EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

Faculty of natural sciences

Machine Learning in Science

Variational methods for simulation-based inference

University Tübingen
Faculty of natural sciences
Department of Computer Science
Machine Learning in Science
Manuel Glöckler, manuel.gloeckler@student.uni-tuebingen.de, 2021

Processing period: 25.05.21-25.11.21

First supervisor: Prof. Dr. Jakob Macke, University Tübingen
Second supervisor: Prof. Dr. Manfred Claassen, University Tübingen

Selbstständigkeitserklärung

Hiermit versichere ich, dass ich die vorliegende Masterarbeit selbständig und nur mit den angegebenen Hilfsmitteln angefertigt habe und dass alle Stellen, die dem Wortlaut oder dem Sinne nach anderen Werken entnommen sind, durch Angaben von Quellen als Entlehnung kenntlich gemacht worden sind. Diese Masterarbeit wurde in gleicher oder ähnlicher Form in keinem anderen Studiengang als Prüfungsleistung vorgelegt.

Manuel Glöckler (Immatrikulationsnummer: 4093712), November 25, 2021

Abstract *

We present Sequential Neural Variational Inference (SNVI), an approach to perform Bayesian inference in models with intractable likelihoods. SNVI combines likelihood-estimation (or likelihood-ratio-estimation) with variational inference to achieve a scalable simulation-based inference approach. SNVI maintains the flexibility of likelihood(-ratio) estimation to allow arbitrary proposals for simulations, while simultaneously providing a functional estimate of the posterior distribution without requiring MCMC sampling. We present several variants of SNVI and demonstrate that they are substantially more computationally efficient than previous algorithms, without loss of accuracy on benchmark tasks. We apply SNVI to a neuroscience model of the pyloric network in the crab and demonstrate that it can infer the posterior distribution with one order of magnitude fewer simulations than previously reported. SNVI vastly reduces the computational cost of simulation-based inference while maintaining accuracy and flexibility, making it possible to tackle problems that were previously inaccessible.

*Parts of this chapter are currently under review for publication and hence revised by Michael Deistler and Jakob Macke (see `https://openreview.net/forum?id=kZOUYdhqkNY`).

Zusammenfassung

Wir präsentieren “Sequential Neural Variational inference” (SNVI), ein Ansatz zur Durchführung von Bayesscher Inferenz in Modellen ohne analytischer Likelihood-Funktion. SNVI kombiniert Likelihood-Schätzung (oder Likelihood-Ratio-Schätzung) mit ‘variational inference’ um einen skalierbaren simulationsbasierte Inferenzstrategie zu erhalten. SNVI behält die Flexibilität der Likelihood(-Ratio)-Schätzung bei und kann daher den Großteil des Simulationsbudget in relevanten Bereichen platzieren. Gleichzeitig wird eine funktionale Schätzung der A-posteriori-Verteilung geliefert, ohne dass MCMC erforderlich ist. Wir stellen mehrere Varianten von SNVI vor und zeigen, dass sie wesentlich effizienter sind als frühere Algorithmen, ohne dabei an Genauigkeit bei üblichen Benchmark-Aufgaben zu verlieren. Wir wenden SNVI auf ein neurowissenschaftliches Modell des pylorischen Netzwerkes des Krebses an und zeigen, dass es die A-posteriori-Verteilung mit einer Größenordnung weniger Simulationen ableiten kann als bisher vergleichbare Methoden. SNVI reduziert die Rechenkosten der simulationsbasierten Inferenz erheblich, während die Genauigkeit und Flexibilität beibehalten wird. Das ermöglicht Inferenz Probleme anzugehen, die zuvor unzugänglich waren.

Acknowledgments

I would first like to thank my thesis supervisor Prof. Dr. Jakob Macke of the Machine Learning in Science group at the University of Tübingen, for giving me the opportunity to write this thesis and providing me with all the necessary facilities, guidance and revision for my research.

Special thanks to my thesis advisor Michael Deistler, for providing invaluable insights to the writing of this thesis. Thanks for the stimulating ideas for problems encountered during the writing of the thesis. Without his passionate participation, input and validation this work could not have been successfully conducted.

Contents

1. Introduction*

Many domains in science and engineering use numerical simulations to model empirically observed phenomena. These models are designed by domain experts and are built to produce mechanistic insights. However, in many cases, some parameters of the simulator cannot be experimentally measured and need to be inferred from data. A principled way to identify parameters that match empirical observations is Bayesian inference. However, for many models of interest, one can only *sample* from the model by simulating a (stochastic) computer program, but explicitly evaluating the likelihood $p(\boldsymbol{x}|\boldsymbol{\theta})$ is intractable. Traditional methods to perform Bayesian inference in such *simulation-based inference* (SBI), also known as *likelihood-free* inference scenarios, include Approximate Bayesian computation (ABC) (Beaumont et al., 2002a) and synthetic likelihood (SL) (Wood, 2010) methods. However, these methods generally struggle with high-dimensional data and typically require one to design or learn (Chen et al., 2021) summary statistics and distance functions.

Recently, several methods using neural density(-ratio) estimation have emerged. These methods train neural networks to learn the posterior (SNPE, Papamakarios and Murray, 2016; Lueckmann et al., 2017; Greenberg et al., 2019), the likelihood (SNLE, Papamakarios et al., 2019; Lueckmann et al., 2019a), or the likelihood-to-evidence ratio (SNRE, Thomas et al., 2021; Hermans et al., 2020; Durkan et al., 2020a; Miller et al., 2022).

To improve the simulation efficiency of these methods, sequential training schemes have been proposed: Initially, parameters are sampled from the prior distribution to train an estimation-network. Subsequently, new samples are drawn adaptively to focus training on specific regions in parameter space, thus allowing the methods to scale to larger models with more parameters.

In practice, however, it has remained a challenge to realize the full potential of these sequential schemes: For sequential neural posterior estimation (SNPE) techniques, the loss function needs to be adjusted across rounds (Greenberg et al., 2019), and it has been reported that this can be problematic if the proposal distribution is very different from prior, and lead to 'leakage' of probability mass into regions without prior support (Durkan et al., 2020a). Both sequential neural likelihood (SNLE) and likelihood-ratio (SNRE) methods require MCMC sampling, which can become prohibitively slow– MCMC sampling is required for each round of simulations, which, for high-dimensional models, can take more time than running the simulations and training the neural density estimator.

*Parts of this chapter are currently under blind open review for publication and hence revised by Michael Deistler and Jakob Macke (see `https://openreview.net/forum?id=kZOUYdhqkNY`).

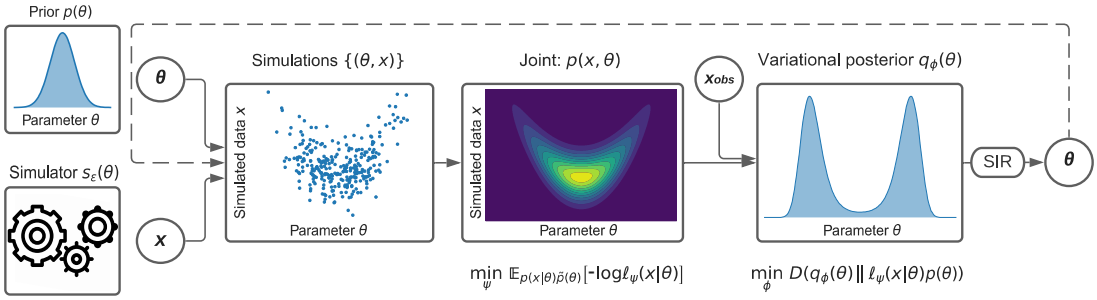

Figure 1.1.: Illustration of SNVI. We first learn the likelihood $p(\boldsymbol{x}|\boldsymbol{\theta})$ for any $\boldsymbol{\theta}$. We then use variational inference to learn the posterior distribution minimizing some divergence measure D. The obtained posterior distribution is sampled with sampling importance resampling (SIR) to run new simulations and refine the likelihood estimator.

Our goal is to provide a method that combines the advantages of posterior-targeting methods and those targeting likelihood(-ratios): Posterior targeting methods allow rapid inference by providing a functional approximation to the posterior which can be evaluated without the need to use MCMC sampling. Conversely, a key advantage of likelihood(-ratio) targeting methods is their flexibility– learned likelihoods can e.g. be used to integrate information from multiple observations, or can be used without retraining if the prior is changed. In addition, they can be applied with any active-learning scheme without requiring modifications of the loss-function.

We achieve this method by combining likelihood(-ratio) estimation with variationally learned inference networks using normalizing flows (Rezende and Mohamed, 2015; Papamakarios et al., 2017; Durkan et al., 2019a) and sampling importance resampling (SIR) (Rubin, 1988). We name our approach Sequential Neural Variational Inference (SNVI). We will show that our simulation-based inference methods are as accurate as SNLE and SNRE, while being substantially faster at inference as they do not require MCMC sampling. To deal with invalid outputs produced by simulators (e.g. infinities or NaNs) which cause problems for some density estimation-based approaches, we introduce a calibration kernel (Lueckmann et al., 2017) which can be used to exclude invalid data in SNVI, SNLE or SNRE.

A recent method termed "Sequential Neural Posterior and Likelihood Approximation" (SNPLA) also proposed to use variational inference (VI) instead of MCMC to speed up inference in likelihood-targeting methods (Wiqvist et al., 2021). While this proposal is related to our approach, their VI approach is based on the reverse Kullback Leibler (rKL) divergence for learning the posterior. As we also show on benchmark tasks, this leads to mode-seeking behavior which can limit its performance. In contrast, we show how this limitation can be overcome through modifying the variational objective in combination with using SIR for adjusting posteriors.

After an introduction on traditional and recent simulation-based inference (SBI) methods we introduce variational methods necessary for our work (Chapter 2). We then present our method, Sequential Neural Variational Inference (SNVI) (Chapter. 3). In Chapter 4.2, we empirically show that SNVI is significantly faster than state-of-the-art SBI methods while achieving similar accuracy on benchmark tasks. In Chapter 4.3, we demonstrate that SNVI is scalable and that it is robust to invalid simulation outputs: We obtain the posterior distribution of a complex neuroscience model with one order of magnitude fewer simulations than previous methods.

Our main contributions are as follows:

1. We introduce a general framework to use variational inference for obtaining the posterior distribution in simulation-based inference.
2. We develop several variational inference methods that are tailored to the multi-modality of posteriors often observed in simulation-based inference.
3. On several benchmark tasks, we show that our method performs on par with MCMC, but give significant speedups. Further, we demonstrate that this opens up new application fields inaccessible by competing methods.
4. We develop a new method to adjust for 'calibration' in likelihood-based methods. This method is especially relevant when simulators produce invalid data, which frequently occurs in scientific domains.

2. Background

Simulation-based inference (SBI) aims to perform Bayesian inference on statistical models for which the likelihood function is only implicitly defined through a stochastic simulator. Given a prior $p(\boldsymbol{\theta})$ and a simulator which implicitly defines the likelihood $p(\boldsymbol{x}|\boldsymbol{\theta})$, the goal is to identify the posterior distribution $p(\boldsymbol{\theta}|\boldsymbol{x}_o)$ for any observation $\boldsymbol{x}_o$.

If the likelihood function is available, then the posterior can be obtained by Bayes' rule:

$$p(\boldsymbol{\theta}|\boldsymbol{x}_o) = \frac{p(\boldsymbol{x}_o|\boldsymbol{\theta})p(\boldsymbol{\theta})}{Z} \text{ with } Z = \int p(\boldsymbol{x}_o|\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta}$$

Even though the evidence Z is typically intractable, we can sample from it with e.g. Markov-chain Monte Carlo (MCMC) or obtain tractable approximations using e.g. variational inference (VI) and related techniques (Bishop, 2006, Chapter 10 and 11). Yet if the likelihood is unavailable, these methods cannot be applied directly.

In this chapter we will first discuss why *simulation-based inference* can often be considered as *likelihood-free* (Sec. 2.1) and discuss both traditional and recent neural methods to perform *likelihood-free* inference (Sec. 2.2). Next, we present recent advances in variational inference which we will use to combine the best of both worlds, leading to our main contribution: "Sequential Neural Variational Inference" presented in Chapter 3.

2.1. Is simulation-based inference likelihood-free?

An inference task is deemed *likelihood-free* if the likelihood function cannot be derived in closed form or if the calculations are too expensive. To demonstrate that this is often the case for SBI, we will first formalize the implicit likelihood defined by a simulator.

An simulation-based inference problem considers the following generative model

$$\boldsymbol{\theta} \sim p(\boldsymbol{\theta}) \quad \text{and} \quad \boldsymbol{x} = s_{\boldsymbol{z}}(\boldsymbol{\theta}) \quad \text{with } \boldsymbol{z} \sim p(\boldsymbol{z}|\boldsymbol{\theta}).$$

Above we denote the 'simulator' with $s_{\boldsymbol{z}}$ and it's parameters $\boldsymbol{\theta} \in \Theta$. The parameters are drawn from a prior distribution $p(\boldsymbol{\theta})$. The simulator produce observable data $\boldsymbol{x} \in \mathcal{X}$ for a given parameter $\boldsymbol{\theta}$. In essence, a simulator can be anything, but let's assume it is a complex stochastic algorithm. These typically use a random latent state $\boldsymbol{z} = (z_1, \ldots, z_n)$ with joint distribution

$$p(\boldsymbol{z}|\boldsymbol{\theta}) = p(z_1; f_1(\boldsymbol{\theta}))p(z_2; f_2(z_1, \boldsymbol{\theta})) \cdots p(z_n; f_n(z_1, \ldots, z_{n-1}, \boldsymbol{\theta}))$$

i.e. the parameters of each latent state's distribution are sequentially determined by a mapping f_i using all previous variables as input. Given the latent state the data $\boldsymbol{x}$ is obtained through $\boldsymbol{x} \sim p(\boldsymbol{x}|\boldsymbol{z}, \boldsymbol{\theta})$. In practise, the corresponding densities are often not made explicit but arise from using a random-number generator within the simulator.

Assuming that all distributions admit densities, the joint density of latent variables $\boldsymbol{z}$ and observed data $\boldsymbol{x}$ is always tractable and given by $p(\boldsymbol{x}, \boldsymbol{z}|\boldsymbol{\theta}) = p(\boldsymbol{x}|\boldsymbol{z}, \boldsymbol{\theta})p(\boldsymbol{z}|\boldsymbol{\theta})$. The likelihood of $s_{\boldsymbol{z}}$ can thus be calculated by

$$p(\boldsymbol{x}|\boldsymbol{\theta}) = \int p(\boldsymbol{x}, \boldsymbol{z}|\boldsymbol{\theta}) d\boldsymbol{z} = \int p(\boldsymbol{x}|\boldsymbol{z}, \boldsymbol{\theta}) p(\boldsymbol{z}|\boldsymbol{\theta}) d\boldsymbol{z} = \mathbb{E}_{\boldsymbol{z} \sim p(\boldsymbol{z}|\boldsymbol{\theta})}[p(\boldsymbol{x}, \boldsymbol{z}|\boldsymbol{\theta})] \tag{2.1}$$

This potentially high dimensional integral is generally intractable. However, if we have complete knowledge about the latent state's distribution $p(\boldsymbol{z}|\boldsymbol{\theta})$ and $p(\boldsymbol{x}|\boldsymbol{z}, \boldsymbol{\theta})$, then several methods become applicable. As the joint likelihood $p(\boldsymbol{x}, \boldsymbol{z}|\boldsymbol{\theta}) = p(\boldsymbol{x}|\boldsymbol{z}, \boldsymbol{\theta})p(\boldsymbol{z}|\boldsymbol{\theta})$ is tractable we can perform inference on the joint space inferring $p(\boldsymbol{\theta}, \boldsymbol{z}|\boldsymbol{x}_o)$ using e.g. MCMC (see e.g. Beaumont (1999) for application in genetics). We then obtain posterior samples by marginalizing out $\boldsymbol{z}$ (i.e. drop $\boldsymbol{z}$). On the other hand, if the joint space is high dimensional this approach can become inefficient. Alternatively, Beaumont (2003) proposed to use an unbiased likelihood estimator, which can be obtained by Monte Carlo estimation of Equation 2.1. The algorithm was later generalized by Andrieu and Roberts (2009) and is known as pseudo-marginal MCMC. We mainly consider variational methods, which we further introduce in Sec. 2.3. Some variational Bayesian methods can also use unbiased estimates of the likelihood (Tran et al., 2016; He et al., 2021) or of the log likelihood's gradient (Gunawan et al., 2017). Yet, the availability of these estimators requires detailed knowledge about the latent states of the simulator.

Many domains of science have developed mechanistic simulator models. We focus on applications in computational neuroscience (see e.g. Gonçalves et al. (2020), Deistler et al. (2021), Schröder et al. (2019), West et al. (2021) or Oesterle et al. (2020)). Such mechanistic models directly describe the causal process generating the observed data. Hence these models are often based on biophysical equations using e.g. dynamical systems which naturally evolve from our understanding of the system. Yet, these simulators typically lag the mathematical abstraction as density models, making the above procedures not applicable. Biological neural networks are also often constructed using external libraries as e.g. NEURON (Hines and Carnevale, 2001). Hence we may not have access to the internal working of the simulator at all.

For this reason we consider the simulator as black-box, that is we have no knowledge about the internal workings (i.e $p(\boldsymbol{z}|\boldsymbol{\theta})$ and $p(\boldsymbol{x}|\boldsymbol{z}, \boldsymbol{\theta})$) but can collect samples $\boldsymbol{x} \sim p(\boldsymbol{x}|\boldsymbol{\theta})$ for any $\boldsymbol{\theta} \in \Theta$ by running the simulator. Such a general and flexible inference approach is of high value in scientific domains since the user only needs to run the simulator without having to abstract it into forms more amendable for inference.

Algorithm 1: Rejection ABC

1 **repeat**
2 | sample $\boldsymbol{\theta} \sim p(\boldsymbol{\theta})$;
3 | simulate $\boldsymbol{x} \sim p(\boldsymbol{x}|\boldsymbol{\theta})$
4 **until** $||\boldsymbol{x} - \boldsymbol{x}_o|| \leq \epsilon$;
5 **return** $\boldsymbol{\theta}$

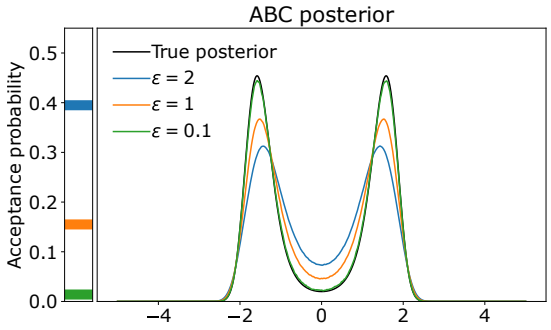

Figure 2.1.: Basic rejection ABC algorithm on the left side. One the right side the ABC posterior and corresponding acceptance probabilities for $\boldsymbol{\theta} \sim \mathcal{N}(\boldsymbol{\theta}; 0, 1)$, $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{x}; \boldsymbol{\theta}^2, 1)$ and $\boldsymbol{x}_0 = 2$.

