) = p(\mathbf{Z} | \boldsymbol{\theta})p(\boldsymbol{\theta})/p(\mathbf{Z})$ is the posterior density function evaluated at $\boldsymbol{\theta} \in \Theta$ and $\mathbf{Z} \in \mathcal{Z}$. We refer to $p(\mathbf{Z} | \boldsymbol{\theta})$ as the likelihood function, irrespective of whether it is expressing a density over \mathbf{Z} for some fixed $\boldsymbol{\theta}$, or a function over $\boldsymbol{\theta}$ for some fixed \mathbf{Z} , with its definition taken from the context in which it is used. We denote an approximate function (e.g., approximate likelihood function or posterior density function) generically as $q(\cdot)$; we let $\boldsymbol{\kappa}$ parameterise the approximation when $q(\cdot)$ refers to an approximate posterior density, and use $\boldsymbol{\omega}$ when $q(\cdot)$ refers to an approximate likelihood function. We often make these parameters themselves a function of the data; for example, $\boldsymbol{\kappa}_\gamma(\mathbf{Z})$ returns parameters characterising the posterior density from data, and is itself parameterised by γ .

2 NEURAL POSTERIOR INFERENCE

The Bayesian framework provides a natural setting in which to consider amortised inference with neural networks. In this section, we review two prominent classes of amortised Bayesian inferential approaches: neural Bayes estimation (Section 2.1) and methods for neural posterior inference (Section 2.2) that approximate Bayesian posterior distributions via the minimisation of an expected Kullback–Leibler (KL) divergence (Kullback and Leibler, 1951). The methods discussed here assume the existence of a mapping that returns optimal inferences on $\boldsymbol{\theta}$ for any given \mathbf{Z} ; in Section 2.3, we explain how the existence result of Brown and Purves (1973) relates to several of the amortisation schemes we review.

Neural-network architecture: The specific design of a neural network, including the operations executed in each layer, how the layers are connected, and the number of layers.

Normalising flow: A sequence of invertible mappings used to establish a flexible family of distributions.

2.1 Neural Bayes Estimators

One often wishes to obtain point estimates of parameters from data. We call a mapping $\hat{\boldsymbol{\theta}} : \mathcal{Z} \rightarrow \Theta$ a neural point estimator when the mapping is constructed using a neural network. It is straightforward to construct such a mapping from simulations: Assume we have N simulations of parameters and corresponding datasets from an underlying model, $\{\{\boldsymbol{\theta}^{(i)}, \mathbf{Z}^{(i)}\} : i = 1, \dots, N\}$. One can perform point estimation by training (i.e., fitting) a neural network that maps the input \mathbf{Z} to the output $\boldsymbol{\theta}$, using the simulations and a chosen loss function $L(\cdot, \cdot)$. Denote the neural point estimator as $\hat{\boldsymbol{\theta}}_{\boldsymbol{\gamma}}(\cdot)$, where $\boldsymbol{\gamma}$ are the neural network parameters. The neural network is trained by solving the optimisation problem

$$\boldsymbol{\gamma}^* = \arg \min_{\boldsymbol{\gamma}} \sum_{i=1}^N L(\boldsymbol{\theta}^{(i)}, \hat{\boldsymbol{\theta}}_{\boldsymbol{\gamma}}(\mathbf{Z}^{(i)})), \quad (1)$$

where $L(\cdot, \cdot)$ is the chosen loss function, such as the absolute or squared-error loss. The architecture of $\hat{\boldsymbol{\theta}}_{\boldsymbol{\gamma}}(\cdot)$ is largely determined by the structure of the data (see Section S1 of the Supplementary Material), and optimisation is typically done using stochastic gradient descent in conjunction with automatic differentiation, implemented using various machine-learning libraries (see Section 5 for examples). Once trained, the neural network $\hat{\boldsymbol{\theta}}_{\boldsymbol{\gamma}^*}(\cdot)$ returns point estimates through simple feedforward evaluation, and is often orders of magnitude faster than classical likelihood-based methods. This is the simplest form of neural amortised inference.

Neural networks have been extensively used for parameter point estimation in recent years. Gerber and Nychka (2021), Lenzi et al. (2023), and Sainsbury-Dale et al. (2024) use a convolutional neural network (CNN) to estimate covariance parameters in spatial Gaussian process models or models of spatial extremes, the latter of which are traditionally considered computationally difficult to fit. Liu et al. (2020) also estimate the parameters of a Gaussian process, but adopt a transformer network to cater for realisations of variable dimension and highly-parameterised covariance functions. Zammit-Mangion and Wikle (2020) use a CNN with three-dimensional kernels to estimate the dynamical parameters of an advection-diffusion equation, while Rudi et al. (2022) apply a CNN with one-dimensional kernels for parameter estimation with ordinary differential equations. In all cases, the neural networks are seen to provide good estimates at a small fraction of the time required by likelihood-based estimators (a speedup of at least a hundred-fold is typical). As shown in several of these works, a side-benefit of being able to estimate parameters very quickly is that uncertainty can be naturally quantified through parametric or non-parametric bootstrap techniques with little extra computational effort.

To glean insight into the estimators' properties, it is helpful to consider them within an inferential decision-theoretic framework (Casella and Berger, 2001). Consider the Bayes risk, that is, the expected loss over $\mathbf{Z} \in \mathcal{Z}$ and $\boldsymbol{\theta} \in \Theta$, of an estimator $\hat{\boldsymbol{\theta}}(\cdot)$,

$$r_{\text{B}}(\hat{\boldsymbol{\theta}}(\cdot)) \equiv \int_{\Theta} \int_{\mathcal{Z}} L(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}}(\mathbf{Z})) p(\mathbf{Z} | \boldsymbol{\theta}) p(\boldsymbol{\theta}) d\mathbf{Z} d\boldsymbol{\theta}. \quad (2)$$

A minimiser of $r_{\text{B}}(\hat{\boldsymbol{\theta}}(\cdot))$ is said to be a Bayes estimator. Assume now that the neural point estimator $\hat{\boldsymbol{\theta}}_{\boldsymbol{\gamma}}(\cdot)$ is flexible enough to approximate the true Bayes estimator arbitrarily well. A simulation-based approach to constructing a neural point estimator proceeds as follows: Replace the estimator in Equation 2 with the neural point estimator, approximate the Bayes risk in Equation 2 with a Monte Carlo approximation, and then minimise the empirical risk function with respect to the neural network parameters $\boldsymbol{\gamma}$. This Monte-Carlo-based empirical risk minimisation problem is precisely that given in Equation 1. For this reason, we call a neural network trained as in Equation 1 a neural Bayes estimator; see Sainsbury-Dale et al. (2024, Section 2) for more discussion and for an example comparing the neural Bayes estimator to the analytic Bayes estimator for a simple model.

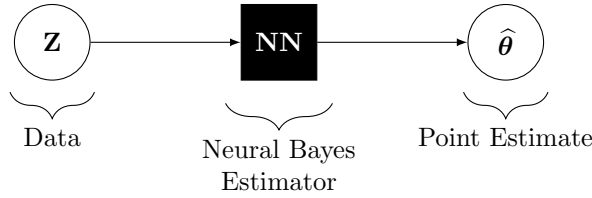
Point estimator: Any function of the data \mathbf{Z} that returns an estimate of an unknown model parameter $\boldsymbol{\theta}$ (i.e., any d -dimensional statistic; Casella and Berger, 2001, Chapter 7).

Loss function: A nonnegative function that quantifies the incurred loss from an estimate not equalling the true value.

Convolutional neural network: A neural network architecture involving multiple convolutional operations, where the convolution kernels are estimated during training.

The connection between neural point estimators and Bayes estimators is important for at least two reasons: First, it clearly shows that the way in which the parameters are simulated for training the neural network matters, as it induces a prior distribution over the parameters on which the resulting estimator depends. Second, Bayes estimators have well-understood properties that can be drawn on to further understand those of the neural estimators. Importantly, Bayes estimators are functionals of the posterior distribution: Under the squared-error loss $L_{sel}(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}}(\mathbf{Z})) \equiv \|\hat{\boldsymbol{\theta}}(\mathbf{Z}) - \boldsymbol{\theta}\|^2$, the Bayes estimator is the posterior mean $\mathbb{E}(\boldsymbol{\theta} \mid \mathbf{Z})$; under the loss $L_{var}(\theta_j, \hat{\theta}_j(\mathbf{Z})) \equiv ((\theta_j - \mathbb{E}(\theta_j \mid \mathbf{Z}))^2 - \hat{\theta}_j(\mathbf{Z}))^2$ for some parameter θ_j , the Bayes estimator is the posterior variance $\mathbb{V}(\theta_j \mid \mathbf{Z})$ (Fan and Yao, 1998); and under the quantile loss function $L_\rho(\theta_j, \hat{\theta}_j(\mathbf{Z})) \equiv (\hat{\theta}_j(\mathbf{Z}) - \theta_j)(\mathbb{1}\{\hat{\theta}_j(\mathbf{Z}) > \theta_j\} - \rho)$ the Bayes estimator $\hat{\theta}_j$ is the posterior ρ -quantile. Posterior quantiles can be used for fast quantification of posterior uncertainty of $\boldsymbol{\theta}$ via the construction of credible intervals. For example, Sainsbury-Dale et al. (2023) obtain posterior medians and 95% credible intervals of Gaussian-process-covariance-function parameters in over 2000 spatial regions from a million observations of sea-surface temperature in just three minutes on a single GPU. The required time is orders of magnitude less than that required by classical techniques such as (non-amortised) MCMC or variational inference.

In all cases discussed in this section, the neural network, once trained, returns a point estimate via a fast feedforward operation, and has the following graphical representation:



Despite being a point estimator, the Bayes estimator can be viewed as the solution to a KL optimisation problem. Specifically, consider the approximate posterior density $q(\boldsymbol{\theta}; \hat{\boldsymbol{\theta}}(\mathbf{Z})) \propto \exp(-L(\boldsymbol{\theta}; \hat{\boldsymbol{\theta}}(\mathbf{Z})))$. Then, from Equation 2, the Bayes estimator $\hat{\boldsymbol{\theta}}^*(\cdot)$ is

$$\begin{aligned} \hat{\boldsymbol{\theta}}^*(\cdot) &= \arg \min_{\hat{\boldsymbol{\theta}}(\cdot)} - \int_{\mathbf{Z}} \int_{\Theta} p(\mathbf{Z}) p(\boldsymbol{\theta} \mid \mathbf{Z}) \log q(\boldsymbol{\theta}; \hat{\boldsymbol{\theta}}(\mathbf{Z})) d\boldsymbol{\theta} d\mathbf{Z} \\ &= \arg \min_{\hat{\boldsymbol{\theta}}(\cdot)} \mathbb{E}_{\mathbf{Z}} [\text{KL}(p(\boldsymbol{\theta} \mid \mathbf{Z}) \parallel q(\boldsymbol{\theta}; \hat{\boldsymbol{\theta}}(\mathbf{Z})))], \end{aligned} \quad (3)$$

where here $\text{KL}(p(\boldsymbol{\theta} \mid \mathbf{Z}) \parallel q(\boldsymbol{\theta}; \hat{\boldsymbol{\theta}}(\mathbf{Z})))$ is the forward KL divergence between $p(\cdot \mid \mathbf{Z})$ and $q(\cdot; \hat{\boldsymbol{\theta}}(\mathbf{Z}))$. Neural Bayes estimators are therefore a special case of the more general class of approximate Bayesian techniques discussed in Section 2.2. The outer expectation in Equation 3 leads to the point estimator $\hat{\boldsymbol{\theta}}^*(\cdot)$ being optimal, in a Bayes sense, for any \mathbf{Z} (see Section 2.3 for details). However, in practice, there will be some discrepancy between a trained neural network $\hat{\boldsymbol{\theta}}_{\gamma^*}(\cdot)$ and the true Bayes estimator $\boldsymbol{\theta}^*(\cdot)$. Any extra bias or variance introduced that makes the trained estimator sub-optimal with respect to the target estimator (assumed to be globally optimal in a KL sense for any data \mathbf{Z}) is referred to as the amortisation gap (Cremer et al., 2018), and this is a consideration for all the remaining approaches discussed in this review.

Finally, we note that, in addition to parameter inference, neural Bayes estimators are also often used to generate useful summary statistics quickly for use with other downstream inferential algorithms, such as ABC; see Section 3 for examples and further discussion.

Kullback–Leibler divergence: A dissimilarity measure between two distributions. For two densities $p(\cdot)$ and $q(\cdot)$, we denote the forward KL divergence between $p(\cdot)$ and $q(\cdot)$ as $\text{KL}(p(\mathbf{X}) \parallel q(\mathbf{X})) \equiv \int p(\mathbf{X}) \log \frac{p(\mathbf{X})}{q(\mathbf{X})} d\mathbf{X}$. We denote the reverse KL divergence as $\text{KL}(q(\mathbf{X}) \parallel p(\mathbf{X}))$.

Amortisation gap: Error introduced in amortised inference because of incomplete training (e.g., not enough simulations), or because of neural network inflexibility, or both.

2.2 Approximate Bayesian inference via KL-divergence minimisation

Finding an approximate distribution by minimising the KL divergence between $p(\boldsymbol{\theta} \mid \mathbf{Z})$ and an approximate posterior density forms the basis of many approximate inference techniques. In this review, we focus on amortised versions of the two main classes of such techniques: those that minimise the forward KL divergence, and those that minimise the reverse KL divergence (i.e., variational Bayes). Both classes are approximate Bayesian methods that are often used when other asymptotically exact methods such as MCMC are computationally infeasible. The approximate density $q(\boldsymbol{\theta}; \boldsymbol{\kappa})$ has distributional parameters $\boldsymbol{\kappa}$ that need to be estimated. For example, when $q(\boldsymbol{\theta}; \boldsymbol{\kappa})$ is chosen to be Gaussian, the parameters $\boldsymbol{\kappa} = (\boldsymbol{\mu}', \text{vech}(\mathbf{L})')'$ contain a d -dimensional mean parameter $\boldsymbol{\mu}$, and the $d(d+1)/2$ non-zero parameters of the Cholesky factor \mathbf{L} of a covariance matrix, where $\text{vech}(\cdot)$ is the half-vectorisation operator.

Both forward and reverse KL minimisation approaches target the true posterior distribution, in the sense that in both cases the KL divergence is zero if and only if the approximate distribution is identical to the true posterior distribution. However, when the true posterior distribution is not in the class of approximating distributions, the optimal approximate distribution depends on the direction of the KL divergence. As shown by Murphy (2012, Chapter 21), minimising the reverse KL divergence leads to approximate distributions that are under-dispersed and that tend to concentrate on a single mode of the target distribution. On the other hand, minimising the forward KL divergence leads to approximate distributions that are over-dispersed and that cover all modes of the target distribution. Although both approaches are ubiquitous, in simulation-based settings forward KL approaches offer some advantages over their reverse KL counterparts; namely, they are natively likelihood-free and they have a more straightforward implementation.

2.2.1 Forward KL-divergence minimisation

We first consider the non-amortised context, where the optimal approximate-distribution parameters $\boldsymbol{\kappa}$ are found by minimising the forward KL divergence between the posterior distribution and the approximate distribution:

$$\boldsymbol{\kappa}^* = \arg \min_{\boldsymbol{\kappa}} \text{KL}(p(\boldsymbol{\theta} \mid \mathbf{Z}) \parallel q(\boldsymbol{\theta}; \boldsymbol{\kappa})). \quad (4)$$

The optimisation problem in Equation 4 needs to be solved for every \mathbf{Z} , and will be computationally expensive if it needs to be done repeatedly. Amortisation can be imbued in the learning problem by making the approximate-posterior-distribution parameters themselves a function of the data. The amortised form of the approximate posterior distribution is given by $q(\boldsymbol{\theta}; \boldsymbol{\kappa}(\mathbf{Z}))$ for $\boldsymbol{\theta} \in \Theta, \mathbf{Z} \in \mathcal{Z}$, where the function $\boldsymbol{\kappa}(\cdot)$ maps the data to the approximate-posterior-density parameters. We then find the function $\boldsymbol{\kappa}^*(\cdot)$ that minimises the expected forward KL divergence:

$$\boldsymbol{\kappa}^*(\cdot) = \arg \min_{\boldsymbol{\kappa}(\cdot)} \mathbb{E}_{\mathbf{Z}}[\text{KL}(p(\boldsymbol{\theta} \mid \mathbf{Z}) \parallel q(\boldsymbol{\theta}; \boldsymbol{\kappa}(\mathbf{Z}))]. \quad (5)$$

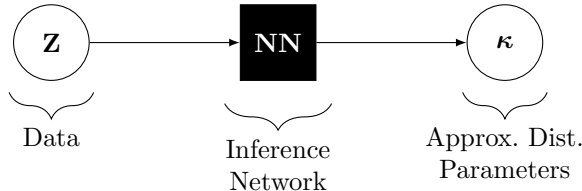
The function $\boldsymbol{\kappa}^*(\cdot)$ has the property that it minimises the KL divergence between $p(\boldsymbol{\theta} \mid \mathbf{Z})$ and the approximate posterior density for every $\mathbf{Z} \in \mathcal{Z}$, and hence yields the optimal approximate-posterior-density parameters for every $\mathbf{Z} \in \mathcal{Z}$ (see Section 2.3 for more details). In neural amortised inference, one models the mapping $\boldsymbol{\kappa}(\cdot)$ from the data to the distribution parameters using a neural network. This mapping, which we write as $\boldsymbol{\kappa}_{\gamma}(\cdot)$, can be constructed using the architectures discussed in Section S1 of the Supplementary Material, with γ denoting neural network parameters. The function $\boldsymbol{\kappa}_{\gamma}(\cdot)$ is often referred to as an inference network, and γ is found by minimising a Monte Carlo approximation to the expected KL divergence. Specifically, the optimisation problem reduces to

Variational Bayes: A type of approximate Bayesian inference where the approximate distribution is that which minimises the reverse KL divergence between the true posterior distribution and its approximation.

Inference network: A neural network whose output can be used to construct an approximate posterior distribution when supplied with observational data \mathbf{Z} or summaries thereof.

$$\gamma^* = \arg \min_{\gamma} - \sum_{i=1}^N \log q(\boldsymbol{\theta}^{(i)}; \boldsymbol{\kappa}_{\gamma}(\mathbf{Z}^{(i)})), \quad (6)$$

where $\boldsymbol{\theta}^{(i)} \sim p(\boldsymbol{\theta})$ and $\mathbf{Z}^{(i)} \sim p(\mathbf{Z} | \boldsymbol{\theta}^{(i)})$. This approach to neural posterior inference has the following graphical representation:



Chan et al. (2018) let $q(\cdot; \boldsymbol{\kappa}_{\gamma}(\mathbf{Z}))$ be Gaussian with mean and precision both functions of the data $\mathbf{Z} \in \mathcal{Z}$. Papamakarios and Murray (2016) go a step further and let $q(\cdot; \boldsymbol{\kappa}_{\gamma}(\mathbf{Z}))$ be a Gaussian mixture. Further flexibility can be achieved by modelling $q(\cdot; \boldsymbol{\kappa}_{\gamma}(\mathbf{Z}))$ using a normalising flow. Here, the architecture used to construct $q(\cdot; \boldsymbol{\kappa}_{\gamma}(\mathbf{Z}))$ is often termed a conditional invertible neural network, since the invertible map constructed using the neural network has parameters determined by \mathbf{Z} ; see, for example, Ardizzone et al. (2019) and Radev et al. (2022).

From Equation 6 note that we must be able to evaluate $q(\boldsymbol{\theta}; \boldsymbol{\kappa}(\mathbf{Z}))$ for any $\boldsymbol{\theta} \in \Theta$ and $\mathbf{Z} \in \mathcal{Z}$ when training the neural network. This is possible using normalising flows, which are invertible by construction. However, invertibility can be a restrictive requirement, and a more flexible approximate distribution could be obtained by foregoing the need of density evaluation, and minimising a different objective function instead. To this end, Pacchiardi and Dutta (2022a) implicitly define the approximate distribution as a (generally non-invertible) conditional transformation of a random variable with a simple known distribution. Specifically, let \mathbf{W} be a latent variable with known distribution and construct a (generally non-invertible) neural network $\mathbf{g}_{\gamma}(\mathbf{Z}, \mathbf{W})$ with parameters γ that takes in both \mathbf{W} and the data \mathbf{Z} as input, and outputs $\boldsymbol{\theta}$. The variable \mathbf{W} establishes an implicit conditional distribution of $\boldsymbol{\theta}$ for a given \mathbf{Z} , which we denote as $q_{\gamma}(\boldsymbol{\theta}; \mathbf{Z})$. Then, instead of minimising the forward KL divergence, minimise an expected score, that is, solve

$$\gamma^* = \arg \min_{\gamma} \sum_{i=1}^N s(q_{\gamma}(\cdot; \mathbf{Z}^{(i)}), \boldsymbol{\theta}^{(i)}), \quad (7)$$

where $\boldsymbol{\theta}^{(i)} \sim p(\boldsymbol{\theta})$, $\mathbf{Z}^{(i)} \sim p(\mathbf{Z} | \boldsymbol{\theta}^{(i)})$, and $s(\cdot, \cdot)$ is a proper scoring rule (Gneiting et al., 2007), such as the energy score. The score is intractable (since $q_{\gamma}(\cdot; \mathbf{Z})$ cannot be evaluated), but for most scores of interest, one can estimate it for a given $\boldsymbol{\theta}^{(i)}$ and $\mathbf{Z}^{(i)}$ with a Monte Carlo approximation using samples from $q_{\gamma}(\cdot; \mathbf{Z}^{(i)})$. These, in turn, are straightforward to generate by sampling \mathbf{W} from its known distribution and computing $\mathbf{g}_{\gamma}(\mathbf{Z}^{(i)}, \mathbf{W})$ for each realisation of \mathbf{W} . The objective function in Equation 7 is a generalisation of that in Equation 6, and the two are identical if $s(\cdot, \cdot)$ is chosen to be the logarithmic score. The approach of Pacchiardi and Dutta (2022a) can thus be seen as a form of amortised approximate generalised Bayesian inference (Bissiri et al., 2016).

