JANA: Jointly Amortized Neural Approximation of Complex Bayesian Models

Abstract

This work proposes "jointly amortized neural approximation" (JANA) of intractable likelihood functions and posterior densities arising in Bayesian surrogate modeling and simulation-based inference. We train three complementary networks in an end-to-end fashion: 1) a summary network to compress individual data points, sets, or time series into informative embedding vectors; 2) a posterior network to learn an amortized approximate posterior; and 3) a likelihood network to learn an amortized approximate likelihood. Their interaction opens a new route to amortized marginal likelihood and posterior predictive estimation – two important ingredients of Bayesian workflows that are often too expensive for standard methods. We benchmark the fidelity of JANA on a variety of simulation models against state-of-the-art Bayesian methods and propose a powerful and interpretable diagnostic for joint calibration. In addition, we investigate the ability of recurrent likelihood networks to emulate complex time series models without resorting to hand-crafted summary statistics.

1 INTRODUCTION

Surrogate modeling (SM) and simulation-based inference (SBI) are two crucial ingredients of a new generation of methods for simulation science (Lavin et al., 2021). From a Bayesian perspective, SM seeks to approximate the intractable likelihood function, whereas SBI aims to approximate the intractable posterior distribution of a complex generative model. Both problems are hard, as they involve multidimensional integrals which cannot be solved with standard analytical or numerical methods. Thus, specialized neural approximators have emerged as novel tools for taming the intractable (Cranmer et al., 2020).

We propose JANA ("Jointly Amortized Neural Approximation"), a Bayesian neural framework for *simultaneous amortized* SM and SBI, and show how it enables accurate solutions to challenging downstream tasks like the estimation of marginal and posterior predictive distributions (see Figure 1). JANA also presents a major qualitative upgrade to the BayesFlow framework (Radev et al., 2020), which was originally designed for amortized SBI alone.

It is commonly presumed that amortized SBI is wasteful (Greenberg et al., 2019; Papamakarios & Murray, 2016) and requires much larger simulation budgets than case-based SBI to make up for the much larger prediction domain. Our results challenge this premise. Given identical simulation budgets, JANA outperforms or is on par with ABC-SMC, SNL, SNPE, SNRE, and SNPLA (see Figure 4). We hypothesize that modern neural networks benefit strongly from a broad simulation scope. Thanks to their excellent generalization capabilities, they can exploit outcomes from the entire prior predictive distribution of a simulation to improve local accuracy for each specific case. In this sense, amortized inference seems to be a natural by-product of deep neural modeling, and the initial training effort more than repays with global diagnostics, nearly instant estimation at test time, and no loss in accuracy.

JANA is thus a promising catalyst for the fully Bayesian analysis of massive data sets that used to be out of reach for existing methods[1]. We show that jointly amortized SM and SBI unlock the potential of powerful Bayesian tools for model comparison, validation, and calibration, which are essential in Bayesian workflows (Gelman et al., 2020), but widely underutilized in current simulation-based contexts. For one, JANA offers an efficient way to compute *marginal likelihoods* via the probabilistic change-of-variables formula (instead of integration over the model's entire prior space) as a prerequisite for *prior predictive* model selection (i.e., probabilistic Occam's razor). For another, it can rapidly produce both posterior samples and normalized likelihood estimates

[1]Source code for JANA will be released upon acceptance.

Submitted to the 39th Conference on Uncertainty in Artificial Intelligence (UAI 2023). **To be used for reviewing only**.

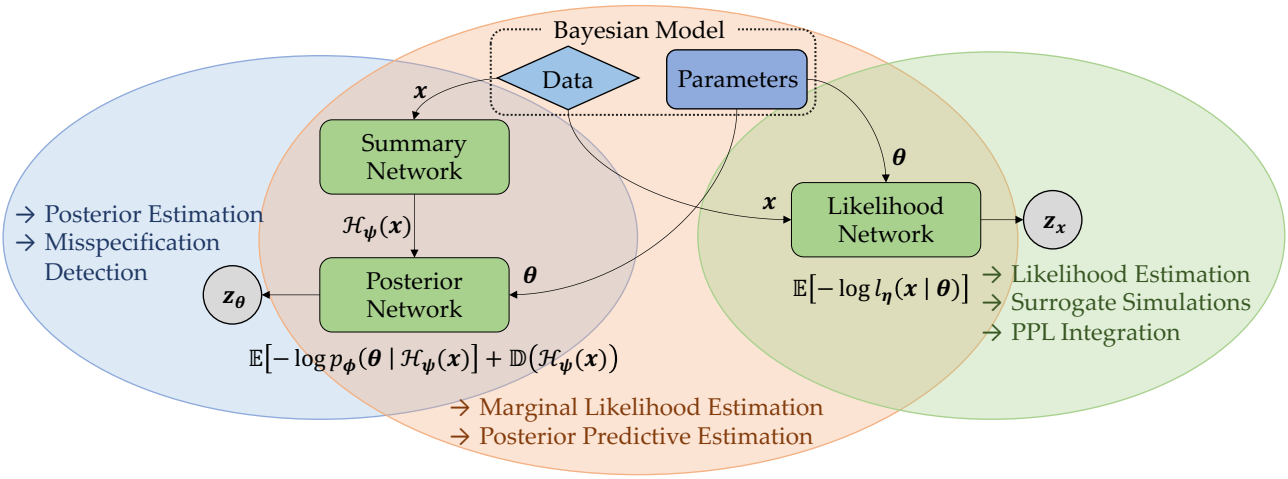


Figure 1: A conceptual illustration of our method for jointly amortized neural approximation (JANA). On the one hand, the summary and posterior network can perform amortized posterior estimation and detect model misspecification. On the other hand, the likelihood network can perform amortized likelihood estimation, surrogate simulations, and interact with probabilistic programming languages (PPLs). Together, the two networks enable posterior predictive and marginal likelihood estimation, which allow for amortized Bayesian model comparison and validation.

of new data instances, as are needed in strong validation procedures of the *posterior predictive* performance (Vehtari & Ojanen, 2012). In other words, JANA can directly quantify both prior and posterior predictive performance without resorting to Markov chain Monte Carlo (MCMC) sampling or costly model re-fits, in addition to the well-studied advantages of individual posterior or likelihood networks (see Figure 1).

In summary, our key contributions are:

1. We develop a neural architecture for fully amortized joint posterior estimation and likelihood emulation;
2. We propose a powerful and interpretable method to test for joint calibration of the networks;
3. We extensively validate our new architecture on analytic toy examples and complex simulation models;
4. We show how our joint architecture solves the challenges of computing both out-of-sample predictive performance and intractable marginal likelihoods;
5. We demonstrate a recurrent neural likelihood for surrogate simulations in a complex time series model.