2.2. Methods to circumvent the likelihood

We will introduce traditional methods applicable for SBI in Subsec. 2.2.1 and Subsec. 2.2.2. Then introduce recent neural network-based methods which are particularly relevant for our work in Subsec. 2.2.3.

2.2.1. Approximate Bayesian Computation (ABC)

The fundamental idea of Approximate Bayesian Computation goes back to Rubin (1984), by the observation that we obtain samples from any posterior distribution by a simple accept-reject method: Generate parameters $\boldsymbol{\theta}$ from the prior $p(\boldsymbol{\theta})$, simulate synthetic data $\boldsymbol{x} \sim p(\boldsymbol{x}|\boldsymbol{\theta})$ and accept only those samples which are equal to the observation $\boldsymbol{x}_o \in \mathcal{X}$. If $\mathcal{X}$ is at most countable it is straightforward to show that the outcome from this algorithm is an independent sample from the posterior:

$$p^{ABC}(\boldsymbol{\theta}|\boldsymbol{x}_o) \propto \sum_{\boldsymbol{x} \in \mathcal{X}} \mathbb{1}(\boldsymbol{x} = \boldsymbol{x}_o) p(\boldsymbol{x}|\boldsymbol{\theta}) p(\boldsymbol{\theta}) = P(\boldsymbol{x} = \boldsymbol{x}_o|\boldsymbol{\theta}) p(\boldsymbol{\theta}) \propto p(\boldsymbol{\theta}|\boldsymbol{x}_o)$$

However, if $\boldsymbol{x}$ is continuous the above procedure is ill-defined as the probability of an exact match between simulated and observed data is zero. Pritchard et al. (1999) extend the above algorithm to continuous spaces, by instead accepting any simulation that is 'almost' identical to the observed one. We can thus write a rejection ABC algorithm, as shown in Alg. 1.

This relaxation produce samples not from the posterior distribution but from an approximation, which can be written as

$$p_\epsilon^{ABC}(\boldsymbol{\theta}|\boldsymbol{x}_o) \propto \int k_\epsilon(\boldsymbol{x}_o - \boldsymbol{x}) p(\boldsymbol{x}|\boldsymbol{\theta}) d\boldsymbol{x} p(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x}|\boldsymbol{\theta})} \left[k_\epsilon(\boldsymbol{x}_o - \boldsymbol{x}) \right] p(\boldsymbol{\theta}) \tag{2.2}$$

with $k_\epsilon(\boldsymbol{x}_o - \boldsymbol{x}) \propto \mathbb{1}(||\boldsymbol{x}_o - \boldsymbol{x}|| \leq \epsilon)$. Although other choices of kernel k_ϵ are possible, by instead accepting samples with probability proportional to $k_\epsilon(\boldsymbol{x}_o - \boldsymbol{x})$, where k_ϵ is a well-chosen bounded kernel (i.e. an RBF kernel $k_\epsilon(\boldsymbol{x}_o - \boldsymbol{x}) = \exp(-\frac{1}{2\epsilon}||\boldsymbol{x}_o - \boldsymbol{x}||_2^2)$) (Wilkinson, 2013). The exact posterior can be thought as limiting case if the kernel k_ϵ satisfies $\lim_{\epsilon \to 0} k_\epsilon(\boldsymbol{x}_o - \boldsymbol{x}) \propto \delta(\boldsymbol{x}_o - \boldsymbol{x})^*$. Hence ϵ can be thought as parameter controlling a trade off between approximation quality and computational efficiency as shown in Fig. 2.1.

More efficient methods make use of MCMC and perturb previously accepted parameters, which can be more likely to be accepted (Marjoram et al., 2003; Meeds et al., 2015). Or use Sequential Monte Carlo samplers, which use importance sampling to sequentially sample from more accurate posterior by gradually decreasing ϵ (Sisson et al., 2007).

Yet, the fundamental problem of any ABC algorithm is that it suffers from the curse of dimensionality. The efficiency of Alg. 1 is determined by the probability of accepting an proposed $\boldsymbol{\theta}$. The acceptance probability for an appropriate kernel k_ϵ can be estimated by

$$a(\boldsymbol{\theta}) = \int k_\epsilon(\boldsymbol{x}_o - \boldsymbol{x}) p(\boldsymbol{x}|\boldsymbol{\theta}) d\boldsymbol{x} \approx p(\boldsymbol{x}_o|\boldsymbol{\theta}) \int k_\epsilon(\boldsymbol{x}) d\boldsymbol{x}$$

for small ϵ assuming $\sup_{\boldsymbol{x} \in \mathcal{X}} k_\epsilon(\boldsymbol{x}) = 1$ [†]. We can interpret the second term as an 'volume penalty' for different choices of kernels. If we choose $k_\epsilon(\boldsymbol{x}) = \mathbb{1}(||\boldsymbol{x}||_2 \leq \epsilon)$ and $\boldsymbol{x} \in \mathbb{R}^d$ then $\int k_\epsilon(\boldsymbol{x}) d\boldsymbol{x} = \Gamma(d/2+1)^{-1} (\sqrt{\pi}\epsilon)^d$, whereas d denotes the dimensionality of $\boldsymbol{x}$ and Γ the gamma function. Hence the acceptance probability for any ABC algorithm independent of prior and simulator approaches zero at rate $O(\epsilon^d)$ for $\epsilon < 1/\sqrt{\pi}$. As a consequence ABC approaches require a careful design of kernel k_ϵ and low dimensional (sufficient) summary statistics (Prangle, 2015; Pacchiardi et al., 2021).

2.2.2. Synthetic likelihood

ABC circumvents the likelihood to obtain approximate samples from the posterior distribution. In contrast, synthetic likelihood methods approximate the likelihood using repeated simulations for a fixed $\boldsymbol{\theta}$ (Wood, 2010; Price et al., 2018). The likelihood has an unknown analytical form, but in some cases, it may be reasonable to assume that it belongs to a certain parametric family. For example, if we observe an average of independent samples, the central limit theorem suggest that the likelihood may be Gaussian:

$$p(\boldsymbol{x}|\boldsymbol{\theta}) \approx \mathcal{N}(\boldsymbol{x}; \mu(\boldsymbol{\theta}), \Sigma(\boldsymbol{\theta}))$$

The functional form of the mean $\mu(\boldsymbol{\theta})$ and covariance $\Sigma(\boldsymbol{\theta})$ is unknown, but we can use the simulator to estimate them via Monte Carlo:

$$\mu(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x}|\boldsymbol{\theta})}[\boldsymbol{x}] \qquad \Sigma(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x}|\boldsymbol{\theta})}[(\boldsymbol{x} - \mu(\boldsymbol{\theta}))^T (\boldsymbol{x} - \mu(\boldsymbol{\theta}))]$$

*We use the Dirac delta function δ, a generalized function whose value is zero everywhere except at zero and satisfies $\int \delta(x) dx = 1$ and $\int f(x)\delta(x)dx = f(0)$.

†Consider $k_\epsilon(\boldsymbol{x}) = \mathbb{1}(||\boldsymbol{x} - \boldsymbol{x}_o|| \leq \epsilon)$, then $a(\boldsymbol{\theta}) = P(||\boldsymbol{x} - \boldsymbol{x}_o|| \leq \epsilon|\boldsymbol{\theta})$. The density can be approximated as the mass divided by the volume i.e. $p(\boldsymbol{x}_o|\boldsymbol{\theta}) \approx a(\boldsymbol{\theta})/|B_\epsilon(\boldsymbol{x}_o)|$ thus also $a(\boldsymbol{\theta}) \approx p(\boldsymbol{x}_o|\boldsymbol{\theta})|B_\epsilon(\boldsymbol{x}_o)|$, whereas $|B_\epsilon(\boldsymbol{x}_o)|$ denotes the volume of the corresponding ϵ ball implied by k_ϵ.

If the true likelihood is Gaussian then we obtain an unbiased likelihood estimate. Generalizations for the non-Gaussian case exist e.g. by using kernel density estimation instead (Gutmann et al., 2016). The constructed likelihood surrogate can then be used in MCMC or variational methods to obtain samples from the approximate posterior (Ong et al., 2017; Andrieu and Roberts, 2009).

Price et al. (2018) demonstrated that synthetic likelihoods can scale better to high dimensional problems and are easier to tune than ABC. Yet, each evaluation of the likelihood for different parameters $\boldsymbol{\theta}$ requires running the simulator to obtain samples. As most likelihood-based methods evaluate the likelihood many times, the number of required simulations can become costly. Therefore it may be more efficient to learn a functional approximation for $\mu(\boldsymbol{\theta})$ and $\Sigma(\boldsymbol{\theta})$ from simulations. In other words to perform conditional density estimation, leading to the next section.

2.2.3. Likelihood-free inference by density estimation

For a simulation-based model, we can easily produce samples from the joint distribution $p(\boldsymbol{x}, \boldsymbol{\theta})$ by producing parameters from a prior $\boldsymbol{\theta} \sim p(\boldsymbol{\theta})$ then simulate it using the simulator $\boldsymbol{x} \sim p(\boldsymbol{x}|\boldsymbol{\theta})$. Unfortunately, we cannot evaluate the density $p(\boldsymbol{x}, \boldsymbol{\theta}) = p(\boldsymbol{x}|\boldsymbol{\theta})p(\boldsymbol{\theta}) = p(\boldsymbol{\theta}|\boldsymbol{x})p(\boldsymbol{x})$, because we neither can evaluate the likelihood $p(\boldsymbol{x}|\boldsymbol{\theta})$ nor $p(\boldsymbol{\theta}|\boldsymbol{x})$. Yet, we can estimate the conditional densities $p(\boldsymbol{x}|\boldsymbol{\theta})$ (i.e. the likelihood) or $p(\boldsymbol{\theta}|\boldsymbol{x})$ (i.e. the amortized posterior) from pairs $(\boldsymbol{x}, \boldsymbol{\theta}) \sim p(\boldsymbol{x}, \boldsymbol{\theta})$.

Recent deep neural density estimators admit good performance also among high-dimensional problems (Papamakarios, 2019; Papamakarios et al., 2017; Durkan et al., 2019a). However, density estimation at it's core suffers from the curse of dimensionality (Papamakarios, 2019). In Fig. 2.2A we show a complex joint density. Conditional density estimation of $p(\boldsymbol{\theta}|\boldsymbol{x})$ corresponds to learn all horizontal slices. Estimating $p(\boldsymbol{x}|\boldsymbol{\theta})$ corresponds to learn all vertical slices. A global estimate is clearly challenging, and we may have to run many 'simulations' in order to do so. Yet, to identify the posterior distribution $p(\boldsymbol{\theta}|\boldsymbol{x}_o) \propto p(\boldsymbol{x}_o|\boldsymbol{\theta})p(\boldsymbol{\theta})$ we only require an estimate of the conditional densities at the observation $\boldsymbol{x}_o$. This allows to simplify the corresponding problem by instead proposing $(\boldsymbol{\theta}, \boldsymbol{x}) \sim \tilde{p}(\boldsymbol{x}, \boldsymbol{\theta})$ for which $\boldsymbol{x} \approx \boldsymbol{x}_o$. There are two major strategies to enforce this property:

Sequential methods: We can propose parameters $\boldsymbol{\theta}$ which likely produce simulation similar to the observation. A good choice would be $\tilde{p}(\boldsymbol{\theta}) = p(\boldsymbol{\theta}|\boldsymbol{x}_o)$, which strongly simplifies the corresponding density estimation problem as illustrated in Fig. 2.2B (it focus the estimation problem on relevant regions in parameter space). In practice, this is not possible as the posterior is unavailable, yet it motivates many methods particularly relevant for our work (Papamakarios and Murray, 2016; Lueckmann et al., 2017; Greenberg et al., 2019; Papamakarios et al., 2019; Wiqvist et al., 2021; Hermans et al., 2020; Durkan et al., 2020a). Instead of the true posterior, these methods use a posterior approximation, which is sequentially refined over multiple rounds. As shown in Fig. 2.2B methods that target the likelihood are unaffected by this modification (Papamakarios et al., 2019). Conversely, direct posterior estimation methods are affected, but several methods exist to

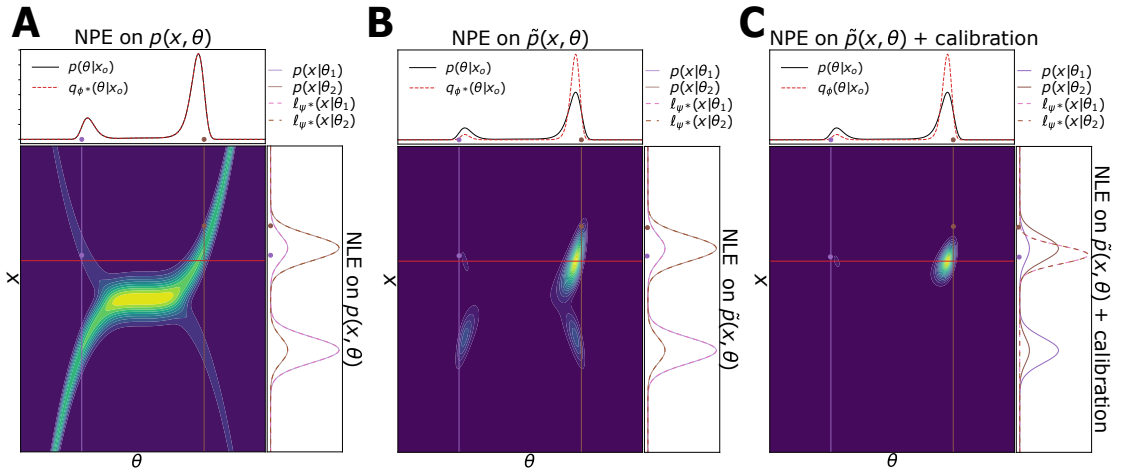

Figure 2.2.: The joint density is shown in the center. On the top the normalized posterior (i.e. the red horizontal slice). On the left the normalized likelihoods (i.e. vertical slices in brown and magenta).**A** shows it for $p(\boldsymbol{x}, \boldsymbol{\theta})$ where $\boldsymbol{\theta} \sim \mathcal{N}(0, 1)$ and $p(\boldsymbol{x}|\boldsymbol{\theta}) = 0.5(\mathcal{N}(\boldsymbol{x}; \boldsymbol{\theta}^3, I) + \mathcal{N}(\boldsymbol{x}; -\boldsymbol{\theta}^3, 1))$. **B** shows it for $\tilde{p}(\boldsymbol{x}, \boldsymbol{\theta}) = p(\boldsymbol{x}|\boldsymbol{\theta})\tilde{p}(\boldsymbol{\theta})$ with $\tilde{p}(\boldsymbol{\theta}) = p(\boldsymbol{\theta}|\boldsymbol{x}_o)$. **C** shows it for $\tilde{p}(\boldsymbol{x}, \boldsymbol{\theta}) = K(\boldsymbol{x}, \boldsymbol{x}_o)p(\boldsymbol{x}|\boldsymbol{\theta})\tilde{p}(\boldsymbol{\theta})$ with calibration kernel $K(\boldsymbol{x}, \boldsymbol{x}_o) = \exp(-1/2\epsilon||\boldsymbol{x} - \boldsymbol{x}_o||_2)$ for $\epsilon = 0.1$.

correct for this bias (Papamakarios and Murray, 2016; Lueckmann et al., 2017; Greenberg et al., 2019). Nonetheless, these corrections can come with problems e.g. high variance due to importance weights or density leakage (Durkan et al., 2020b; Lueckmann et al., 2017). This excludes the usage of SNPE for the certain relevant application, as we will demonstrate in Sec. 4.3. Lueckmann et al. (2021) demonstrated empirically that 'sequential' methods can strongly increase the simulation efficiency which is particularly relevant for computationally expensive simulators often encountered in scientific domains.

Calibration: Alternatively we can discard simulations $(\boldsymbol{\theta}, \boldsymbol{x})$ if $\boldsymbol{x}$ differs from the observation. This does not increase the simulation efficiency as it is applied after simulation but does simplify the density estimation problem as demonstrated in Fig. 2.2C (it focus the estimation problem on relevant regions in data space). Lueckmann et al. (2017) introduced this technique using a calibration kernel in SNPE and proofed that calibration does not affect direct posterior estimation. The loss is thereby weighted by a calibration kernel $K(\boldsymbol{x}, \boldsymbol{x}_o)$ which measures the similarity between $\boldsymbol{x}$ and $\boldsymbol{x}_o$ (e.g. similar to an ABC kernel). Additionally, calibration can solve the ubiquitous problem that scientific simulators, when fed with unrealistic parameters, often return invalid outputs. These invalid simulations are not useful to accurately learn the likelihood/posterior and we would like to exclude them e.g. by setting $K(\boldsymbol{x}, \boldsymbol{x}_o) = 0$ if $\boldsymbol{x}$ is invalid. Yet as visualized in Fig. 2.2C this does affect methods that estimate the likelihood. Current likelihood-based methods can thus not be applied to such problems and no correction strategy was developed yet. We will derive the exact bias-factor that emerges from calibration and include an efficient correction procedure within our method in Chapter 3.

Our method focus on improving likelihood-estimation (SNLE) and likelihood-ratio-estimation (SNRE) methods. SNLE trains a deep neural density estimator $\ell_\psi(\boldsymbol{x}|\boldsymbol{\theta})$ by minimizing the forward Kullback-Leibler divergence (fKL) between $\ell_\psi(\boldsymbol{x}|\boldsymbol{\theta})$ and $p(\boldsymbol{x}|\boldsymbol{\theta})$ using samples $(\boldsymbol{x}, \boldsymbol{\theta}) \sim \tilde{p}(\boldsymbol{x}, \boldsymbol{\theta}) = p(\boldsymbol{x}|\boldsymbol{\theta})\tilde{p}(\boldsymbol{\theta})$ from the simulator,

$$\mathcal{L}(\psi) = D_{\mathrm{KL}}(p(\boldsymbol{x}|\boldsymbol{\theta})\tilde{p}(\boldsymbol{\theta})||\ell_\psi(\boldsymbol{x}|\boldsymbol{\theta})\tilde{p}(\boldsymbol{\theta})) = -\frac{1}{N}\sum_{i=1}^{N}\log \ell_\psi(\boldsymbol{x}_i|\boldsymbol{\theta}_i) + \text{ const } .$$

Here, $\ell_\psi(\boldsymbol{x}|\boldsymbol{\theta})$ is a conditional density estimator learning the conditional density $p(\boldsymbol{x}|\boldsymbol{\theta})$ from $(\boldsymbol{\theta}, \boldsymbol{x})$ pairs, ψ are its learnable parameters, and $\tilde{p}(\boldsymbol{\theta})$ is the proposal distribution from which the parameters $\boldsymbol{\theta}$ are drawn e.g. a previous estimate of the posterior or by an active learning scheme (Papamakarios et al., 2019; Lueckmann et al., 2019a). Analogously, SNRE uses a discriminator, e.g. a deep logistic regression network, to estimate the density *ratio* (Hermans et al., 2020; Durkan et al., 2020a),

$$r(\boldsymbol{x}, \boldsymbol{\theta}) = \frac{\tilde{p}(\boldsymbol{x}, \boldsymbol{\theta})}{\tilde{p}(\boldsymbol{x})\tilde{p}(\boldsymbol{\theta})} = \frac{p(\boldsymbol{x}|\boldsymbol{\theta})}{\tilde{p}(\boldsymbol{x})}.$$

If the proposal is given by the prior, then one can recover the exact posterior density, otherwise the posterior can be recovered up to a normalizing constant (Durkan et al., 2020a). Once the likelihood (or likelihood-ratio) has been learned, the posterior can be sampled with MCMC. In sequential schemes, the proposal $\tilde{p}$ is updated each round using the current estimate of the posterior – thus, computationally expensive MCMC sampling needs to be run in each round*. Our method exactly aims to improve this bottleneck with variational methods as described in the next section.