2.2.2 Reverse KL-divergence minimisation

Several methods to amortised inference minimise the expected reverse KL divergence between the true posterior density and the approximate posterior density,

$$\boldsymbol{\kappa}^*(\cdot) = \arg \min_{\boldsymbol{\kappa}(\cdot)} \mathbb{E}_{\mathbf{Z}} [\text{KL}(q(\boldsymbol{\theta}; \boldsymbol{\kappa}(\mathbf{Z})) \parallel p(\boldsymbol{\theta} | \mathbf{Z}))].$$

Invertible neural network:

A neural network whose architecture constrains it to the space of invertible mappings, often used to construct normalising flows.

Scoring rule: Real-valued function $s(\cdot, \cdot)$ taking distribution F and observation y as inputs, and returning the (negatively oriented) score when issuing a probabilistic forecast F and when y materialises.

Proper scoring rule: Defining $\bar{s}(F, Q) = \int s(F, y) dQ(y)$ for distributions F, Q , the scoring rule $s(\cdot, \cdot)$ is proper if $\bar{s}(Q, Q) \leq \bar{s}(F, Q)$ for all F, Q , and strictly proper if equality holds if and only if $F = Q$.

In this case, the approximate posterior distribution is referred to as a variational posterior distribution. As in Section 2.2.1, the parameters appearing in the variational posterior distribution are functions of the data, and $\kappa^*(\cdot)$ has the property that it minimises the KL divergence between the true posterior density $p(\boldsymbol{\theta} \mid \mathbf{Z})$ and the approximate posterior density for every $\mathbf{Z} \in \mathcal{Z}$, and hence yields optimal variational-posterior-distribution parameters for every $\mathbf{Z} \in \mathcal{Z}$ (see Section 2.3 for more details). We now replace $\kappa(\cdot)$ with an inference (neural) network $\kappa_\gamma(\cdot)$ (Mnih and Gregor, 2014). The neural network parameters γ can be optimised by minimising a Monte Carlo approximation of the expected reverse KL divergence,

$$\gamma^* \approx \arg \min_{\gamma} \sum_{k=1}^K \sum_{j=1}^J \left(\log q(\boldsymbol{\theta}^{(j)}; \kappa_\gamma(\mathbf{Z}^{(k)})) - \log(p(\mathbf{Z}^{(k)} \mid \boldsymbol{\theta}^{(j)})p(\boldsymbol{\theta}^{(j)})) \right), \quad (8)$$

where $\mathbf{Z}^{(k)} \sim p(\mathbf{Z})$, and $\boldsymbol{\theta}^{(j)} \sim q(\boldsymbol{\theta}; \kappa_\gamma(\mathbf{Z}^{(k)}))$. This objective now involves samples from both the model and the variational distribution, and generally requires one to invoke the so-called reparameterisation trick so that gradients with respect to parameters appearing in the inference network can be easily evaluated (Kingma and Welling, 2013).

Reparameterisation trick: Implicitly defining a parameter-dependent density function via a generative model through which gradients can be propagated. For example, re-expressing $X \sim \text{Gau}(\mu, \sigma^2)$ as $X = \mu + \sigma\epsilon$, $\epsilon \sim \text{Gau}(0, 1)$.

The concept of an inference network that returns parameters of a variational posterior distribution dates to at least Dayan et al. (1995), where $\kappa_\gamma(\cdot)$ was called a recognition model. It has since been used in various ways, for example for making inference on hyperparameters in Gaussian process models (Rehn, 2022) or for making inference on latent variables within deep Gaussian process models by Dai et al. (2015). Rezende and Mohamed (2015) and Kingma et al. (2016) add flexibility to the variational posterior by constructing $q(\cdot; \kappa_\gamma(\mathbf{Z}))$, $\mathbf{Z} \in \mathcal{Z}$, using a normalising flow, so that the inference network $\kappa_\gamma(\cdot)$ returns both the parameters of the flow’s starting distribution and those governing the flow (see Section S2 of the Supplementary Material). Variational auto-encoders (Kingma and Welling, 2013), comprise an inference network in the encoding stage that maps the data directly to parameters of the approximate posterior distribution (see Section S4.1 of the Supplementary Material). An in-depth treatment of amortised variational inference can be found in Zhang et al. (2018), Ganguly et al. (2023), and Margossian and Blei (2023).

Despite their appeal, variational learning networks as discussed in this section are limited by the fact that Equation 8 involves the likelihood term $p(\mathbf{Z} \mid \boldsymbol{\theta})$, whose evaluation thus has to be computationally feasible for any $\boldsymbol{\theta} \in \Theta$, $\mathbf{Z} \in \mathcal{Z}$. In some cases, this conditional density is known and tractable; for example, Equation 8 can be used to develop an amortised variational inference tool for solving inverse problems (Goh et al., 2019, Svendsen et al., 2023), where the conditional distribution of \mathbf{Z} is assumed to be Gaussian with mean equal to the output of a forward physics model that takes $\boldsymbol{\theta}$ as input. Zhang et al. (2023) also adopt amortised variational inference for estimating parameters in a hierarchical model for spatial extremes that has a tractable likelihood function. When the likelihood function is intractable, however, reverse-KL-minimisation techniques are often accompanied with neural-network approaches for likelihood approximation; see Section 4.1 for further details. We outline two related reverse-KL minimisation techniques that naturally avoid the need to explicitly define a likelihood term, the conditional generative adversarial network and the variational auto-encoder, in Section S4.1 of the Supplementary Material.

2.3 Existence of amortisers

The existence of a well-behaved globally optimal (i.e., amortised) estimator $\boldsymbol{\theta}^*(\cdot)$ or inference network $\kappa^*(\cdot)$ is not obvious, but has been established in the work of Brown and Purves (1973). Their existence result explains what is seen in experiments: there is often no advantage in using a non-amortised inferential method tailored for a specific dataset \mathbf{Z} over an amortised one that applies to any $\mathbf{Z} \in \mathcal{Z}$ (e.g., Radev et al., 2023a).

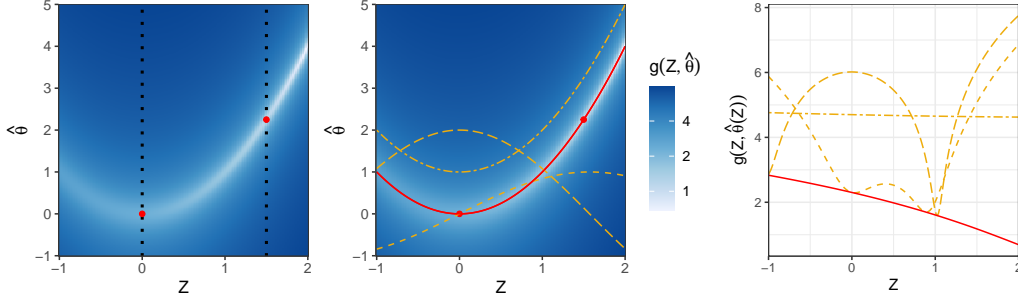


Figure 1: Illustration of the result of Brown and Purves (1973) in the context of amortised point estimation. (Left) A hypothetical nonnegative function $g(Z, \hat{\theta})$. To obtain the Bayes estimate $\hat{\theta}^*$ for a given Z (shown as red dots for $Z = 0$ and $Z = 1.5$), $g(Z, \hat{\theta})$ needs to be minimised along a slice at Z (dotted lines). (Centre) An absolutely measurable function $\hat{\theta}^*(\cdot)$ (red) that minimises $g(\cdot, \cdot)$ for any $Z \in \mathcal{Z}$, whose existence is proven by Brown and Purves (1973), and alternative estimators (orange). If known, $\hat{\theta}^*(\cdot)$ can be used to quickly find Bayes estimates for any $Z \in \mathcal{Z}$. (Right) The optimal estimator $\hat{\theta}^*(\cdot)$ minimises $g(Z, \hat{\theta}(Z))$ pointwise for each $Z \in \mathcal{Z}$, and therefore minimises $\int_{\mathcal{Z}} g(Z, \hat{\theta}(Z)) d\mu(Z)$ among all candidate estimators under any strictly positive measure $\mu(\cdot)$ on \mathcal{Z} .

Consider first the case of point estimation. The Bayes estimate $\hat{\theta}^*$ minimises the posterior expected loss given a specific \mathbf{Z} , or, equivalently, the forward KL divergence between the true posterior density and the density $q(\boldsymbol{\theta}; \hat{\boldsymbol{\theta}}) \propto \exp(-L(\boldsymbol{\theta}; \hat{\boldsymbol{\theta}}))$:

$$\hat{\boldsymbol{\theta}}^* = \arg \min_{\hat{\boldsymbol{\theta}}} \int_{\Theta} L(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}}) p(\boldsymbol{\theta} | \mathbf{Z}) d\boldsymbol{\theta} = \arg \min_{\hat{\boldsymbol{\theta}}} \text{KL}(p(\boldsymbol{\theta} | \mathbf{Z}) \| q(\boldsymbol{\theta}; \hat{\boldsymbol{\theta}})).$$

Define $g(\mathbf{Z}, \hat{\boldsymbol{\theta}}) \equiv \text{KL}(p(\boldsymbol{\theta} | \mathbf{Z}) \| q(\boldsymbol{\theta}; \hat{\boldsymbol{\theta}}))$, and suppose that for each $\mathbf{Z} \in \mathcal{Z}$ there is at least one $\hat{\boldsymbol{\theta}}^*$ such that $g(\mathbf{Z}, \hat{\boldsymbol{\theta}}^*) = \inf_{\hat{\boldsymbol{\theta}}} g(\mathbf{Z}, \hat{\boldsymbol{\theta}})$. Brown and Purves (1973) show that, under mild conditions, there exists an absolutely measurable function $\hat{\boldsymbol{\theta}}^*(\cdot)$ such that

$$g(\mathbf{Z}, \hat{\boldsymbol{\theta}}^*(\mathbf{Z})) = \inf_{\hat{\boldsymbol{\theta}}} g(\mathbf{Z}, \hat{\boldsymbol{\theta}}), \quad \text{for all } \mathbf{Z} \in \mathcal{Z}. \quad (9)$$

For illustration, consider the cost function $g(Z, \hat{\theta}) \equiv \log((3 - Z)^2 + 100(\hat{\theta} - Z^2)^2 + 1)$ for a model incorporating a single parameter θ and a single data point Z ; Figure 1, left panel, shows this hypothetical cost function. In a non-amortised setting, after observing Z , the Bayes estimate $\hat{\theta}^*$ is found by minimising $g(Z, \hat{\theta})$ with Z fixed to the observed data point. This potentially-burdensome optimisation problem must be repeated each time a new data point is collected. On the other hand, in an amortised setting, one finds a decision rule $\hat{\theta}^*(\cdot)$ that returns the Bayes estimate for any $Z \in \mathcal{Z}$ and, in this case, the Bayes estimator is $\hat{\theta}^*(Z) = Z^2$; see Figure 1, centre panel. Brown and Purves (1973) show that such a function exists and is well-behaved under mild conditions. Further, since $g(Z, \hat{\theta}(Z))$ is minimised pointwise by $\hat{\theta}^*(\cdot)$ for each $Z \in \mathcal{Z}$ (see Figure 1, right panel) and $g(\cdot, \cdot)$ is nonnegative, the integral of $g(Z, \hat{\theta}^*(Z))$ over Z under any strictly positive measure $\mu(\cdot)$ on \mathcal{Z} will be the smallest among any other possible candidate estimators; that is, $\hat{\theta}^*(\cdot) = \arg \min_{\hat{\theta}(\cdot)} \int_{\mathcal{Z}} g(Z, \hat{\theta}(Z)) d\mu(Z)$ for any strictly positive measure $\mu(\cdot)$ on \mathcal{Z} .

Returning to the multivariate setting and taking the integral to be the expectation with respect to the marginal likelihood yields the expected KL divergence in Equation 3. This important result is well known in decision theory (e.g., Casella and Berger, 2001), but it applies more generally to the other amortised frameworks discussed in this paper. In particular, for the inference network in Section 2.2.1 we have that $g(\mathbf{Z}, \boldsymbol{\kappa}) \equiv \text{KL}(p(\boldsymbol{\theta} | \mathbf{Z}) \| q(\boldsymbol{\theta}; \boldsymbol{\kappa}))$ while, for that in Section 2.2.2,

$g(\mathbf{Z}, \boldsymbol{\kappa}) \equiv \text{KL}(q(\boldsymbol{\theta}; \boldsymbol{\kappa}) \parallel p(\boldsymbol{\theta} \mid \mathbf{Z}))$, for $\mathbf{Z} \in \mathcal{Z}$. In both cases $g(\cdot, \cdot)$ is nonnegative, and therefore an amortised replacement $\boldsymbol{\kappa}^*(\cdot)$ for $\boldsymbol{\kappa}^* = \arg \min_{\boldsymbol{\kappa}} g(\mathbf{Z}, \boldsymbol{\kappa})$ exists, and can be found by minimising the expected KL divergence over the distribution of the data. Interestingly, one need not necessarily minimise the expected KL divergence with respect to $p(\mathbf{Z})$ to fit the inference network; as shown above, this is a matter of convenience that leads to tractable computations. Furthermore, from a practical standpoint, it is reasonable to expend computing effort to fit the function in regions of the sample space which are more likely to be visited; this leads to the amortisation gap being low in regions of interest, albeit large in areas of low probability. We briefly review sequential methods designed to mitigate this issue in Section S4.2 of the Supplementary Material.

3 NEURAL SUMMARY STATISTICS

Indirect inference: Fitting an auxiliary model, that is simpler than the target model, to data, and making inference on target parameters from the fitted auxiliary parameters.

Simulation-based inferential approaches typically require the pre-specification of summary statistics. These could be provided, for example, via indirect inference (e.g., Drovandi et al., 2011) or as point summaries of the posterior distribution (e.g., Fearnhead and Prangle, 2012). The methods outlined in Section 2 also employ summary statistics, either implicitly or explicitly. At one end of the spectrum, the first few layers in neural Bayes estimators or inference networks implicitly extract summaries that are then mapped to the point estimates or approximate-posterior-distribution parameters, respectively. Other methods, such as that by Radev et al. (2022) (Section 2.2.1) and the variational version proposed by Wqvist et al. (2021) (Section 2.2.2) employ summary networks that are differentiated from the main inference network, which then take summary statistics as input. At the other end of the spectrum, inference networks are trained using pre-constructed summary statistics as input. For example, Blum and Francois (2010), Creel (2017), and Gerber and Nychka (2021) ingest hand-crafted statistics into neural Bayes estimators rather than raw data. In all cases, the summary statistics are an informative low-dimensional representation of the data that allow for a parsimonious inference network that is potentially easier to train. In this section we briefly review methods that employ neural networks to automatically construct summary statistics for downstream inference. In Section 3.1, we review methods that employ neural networks to explicitly construct summary statistics for downstream inference; in Section 3.2, we discuss methods that integrate summary-statistic construction implicitly in an amortised inference framework; and, in Section 3.3, we discuss how the number of summary statistics might be chosen.

3.1 Explicit neural summary statistics

One of the most common neural summary statistics is the neural point estimator. Jiang et al. (2017), for example, use a neural Bayes estimator under squared error loss to generate posterior means quickly for use in ABC, while Dinev and Gutmann (2018) use neural Bayes estimators in conjunction with the ratio-based likelihood-free inference method of Thomas et al. (2022); similar approaches are also proposed by Creel (2017), Åkesson et al. (2022) and Albert et al. (2022). One can move beyond point estimators as summary statistics and train neural networks to return other summaries that are highly informative of $\boldsymbol{\theta}$. A common approach is to target sufficiency of the summary statistics $\mathbf{S}(\mathbf{Z})$ by maximising the mutual information between $\boldsymbol{\theta} \in \Theta$ and $\mathbf{S}(\mathbf{Z}) \in \mathcal{S}$, with \mathcal{S} the sample space corresponding to $\mathbf{S}(\mathbf{Z})$ (see Barnes et al. (2011) and the discussion by Barnes, Filippi and Stumpf in Fearnhead and Prangle (2012)). The mutual information, $\text{MI}(\boldsymbol{\theta}; \mathbf{S}(\mathbf{Z}))$, is defined as the KL divergence between the joint $p(\boldsymbol{\theta}, \mathbf{S}(\mathbf{Z}))$ and the product of marginals $p(\boldsymbol{\theta})p(\mathbf{S}(\mathbf{Z}))$. Intuitively, when the mutual information is small, then $\boldsymbol{\theta}$ and $\mathbf{S}(\mathbf{Z})$ are nearly independent, which implies that the summary statistic $\mathbf{S}(\mathbf{Z})$ is irrelevant for describing or predicting $\boldsymbol{\theta}$; the opposite behaviour holds when the mutual information is large. Assume now that we use a neural network to construct summary statistics from

data. For this so-called summary network $\mathbf{S}_\tau(\cdot)$ with network parameters τ , summary statistics are found by solving

$$\tau^* = \arg \max_{\tau} \text{MI}(\boldsymbol{\theta}; \mathbf{S}_\tau(\mathbf{Z})) = \arg \max_{\tau} \text{KL}(p(\boldsymbol{\theta}, \mathbf{S}_\tau(\mathbf{Z})) \parallel p(\boldsymbol{\theta})p(\mathbf{S}_\tau(\mathbf{Z}))). \quad (10)$$

The objective in Equation 10 is not straightforward to approximate using Monte Carlo techniques since it requires knowledge of the density function of the unknown summary statistics. To circumvent this issue, the mutual information neural estimator (MINE; Belghazi et al., 2018) replaces the KL divergence with its Donsker–Varadhan representation (Donsker and Varadhan, 1983). Writing $(\boldsymbol{\theta}', \mathbf{S}_\tau(\mathbf{Z}))' \sim p(\boldsymbol{\theta}, \mathbf{S}_\tau(\mathbf{Z}))$ and $(\tilde{\boldsymbol{\theta}}', \tilde{\mathbf{S}}_\tau(\mathbf{Z}))' \sim p(\boldsymbol{\theta})p(\mathbf{S}_\tau(\mathbf{Z}))$, we have

$$\text{MI}(\boldsymbol{\theta}; \mathbf{S}(\mathbf{Z})) = \sup_{T: \Theta \times \mathcal{S} \rightarrow \mathbb{R}} \mathbb{E}_{(\boldsymbol{\theta}', \mathbf{S}_\tau(\mathbf{Z}))'} [T(\boldsymbol{\theta}, \mathbf{S}_\tau(\mathbf{Z}))] - \log(\mathbb{E}_{(\tilde{\boldsymbol{\theta}}', \tilde{\mathbf{S}}_\tau(\mathbf{Z}))'} [\exp(T(\tilde{\boldsymbol{\theta}}, \tilde{\mathbf{S}}_\tau(\mathbf{Z})))]).$$

The appeal of the Donsker–Varadhan representation is that it does not require any knowledge of the distribution of the summary statistics when approximated using Monte Carlo methods. In MINE, the function $T(\cdot, \cdot)$ is modelled using a neural network $T_\zeta(\cdot, \cdot)$ that is trained in tandem with $\mathbf{S}_\tau(\cdot)$ via

$$(\tau^{*'}, \zeta^{*'})' = \arg \max_{(\tau', \zeta')'} \frac{1}{N} \sum_{i=1}^N T_\zeta(\boldsymbol{\theta}^{(i)}, \mathbf{S}_\tau(\mathbf{Z}^{(i)})) - \log \left(\frac{1}{N} \sum_{i=1}^N \exp(T_\zeta(\boldsymbol{\theta}^{(i)}, \mathbf{S}_\tau(\mathbf{Z}^{\pi(i)}))) \right),$$

where $\boldsymbol{\theta}^{(i)} \sim p(\boldsymbol{\theta})$, $\mathbf{Z}^{(i)} \sim p(\mathbf{Z} \mid \boldsymbol{\theta}^{(i)})$ and $\pi(\cdot)$ is a random permutation function used to ensure that $\boldsymbol{\theta}$ and $\mathbf{S}(\mathbf{Z})$ are (nearly) independent in the second term of the objective function.

Hjelm et al. (2018) found that optimisation routines were more stable when replacing the Donsker–Varadhan objective with one based on the Jensen–Shannon divergence, which can be seen as a robust version of the former:

$$\text{MI}(\boldsymbol{\theta}; \mathbf{S}(\mathbf{Z})) \approx \sup_{T: \Theta \times \mathcal{S} \rightarrow \mathbb{R}} \mathbb{E}_{(\boldsymbol{\theta}', \mathbf{S}_\tau(\mathbf{Z}))'} [-\text{sp}(-T(\boldsymbol{\theta}, \mathbf{S}_\tau(\mathbf{Z})))] - \mathbb{E}_{(\tilde{\boldsymbol{\theta}}', \tilde{\mathbf{S}}_\tau(\mathbf{Z}))'} [\text{sp}(T(\tilde{\boldsymbol{\theta}}, \tilde{\mathbf{S}}_\tau(\mathbf{Z})))],$$

where $\text{sp}(z) \equiv \log(1 + \exp(z))$ is the softplus function. The Jensen–Shannon divergence was employed to extract summary statistics for use with both classical simulation-based inference methods and neural based inference methods by Chen et al. (2021). Chen et al. (2023) propose to further tame the optimisation problem by using a slice technique that effectively breaks down the high-dimensional information objective into many smaller, lower-dimensional ones.

Other neural-network-based approaches to find summary statistics include that by Charnock et al. (2018) and de Castro and Dorigo (2019), who target statistics that maximise the determinant of the Fisher information matrix. Pacchiardi and Dutta (2022b) take yet a different approach and fit a general exponential family model in canonical form to simulated data. Their model comprises a summary statistics network $\mathbf{S}_\tau(\cdot)$ and a network for the canonical parameters that together form a so-called neural likelihood function. The two neural networks are fitted by minimising the Fisher divergence between the true likelihood function and the neural likelihood function. Although the neural likelihood function is inferred up to a normalising constant, and can be used with certain MCMC algorithms like the exchange algorithm, the main appeal of the method is that the extracted neural summary statistics are the sufficient statistics of the best (in a Fisher divergence sense) exponential family approximation to the true likelihood function.