2 METHOD

2.1 PROBLEM FORMULATION

Bayesian Models We focus on generative Bayesian models specified as a triple $\mathcal{M}=\big(G(\boldsymbol{\theta},\boldsymbol{\xi}), p(\boldsymbol{\xi}\,|\,\boldsymbol{\theta}), p(\boldsymbol{\theta})\big)$. Such models yield observables $\boldsymbol{x} \in \mathcal{X}$ according to the system

$$\boldsymbol{x} = G(\boldsymbol{\theta},\boldsymbol{\xi}) \quad \text{with} \quad \boldsymbol{\xi} \sim p(\boldsymbol{\xi}\,|\,\boldsymbol{\theta}),\ \boldsymbol{\theta} \sim p(\boldsymbol{\theta}), \tag{1}$$

where G denotes a simulation program, $\boldsymbol{\xi} \in \Xi$ denotes externalized randomness (i.e., noise or pseudorandom program states) with density function $p(\boldsymbol{\xi}\,|\,\boldsymbol{\theta})$, and $p(\boldsymbol{\theta})$ encodes prior knowledge about plausible simulation parameters $\boldsymbol{\theta} \in \Theta$.

Forward Inference Running the simulator G with a fixed parameter configuration $\boldsymbol{\theta}$ and different values of $\boldsymbol{\xi}$ is equivalent to random draws from an *implicit likelihood* $p(\boldsymbol{x}\,|\,\boldsymbol{\theta})$:

$$\boldsymbol{x} \sim p(\boldsymbol{x}\,|\,\boldsymbol{\theta}) \iff \boldsymbol{x} = G(\boldsymbol{\theta},\boldsymbol{\xi}) \quad \text{with} \quad \boldsymbol{\xi} \sim p(\boldsymbol{\xi}\,|\,\boldsymbol{\theta}) \tag{2}$$

In theory, implicit likelihoods can be obtained by marginalizing the joint distribution $p(\boldsymbol{\xi},\boldsymbol{x}\,|\,\boldsymbol{\theta})$ over all possible execution trajectories of the simulation program (i.e., over $\boldsymbol{\xi}$), but this is typically intractable (Cranmer et al., 2020).

Inverse Inference In Bayesian analysis, we want to infer a model's latent parameters $\boldsymbol{\theta}$ from manifest data $\boldsymbol{x}$ through the probabilistic factorization of the joint distribution into prior and (implicit) likelihood:

$$p(\boldsymbol{\theta}\,|\,\boldsymbol{x}) \propto p(\boldsymbol{\theta},\boldsymbol{x}) = p(\boldsymbol{\theta}) \int_{\Xi} p(\boldsymbol{\xi},\boldsymbol{x}\,|\,\boldsymbol{\theta})\,\mathrm{d}\boldsymbol{\xi}. \tag{3}$$

Since we assume that the likelihood is not available in closed form, we also cannot access the posterior $p(\boldsymbol{\theta}\,|\,\boldsymbol{x})$ and perform parameter estimation through gold-standard Bayesian methods, such as Hamiltonian Monte Carlo (HMC)-MCMC (Carpenter et al., 2017).

Marginal Likelihoods In addition to estimating parameters, modelers often want to compare and assign preferences to competing models. From a Bayesian perspective, the canonical measure of evidence for a given model is the

marginal likelihood (aka the *prior predictive distribution*),

$$p(\boldsymbol{x}) = \int_{\Theta} \int_{\Xi} p(\boldsymbol{\theta})\, p(\boldsymbol{\xi}, \boldsymbol{x} \,|\, \boldsymbol{\theta})\, \mathrm{d}\boldsymbol{\xi}\, \mathrm{d}\boldsymbol{\theta}, \tag{4}$$

which is doubly intractable for complex models because both involved integrals are highly difficult to approximate with sufficient precision (Meng & Wong, 1996). The estimation of the marginal likelihood is central to Bayesian model comparison (BMC), since it naturally embodies a probabilistic version of Occam's razor by penalizing the prior complexity of a model (MacKay, 2003). Thus, it allows us to express our preference for a simpler model over a more complex one, given that both models can account for the observed data equally well.

Posterior Predictive Distribution Bayesian models can also be compared and validated on the basis of their posterior predictive performance (Vehtari & Ojanen, 2012). However, many posterior predictive metrics rely on the likelihood density being available analytically. In particular, this is true for the expected log-predictive density (ELPD), which is a widely-applied, general-purpose metric to measure (out-of-sample) posterior predictive performance when no application-specific utilities are known (Vehtari et al., 2017). For K (new) observations $\boldsymbol{x}_{\text{new}}^{(k)}$ not previously seen by the model, the ELPD can be defined as

$$\text{ELPD} = \sum_{k=1}^{K} \log \int_{\Theta} p(\boldsymbol{x}_{\text{new}}^{(k)} \,|\, \boldsymbol{\theta})\, p(\boldsymbol{\theta} \,|\, \boldsymbol{x})\, \mathrm{d}\boldsymbol{\theta}. \tag{5}$$

The ELPD has a strong connection to information theory (Vehtari & Ojanen, 2012) and is widely used in Bayesian cross-validation (Vehtari et al., 2017), where it is one of the most prominent sources of computational intractability.

Probabilistic Symmetry Our joint training will leverage the symmetry in the arguments of $p(\boldsymbol{\theta} \,|\, \boldsymbol{x})$ and $p(\boldsymbol{x} \,|\, \boldsymbol{\theta})$, along with the fact that a single run of the simulator (Eq. 1) yields a reusable tuple of parameters and synthetic data $(\boldsymbol{\theta}, \boldsymbol{x})$. However, many simulation models are characterized by a relatively low-dimensional parameter space Θ (e.g., low-dimensional vectors) and a rather high-dimensional data space $\mathcal{X}$ (e.g., multivariate time series or sets of exchangeable observations). Thus, we need different neural architectures, each separately aligned with the structural properties of $p(\boldsymbol{\theta} \,|\, \boldsymbol{x})$ and $p(\boldsymbol{x} \,|\, \boldsymbol{\theta})$.

2.2 POSTERIOR NETWORK

The posterior network $\mathcal{P}_{\boldsymbol{\phi}}$ implements a normalizing flow between $\boldsymbol{\theta}$ and a latent variable $\boldsymbol{z}_{\boldsymbol{\theta}}$ with a simple density (e.g., Gaussian) given observed or simulated data $\boldsymbol{x}$:

$$p_{\boldsymbol{\phi}}(\boldsymbol{\theta} \,|\, \boldsymbol{x}) = p(\boldsymbol{z}_{\boldsymbol{\theta}}) \left| \det \left(\frac{\partial \boldsymbol{z}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}} \right) \right| \tag{6}$$

$$\boldsymbol{z}_{\boldsymbol{\theta}} = \mathcal{P}_{\boldsymbol{\phi}}(\boldsymbol{\theta}; \boldsymbol{x}). \tag{7}$$

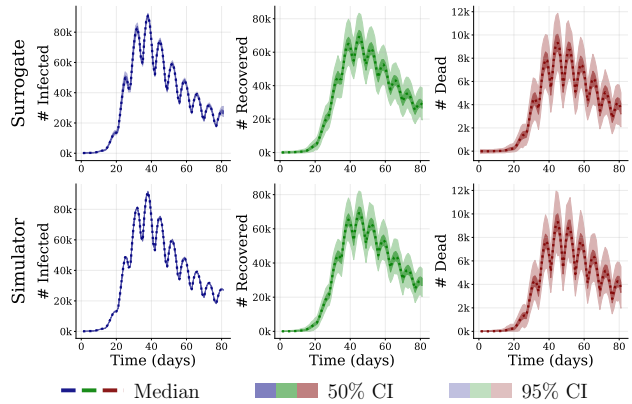


Figure 2: Recurrent likelihood networks can emulate complex Bayesian stochastic differential equation (SDE) models of disease outbreaks (see **Experiment 4**).The top and bottom row each depict 1 000 simulations (same $\boldsymbol{\theta}$) from the surrogate and the actual simulator, respectively.