2.3. Variational inference

Inference in probabilistic models if often infeasible, even if the likelihood is tractable. Variational inference formulates an optimization problem over a class of tractable distributions $\mathcal{Q}$ to find parameters ϕ^* such that $q_{\phi^*} \in \mathcal{Q}$ is closest to the true posterior $p(\boldsymbol{\theta}|\boldsymbol{x}_o)$ according to some divergence D (Blei et al., 2017). Formally,

$$\phi^* = \arg\min_\phi D(q_\phi(\boldsymbol{\theta})||p(\boldsymbol{\theta}|\boldsymbol{x}_o))$$

with $q_{\phi^*}(\boldsymbol{\theta}) = p(\boldsymbol{\theta}|\boldsymbol{x}_o) \iff D(q_{\phi^*}(\boldsymbol{\theta})||p(\boldsymbol{\theta}|\boldsymbol{x}_o)) = 0$. Recent work has introduced normalizing flows as a variational family for VI (Ranganath et al., 2014; Agrawal et al., 2020; Rezende and Mohamed, 2015). Normalizing flows define a distribution $q_\phi(\boldsymbol{\theta})$ by learning a bijection T_ϕ which transforms a simpler distribution q_0 into a complex distribution $p(\boldsymbol{\theta}|\boldsymbol{x}_o)$. Normalizing flows provide a highly flexible variational family, while at the same time allowing low variance gradient estimation of an expectation by the reparameterization

*Parts of the paragraph are currently under review for publication and hence revised by Michael Deistler and Jakob Macke (see `https://openreview.net/forum?id=kZ0UYdhqkNY`).

trick, i.e. $\nabla_\phi \mathbb{E}_{\boldsymbol{\theta}\sim q_\phi}[f(\boldsymbol{\theta})] = \mathbb{E}_{\boldsymbol{\theta}_0\sim q_0}[\nabla_\phi f(T_\phi(\boldsymbol{\theta}_0))]$ with $\boldsymbol{\theta} = T_\phi(\boldsymbol{\theta}_0)$ (Kingma and Welling, 2014a; Rezende et al., 2014; Rezende and Mohamed, 2015) *.

After running SNLE a likelihood surogate ℓ_ψ is available and thus the unnormalized posterior can be approximated $p(\boldsymbol{x}_o, \boldsymbol{\theta}) \approx \ell_\psi(\boldsymbol{x}_o|\boldsymbol{\theta})p(\boldsymbol{\theta})$. This renders Black-Box-Variational-Inference (BBVI) methods tractable (Ranganath et al., 2014; Agrawal et al., 2020). In this section, we will briefly introduce the main ideas used within our method. We introduce relevant divergence families in Subsec. 2.3.1, Monte Carlo refininments in Subsec. 2.3.2 and discuss improved gradient estimators in Subsec. 2.3.3.

2.3.1. Divergence measures

The choice of divergence measure D can have a major impact on the resulting variational approximation. We typically distinguish between mode-seeking and mass-covering divergences. A mode-seeking divergence emphasizes modeling the tails instead of the bulk of the distribution e.g. a Gaussian approximation of a mixture of Gaussians will try to fit the single component with the largest variance. In contrast, a mass-covering divergence will try to stretch the approximation across all components. Within this section, we will quickly introduce two divergence families particularly relevant for our work.

The Kullback-Leibler divergence

The Kullback-Leibler (KL) divergence (Kullback and Leibler, 1951) between $q_\phi(\boldsymbol{\theta})$ and $p(\boldsymbol{\theta}|\boldsymbol{x}_o)$ is given by

$$D_{KL}(q_\phi||p) = \int_{-\infty}^{\infty} q_\phi(\boldsymbol{\theta}) \log\left(\frac{q_\phi(\boldsymbol{\theta})}{p(\boldsymbol{\theta}|\boldsymbol{x}_o)}\right) d\boldsymbol{\theta} = \mathbb{E}_{\boldsymbol{\theta}\sim q_\phi}\left[\log\left(\frac{q_\phi(\boldsymbol{\theta})}{p(\boldsymbol{\theta}|\boldsymbol{x}_o)}\right)\right].$$

The divergence is not symmetric. We distinguish between the reverse KL (rKL) divergence $D_{KL}(q_\phi||p)$ and the forward KL divergence (fKL) $D_{KL}(p||q_\phi)$, as they inherit vastly different properties. Classically VI minimizes the reverse KL divergences, by maximizing the ELBO (Blei et al., 2017) a lower bound to the evidence $\log p(x)$:

$$\mathcal{L}_{rKL}(\phi) = \mathbb{E}_{\boldsymbol{\theta}\sim q_\phi}\left[\log p(\boldsymbol{x}_o, \boldsymbol{\theta}) - \log q_\phi(\boldsymbol{\theta})\right] = \log p(x) - D_{KL}(q_\phi||p)$$

This is convenient as the ELBO only involves a expectation with respect to q_ϕ, which is chosen to be tractable. In contrast the forward KL minimizes

$$\mathcal{L}_{fKL}(\phi) = \mathbb{E}_{\boldsymbol{\theta}\sim p}\left[\log p(\boldsymbol{\theta}|\boldsymbol{x}_o) - \log q_\phi(\boldsymbol{\theta})\right] = -\mathbb{E}_{\boldsymbol{\theta}\sim p}\left[\log q_\phi(\boldsymbol{\theta})\right] + \text{const.}$$

which involves a expectation with respect to the intractable posterior distribution. We discuss in Sec. 3.2 how we can obtain a efficient approximation to avoid this problem.

The reverse KL enforces $q_\phi(\boldsymbol{\theta}) = 0$, whenever $p(\boldsymbol{\theta}|\boldsymbol{x}_o) = 0$, leading to a mode seeking behaviour. In contrast the forward KL enforces $q_\phi(\boldsymbol{\theta}) > 0$, whenever $p(\boldsymbol{\theta}|\boldsymbol{x}_o) > 0$ inducing a mass-covering property. We demonstrate this behavior in Figure 2.3 A, B. Even if the

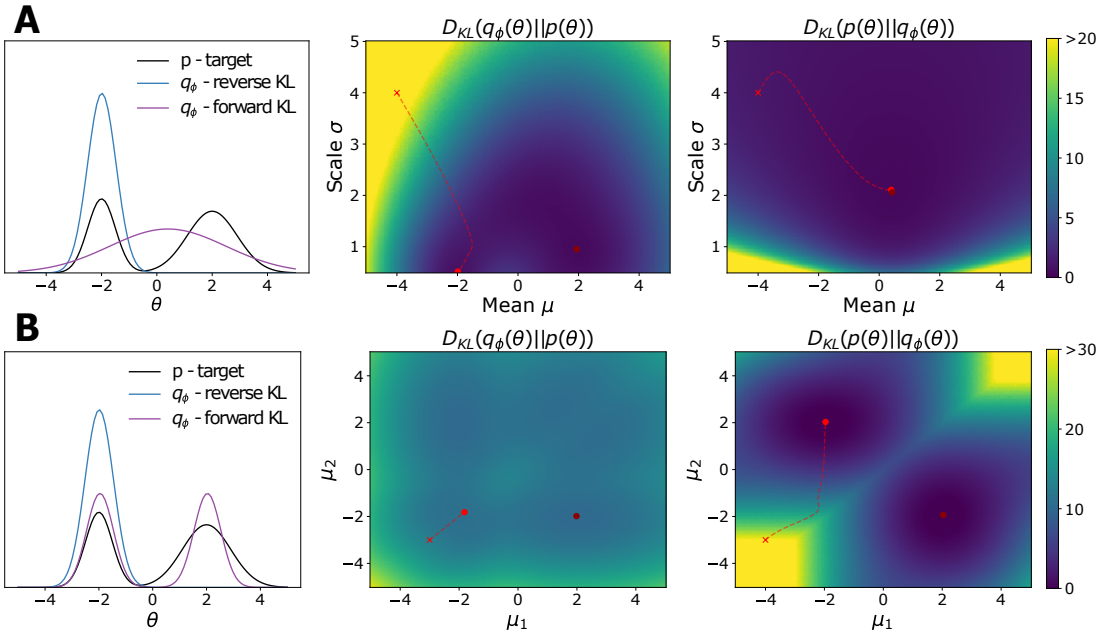

Figure 2.3.: Results of KL divergence minimization using stochastic gradient descent (SGD). The red line illustrates the SGD path, with the starting point marked by an $\times$. We show in **A** results using a Gaussian variational family with unknown mean μ and scale σ. In **B** we show results using a mixture of Gaussians with unknown means μ_1, μ_2 as variational family.

variational family can represent a bimodal target, a stochastic gradient descent algorithm that minimizes the reverse KL divergence will likely only cover one. As illustrated this is because every q_ϕ that occupies a mode of p represents a sharp local minimum, in which SGD can get 'stuck', hence initialization of the parameters heavily affect the result. On the contrary, the forward KL divergence shows broad global minimas and SGD can easily find the global minimum. Thus, we should prefer the forward KL when the posterior has multimodal structure and the variational family is sufficiently flexible.

Rényi's alpha divergence

There exists many α-divergence definitions, yet we focus on Rényi's definition (Rényi et al., 1961). For some $\alpha \in \{\alpha | \alpha > 0, \alpha \neq 1, |D_\alpha| < \infty\}$ and is given by

$$D_\alpha(q_\phi||p) = \frac{1}{\alpha - 1} \log \int q_\phi(\boldsymbol{\theta})^\alpha p(\boldsymbol{\theta}|\boldsymbol{x}_o)^{1-\alpha} d\boldsymbol{\theta}$$

which can be extended to $\alpha = 0, 1, +\infty$ by continuity.

This family contains several well-known divergences as a special case. To name some examples, for $\alpha = 0.5$ we obtain a function of the square Hellinger distance. For $\alpha = 2$ a

Algorithm 2: SIR (Sampling Importance Resampling)

Input: K the number of importance samples , proposal q_ϕ.
for $i \in [1, ..., K]$ **do**
  $\boldsymbol{\theta}_i \sim q_\phi(\boldsymbol{\theta})$
  $w_i = \frac{p(\boldsymbol{x}_o, \boldsymbol{\theta}_i)}{q_\phi(\boldsymbol{\theta}_i)}$
end
Each $\tilde{w}_i = w_i / \sum_{k=1}^{K} w_i$
$j \sim \text{Categorical}(\tilde{w})$
return $\boldsymbol{\theta}_j$

function proportional to the χ^2-divergence. And for $\alpha \to 1$ we retain the KL divergence (Li and Turner, 2016).

In contrast to the KL divergence the difference between $D_\alpha(p||q_\phi)$ and $D_\alpha(q_\phi||p)$ is less important as D_α is skew symmetric for any $\alpha \notin \{0, 1\}$ we have that $D_\alpha(p||q) = \frac{\alpha}{1-\alpha} D_{1-\alpha}(q||p)$. Hence a 'forward' divergence can be related to a 'reverse' divergence up to a multiplicative constant. Yet α-divergences allow to tune the mass-covering or mode-seeking behaviour based on the hyperparameter α (Li and Turner, 2016).

2.3.2. Monte Carlo objectives and sampling

Recent progress in deep latent variable models as the Variational Autoencoder (VAE) (Kingma and Welling, 2014b), involves producing tighter evidence lower bounds through Monte Carlo techniques. This is because VAEs use the bound as a surrogate for the evidence (i.e. the log marginal likelihood) to train the generative model. Hence a tighter bound is always better. Burda et al. (2016) introduced a simple way to tighten the bound by averaging over multiple samples resulting in an importance sampling estimate of the evidence i.e.

$$\log p(\boldsymbol{x}_o) \geq \mathcal{L}_{IW}^{(K)}(\phi) = E_{\boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_K} \left[\log \left(\frac{1}{K} \sum_{k=1}^{K} \frac{p(\boldsymbol{x}_o, \boldsymbol{\theta})}{q_\phi(\boldsymbol{\theta})} \right) \right] \geq \mathcal{L}_{rKL}(\phi)$$

As $K \to \infty$, also $\mathcal{L}_{IW}^{(K)}(\phi) \to \log p(\boldsymbol{x}_o)$ with $\mathcal{L}_{IW}^{(K+1)}(\phi) \geq \mathcal{L}_{IW}^{(K)}(\phi)$ (Burda et al., 2016). However, to drive q_ϕ towards the true posterior we are not necessarily interested in the tightness of the bound. In fact, tighter bounds can be harder to optimize. If $\mathcal{L}(\phi) \to \log p(\boldsymbol{x})$ then respectively also $\nabla_\phi \mathcal{L}(\phi) \to 0$ as the evidence is independent of ϕ, indicating a vanishing gradient (Rainforth et al., 2018). We will consider a potential solution for this problem in the next subsection.

Cremer et al. (2017) presents a reinterpretation, which motivates the usage in VI as well. Instead of driving q_ϕ toward p, maximizing the IW-ELBO can be interpreted to instead drive $q_{SIR}^{(K)}$ towards p. Thereby $q_{SIR}^{(K)}$ denotes a nonparametric density obtained by

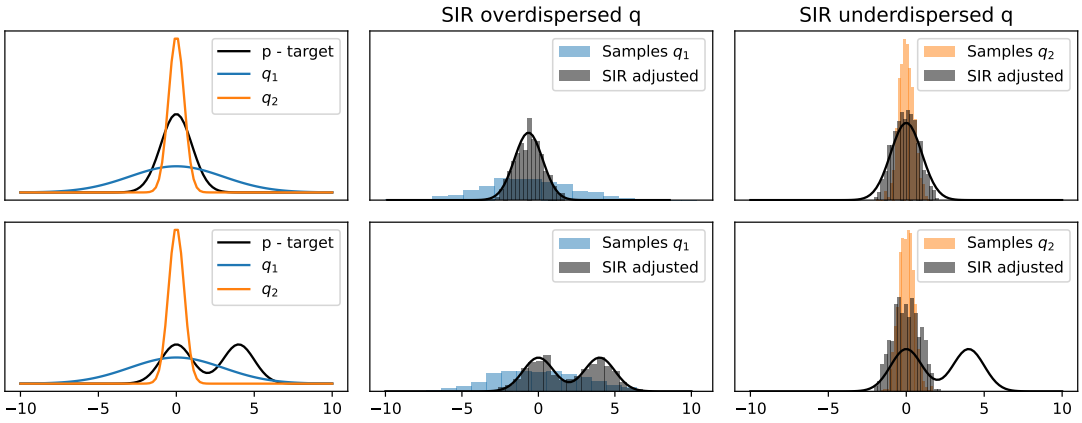

Figure 2.4.: SIR with either overdispersed or underdispersed proposals q and fixed number of proposals $K = 32$.

Sampling Importance Resampling (SIR) with q_ϕ as proposal (Rubin, 1988). We describe SIR in Alg. 2, it can be interpreted as a Monte Carlo technique to obtain samples from the posterior distribution as $K \to \infty$ independent of q_ϕ (Rubin, 1988; Cremer et al., 2017). Maximizing the IW-ELBO can thus be interpreted to directly optimize q_ϕ for parameters that will perform well with using SIR. Agrawal et al. (2020) demonstrated empirically that SIR after VI can generally improve performance and never hurts.

We demonstrate the capabilities of SIR in Figure 2.4. Indeed it can both enrich and refine variational approximations. Notably, the quality of SIR-adjusted samples strongly depends on the proposal. It is especially important that the proposal covers most of the mass of the target since too narrow proposals show only marginally improved results. As a result, SIR is expected to especially improve mass-covering variational approximations.

2.3.3. Improved gradient estimation

In black-box VI we estimate the gradient via Monte Carlo estimation. Yet the variance of gradient estimators can become too large to be useful. Particularly estimators with a small expected gradient require a proportionally smaller variance. We can measure this property by the signal-to-noise ratio (SNR), defined as the absolute expected gradient estimate divided by its standard deviation. A low SNR is always problematic as it indicates that the gradient estimate is dominated by noise.

Be $H(\phi) = -\mathbb{E}_{\boldsymbol{\theta}\sim q_\phi}[\log q_\phi(\boldsymbol{\theta})]$ the entropy of q_ϕ, then the gradient of the ELBO can be written as $\nabla_\phi \mathcal{L}_{rKL}(\phi) = \nabla_\phi \mathbb{E}_{\boldsymbol{\theta}\sim q_\phi}\left[p(\boldsymbol{x}_o, \boldsymbol{\theta})\right] + \nabla_\phi H(\phi)$ (similar for other objectives). Using the reparameterization trick $\boldsymbol{\theta} = T_\phi(\boldsymbol{\theta}_0)$, we can expand the total derivative of the entropy:

$$\begin{aligned}\nabla_\phi H(\phi) &= -\mathbb{E}_{\boldsymbol{\theta}_0\sim q_0}\left[\nabla_\phi \log q_\phi(T_\phi(\boldsymbol{\theta}_0))\right] \\ &= -\mathbb{E}_{\boldsymbol{\theta}_0\sim q_0}\left[\nabla_{\boldsymbol{\theta}} \log q_\phi(\boldsymbol{\theta}) \nabla_\phi T_\phi(\boldsymbol{\theta}_0)\right] - \mathbb{E}_{\boldsymbol{\theta}\sim q_\phi}\left[\nabla_\phi \log q_\phi(\boldsymbol{\theta})\right]\end{aligned}$$

The second term is known as the 'score function', which is zero in expectation. We can therefore remove it and still obtain an unbiased estimate. This can be achieved by holding ϕ constant under the gradient i.e. $\nabla_\phi H(\phi) = -\nabla_\phi \mathbb{E}_{\boldsymbol{\theta}\sim q_\phi}[\log q_\kappa(\boldsymbol{\theta})]$ with $\kappa = \phi$. The new "Sticking the Landing" (STL) estimator has the desirable property that as q_ϕ approaches p the variance goes to zero (Roeder et al., 2017). In contrast the score function component is not necessarily zero for any finite sample approximation, hence the full estimator has non-zero variance even if $q_\phi = p$.

A related estimator for unbiased gradient estimation on different Monte Carlo objectives was introduced by Tucker et al. (2018). In this situation, the STL can introduce some bias. But more prominently both estimators can stabilize the SNR on objectives that otherwise suffer from a vanishing SNR. In which case the STL estimator shows a slightly better performance (Roeder et al., 2017; Tucker et al., 2018; Rainforth et al., 2018). In general, both show slight improvements for VI also using normalizing flows as variational family (Agrawal et al., 2020).