3.2 Implicit neural summary statistics

The ability of neural networks to extract relevant summary statistics from the data is analogous to their ability to learn features that are useful for predicting an outcome, for

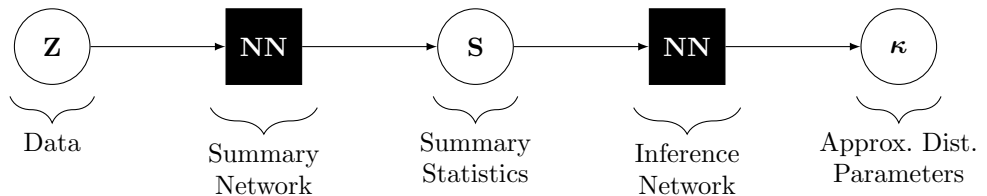
Sufficient statistic: A statistic $\mathbf{S}(\mathbf{Z})$ is sufficient for $\boldsymbol{\theta}$ if and only if $p(\mathbf{Z} \mid \boldsymbol{\theta}) = h(\mathbf{Z})g(\mathbf{S}(\mathbf{Z}); \boldsymbol{\theta})$ for some nonnegative functions $h(\cdot), g(\cdot)$ (Fisher–Neyman Factorisation Theorem).

Softplus function: The function $\text{sp}(z) \equiv \log(1 + \exp(z))$ commonly used as a smooth approximation to the rectified linear unit, $\text{ReLU}(z) = \max(0, z)$. Asymptotically, $\text{sp}(z) \approx \exp(z)$ as $z \rightarrow -\infty$ and $\text{sp}(z) \approx z$ as $z \rightarrow \infty$.

example in image recognition; this property is often referred to as representation learning. Feature extraction happens in the early layers of neural networks; similarly one can design networks for amortised inference with the view that the first few layers extract useful (i.e., highly informative) summary statistics.

All methods discussed in Section 2 can incorporate this notion by splitting the underlying neural-network architecture into two distinct sub-architectures: a summary network for extracting summary statistics from the data, and an inference network that now takes those summary statistics as input (instead of the raw data), and outputs estimates or distributional parameters. The summary network can be trained in tandem with this now-simplified inference network in an end-to-end fashion. The trained summary network then implicitly targets summary statistics that best enable the inference network to match the true posterior distribution in a KL sense; see, for example, Sainsbury-Dale et al. (2024) for this setup with neural Bayes estimators, Radev et al. (2022) for this setup in an approximate Bayes forward KL divergence setting, and Wqvist et al. (2021) in a reverse KL divergence setting. The inference network combined with a summary network has the following graphical representation (here, in an approximate Bayes setting):

Representation learning: The learning of features or summaries from data that are useful for downstream inference tasks.



When a set of hand-crafted statistics are known to be informative of the parameter vector, one may utilise both hand-crafted and implicitly learned (neural) summary statistics simultaneously (see, e.g., Sainsbury-Dale et al., 2024, Sec. 2.2.1).

3.3 Number of summary statistics

With both explicit and implicit summary statistics, the dimensionality of $\mathbf{S}(\cdot)$ (i.e., the number of summary statistics) is a design choice. Pacchiardi and Dutta (2022b) let the number of summary statistics equal the number of parameters while Chen et al. (2021) use double the amount. Chen et al. (2023) propose letting $\mathbf{S}(\cdot)$ be reasonably high-dimensional, fitting the neural networks, and then ordering the summary statistics according to their contribution to the mutual information; only the summary statistics that substantially contribute to the mutual information need to be retained. They note that setting this number to twice the dimension of θ appears to be sufficient in most applications. In the context of neural Bayes estimators, Gerber and Nychka (2021) and Sainsbury-Dale et al. (2024) set the number of (implicitly defined) summary statistics to 128. This number is much larger than the number of parameters in the models they considered, but the contribution of irrelevant summary statistics is automatically down-weighted during training. Generally, however, one might expect including an excessive number of summary statistics to be computationally wasteful and, in practice, some experimentation might be needed to determine a suitable number for a desired level of accuracy.

4 NEURAL LIKELIHOOD AND LIKELIHOOD-TO-EVIDENCE RATIO

Section 2 describes various popular methods for amortised neural posterior inference, while Section 3 outlines the role of summary statistics and how they can be automatically constructed from data. In this section we discuss methods for amortised approximation

of the likelihood function, $p(\mathbf{Z} | \boldsymbol{\theta})$, and a closely related quantity that is proportional to the likelihood function, the likelihood-to-evidence ratio, $r(\boldsymbol{\theta}, \mathbf{Z}) = p(\mathbf{Z} | \boldsymbol{\theta})/p(\mathbf{Z})$.

Amortised neural likelihood and likelihood-to-evidence-ratio approximators have several common motivations. First, the likelihood function is the cornerstone of frequentist inference, since it is needed for maximum likelihood estimation, while likelihood ratios of the form $p(\mathbf{Z} | \boldsymbol{\theta}_0)/p(\mathbf{Z} | \boldsymbol{\theta}_1) = r(\boldsymbol{\theta}_0, \mathbf{Z})/r(\boldsymbol{\theta}_1, \mathbf{Z})$ are central to hypothesis testing, model comparison, and naturally appear in the transition probabilities of most standard MCMC algorithms used for Bayesian inference. Second, amortised likelihood and likelihood-to-evidence-ratio approximators enable the straightforward treatment of independent and identically distributed (iid) replicates since, under independence of $\mathbf{Z}_1, \dots, \mathbf{Z}_m$, one has $p(\mathbf{Z}_1, \dots, \mathbf{Z}_m | \boldsymbol{\theta}) = \prod_{i=1}^m p(\mathbf{Z}_i | \boldsymbol{\theta})$, and the multiple-replicate likelihood-to-evidence ratio is of the form $p(\mathbf{Z}_1, \dots, \mathbf{Z}_m | \boldsymbol{\theta})/p(\mathbf{Z}_1, \dots, \mathbf{Z}_m) \propto \prod_{i=1}^m r(\mathbf{Z}_i, \boldsymbol{\theta})$. Hence, an amortised likelihood or likelihood-to-evidence-ratio approximator constructed with single-replicate datasets can be used to make (frequentist or Bayesian) inference with an arbitrary number of iid replicates. Third, as these amortised approximators target prior-free quantities, they are ideally placed for Bayesian inference that require the same model to be fitted multiple times under different prior distributions. However, as we shall explain below, these methods still require the user to define a proposal distribution $\tilde{p}(\boldsymbol{\theta})$ from which parameters will be sampled during the training stage, and this distribution determines the regions of the parameter space where the approximation will be most accurate.

We now review popular neural approaches for constructing amortised approximate likelihood functions (Section 4.1) and likelihood-to-evidence ratios (Section 4.2). There are also so-called doubly-approximate approaches to likelihood approximation, that instead of targeting the true likelihood function, target an approximation to it, such as a kernel density or Vecchia approximation; we review these in Section S4.3 of the Supplementary Material.

4.1 Neural Likelihood

4.1.1 Neural synthetic likelihood

A popular method to approximate a likelihood function is the synthetic likelihood framework of Wood (2010). In this framework, one replaces the likelihood function $p(\mathbf{Z} | \boldsymbol{\theta})$ with a synthetic one of the form $q(\mathbf{S}(\mathbf{Z}); \boldsymbol{\omega}(\boldsymbol{\theta}))$ based on summary statistics $\mathbf{S}(\mathbf{Z})$, where $\boldsymbol{\omega}(\boldsymbol{\theta})$ is a binding function linking the parameter vector $\boldsymbol{\theta}$ to the (approximate) density of $\mathbf{S}(\mathbf{Z})$ (e.g., Moores et al., 2015). Note that here $q(\mathbf{S}(\mathbf{Z}); \boldsymbol{\omega}(\boldsymbol{\theta}))$ refers to a generic (approximate) density for $\mathbf{S}(\mathbf{Z})$ evaluated at $\mathbf{S}(\mathbf{Z})$ itself. The summary statistics $\mathbf{S}(\cdot)$ are often modelled as Gaussian with mean parameter $\boldsymbol{\mu}(\boldsymbol{\theta})$ and covariance matrix $\boldsymbol{\Sigma}(\boldsymbol{\theta})$. Observe that if the binding function $\boldsymbol{\omega}(\boldsymbol{\theta}) = \{\boldsymbol{\mu}(\boldsymbol{\theta}), \boldsymbol{\Sigma}(\boldsymbol{\theta})\}$ is known, the synthetic likelihood is a form of amortised likelihood approximation, as it allows one to evaluate $p(\cdot | \boldsymbol{\theta})$ quickly for any $\boldsymbol{\theta}$. When a neural network is used to model the binding function, we obtain a neural synthetic likelihood. In the Gaussian case, we let $q(\mathbf{S}(\mathbf{Z}); \boldsymbol{\omega}_\eta(\boldsymbol{\theta}))$ be a Gaussian function with mean parameter $\boldsymbol{\mu}_\eta(\boldsymbol{\theta})$ and covariance matrix $\boldsymbol{\Sigma}_\eta(\boldsymbol{\theta})$, where the neural binding function is $\boldsymbol{\omega}_\eta(\boldsymbol{\theta}) = \{\boldsymbol{\mu}_\eta(\boldsymbol{\theta}), \boldsymbol{\Sigma}_\eta(\boldsymbol{\theta})\}$, and $\boldsymbol{\eta}$ are neural network parameters to be estimated. Often, the Gaussianity assumption for $\mathbf{S}(\mathbf{Z})$ is deemed too restrictive, and Radev et al. (2023a) instead model $q(\mathbf{S}(\mathbf{Z}); \boldsymbol{\omega}_\eta(\boldsymbol{\theta}))$ using normalising flows (Section S2 of the Supplementary Material).

The neural binding function can be trained via minimisation of the expected forward KL divergence (similar to other approaches in Section 2.2.1). In other words, one solves

$$\boldsymbol{\eta}^* = \arg \min_{\boldsymbol{\eta}} \mathbb{E}_{\boldsymbol{\theta}} [\text{KL}(p(\mathbf{S}(\mathbf{Z}) | \boldsymbol{\theta}) \| q(\mathbf{S}(\mathbf{Z}); \boldsymbol{\omega}_\eta(\boldsymbol{\theta}))), \quad (11)$$

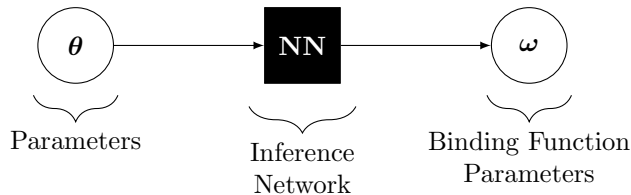
which leads to the empirical risk minimisation problem

$$\boldsymbol{\eta}^* = \arg \min_{\boldsymbol{\eta}} - \sum_{i=1}^N \log q(\mathbf{S}(\mathbf{Z}^{(i)}); \boldsymbol{\omega}_\eta(\boldsymbol{\theta}^{(i)})), \quad (12)$$

Synthetic likelihood function: The distribution of summary statistics (typically assumed to be Gaussian) with parameters that are a function of the underlying model parameters of interest. Often used as a replacement for an intractable likelihood function.

Binding function: Here used to refer to a function mapping the parameters of interest to the parameters of the approximate (synthetic) likelihood function.

where $\theta^{(i)} \sim \tilde{p}(\theta), i = 1, \dots, N$, are drawn from some proposal distribution (further discussed below), and $\mathbf{Z}^{(i)} \sim p(\mathbf{Z} | \theta^{(i)})$. The synthetic-likelihood network has the following graphical representation:



Once trained, the neural synthetic likelihood can be used instead of the true one in other inferential techniques, for example in amortised variational inference (Equation 8), as suggested by Wqvist et al. (2021). The resulting framework is an amortisation of the conventional variational-inference-with-synthetic-likelihood approach of Ong et al. (e.g., 2018); we demonstrate its use in Section 5.

4.1.2 Neural full likelihood

While Equation 11 leads to an amortised likelihood approximation framework, replacing the full likelihood with a synthetic likelihood may sometimes be undesirable (e.g., when one wishes to emulate the data \mathbf{Z} for any given θ). One can instead target the full likelihood function, which is obtained by setting $\mathbf{S}(\cdot)$ to be the identity function. Equations 11 and 12 then become

$$\eta^* = \arg \min_{\eta} \mathbb{E}_{\theta} [\text{KL}(p(\mathbf{Z} | \theta) \| q(\mathbf{Z}; \omega_{\eta}(\theta)))] = \arg \min_{\eta} \mathbb{E}_{(\theta', \mathbf{Z}')} [-\log(q(\mathbf{Z}; \omega_{\eta}(\theta')))], \quad (13)$$

and

$$\eta^* = \arg \min_{\eta} - \sum_{i=1}^N \log(q(\mathbf{Z}^{(i)}; \omega_{\eta}(\theta^{(i)}))), \quad (14)$$

respectively. Such a framework was considered by Lueckmann et al. (2017) (using Gaussian mixture density networks; see Bishop, 1995, Chapter 5) and Papamakarios et al. (2019) (using masked autoregressive flows; see Section S2 of the Supplementary Material).

4.1.3 Choice of proposal distribution

Unlike the methods in Section 2, we can use any arbitrary proposal distribution $\tilde{p}(\theta)$ that is supported over the parameter space (or a subset thereof) to obtain samples for Equation 12 or 14. While the choice of proposal distribution $\tilde{p}(\theta)$ does not matter in theory when solving Equation 11 or 13 for parameter values θ in its support, it is an important consideration in practice when solving Equation 12 or 14 because $\tilde{p}(\theta)$ determines the area of the parameter space where the parameters $\{\theta^{(i)}\}$ will be most densely sampled, and thus, where the likelihood approximation will be most accurate. In particular, if $\tilde{p}(\theta)$ is not fully supported over the parameter space, the accuracy of the neural likelihood approximation beyond the support of $\tilde{p}(\theta)$ will rely on the extrapolation ability of neural networks, which is known to be poor. Therefore, the proposal distribution should be sufficiently vague to cover the plausible values of θ , even though, if used in a Bayesian context, it does not need to be the same as the prior distribution $p(\theta)$. Papamakarios et al. (2019) develop a sequential training scheme aimed at improving simulation efficiency (and thus likelihood approximation accuracy) for specific data \mathbf{Z} ; since our review focuses on amortised inference, we defer discussion on sequential training methods to Section S4.2 of the Supplementary Material.

Gaussian mixture density network: A Gaussian mixture conditional probability distribution, where the means, variances, and mixing coefficients are the outputs of neural networks that take the conditioning variable as input.

4.1.4 Joint amortisation

Recently, Radev et al. (2023a) proposed to simultaneously approximate both the posterior distribution and the likelihood function within a single unified training regime, using two jointly trained normalising flows. The method, dubbed jointly amortised neural approximation (JANA), involves extending Equations 5 and 13 to the joint optimisation problem,

$$(\boldsymbol{\gamma}^{*'}, \boldsymbol{\tau}^{*'}, \boldsymbol{\eta}^{*'})' = \arg \max_{(\boldsymbol{\gamma}', \boldsymbol{\tau}', \boldsymbol{\eta}')'} \mathbb{E}_{(\boldsymbol{\theta}', \mathbf{Z}')'} [\log(q(\boldsymbol{\theta}; \boldsymbol{\kappa}_{\boldsymbol{\gamma}}(\mathbf{S}_{\boldsymbol{\tau}}(\mathbf{Z})))) + \log(q(\mathbf{Z}; \boldsymbol{\omega}_{\boldsymbol{\eta}}(\boldsymbol{\theta})))] - \lambda \cdot (\text{MMD}[p(\mathbf{S}_{\boldsymbol{\tau}}(\mathbf{Z})) \parallel \text{Gau}(\mathbf{0}, \mathbf{I})])^2, \quad \lambda > 0, \quad (15)$$

where the posterior network $q(\boldsymbol{\theta}; \boldsymbol{\kappa}_{\boldsymbol{\gamma}}(\mathbf{S}_{\boldsymbol{\tau}}(\mathbf{Z})))$ approximates the posterior density $p(\boldsymbol{\theta} \mid \mathbf{Z})$ through the summary network $\mathbf{S}_{\boldsymbol{\tau}}(\mathbf{Z})$, and where the likelihood network $q(\mathbf{Z}; \boldsymbol{\omega}_{\boldsymbol{\eta}}(\boldsymbol{\theta}))$ (or a synthetic likelihood version, based on $\mathbf{S}_{\boldsymbol{\tau}}(\mathbf{Z})$) approximates the likelihood $p(\mathbf{Z} \mid \boldsymbol{\theta})$; the rightmost term on the right-hand side of Equation 15 involves the maximum mean discrepancy (MMD) between the density of the summary statistics, $p(\mathbf{S}_{\boldsymbol{\tau}}(\mathbf{Z}))$, and the standard multivariate Gaussian density, $\text{Gau}(\mathbf{0}, \mathbf{I})$; this serves as a penalty to impose some probabilistic structure on the summary space and allows for the detection of model misspecification (for further details, see Radev et al., 2023a). Since JANA approximates both the (normalised) posterior density and the likelihood function, it also automatically yields an amortised approximation to the marginal likelihood $p(\mathbf{Z}) = p(\mathbf{Z} \mid \boldsymbol{\theta})p(\boldsymbol{\theta})/p(\boldsymbol{\theta} \mid \mathbf{Z})$, which is key for Bayesian model comparison and selection. Moreover, it also allows one to measure out-of-sample posterior predictive performance via the expected log-predictive density.

4.2 Neural Likelihood-to-Evidence Ratio

4.2.1 General framework

Another popular approach to amortised inference is the approximation of likelihood ratios. Some approaches (e.g., Cranmer et al., 2015, Baldi et al., 2016) target ratios of the form $p(\mathbf{Z} \mid \boldsymbol{\theta})/p(\mathbf{Z} \mid \boldsymbol{\theta}_{\text{ref}})$ for some arbitrary but fixed “reference” parameter $\boldsymbol{\theta}_{\text{ref}}$. In this review we focus on other methods that obviate the need for a reference parameter by instead targeting the likelihood-to-evidence ratio (Hermans et al., 2020),

$$r(\boldsymbol{\theta}, \mathbf{Z}) = p(\mathbf{Z} \mid \boldsymbol{\theta})/p(\mathbf{Z}). \quad (16)$$

This ratio can be approximated by solving a relatively straightforward binary classification problem, as we now show. Consider a binary classifier $c(\boldsymbol{\theta}, \mathbf{Z})$ that distinguishes dependent parameter–data pairs $(\boldsymbol{\theta}', \mathbf{Z}')' \sim p(\boldsymbol{\theta}, \mathbf{Z} \mid Y = 1) = p(\boldsymbol{\theta}, \mathbf{Z})$ from independent parameter–data pairs $(\tilde{\boldsymbol{\theta}}', \tilde{\mathbf{Z}})' \sim p(\boldsymbol{\theta}, \mathbf{Z} \mid Y = 0) = p(\boldsymbol{\theta})p(\mathbf{Z})$, where Y denotes the class label and where the classes are balanced. Then, we define the optimal classifier $c^*(\cdot, \cdot)$ as that which minimises the Bayes risk under binary cross-entropy loss, $L_{bce}(\cdot, \cdot)$,

$$\begin{aligned} c^*(\cdot, \cdot) &\equiv \arg \min_{c(\cdot, \cdot)} \sum_{y \in \{0,1\}} \Pr(Y = y) \int_{\Theta} \int_{\mathcal{Z}} p(\boldsymbol{\theta}, \mathbf{Z} \mid Y = y) L_{bce}(y, c(\boldsymbol{\theta}, \mathbf{Z})) d\mathbf{Z} d\boldsymbol{\theta} \\ &= \arg \max_{c(\cdot, \cdot)} \sum_{y \in \{0,1\}} \int_{\Theta} \int_{\mathcal{Z}} p(\boldsymbol{\theta}, \mathbf{Z} \mid Y = y) \{y \log(c(\boldsymbol{\theta}, \mathbf{Z})) + (1 - y) \log(1 - c(\boldsymbol{\theta}, \mathbf{Z}))\} d\mathbf{Z} d\boldsymbol{\theta} \\ &= \arg \max_{c(\cdot, \cdot)} \int_{\Theta} \int_{\mathcal{Z}} p(\boldsymbol{\theta}, \mathbf{Z}) \log c(\boldsymbol{\theta}, \mathbf{Z}) d\mathbf{Z} d\boldsymbol{\theta} + \int_{\Theta} \int_{\mathcal{Z}} p(\boldsymbol{\theta}) p(\mathbf{Z}) \log(1 - c(\boldsymbol{\theta}, \mathbf{Z})) d\mathbf{Z} d\boldsymbol{\theta} \\ &= \arg \max_{c(\cdot, \cdot)} \left[\mathbb{E}_{(\boldsymbol{\theta}', \mathbf{Z}')'} \log c(\boldsymbol{\theta}, \mathbf{Z}) + \mathbb{E}_{(\tilde{\boldsymbol{\theta}}', \tilde{\mathbf{Z}})'} \log(1 - c(\tilde{\boldsymbol{\theta}}, \tilde{\mathbf{Z}})) \right], \end{aligned} \quad (17)$$

where $\Pr(Y = y)$, $y \in \{0, 1\}$, denotes the class probability. It can be shown that $c^*(\boldsymbol{\theta}, \mathbf{Z}) = p(\boldsymbol{\theta}, \mathbf{Z})\{p(\boldsymbol{\theta}, \mathbf{Z}) + p(\boldsymbol{\theta})p(\mathbf{Z})\}^{-1}$ (Hermans et al., 2020, App. B), which can be recognised

Maximum mean discrepancy: A distance measure between two distributions. For two densities $p(\cdot)$ and $q(\cdot)$ we denote the discrepancy between $p(\cdot)$ and $q(\cdot)$ as $\text{MMD}[p(\mathbf{X}) \parallel q(\mathbf{Y})] = \sup_{g(\cdot) \in \mathcal{G}} \{\mathbb{E}_{\mathbf{X}}(g(\mathbf{X})) - \mathbb{E}_{\mathbf{Y}}(g(\mathbf{Y}))\}$, for $\mathbf{X} \sim p(\cdot)$, $\mathbf{Y} \sim q(\cdot)$ and a suitable class of real-valued functions \mathcal{G} (Gretton et al., 2012).

as the optimal Bayes classifier under equal class probability (i.e., with $\Pr(Y = 0) = \Pr(Y = 1) = 1/2$). Hence,

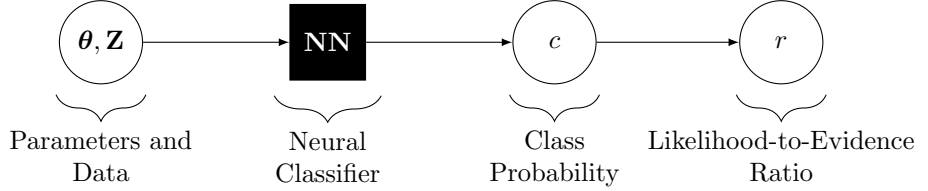
$$r(\boldsymbol{\theta}, \mathbf{Z}) = \frac{c^*(\boldsymbol{\theta}, \mathbf{Z})}{1 - c^*(\boldsymbol{\theta}, \mathbf{Z})}, \quad \boldsymbol{\theta} \in \Theta, \mathbf{Z} \in \mathcal{Z}. \quad (18)$$

In the typical case that $c^*(\cdot, \cdot)$ is unavailable, it can be approximated by adopting a flexible parametric representation for $c(\cdot, \cdot)$ and maximising a Monte Carlo approximation of the objective function in Equation 17. Specifically, let $c_\gamma(\cdot, \cdot)$ denote a binary classifier parameterised by γ , typically a neural network with a sigmoid output activation function, although other representations are possible (e.g., a logistic regression based on user-specified or learned summary statistics; Dinev and Gutmann, 2018, Thomas et al., 2022). Then, the Bayes classifier may be approximated by $c_{\gamma^*}(\cdot, \cdot)$, where

$$\gamma^* = \arg \max_{\gamma} \sum_{i=1}^N \left[\log\{c_\gamma(\boldsymbol{\theta}^{(i)}, \mathbf{Z}^{(i)})\} + \log\{1 - c_\gamma(\boldsymbol{\theta}^{(i)}, \mathbf{Z}^{(\pi(i))})\} \right], \quad (19)$$

where $\boldsymbol{\theta}^{(i)} \sim p(\boldsymbol{\theta})$, $\mathbf{Z}^{(i)} \sim p(\mathbf{Z} \mid \boldsymbol{\theta}^{(i)})$, and where $\pi(\cdot)$ is a random permutation of $\{1, \dots, N\}$. Figure 2 demonstrates this learning task on the simple model $Z \mid \theta \sim N(\theta, \theta^2)$, $\theta \sim U(0, 1)$, for $c_\gamma(\cdot, \cdot)$ a fully-connected neural network with six layers (each of width 64).