The normalizing flow is realized via a conditional invertible neural network (cINN) composed by a series of conditional coupling layers, as utilized in Ardizzone, Kruse, et al. (2019) and Radev et al. (2020). Since the observed or simulated data will typically have a complex structure and/or contain varying numbers of observations, the posterior cINN includes a trainable summary network sub-module $\mathcal{H}_{\boldsymbol{\psi}}$ (see Radev et al., 2020) which we optimize alongside to extract maximally informative data representations $\mathcal{H}_{\boldsymbol{\psi}}(\boldsymbol{x})$ in an end-to-end manner.

The design of the conditional coupling layers follows the work of (Ardizzone, Kruse, et al., 2019; Ardizzone, Lüth, et al., 2019; Radev et al., 2020), since compositions of such layers exhibit favorable theoretical properties (Draxler et al., 2022) and remarkable empirical performance on high-dimensional unstructured data (Dinh et al., 2016; Kingma & Dhariwal, 2018) or complex Bayesian models in various domains (Bellagente et al., 2022; Bieringer et al., 2021; Radev et al., 2021; von Krause et al., 2022). However, any other coupling design can be used as a plug-in replacement.

2.3 LIKELIHOOD NETWORK

The likelihood network $\mathcal{L}_{\boldsymbol{\eta}}$ implements a normalizing flow between $\boldsymbol{x}$ and a (multivariate) Gaussian latent variable $\boldsymbol{z}_{\boldsymbol{x}} = \mathcal{L}_{\boldsymbol{\eta}}(\boldsymbol{x}; \boldsymbol{\theta})$ given a parameter configuration $\boldsymbol{\theta}$,

$$l_{\boldsymbol{\eta}}(\boldsymbol{x} \,|\, \boldsymbol{\theta}) = p(\boldsymbol{z}_{\boldsymbol{x}}) \left| \det \left(\frac{\partial \boldsymbol{z}_{\boldsymbol{x}}}{\partial \boldsymbol{x}} \right) \right|. \tag{8}$$

This formulation is similar to the pushforward expression for the posterior network (Eq. 6), but with $\boldsymbol{\theta}$ swapped for $\boldsymbol{x}$. The likelihood network, like the posterior network, is also implemented as a cINN. As the conditioning information is now the parameter vector $\boldsymbol{\theta}$ (and not a complex data

structure), it can be fed directly to the conditional coupling layers of the cINN without an additional summary network.

However, since the data $\boldsymbol{x}$ (i.e., simulator outputs) is typically in non-vector form, the design of the coupling layers needs to be tailored according to the probabilistic symmetry of $p(\boldsymbol{x} \mid \boldsymbol{\theta})$. Learning $p(\boldsymbol{x} \mid \boldsymbol{\theta})$ in its raw form is typically much harder than learning the likelihood $p(\mathcal{H}(\boldsymbol{x}) \mid \boldsymbol{\theta})$ of some (learned or hand-crafted) summary statistics $\mathcal{H}(\boldsymbol{x})$, since the latter are already in a compressed vector form and do not require specialized architectures. JANA can learn either $p(\mathcal{H}(\boldsymbol{x}) \mid \boldsymbol{\theta})$ or $p(\boldsymbol{x} \mid \boldsymbol{\theta})$, as required by the particular application or dictated by the (un-)availability of good summary statistics. In our experiments, we directly target $p(\boldsymbol{x} \mid \boldsymbol{\theta})$ and the **Appendix** details how to design likelihood networks for exchangeable or Markovian data.

2.4 SIMULATION-BASED TRAINING

In contrast to previous joint learning approaches (Wiqvist et al., 2021), we aim for a fully amortized approach: Once the networks have converged, we want to evaluate the normalized densities $p_{\boldsymbol{\phi}}(\boldsymbol{\theta} \mid \boldsymbol{x})$ and $l_{\boldsymbol{\eta}}(\boldsymbol{x} \mid \boldsymbol{\theta})$ for *any* pair $(\boldsymbol{\theta}, \boldsymbol{x})$ consistent with a generative model $\mathcal{M}$. In addition, we want to generate conditional random draws $\boldsymbol{\theta} \mid \boldsymbol{x}$ and $\boldsymbol{x} \mid \boldsymbol{\theta}$ from both networks for parameter estimation and surrogate modeling. Finally, we want to prescribe a simple distribution to the summary network outputs $q\big(\mathcal{H}_{\boldsymbol{\psi}}(\boldsymbol{x})\big)$ in order to detect atypical data during inference (i.e., model misspecification) and highlight potential posterior errors (Schmitt et al., 2022). Thus, we minimize the following criterion:

$$\min_{\boldsymbol{\phi},\boldsymbol{\psi},\boldsymbol{\eta}} \mathbb{E}_{p(\boldsymbol{\theta},\boldsymbol{x})}\big[-(\log p_{\boldsymbol{\phi}}(\boldsymbol{\theta} \mid \mathcal{H}_{\boldsymbol{\psi}}(\boldsymbol{x})) + \log l_{\boldsymbol{\eta}}(\boldsymbol{x} \mid \boldsymbol{\theta}))\big] + \lambda \cdot \mathbb{MMD}^2\big[p(\mathcal{H}_{\boldsymbol{\psi}}(\boldsymbol{x})) \,||\, \mathcal{N}(\mathbf{0}, \mathbb{I})\big] \tag{9}$$

where $\mathbb{MMD}^2$ is the maximum mean discrepancy (MMD; Gretton et al., 2012) between the distribution of summary network outputs and a unit Gaussian density. This divergence imposes a probabilistic structure on the summary space learned by $\mathcal{H}_{\boldsymbol{\psi}}(\boldsymbol{x})$ and enables error detection and model criticism during inference (to be explained shortly, see also Schmitt et al., 2022). We approximate the expectation over $p(\boldsymbol{\theta}, \boldsymbol{x})$ via simulations from the generative model $\mathcal{M}$ and repeat this process until the networks converge (i.e., simulation-based training).

Proper minimization of the criterion in Eq. 9 results in correct posterior and likelihood approximation, along with an interpretable summary space. However, the objective promises self-consistency only in the "small world", as it does not guarantee correct posterior inference or likelihood evaluation in the real world when there may be a simulation gap. This is due to the fact that simulation-based training optimizes the expectation with respect to the Bayesian joint model $p(\boldsymbol{\theta}, \boldsymbol{x})$, but not (necessarily) the empirical data distribution $p^*(\boldsymbol{x})$. Thus, the MMD term allows us to detect potential simulation gaps during inference via distribution matching (Schmitt et al., 2022). Moreover, the posterior network can serve as a "critic" for the likelihood network by rejecting surrogate simulations which are judged to be highly unlikely under the true simulator.