3. Sequential Neural Variational Inference *

We introduce the general "Sequential Neural Variational inference" (SNVI) algorithm in Sec. 3.1. We then list variational objectives we found to excel on SBI problems in Sec. 3.2 and elaborate possible extensions to refine the approximate posterior estimate using Sampling Importance Resampling (SIR, Sec. 3.3). Last but not least, we introduce a method to use calibration within likelihood-based methods which is especially relevant if the simulator produces invalid data (Sec. 3.4).

3.1. Key ingredients

We propose a framework to use variational inference (VI) for simulation-based inference. Our method consists of three parts: A learnable likelihood (or likelihood-ratio) model, a posterior model (typically parameterized as a normalizing flow, we discuss similar expressive alternatives in Appendix Sec. A.8) to be learned with VI, and sampling importance resampling (SIR) (Rubin, 1988) to refine the accuracy of the posterior (Fig. 1.1). The likelihood(-ratio) model $\ell_\psi(\boldsymbol{x}|\boldsymbol{\theta})$ learns to approximate the likelihood $p(\boldsymbol{x}|\boldsymbol{\theta})$ or the likelihood-ratio $\frac{p(\boldsymbol{x}|\boldsymbol{\theta})}{p(\boldsymbol{x})}$ from pairs of parameters and simulation outputs $(\boldsymbol{\theta}, \boldsymbol{x})$. We use the term SNLVI to refer to SNVI with likelihoods, and SNRVI with likelihood-ratios. After a likelihood(-ratio) model has been trained, the posterior model $q_\phi(\boldsymbol{\theta})$ is trained with variational inference using normalizing flows. Finally, SIR is used to correct potential inaccuracies in the posterior $q_\phi(\boldsymbol{\theta})$ – as we will show below, the SIR step leads to empirical improvements at modest computational overhead.

To refine the likelihood(-ratio) model and the posterior, the procedure can be repeated across several 'rounds'. We opt to sample the parameters $\boldsymbol{\theta}$ from the previous posterior estimate $q_\phi(\boldsymbol{\theta})$, but other strategies for active learning (e.g. Lueckmann et al., 2019b) could be plugged into SNVI. The algorithm is summarized in Alg. 3. We will now describe three variational objectives that can be used with SNVI, the SIR procedure to refine the posterior, and a calibration kernel for dealing with invalid simulation outputs.

*Parts of this chapter are currently under review for publication and hence revised by Michael Deistler and Jakob Macke (see `https://openreview.net/forum?id=kZOUYdhqkNY`).

Algorithm 3: SNVI

Inputs: prior $p(\boldsymbol{\theta})$, observation $\boldsymbol{x}_o$, divergence D, simulations per round N, number of rounds R, selection strategy $\mathcal{S}$.
Outputs: Approximate likelihood ℓ_ψ and variational posterior q_ϕ.
Initialize: Proposal $\tilde{p}(\boldsymbol{\theta}) = p(\boldsymbol{\theta})$, simulation dataset $\mathcal{X} = \{\}$
for $r \in [1, ..., R]$ **do**
  for $i \in [1, ..., N]$ **do**
    $\boldsymbol{\theta}_i = \mathcal{S}(\tilde{p}, \ell_\phi, p)$; `// sample` $\boldsymbol{\theta}_i \sim \tilde{p}(\boldsymbol{\theta})$
    simulate $\boldsymbol{x}_i \sim p(\boldsymbol{x}|\boldsymbol{\theta}_i)$; `// run the simulator on` $\boldsymbol{\theta}_i$
    add $(\boldsymbol{\theta}_i, \boldsymbol{x}_i)$ to $\mathcal{X}$
  end
  (re-)train ℓ_ψ; $\psi^* = \arg\min_\psi -\frac{1}{N} \sum_{(\boldsymbol{x}_i, \boldsymbol{\theta}_i) \in \mathcal{X}} \log \ell_\psi(\boldsymbol{x}_i|\boldsymbol{\theta}_i)$; `// or SNRE loss`
  (re-)train q_ϕ; $\phi^* = \arg\min_\phi D(q_\phi(\boldsymbol{\theta}) || p(\boldsymbol{\theta}|\boldsymbol{x}_o))$ with

$$p(\boldsymbol{\theta}|\boldsymbol{x}_o) \propto p(\boldsymbol{x}_o|\boldsymbol{\theta}) p(\boldsymbol{\theta}) \approx \ell_{\psi^*}(\boldsymbol{x}_o|\boldsymbol{\theta}) p(\boldsymbol{\theta})$$

  $\tilde{p}(\boldsymbol{\theta}) = q_\phi(\boldsymbol{\theta})$
end

3.2. Variational objectives for SBI

Because of the expressiveness of normalizing flows, the true posterior can likely be approximated well by a member of the variational family (Papamakarios et al., 2021). Thus, the quality of the variational approximation is strongly linked to the ability to achieve the best possible approximation through optimization, which in turn depends on the choice of variational objective (or divergence) D. Using the reverse Kullback-Leibler Divergence (rKL) as proposed by Wiqvist et al. (2021) can give rise to mode-seeking behavior and q_ϕ might not cover all regions of the posterior (Bishop, 2006; Blei et al., 2017).

Therefore, some regions in parameter space might be left and consequently, the likelihood within these regions will be poorly approximated as well. As a complementary approach, we suggest and evaluate three alternative variational objectives that induce a *mass-covering* behavior and posit that this strategy will be particularly important in sequential schemes.

1. Forward KL divergence (fKL) In contrast to the reverse KL (rKL), the forward Kullback-Leibler divergence (fKL) is mass-covering (Bishop, 2006). Wan et al. (2020) minimize the following upper bound to the evidence, which implicitly minimizes the fKL: $\mathcal{L}(\phi) = \mathbb{E}_{\boldsymbol{\theta} \sim q_\phi} [w(\boldsymbol{\theta}) \log (w(\boldsymbol{\theta}))]$ with $w(\boldsymbol{\theta}) = p(\boldsymbol{x}_o, \boldsymbol{\theta}) / q_\phi(\boldsymbol{\theta})$. This expression is hard to estimate with samples: If $q_\phi(\boldsymbol{\theta})$ is different from $p(\boldsymbol{x}_o, \boldsymbol{\theta})$ then $w(\boldsymbol{\theta}) \approx 0$ for most $\boldsymbol{\theta} \sim q_\phi(\boldsymbol{\theta})$, thus $\nabla_\phi \mathcal{L}(\phi) \approx 0$, which would prevent learning (see Appendix Sec. A.1).

To overcome this problem, we rewrite the fKL using self-normalized importance sampling (Jerfel et al., 2021). Let $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_N \sim \pi$ be samples from an arbitrary proposal distribution π. We then minimize the loss:

$$\mathcal{L}_{\text{fKL}}(\phi) = D_{KL}(p||q_\phi) \approx \sum_{i=1}^{N} \frac{w(\boldsymbol{\theta}_i)}{\sum_{j=1}^{N} w(\boldsymbol{\theta}_j)} \log\left(\frac{p(\boldsymbol{x}_o, \boldsymbol{\theta}_i)}{q_\phi(\boldsymbol{\theta}_i)}\right)$$

where $w(\boldsymbol{\theta}) = p(\boldsymbol{x}_o, \boldsymbol{\theta})/\pi(\boldsymbol{\theta})$. The corresponding gradient estimator can be written as

$$\nabla_\phi \mathcal{L}_{\text{fKL}}(\phi) = -\mathbb{E}_{\boldsymbol{\theta}\sim p}\left[\nabla_\phi \log\left(q_\phi(\boldsymbol{\theta})\right)\right] \approx -\sum_{i=1}^{N} \frac{w(\boldsymbol{\theta}_i)}{\sum_{j=1}^{N} w(\boldsymbol{\theta}_j)} \nabla_\phi \log q_\phi(\boldsymbol{\theta}_i)$$

As a self-normalized importance sampling scheme, this estimate is biased, but the bias vanishes at rate $\mathcal{O}(1/N)$ (Hesterberg, 2003). In our experiments, we use $\pi = q_\phi$, which provides a good proposal when q_ϕ is close to p (Chatterjee and Diaconis, 2018). Even though q_ϕ will differ from p initially, sufficient gradient information is available to drive q_ϕ towards p, as we demonstrate in Appendix Sec. A.1.

2. Importance weighted ELBO The importance weighted ELBO (IW-ELBO) introduced by Burda et al. (2016) uses the importance-weighted gradient of the evidence lower bound (ELBO). It minimizes the KL divergence between the self-normalized importance sampling distribution of q_ϕ and the posterior and thus provides a good proposal for sampling importance resampling (Cremer et al., 2017; Domke and Sheldon, 2018; Ranganath et al., 2014). It can be formulated as

$$\mathcal{L}_{IW}^{(K)}(\phi) = \mathbb{E}_{\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_k\sim q_\phi}\left[\log \frac{1}{K}\sum_{k=1}^{K} \frac{p(\boldsymbol{x}_o, \boldsymbol{\theta}_k)}{q_\phi(\boldsymbol{\theta}_k)}\right].$$

Using the reparameterization trick an unbiased gradient estimator is given by

$$\nabla_\phi \mathcal{L}_{IW}^{(K)}(\phi) = \mathbb{E}_{\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_K\sim q_\phi}\left[\sum_{k=1}^{K} \tilde{w}(\boldsymbol{\theta}_k)\nabla_\phi \log\left(\frac{p(\boldsymbol{x}_o, \boldsymbol{\theta}_k)}{q_\phi(\boldsymbol{\theta}_k)}\right)\right] \quad \tilde{w}(\boldsymbol{\theta}_k) = \frac{w(\boldsymbol{\theta}_k)}{\sum_{j=1}^{K} w(\boldsymbol{\theta}_j)}$$

To avoid a low SNR of the gradient estimator (Rainforth et al., 2018), we use the 'Sticking the Landing' (STL) estimator introduced by Roeder et al. (2017).

3. Rényi α-divergences Rényi α-divergences are a divergence family with a hyperparameter α which allows to tune the mass-covering (or mode-seeking) behaviour of the algorithm. For $\alpha \to 1$, the divergence approaches the rKL. For $\alpha < 1$, the divergence becomes more mass-covering, for $\alpha > 1$ more mode-seeking. We use $\alpha = 0.1$ in our experiments. A Rényi variational bound was established by Li and Turner (2016) and is given by

$$\mathcal{L}_\alpha(\phi) = \frac{1}{1-\alpha} \log\left(\mathbb{E}_{\boldsymbol{\theta}\sim q_\phi}\left[\left(\frac{p(\boldsymbol{x}_o, \boldsymbol{\theta})}{q_\phi(\boldsymbol{\theta})}\right)^{1-\alpha}\right]\right).$$

If $|\mathcal{L}_{(\alpha)}| < \infty$ any naive Monte Carlo estimator with $K < \infty$ samples will be biased. Yet the bias is well-behaved in the sense that for $\alpha \leq 1$

$$\mathcal{L}_{rKL} = \mathbb{E}_{\{\theta_i\}_{i=1}^1}[\mathcal{L}_\alpha^{(1)}] \leq \cdots \leq \mathbb{E}_{\{\theta_i\}_{i=1}^K}[\mathcal{L}_\alpha^{(K)}] \to \mathcal{L}_\alpha \text{ as } K \to \infty \text{ almost surely,}$$

where $\mathcal{L}_{rKL}$ is the ELBO. Consequently the bound is biased towards the reveres KL-divergence, but the approximation quality can be improved using more samples (Li and Turner, 2016). By the reparameterization trick a biased gradient estimate is given by

$$\nabla_\phi \mathcal{L}_\alpha(\phi) = \mathbb{E}_{\boldsymbol{\theta}\sim q_\phi}\left[\tilde{w}_\alpha(\boldsymbol{\theta})\nabla_\phi \log\left(\frac{p(\boldsymbol{x}_o, \boldsymbol{\theta})}{q_\phi(\boldsymbol{\theta})}\right)\right] \qquad \tilde{w}_\alpha = \frac{w(\boldsymbol{\theta})^{1-\alpha}}{\mathbb{E}_{\boldsymbol{\theta}\sim q_\phi}\left[w(\boldsymbol{\theta})^{1-\alpha}\right]}.$$

For $\alpha = 0$, $\mathcal{L}_\alpha$ is a single sample Monte Carlo estimate of the IW-ELBO (when using K samples to estimate the expectation in $\mathcal{L}_\alpha(\phi)$) and thus also suffers from a low SNR as $\alpha \to 0$ (Rainforth et al., 2018; Li and Turner, 2016). Just as for the IW-ELBO, we alleviate this issue by combining the α-divergences with the STL estimator.

3.3. Sampling Importance Resampling

After the variational posterior has been trained, q_ϕ approximates $\ell_\psi(\boldsymbol{x}_o|\boldsymbol{\theta})p(\boldsymbol{\theta})/Z$ with normalization constant Z. We propose to improve the quality of posterior samples by applying Sampling Importance Resampling (SIR) (Rubin, 1988). We sample $K = 32$ samples from $\boldsymbol{\theta} \sim q_\phi(\boldsymbol{\theta})$, compute the corresponding importance weights $w_i = \ell_\psi(\boldsymbol{x}_o|\boldsymbol{\theta}_i)p(\boldsymbol{\theta}_i)/q_\phi(\boldsymbol{\theta}_i)$ and resample a single sample from a categorical distribution whose probabilities equal the normalized importance weights. The algorithm is detailed in Alg. 2.

This strategy enriches the variational family with minimal computational cost (Agrawal et al., 2020). SIR is particularly useful when $q_\phi(\boldsymbol{\theta})$ covers the true posterior and is thus well-suited for the objectives described above. See Fig. A.2 which demonstrates that SIR can reliably refine the posterior approximation, even using expressive normalizing flows.

3.4. Calibration kernel for dealing with invalid data

Simulators may produce unreasonable or undefined values (Lueckmann et al., 2017), as we will also see in the pyloric network model described later. To deal with the resulting 'missing' simulations, we introduce a calibration kernel $K(\boldsymbol{x}, \boldsymbol{x}_o)$ (Lueckmann et al., 2017) for loss-reweighing of the likelihood(-ratio) model. When using SNLVI, we use the loss

$$\mathcal{L}(\psi) = -\mathbb{E}_{\boldsymbol{\theta},\boldsymbol{x}\sim\tilde{p}(\boldsymbol{\theta},\boldsymbol{x})}[K(\boldsymbol{x}, \boldsymbol{x}_o)\log(\ell_\psi(\boldsymbol{x}|\boldsymbol{\theta}))],$$

where $K(\boldsymbol{x}, \boldsymbol{x}_o)$ is a kernel which allows the likelihood-model to focus on a specific region in data-space. However, unlike for posterior estimation methods (Lueckmann et al., 2017), this kernel biases the estimated likelihood (Appendix Sec. A.4). Indeed we

obtain $\ell_{\psi^*}(\boldsymbol{x}|\boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} K(\boldsymbol{x}, \boldsymbol{x}_o) p(\boldsymbol{x}|\boldsymbol{\theta})$ with $Z(\boldsymbol{\theta}) = \mathbb{E}_{p(\boldsymbol{x}|\boldsymbol{\theta})}[K(\boldsymbol{x}, \boldsymbol{x}_o)]$ as minimizer (see Appendix Lemma A.3). As a result the implied posterior is biased by a factor of $Z(\boldsymbol{\theta})^{-1}$.

To exclude invalid simulations we use the kernel $K(\boldsymbol{x}, \boldsymbol{x}_o) = \mathbb{1}(\boldsymbol{x} \text{ is valid})$, hence $Z(\boldsymbol{\theta}) = P(\boldsymbol{x} \text{ is valid}|\boldsymbol{\theta})$. Without any correction the learned likelihood-function will be biased towards parameter regions which often produce 'invalid' simulations. This prohibits any method that estimates likelihood(-ratio) (i.e. SNVI, SNLE, SNRE) from excluding 'invalid' simulations, and thus prohibit their use on simulators which produce such data. Also other choices of kernels can be beneficial i.e. by using an ABC kernel k_ϵ (see Subsec. 2.2.1) we can focus on especially relevant regions in data-space. We discuss this relation to ABC methods in Sec. A.5.

We overcome this bias by estimating $Z(\boldsymbol{\theta})$ with a feed-forward neural network $c_\zeta(\boldsymbol{\theta})$, allowing us to recover the posterior distribution $p(\boldsymbol{\theta}|\boldsymbol{x}_o)$ up to proportionality. After $\ell_\psi(\boldsymbol{x}|\boldsymbol{\theta})$ and $c_\zeta(\boldsymbol{\theta})$ are trained, we sample from the (unnormalized) posterior distribution

$$\mathcal{P}(\boldsymbol{\theta}) = \ell_{\psi^*}(\boldsymbol{x}_o|\boldsymbol{\theta}) p(\boldsymbol{\theta}) c_{\zeta^*}(\boldsymbol{\theta}) \propto p(\boldsymbol{x}_o|\boldsymbol{\theta}) p(\boldsymbol{\theta}) \propto p(\boldsymbol{\theta}|\boldsymbol{x}_o)$$

with VI in combination with SIR (details, proof and extension to SNRVI in Appendix Sec. A.4). We show the SNVI algorithm with calibration in Alg. 4.

4. Results *

We demonstrate the accuracy and the computational efficiency of SNVI on several examples. First, we apply SNVI to an illustrative example to demonstrate its ability to capture complex posteriors without mode-collapse (Sec. 4.1). Second, we compare SNVI to alternative methods on several benchmark tasks (Sec. 4.2). Third, we demonstrate in Sec. 4.3 that SNVI can obtain the posterior distribution in models with many parameters by applying it to a neuroscience model of the pyloric network in the crab *Cancer borealis*.

4.1. Illustrative example: Two moons

We use the 'two moons' simulator (Greenberg et al., 2019) to illustrate the ability of SNVI to capture complex posterior distributions. The two moons simulator has two parameters with a uniform prior and generates a posterior that has both local and global structure. Fig. 4.1A shows the ground truth posterior distribution as well as approximations learned by several methods using 10^5 simulations.

SNLE with MCMC (in the form of Slice Sampling with axis-aligned updates (Neal, 2003)) can recover the bimodality when running 100 chains in parallel (Lueckmann et al., 2021) (not shown: individual chains typically only explore a single mode). SNPLA, which is based on the mode-seeking rKL (and could thus also be considered as SNVI+rKL, see Appendix Sec. A.6) captures only a single mode. In contrast, SNLVI (using the fKL and SIR, denoted as SNVI+fKL) recovers both the local and the global structure of the posterior accurately. In terms of runtime, SNPLA and SNVI+fKL are up to twenty times faster than 100 chain MCMC in our implementation (Fig. 4.1B), and two to four orders of magnitude faster than single chain MCMC (single chain not shown, the relative speed-up for multi-chain MCMC is due to vectorization).

4.2. Results on benchmark problems

We compare the accuracy and computational cost of SNVI to that of previous methods, using SBI benchmark tasks (Lueckmann et al., 2021):

Bernoulli GLM: Generalized linear model with Bernoulli observations. Inference is performed on 10-dimensional sufficient summary statistics of the originally 100 dimensional raw data. The resulting posterior is 10-dimensional, unimodal, and concave.

*Parts of this chapter are currently under review for publication and hence revised by Michael Deistler and Jakob Macke (see `https://openreview.net/forum?id=kZOUYdhqkNY`).