Once trained, the classifier $c_{\gamma^*}(\cdot, \cdot)$ may be used to quickly approximate the likelihood-to-evidence ratio via Equation 18, as summarised in the following graphical representation:



Inference based on an amortised approximate likelihood-to-evidence ratio may proceed as discussed in the introduction to Section 4, namely, in a frequentist setting via maximum likelihood estimation and likelihood ratios (e.g., Walchessen et al., 2023), and in a Bayesian setting by facilitating the computation of transition probabilities in Hamiltonian Monte Carlo and Markov chain Monte Carlo algorithms (e.g., Hermans et al., 2020, Begy and Schikuta, 2021). Furthermore, an approximate posterior density can be obtained via the identity $p(\boldsymbol{\theta} \mid \mathbf{Z}) = p(\boldsymbol{\theta})r(\boldsymbol{\theta}, \mathbf{Z})$, and sampled from using standard sampling techniques (e.g., Thomas et al. (2022); see also Section 5.2). Finally, if one also has an amortised likelihood approximator (Section 4.1), then one may approximate the model evidence via $p(\mathbf{Z}) = p(\mathbf{Z} \mid \boldsymbol{\theta})/r(\boldsymbol{\theta}, \mathbf{Z})$.

4.2.2 Variants of the target ratio

Several variants of the learning task described above have been investigated, with a variety of different aims. One of these aims is to safeguard against over-optimistic inferences when the true Bayes classifier $c^*(\cdot, \cdot)$ cannot be approximated well by the neural classifier $c_\gamma(\cdot, \cdot)$; in this situation it is widely accepted that one should err on the side of caution and make $c_\gamma(\cdot, \cdot)$ conservative (Hermans et al., 2022). Delaunoy et al. (2022) propose a way to do this by encouraging $c_\gamma(\cdot, \cdot)$ to satisfy the following ‘balance’ condition

$$\mathbb{E}_{(\boldsymbol{\theta}', \mathbf{Z}')'} \{c_\gamma(\boldsymbol{\theta}', \mathbf{Z}')\} + \mathbb{E}_{(\tilde{\boldsymbol{\theta}}', \tilde{\mathbf{Z}}')'} \{c_\gamma(\tilde{\boldsymbol{\theta}}', \tilde{\mathbf{Z}}')\} = 1; \quad (20)$$

it is straightforward to see that the Bayes classifier under equal class probability, $c^*(\cdot, \cdot)$, satisfies the condition. Now, one can show that any classifier satisfying Equation 20 also

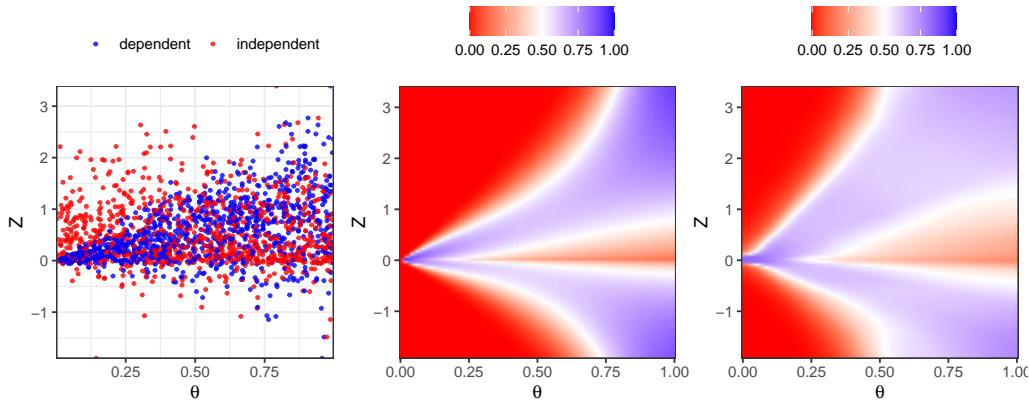


Figure 2: Illustration of amortised likelihood-to-evidence ratio approximation using the model $Z \mid \theta \sim N(\theta, \theta^2)$, $\theta \sim U(0, 1)$. (Left) Samples of dependent pairs $\{\theta, Z\} \sim p(\theta, Z)$ (blue) and independent pairs $\{\tilde{\theta}, \tilde{Z}\} \sim p(\theta)p(Z)$ (red). (Centre) Class probabilities from the exact Bayes classifier $c^*(\theta, Z) = p(\theta, Z)\{p(\theta, Z) + p(\theta)p(Z)\}^{-1}$, linked to the likelihood-to-evidence ratio via the relation $r(\theta, Z) = c^*(\theta, Z)\{1 - c^*(\theta, Z)\}^{-1}$. (Right) Class probabilities from an amortised neural approximation $c_{\gamma^*}(\cdot, \cdot)$ of the Bayes classifier.

satisfies $\mathbb{E}_{(\theta', \mathbf{Z}')'} \left[\frac{c^*(\theta, \mathbf{Z})}{c_{\gamma^*}(\theta, \mathbf{Z})} \right] \geq 1$ and $\mathbb{E}_{(\tilde{\theta}', \tilde{\mathbf{Z}})'}$ $\left[\frac{1 - c^*(\tilde{\theta}, \tilde{\mathbf{Z}})}{1 - c_{\gamma^*}(\tilde{\theta}, \tilde{\mathbf{Z}})} \right] \geq 1$, which informally means that it tends to produce class-probability estimates that are smaller (i.e., less confident and more conservative) than those produced by the (optimal) Bayes classifier. To encourage the trained estimator to satisfy the balancing condition, and hence be conservative, one may append to the objective function in Equation 17 the penalty term,

$$\lambda \left[\mathbb{E}_{(\theta', \mathbf{Z}')'} \{c(\theta, \mathbf{Z})\} + \mathbb{E}_{(\tilde{\theta}', \tilde{\mathbf{Z}})' } \{c(\tilde{\theta}, \tilde{\mathbf{Z}})\} - 1 \right]^2, \quad \lambda > 0, \quad (21)$$

which is equal to zero whenever the balancing condition is satisfied. Since the Bayes classifier with equal class probabilities is balanced, this penalty term only becomes relevant when the Bayes classifier cannot be approximated well using the neural network, as intended.

Another avenue of investigation is concerned with improving computational efficiency when making inference with amortised ratio approximators. For instance, approximation of likelihood ratios of the form $\tilde{r}(\theta_0, \theta_1, \mathbf{Z}) \equiv p(\mathbf{Z} \mid \theta_0)/p(\mathbf{Z} \mid \theta_1) = r(\theta_0, \mathbf{Z})/r(\theta_1, \mathbf{Z})$ requires two forward passes through a neural classifier $c_{\gamma^*}(\cdot, \cdot)$. Recently, however, Cobb et al. (2023) proposed to construct an amortised approximation of $\tilde{r}(\cdot, \cdot, \cdot)$ using a single neural network that takes as input two parameter vectors. This approach has the advantage of requiring only a single forward pass through a neural classifier, which can improve computational efficiency; however, this improvement can be offset if a larger (and therefore slower) neural network is required to learn the more complicated mapping. Finally, several approaches have been developed to construct amortised ratio approximators for a subset of θ . These marginal methods include introducing a binary mask as input to the ratio approximator that encodes the desired subset of parameters (Rozet and Louppe, 2021), and constructing a separate ratio approximator for each subset of interest (Miller et al., 2021, 2022).

5 SOFTWARE AND ILLUSTRATIVE EXAMPLE

Although amortised neural inference is a relatively new field, there are a number of easy-to-use software packages available. We outline these in Section 5.1 and demonstrate their

application through a simple illustrative example in Section 5.2 where the model parameters are easily inferred using MCMC. We choose this model in order to show the similarity between the inferences obtained using the neural methods and those obtained using MCMC. An additional illustration that considers a model for which MCMC cannot be used is given in Section S3 of the Supplementary Material. Code for these two examples is available at https://github.com/andrewzm/Amortised_Neural_Inference_Review/.

5.1 Software

Software availability for neural-network based inference has increased considerably in recent years. The majority of this software is written in **Python**, and leverage the deep-learning libraries **TensorFlow** (Abadi et al., 2016) and **PyTorch** (Paszke et al., 2019). Software in the numerical programming language **Julia** and its deep-learning library **Flux** (Innes, 2018) are also available. There are **R** interfaces to several of these libraries written in other languages. We note that software evolves quickly; the description we provide below is based on the software availability and functionality as of early 2024.

The package **NeuralEstimators** (Sainsbury-Dale et al., 2024) facilitates the construction of neural Bayes estimators (Section 2.1) using arbitrary loss functions, and likelihood-to-evidence ratio approximators (Section 4.2). The package is written in **Julia** and leverages the deep-learning library **Flux**, and is accompanied by a user-friendly **R** interface. The package is specially designed to deal with independent replicates of the statistical model, including independent replicates from spatial processes. The package also implements methods for bootstrap-based uncertainty quantification and for handling censored (Richards et al., 2023) and missing (Wang et al., 2023) data.

The package **sbi** (Tejero-Cantero et al., 2020), short for simulation-based inference, is a **Python** package built on **PyTorch**. It provides methods that target the posterior distribution (Section 2.2.1), the likelihood function (Section 4.1), or the likelihood-to-evidence ratio (Section 4.2), with posterior inference using the likelihood or likelihood-to-evidence ratio facilitated with MCMC sampling, rejection sampling, or (non-amortised) variational inference. The package implements both amortised and sequential methods, the latter of which are aimed at improving simulation efficiency (see Section S4.2 of the Supplementary Material).

The package **LAMPE** (Rozet et al., 2021), short for likelihood-free amortised posterior estimation, is another **PyTorch** package that focuses on amortised methods for approximating the posterior distribution (Section 2.2.1) or the likelihood-to-evidence ratio (Section 4.2), with posterior inference using the likelihood-to-evidence ratio facilitated with MCMC or nested sampling. The package allows for training data to be stored on disk and dynamically loaded on demand, instead of being cached in memory; this technique facilitates the use of very large datasets that do not fit in memory.

The package **BayesFlow** (Radev et al., 2023b) is built on **TensorFlow** and focuses on amortised inference. It implements methods for approximating the posterior distribution (Section 2.2.1) and the likelihood function (Section 4.1), possibly in a joint manner (Radev et al., 2023a); detecting model misspecification (Schmitt et al., 2024); and for performing amortised model comparisons via posterior model probabilities or Bayes Factors. Recently, neural Bayes estimators (Section 2.1) have also been incorporated into the package. The package is well documented and provides a user-friendly application programming interface.

The package **swyft** (Miller et al., 2022) is built on **PyTorch**, and implements methods for estimating likelihood-to-evidence ratios for subsets of the parameter vector, using both amortised and sequential training algorithms. The package also allows for data to be stored on disk and dynamically loaded, in order to facilitate the use of datasets that are too large to fit in memory.

Finally, we note that the above software packages provide the tools necessary to easily train the required neural networks from scratch, but that they are general-purpose. We

Censored data: Data whose precise values are not known, but which are known to lie in some interval. Multivariate censored data often lead to intractable likelihood functions.
Missing data: Data that have not been observed, and that one often needs to predict or impute.

anticipate that, in the future, amortised inference tools may be bundled with packages primarily designed around a specific modelling framework. For instance, a package that implements a specific statistical model could include a pre-trained neural network, or several such neural networks, designed to make fast inferences for that specific statistical model. This could be a natural evolution since there are incentives for the developers of such packages to make inference straightforward for increased accessibility and uptake.

5.2 Illustrative Example

In this section we showcase a few of the principal techniques discussed in this review. We consider a model with only one unknown parameter, where the neural networks are straightforward to train without the need for high-end GPUs, and for which MCMC is straightforward to implement for comparison. We consider a spatial Gaussian process with exponential covariance function with unit variance and unknown length scale $\theta > 0$. We assume that data \mathbf{Z} are on a 16×16 gridding, \mathcal{D}^G , of the unit square $\mathcal{D} = [0, 1]^2$. The process model is therefore given by $\mathbf{Z} \mid \theta \sim \text{Gau}(\mathbf{0}, \Sigma(\theta))$, where $\Sigma \equiv (\exp(-\|\mathbf{s}_i - \mathbf{s}_j\|/\theta) : \mathbf{s}_i, \mathbf{s}_j \in \mathcal{D}^G)$, and we let $\theta \sim \text{Unif}(0, 0.6)$. We train the neural networks on 160,000 draws from $p(\theta)$ and $p(\mathbf{Z} \mid \theta)$, and test the neural methods on another 1,000 independent draws. We consider the more complicated case of an inverted max-stable process, for which MCMC is impossible, in Section S3 of the Supplementary Material.

The techniques we consider, their acronyms, and their software implementations are outlined in Table S1 in the Supplementary Material. The gold standard is provided by a (non-amortised) Metropolis–Hastings algorithm, **MCMC**, run on the 1,000 test datasets. We implement a neural Bayes estimator, **NBE** (Section 2.1), using **NeuralEstimators**, targeting the posterior mean, the posterior 5th percentile and the posterior 95th percentile. We implement amortised posterior inference via forward KL minimisation, **fKL** (Section 2.2.1), using **BayesFlow** with a normalising flow model for the posterior density constructed using affine coupling blocks (see Section S2 of the Supplementary Material). In order to ensure that the approximate posterior densities are zero outside the prior support of $[0, 0.6]$, we instead make inference on $\tilde{\theta} \equiv \Phi^{-1}(\theta/0.6) \sim \text{Gau}(0, 1)$, and obtain samples from the posterior distribution of θ through the inverse $\theta = 0.6 \cdot \Phi(\tilde{\theta})$, where $\Phi(\cdot)$ is the cumulative distribution function of the standard normal distribution.

We implement three types of reverse-KL methods using **TensorFlow**, **rKL1**, **rKL2**, and **rKL3**, that differ in the likelihood term used. All three target an approximate Gaussian posterior distribution of $\tilde{\theta} \equiv \log(\theta/(0.6 - \theta))$; samples from the approximate (non-Gaussian) posterior distribution of θ are obtained by simply back-transforming samples drawn from the approximate Gaussian distribution of $\tilde{\theta}$. The first variant, **rKL1**, uses the true likelihood (Section 2.2.2); the second one, **rKL2**, uses a synthetic likelihood constructed using an “expert” summary statistic, in this case given by the mean of the squared differences between neighbouring pixels in \mathbf{Z} (Section 4.1.1); and the third one, **rKL3**, uses a synthetic likelihood constructed using a summary statistic found by maximising mutual information (Section 3.1). Finally, we also consider the neural ratio estimation method, **NRE**, implemented in the package **sbi** (Section 4.2); we use the amortised ratio to quickly evaluate the posterior distribution on a fine gridding of the parameter space, from which we then draw samples. For all approaches we use similar architectures, largely based on a two-layer CNN; for more details see Table S1 in the Supplementary Material. All the neural networks needed a similar amount of time to train (on the order of a few minutes) using the CPU of a standard laptop.

We summarise results on the test set of size 1,000 in Figure 3 and Table S2 in the Supplementary Material. In Figure 3a we plot the summary statistics for **rKL2** and **rKL3**, along with the inferred mean $\mu_{\tau}(\theta)$ and the $2\sigma_{\tau}(\theta)$ interval used to construct the synthetic likelihood. The expert summary statistic is non-linear, and, for large θ , a broad range of parameters lead to a small expert summary statistic. The statistic constructed

Pre-trained neural network:

A neural network trained for a generic task, that can be used as a good initial condition when training for specialised tasks (e.g., in our context, for a slightly different model or prior distribution).

Affine coupling block: An invertible transformation, where the input is partitioned into two blocks. One of the blocks undergoes an affine transformation that depends on the other block; see Section S2 of the Supplementary Material.

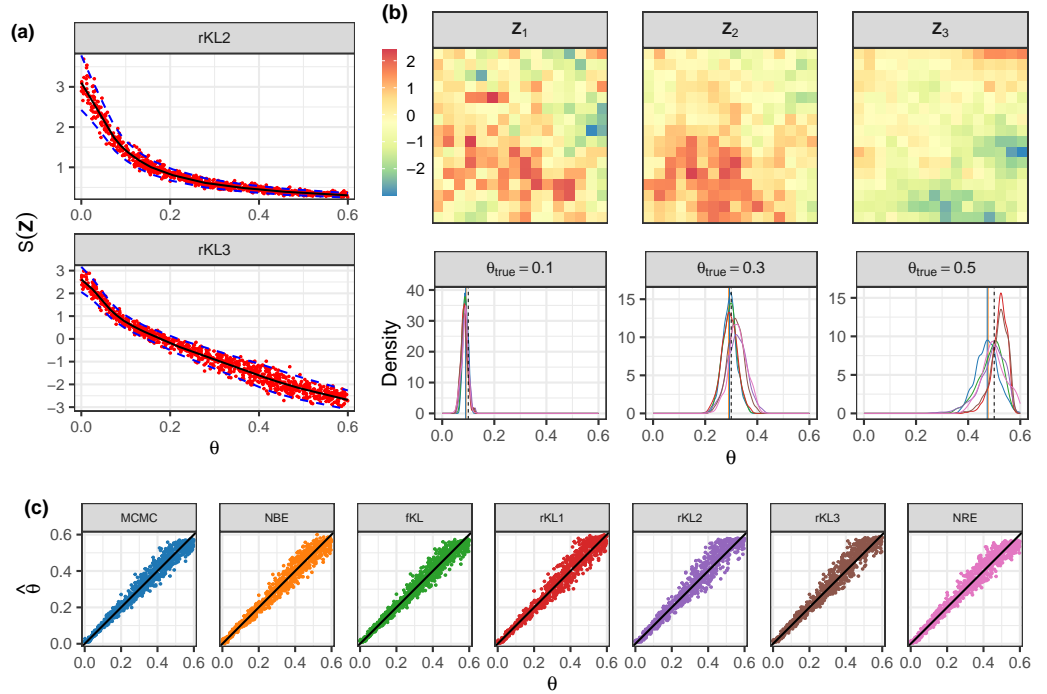


Figure 3: Results from the illustration of Section 5.2. Panel (a) plots the summary statistics of the test data (red), together with the mean (solid black line) and 2σ bands constructed from the fitted neural binding functions. The top sub-panel shows the expert summary statistic (used in **rKL2**), while the bottom sub-panel shows the statistic constructed by maximising mutual information (used in **rKL3**). Panel (b) shows the results from all methods on three test datasets. The top sub-panels show the data, while the bottom sub-panels show estimates (solid vertical lines, **MCMC** posterior mean and **NBE**, which largely overlap) and the inferred posterior distributions (solid lines, all methods except **NBE**), as well as the true parameter value (dashed line). Panel (c) plots parameter point estimates (in this case the posterior mean) against the true parameter value in the test dataset for all methods. The colours of the lines in panel (b) correspond to the colours used for the different methods in panel (c).

by maximising the mutual information on the other hand is largely linear over the entire support of $p(\theta)$; as we will soon show, this leads to slightly better performance. Figure 3b shows the results from applying the methods to three (test) spatial fields, $\mathbf{Z}_1, \mathbf{Z}_2, \mathbf{Z}_3$. All methods perform well, with posterior variance increasing with θ as expected. Differences between the approaches are mostly evident for larger θ where inferences are more uncertain. Figure 3c plots the posterior means against the true values; all methods again perform as expected, with lower variance for small θ and large variance for large θ , where the length scale is large in comparison to the size of the spatial domain.

In Table S2 in the Supplementary Material we score the methods based on the root-median-squared prediction error (RMSPE), the median-absolute prediction error (MAPE), the median 90% interval score (MIS90), the 90% coverage (COV90), and the median continuous ranked probability score (MCRPS); see Gneiting and Raftery (2007) for more details on these scoring rules and Lueckmann et al. (2021) for other benchmarks that could be used to assess simulation-based inference methods (note, we use the median instead of the mean since the distribution of the scores is highly skewed; see Figure S2 in the Supplementary Material), as well as an extensive comparison study featuring some of the methods considered in this review. Again, there is very little difference between all approaches, but we see that those methods that do not place any restrictions on the approximate posterior (NBE, fKL, NRE) yield close-to-nominal empirical coverages. The rKL methods we implemented have a restricted class of approximate posterior distributions, and have lower coverage. This further highlights the importance of flexible approximate posterior distributions when making reliable variational inference (Rezende and Mohamed, 2015). We also note that replacing the likelihood in rKL1 with a synthetic one (rKL2 and rKL3) leads to slightly higher RMSPE, with the network using the mutual-information-constructed summary statistic (rKL3) faring better than the one with an expert summary statistic (rKL2), yet with lower coverage.