2.5 VALIDATION METHODOLOGY: JOINT CALIBRATION

Faithful uncertainty representation (i.e., calibration) is an essential precondition for self-consistent and interpretable simulation-based inference. Simulation-based calibration (SBC; Talts et al., 2018) is a general diagnostic method which considers the performance of a sampling algorithm over the entire joint distribution $p(\boldsymbol{\theta}, \boldsymbol{x})$, regardless of the specific probabilistic structure of a model.

SBC leverages the generative nature of Bayesian models as well as the self-consistency of the Bayesian joint model $G(\boldsymbol{\theta}, \boldsymbol{\xi})$ in the following sense: For all quantiles $q \in (0, 1)$, all uncertainty regions $U_q(\boldsymbol{\theta} \mid \boldsymbol{x})$ of $p(\boldsymbol{\theta} \mid \boldsymbol{x})$ are well calibrated, as long as the generating distribution of the assumed model is equal to true data-generating distribution and posterior computation is exact (Talts et al., 2018). We can formally write this property as

$$q = \int_{\mathcal{X}} \int_{\Theta} \mathbb{I}_{[\boldsymbol{\theta}^* \in U_q(\boldsymbol{\theta} \mid \boldsymbol{x})]}\, p(\boldsymbol{x} \mid \boldsymbol{\theta}^*)\, p(\boldsymbol{\theta}^*)\, \mathrm{d}\boldsymbol{x}\, \mathrm{d}\boldsymbol{\theta}^*, \tag{10}$$

where $\boldsymbol{\theta}^*$ is the true data-generating parameter and $\mathbb{I}_{[\cdot]}$ is the indicator function. If the posterior network $\mathcal{P}_{\boldsymbol{\phi}}$ generates draws from the true posterior and the likelihood network $\mathcal{L}_{\boldsymbol{\eta}}$ mimics the simulator perfectly, then the equality implied by Eq. 10 holds regardless of the particular form of the true likelihood or the true posterior. Thus, any violation of this equality indicates some error incurred by joint training, so we refer to our validation procedure as joint simulation-based calibration (JSBC).

The reasons for faulty JSBC can be any combination of (i) inaccurate representation of the posterior; (ii) inaccurate representation of the likelihood; or (iii) an erroneous implementation of the simulation model itself. To differentiate between (i) and (ii), we can first run standard SBC for the posterior network using data draws from the actual simulator instead of the likelihood network. If this check passes, but subsequently JSBC fails, the calibration problems must stem from the likelihood network. Thereby, we can use the posterior network for *model criticism* of the likelihood network, which would otherwise be infeasible for most Bayesian models.

As part of a Bayesian workflow (Gelman et al., 2020), SBC can quickly become infeasible for case-based methods, as it requires independent posterior draws from hundreds or thousands of simulated data sets. However, it is effortless for amortized methods, as we can obtain many posterior draws

from multiple data sets in a matter of seconds. In practice, we follow Säilynoja et al. (2022) by transforming the posterior draws intro fractional rank statistics and computing their empirical cumulative distribution functions (ECDFs). This method provides *simultaneous confidence bands* and eliminates the need to manually select a binning parameter (e.g., as required by histogram-based methods).

2.6 USE CASES FOR JOINT LEARNING

Posterior Predictive Estimation Estimating the expected predictive performance of a Bayesian model (Eq. 5) requires an analytic expression for the pointwise (i.e., per-observation) likelihood function $p(\boldsymbol{x}_{\text{new}}^{(k)} \mid \boldsymbol{\theta})$ at arbitrary new data $\boldsymbol{x}_{\text{new}}^{(k)}$ (Bürkner et al., 2021). For this reason, the ELPD cannot be computed for Bayesian models with intractable likelihoods or sequential neural estimators.

Moreover, even if the likelihood itself were analytic, the integral in Eq. (5) would still be intractable for most models. It can be efficiently approximated using posterior draws, but doing so in the context of cross-validation requires importance sampling or costly model refits (Vehtari et al., 2017). Hence, evaluating the ELPD for arbitrary cross-validation schemes critically requires both the amortized likelihood and posterior approximator.

Given data used for model fitting $\boldsymbol{x}$ and upcoming data $\boldsymbol{x}_{\text{new}}^{(k)}$, the two networks can estimate a model's expected predictive performance in two steps. First, we can obtain a large amount of S random draws from the amortized posterior given $\boldsymbol{x}$:

$$\boldsymbol{\theta}^{(s)} \sim p_{\boldsymbol{\phi}}(\boldsymbol{\theta} \mid \mathcal{H}_{\boldsymbol{\psi}}(\boldsymbol{x})) \text{ for } s = 1, ..., S. \tag{11}$$

Then, the likelihood network can approximate the ELPD at all $\boldsymbol{x}_{\text{new}}^{(k)}$ given $\{\boldsymbol{\theta}^{(s)}\}$ via its Monte Carlo estimate:

$$\widehat{\text{ELPD}} = \sum_{k=1}^{K} \log \frac{1}{S} \sum_{s=1}^{S} l_{\boldsymbol{\psi}}(\boldsymbol{x}_{\text{new}}^{(k)} \mid \boldsymbol{\theta}^{(s)}) \tag{12}$$

In the context of cross-validation (CV), $\boldsymbol{x}$ and $\boldsymbol{x}_{\text{new}}$ refer to a random data split, and we can estimate the predictive performance of a Bayesian model by summing over the $\widehat{\text{ELPD}}$s from all data splits. Through amortization, JANA can efficiently compute the ELPD for any CV scheme. In **Experiment 3**, we demonstrate this for leave-one-out (LOO)-CV, which is one of the most expensive validation methods.

Marginal Likelihood Estimation Bayesian (prior) predictive model comparison depends on computing a marginal likelihood (Eq. 4). We can leverage the probabilistic change of variable, which results directly from Bayes' rule:

$$\log \widehat{p}(\boldsymbol{x}) = \log l_{\boldsymbol{\eta}}(\boldsymbol{x} \mid \boldsymbol{\theta}) + \log p(\boldsymbol{\theta}) - \log p_{\boldsymbol{\phi}}(\boldsymbol{\theta} \mid \mathcal{H}_{\boldsymbol{\psi}}(\boldsymbol{x})). \tag{13}$$

Thus, for any dataset, we can obtain an estimate of the log marginal likelihood (LML) by evaluating Eq. 6 and Eq. 8, along with the prior density $p(\boldsymbol{\theta})$. Evaluating all above terms is infeasible with standard Bayesian methods, since either the normalized posterior, the likelihood, or both quantities are typically intractable. Bridge sampling (Meng & Wong, 1996) enables the approximation of marginal likelihoods from posterior draws, but only works for models with analytical likelihoods and in tandem with non-amortized MCMC.