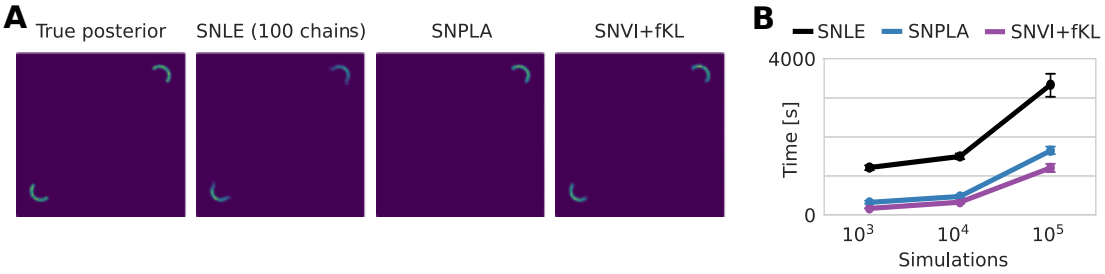

Figure 4.1.: **A** Posterior approximations of SNLE, SNPLA, and SNVI+fKL for the two moons benchmark example. **B** Runtime of all algorithms.

Lotka Volterra: A traditional model in ecology (Wangersky, 1978), which describes a predator-prey interaction between species, illustrating a task with complex likelihood and unimodal posterior.

Two moons: Same as described in the previous section.

SLCP: A task introduced by Papamakarios et al. (2019) with a simple likelihood and complex posterior. The prior is uniform, the likelihood has Gaussian noise but is nonlinearly related to the parameters, resulting in a posterior with four symmetrical modes.

For each task, we perform inference for ten different runs, each with a different observation. As performance metric, we used classifier 2-sample tests (C2ST) (best is 0.5, worst is 1.0) (Friedman, 2004; Lopez-Paz and Oquab, 2017). For each method, we perform inference given a total of 10^3, 10^4 and 10^5 simulations, evenly distributed across ten rounds of simulation and training. Details on the hyperparameters are provided in Appendix Sec. A.7, details on results in Appendix Fig. A.7.

We show results for two reference methods, SNLE with MCMC sampling, and SNPLA, and compare them to three variants of SNVI using the forward KL (SNVI+fKL), the importance-weighted ELBO (SNVI+IW) as well as an alpha-divergence (SNVI+α). We find that all three SNVI-variants achieve performance comparable to MCMC across all four tasks (Fig. 4.2 A-D, left), and outperform SNPLA on the two tasks with multi-modal posteriors (Two moons and SLCP). We find that omitting the SIR-adjustment (dotted lines) leads to a small but consistent degradation in inference performance for all SNVI-variants, but not for SNPLA with the rKL: When using the rKL, the approximate posterior q_ϕ is generally narrower than the posterior and thus ill-suited for SIR. See Appendix Fig. A.2 for demonstration on an 'two moons' posterior. Qualitatively similar results were found when using likelihood-ratio approaches with the same hyperparameters, see Appendix Fig. A.5. We compare alternative estimation methods of the proposed objectives in Appendix Fig. A.6, which demonstrates that self-normalization or the STL estimator is necessary for the proposed objectives to achieve the presented accuracy.

In terms of runtime, all three variants of SNLVI are substantially faster than SNLE on every task (bars in Fig. 4.2 on the right), in some cases by more than an order of magnitude. When using likelihood-ratio estimation, MCMC with 100 chains can be as

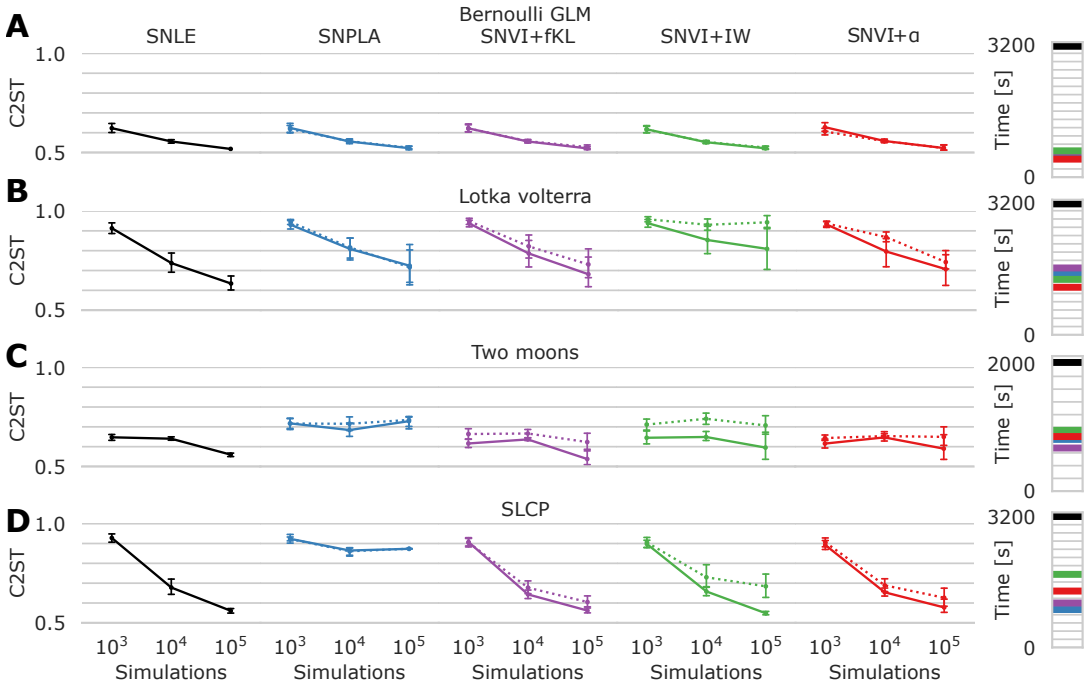

Figure 4.2.: C2ST benchmark results for SNVI for four models, Bernoulli GLM (**A**), Lotka volterra (**B**), Two moons (**C**) and SLCP (**D**). Each point represents the average metric value for ten different observations, as well as the confidence intervals. Bars on the right indicate the average runtime. Two reference methods: SNLE with MCMC sampling, and SNPLA (which uses rKL), as well as three variants of SNVI, with forward KL (SNVI+fKL), importance-weighted ELBO (SNVI+IW) and α-divergence (SNVI+α). Dotted lines: Performance when not using SIR.

fast as SNRVI on tasks with few parameters (Appendix Fig. A.5). On tasks with many parameters, however, SNRVI is significantly faster than SNRE (see e.g. Bernoulli GLM with 10 parameters). This can be attributed to the fast forward pass in SNR (classifier vs. normalizing flow) which majorly benefits MCMC.

We discuss results obtained for mixture distributions as alternative variational family in Appendix Fig. A.9. In general this family can be computationally more efficient when using Mixture of Gaussians in combination with the fKL estimator. Yet the limited expressiveness does affect the performance. Aside from that reparameterization is complicated in mixture distributions, making SNPLA, SNVI+IW and SNVI+α inefficient. See Appendix Sec. A.8 for more details.

The benchmark problems do not produce invalid data, so we did not benchmark calibration in this case. Nonetheless, we still can calibrate the likelihood using an ABC kernel, see Appendix Fig. A.3 for a proof of concept.

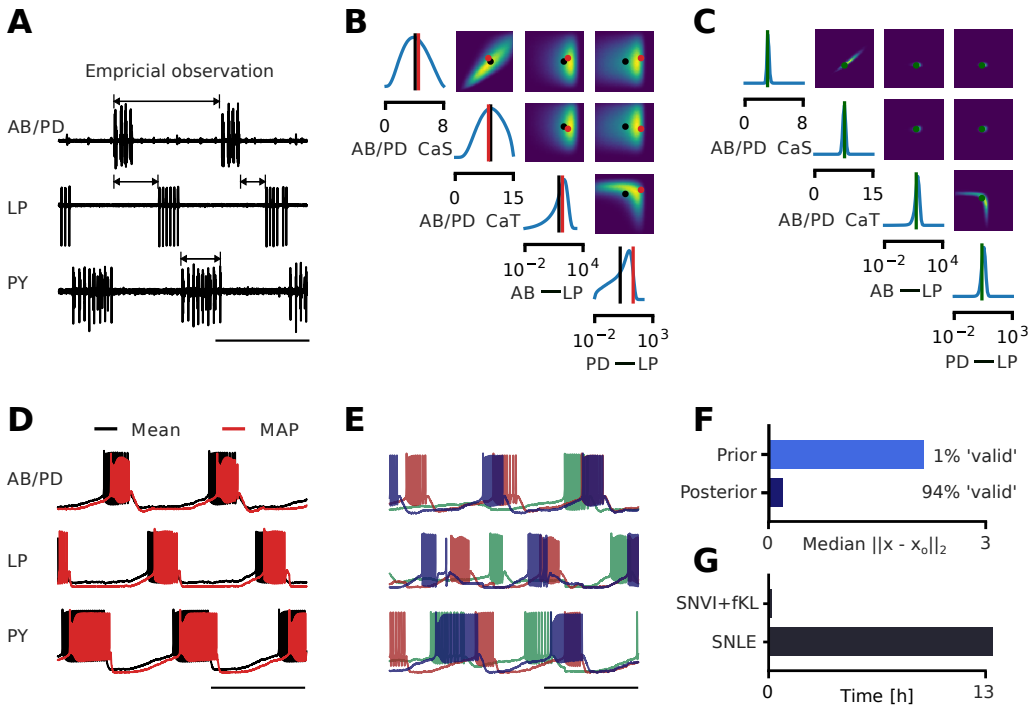

Figure 4.3.: **(A)** Empirical observation, arrows indicate some of the summary statistics. Scale bar is one second. **(B)** Cornerplot showing a subset of the marginal and pairwise marginal distributions of the 31-dimensional posterior (full posterior in Appendix Fig. A.10). Red dot: MAP. Black dot: Posterior mean. **(C)** Conditional distributions $p(\boldsymbol{\theta}_{i,j}|\boldsymbol{x}, \boldsymbol{\theta}_{\neq i,j})$. Green dot shows the sample on which we condition. **(D)** Simulated traces from the posterior mean and MAP. **(E)** Simulated traces of three posterior samples. **(F)** Posterior predictive and prior predictive median (z-scored) distances from the observation. **(G)** Time required to obtain 10k samples: SNVI takes 11 minutes and SNLE with 100-chain MCMC 808 minutes, i.e. over 13 hours.

4.3. Inference in a neuroscience model of the pyloric network

Finally, we applied SNVI to a simulator of the pyloric network in the stomatogastric ganglion (STG) of the crab *Cancer Borealis*, a well-characterized circuit producing rhythmic activity. The model consists of three model neurons (each with eight membrane conductances) with seven synapses (31 parameters in total) and produces voltage traces that can be characterized with 15 established summary statistics (Prinz et al., 2003; 2004). In this model, disparate parameter sets can produce similar activity, leading to a posterior distribution with broad marginals but narrow conditionals (Prinz et al., 2004; Gonçalves et al., 2020). Previous work has used millions of simulations from prior samples and performed amortized inference with NPE (18 million simulations in Gonçalves et al. (2020), 9 million in Deistler et al. (2021)). Sequential neural posterior estimation (SNPE) struggles

on this problem due to leakage, whereas SNLE and SNRE with MCMC are inefficient (Durkan et al., 2020a). Here, we apply SNVI to identify the posterior distribution given an extracellular recording of the stomatogastric motor neuron (Fig. 4.3A) (Haddad and Marder, 2021; 2018). We demonstrate that SNVI can perform multi-round inference and obtains the posterior distribution with only 350,000 simulations – 25 times fewer than previous methods!

We ran SNVI with a likelihood-estimator with the fKL divergence (SNVI+fKL), SIR, and a binary calibration kernel. The calibration kernel is 1 for valid simulations and 0 for simulations that produce NaN in at least 1 summary statistic (e.g. gaps between bursts cannot be defined if there are no bursts). Because only 1% of the simulations from prior samples are valid (Fig. 4.3F), we used 50,000 simulations in the first round and continued for 30 rounds with 10,000 simulations each.

The posterior is complex and reveals strong correlations and nonlinear relationships between parameters (Fig. 4.3B showing 4 out of 31 dimensions, full posterior in Appendix Fig. A.10). The conditional distributions $p(\boldsymbol{\theta}_{i,j}|\boldsymbol{x}, \boldsymbol{\theta}_{\neq i,j})$ given a posterior sample (Fig. 4.3C) are narrow, demonstrating that parameters have to be finely tuned to generate the summary statistics of the experimentally measured activity.

We used posterior predictive checks to inspect the quality of the posterior. When simulating data from the posterior mean and posterior mode (MAP), we find that both of them match the statistics of the experimental activity (Fig. 4.3D). Similarly, samples from the posterior distribution closely match statistics of the experimental activity (Fig. 4.3E). Out of 10,000 posterior samples, 9366 (≈94%) generated activity with well-defined summary statistics (compared to 1% of prior samples). For the samples which generate well-defined summary statistics, the (z-scored) median distance between the observed data $\boldsymbol{x}_o$ and generated activity is smaller for posterior samples than for prior samples (Fig. 4.3F). We emphasize that an application of SNLE with MCMC would be estimated to take an additional 400 hours, due to 30 rounds of slow MCMC sampling (Fig. 4.3G) that would be required– instead of 27 hours for SNVI. Likewise, when running SNPE-C on this example, only 1 out of 2 million samples were within the prior bounds after the second round, requiring computationally expensive rejection sampling (Greenberg et al., 2019; Durkan et al., 2020a).

These results show that SNVI makes it possible to overcome the limitations of previous methods and allows sequential neural simulation-based inference methods to effectively and robustly scale to challenging inference problems of scientific interest. While it is difficult to rigorously evaluate the accuracy of the obtained posterior distribution due to a lack of ground truth, we observed that almost all posterior predictives have well-defined summary statistics (94% vs 80% in Gonçalves et al. (2020)) and that the posterior predictives closely match $\boldsymbol{x}_o$.

5. Discussion *

We introduced Sequential Neural Variational Inference (SNVI), an efficient, flexible, and robust approach to perform Bayesian inference in models with an intractable likelihood. We achieve this by combining likelihood-estimation (or likelihood-ratio estimation) with variational inference, further improved by using SIR for refining posteriors. We demonstrate that SNVI reduces the computational cost of inference while maintaining accuracy. We applied our approach to a neuroscience model of the pyloric network with 31 parameters and showed that it is 25 times more efficient than previous methods. Our results demonstrate that SNVI is a scalable and robust method for simulation-based inference, opening up new possibilities for Bayesian inference in models with intractable likelihoods.

We selected three variational objectives for SNVI which induce mass-covering behavior and are, therefore, well suited as a proposal for sampling from complex posterior distributions. We empirically evaluated all of these methods in terms of runtime and accuracy on four benchmark tasks. We found that, while their performance differed when using the raw VI output, they all showed similar performance after an additional, computationally cheap, sampling importance resampling (SIR) step. After the SIR step, all methods had similar accuracy as MCMC, and all methods outperformed a mode-seeking variational objective (reverse KL) which was used in a previously proposed method Wiqvist et al. (2021). Our results suggest that mass-covering VI objectives (regardless of their exact implementation) provide a means to perform fast and accurate inference in models with intractable likelihood, without loss of accuracy compared to MCMC. We include technical consideration on the choice of objective in Appendix Sec. A.2.

Furthermore, we present a way to calibrate likelihood-based methods towards appropriate regions in data space. This method can be efficiently applied in the case of invalid data as we demonstrate in Sec. 4.3. It was also applied to invalid data in SNPE by Lueckmann et al. (2017); however, the benefit of other calibration kernels is less clear. Using likelihood-based methods, we show that calibration requires a correction and derive the exact bias factor (see Appendix Sec. A.4). We give a proof of concept that calibration with ABC kernels can improve approximation quality in Appendix Fig. A.3. This can be attributed to a more accurate likelihood estimate since the variational posterior is obtained equivalently. Additionally, this correction factor has a close relationship to the likelihood implicitly used in Approximate Bayesian Computation, as demonstrated in Appendix Sec. A.5. Hence we can solely use the correction factor as a likelihood surrogate to obtain a simulation-efficient ABC method (similar to Wilkinson (2014)). We let further exploration to future work.

*Parts of this chapter are currently under review for publication and hence revised by Michael Deistler and Jakob Macke (see `https://openreview.net/forum?id=kZOUYdhqkNY`).

A common approach in sequential methods is to use the current posterior estimate as the proposal distribution for the next round, but more elaborate active-learning strategies for choosing new simulations are possible (Papamakarios and Murray, 2016; Papamakarios et al., 2019; Lueckmann et al., 2019a). SNVI can flexibly be combined with any active learning scheme, and unlike neural likelihood(-ratio) methods, does not require expensive MCMC sampling for updating posterior estimates. While this comes at the cost of having to train two neural networks (a likelihood-model and a posterior-model), the cost of training these neural networks is often negligible compared to the cost of simulations. Another method that trains both a likelihood- and a posterior network is Posterior-Aided Regularization (Kim et al., 2021), which regularizes the likelihood-estimate with a simultaneously trained posterior-estimate. This improves the modelling of multimodal posteriors, but the method still requires MCMC and thus scales poorly with the number of samples and dimensions. Likelihood-free variational inference (Tran et al., 2017) avoids learning a likelihood model by learning an implicit posterior distribution, but it requires an adversarial training objective which can be difficult to optimize and requires extensive hyperparameter tuning (Huszár, 2017).

Overall, SNVI combines the desirable properties of current methods: It can be combined with any active learning scheme, it can flexibly combine information from multiple datapoints, it returns a posterior distribution that can be sampled quickly, and it can robustly deal with missing data. SNVI speeds up inference relative to MCMC-based methods, sometimes by orders of magnitude, and can perform inference in large models with many parameters. SNVI therefore has potential to provide a new 'go-to' approach for simulation-based inference, and to open up new application domains for simulation-based Bayesian inference.

Bibliography

Abbott, L. and Marder, E.: 1998, Modeling small networks, *Methods in Neuronal Modeling* pp. 361–410.

Agrawal, A., Sheldon, D. and Domke, J.: 2020, Advances in black-box vi: Normalizing flows, importance weighting, and optimization, *arXiv preprint arXiv:2006.10343* .

Andrieu, C. and Roberts, G. O.: 2009, The pseudo-marginal approach for efficient Monte Carlo computations, *The Annals of Statistics* **37**(2), 697 – 725.

Beaumont, M. A.: 1999, Detecting population expansion and decline using microsatellites, *Genetics* **153**(4), 2013–2029.

Beaumont, M. A.: 2003, Estimation of population growth or decline in genetically monitored populations, *Genetics* **164**(3), 1139–1160.

Beaumont, M. A., Zhang, W. and Balding, D. J.: 2002a, Approximate bayesian computation in population genetics, *Genetics* **162**(4), 2025–2035.

Beaumont, M. A., Zhang, W. and Balding, D. J.: 2002b, Approximate bayesian computation in population genetics, *Genetics* **162**(4), 2025–2035.