Although all methods perform slightly worse than the gold-standard MCMC due to either the amortisation gap (NBE, fKL, NRE) or the use of an inflexible approximation to the posterior distribution (in this case the rKL variants), or both, the significance of amortisation lies in the speed these usable inferences can be obtained, and the general class of models these neural methods apply to. In this illustrative example, one run of MCMC required around one minute of computing time to generate 24,000 samples (which we thinned down to 1,000), while the neural methods only required a few dozens of milliseconds to yield the approximate posterior inferences.

6 FURTHER READING

In Section S4 of the Supplementary Material we briefly discuss some important topics related to the principal subject of this review: in Section S4.1 we discuss conditional generative adversarial networks and variational auto-encoders in the context of amortised inference; in Section S4.2 we discuss sequential methods and semi-amortisation for narrowing the amortisation gap within regions of interest in the parameter/sample space; in Section S4.3 we discuss so-called doubly approximate likelihood approaches; in Section S4.4 we discuss simulation-based calibration of confidence intervals for neural inference methods; in Section S4.5 we discuss neural inference under model misspecification; in Section S4.6 we discuss neural amortisation of complex data types; in Section S4.7 we discuss neural model selection; and finally in Section S4.8 we give a non-exhaustive list of applications that have benefitted to date from amortised inference.

7 EPILOGUE

Amortised neural inference methods are relatively new, and yet they have already shown enormous potential for making highly accurate and extremely fast statistical inference

Continuous ranked probability score: Given a forecast distribution F and an observation y , the continuous ranked probability score is $\text{CRPS}(F, y) \equiv \int_{-\infty}^{\infty} (F(x) - \mathbb{1}\{y \leq x\})^2 dx$.

Interval score: Given the lower and upper quantiles, y_l and y_u , of a forecast distribution, and an observation y , the $(1 - \alpha) \times 100\%$ interval score is $\text{IS}_{\alpha}(y_l, y_u; y) \equiv (y_u - y_l) + \frac{2}{\alpha}(y_l - y)\mathbb{1}\{y < y_l\} + \frac{2}{\alpha}(y - y_u)\mathbb{1}\{y > y_u\}$.

in a wide range of applications with models that are either defined explicitly but are computationally intractable (or take overly long to fit with classical likelihood-based approaches), or are defined implicitly through a stochastic generator. The amortised nature of the neural methods presented in this review paper, made possible through the use of neural networks that can be quickly evaluated, opens a new era in statistical inference, an era where one can construct and share pre-trained inference tools that can be used off-the-shelf with new data at a negligible computational cost.

A number of challenges remain for future research. From a theoretical viewpoint, there is a need to gain further understanding on the asymptotic properties of neural estimators (e.g., consistency and asymptotic distribution of estimators, their rates of convergence as a function of the network architecture and the size of the training set) in order to facilitate their design and establish rigorous implementation strategies. From a methodological viewpoint, there are several avenues that remain to be explored. For example, for a wide range of hierarchical models, one is also interested in recovering distributions of random effects; this functionality could be achieved by combining amortised methods to make inference on latent variables (e.g., Liu and Liu, 2019) with the parameter inference methods discussed in this review. Additionally, basic rejection and importance sampling ABC methods can also be amortised (e.g., Mestdagh et al., 2019), and it is not yet clear what advantages these have, if any, over neural-network approaches. Finally, from an application viewpoint, we believe there are several more traditional branches of statistics that can benefit from these advances; these include analyses of survey data, crop yield, and experimental design. Amortised neural inference approaches also enable online frequentist or Bayesian inference where data arrive sequentially, and where parameters need to be tracked. Overall, we anticipate that several fields in statistics will, in the near future, be positively impacted by this relatively new enabling technology.

DISCLOSURE STATEMENT

The authors are not aware of any affiliations, memberships, funding, or financial holdings that might be perceived as affecting the objectivity of this review.

ACKNOWLEDGEMENTS

All authors were supported by the King Abdullah University of Science and Technology (KAUST) Opportunity Fund Program ORFS-2023-OFP-5550.2. Further, this material is based upon work supported by the Air Force Office of Scientific Research under award number FA2386-23-1-4100 (AZ-M). MS-D’s research was additionally supported by an Australian Government Research Training Program Scholarship, a 2022 Statistical Society of Australia (SSA) topup scholarship, and the KAUST Office of Sponsored Research (OSR) under Award No. OSR-CRG2020-4394 and RH’s baseline funds.

References

- Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2016). TensorFlow: Large-scale machine learning on heterogeneous systems. arXiv:1603.04467 [cs.DC].
- Albert, C., Ulzega, S., Ozdemir, F., Perez-Cruz, F., and Mira, A. (2022). Learning summary statistics for Bayesian inference with autoencoders. arXiv:2201.12059 [cs.LG].

- Ardizzone, L., Kruse, J., Rother, C., and Köthe, U. (2019). Analyzing inverse problems with invertible neural networks. In *Proceedings of the 7th International Conference on Learning Representations (ICLR 2019)*. New Orleans, LA: OpenReview. <https://openreview.net/forum?id=rJed6j0cKX>.
- Baldi, P., Cranmer, K., Faucett, T., Sadowski, P., and Whiteson, D. (2016). Parameterized neural networks for high-energy physics. *The European Physical Journal C*, 76:235.
- Barnes, C., Filippi, S., Stumpf, M., and Thorne, T. (2011). Considerate approaches to achieving sufficiency for ABC model selection. arXiv:1106.6281 [stat.CO].
- Begy, V. and Schikuta, E. (2021). Error-guided likelihood-free MCMC. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7, Piscataway, NJ. IEEE.
- Belghazi, M. I., Baratin, A., Rajeshwar, S., Ozair, S., Bengio, Y., Courville, A., and Hjelm, D. (2018). Mutual information neural estimation. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pages 531–540. PMLR.
- Besag, J. (1986). On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society B*, 48:259–302.
- Bishop, C. (1995). *Neural Networks for Pattern Recognition*. Clarendon Press, Oxford, UK.
- Bissiri, P. G., Holmes, C. C., and Walker, S. G. (2016). A general framework for updating belief distributions. *Journal of the Royal Statistical Society B*, 78:1103–1130.
- Blum, M. G. and Francois, O. (2010). Non-linear regression models for approximate Bayesian computation. *Statistics and Computing*, 20:63–73.
- Blum, M. G., Nunes, M. A., Prangle, D., and Sisson, S. A. (2013). A comparative review of dimension reduction methods in approximate Bayesian computation. *Statistical Science*, 28:189–208.
- Brown, L. D. and Purves, R. (1973). Measurable selections of extrema. *The Annals of Statistics*, 1:902–912.
- Casella, G. and Berger, R. (2001). *Statistical Inference*. Duxbury, Belmont, CA, second edition.
- Chan, J., Perrone, V., Spence, J., Jenkins, P., Mathieson, S., and Song, Y. (2018). A likelihood-free inference framework for population genetic data using exchangeable neural networks. In *Proceedings of the 32nd Conference on Neural Information Processing Systems (NeurIPS 2018)*, pages 8594–8605, Red Hook, NY. Curran.
- Charnock, T., Lavaux, G., and Wandelt, B. D. (2018). Automatic physical inference with information maximizing neural networks. *Physical Review D*, 97:083004.
- Chen, Y., Gutmann, M. U., and Weller, A. (2023). Is learning summary statistics necessary for likelihood-free inference? In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, pages 4529–4544. PMLR.
- Chen, Y., Zhang, D., Gutmann, M. U., Courville, A., and Zhu, Z. (2021). Neural approximate sufficient statistics for implicit models. In *Proceedings of the 9th International Conference on Learning Representations (ICLR 2021)*. Virtual: OpenReview. <https://openreview.net/pdf?id=SRDuJssQud>.

- Cobb, A. D., Matejek, B., Elenius, D., Roy, A., and Jha, S. (2023). Direct amortized likelihood ratio estimation. arXiv:2311.10571 [stat.ML].
- Cranmer, K., Brehmer, J., and Louppe, G. (2020). The frontier of simulation-based inference. *Proceedings of the National Academy of Sciences*, 117:30055–30062.
- Cranmer, K., Pavez, J., and Louppe, G. (2015). Approximating likelihood ratios with calibrated discriminative classifiers. arXiv:1506.02169 [stat.ML].
- Creel, M. (2017). Neural nets for indirect inference. *Econometrics and Statistics*, 2:36–49.
- Cremer, C., Li, X., and Duvenaud, D. (2018). Inference suboptimality in variational autoencoders. In *Proceedings of the 35th International Conference on Machine Learning (ICML 2018)*, volume 80, pages 1078–1086. PMLR.
- Cressie, N. (2018). Mission CO₂ntrol: A statistical scientist’s role in remote sensing of atmospheric carbon dioxide. *Journal of the American Statistical Association*, 113:152–168.
- Dai, Z., Damianou, A., González, J., and Lawrence, N. (2015). Variational auto-encoded deep Gaussian processes. arXiv:1511.06455 [cs.LG].
- David, L., Bréon, F.-M., and Chevallier, F. (2021). XCO₂ estimates from the OCO-2 measurements using a neural network approach. *Atmospheric Measurement Techniques*, 14:117–132.
- Davison, A. C., Padoan, S. A., and Ribatet, M. (2012). Statistical modeling of spatial extremes. *Statistical Science*, 27:161–186.
- Dayan, P., Hinton, G. E., Neal, R. M., and Zemel, R. S. (1995). The Helmholtz machine. *Neural Computation*, 7:889–904.
- de Castro, P. and Dorigo, T. (2019). INFERNO: Inference-aware neural optimisation. *Computer Physics Communications*, 244:170–179.
- Delaunoy, A., Hermans, J., Rozet, F., Wehenkel, A., and Louppe, G. (2022). Towards reliable simulation-based inference with balanced neural ratio estimation. In *Proceedings of the 36th Conference on Neural Information Processing Systems (NeurIPS 2022)*, pages 20025–20037, Red Hook, NY. Curran.
- Diggle, P. J. and Gratton, R. J. (1984). Monte Carlo methods of inference for implicit statistical models. *Journal of the Royal Statistical Society B*, 46:193–212.
- Dinev, T. and Gutmann, M. U. (2018). Dynamic likelihood-free inference via ratio estimation (DIRE). arXiv:1810.09899 [stat.ML].
- Donsker, M. D. and Varadhan, S. S. (1983). Asymptotic evaluation of certain Markov process expectations for large time. IV. *Communications on Pure and Applied Mathematics*, 36:183–212.
- Drovandi, C. and Frazier, D. T. (2022). A comparison of likelihood-free methods with and without summary statistics. *Statistics and Computing*, 32:42.
- Drovandi, C. C. (2018). ABC and indirect inference. In Sisson, S. A., Fan, Y., and Beaumont, M., editors, *Handbook of Approximate Bayesian Computation*, pages 179–209. Chapman & Hall/CRC Press, Boca Raton, FL.
- Drovandi, C. C., Pettitt, A. N., and Faddy, M. J. (2011). Approximate Bayesian computation using indirect inference. *Journal of the Royal Statistical Society C*, 60:317–337.

- Dyer, J., Cannon, P., Farmer, J. D., and Schmon, S. M. (2024). Black-box Bayesian inference for agent-based models. *Journal of Economic Dynamics and Control*, 161:104827.
- Fan, J. and Yao, Q. (1998). Efficient estimation of conditional variance functions in stochastic regression. *Biometrika*, 85:645–660.
- Fasiolo, M., Pya, N., and Wood, S. N. (2016). A comparison of inferential methods for highly nonlinear state space models in ecology and epidemiology. *Statistical Science*, 31:96–118.
- Fearnhead, P. and Prangle, D. (2012). Constructing summary statistics for approximate Bayesian computation: semi-automatic approximate Bayesian computation. *Journal of the Royal Statistical Society B*, 74:419–474.
- Fengler, A., Govindarajan, L., Chen, T., and Frank, M. J. (2021). Likelihood approximation networks (LANs) for fast inference of simulation models in cognitive neuroscience. *eLife*, 10:e65074.
- Ganguly, A., Jain, S., and Watchareeruetai, U. (2023). Amortized variational inference: A systematic review. *Journal of Artificial Intelligence Research*, 78:167–215.
- Gerber, F. and Nychka, D. (2021). Fast covariance parameter estimation of spatial Gaussian process models using neural networks. *Stat*, 10:e382.
- Gershman, S. and Goodman, N. (2014). Amortized inference in probabilistic reasoning. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 36, pages 517–522.
- Gneiting, T., Balabdaoui, F., and Raftery, A. E. (2007). Probabilistic forecasts, calibration and sharpness. *Journal of the Royal Statistical Society B*, 69:243–268.
- Gneiting, T. and Raftery, A. E. (2007). Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102:359–378.
- Goh, H., Sherifdeen, S., Wittmer, J., and Bui-Thanh, T. (2019). Solving Bayesian inverse problems via variational autoencoders. In *Proceedings of the 2nd Mathematical and Scientific Machine Learning Conference (MSML 2022)*, volume 145, pages 386–425. PMLR.
- Gourieroux, C., Monfort, A., and Renault, E. (1993). Indirect inference. *Journal of Applied Econometrics*, 8:S85–S118.
- Grazian, C. and Fan, Y. (2020). A review of approximate Bayesian computation methods via density estimation: Inference for simulator-models. *Wiley Interdisciplinary Reviews: Computational Statistics*, 12:e1486.
- Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., and Smola, A. (2012). A kernel two-sample test. *Journal of Machine Learning Research*, 13:723–773.
- Hermans, J., Begy, V., and Louppe, G. (2020). Likelihood-free MCMC with amortized approximate ratio estimators. In *Proceedings of the 37th International Conference on Machine Learning (ICML 2020)*, volume 119, pages 4239–4248. PMLR.
- Hermans, J., Delaunoy, A., Rozet, F., Wehenkel, A., Begy, V., and Louppe, G. (2022). A crisis in simulation-based inference? Beware, your posterior approximations can be unfaithful. *Transactions on Machine Learning Research*. OpenReview. <https://openreview.net/pdf?id=LHAbHkt6Aq>.

- Hjelm, R. D., Fedorov, A., Lavoie-Marchildon, S., Grewal, K., Bachman, P., Trischler, A., and Bengio, Y. (2018). Learning deep representations by mutual information estimation and maximization. In *Proceedings of the 6th International Conference on Learning Representations (ICLR 2018)*. Vancouver, Canada: OpenReview. <https://openreview.net/forum?id=Bklr3j0cKX>.
- Hoel, D. G. and Mitchell, T. J. (1971). The simulation, fitting, and testing of a stochastic cellular proliferation model. *Biometrics*, 27:191–199.
- Hornik, K. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, 2:359–366.
- Huser, R. and Wadsworth, J. L. (2022). Advances in statistical modeling of spatial extremes. *Wiley Interdisciplinary Reviews: Computational Statistics*, 14:e1537.
- Innes, M. (2018). Flux: Elegant machine learning with Julia. *Journal of Open Source Software*, 3:602.
- Jiang, B., Wu, T.-y., Zheng, C., and Wong, W. H. (2017). Learning summary statistic for approximate Bayesian computation via deep neural network. *Statistica Sinica*, 27:1595–1618.
- Kingma, D. P., Salimans, T., Jozefowicz, R., Chen, X., Sutskever, I., and Welling, M. (2016). Improving variational autoencoders with inverse autoregressive flow. In *Proceedings of the 30th Conference on Neural Information Processing Systems (NeurIPS 2016)*, pages 4743–4751, Red Hook, NY. Curran.
- Kingma, D. P. and Welling, M. (2013). Auto-encoding variational Bayes. arXiv:1312.6114 [stat.ML].
- Kullback, S. and Leibler, R. A. (1951). On information and sufficiency. *Annals of Mathematical Statistics*, 22:79–86.
- Lenzi, A., Bessac, J., Rudi, J., and Stein, M. L. (2023). Neural networks for parameter estimation in intractable models. *Computational Statistics & Data Analysis*, 185:107762.
- Liu, L. and Liu, L. (2019). Amortized variational inference with graph convolutional networks for Gaussian processes. In *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS 2019)*, volume 89, pages 2291–2300. PMLR.
- Liu, S., Sun, X., Ramadge, P. J., and Adams, R. P. (2020). Task-agnostic amortized inference of Gaussian process hyperparameters. In *Proceedings of the 34th Conference on Neural Information Processing Systems (NeurIPS 2020)*, pages 21440–21452, Red Hook, NY. Curran.
- Luccioni, A. S., Viguier, S., and Ligozat, A.-L. (2023). Estimating the carbon footprint of BLOOM, a 176B parameter language model. *Journal of Machine Learning Research*, 24:1–15.
- Lueckmann, J.-M., Boelts, J., Greenberg, D., Goncalves, P., and Macke, J. (2021). Benchmarking simulation-based inference. In *Proceedings of the 24th International Conference on Artificial Intelligence and Statistics (AISTATS 2021)*, volume 130, pages 343–351. PMLR.
- Lueckmann, J.-M., Goncalves, P. J., Bassetto, G., Öcal, K., Nonnenmacher, M., and Macke, J. H. (2017). Flexible statistical inference for mechanistic models of neural dynamics. In *Proceedings of the 31st Conference on Neural Information Processing Systems (NeurIPS 2017)*, pages 1290–1300, Red Hook, NY. Curran.

- Margossian, C. C. and Blei, D. M. (2023). Amortized variational inference: When and why? arXiv:2307.11018 [stat.ML].
- Mestdagh, M., Verdonck, S., Meers, K., Loossens, T., and Tuerlinckx, F. (2019). Prepaid parameter estimation without likelihoods. *PLoS Computational Biology*, 15:e1007181.
- Miller, B. K., Cole, A., Forré, P., Louppe, G., and Weniger, C. (2021). Truncated marginal neural ratio estimation. In *Proceedings of the 35th Conference on Neural Information Processing Systems (NeurIPS 2021)*, pages 129–143, Red Hook, NY. Curran.
- Miller, B. K., Cole, A., Weniger, C., Nattino, F., Ku, O., and Grootes, M. W. (2022). swyft: Truncated marginal neural ratio estimation in Python. *Journal of Open Source Software*, 7:4205.
- Mnih, A. and Gregor, K. (2014). Neural variational inference and learning in belief networks. In *Proceedings of the 31st International Conference on Machine Learning (ICML 2014)*, volume 22, pages 1791–1799. PMLR.
- Moores, M. T., Drovandi, C. C., Mengersen, K., and Robert, C. P. (2015). Pre-processing for approximate Bayesian computation in image analysis. *Statistics and Computing*, 25:23–33.
- Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective*. MIT Press, Cambridge, MA.
- Ong, V. M., Nott, D. J., Tran, M.-N., Sisson, S. A., and Drovandi, C. C. (2018). Variational Bayes with synthetic likelihood. *Statistics and Computing*, 28:971–988.
- Pacchiardi, L. and Dutta, R. (2022a). Likelihood-free inference with generative neural networks via scoring rule minimization. arXiv:2205.15784 [stat.CO].
- Pacchiardi, L. and Dutta, R. (2022b). Score matched neural exponential families for likelihood-free inference. *Journal of Machine Learning Research*, 23:1–71.
- Papamakarios, G. and Murray, I. (2016). Fast ε -free inference of simulation models with Bayesian conditional density estimation. In *Proceedings of the 30th Conference on Neural Information Processing Systems (NeurIPS 2016)*, pages 1036–1044, Red Hook, NY. Curran.
- Papamakarios, G., Sterratt, D., and Murray, I. (2019). Sequential neural likelihood: Fast likelihood-free inference with autoregressive flows. In *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS 2019)*, volume 89, pages 837–848. PMLR.
- Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. (2019). PyTorch: An imperative style, high-performance deep learning library. In *Proceedings of the 33rd Conference on Neural Information Processing Systems (NeurIPS 2019)*, pages 7994–8005. Curran, Red Hook, NY.
- Radev, S. T., Mertens, U. K., Voss, A., Ardizzone, L., and Köthe, U. (2022). BayesFlow: Learning complex stochastic models with invertible neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 33:1452–1466.
- Radev, S. T., Schmitt, M., Pratz, V., Picchini, U., Köthe, U., and Buerkner, P.-C. (2023a). JANA: Jointly amortized neural approximation of complex Bayesian models. In *Proceedings of the 39th Conference on Uncertainty in Artificial Intelligence (UAI 2023)*, volume 216, pages 1695–1706. PMLR.

- Radev, S. T., Schmitt, M., Schumacher, L., Elsemüller, L., Pratz, V., Schälte, Y., Köthe, U., and Bürkner, P.-C. (2023b). BayesFlow: Amortized Bayesian workflows with neural networks. *Journal of Open Source Software*, 8:5702.
- Rehn, A. (2022). Amortized Bayesian inference of Gaussian process hyperparameters. Master’s Thesis, University of Helsinki.
- Rezende, D. and Mohamed, S. (2015). Variational inference with normalizing flows. In *Proceedings of the 32nd International Conference on Machine Learning (ICML 2015)*, volume 37, pages 1530–1538. PMLR.
- Richards, J., Sainsbury-Dale, M., Zammit-Mangion, A., and Huser, R. (2023). Likelihood-free neural Bayes estimators for censored peaks-over-threshold models. arXiv:2306.15642 [stat.ME].
- Ross, G. (1971). Stochastic model fitting by evolutionary operation. In *A Symposium of the British Ecological Society*, pages 297–308.
- Rozet, F., Delaunoy, A., Miller, B., et al. (2021). LAMPE: Likelihood-free amortized posterior estimation. <https://pypi.org/project/lampe>.
- Rozet, F. and Louppe, G. (2021). Arbitrary marginal neural ratio estimation for simulation-based inference. arXiv:2110.00449 [cs.LG].
- Rudi, J., Bessac, J., and Lenzi, A. (2022). Parameter estimation with dense and convolutional neural networks applied to the FitzHugh–Nagumo ODE. In *Proceedings of the 2nd Mathematical and Scientific Machine Learning Conference (MSML 2022)*, volume 145, pages 781–808. PMLR.
- Sainsbury-Dale, M., Richards, J., Zammit-Mangion, A., and Huser, R. (2023). Neural Bayes estimators for irregular spatial data using graph neural networks. arXiv:2310.02600 [stat.ME].
- Sainsbury-Dale, M., Zammit-Mangion, A., and Huser, R. (2024). Likelihood-free parameter estimation with neural Bayes estimators. *The American Statistician*, 78:1–14.
- Schmitt, M., Bürkner, P.-C., Köthe, U., and Radev, S. T. (2024). Detecting model misspecification in amortized Bayesian inference with neural networks. In Köthe, U. and Rother, C., editors, *Pattern Recognition. DAGM GCPR 2023*, pages 541–557, Cham, Switzerland. Springer.
- Siahkoobi, A., Rizzuti, G., Orozco, R., and Herrmann, F. J. (2023). Reliable amortized variational inference with physics-based latent distribution correction. *Geophysics*, 88:297–322.
- Sisson, S. A., Fan, Y., and Beaumont, M. (2018). *Handbook of Approximate Bayesian Computation*. Chapman & Hall/CRC Press, Boca Raton, FL.
- Svendsen, D. H., Hernandez-Lobato, D., Martino, L., Laparra, V., Moreno-Martinez, A., and Camps-Valls, G. (2023). Inference over radiative transfer models using variational and expectation maximization methods. *Machine Learning*, 112:921–937.
- Tejero-Cantero, A., Boelts, J., Deistler, M., Lueckmann, J.-M., Durkan, C., Goncalves, P. J., Greenberg, D. S., and Macke, J. H. (2020). sbi: A toolkit for simulation-based inference. *Journal of Open Source Software*, 5:2505.
- Thomas, O., Dutta, R., Corander, J., Kaski, S., and Gutmann, M. U. (2022). Likelihood-free inference by ratio estimation. *Bayesian Analysis*, 17:1–31.