From a Bayesian perspective, evaluating Eq. 13 across multiple data sets amounts to *amortized bridge sampling*. At the same time, we can use Eq. 13 for assessing non-convergence or problems during inference by evaluating the RHS for a fixed $\boldsymbol{x}$ and different $\boldsymbol{\theta}$ drawn from the approximate posterior. Under perfect convergence, the RHS of Eq. 13 is independent of $\boldsymbol{\theta}$, so any ensuing variation is a measure of pure approximation error.

Surrogate Simulators In some modeling scenarios, the simulator might be a large-scale computer program implementing a complex generative algorithm (Lavin et al., 2021). Thus, a simulation-based inference workflow might be severely limited by the inability to obtain a large amount of simulations in a reasonable time. In such cases, an amortized surrogate simulator can generate additional data for the posterior network or a black-box optimizer (Gutmann & Corander, 2016). A notable advantage of neural surrogate simulators is that they can directly emulate complex data without summary statistics (see Figure 2). In addition, they can render a non-differentiable simulator differentiable for downstream tasks, such as amortized design optimization (Ivanova et al., 2021) or interact with MCMC samplers (Boelts et al., 2022; Fengler et al., 2021).

3 RELATED WORK

Approximate Bayesian Computation An established approach to SBI is embodied by approximate Bayesian computation (ABC; Marin et al., 2012; Sisson et al., 2018). ABC is a family of algorithms where the simplest one, "ABC rejection", generates draws from an approximate posterior by repeatedly proposing parameters from the prior distribution, and then simulating a corresponding synthetic data set by running the simulator with the proposed parameters. More sophisticated ABC samplers are Sequential Monte Carlo (ABC-SMC; Beaumont et al., 2009; Del Moral et al., 2012; Picchini & Tamborrino, 2022; Toni, 2011) and Markov chain Monte Carlo ABC (ABC-MCMC; Marjoram et al., 2003; Picchini, 2014). In ABC, raw data are typically reduced via summary functions. However, *handcrafted* summary statistics are often insufficient, which results in a leak of information about the parameters (Marin et al., 2018). Recent work has used neural networks to learn informative summary statistics of model parameters in ABC (Chen et al., 2021; Jiang et al., 2017; Wiqvist et al., 2019).

Synthetic Likelihoods and Particle MCMC Despite being intuitive to grasp and use, the above ABC methods are notoriously inefficient, typically requiring millions of model simulations, which can be prohibitive for expensive simulators. Another established SBI alternative, also based on data-reduction via summary statistics, is *synthetic likelihood* (Price et al., 2018; Wood, 2010), which is more suitable for high-dimensional summary statistics. However, since synthetic likelihood is typically implemented as an MCMC sampler where multiple datasets are simulated at each proposed $\boldsymbol{\theta}$, it can also be computationally intensive. Particle MCMC (Andrieu et al., 2010) is a simulation-based method for exact Bayesian inference which has found considerable success, especially for state-space models. However, particle MCMC could be infeasible when multiple inference runs are required to separately fit several different data sets.

Neural Posterior Estimation Methods for neural posterior estimation either specialize a neural approximator for inference on a single observation[2] (Deistler et al., 2022; Durkan et al., 2020; Greenberg et al., 2019; Lueckmann et al., 2017; Papamakarios & Murray, 2016), or inference across arbitrary many observations (Ardizzone, Kruse, et al., 2019; Avecilla et al., 2022; Gonçalves et al., 2020; Pacchiardi & Dutta, 2022; Radev et al., 2020). The former methods perform *sequential estimation* by iteratively refining the prior to generate simulations in the vicinity of the observation. Thus, they are *not amortized*, as each new observation necessitates a costly re-training of the neural approximator. In contrast, the latter methods can perform *amortized inference*, as the neural approximator is trained to generalize over the entire prior predictive distribution and can be queried for any observation assumed to arise from the Bayesian model. Importantly, amortization can be performed over any aspect of the model, including data sets (Gonçalves et al., 2020) or other contextual factors, such as the number of observations in a data set or the number of time points in a time series (Radev et al., 2020).

Neural Likelihood Estimation A related family of neural methods directly targets the intractable likelihood function instead of the posterior (Boelts et al., 2022; Fengler et al., 2021; Hermans et al., 2020; Lueckmann et al., 2019; Papamakarios et al., 2019). The endpoint of these methods is an *amortized likelihood approximator* which can mimic a complex simulator or be used in tandem with a (non-amortized) MCMC sampler for posterior estimation. The latter can be prohibitively time-consuming, since it not only requires expensive simulation-based training, but also integrating likelihood approximators into non-amortized MCMC. This makes validating the posteriors (e.g., via simulation-based-calibration; SBC; Säilynoja et al., 2022; Talts et al., 2018) challenging or even impossible in practice. Nevertheless, likelihood approximators have certain advantages over posterior approximators, for instance, they do not need to be retrained for different priors and can emulate the behavior of large-scale simulators (Lavin et al., 2021).

[2]The term *observation* may refer to an entire data set, depending on how the data is used to update the posterior.

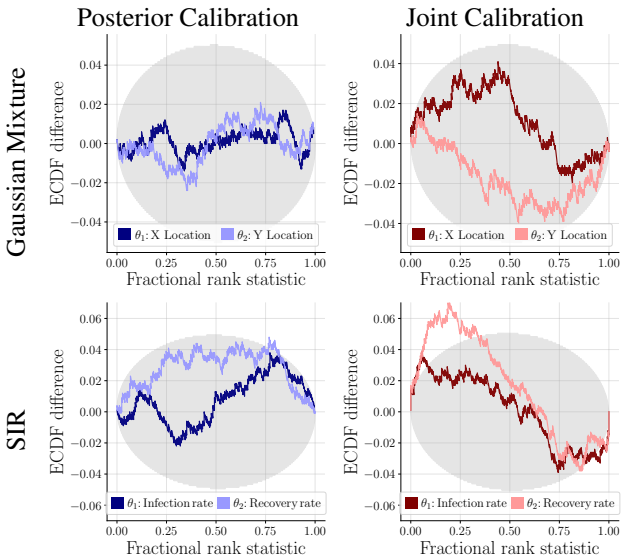


Figure 3: **Experiment 1.** Example calibration tests for 2 of the more challenging benchmarks. *Top row:* Good posterior and joint calibration of JANA for the Gaussian Mixture model. *Bottom row:* Posterior and joint calibration can be used in tandem to detect an underperforming likelihood network in the SIR model. The posterior network alone induces no systematic deviations when applied to simulator outputs (bottom left), but overestimates the parameters given the outputs of the surrogate network (bottom right).

Joint Posterior and Likelihood Estimation In a pioneering work, Wiqvist et al. (2021) propose the SNPLA method to embody the best of both worlds by jointly training a posterior and a likelihood approximator. However, SNPLA operates in a sequential manner and thus does not yield fully amortized approximators. Moreover, it relies on summary statistics for the likelihood approximation. Accordingly, in the current work, we unify BayesFlow (Radev et al., 2020) and SNPLA (Wiqvist et al., 2021) into a framework for fully amortized joint estimation without manual summary statistics and explore the ensuing benefits for Bayesian modeling.