Bingham, E., Chen, J. P., Jankowiak, M., Obermeyer, F., Pradhan, N., Karaletsos, T., Singh, R., Szerlip, P. A., Horsfall, P. and Goodman, N. D.: 2019, Pyro: Deep universal probabilistic programming, *J. Mach. Learn. Res.* **20**, 28:1–28:6.

Bishop, C. M.: 2006, Pattern recognition, *Machine learning* **128**(9).

Blei, D. M., Kucukelbir, A. and McAuliffe, J. D.: 2017, Variational inference: A review for statisticians, *Journal of the American Statistical Association* **112**(518), 859–877.

Blum, M. G. and François, O.: 2010, Non-linear regression models for approximate bayesian computation, *Statistics and computing* **20**(1), 63–73.

Burda, Y., Grosse, R. and Salakhutdinov, R.: 2016, Importance weighted autoencoders, *International Conference on Learning Representations* .

Chatterjee, S. and Diaconis, P.: 2018, The sample size required in importance sampling, *The Annals of Applied Probability* **28**(2), 1099–1135.

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

Cranmer, K., Pavez, J. and Louppe, G.: 2015, Approximating likelihood ratios with calibrated discriminative classifiers, *arXiv preprint arXiv:1506.02169* .

Cremer, C., Morris, Q. and Duvenaud, D.: 2017, Reinterpreting importance-weighted autoencoders.

Deistler, M., Macke, J. H. and Gonçalves, P. J.: 2021, Disparate energy consumption despite similar network activity, *bioRxiv* .

Dinh, L., Sohl-Dickstein, J. and Bengio, S.: 2017, Density estimation using real nvp, *International Conference in Learning Representations 2017* .

Domke, J. and Sheldon, D.: 2018, Importance weighting and variational inference, *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pp. 4475–4484.

Durkan, C., Bekasov, A., Murray, I. and Papamakarios, G.: 2019a, Neural spline flows, *Advances in Neural Information Processing Systems* **32**, 7511–7522.

Durkan, C., Bekasov, A., Murray, I. and Papamakarios, G.: 2019b, Neural spline flows, *in* H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox and R. Garnett (eds), *Advances in Neural Information Processing Systems*, Vol. 32, Curran Associates, Inc.

Durkan, C., Murray, I. and Papamakarios, G.: 2020a, On contrastive learning for likelihood-free inference, *International Conference on Machine Learning*, PMLR, pp. 2771–2781.

Durkan, C., Murray, I. and Papamakarios, G.: 2020b, On contrastive learning for likelihood-free inference, *International Conference on Machine Learning*, PMLR, pp. 2771–2781.

Figurnov, M., Mohamed, S. and Mnih, A.: 2018, Implicit reparameterization gradients, *CoRR* **abs/1805.08498**.

Friedman, J.: 2004, On multivariate goodness-of-fit and two-sample testing, *Technical report*, Citeseer.

Gonçalves, P. J., Lueckmann, J.-M., Deistler, M., Nonnenmacher, M., Öcal, 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.

Graves, A.: 2016, Stochastic backpropagation through mixture density distributions.

Greenberg, D., Nonnenmacher, M. and Macke, J.: 2019, Automatic posterior transformation for likelihood-free inference, *International Conference on Machine Learning*, pp. 2404–2414.

Gunawan, D., Tran, M.-N. and Kohn, R.: 2017, Fast inference for intractable likelihood problems using variational bayes.

Gutmann, M. U., Cor, J. and er: 2016, Bayesian optimization for likelihood-free inference of simulator-based statistical models, *Journal of Machine Learning Research* **17**(125), 1–47.

Gutmann, M. U., Dutta, R., Kaski, S. and Corander, J.: 2018, Likelihood-free inference via classification, *Statistics and Computing* **28**(2), 411–425.

Haddad, S. A. and Marder, E.: 2018, Circuit robustness to temperature perturbation is altered by neuromodulators, *Neuron* **100**(3), 609–623.e3.

Haddad, S. A. and Marder, E.: 2021, Recordings from the C. borealis Stomatogastric Nervous System at different temperatures in the decentralized condition.

He, Z., Xu, Z. and Wang, X.: 2021, Unbiased mlmc-based variational bayes for likelihood-free inference.

Hermans, J., Begy, V. and Louppe, G.: 2020, Likelihood-free mcmc with amortized approximate ratio estimators, *International Conference on Machine Learning*, PMLR, pp. 4239–4248.

Hesterberg, T. C.: 2003, *Advances in importance sampling*, PhD thesis, Citeseer.

Hines, M. L. and Carnevale, N. T.: 2001, Neuron: A tool for neuroscientists, *The Neuroscientist* **7**(2), 123–135. PMID: 11496923.

Huszár, F.: 2017, Variational inference using implicit distributions.

Jerfel, G., Wang, S., Fannjiang, C., Heller, K. A., Ma, Y. and Jordan, M. I.: 2021, Variational refinement for importance sampling using the forward kullback-leibler divergence.

Kim, D., Song, K., Shin, S., Kang, W. and Moon, I.-C.: 2021, Posterior-aided regularization for likelihood-free inference, *arXiv preprint arXiv:2102.07770* .

Kingma, D. P., Salimans, T., Jozefowicz, R., Chen, X., Sutskever, I. and Welling, M.: 2016, Improved variational inference with inverse autoregressive flow, *Advances in neural information processing systems* **29**, 4743–4751.

Kingma, D. P. and Welling, M.: 2014a, Auto-encoding variational bayes.

Kingma, D. P. and Welling, M.: 2014b, An introduction to variational autoencoders, *International Conference on Learning Representations* .

Kullback, S. and Leibler, R. A.: 1951, On information and sufficiency, *The annals of mathematical statistics* **22**(1), 79–86.

Li, Y. and Turner, R.: 2016, Rényi divergence variational inference, *Advances in Neural Information Processing Systems 29 (NIPS 2016)* pp. 1073–1081.

Lopez-Paz, D. and Oquab, M.: 2017, Revisiting classifier two-sample tests, *International Conference on Learning Representations*.

Lueckmann, J.-M., Bassetto, G., Karaletsos, T. and Macke, J. H.: 2019a, Likelihood-free inference with emulator networks, *in* F. Ruiz, C. Zhang, D. Liang and T. Bui (eds), *Proceedings of The 1st Symposium on Advances in Approximate Bayesian Inference*, Vol. 96 of *Proceedings of Machine Learning Research*, pp. 32–53.

Lueckmann, J.-M., Bassetto, G., Karaletsos, T. and Macke, J. H.: 2019b, Likelihood-free inference with emulator networks, *Symposium on Advances in Approximate Bayesian Inference*, PMLR, pp. 32–53.

Lueckmann, J.-M., Boelts, J., Greenberg, D., Goncalves, P. and Macke, J.: 2021, Benchmarking simulation-based inference, *in* A. Banerjee and K. Fukumizu (eds), *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, Vol. 130 of *Proceedings of Machine Learning Research*, PMLR, pp. 343–351.

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, *Advances in Neural Information Processing Systems*, pp. 1289–1299.

Marjoram, P., Molitor, J., Plagnol, V. and Tavaré, S.: 2003, Markov chain monte carlo without likelihoods, *Proceedings of the National Academy of Sciences* **100**(26), 15324–15328.

Meeds, E., Leenders, R. and Welling, M.: 2015, Hamiltonian abc.

Miller, A. C., Foti, N. and Adams, R. P.: 2017, Variational boosting: Iteratively refining posterior approximations.

Miller, B. K., Cole, A., Forré, P., Louppe, G. and Weniger, C.: 2022, Truncated marginal neural ratio estimation, *Advances in Neural Information Processing Systems* .

Mohamed, S. and Lakshminarayanan, B.: 2016, Learning in implicit generative models, *arXiv preprint arXiv:1610.03483* .

Mohamed, S., Rosca, M., Figurnov, M. and Mnih, A.: 2020, Monte carlo gradient estimation in machine learning., *J. Mach. Learn. Res.* **21**(132), 1–62.

Morningstar, W., Vikram, S., Ham, C., Gallagher, A. and Dillon, J.: 2021, Automatic differentiation variational inference with mixtures, *International Conference on Artificial Intelligence and Statistics*, PMLR, pp. 3250–3258.

Neal, R. M.: 2003, Slice sampling, *The annals of statistics* **31**(3), 705–767.

Nguyen, H.: 2017, A universal approximation theorem for gaussian-gated mixture of experts models, *SSRN Electronic Journal* .

Oesterle, J., Behrens, C., Schröder, C., Hermann, T., Euler, T., Franke, K., Smith, R. G., Zeck, G. and Berens, P.: 2020, Bayesian inference for biophysical neuron models enables stimulus optimization for retinal neuroprosthetics, *Elife* **9**, e54997.

Ong, V. M. H., Nott, D. J., Tran, M.-N., Sisson, S. A. and Drovandi, C. C.: 2017, Variational bayes with synthetic likelihood, *Statistics and Computing* **28**(4), 971–988.

Pacchiardi, L., Künzli, P., Schöngens, M., Chopard, B. and Dutta, R.: 2021, Distance-learning for approximate bayesian computation to model a volcanic eruption, *Sankhya B* **83**(1), 288–317.

Papamakarios, G.: 2019, *Neural Density Estimation and Likelihood-free Inference*, PhD thesis.

Papamakarios, G. and Murray, I.: 2016, Fast ε-free inference of simulation models with bayesian conditional density estimation, *Advances in Neural Information Processing Systems*, pp. 1028–1036.

Papamakarios, G., Nalisnick, E., Rezende, D. J., Mohamed, S. and Lakshminarayanan, B.: 2021, Normalizing flows for probabilistic modeling and inference, *Journal of Machine Learning Research* **22**(57), 1–64.

Papamakarios, G., Pavlakou, T. and Murray, I.: 2017, Masked autoregressive flow for density estimation, *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 2335–2344.

Papamakarios, G., Sterratt, D. and Murray, I.: 2019, Sequential neural likelihood: Fast likelihood-free inference with autoregressive flows, *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 837–848.

Pires, G. G. P. F. and Figueiredo, M. A. T.: 2020, Variational mixture of normalizing flows.

Prangle, D.: 2015, Summary statistics in approximate bayesian computation.

Price, L. F., Drovandi, C. C., Lee, A. and Nott, D. J.: 2018, Bayesian synthetic likelihood, *Journal of Computational and Graphical Statistics* **27**(1), 1–11.

Prinz, A. A., Billimoria, C. P. and Marder, E.: 2003, Alternative to hand-tuning conductance-based models: construction and analysis of databases of model neurons, *Journal of neurophysiology* .

Prinz, A. A., Bucher, D. and Marder, E.: 2004, Similar network activity from disparate circuit parameters, *Nature neuroscience* **7**(12), 1345–1352.

Pritchard, J. K., Seielstad, M. T., Perez-Lezaun, A. and Feldman, M. W.: 1999, Population growth of human y chromosomes: a study of y chromosome microsatellites., *Molecular biology and evolution* **16**(12), 1791–1798.

Rainforth, T., Kosiorek, A., Le, T. A., Maddison, C., Igl, M., Wood, F. and Teh, Y. W.: 2018, Tighter variational bounds are not necessarily better, *International Conference on Machine Learning*, PMLR, pp. 4277–4285.

Ranganath, R., Gerrish, S. and Blei, D.: 2014, Black box variational inference, *Artificial Intelligence and Statistics*, PMLR, pp. 814–822.

Rényi, A. et al.: 1961, On measures of entropy and information, *Proceedings of the*

Fourth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics, The Regents of the University of California.

Rezende, D. J., Mohamed, S. and Wierstra, D.: 2014, Stochastic backpropagation and approximate inference in deep generative models, *International conference on machine learning*, PMLR, pp. 1278–1286.

Rezende, D. and Mohamed, S.: 2015, Variational inference with normalizing flows, *International conference on machine learning*, PMLR, pp. 1530–1538.

Roeder, G., Wu, Y. and Duvenaud, D. K.: 2017, Sticking the landing: Simple, lower-variance gradient estimators for variational inference, *Advances in Neural Information Processing Systems* **30**, 6925–6934.

Rubin, D. B.: 1984, Bayesianly justifiable and relevant frequency calculations for the applied statistician, *The Annals of Statistics* pp. 1151–1172.

Rubin, D. B.: 1988, Using the sir algorithm to simulate posterior distributions, *Bayesian statistics* **3**, 395–402.

Schröder, C., James, B., Lagnado, L. and Berens, P.: 2019, Approximate bayesian inference for a mechanistic model of vesicle release at a ribbon synapse, *bioRxiv* p. 669218.

Sisson, S. A., Fan, Y. and Tanaka, M. M.: 2007, Sequential monte carlo without likelihoods, *Proceedings of the National Academy of Sciences* **104**(6), 1760–1765.

Tejero-Cantero, A., Boelts, J., Deistler, M., Lueckmann, J.-M., Durkan, C., Gonçalves, P. J., Greenberg, D. S. and Macke, J. H.: 2020, sbi: A toolkit for simulation-based inference, *Journal of Open Source Software* **5**(52), 2505.

Thomas, O., Dutta, R., Corander, J., Kaski, S., Gutmann, M. U. et al.: 2021, Likelihood-free inference by ratio estimation, *Bayesian Analysis* .

Tran, D., Ranganath, R. and Blei, D. M.: 2017, Hierarchical implicit models and likelihood-free variational inference, *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 5529–5539.

Tran, M.-N., Nott, D. J. and Kohn, R.: 2016, Variational bayes with intractable likelihood.

Tucker, G., Lawson, D., Gu, S. and Maddison, C. J.: 2018, Doubly reparameterized gradient estimators for monte carlo objectives, *International Conference on Learning Representations*.

Wan, N., Li, D. and Hovakimyan, N.: 2020, f-divergence variational inference, *Advances in Neural Information Processing Systems* **33**.

Wangersky, P. J.: 1978, Lotka-volterra population models, *Annual Review of Ecology and Systematics* **9**(1), 189–218.

West, T. O., Berthouze, L., Farmer, S. F., Cagnan, H. and Litvak, V.: 2021, Inference of

brain networks with approximate bayesian computation–assessing face validity with an example application in parkinsonism, *NeuroImage* **236**, 118020.

Wilkinson, R.: 2014, Accelerating ABC methods using Gaussian processes, *in* S. Kaski and J. Corander (eds), *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics*, Vol. 33 of *Proceedings of Machine Learning Research*, PMLR, Reykjavik, Iceland, pp. 1015–1023.

Wilkinson, R. D.: 2013, Approximate bayesian computation (abc) gives exact results under the assumption of model error, *Statistical Applications in Genetics and Molecular Biology* **12**(2).

Wiqvist, S., Frellsen, J. and Picchini, U.: 2021, Sequential neural posterior and likelihood approximation.

Wood, S. N.: 2010, Statistical inference for noisy nonlinear ecological dynamic systems, *Nature* **466**(7310), 1102–1104.

A. Appendix

A.1. Overcoming vanishing gradients in the forward KL estimator*

We use an estimator of the forward Kullback-Leibler divergence (fKL) that is based on self-normalized importance sampling. In this section, we demonstrate that this estimator moves the variational distribution $q_\phi(\boldsymbol{\theta})$ towards the target density $p(\boldsymbol{x}_o, \boldsymbol{\theta})$ even if $q_\phi(\boldsymbol{\theta})$ and $p(\boldsymbol{x}_o, \boldsymbol{\theta})$ differ strongly.

For this analysis, we consider a Gaussian toy example with prior $p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}; 0, 4)$, likelihood $p(\boldsymbol{x}|\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{\theta}, 1)$, and observation $\boldsymbol{x}_o = 1$. The posterior distribution can be computed in closed-form as $p(\boldsymbol{\theta}|\boldsymbol{x}_o) = \mathcal{N}(\boldsymbol{\theta}; 4/5, 4/5)$. We aim to learn the posterior distribution using variational inference with the variational family $q_\mu(\boldsymbol{\theta}) = \mathcal{N}(\mu, 4/5)$ (note that μ is the only parameter). The best approximation within this family is $\mu^* = 4/5$.

We use this toy example to compare the gradient and the signal-to-noise ratio (SNR) of the self-normalized fKL estimator to the fKL estimator introduced by Wan et al. (2020). Fig. A.1A (left) shows the gradient of the loss for different values of μ. When $\mu \approx \mu^* = 4/5$, the fKL (gray) without self-normalization closely matches the true gradient (red). However, as μ is further from μ^*, the fKL first points in the wrong direction and then vanishes, which prevents learning. The self-normalized fKL (blue, orange, green) closely matches the gradient around $\mu \approx \mu^* = 4/5$ and does not vanish for μ that are far from μ^*. The gradient is stronger if more samples N are used to approximate the fKL. Similarly, the $\mathrm{SNR}(\nabla_\phi \mathcal{L}(\phi)) = |\mathbb{E}[\nabla_\phi \mathcal{L}(\phi)] / \sqrt{\mathrm{Var}(\nabla_\phi \mathcal{L}(\phi))}|$ does not vanish for μ that are far from μ^* for the self-normalized fKL.

To understand this behavior of the self-normalized fKL, we computed an approximation to the gradient $\nabla_\mu \mathcal{L}_{\mathrm{fKL}}(\mu)$ in this toy example. The fKL loss is given as:

$$\nabla_\mu \mathcal{L}_{\mathrm{fKL}}(\mu) = -\mathbb{E}_{\boldsymbol{\theta} \sim q_\phi} \left[\frac{w(\boldsymbol{\theta})}{\sum_{i=1}^N w(\boldsymbol{\theta})} \nabla_\mu \log q_\mu(\boldsymbol{\theta}) \right] \approx -\sum_{i=1}^N \frac{w(\boldsymbol{\theta}_i)}{\sum_{i=1}^N w(\boldsymbol{\theta}_i)} \nabla_\mu \log q_\mu(\boldsymbol{\theta}_i)$$

with weights $w(\boldsymbol{\theta}_i) = \frac{p(\boldsymbol{x}_o, \boldsymbol{\theta})}{q_\phi(\boldsymbol{\theta})}$. In the case where $q_\phi(\boldsymbol{\theta})$ differs strongly from $p(\boldsymbol{x}_o, \boldsymbol{\theta})$, the weights are often degenerate, i.e. the strongest weight is much larger than all others. In

*Parts of this section are currently under review for publication and hence revised by Michael Deistler and Jakob Macke (see `https://openreview.net/forum?id=kZOUYdhqkNY`).