- Walchessen, J., Lenzi, A., and Kuusela, M. (2023). Neural likelihood surfaces for spatial processes with computationally intensive or intractable likelihoods. arXiv:2305.04634 [stat.ME].
- Wang, Z., Hasenauer, J., and Schälte, Y. (2023). Missing data in amortized simulation-based neural posterior estimation. bioRxiv:2023.01.09.523219.
- Wiqvist, S., Frellsen, J., and Picchini, U. (2021). Sequential neural posterior and likelihood approximation. arXiv:2102.06522 [stat.ML].
- Wood, S. N. (2010). Statistical inference for noisy nonlinear ecological dynamic systems. *Nature*, 466:1102–1104.
- Zammit-Mangion, A. and Wikle, C. K. (2020). Deep integro-difference equation models for spatio-temporal forecasting. *Spatial Statistics*, 37:100408.
- Zhang, C., Bütepage, J., Kjellström, H., and Mandt, S. (2018). Advances in variational inference. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41:2008–2026.
- Zhang, L., Ma, X., Wikle, C. K., and Huser, R. (2023). Flexible and efficient spatial extremes emulation via variational autoencoders. arXiv:2307.08079 [stat.ML].
- Åkesson, M., Singh, P., Wrede, F., and Hellander, A. (2022). Convolutional neural networks as summary statistics for approximate Bayesian computation. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 19:3353–3365.

Supplementary Material

The supplementary material is organised as follows. In Section S1 we discuss neural network architectures most commonly used for amortised parameter inference. In Section S2 we outline normalising flows and two popular ones: affine coupling flows and masked autoregressive flows. In Section S3 we present a comparative study on a model of spatial extremes. In Section S4 we discuss additional topics. Finally, in Section S5 we provide additional figures and tables that complement the main text.

S1 NEURAL NETWORK ARCHITECTURES

Neural networks are nonlinear functions that are used for function approximation, typically in high-dimensional settings. The functional form of a neural network is known as its architecture. Throughout the review we generally treat the neural network as a black-box function characterised by parameters that need to be optimised according to some criterion. The purpose of this section is to give a brief insight into the black boxes that are most pertinent to this field. Since neural networks are used in various ways when making amortised inference, here we keep the notation general and denote the input to the neural network as \mathbf{X} and the output as \mathbf{Y} , which we assume are real-valued and of different dimension. The type of neural network used to capture a specific input/output relationship largely depends on the multivariate structure of the input \mathbf{X} .

Unstructured data: When the input \mathbf{X} has no discernible multivariate structure (i.e., it is not spatially, temporally, or otherwise structured), a commonly-used architecture is the multi-layer perceptron (MLP, e.g., Bishop, 1995, Chapter 5). For some $L > 1$ layers, the MLP is a hierarchy of so-called fully-connected layers,

$$\begin{aligned} \mathbf{Y} &= \varphi_L(\mathbf{W}_L \mathbf{h}_{L-1} + \mathbf{b}_L), \\ \mathbf{h}_l &= \varphi_l(\mathbf{W}_l \mathbf{h}_{l-1} + \mathbf{b}_l), \quad l = 2, \dots, L-1, \\ \mathbf{h}_1 &= \varphi_1(\mathbf{W}_1 \mathbf{X} + \mathbf{b}_1), \end{aligned} \tag{S1}$$

where, for $l = 1, \dots, L$, $\gamma_l \equiv (\text{vec}(\mathbf{W}_l)', \mathbf{b}_l)'$ are the neural-network parameters at layer l comprising both the matrix of weights \mathbf{W}_l and vector of biases \mathbf{b}_l of appropriate dimension, and $\varphi_l(\cdot)$ are non-linear activation functions applied elementwise over their arguments. Popular activation functions include the sigmoid function, the hyperbolic tangent function, and the rectified linear unit, which returns its input multiplied by the Heaviside step function. The quantities $\mathbf{h}_l, l = 1, \dots, L-1$, are often referred to as hidden states. Training the network is the process of estimating the neural-network parameters $\gamma \equiv (\gamma_1', \dots, \gamma_L)'$ by minimising some empirical risk function, usually via stochastic gradient descent. The MLP is highly parameterised, and is therefore most often used as a small component of the more parsimonious architectures discussed below.

Gridded data: The mainstream neural network architecture used with gridded data is the convolutional neural network (CNN, LeCun et al., 1998). Assume, for ease of exposition, that $\mathbf{X} \in \mathbb{R}^n$ are ordered pixel values of a 1-D image, where n is the number of pixels. Despite their name, CNNs typically implement cross-correlations and not convolutions: an output of the first layer of a 1-D CNN is

$$h_{1,i} = \varphi_1 \left(\sum_{j=1}^{n_K} \tilde{X}_{i+j-1} K_{1,j} + b_{1,i} \right); \quad i = 1, \dots, n,$$

where $\mathbf{h}_1 \equiv (h_{1,i}, \dots, h_{1,n})'$, $\mathbf{K}_1 \equiv (K_{1,1}, \dots, K_{1,n_K})'$ is an n_K -dimensional discrete kernel, and $\tilde{\mathbf{X}}$ is an appropriately padded extension of \mathbf{X} of size $n + n_K - 1$, where typically

the padded values are equal to zero. Subsequent convolutional layers take the same form and training then involves estimating $\gamma_l \equiv (\mathbf{K}'_l, \mathbf{b}'_l)'$ for $l = 1, \dots, L$. The kernels play the role of weight sharing, and thus CNN layers are considerably more parsimonious than fully-connected ones. In practice, inputs are often 2-D, and many “convolutions” are done in each layer, so that a set of convolved quantities known as channels, are input to the subsequent layer. When one has multiple 2-D image channels input to a layer, 3-D convolutional filters are used to do the convolution. The final layer of a CNN is typically a small MLP that maps the output of the convolutional layers to the output. See, for example, Gerber and Nychka (2021), Lenzi et al. (2023), or Richards et al. (2023) for examples of CNN-based neural Bayes estimator used to make amortised inference with gridded spatial data.

Time series/spatio-temporal data: Time series data are also highly structured, and the 1-D CNN described above is often used for time series data (e.g., Dinev and Gutmann, 2018, Åkesson et al., 2022, Rudi et al., 2022); extensions have been proposed for using them for estimating parameters in a spatio-temporal setting (e.g., de Bézenac et al., 2018, Zammit-Mangion and Wikle, 2020). However, CNNs do not take into account the temporal ordering of the data, and do not explicitly account for temporal dynamics. A classic architecture that takes temporal ordering into account is the recurrent neural network (RNN, Williams and Zipser, 1989). A layer in a vanilla RNN is given by

$$\mathbf{h}_l = \varphi_l(\mathbf{W}\mathbf{h}_{l-1} + \mathbf{b} + \mathbf{U}\mathbf{x}_{l-1}),$$

where l now denotes the position in a sequence (i.e., the time point), \mathbf{x}_{l-1} is the data (e.g., a covariate) at $l - 1$, and \mathbf{U} is a matrix that maps the input to the hidden states. Since the matrices \mathbf{W} and \mathbf{U} , and the bias \mathbf{b} , are time invariant, the RNN is reasonably parsimonious. A popular variant of the RNN is the long-short-term-memory (LSTM, Hochreiter and Schmidhuber, 1997) model, which is designed to capture longer-range dependencies. Despite successes in various machine learning applications, both RNNs and LSTMs are being superseded by transformer-based architectures that incorporate so-called self-attention modules intended to capture long-range dependencies and that are amenable to more efficient training algorithms (Vaswani et al., 2017).

Graphical data and irregular spatial data: Graph neural networks (GNNs) are designed for use with data that can be represented as a graph, where entities (nodes) and their relationships (edges) encapsulate the data structure (e.g., Zhang et al., 2019, Zhou et al., 2020, Wu et al., 2021). When performing “graph-level regression”, where the entire graph (e.g., data) is associated with some fixed-dimensional vector (e.g., unknown model parameters) that we wish to infer, a GNN typically consists of three modules: a propagation module, a readout module, and a mapping module. The propagation module consists of several layers of graph operations. For layer l , node j , a possible GNN propagation layer is given by

$$\mathbf{h}_{l,j} = \varphi_l \left(\mathbf{W}_{l,1}\mathbf{h}_{l-1,j} + \frac{1}{|\mathcal{N}(j)|} \sum_{j' \in \mathcal{N}(j)} a_{\beta_l}(j, j') \mathbf{W}_{l,2}\mathbf{h}_{l-1,j'} + \mathbf{b}_l \right),$$

where $\mathbf{h}_{l,j}$ is the hidden state of the j th node at layer l , $\mathcal{N}(j)$ denotes the set of neighbours of node j , and $a_{\beta_l}(j, j')$ is a weight that is a function of properties associated with nodes j and j' (e.g., the spatial distance between the nodes), parameterised by β_l . The trainable parameters at layer l are therefore given by $\gamma_l \equiv (\text{vec}(\mathbf{W}_{l,1})', \text{vec}(\mathbf{W}_{l,2})', \mathbf{b}'_l, \beta'_l)'$. The readout module maps the hidden state at the final layer of the propagation module to a vector of fixed dimension by summarising each dimension separately (e.g., by averaging), while the mapping module maps this fixed length vector to the output \mathbf{Y} , typically using an MLP. See Sainsbury-Dale et al. (2023) for an example of GNN-based neural Bayes estimator used to make amortised inference with spatial data observed at irregular locations.

Exchangeable data: Inferences made from exchangeable data (e.g., data with independent replicates) should be invariant to permutations of the data. This permutation-invariance property is satisfied by design with the neural network architecture commonly known as DeepSets (Zaheer et al., 2017). Consider an input consisting of m replicates, $\mathbf{X} \equiv (\mathbf{X}'_1, \dots, \mathbf{X}'_m)'$. The DeepSets architecture is:

$$\mathbf{Y} = \phi(\mathbf{a}(\{\psi(\mathbf{X}_i) : i = 1, \dots, m\})), \quad (\text{S1})$$

where $\phi(\cdot)$ is typically an MLP, $\mathbf{a}(\cdot)$ is a permutation-invariant set function which combines its set elements through elementwise operations (e.g., elementwise addition or average), and $\psi(\cdot)$ is a neural network whose structure depends on that of each input datum (e.g., $\psi(\cdot)$ could be a CNN if \mathbf{X}_i is gridded spatial data). Note that $\psi(\cdot)$ is common to all replicates in the dataset; any function of the form of Equation S1 is guaranteed to be permutation invariant. See Chan et al. (2018) for a DeepSets-based approximate posterior distribution for making inference with population-genetics data, and Sainsbury-Dale et al. (2024) for a DeepSets-based neural Bayes estimator used to make amortised inference with replicated spatial data.

S2 NORMALISING FLOWS

Approximate posterior distributions or likelihood functions in amortised inference are often modelled using normalising flows, which map a target density to a simple, reference, density. Consider an approximate posterior distribution for illustration. Let $\boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^d$ be the target variable of interest whose density we wish to model, and let $\tilde{\boldsymbol{\theta}} \in \tilde{\Theta} \subseteq \mathbb{R}^d$ denote a reference variable whose density is known; here we take this to be a spherical Gaussian density. A normalising flow is a diffeomorphism $\mathbf{g} : \Theta \rightarrow \tilde{\Theta}$ parameterised by some parameters $\boldsymbol{\kappa}$ such that $\mathbf{g}(\boldsymbol{\theta}; \boldsymbol{\kappa}) = \tilde{\boldsymbol{\theta}} \sim \text{Gau}(\mathbf{0}, \mathbf{I})$. By the change-of-variables formula, the target log-density is

$$\log q(\boldsymbol{\theta}; \boldsymbol{\kappa}) = -\frac{d}{2} \log(2\pi) - \frac{1}{2} \|\mathbf{g}(\boldsymbol{\theta}; \boldsymbol{\kappa})\|^2 + \log |\det \nabla \mathbf{g}(\boldsymbol{\theta}; \boldsymbol{\kappa})|. \quad (\text{S2})$$

Normalising flows shift the burden from modelling a density over a high-dimensional space to that of modelling a flexible differentiable invertible mapping $\mathbf{g}(\cdot; \boldsymbol{\kappa})$. Recall from Section 2 that amortisation can be imbued into the learning problem by making the parameters $\boldsymbol{\kappa}$ themselves outputs of neural networks, which we denote as $\boldsymbol{\kappa}_\gamma(\mathbf{Z})$.

There are several types of normalising flows and excellent reviews include those by Kobzyev et al. (2020) and Papamakarios et al. (2021). Here, we outline two popular ones used in an amortised inferential setting:

Affine coupling flows: An affine coupling flow is a composition of several affine coupling blocks constructed as follows. Consider the l th coupling block in a flow comprising L such blocks, that takes in $\tilde{\boldsymbol{\theta}}^{(l-1)}$ as input, and that outputs $\tilde{\boldsymbol{\theta}}^{(l)}$, with $\tilde{\boldsymbol{\theta}}^{(0)} \equiv \boldsymbol{\theta}$ and $\tilde{\boldsymbol{\theta}}^{(L)} \equiv \tilde{\boldsymbol{\theta}}$. Each affine coupling block partitions the input $\tilde{\boldsymbol{\theta}}^{(l-1)}$ into two disjoint components, $\{\tilde{\boldsymbol{\theta}}_1^{(l-1)}, \tilde{\boldsymbol{\theta}}_2^{(l-1)}\}$, and then outputs

$$\begin{aligned} \tilde{\boldsymbol{\theta}}_1^{(l)} &= \tilde{\boldsymbol{\theta}}_1^{(l-1)}, \\ \tilde{\boldsymbol{\theta}}_2^{(l)} &= \tilde{\boldsymbol{\theta}}_2^{(l-1)} \odot \exp(\boldsymbol{\kappa}_{\gamma,1}^{(l)}(\tilde{\boldsymbol{\theta}}_1^{(l-1)}, \mathbf{Z})) + \boldsymbol{\kappa}_{\gamma,2}^{(l)}(\tilde{\boldsymbol{\theta}}_1^{(l-1)}, \mathbf{Z}), \end{aligned}$$

where $\boldsymbol{\kappa}_\gamma(\cdot) \equiv \{\boldsymbol{\kappa}_{\gamma,1}^{(1)}(\cdot), \dots, \boldsymbol{\kappa}_{\gamma,1}^{(L)}(\cdot)\}$ with $\boldsymbol{\kappa}_\gamma^{(l)} \equiv \{\boldsymbol{\kappa}_{\gamma,1}^{(l)}(\cdot), \boldsymbol{\kappa}_{\gamma,2}^{(l)}(\cdot)\}$, $l = 1, \dots, L$, are (generic, not necessarily invertible) neural networks that take in both the data and subsets of the input/output variables. The mapping of each block is invertible, with inverse given by,

$$\begin{aligned} \tilde{\boldsymbol{\theta}}_1^{(l-1)} &= \tilde{\boldsymbol{\theta}}_1^{(l)}, \\ \tilde{\boldsymbol{\theta}}_2^{(l-1)} &= (\tilde{\boldsymbol{\theta}}_2^{(l)} - \boldsymbol{\kappa}_{\gamma,2}^{(l)}(\tilde{\boldsymbol{\theta}}_1^{(l)}, \mathbf{Z})) \odot \exp(-\boldsymbol{\kappa}_{\gamma,1}^{(l)}(\tilde{\boldsymbol{\theta}}_1^{(l)}, \mathbf{Z})). \end{aligned}$$

The Jacobian of the transformation is triangular and therefore its determinant is straightforward to compute. Note that the variables need to be permuted between coupling blocks in order for all input components to (eventually) be transformed. Affine coupling flows find their roots in the work of Dinh et al. (2016), and one of its variants is implemented in the popular package **BayesFlow** that does amortised inference by forward KL minimisation (Section 2.2.1).

Masked autoregressive flows: An autoregressive flow is a composition of several layers of the form,

$$\begin{aligned}\tilde{\theta}_1^{(l)} &= h(\tilde{\theta}_1^{(l-1)}; \kappa_{\gamma,1}^{(l)}(\mathbf{Z})), \\ \tilde{\theta}_t^{(l)} &= h(\tilde{\theta}_t^{(l-1)}; \kappa_{\gamma,t}^{(l)}(\tilde{\theta}_{1:(t-1)}^{(l-1)}, \mathbf{Z})), \quad t = 2, \dots, d,\end{aligned}$$

where $\kappa^{(l)}(\cdot) \equiv \{\kappa_{\gamma,1}^{(l)}(\cdot), \kappa_{\gamma,2}^{(l)}(\cdot), \dots, \kappa_{\gamma,d}^{(l)}(\cdot)\}$ are (generic, not necessarily invertible) neural networks that output the parameters of an invertible transformation $h(\cdot)$. The resulting Jacobian is triangular and hence the determinant required for the transformation is easy to compute. There are several connections between the vanilla autoregressive flow presented here and the affine coupling flows; these are discussed in detail by Papamakarios et al. (2021).

A computationally attractive implementation of the autoregressive flow is the masked autoregressive flow, where $\kappa^{(l)}(\cdot)$ is a single neural network that outputs the required parameters for all $t = 1, \dots, d$. This is achieved by ensuring there are no connections between $\tilde{\theta}_{t:d}^{(l-1)}$ and the flow parameters corresponding to the t th transformation, through appropriate use of masking 0-1 matrices. Masked autoregressive flows were developed by Papamakarios et al. (2017), and have been extensively used for both amortised posterior inference and amortised likelihood estimation (e.g., Greenberg et al., 2019, Papamakarios et al., 2019, Brehmer et al., 2020, Wqvist et al., 2021, Glöckler et al., 2022).

S3 COMPARATIVE STUDY WITH INVERTED MAX-STABLE PROCESSES

Likelihood-based inference with popular models for spatial extremes is notoriously challenging (Huser and Wadsworth, 2022). Among such models, max-stable processes (Padoan et al., 2010, Davison et al., 2012) have perhaps been the most widely used, despite the fact that their likelihood function is computationally intractable in high dimensions (Castruccio et al., 2016). Consider a spatial domain $\mathcal{D} \subset \mathbb{R}^2$. On unit Fréchet margins (i.e., with cumulative distribution function $\exp(-1/z)$, $z > 0$), such models can be represented as $Z(\mathbf{s}) = \sup_{i \geq 1} \xi_i W_i(\mathbf{s})$, $\mathbf{s} \in \mathcal{D}$, where $\{\xi_i\}_{i=1}^\infty$ are points from a Poisson point process with intensity $\xi^{-2} d\xi$ on $(0, \infty)$ and $W_i(\cdot)$ are independent copies of a nonnegative process $W(\cdot)$ with unit mean (also independent of ξ_i). From this representation, one can show that the joint distribution at sites $\{\mathbf{s}_1, \dots, \mathbf{s}_n\} \subset \mathcal{D}$ has the form $\Pr(Z(\mathbf{s}_1) \leq z_1, \dots, Z(\mathbf{s}_n) \leq z_n) = \exp(-V(z_1, \dots, z_n))$ with $V(z_1, \dots, z_n) = \mathbb{E}(\max_{j=1, \dots, n} \{W(\mathbf{s}_j)/z_j\})$ the so-called exponent function. Various choices for the process $W(\cdot)$ lead to different types of max-stable processes: Schlather (2002), for instance, proposed setting $W(\mathbf{s}) = \sqrt{2\pi} \max\{0, \varepsilon(\mathbf{s})\}$, $\mathbf{s} \in \mathcal{D}$, for a standard Gaussian process $\varepsilon(\cdot)$ with some chosen correlation function, which leads to an analytical form for the exponent function, but there are several other possibilities; see Davison et al. (2012), Davison and Huser (2015) and Davison et al. (2019) for exhaustive reviews.

While max-stable processes are often used in applications, recently Huser et al. (2024) argued against the systematic use of max-stable processes in environmental applications due to the rigidity of their dependence structure (especially in the tail) and their inability to capture weakening dependence and asymptotic independence. Huser et al. (2024) instead argued in favour of other more flexible, computationally efficient, and

pragmatic alternative modelling solutions. One possible solution to flexibly capture the sub-asymptotic tail behavior of spatial processes displaying asymptotic independence is to consider so-called inverted max-stable processes (Wadsworth and Tawn, 2012). These models effectively swap the roles of the lower and upper tails in max-stable processes: If $Z(\cdot)$ is a max-stable process with unit Fréchet margins, then $\tilde{Z}(\cdot) = 1/Z(\cdot)$ is the corresponding inverted max-stable process, yet now expressed on standard exponential margins.