4 EXPERIMENTS

In the following, we will illustrate the utility of JANA in twelve Bayesian models across four experiments. For **Experiments 1–3**, we train the networks without the MMD criterion in Eq. 9 (i.e., $\lambda = 0$), because our validations feature no model misspecification. The **Appendix** contains code for reproducing all experiments.

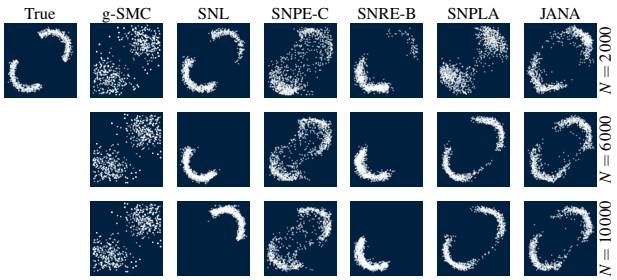
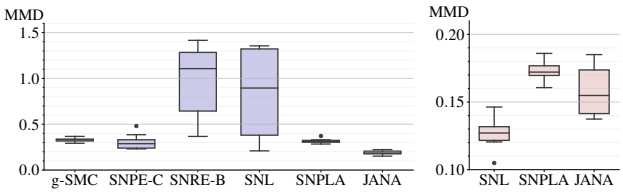


Figure 4: **Experiment 2**. Samples from the approximate posterior distribution for one repetition of the Two Moons experiment (repetition #2).



(a) Posterior MMD (b) Joint MMD

Figure 5: **Experiment 2**. Performance with N=2 000 training simulations, as indexed by sampling-based MMD estimate to analytical distribution (lower is better).

4.1 TEN BENCHMARK EXPERIMENTS

Setup This experiment demonstrates the fidelity of our proposed architecture as well as the utility of our calibration checks to diagnose approximation faults on a set of ten benchmark simulation models proposed by Lueckmann et al. (2021). Since these benchmarks were originally designed for (non-amortized) neural posterior estimation, we deviate from the original problem setting by (i) approximating both posterior and likelihood; and (ii) validating our results on a much larger held-out set of 1 000 simulations (as compared to just 10). Our goal here is *not to propose a better method for posterior estimation*, but to demonstrate the feasibility of joint amortization and the utility of the JSBC diagnostic on a set of popular and rather diverse models.

For each benchmark, we train our networks with a fixed budget of 10 000 simulations, as we consider this to be a challenging setting in the low-to-medium training data availability. All benchmarks and neural approximators are implemented in the BayesFlow library. See the **Appendix** and the accompanying code for more details and complete diagnostics.

Results Overall, we observe stable training and good calibration across the ten benchmarks models, with Bernoulli GLM Raw and SIR exhibiting systematic joint miscalibration due likelihood approximations errors (see **Appendix** for detailed results). Figure 3 illustrates the utility of our JSBC diagnostic to reveal both good calibration (i.e., ECDF trajectories completely contained in the confidence ellipsis for the Gaussian Mixture benchmark) as well as systematic deviations owing to the likelihood network (i.e., ECDF trajectories partially outside the confidence ellipsis for SIR). Moreover, due to the inherent interpretability of JSBC, we can pinpoint the reasons for joint miscalibration of the SIR model. The likelihood network tends to generate more rapid synthetic outbreaks than the actual model, which leads to the posterior network overestimating the parameters of surrogate simulations.

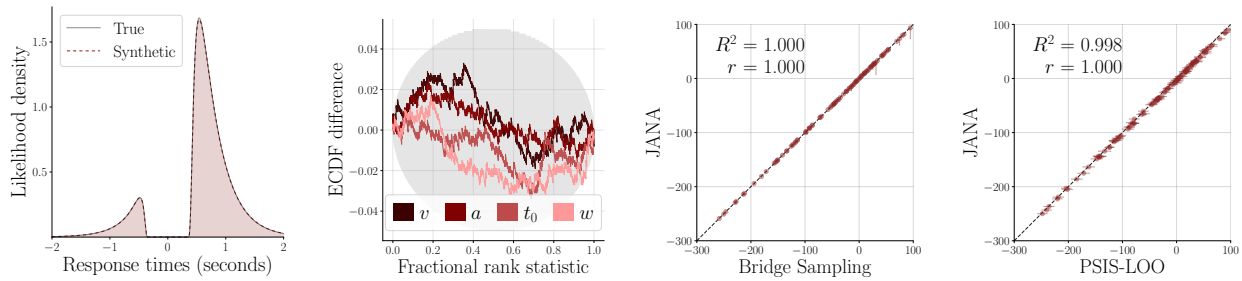
4.2 TWO MOONS: METHOD COMPARISON

Setup Here, we focus specifically on the Two Moons benchmark (Greenberg et al., 2019; Lueckmann et al., 2021) and use the code from Wiqvist et al. (2021) to compare JANA with the popular sequential methods SNL (Papamakarios et al., 2019), SNPE-C (Greenberg et al., 2019), SNRE-B (Durkan et al., 2020), SNPLA (Wiqvist et al., 2021), and a recent ABC-SMC algorithm with "guided particles" (here abbreviated with g-SMC, which is the method called "hybrid" in Picchini & Tamborrino, 2022). The model is characterized by a bimodal posterior with two separated crescent moons for the observed point $\boldsymbol{x}_{\text{new}} = (0, 0)^\top$ which a posterior approximator needs to recover. We train SNL, SNPE-C, SNRE-B, SNPLA, g-SMC, and JANA following the same setup from Wiqvist et al. (2021). For each method, we repeat the experiment ten times using a fixed budget of 2 000, 6 000, and 10 000 simulations and subsequently obtain 1 000 posterior draws from the converged methods. For a numerical evaluation, we apply MMD between the approximate and analytical distributions.

Results JANA consistently explores both crescent moons throughout all repetitions, and already captures the local patterns of the posterior after 2 000 training samples (see Figure 4). With respect to posterior performance, JANA even outperforms all sequential methods which are tailored to one observed data set (see Figure 5a). The joint performance of our amortized method is comparable to non-amortized sequential methods, see Figure 5b. In light of these and previous results, *amortization across data sets* seems to be a reasonable choice even with limited simulation budgets, especially since sequential (non-amortized) methods may be infeasible for large data sets (Hermans et al., 2021).

4.3 EXCHANGEABLE DIFFUSION MODEL

Setup This example demonstrates amortized marginal likelihood and ELPD estimation based on a mechanistic model of decision making: the diffusion model (DM; Ratcliff & McKoon, 2008). We benchmark our results on state-of-the-art likelihood-based methods. First, we compare our marginal likelihood estimates with those obtained using



(a) Likelihood emulation (b) Joint calibration (c) log marginal likelihood (d) ELPD

Figure 6: **Experiment 3.** The true and synthetic likelihood align perfectly (a). The joint approximation of all parameters is well calibrated (b). Both the log marginal likelihood (c) and ELPD (d) estimates of JANA closely approximate those obtained via bridge sampling and PSIS-LOO. Each point in (c) and (d) represents one out of 100 held-out simulations.

bridge sampling (Gronau et al., 2017). Second, we compare our ELPD estimates with those obtained using Pareto smoothed importance sampling (PSIS)-LOO (Vehtari et al., 2017). Both methods use random draws obtained via HMC-MCMC as implemented in Stan (Carpenter et al., 2017).