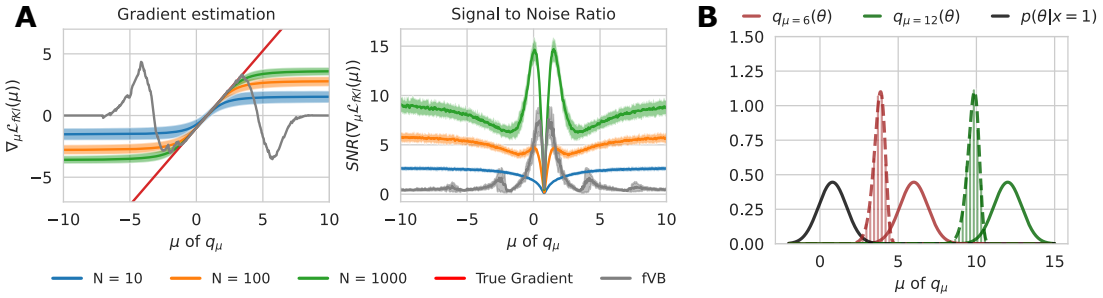

Figure A.1.: **A** Left: Gradient estimation on the Gaussian example. For values of μ around $\mu^* = 4/5$, all estimators provide good gradients. As μ is farther from μ^*, the forward variational bound (fVB) (grey) vanishes, whereas the self-normalized fVB approaches a constant. Right: SNR for the fVB and the self-normalized fVB. **B** Theoretical and empirical densities $p_R(r)$ for $\mu = 6$ and $\mu = 12$.

the worst case, $\tilde{w}(\boldsymbol{\theta}_i) = \frac{w(\boldsymbol{\theta}_i)}{\sum_{i=1}^N w(\boldsymbol{\theta}_i)} = 1$ for some i and the gradient estimator reduces to

$$\nabla_\mu \mathcal{L}_{\text{fKL}}(\mu) = -\nabla_\mu \log q_\mu(\arg\max_{\boldsymbol{\theta}_1,\dots,\boldsymbol{\theta}_N} w(\boldsymbol{\theta})) = -\nabla_\mu \log q_\mu(r)$$

The gradient of μ is thus determined by $r = \arg\max_{\boldsymbol{\theta}_1,\dots,\boldsymbol{\theta}_N} w(\boldsymbol{\theta})$, which itself can be considered a draw from a random variable R. We will now derive the probability density $p_R(r)$ of R.

If $\mu > \mu^*$, then $w(\boldsymbol{\theta})$ is monotonically decreasing in $\boldsymbol{\theta}$ because $w(\boldsymbol{\theta}) \propto \frac{\mathcal{N}(\boldsymbol{\theta};\mu^*,4/5)}{\mathcal{N}(\boldsymbol{\theta};\mu,4/5)} \propto \exp(5/4 \cdot \boldsymbol{\theta}(\mu^* - \mu))$. The cumulative distribution function $F_R(R \leq r)$ can then be written as

$$\begin{aligned} F_R(R \leq r) &= P(\arg\max_{\boldsymbol{\theta}_1,\dots,\boldsymbol{\theta}_n} w(\boldsymbol{\theta}) \leq r) = P(\min(\boldsymbol{\theta}_1,\dots,\boldsymbol{\theta}_n) \leq r) \\ &= 1 - P(\min(\boldsymbol{\theta}_1,\dots,\boldsymbol{\theta}_n) > r) = 1 - \prod_{i=1}^N P(\boldsymbol{\theta}_i > r) \\ &= 1 - (1 - F_{q_\phi}(r))^N \end{aligned}$$

Thus, R has the density $p_R(r) = \frac{d}{dr} F(R \leq r) = N(1 - F_{q_\phi}(r))^{N-1} q_\mu(r)$. The derivation is analogous for the case $\mu < \mu^*$. Because $\nabla_\mu \mathcal{L}_{\text{fKL}}(\mu) = \nabla_\mu \log q_\mu(r)$ for $r \sim p_R$, this allows us to compute the distribution of the gradient of μ (under the assumption that weights are degenerate).

We empirically validate this result on the Gaussian toy example. Fig. A.1B (left) shows the true posterior distribution (black), the variational density $q_\mu(\boldsymbol{\theta})$ for $\mu = 6$ and $\mu = 10$ and the corresponding $p_R(r)$ for $N = 1000$. The theoretically computed density $p_R(r)$ (dashed lines) matches the empirically observed distribution of $\arg\max_{\boldsymbol{\theta}_i=1\dots N} w(\boldsymbol{\theta}_i)$. For almost every value of $r \sim p_R(r)$, the gradient $\nabla_\mu \log q_\mu(r)$ is negative, thus driving $\nabla_\mu \mathcal{L}_{\text{fKL}}(\mu)$

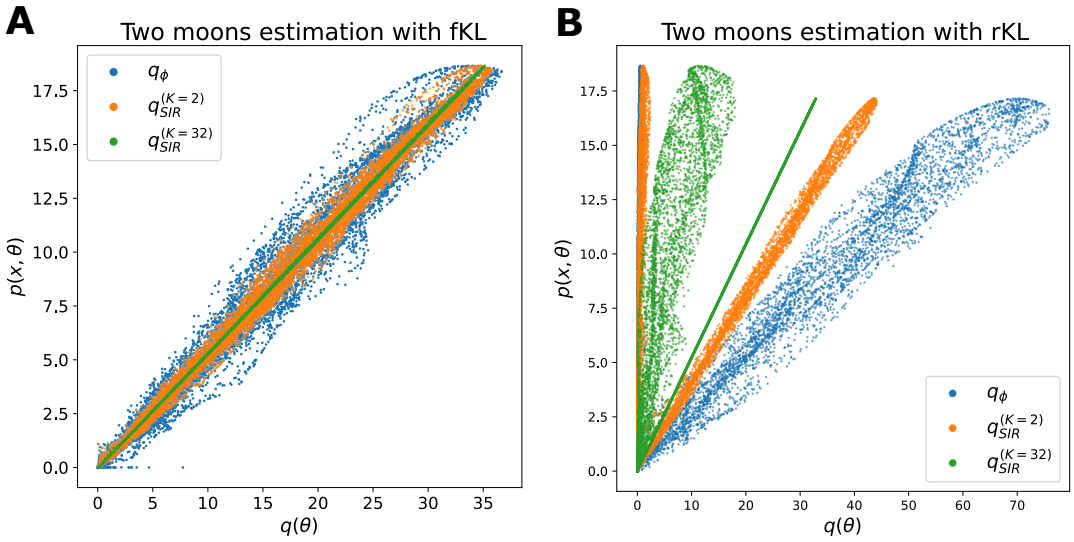

Figure A.2.: **A** SIR improvements on two moons example. We plot the joint density as learned by the likelihood-model $p(\boldsymbol{x}_o, \boldsymbol{\theta}) = \ell_\psi(\boldsymbol{x}_o|\boldsymbol{\theta})p(\boldsymbol{\theta})$ against the variational posterior q_ϕ (blue, obtained with the fKL), as well as the SIR-corrected density with $K = 2$ (orange) and $K = 32$ (green). Despite using an expressive normalizing flow as q_ϕ, SIR improves the accuracy. **B** shows the same but on a mode collapsed variational posterior obtained using the rKL. SIR can only slightly improve it.

into the correct direction. For larger N, the distribution $p_R(r)$ shifts towards the true posterior distribution and thus also the gradient signal increases.

Notably, for μ that are even further from μ^*, $\nabla_\mu \log q_\mu(r)$ remains relatively constant (Fig. A.1B, right). This explains why the gradient $\nabla_\mu \mathcal{L}_{\text{fKL}}(\mu)$ becomes constant in Fig. A.1A (left).

A.2. Choice of divergence†

In Fig. 4.2, we demonstrated that all mass-covering objectives perform similarly in terms of accuracy and runtime on the problems we considered. We here give technical recommendations for choosing a variational objective:

(i) *Closed-form posterior:* The variational posterior provides a closed-form approximation to the posterior, but this is no longer the case when SIR is used. While, in our results, all three approaches performed similarly *with* SIR, they can differ in their performance *without it*, and the forward KL and the α-divergence provided

†Parts of this section are currently under review for publication and hence revised by Michael Deistler and Jakob Macke (see `https://openreview.net/forum?id=kZOUYdhqkNY`).

better approximations than the IW-ELBO. Thus, if one seeks a posterior density that can be evaluated in closed-form, our results suggest using the forward KL or the α-divergence.

(ii) *Dimensionality of the parameter space*: We use an autoregressive normalizing flow as a variational family. These flows are very expressive, yet the computation time of their forward or backward pass scale with the dimensionality of $\boldsymbol{\theta}$ (Papamakarios et al., 2017; Kingma et al., 2016; Durkan et al., 2019a). The IW-ELBO and the α-divergences only require forward passes, whereas the forward KL requires forward and backward passes, thus making the forward KL expensive for high-dimensional parameter spaces. The STL estimator used in the IW-ELBO and the α-divergences also require forward and backward passes. We found that the STL estimator improves the performance of the IW-ELBO only weakly (Fig. A.6). Thus, in cases where computational cost is critical, our results suggest that using the IW-ELBO without the STL can give high accuracy at a low computational cost. Another way to reduce computational cost is to use alternative architectures for the normalizing flow, e.g. coupling layers (Durkan et al., 2019a; Papamakarios et al., 2021).

(iii) *Trading-off the mass-covering property with computational cost*: For α-divergences, one can trade-off the extent to which the divergence is mass-covering by choosing the value of α (low α is more mass-covering). As shown in Fig. A.6, high values of α benefit less from using the STL estimator. Thus, in cases where mass-covering behavior of the algorithm is less crucial, the STL estimator can be waived, leading to lower computational cost because the normalizing flow requires only forward passes (see point (ii)).

A.3. Improvement through SIR[‡]

We use SIR to refine samples obtained from the variational posterior. Consistent with Agrawal et al. (2020), we found that using SIR always helps to improve the approximation quality even using complex variational families such as normalizing flows (compare dotted and solid lines in Fig. 4.2).

SIR particularly improves the posterior estimate when the proposal (i.e. the variational posterior) is overdispersed. This provides an explanation for why SIR is particularly useful for the mass-covering divergences used in SNVI, and less so for mode-covering divergences (as used in SNPLA). We show this in Fig. A.2 on Two moons. Using SIR, the slightly overdispersed variational approximation obtained through the fKL estimator can be strongly refined. Yet on variational approximations with e.g. collapsed mode, SIR leads only to slight improvements.

[‡]Parts of this section are currently under review for publication and hence revised by Michael Deistler and Jakob Macke (see `https://openreview.net/forum?id=kZOUYdhqkNY`).

Algorithm 4: SNVI with calibration

Inputs: prior $p(\boldsymbol{\theta})$, observation $\boldsymbol{x}_o$, divergence D, simulations per round N, number of rounds R, selection strategy $\mathcal{S}$ and calibration kernel K.
Outputs: Approximate likelihood ℓ_ψ , variational posterior q_ϕ and calibration network c_ζ.
Initialize: Proposal $\tilde{p}(\boldsymbol{\theta}) = p(\boldsymbol{\theta})$, simulation dataset $\mathcal{X} = \{\}$, calibration dataset $\mathcal{C} = \{\}$
for $r \in [1, ..., R]$ **do**
  for $i \in [1, ..., N]$ **do**
    $\boldsymbol{\theta}_i = \mathcal{S}(\tilde{p}, \ell_\phi, p)$; `// sample` $\boldsymbol{\theta}_i \sim \tilde{p}(\boldsymbol{\theta})$
    simulate $\boldsymbol{x}_i \sim p(\boldsymbol{x}|\boldsymbol{\theta}_i)$; `// run the simulator on` $\boldsymbol{\theta}_i$
    add $(\boldsymbol{\theta}_i, K(\boldsymbol{x}_i, \boldsymbol{x}_o))$ to $\mathcal{C}$
    if $K(\boldsymbol{x}, \boldsymbol{x}_o) > 0$ **then**
      add $(\boldsymbol{\theta}_i, \boldsymbol{x}_i)$ to $\mathcal{X}$
    end
  end
  (re-)train ℓ_ψ; $\psi^* = \arg\min_\psi -\frac{1}{N} \sum_{(\boldsymbol{x}_i, \boldsymbol{\theta}_i) \in \mathcal{X}} K(\boldsymbol{x}_i, \boldsymbol{x}_o) \log \ell_\psi(\boldsymbol{x}_i|\boldsymbol{\theta}_i)$; `// or SNRE`
  (re-)train c_ζ; $\zeta^* = \arg\min_\zeta \frac{1}{N} \sum_{(\boldsymbol{\theta}_i, K(\boldsymbol{x}_i, \boldsymbol{x}_o))} \mathcal{L}(c_\zeta(\boldsymbol{\theta}_i), K(\boldsymbol{x}_i, \boldsymbol{x}_o))$; `// MSE or cross-entropy for binary calibration kernel`
  (re-)train q_ϕ; $\phi^* = \arg\min_\phi D(q_\phi(\boldsymbol{\theta}) || p(\boldsymbol{\theta}|\boldsymbol{x}_o))$ with

$$p(\boldsymbol{\theta}|\boldsymbol{x}_o) \propto p(\boldsymbol{x}_o|\boldsymbol{\theta}) p(\boldsymbol{\theta}) \approx \ell_{\psi^*}(\boldsymbol{x}_o|\boldsymbol{\theta}) c_{\zeta^*}(\boldsymbol{\theta}) p(\boldsymbol{\theta})$$

  $\tilde{p}(\boldsymbol{\theta}) = q_\phi(\boldsymbol{\theta})$
end

A.4. Proofs for calibration kernel[§]

Many simulators can produce unreasonable or undefined values (Lueckmann et al., 2017). To be able to use SNVI in these cases, we developed a loss-reweighing strategy with calibration kernels. Theorem A.1 and Lemma A.3 are relevant to SNLVI, Theorem A.2 and Lemma A.4 are relevant to SNRVI, and Lemma A.5 is relevant to both methods.

Theorem A.1 and Theorem A.2 provide a means to use a calibration kernel in the training of the likelihood(-ratio)-model such that one can still recover the posterior density. In SNVI, we sample from the (unnormalized) potential function with variational inference. However, one can also use Theorem A.1 and Theorem A.2 in combination with SNLE and SNRE and draw samples from the potential function with MCMC.

Both SNLVI and SNRVI with calibration kernels rely on the estimation of $\mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x}|\boldsymbol{\theta})}[K(\boldsymbol{x}, \boldsymbol{x}_o)]$.

[§]Parts of this section are currently under review for publication and hence revised by Michael Deistler and Jakob Macke (see `https://openreview.net/forum?id=kZOUYdhqkNY`).

We estimate this term with a feed-forward regression neural network $c_\zeta(\boldsymbol{\theta})$ (see Lemma A.5). The network is trained on pairs $(\boldsymbol{\theta}, K(\boldsymbol{x}, \boldsymbol{x}_o))$, where $\boldsymbol{\theta}$ and $\boldsymbol{x}$ are the same pairs as used for training the likelihood(-ratio)-model. For general calibration kernels $K(\boldsymbol{x}, \boldsymbol{x}_o)$, we use a mean-squared error loss, whereas in the case of invalid data, we parameterize $c_\zeta(\boldsymbol{\theta})$ as a logistic regression network and train it with a cross-entropy loss (since the calibration kernel $K(\boldsymbol{x}, \boldsymbol{x}_o)$ is a binary function: 0 for invalid data, 1 for valid data).

In the case of excluding invalid data from the dataset, $K(\boldsymbol{x}, \boldsymbol{x}_o)$ is a binary function that is 1 for valid data and 0 for invalid data. In practice, we remove the simulations for which $K(\boldsymbol{x}, \boldsymbol{x}_o) = 0$ from the dataset that is used to train the likelihood(-ratio)-model (since they do not contribute to the loss) and train $\ell_\psi(\boldsymbol{x}|\boldsymbol{\theta})$ only on the valid simulations.

Theorem A.1. *Let $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^+$ be a kernel. Let $\ell_{\psi^*}(\boldsymbol{x}|\boldsymbol{\theta})$ be the minimizer of the objective*

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{\theta},\boldsymbol{x}\sim\tilde{p}(\boldsymbol{\theta},\boldsymbol{x})}[K(\boldsymbol{x}, \boldsymbol{x}_o) \log(\ell_\psi(\boldsymbol{x}|\boldsymbol{\theta}))]$$

and let $c_{\zeta^*}(\boldsymbol{\theta})$ be the minimizer of

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{\theta},\boldsymbol{x}\sim\tilde{p}(\boldsymbol{\theta},\boldsymbol{x})}[(c_\zeta(\boldsymbol{\theta}) - K(\boldsymbol{x}, \boldsymbol{x}_o))^2]$$

Then the potential function

$$\mathcal{P}(\boldsymbol{\theta}) = \ell_{\psi^*}(\boldsymbol{x}_o|\boldsymbol{\theta})p(\boldsymbol{\theta})c_{\zeta^*}(\boldsymbol{\theta})$$

is proportional to the posterior density $p(\boldsymbol{\theta}|\boldsymbol{x}_o)$.

Proof. Using Lemma A.3 and Lemma A.4, we get

$$\begin{aligned}
\mathcal{P}(\boldsymbol{\theta}) &= \ell_{\psi^*}(\boldsymbol{x}_o|\boldsymbol{\theta})p(\boldsymbol{\theta})c_{\zeta^*}(\boldsymbol{\theta}) \\
&= \frac{K(\boldsymbol{x}, \boldsymbol{x}_o)p(\boldsymbol{x}_o|\boldsymbol{\theta})}{\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x}|\boldsymbol{\theta})}[K(\boldsymbol{x}, \boldsymbol{x}_o)]} p(\boldsymbol{\theta})\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x}|\boldsymbol{\theta})}[K(\boldsymbol{x}, \boldsymbol{x}_o)] \\
&= K(\boldsymbol{x}, \boldsymbol{x}_o)p(\boldsymbol{x}_o|\boldsymbol{\theta})p(\boldsymbol{\theta}) \\
&\propto p(\boldsymbol{\theta}|\boldsymbol{x}_o)
\end{aligned}$$

□

Theorem A.2. *Let $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^+$ be a kernel. Let $\ell_{\psi^*}(\boldsymbol{x}, \boldsymbol{\theta})$ be the minimizer of the objective*

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{\theta},\boldsymbol{x}\sim\tilde{p}(\boldsymbol{\theta},\boldsymbol{x})}\left[K(\boldsymbol{x}, \boldsymbol{x}_o) \log(\ell_{\psi^*}(\boldsymbol{x}, \boldsymbol{\theta}))\right] + \mathbb{E}_{\boldsymbol{\theta},\boldsymbol{x}\sim p(\boldsymbol{\theta})p(\boldsymbol{x})}\left[K(\boldsymbol{x}, \boldsymbol{x}_o) \log(1 - \ell_{\psi^*}(\boldsymbol{x}, \boldsymbol{\theta}))\right]$$

and let $c_{\zeta^*}(\boldsymbol{\theta})$ be the minimizer of

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{\theta},\boldsymbol{x}\sim\tilde{p}(\boldsymbol{\theta},\boldsymbol{x})}[(c_\zeta(\boldsymbol{\theta}) - K(\boldsymbol{x}, \boldsymbol{x}_o))^2]$$

Then the potential function

$$\mathcal{P}(\boldsymbol{\theta}) = \ell_{\psi^*}(\boldsymbol{x}_o, \boldsymbol{\theta})p(\boldsymbol{\theta})c_{\zeta^*}(\boldsymbol{\theta})$$

is proportional to the posterior density $p(\boldsymbol{\theta}|\boldsymbol{x}_o)$.