Unlike max-stable processes, inverted max-stable processes can flexibly capture asymptotic independence in the upper tail, but they remain challenging to fit since they are constructed from max-stable processes themselves, which admit an intractable likelihood function; they are therefore ideal candidates to illustrate the merits of likelihood-free neural approaches summarised in this review paper (as also advocated by Huser et al. (2024)). From their definition, the joint bivariate survival function of the bivariate vector $(\tilde{Z}(\mathbf{s}_j), \tilde{Z}(\mathbf{s}_k))'$, $\mathbf{s}_j, \mathbf{s}_k \in \mathcal{D}$, can be straightforwardly obtained as $\Pr(\tilde{Z}(\mathbf{s}_j) > z_j, \tilde{Z}(\mathbf{s}_k) > z_k) = \exp(-V(1/z_j, 1/z_k))$, where $V(\cdot)$ denotes the bivariate restriction of the exponent function of the underlying max-stable process $Z(\cdot)$ to the pair of sites $\{\mathbf{s}_j, \mathbf{s}_k\}$. This formula can be used to derive the bivariate density function $p(Z_j, Z_k | \boldsymbol{\theta})$ for any pair of variables $(\tilde{Z}(\mathbf{s}_j), \tilde{Z}(\mathbf{s}_k))'$, $j, k = 1, \dots, n$, where $\boldsymbol{\theta}$ denotes the model parameters. Classical likelihood-based inference then proceeds by maximising (with respect to $\boldsymbol{\theta}$) a pairwise log-likelihood constructed as $\sum_{j < k} w_{jk} \log(p(Z_j, Z_k | \boldsymbol{\theta}))$, where $w_{jk} \geq 0$ are suitable weights chosen to improve both statistical and computational efficiency. Often, one sets $w_{jk} = 1$ if $\|\mathbf{s}_j - \mathbf{s}_k\| < \delta_{\max}$ for some cut-off distance $\delta_{\max} > 0$, and $w_{jk} = 0$ otherwise, thus discarding distant pairs of sites. Here, in our simulation study, we consider the Schlather inverted max-stable model parameterised in terms of the length scale $\lambda > 0$ and smoothness parameter $\nu > 0$ of its underlying Matérn correlation function; that is, $\boldsymbol{\theta} = (\lambda, \nu)'$. As for the prior distributions, we let $\lambda \sim \text{Unif}(0, 0.6)$ and $\nu \sim \text{Unif}(0.5, 3)$. We simulate datasets on a 16×16 spatial gridding of $[0, 1]^2$ (i.e., at 256 locations) with a single replicate in each dataset. To make inference on $\boldsymbol{\theta}$, we then consider maximum composite likelihood, MCL, with $\delta_{\max} = 0.2$ (i.e., keeping only about 10% of pairs in the pairwise likelihood for major improvements in speed and accuracy), as well as the fKL minimisation approach with a normalising flow implementation of the approximate posterior distribution using **BayesFlow**.

Figure S1a shows the results when applying the methods to three (test) spatial fields, $\mathbf{Z}_1, \mathbf{Z}_2, \mathbf{Z}_3$. The bivariate posterior distributions are relatively diffuse compared to the single-parameter GP example of Section 5.2, but corroborate the true value in every instance. The MCL estimator also yields reasonable estimates generally, although for the second test case it estimates a value for ν that is on the boundary of the parameter space. Figure S1b plots the posterior means and the MCL estimates for the two parameters. It is well known that estimating these parameters from a single replicate is difficult, but we can see a clear correlation between the posterior mean estimates and the true value; this is less true for the MCL estimates, which often lie on the boundary of the parameter space. The root-mean-squared error between the estimated and the true value by the amortised estimator was about half that of the pairwise likelihood estimator, for both λ (0.119 vs. 0.212) and ν (0.506 vs. 0.842). In this case, these improved estimates (and accompanying full posterior distributions) were also obtained with a 50-fold speedup.

S4 ADDITIONAL TOPICS

S4.1 Likelihood-free reverse-KL approaches

In this section we outline two additional methods related to reverse-KL minimisation that are likelihood-free by design: conditional generative adversarial networks, and variational auto-encoders.

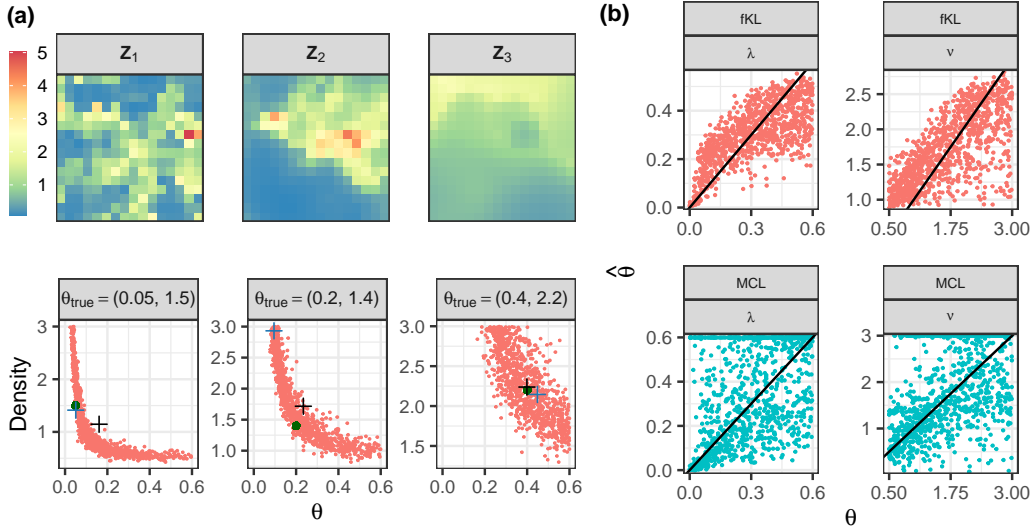


Figure S1: Results from the illustration of Section S3. Panel (a) shows the results from the two methods on three test datasets. The top sub-panels show the data, while the bottom sub-panels show samples from the inferred posterior distribution (red dots), the posterior mean (black cross), the pairwise likelihood estimate (blue cross), as well as the true parameter value (green dot). Panel (b) plots parameter point estimates against the true parameter value in the test dataset for all methods.

Conditional generative adversarial networks: Generative adversarial networks (GANs Goodfellow et al., 2014) are widely known for their ability to generate realistic high-dimensional data, such as text, images, and video. They comprise two neural networks, a generator, which generates outputs of interest from some latent noise input that is arbitrarily distributed (e.g., normally distributed), and a discriminator whose role is to discriminate outputs from the generator and data from the underlying data generating process. GANs are trained using a minimax loss function, where the discriminator parameters are optimised to be able to correctly classify a datum as coming from the data generating process or from the generator, and the generator network is optimised to generate data that resemble samples from the true generating process as much as possible.

When used in the context of simulation-based inference, the generator in a GAN generates samples from a posterior distribution through the use of data \mathbf{Z} as a conditioning input (Ramesh et al., 2022). Specifically, the inference network in the conditional GAN is the generator $\mathbf{g}_\gamma(\mathbf{Z}, \mathbf{W})$, that takes in the conditioning data \mathbf{Z} and latent quantities \mathbf{W} as input (that are independent from \mathbf{Z} , arbitrarily distributed, and that can be easily sampled), and returns parameters θ . For any fixed \mathbf{Z} it can be shown that, once trained, $\theta \sim q_\gamma(\theta | \mathbf{Z})$, where $q_\gamma(\cdot | \mathbf{Z})$ minimises the reverse KL divergence between the true posterior distribution and the approximate posterior distribution (Ramesh et al., 2022, Appendix A.2). Specifically, generator parameters γ are the solution to

$$\gamma^* = \arg \min_{\gamma} \mathbb{E}_{\mathbf{Z}}[\text{KL}(q_\gamma(\theta | \mathbf{Z}) \| p(\theta | \mathbf{Z}))].$$

Training of the GANs is sample-based and, although it yields a variational approximate posterior distribution, it does not require knowledge of the likelihood.

Variational auto-encoders: The variational auto-encoder (VAE, Kingma and Welling, 2013) is a type of amortised variational inference where the likelihood is assumed to have a simple form with distributional parameters expressed through a decoder (the in-

ference network is analogously referred to as an encoder). Consider the case where the assumed likelihood function $q(\mathbf{Z}; \boldsymbol{\mu}_\eta(\boldsymbol{\theta}), \boldsymbol{\Sigma}_\eta(\boldsymbol{\theta}))$ is a Gaussian function, with mean parameter $\boldsymbol{\mu}_\eta(\boldsymbol{\theta})$ and covariance matrix $\boldsymbol{\Sigma}_\eta(\boldsymbol{\theta})$ outputs of a neural network that has parameters $\boldsymbol{\eta}$. Then, given a sample $\boldsymbol{\theta}^{(j)}$ from the variational distribution over $\boldsymbol{\theta}$, one produces a variational posterior-predictive sample from $q(\mathbf{Z}; \boldsymbol{\mu}_\eta(\boldsymbol{\theta}^{(j)}), \boldsymbol{\Sigma}_\eta(\boldsymbol{\theta}^{(j)}))$. In a VAE, the likelihood network and the inference network are trained jointly, and the optimisation problem of Equation 8 in the main text becomes

$$(\boldsymbol{\gamma}^{*'}, \boldsymbol{\eta}^{*'})' \approx \arg \min_{(\boldsymbol{\gamma}', \boldsymbol{\eta}')} \frac{1}{KJ} \sum_{k=1}^K \sum_{j=1}^J \left(\log q(\boldsymbol{\theta}^{(j)}; \boldsymbol{\kappa}_\gamma(\mathbf{Z}^{(k)})) - \log q(\mathbf{Z}^{(k)}; \boldsymbol{\mu}_\eta(\boldsymbol{\theta}^{(j)}), \boldsymbol{\Sigma}_\eta(\boldsymbol{\theta}^{(j)})) p(\boldsymbol{\theta}^{(j)}) \right),$$

where $\mathbf{Z}^{(k)} \sim p(\mathbf{Z})$ and $\boldsymbol{\theta}^{(j)} \sim q(\boldsymbol{\theta}; \boldsymbol{\kappa}_\gamma(\mathbf{Z}^{(k)}))$. Although the VAE has been used in an amortised inference framework by Goh et al. (2019), it is most often used in situations where one only has realisations of \mathbf{Z} and no knowledge of the underlying data generating process. In these cases $\boldsymbol{\theta}$ plays the role of a latent variable encoding a lower dimensional representation of the data (e.g., Doersch, 2016, Kingma et al., 2019, Cartwright et al., 2023).

S4.2 Mitigating the amortisation gap with sequential methods and semi-amortisation

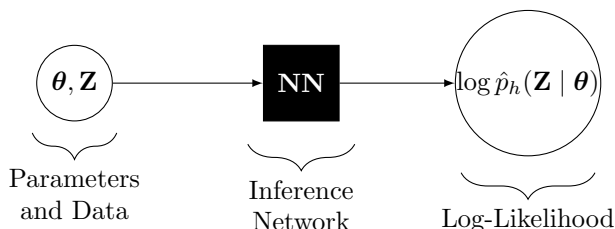
The amortisation gap, which all methods discussed in this review may suffer from, can be reduced by adopting a sequential strategy, where re-training of neural networks is done in regions of interest. Sequential methods are not-amortised *per se* (since the region of interest varies with the dataset) but can build and extend on the methods discussed in this review. In sequential methods for posterior inference, parameters are sampled from a proposal distribution that generally differs from the prior distribution, and require importance-weight adjustment to the loss function (e.g., Lueckmann et al., 2017) or to the approximate posterior distribution post-training (e.g., Papamakarios and Murray, 2016); see also Greenberg et al. (2019) and Goncalves et al. (2020). Sequential methods are also used to refine the neural likelihood function in region of high parameter interest (Papamakarios et al., 2019) and can also be used to specifically lower the amortisation gap globally. For example, using a Bayesian neural network for the likelihood network, Lueckmann et al. (2019) propose a training strategy that sequentially minimises the uncertainty on the likelihood approximation. Another approach to deal with the amortisation gap when making posterior inference is semi-amortisation. Here, a global amortised inference network is used as an initialisation for standard (non-amortised) inference. This is attractive in the reverse-KL scenario, where (non-amortised) variational inference is straightforward; see, for example, Hjelm et al. (2016) and Kim et al. (2018).

S4.3 Neural “doubly-approximate” likelihood approaches

Some amortised neural likelihood methods proposed in the recent literature are doubly-approximate in that two levels of approximation are involved: (i) neural networks are used to approximate a certain target and (ii) this target is itself an approximate likelihood (rather than the true likelihood). Here we outline two such methods.

Kernel density target: Fengler et al. (2021) propose using so-called likelihood approximation networks (LANs), which are divided into two distinct methods: the “pointwise approach” and the “histogram approach”. The pointwise approach learns the log-likelihood function $\log p(\mathbf{Z} | \boldsymbol{\theta})$ by representing it through a neural network taking a parameter vector and a data point as (multi-dimensional) input and returning the log-likelihood value as (uni-dimensional) output. To train the network, empirical likelihood values are first

obtained for each specific parameter value θ using a kernel density estimator of the form $\hat{p}_h(\mathbf{Z} | \theta) = \frac{1}{N} \sum_{i=1}^N f_h(\|\mathbf{Z} - \mathbf{Z}^{(i)}\|)$ for some kernel function $f_h(\cdot)$ with bandwidth $h > 0$ (e.g., a centred Gaussian density with standard deviation h), where $\mathbf{Z}^{(i)} \sim p(\mathbf{Z} | \theta)$; these empirical values serve as surrogates for the (unknown) true likelihood values. Then, the neural approximate likelihood can be trained by minimising an appropriate loss function contrasting neural log-likelihood and empirical log-likelihood values $\hat{p}_h(\mathbf{Z} | \theta)$. Under the squared (respectively absolute) error loss, the pointwise LAN approach boils down to performing a deep non-linear mean (respectively median) regression with empirical likelihood values playing the role of response variables and the combination of parameters and data points playing the role of covariates. Fengler et al. (2021) instead consider the Huber loss (Huber, 1964) due to its robustness, but many other options are also possible. When the dataset contains multiple independent replicates, the overall approximate log-likelihood function is then obtained by evaluating, one by one, all single-observation log-likelihoods using the trained network and then summing them up. This pointwise approach admits the following graphical representation:



The histogram approach, in contrast, seeks to provide a method that can directly evaluate the overall (log-)likelihood function for an arbitrary number of data points via a single forward pass through the network. This is achieved by modifying the response variable to be binned empirical likelihood values (using the same kernel density estimator technique), and regressing it against parameter values alone using an appropriate neural network architecture and loss function (e.g., the KL divergence between binned densities).

While both of these approaches were shown by Fengler et al. (2021) to provide good results in a relatively low-dimensional cognitive neuroscience data example (using a multi-layer perceptron for the neural network in the pointwise approach, and a different network with convolutional layers in the histogram approach), LANs have several drawbacks (Boelts et al., 2022). First, the response variables in the described deep regression problem are not true likelihood values, but based on empirical kernel-density likelihood values, which are approximate and have their own limitations, especially in their ability to capture the tails. Second, since the accuracy of the method depends on the accuracy of the kernel-density-estimated empirical likelihoods, a large number of training samples are typically required to ensure that the empirical kernel-density likelihood values accurately approximate their true likelihood counterparts. For example, Fengler et al. (2021) used 1.5 million parameter values and 100,000 data points per parameter, i.e., more than 100 billion training data points in total. This requirement is likely to make the training phase computationally prohibitive in many applications.

Vecchia likelihood target: Another recently-introduced doubly-approximate approach consists in exploiting the Vecchia likelihood approximation (Vecchia, 1988, Stein et al., 2004, Katzfuss and Guinness, 2021), which represents the likelihood function as a product of univariate conditional densities, yet with a reduced number of conditioning variables (hence the approximation); then, amortised neural conditional density estimators can be trained for each density of the form $p(Z_j | \mathbf{Z}_{\text{Nei}(j)})$, where Z_j denotes the j th element of the vector \mathbf{Z} and $\mathbf{Z}_{\text{Nei}(j)}$ contains some neighboring variables within its “history” according a predefined sequence; after training, these approximate conditional densities are put back together to get an amortised likelihood approximation of $p(\mathbf{Z} | \theta)$ (Majumder and Reich, 2023, Majumder et al., 2023).

S4.4 Reliable inference with simulation-based calibration

Due to the amortisation gap, credible and confidence intervals/regions obtained from the neural inference approaches described in this review paper can be over-confident with poor frequentist properties, thus biasing inferences and subsequent decision-making (Hermans et al., 2022). Various methods have been proposed to mitigate this issue. One possible approach called WALDO (Masserano et al., 2023), which builds on the work of Dalmaso et al. (2021), aims at constructing confidence regions with finite-sample conditional validity through Neyman inversion, mimicking the well-known Wald test. This method considers a Wald-like test statistic of the form $W(\mathbf{Z}; \boldsymbol{\theta}_0) = (\mathbb{E}(\boldsymbol{\theta} | \mathbf{Z}) - \boldsymbol{\theta}_0)' \mathbb{V}(\boldsymbol{\theta} | \mathbf{Z})^{-1} (\mathbb{E}(\boldsymbol{\theta} | \mathbf{Z}) - \boldsymbol{\theta}_0)$, where $\mathbb{E}(\boldsymbol{\theta} | \mathbf{Z})$ and $\mathbb{V}(\boldsymbol{\theta} | \mathbf{Z})$ are, respectively, the conditional mean and the conditional covariance matrix of the model parameter $\boldsymbol{\theta}$ given data \mathbf{Z} , which can be estimated from the neural amortised method under consideration, either analytically or by sampling. The critical values of this test statistic can be learned through a separate deep quantile regression model. Confidence regions resulting from WALDO will, by construction, satisfy conditional coverage regardless of the sample size and the true value of $\boldsymbol{\theta}$, provided the quantile regression fits well. WALDO is applicable to both frequentist and Bayesian neural methods, and when used in a Bayesian context it still yields valid confidence regions at the desired level irrespective of the prior distribution. Other empirical simulation-based approaches for validating an amortised estimator include Talts et al. (2018), Zhao et al. (2021), and Hermans et al. (2022).

S4.5 Inference under model misspecification

Model misspecification (i.e., distribution shift) is a problem that is ubiquitous in statistical modelling and inference. With likelihood-based inference, the effect of model misspecification is well understood from classical asymptotic theory (see, e.g., Davison, 2003, page 147–148); the same is not true with neural inference approaches. This open research question is especially relevant as neural networks are known to extrapolate poorly in general. While there are simple numerical experiments that suggest that these methods have some built-in robustness against misspecification (e.g., supplementary material of Sainsbury-Dale et al. (2024)), others suggest otherwise (e.g., Bon et al., 2023). Efforts to tackle model misspecification include the use of distribution-mismatch-detection of the learned summary statistics with respect to a reference distribution (Schmitt et al., 2024, Radev et al., 2023). Kelly et al. (2023) instead modify summary statistics using auxiliary parameters to build a robust version of the sequential neural likelihood approach of Papamakarios et al. (2019).

S4.6 Inference with complex data types

Certain neural inference approaches, for example those relying on normalising flows (e.g., Papamakarios et al., 2019), cannot be easily applied when the data are partially discrete and partially continuous. To address this issue, Boelts et al. (2022) propose the so-called mixed neural likelihood estimation (MNLE) method, which draws on Papamakarios et al. (2019), but splits the inference problem into two simpler sub-problems: one that trains a conditional density network (e.g., using normalising flows) to model continuous data given model parameters and the discrete data (used as an input to the neural network), and another one that only models discrete data given model parameters. These two networks are then combined together to get the full model.

Similarly, neural inference with missing or censored data is not trivial because most neural network architectures are not made to handle such inputs. Graph neural networks (GNNs, see Section S1), designed to take different graphical structures as input, is a possible architecture that can naturally deal with data missingness. Alternatively, Wang et al. (2023) propose setting the missing data to an arbitrary constant, and augmenting the input with an additional binary dataset of the same size indicating which data are

missing. A neural network can then be trained on this modified and augmented input; the rationale is that the network will learn to treat the missing data differently from the non-missing observations. The drawback of this approach, however, is that the trained neural network is sensitive to the data missingness model used during the training phase, which can bias inference if misspecified. In the context of censored data, Richards et al. (2023) use an approach similar to Wang et al. (2023) for making inference in a spatial extremes application where non-extreme values are censored (but not missing). There is, however, an important difference with respect to Wang et al. (2023), namely that the censoring mechanism is fully determined by the data and the censoring threshold, and thus never misspecified.

S4.7 Neural model selection

Amortised model comparison and selection can naturally proceed by training a neural discriminator taking data (or summary statistics thereof) as input and returning a model probability as output, thus mimicking the architecture of neural Bayes estimators (recall Section 2.1) but with a modified final output layer. Such an approach is closely related to hypothesis testing, and it has been used, for example, by Ahmed et al. (2022) to select among models with different tail structures. Alternatively, another possible model selection approach is to proceed by computing Bayes factors (i.e., ratios of marginal likelihoods), which can be directly obtained when the posterior density and the likelihood function are both available as, e.g., in the JANA framework (Radev et al., 2023). Traditional likelihood-ratio tests to compare nested models can also be performed using (approximate) neural full likelihood and likelihood-ratio methods, even when they are only available up to a normalising constant.

S4.8 Applications employing amortised neural inference

There has been a marked increase in the use of neural networks for making parameter inference in recent years. The following list of applications is non-exhaustive, but serves to show the wide scope of applications neural amortised inference can be useful for.

David et al. (2021) use a neural point estimator to map spectra collected by NASA’s OCO-2 satellite to column-averaged carbon dioxide, using estimates available from classical inversion techniques as target output parameters during training. Amortisation in this application is highly desirable because of the high throughput of remote sensing instruments. Richards et al. (2023) apply neural Bayes estimators to fit hundreds of thousands of models for spatial extremes (which have computationally-intractable likelihoods) to censored particulate matter data in Saudi Arabia in a study on air pollution.

In the context of approximate Bayes, Speiser et al. (2017) use amortised variational methods to infer neural activity from fluorescence traces; their inference network uses a combination of recurrent neural networks and convolutional neural networks (see Section S1), and as approximate posterior distribution they use a Bernoulli distribution for the spike probabilities. Chan et al. (2018) use forward KL minimisation with a Gaussian approximate posterior distribution of the intensity of recombination hotspots from genome data. Siahkoochi et al. (2023) solve a geophysical inverse problem to infer the state of Earth’s subsurface from surface measurements. They use forward KL minimisation with hierarchical normalising flows (Kruse et al., 2021) to approximate the posterior distribution of the state. Similarly, von Krause et al. (2022) use forward KL minimisation to fit diffusion models to response time data from over one million participants and study how mental speed varies with age.