Results Our results indicate well-calibrated joint approximation (see Figure 6b) as well as accurate posterior and likelihood estimation (see Figure 6c and 6d). For the approximation of marginal likelihoods, we first perform amortized posterior sampling on the 100 held-out data sets. We then evaluate the approximate likelihood on these samples and finally apply Eq. 13 to compute the log marginal likelihood (LML). Our results show a very close correspondence between our neural LMLs and those obtained via MCMC-based bridge sampling (see Figure 6c). Furthermore, our amortized LOO-CV estimates align very closely with the estimates obtained via PSIS-LOO (see Figure 6d).

MCMC Integration Surrogate likelihoods provide all information that is needed for MCMC sampling. We provide an interface to PyMC (Salvatier et al., 2016) to allow for easy model building and use of existing samplers. The performance of gradient-based samplers, such as HMC, critically depends on the precision of partial log-likelihood derivatives. Using PyMC's No-U-Turn sampler (NUTS) with our neural likelihood, we obtained results similar to those using Stan. If gradient-based sampling methods fail, we advise to use gradient-free sampling methods, such as slice sampling. For detailed information, see the **Appendix**.

4.4 MARKOVIAN COMPARTMENTAL MODEL

Setup This experiment demonstrates surrogate simulations of a complex non-exchangeable model of infectious diseases. The model features 34 parameters and thus represents a considerable extension of the two-parameter toy SIR model (Lueckmann et al., 2021; Radev et al., 2020). We use the model specification and posterior network from Radev et al. (2021). We implement the likelihood network as a recurrent cINN (see Section 2.3) to test its ability to emulate raw and noisy time series. Further, we train the posterior network using the MMD criterion (Eq. 9) with $\lambda = 1$ to judge the quality of the surrogate simulations numerically.

Results Upon convergence, we use the likelihood network to generate synthetic outbreak trajectories and compare them visually with the outputs of the original simulator. We observe good emulation across a variety of different parameter configurations, each leading to a qualitatively different simulated scenario (see Figure 2 for an example and the **Appendix** for detailed results). Moreover, it seems that the surrogate network is not only able to accurately approximate the median trajectory, but also the variability (i.e., *aleatoric uncertainty*) in simulated trajectories.

Beyond purely visual comparisons, we also compute the posterior and joint calibration of the two networks using (J)SBC on 1 000 held-out simulations. We confirm the good posterior calibration observed by (Radev et al., 2021). In addition, the joint calibration results help us highlight some subtle deficiencies of the likelihood network. For instance, it tends to overestimate the variability of simulated time series, thus "tricking" the posterior network into estimating higher values for the noise parameters (see **Appendix**). We attribute this deficiency to the extremely wide magnitude range of the simulated data (incidence in the order of millions) which is not captured by our simple input standardization procedure.

5 CONCLUSION

We investigated the utility of JANA for Bayesian surrogate modeling and simulation-based inference within the BayesFlow framework. We believe that JANA can greatly enrich applications of amortized Bayesian inference. Future work should investigate weight sharing schemes for the various network components and advance a framework-independent benchmark database for joint estimation of non-trivial scientific models.

References

Andrieu, C., Doucet, A., & Holenstein, R. (2010). Particle markov chain monte carlo methods. *Journal of the Royal Statistical Society: Series B*.

Ardizzone, L., Kruse, J., Wirkert, S., Rahner, D., Pellegrini, E. W., Klessen, R. S., Maier-Hein, L., Rother, C., & Köthe, U. (2019). Analyzing inverse problems with invertible neural networks. *Intl. Conf. on Learning Representations*.

Ardizzone, L., Lüth, C., Kruse, J., Rother, C., & Köthe, U. (2019). Guided image generation with conditional invertible neural networks. *arXiv preprint*.

Avecilla, G., Chuong, J. N., Li, F., Sherlock, G., Gresham, D., & Ram, Y. (2022). Neural networks enable efficient and accurate simulation-based inference of evolutionary parameters from adaptation dynamics. *PLoS Biology*, *20*(5), e3001633.

Beaumont, M. A., Cornuet, J.-M., Marin, J.-M., & Robert, C. P. (2009). Adaptive approximate Bayesian computation. *Biometrika*, *96*(4), 983–990.

Bellagente, M., Haußmann, M., Luchmann, M., & Plehn, T. (2022). Understanding event-generation networks via uncertainties. *SciPost Physics*, *13*(1), 003.

Bieringer, S., Butter, A., Heimel, T., Höche, S., Köthe, U., Plehn, T., & Radev, S. T. (2021). Measuring qcd splittings with invertible networks. *SciPost Physics*, *10*(6), 126.

Boelts, J., Lueckmann, J.-M., Gao, R., & Macke, J. H. (2022). Flexible and efficient simulation-based inference for models of decision-making. *Elife*.

Bürkner, P.-C., Gabry, J., & Vehtari, A. (2021). Efficient leave-one-out cross-validation for Bayesian non-factorized normal and Student-t models [Publisher: Springer]. *Computational Statistics*, *36*(2), 1243–1261.

Carpenter, B., Gelman, A., Hoffman, M. D., Lee, D., Goodrich, B., Betancourt, M., Brubaker, M., Guo, J., Li, P., & Riddell, A. (2017). Stan: A probabilistic programming language. *Journal of statistical software*, *76*(1).

Chen, Y., Zhang, D., Gutmann, M., Courville, A., & Zhu, Z. (2021). Neural approximate sufficient statistics for implicit models. *International Conference on Learning Representations*.

Cranmer, K., Brehmer, J., & Louppe, G. (2020). The frontier of simulation-based inference. *Proceedings of the National Academy of Sciences*.

Deistler, M., Goncalves, P. J., & Macke, J. H. (2022). Truncated proposals for scalable and hassle-free simulation-based inference. *arXiv preprint*.

Del Moral, P., Doucet, A., & Jasra, A. (2012). An adaptive sequential Monte Carlo method for approximate Bayesian computation. *Statistics and Computing*.

Dinh, L., Sohl-Dickstein, J., & Bengio, S. (2016). Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*.

Draxler, F., Schnörr, C., & Köthe, U. (2022). Whitening convergence rate of coupling-based normalizing flows. *arXiv preprint arXiv:2210.14032*.

Durkan, C., Murray, I., & Papamakarios, G. (2020). On contrastive learning for likelihood-free inference. *International Conference on Machine Learning*.

Fengler, A., Govindarajan, L. N., Chen, T., & Frank, M. J. (2021). Likelihood approximation networks (lans) for fast inference of simulation models in cognitive neuroscience. *Elife*, *10*, e65074.

Gelman, A., Vehtari, A., Simpson, D., et al. (2020). Bayesian workflow. *arXiv preprint*.

Gonçalves, P. J., Lueckmann, J.-M., Deistler, M., et al. (2020). Training deep neural density estimators to identify mechanistic models of neural dynamics. *Elife*.

Greenberg, D., Nonnenmacher, M., & Macke, J. (2019). Automatic posterior transformation for likelihood-free inference. *International Conference on Machine Learning*.

Gretton, A., Borgwardt, K., Rasch, M., Schölkopf, B., & Smola, A. (2012). A Kernel Two-Sample Test. *The Journal of Machine Learning Research*, *13*, 723–773.

Gronau, Q. F., Singmann, H., & Wagenmakers, E.-J. (2017). Bridgesampling: An r package for estimating normalizing constants. *arXiv preprint*.

Gutmann, M. U., & Corander, J. (2016). Bayesian optimization for likelihood-free inference of simulator-based statistical models. *Journal of Machine Learning Research*.

Hermans, J., Begy, V., & Louppe, G. (2020). Likelihood-free mcmc with amortized approximate ratio estimators. *International Conference on Machine Learning*, 4239–4248.

Hermans, J., Delaunoy, A., Rozet, F., Wehenkel, A., Begy, V., & Louppe, G. (2021). A trust crisis in simulation-based inference? Your posterior approximations can be unfaithful. *arXiv preprint*.

Ivanova, D. R., Foster, A., Kleinegesse, S., Gutmann, M. U., & Rainforth, T. (2021). Implicit deep adaptive design: Policy-based experimental design without likelihoods. *Advances in Neural Information Processing Systems*, *34*, 25785–25798.

Jiang, B., Wu, T.-y., Zheng, C., & Wong, W. H. (2017). Learning summary statistic for approximate bayesian computation via deep neural network. *Statistica Sinica*, 1595–1618.

Kingma, D. P., & Dhariwal, P. (2018). Glow: Generative flow with invertible 1x1 convolutions. *Advances in neural information processing systems*, *31*.

Lavin, A., Zenil, H., Paige, B., et al. (2021). Simulation intelligence: Towards a new generation of scientific methods. *arXiv preprint*.

Lueckmann, J.-M., Bassetto, G., Karaletsos, T., & Macke, J. H. (2019). Likelihood-free inference with emulator networks. *Symposium on Advances in Approximate Bayesian Inference*.

Lueckmann, J.-M., Boelts, J., Greenberg, D. S., Gonçalves, P. J., & Macke, J. H. (2021). Benchmarking simulation-based inference. *arXiv preprint*.

Lueckmann, J.-M., Gonçalves, P. J., Bassetto, G., Öcal, K., Nonnenmacher, M., & Macke, J. H. (2017). Flexible statistical inference for mechanistic models of neural dynamics. *31st NeurIPS Conference Proceedings*.

MacKay, D. (2003). *Information theory, inference and learning algorithms*. Cambridge University Press.

Marin, J.-M., Pudlo, P., Robert, C. P., & Ryder, R. J. (2012). Approximate Bayesian computational methods. *Statistics and Computing*.

Marin, J.-M., Pudlo, P., Estoup, A., & Robert, C. (2018). *Likelihood-free model choice*. Chapman; Hall/CRC.

Marjoram, P., Molitor, J., Plagnol, V., & Tavaré, S. (2003). Markov chain Monte Carlo without likelihoods. *Proceedings of the National Academy of Sciences*.

Meng, X.-L., & Wong, W. H. (1996). Simulating ratios of normalizing constants via a simple identity: A theoretical exploration. *Statistica Sinica*.

Pacchiardi, L., & Dutta, R. (2022). Score matched neural exponential families for likelihood-free inference. *J. Mach. Learn. Res.*, *23*, 38–1.

Papamakarios, G., & Murray, I. (2016). Fast ε-free inference of simulation models with bayesian conditional density estimation. *Advances in neural information processing systems*, *29*.

Papamakarios, G., Sterratt, D., & Murray, I. (2019). Sequential neural likelihood: Fast likelihood-free inference with autoregressive flows.

Picchini, U. (2014). Inference for SDE models via approximate bayesian computation. *Journal of Computational and Graphical Statistics*, *23*(4), 1080–1100.

Picchini, U., & Tamborrino, M. (2022). Guided sequential schemes for intractable Bayesian models. *arXiv preprint arXiv:2206.12235*.

Price, L. F., Drovandi, C. C., Lee, A., & Nott, D. J. (2018). Bayesian synthetic likelihood. *Journal of Computational and Graphical Statistics*.

Radev, S. T., Graw, F., Chen, S., Mutters, N. T., Eichel, V. M., Bärnighausen, T., & Köthe, U. (2021). Outbreakflow: Model-based bayesian inference of disease outbreak dynamics with invertible neural networks and its application to the covid-19 pandemics in germany. *PLoS computational biology*.

Radev, S. T., Mertens, U. K., Voss, A., Ardizzone, L., & Köthe, U. (2020). Bayesflow: Learning complex stochastic models with invertible neural networks. *IEEE transactions on neural networks and learning systems*.

Ratcliff, R., & McKoon, G. (2008). The diffusion decision model: Theory and data for two-choice decision tasks. *Neural Computation*, *20*(4), 873–922.

Säilynoja, T., Bürkner, P.-C., & Vehtari, A. (2022). Graphical test for discrete uniformity and its applications in goodness-of-fit evaluation and multiple sample comparison. *Statistics and Computing*, *32*(2), 1–21.

Salvatier, J., Wiecki, T. V., & Fonnesbeck, C. (2016). Probabilistic programming in Python using PyMC3. *PeerJ Computer Science*.

Schmitt, M., Bürkner, P.-C., Köthe, U., & Radev, S. T. (2022). Detecting model misspecification in amortized bayesian inference with neural networks. *arXiv preprint*.

Sisson, S. A., Fan, Y., & Beaumont, M. (2018). *Handbook of approximate Bayesian computation*. CRC Press.

Talts, S., Betancourt, M., Simpson, D., Vehtari, A., & Gelman, A. (2018). Validating bayesian inference algorithms with simulation-based calibration. *arXiv preprint*.

Toni, T. (2011). ABC SMC for parameter estimation and model selection with applications in systems biology. *Nature Precedings*.

Vehtari, A., Gelman, A., & Gabry, J. (2017). Practical bayesian model evaluation using leave-one-out cross-validation and waic. *Statistics and Computing*.

Vehtari, A., & Ojanen, J. (2012). A survey of Bayesian predictive methods for model assessment, selection and comparison. *Statistics Surveys*.

von Krause, M., Radev, S. T., & Voss, A. (2022). Mental speed is high until age 60 as revealed by analysis of over a million participants. *Nature Human Behaviour*, *6*(5), 700–708.

Wiqvist, S., Frellsen, J., & Picchini, U. (2021). Sequential neural posterior and likelihood approximation. *arXiv preprint*.

Wiqvist, S., Mattei, P.-A., Picchini, U., & Frellsen, J. (2019). Partially exchangeable networks and architectures for learning summary statistics in approximate bayesian computation. *International Conference on Machine Learning*.

Wood, S. N. (2010). Statistical inference for noisy nonlinear ecological dynamic systems. *Nature*.