There have also been many applications of neural likelihood and likelihood-to-evidence-ratio approximators. For instance, Lueckmann et al. (2019) construct neural synthetic likelihoods to make inference on the evolution of membrane potential in (brain) neurons through the Hodgkin–Huxley biophysical model. Fengler et al. (2021) apply the

LAN method to make inference with stochastic differential equation models commonly used in cognitive neurosciences. Using amortised neural likelihood ratio approximators, Baldi et al. (2016) make inference with high-energy particle physics models; Delaunoy et al. (2020) make inference on gravitational waves; and Cole et al. (2022) infer cosmological parameters based on measurements of the cosmic microwave background.

S5 ADDITIONAL TABLES AND FIGURES

Table S1: Outline of software and architectures used in the illustration of Section 5.2

Acronym	Method	Implementation
MCMC	Metropolis–Hastings.	Base R; (R Core Team, 2023).
NBE	Neural Bayes estimation.	NeuralEstimators ; Sainsbury-Dale et al. (2024). Summary network: 2-layer CNN followed by two fully-connected layers. Inference network: two fully-connected layers to output posterior mean and 0.05/0.95 posterior quantiles.
fKL	Forward KL minimisation with normalising flow approximate posterior.	BayesFlow ; (Radev et al., 2022). Summary network: 2-layer CNN followed by two fully-connected layers. Inference network: Invertible neural network with four coupling layers to sample from the posterior distribution.
rKL1	Reverse KL minimisation with logit-Gaussian approximate posterior.	TensorFlow ; (Abadi et al., 2016). Inference network: 2-layer CNN followed by two fully-connected layers to output variational mean and variance.
rKL2	Reverse KL minimisation with logit-Gaussian approximate posterior and with Gaussian synthetic likelihood constructed using one “expert” summary statistic.	TensorFlow ; (Abadi et al., 2016). Binding function network: 2 fully-connected layers that output mean and variance of summary statistic. Inference network: 2-layer CNN followed by two fully-connected layers to output variational mean and variance.
rKL3	Reverse KL minimisation with logit-Gaussian approximate posterior and with Gaussian synthetic likelihood constructed using one summary statistic found by maximising mutual information.	TensorFlow ; (Abadi et al., 2016). Summary network: 2-layer CNN followed by two fully-connected layers to output summary statistic from input data. Binding function network: 2 fully-connected layers that output mean and variance of summary statistic. Inference network: 2-layer CNN followed by two fully-connected layers to output variational mean and variance.
NRE	Neural ratio estimation with sampling from the posterior distribution on a fine gridding of the parameter space.	sbi ; (Tejero-Cantero et al., 2020) Summary network: 2-layer CNN followed by two fully-connected layers. Inference network: three fully-connected layers to output likelihood-to-evidence ratio.

Table S2: Evaluation of the methods in Section 5.2 on test data using the root-median-squared prediction error (RMSPE), the median-absolute prediction error (MAPE), the median 90% interval score (MIS90), the 90% coverage (COV90), and the median continuous-ranked probability score (MCRPS). All scores excluding COV90 (which should be close to 0.9) are negatively oriented (lower is better).

Method	RMSPE	MAPE	MIS90	COV90	MCRPS
MCMC	0.015	0.015	0.095	0.896	0.014
NBE	0.016	0.016	0.106	0.877	
fKL	0.015	0.015	0.099	0.906	0.015
rKL1	0.016	0.016	0.098	0.777	0.014
rKL2	0.019	0.019	0.114	0.871	0.016
rKL3	0.018	0.018	0.091	0.816	0.014
NRE	0.017	0.017	0.110	0.913	0.016

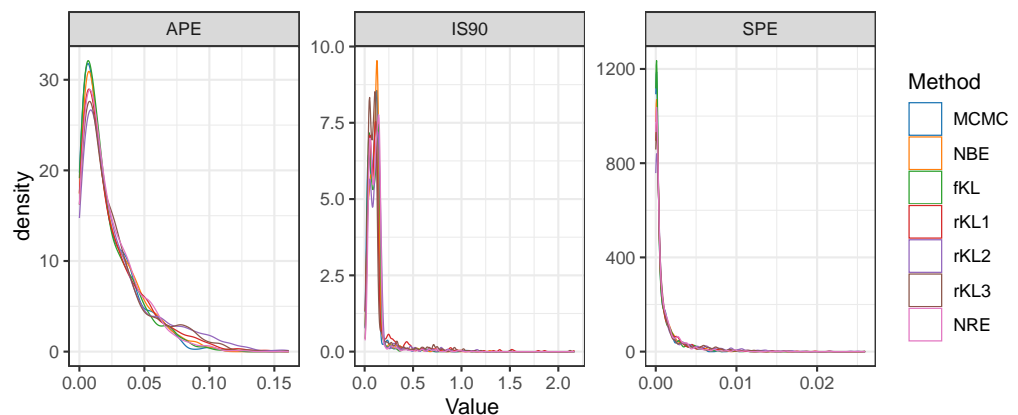


Figure S2: Empirical distribution of the absolute prediction errors (left), 90% interval scores (centre), and squared prediction errors (right) for the different methods considered in the illustration of Section 5.2.

References

- Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2016). TensorFlow: Large-scale machine learning on heterogeneous systems. arXiv:1603.04467 [cs.DC].
- Ahmed, M., Maume-Deschamps, V., and Ribereau, P. (2022). Recognizing a spatial extreme dependence structure: A deep learning approach. *Environmetrics*, 33:e2714.
- Baldi, P., Cranmer, K., Faucett, T., Sadowski, P., and Whiteson, D. (2016). Parameterized neural networks for high-energy physics. *The European Physical Journal C*, 76:235.
- Bishop, C. (1995). *Neural Networks for Pattern Recognition*. Clarendon Press, Oxford, UK.

- Boelts, J., Lueckmann, J.-M., Gao, R., and Macke, J. H. (2022). Flexible and efficient simulation-based inference for models of decision-making. *eLife*, 11:e77220.
- Bon, J. J., Bretherton, A., Buchhorn, K., Cramb, S., Drovandi, C., Hassan, C., Jenner, A. L., Mayfield, H. J., McGree, J. M., Mengersen, K., et al. (2023). Being Bayesian in the 2020s: opportunities and challenges in the practice of modern applied Bayesian statistics. *Philosophical Transactions of the Royal Society A*, 381:20220156.
- Brehmer, J., Louppe, G., Pavez, J., and Cranmer, K. (2020). Mining gold from implicit models to improve likelihood-free inference. *Proceedings of the National Academy of Sciences*, 117:5242–5249.
- Cartwright, L., Zammit-Mangion, A., and Deutscher, N. M. (2023). Emulation of greenhouse-gas sensitivities using variational autoencoders. *Environmetrics*, 34:e2754.
- Castruccio, S., Huser, R., and Genton, M. G. (2016). High-order composite likelihood inference for max-stable distributions and processes. *Journal of Computational and Graphical Statistics*, 25:1212–1229.
- Chan, J., Perrone, V., Spence, J., Jenkins, P., Mathieson, S., and Song, Y. (2018). A likelihood-free inference framework for population genetic data using exchangeable neural networks. In *Proceedings of the 32nd Conference on Neural Information Processing Systems (NeurIPS 2018)*, pages 8594–8605, Red Hook, NY. Curran.
- Cole, A., Miller, B. K., Witte, S. J., Cai, M. X., Grootes, M. W., Nattino, F., and Weniger, C. (2022). Fast and credible likelihood-free cosmology with truncated marginal neural ratio estimation. *Journal of Cosmology and Astroparticle Physics*, 2022:004.
- Dalmaso, N., Masserano, L., Zhao, D., Izbicki, R., and Lee, A. B. (2021). Likelihood-free frequentist inference: Bridging classical statistics and machine learning for reliable simulator-based inference. arXiv:2107.03920 [stat.ML].
- David, L., Bréon, F.-M., and Chevallier, F. (2021). XCO₂ estimates from the OCO-2 measurements using a neural network approach. *Atmospheric Measurement Techniques*, 14:117–132.
- Davison, A. C. (2003). *Statistical Models*. Cambridge University Press, Cambridge, UK.
- Davison, A. C. and Huser, R. (2015). Statistics of extremes. *Annual Review of Statistics and its Application*, 2:203–235.
- Davison, A. C., Huser, R., and Thibaud, E. (2019). Spatial extremes. In Gelfand, A. E., Fuentes, M., Hoeting, J. A., and Smith, R. L., editors, *Handbook of Environmental and Ecological Statistics*, pages 711–744. Chapman & Hall/CRC Press, Boca Raton, FL.
- Davison, A. C., Padoan, S. A., and Ribatet, M. (2012). Statistical modeling of spatial extremes. *Statistical Science*, 27:161–186.
- de Bézenac, E., Pajot, A., and Gallinari, P. (2018). Deep learning for physical processes: Incorporating prior scientific knowledge. Vancouver, Canada: OpenReview. <https://openreview.net/forum?id=By4HsfWAZ>.
- Delaunoy, A., Wehenkel, A., Hinderer, T., Nissanke, S., Weniger, C., Williamson, A. R., and Louppe, G. (2020). Lightning-fast gravitational wave parameter inference through neural amortization. arXiv:2010.12931 [astro-ph.IM].
- Dinev, T. and Gutmann, M. U. (2018). Dynamic likelihood-free inference via ratio estimation (DIRE). arXiv:1810.09899 [stat.ML].

- Dinh, L., Sohl-Dickstein, J., and Bengio, S. (2016). Density estimation using Real NVP. In *Proceedings of the 4th International Conference on Learning Representations (ICLR 2016)*. San Juan, Puerto Rico: OpenReview. <https://openreview.net/forum?id=HkpbnH91x>.
- Doersch, C. (2016). Tutorial on variational autoencoders. arXiv:1606.05908 [stat.ML].
- Fengler, A., Govindarajan, L., Chen, T., and Frank, M. J. (2021). Likelihood approximation networks (LANs) for fast inference of simulation models in cognitive neuroscience. *eLife*, 10:e65074.
- Gerber, F. and Nychka, D. (2021). Fast covariance parameter estimation of spatial Gaussian process models using neural networks. *Stat*, 10:e382.
- Glöckler, M., Deistler, M., and Macke, J. H. (2022). Variational methods for simulation-based inference. In *Proceedings of the 10th International Conference on Learning Representations (ICLR 2022)*. Virtual: OpenReview. <https://openreview.net/forum?id=kZ0UYdhqkNY>.
- Goh, H., Sherifdeen, S., Wittmer, J., and Bui-Thanh, T. (2019). Solving Bayesian inverse problems via variational autoencoders. In *Proceedings of the 2nd Mathematical and Scientific Machine Learning Conference (MSML 2022)*, volume 145, pages 386–425. PMLR.
- Goncalves, P. J., Lueckmann, J.-M., Deistler, M., Nonnenmacher, M., Ocal, K., Bassetto, G., Chintaluri, C., Podlaski, W. F., Haddad, S. A., Vogels, T. P., et al. (2020). Training deep neural density estimators to identify mechanistic models of neural dynamics. *eLife*, 9:e56261.
- Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Proceedings of the 28th Conference on Neural Information Processing Systems (NeurIPS 2014)*, pages 2672–2680, Red Hook, NY. Curran.
- Greenberg, D., Nonnenmacher, M., and Macke, J. (2019). Automatic posterior transformation for likelihood-free inference. In *Proceedings of the 36th International Conference on Machine Learning (ICML 2019)*, volume 97, pages 2404–2414. PMLR.
- Hermans, J., Delaunoy, A., Rozet, F., Wehenkel, A., Begy, V., and Louppe, G. (2022). A crisis in simulation-based inference? Beware, your posterior approximations can be unfaithful. *Transactions on Machine Learning Research*. OpenReview. <https://openreview.net/pdf?id=LHAbHkt6Aq>.
- Hjelm, D., Salakhutdinov, R. R., Cho, K., Jovic, N., Calhoun, V., and Chung, J. (2016). Iterative refinement of the approximate posterior for directed belief networks. In *Proceedings of the 30th Conference on Neural Information Processing Systems (NeurIPS 2016)*, pages 4698–4706, Red Hook, NY. Curran.
- Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9:1735–1780.
- Huber, P. J. (1964). Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, 53:73–101.
- Huser, R., Opitz, T., and Wadsworth, J. L. (2024). Modeling of spatial extremes in environmental data science: Time to move away from max-stable processes. arXiv:2401.17430 [stat.ME].
- Huser, R. and Wadsworth, J. L. (2022). Advances in statistical modeling of spatial extremes. *Wiley Interdisciplinary Reviews: Computational Statistics*, 14:e1537.

- Katzfuss, M. and Guinness, J. (2021). A general framework for Vecchia approximations of Gaussian processes. *Statistical Science*, 36:124–141.
- Kelly, R. P., Nott, D. J., Frazier, D. T., Warne, D. J., and Drovandi, C. (2023). Misspecification-robust sequential neural likelihood. arXiv:2301.13368 [stat.ME].
- Kim, Y., Wiseman, S., Miller, A., Sontag, D., and Rush, A. (2018). Semi-amortized variational autoencoders. In *Proceedings of the 35th International Conference on Machine Learning (ICML 2018)*, volume 80, pages 2678–2687. PMLR.
- Kingma, D. P. and Welling, M. (2013). Auto-encoding variational Bayes. arXiv:1312.6114 [stat.ML].
- Kingma, D. P., Welling, M., et al. (2019). An introduction to variational autoencoders. *Foundations and Trends in Machine Learning*, 12:307–392.
- Kobyzev, I., Prince, S. J., and Brubaker, M. A. (2020). Normalizing flows: An introduction and review of current methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43:3964–3979.
- Kruse, J., Detommaso, G., Köthe, U., and Scheichl, R. (2021). HINT: Hierarchical invertible neural transport for density estimation and Bayesian inference. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI-21)*, volume 35, pages 8191–8199. AAAI.
- LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86:2278–2324.
- Lenzi, A., Bessac, J., Rudi, J., and Stein, M. L. (2023). Neural networks for parameter estimation in intractable models. *Computational Statistics & Data Analysis*, 185:107762.
- Lueckmann, J.-M., Bassetto, G., Karaletsos, T., and Macke, J. H. (2019). Likelihood-free inference with emulator networks. In *Proceedings of The 1st Symposium on Advances in Approximate Bayesian Inference*, volume 96, pages 32–53. PMLR.
- Lueckmann, J.-M., Goncalves, P. J., Bassetto, G., Öcal, K., Nonnenmacher, M., and Macke, J. H. (2017). Flexible statistical inference for mechanistic models of neural dynamics. In *Proceedings of the 31st Conference on Neural Information Processing Systems (NeurIPS 2017)*, pages 1290–1300, Red Hook, NY. Curran.
- Majumder, R. and Reich, B. (2023). A deep learning synthetic likelihood approximation of a non-stationary spatial model for extreme streamflow forecasting. *Spatial Statistics*, 55:100755.
- Majumder, R., Reich, B., and Shaby, B. (2023). Modeling extremal streamflow using deep learning approximations and a flexible spatial process. *Annals of Applied Statistics*, in press.
- Masserano, L., Dorigo, T., Izbicki, R., Kuusela, M., and Lee, A. B. (2023). Simulator-based inference with WALDO: Confidence regions by leveraging prediction algorithms and posterior estimators for inverse problems. In *Proceedings of the 26th International Conference on Artificial Intelligence and Statistics (AISTATS 2023)*, volume 206, pages 2960–2974. PMLR.
- Padoan, S. A., Ribatet, M., and Sisson, S. A. (2010). Likelihood-based inference for max-stable processes. *Journal of the American Statistical Association*, 105:263–277.
- Papamakarios, G. and Murray, I. (2016). Fast ε -free inference of simulation models with Bayesian conditional density estimation. In *Proceedings of the 30th Conference on Neural Information Processing Systems (NeurIPS 2016)*, pages 1036–1044, Red Hook, NY. Curran.

- Papamakarios, G., Nalisnick, E., Rezende, D. J., Mohamed, S., and Lakshminarayanan, B. (2021). Normalizing flows for probabilistic modeling and inference. *The Journal of Machine Learning Research*, 22:2617–2680.
- Papamakarios, G., Pavlakou, T., and Murray, I. (2017). Masked autoregressive flow for density estimation. In *Proceedings of the 31st Conference on Neural Information Processing Systems (NeurIPS 2017)*, pages 2339–2348, Red Hook, NY. Curran.
- Papamakarios, G., Sterratt, D., and Murray, I. (2019). Sequential neural likelihood: Fast likelihood-free inference with autoregressive flows. In *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS 2019)*, volume 89, pages 837–848. PMLR.
- Radev, S. T., Mertens, U. K., Voss, A., Ardizzone, L., and Köthe, U. (2022). BayesFlow: Learning complex stochastic models with invertible neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 33:1452–1466.
- Radev, S. T., Schmitt, M., Pratz, V., Picchini, U., Köthe, U., and Buerkner, P.-C. (2023). JANA: Jointly amortized neural approximation of complex Bayesian models. In *Proceedings of the 39th Conference on Uncertainty in Artificial Intelligence (UAI 2023)*, volume 216, pages 1695–1706. PMLR.
- Ramesh, P., Lueckmann, J.-M., Boelts, J., Tejero-Cantero, Á., Greenberg, D. S., Goncalves, P. J., and Macke, J. H. (2022). GATSBI: Generative adversarial training for simulation-based inference. In *Proceedings of the 10th International Conference on Learning Representations (ICLR 2022)*. Virtual: OpenReview. <https://openreview.net/forum?id=kR1hC6j48Tp>.
- R Core Team (2023). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.
- Richards, J., Sainsbury-Dale, M., Zammit-Mangion, A., and Huser, R. (2023). Likelihood-free neural Bayes estimators for censored peaks-over-threshold models. arXiv:2306.15642 [stat.ME].
- Rudi, J., Bessac, J., and Lenzi, A. (2022). Parameter estimation with dense and convolutional neural networks applied to the FitzHugh–Nagumo ODE. In *Proceedings of the 2nd Mathematical and Scientific Machine Learning Conference (MSML 2022)*, volume 145, pages 781–808. PMLR.
- Sainsbury-Dale, M., Richards, J., Zammit-Mangion, A., and Huser, R. (2023). Neural Bayes estimators for irregular spatial data using graph neural networks. arXiv:2310.02600 [stat.ME].
- Sainsbury-Dale, M., Zammit-Mangion, A., and Huser, R. (2024). Likelihood-free parameter estimation with neural Bayes estimators. *The American Statistician*, 78:1–14.
- Schlather, M. (2002). Models for stationary max-stable random fields. *Extremes*, 5:33–44.
- Schmitt, M., Bürkner, P.-C., Köthe, U., and Radev, S. T. (2024). Detecting model misspecification in amortized Bayesian inference with neural networks. In Köthe, U. and Rother, C., editors, *Pattern Recognition. DAGM GCPR 2023*, pages 541–557, Cham, Switzerland. Springer.
- Siahkoohi, A., Rizzuti, G., Orozco, R., and Herrmann, F. J. (2023). Reliable amortized variational inference with physics-based latent distribution correction. *Geophysics*, 88:297–322.

- Speiser, A., Yan, J., Archer, E. W., Buesing, L., Turaga, S. C., and Macke, J. H. (2017). Fast amortized inference of neural activity from calcium imaging data with variational autoencoders. In *Proceedings of the 31st Conference on Neural Information Processing Systems (NeurIPS 2017)*, pages 4025–4035, Red Hook, NY. Curran.
- Stein, M. L., Chi, Z., and Welty, L. J. (2004). Approximating likelihoods for large spatial data sets. *Journal of the Royal Statistical Society B*, 66:275–296.
- Talts, S., Betancourt, M., Simpson, D., Vehtari, A., and Gelman, A. (2018). Validating Bayesian inference algorithms with simulation-based calibration. arXiv:1804.06788 [stat.ME].
- Tejero-Cantero, A., Boelts, J., Deistler, M., Lueckmann, J.-M., Durkan, C., Goncalves, P. J., Greenberg, D. S., and Macke, J. H. (2020). sbi: A toolkit for simulation-based inference. *Journal of Open Source Software*, 5:2505.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. In *Proceedings of the 31st Conference on Neural Information Processing Systems (NeurIPS 2017)*, pages 5999–6009, Red Hook, NY. Curran.
- Vecchia, A. V. (1988). Estimation and model identification for continuous spatial processes. *Journal of the Royal Statistical Society B*, 50:297–312.
- von Krause, M., Radev, S. T., and Voss, A. (2022). Mental speed is high until age 60 as revealed by analysis of over a million participants. *Nature Human Behaviour*, 6:700–708.
- Wadsworth, J. L. and Tawn, J. A. (2012). Dependence modelling for spatial extremes. *Biometrika*, 99:253–272.
- Wang, Z., Hasenauer, J., and Schälte, Y. (2023). Missing data in amortized simulation-based neural posterior estimation. bioRxiv:2023.01.09.523219.
- Williams, R. J. and Zipser, D. (1989). A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1:270–280.
- Wiqvist, S., Frellsen, J., and Picchini, U. (2021). Sequential neural posterior and likelihood approximation. arXiv:2102.06522 [stat.ML].
- Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., and Yu, P. S. (2021). A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 32:4–24.
- Zaheer, M., Kottur, S., Ravanbakhsh, S., Póczos, B., Salakhutdinov, R. R., and Smola, A. J. (2017). Deep sets. In *Proceedings of the 31st Conference on Neural Information Processing Systems (NeurIPS 2017)*, pages 3392–3402, Red Hook, NY. Curran.
- Zammit-Mangion, A. and Wikle, C. K. (2020). Deep integro-difference equation models for spatio-temporal forecasting. *Spatial Statistics*, 37:100408.
- Zhang, S., Tong, H., Xu, J., and Maciejewski, R. (2019). Graph convolutional networks: a comprehensive review. *Computational Social Networks*, 6:11.
- Zhao, D., Dalmaso, N., Izbicki, R., and Lee, A. B. (2021). Diagnostics for conditional density models and Bayesian inference algorithms. In *Proceedings of the 37th Conference on Uncertainty in Artificial Intelligence (UAI 2021)*, volume 161, pages 1830–1840. PMLR.

- Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., and Sun, M. (2020). Graph neural networks: A review of methods and applications. *AI Open*, 1:57–81.
- Åkesson, M., Singh, P., Wrede, F., and Hellander, A. (2022). Convolutional neural networks as summary statistics for approximate Bayesian computation. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 19:3353–3365.