Proof. Using Lemma A.4 and Lemma A.5, we get

$$\begin{aligned}
\mathcal{P}(\boldsymbol{\theta}) &= \ell_{\psi^*}(\boldsymbol{x}_o, \boldsymbol{\theta}) p(\boldsymbol{\theta}) c_{\zeta^*}(\boldsymbol{\theta}) \\
&= \frac{\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}[K(\boldsymbol{x}, \boldsymbol{x}_o)]}{\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x}|\boldsymbol{\theta})}[K(\boldsymbol{x}, \boldsymbol{x}_o)]} \frac{p(\boldsymbol{x}_o|\boldsymbol{\theta})}{p(\boldsymbol{x}_o)} p(\boldsymbol{\theta}) \mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x}|\boldsymbol{\theta})}[K(\boldsymbol{x}, \boldsymbol{x}_o)] \\
&= \mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}[K(\boldsymbol{x}, \boldsymbol{x}_o)] \frac{p(\boldsymbol{x}_o|\boldsymbol{\theta})}{p(\boldsymbol{x}_o)} p(\boldsymbol{\theta}) \\
&\propto p(\boldsymbol{\theta}|\boldsymbol{x}_o)
\end{aligned}$$

□

Lemma A.3. *Let $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^+$ be an arbitrary kernel. Then, the objective*

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{\theta}, \boldsymbol{x}\sim\tilde{p}(\boldsymbol{\theta}, \boldsymbol{x})}[K(\boldsymbol{x}, \boldsymbol{x}_o) \log(\ell_\psi(\boldsymbol{x}|\boldsymbol{\theta}))]$$

is minimized if and only if $\ell_\psi(\boldsymbol{x}|\boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} K(\boldsymbol{x}, \boldsymbol{x}_o) p(\boldsymbol{x}|\boldsymbol{\theta})$ for all $\boldsymbol{\theta} \in$ support$(\tilde{p}(\boldsymbol{\theta}))$, with normalizing constant $Z(\boldsymbol{\theta}) = \int K(\boldsymbol{x}, \boldsymbol{x}_o) p(\boldsymbol{x}|\boldsymbol{\theta}) d\boldsymbol{x} = \mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x}|\boldsymbol{\theta})}[K(\boldsymbol{x}, \boldsymbol{x}_o)]$.

Proof.

$$\begin{aligned}
\mathcal{L} &= \mathbb{E}_{\boldsymbol{\theta}, \boldsymbol{x}\sim\tilde{p}(\boldsymbol{\theta}, \boldsymbol{x})}[K(\boldsymbol{x}, \boldsymbol{x}_o) \log(\ell_\psi(\boldsymbol{x}|\boldsymbol{\theta}))] \\
&= \iint K(\boldsymbol{x}, \boldsymbol{x}_o) \tilde{p}(\boldsymbol{\theta}, \boldsymbol{x}) \log(\ell_\psi(\boldsymbol{x}|\boldsymbol{\theta})) d\boldsymbol{x} d\boldsymbol{\theta} \\
&= \iint K(\boldsymbol{x}, \boldsymbol{x}_o) \tilde{p}(\boldsymbol{\theta}) p(\boldsymbol{x}|\boldsymbol{\theta}) \log(\ell_\psi(\boldsymbol{x}|\boldsymbol{\theta})) d\boldsymbol{x} d\boldsymbol{\theta} \\
&= \int \tilde{p}(\boldsymbol{\theta}) \int K(\boldsymbol{x}, \boldsymbol{x}_o) p(\boldsymbol{x}|\boldsymbol{\theta}) \log(\ell_\psi(\boldsymbol{x}|\boldsymbol{\theta})) d\boldsymbol{x} d\boldsymbol{\theta}
\end{aligned}$$

Since $\int K(\boldsymbol{x}, \boldsymbol{x}_o) p(\boldsymbol{x}|\boldsymbol{\theta}) \log(\ell_\psi(\boldsymbol{x}|\boldsymbol{\theta})) d\boldsymbol{x} \propto D_{\mathrm{KL}}\big(\frac{1}{Z(\boldsymbol{\theta})} K(\boldsymbol{x}, \boldsymbol{x}_o) p(\boldsymbol{x}|\boldsymbol{\theta}), \ell_\psi(\boldsymbol{x}|\boldsymbol{\theta})\big)$, this term is minimized if and only if $\ell_\psi(\boldsymbol{x}|\boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} K(\boldsymbol{x}, \boldsymbol{x}_o) p(\boldsymbol{x}|\boldsymbol{\theta})$ for all $\boldsymbol{\theta} \in \mathrm{support}(\tilde{p}(\boldsymbol{\theta}))$ with $Z(\boldsymbol{\theta}) = \int K(\boldsymbol{x}, \boldsymbol{x}_o) p(\boldsymbol{x}|\boldsymbol{\theta}) d\boldsymbol{x} = \mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x}|\boldsymbol{\theta})}[K(\boldsymbol{x}, \boldsymbol{x}_o)]$. □

Lemma A.4. *Let $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^+$ be an arbitrary kernel. Then, the objective*

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{\theta}, \boldsymbol{x}\sim\tilde{p}(\boldsymbol{\theta}, \boldsymbol{x})}\left[K(\boldsymbol{x}, \boldsymbol{x}_o) \log(\ell_{\psi^*}(\boldsymbol{x}, \boldsymbol{\theta}))\right] + \mathbb{E}_{\boldsymbol{\theta}, \boldsymbol{x}\sim\tilde{p}(\boldsymbol{\theta}) p(\boldsymbol{x})}\left[K(\boldsymbol{x}, \boldsymbol{x}_o) \log(1 - \ell_{\psi^*}(\boldsymbol{x}, \boldsymbol{\theta}))\right]$$

is minimized if and only if $\ell_\psi(\boldsymbol{x}, \boldsymbol{\theta}) = \frac{\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}[K(\boldsymbol{x}, \boldsymbol{x}_o)]}{\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x}|\boldsymbol{\theta})}[K(\boldsymbol{x}, \boldsymbol{x}_o)]} \frac{p(\boldsymbol{x}|\boldsymbol{\theta})}{p(\boldsymbol{x})}$ for all $\boldsymbol{\theta} \in$ support$(\tilde{p}(\boldsymbol{\theta}))$.

Proof. We begin by rearranging the expectations:

$$\begin{aligned}
\mathcal{L} &= \mathbb{E}_{\boldsymbol{\theta},\boldsymbol{x}\sim\tilde{p}(\boldsymbol{\theta},\boldsymbol{x})}\left[K(\boldsymbol{x},\boldsymbol{x}_o)\log(\ell_{\psi^*}(\boldsymbol{x},\boldsymbol{\theta}))\right] + \mathbb{E}_{\boldsymbol{\theta},\boldsymbol{x}\sim\tilde{p}(\boldsymbol{\theta})\tilde{p}(\boldsymbol{x})}\left[K(\boldsymbol{x},\boldsymbol{x}_o)\log(1-\ell_{\psi^*}(\boldsymbol{x},\boldsymbol{\theta}))\right] \\
&= \iint \tilde{p}(\boldsymbol{\theta},\boldsymbol{x})K(\boldsymbol{x},\boldsymbol{x}_o)\log(\ell_{\psi^*}(\boldsymbol{x},\boldsymbol{\theta}))d\boldsymbol{\theta}d\boldsymbol{x} + \\
&\qquad\qquad \iint \tilde{p}(\boldsymbol{\theta})\tilde{p}(\boldsymbol{x})K(\boldsymbol{x},\boldsymbol{x}_o)\log(1-\ell_{\psi^*}(\boldsymbol{x},\boldsymbol{\theta}))d\boldsymbol{\theta}d\boldsymbol{x} \\
&= \iint \frac{\tilde{p}(\boldsymbol{\theta},\boldsymbol{x})K(\boldsymbol{x},\boldsymbol{x}_o)}{\iint \tilde{p}(\boldsymbol{\theta},\boldsymbol{x})K(\boldsymbol{x},\boldsymbol{x}_o)d\boldsymbol{\theta}d\boldsymbol{x}}\log(\ell_{\psi^*}(\boldsymbol{x},\boldsymbol{\theta}))d\boldsymbol{\theta}d\boldsymbol{x} + \\
&\qquad\qquad \iint \frac{\tilde{p}(\boldsymbol{\theta})\tilde{p}(\boldsymbol{x})K(\boldsymbol{x},\boldsymbol{x}_o)}{\iint \tilde{p}(\boldsymbol{\theta})\tilde{p}(\boldsymbol{x})K(\boldsymbol{x},\boldsymbol{x}_o)d\boldsymbol{\theta}d\boldsymbol{x}}\log(1-\ell_{\psi^*}(\boldsymbol{x},\boldsymbol{\theta}))d\boldsymbol{\theta}d\boldsymbol{x} \\
&= \mathbb{E}_{\boldsymbol{\theta},\boldsymbol{x}\sim\pi_{\text{joint}}(\boldsymbol{\theta},\boldsymbol{x})}\left[\log(\ell_{\psi^*}(\boldsymbol{x},\boldsymbol{\theta}))\right] + \mathbb{E}_{\boldsymbol{\theta},\boldsymbol{x}\sim\pi_{\text{marginal}}(\boldsymbol{\theta},\boldsymbol{x})}\left[\log(1-\ell_{\psi^*}(\boldsymbol{x},\boldsymbol{\theta}))\right]
\end{aligned}$$

where we introduced

$$\pi_{\text{joint}}(\boldsymbol{\theta},\boldsymbol{x}) = \frac{\tilde{p}(\boldsymbol{\theta},\boldsymbol{x})K(\boldsymbol{x},\boldsymbol{x}_o)}{\iint \tilde{p}(\boldsymbol{\theta},\boldsymbol{x})K(\boldsymbol{x},\boldsymbol{x}_o)d\boldsymbol{\theta}d\boldsymbol{x}} \qquad \pi_{\text{marginal}}(\boldsymbol{\theta},\boldsymbol{x}) = \frac{\tilde{p}(\boldsymbol{\theta})\tilde{p}(\boldsymbol{x})K(\boldsymbol{x},\boldsymbol{x}_o)}{\iint \tilde{p}(\boldsymbol{\theta})\tilde{p}(\boldsymbol{x})K(\boldsymbol{x},\boldsymbol{x}_o)}$$

Since binary classification recovers density ratios (Cranmer et al., 2015; Mohamed and Lakshminarayanan, 2016; Gutmann et al., 2018), we get

$$\begin{aligned}
\ell_{\psi^*}(\boldsymbol{x},\boldsymbol{\theta}) &= \frac{\pi_{\text{joint}}(\boldsymbol{\theta},\boldsymbol{x})}{\pi_{\text{marginal}}(\boldsymbol{\theta},\boldsymbol{x})} \\
&= \frac{\frac{1}{\iint \tilde{p}(\boldsymbol{\theta},\boldsymbol{x})K(\boldsymbol{x},\boldsymbol{x}_o)d\boldsymbol{\theta}d\boldsymbol{x}}\tilde{p}(\boldsymbol{\theta},\boldsymbol{x})K(\boldsymbol{x},\boldsymbol{x}_o)}{\frac{1}{\iint \tilde{p}(\boldsymbol{\theta})p(\boldsymbol{x})K(\boldsymbol{x},\boldsymbol{x}_o)d\boldsymbol{\theta}d\boldsymbol{x}}\tilde{p}(\boldsymbol{\theta})\tilde{p}(\boldsymbol{x})K(\boldsymbol{x},\boldsymbol{x}_o)} \\
&= \frac{\iint \tilde{p}(\boldsymbol{\theta})\tilde{p}(\boldsymbol{x})K(\boldsymbol{x},\boldsymbol{x}_o)d\boldsymbol{\theta}d\boldsymbol{x}}{\iint \tilde{p}(\boldsymbol{\theta})p(\boldsymbol{x}|\boldsymbol{\theta})K(\boldsymbol{x},\boldsymbol{x}_o)d\boldsymbol{\theta}d\boldsymbol{x}}\frac{p(\boldsymbol{x}|\boldsymbol{\theta})}{\tilde{p}(\boldsymbol{x})} \\
&= \frac{\int \tilde{p}(\boldsymbol{x})K(\boldsymbol{x},\boldsymbol{x}_o)\int \tilde{p}(\boldsymbol{\theta})d\boldsymbol{\theta}d\boldsymbol{x}}{\int p(\boldsymbol{x}|\boldsymbol{\theta})K(\boldsymbol{x},\boldsymbol{x}_o)\int \tilde{p}(\boldsymbol{\theta})d\boldsymbol{\theta}d\boldsymbol{x}}\frac{p(\boldsymbol{x}|\boldsymbol{\theta})}{\tilde{p}(\boldsymbol{x})} \\
&= \frac{\mathbb{E}_{\boldsymbol{x}\sim\tilde{p}(\boldsymbol{x})}[K(\boldsymbol{x},\boldsymbol{x}_o)]}{\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x}|\boldsymbol{\theta})}[K(\boldsymbol{x},\boldsymbol{x}_o)]}\frac{p(\boldsymbol{x}|\boldsymbol{\theta})}{\tilde{p}(\boldsymbol{x})}
\end{aligned}$$

□

Lemma A.5. *The objective*

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{\theta},\boldsymbol{x}\sim\tilde{p}(\boldsymbol{\theta},\boldsymbol{x})}[(c_\zeta(\boldsymbol{\theta}) - K(\boldsymbol{x},\boldsymbol{x}_o))^2]$$

is minimized if and only if $c_\zeta(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x}|\boldsymbol{\theta})}[K(\boldsymbol{x},\boldsymbol{x}_o))]$ *for all* $\boldsymbol{\theta} \in$ *support*$(\tilde{p}(\boldsymbol{\theta}))$.

Proof.

$$\begin{aligned}\mathcal{L} &= \mathbb{E}_{\boldsymbol{\theta},\boldsymbol{x}\sim\tilde{p}(\boldsymbol{\theta},\boldsymbol{x})}[(c_\zeta(\boldsymbol{\theta}) - K(\boldsymbol{x},\boldsymbol{x}_o))^2] \\ &= \iint \tilde{p}(\boldsymbol{\theta},\boldsymbol{x})(c_\zeta(\boldsymbol{\theta}) - K(\boldsymbol{x},\boldsymbol{x}_o))^2 d\boldsymbol{x}d\boldsymbol{\theta} \\ &= \int \tilde{p}(\boldsymbol{\theta}) \int p(\boldsymbol{x}|\boldsymbol{\theta})(c_\zeta(\boldsymbol{\theta}) - K(\boldsymbol{x},\boldsymbol{x}_o))^2 d\boldsymbol{x}d\boldsymbol{\theta} \\ &= \int \tilde{p}(\boldsymbol{\theta})\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x}|\boldsymbol{\theta})}[(c_\zeta(\boldsymbol{\theta}) - K(\boldsymbol{x},\boldsymbol{x}_o))^2] d\boldsymbol{\theta}\end{aligned}$$

which is minimized if and only if $c_\zeta(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x}|\boldsymbol{\theta})}[K(\boldsymbol{x},\boldsymbol{x}_o))])$ for all $\boldsymbol{\theta} \in \text{support}(\tilde{p}(\boldsymbol{\theta}))$. □

A.5. Comparison to regression adjustment and ABC

Regression adjustment is typically performed after approximate Bayesian computation (ABC), to correct the bias induced by using non-zero tolerance ϵ. We will show that SNVI with calibration kernel $K(\boldsymbol{x},\boldsymbol{x}_o) = k_\epsilon(\boldsymbol{x}-\boldsymbol{x}_o)$ is less related to regression adjustment but can be related to ABC.

Assume we run rejection ABC with kernel k_ϵ, then as introduced in Subsec. 2.2.1 the obtained samples are from the approximate ABC posterior

$$p_\epsilon^{ABC}(\boldsymbol{\theta}|\boldsymbol{x}_o) \propto \mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x}|\boldsymbol{\theta})}[k_\epsilon(\boldsymbol{x}-\boldsymbol{x}_o)]p(\boldsymbol{\theta})$$

which implicitly use an approximate likelihood estimate $p_\epsilon^{ABC}(\boldsymbol{x}|\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x}_o|\boldsymbol{\theta})}[k_\epsilon(\boldsymbol{x}-\boldsymbol{x}_o)]$.

The goal of regression adjustment is to transform each sample $\boldsymbol{\theta}_i \sim p_\epsilon^{ABC}(\boldsymbol{\theta}|\boldsymbol{x}_o)$ to an 'adjusted' sample $\boldsymbol{\theta}_i' \sim p(\boldsymbol{\theta}|\boldsymbol{x}_o)$. Be u independent noise, the first step is to learn a stochastic map $\boldsymbol{\theta} = f_\phi(u,\boldsymbol{x})$ from $\boldsymbol{x}$ to $\boldsymbol{\theta}$ on all $(\boldsymbol{\theta},\boldsymbol{x})$ pairs for which $\boldsymbol{x}$ got accepted by the ABC criterion i.e. with probability proportional to $k_\epsilon(\boldsymbol{x}-\boldsymbol{x}_o)$.The second step is to adjust $\boldsymbol{\theta}_i$ by solving $\boldsymbol{\theta}_i = f_{\phi^*}(u_i,\boldsymbol{x}_i)$ for u_i i.e. $u_i = f_{\phi^*}^{-1}(\boldsymbol{\theta}_i,\boldsymbol{x}_i)$. If this is feasible, then we can generate adjusted samples from the exact posterior $\boldsymbol{\theta}_i' = f_{\phi^*}(u_i,\boldsymbol{x}_o)$ if f_{ϕ^*} does capture the exact stochastic mapping from $\boldsymbol{x}$ to $\boldsymbol{\theta}$. However, for efficient invertibility we typically have to restrict the functional form of f_ϕ and thus this will only hold approximately. Typical choice of f_ϕ are linear functions (Beaumont et al., 2002b) or non-linear affine transformations (Blum and François, 2010). Papamakarios (2019) pointed out that any form of SNPE is naturally related to regression adjustment as we directly model the relationship from $\boldsymbol{x}$ to $\boldsymbol{\theta}$. The effect is particularly evident if we use a normalizing flow as a conditional density estimator in SNPE. Then we explicitly learn an invertible transformation of noise which thus can be used for regression adjustment.

In contrast in SNVI we learn an stochastic map from $\boldsymbol{\theta}$ to $\boldsymbol{x}$ and thus cannot use it to adjust approximate posterior samples. Yet there is an interesting relation to the implicit

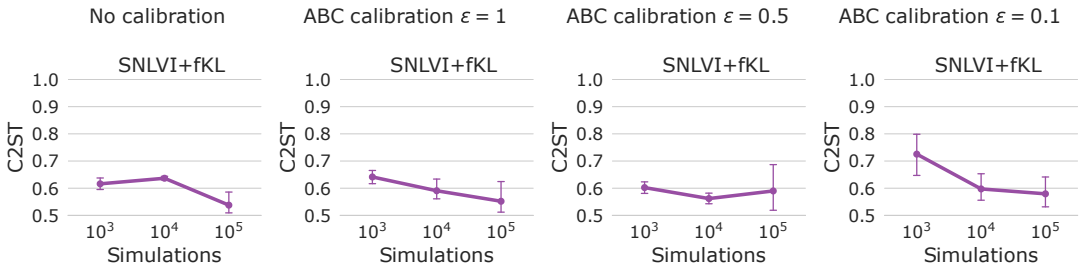

Figure A.3.: SNLVI results obtained on Two moons using the fKL estimator and an ABC calibration kernel $k_\epsilon = \mathbb{1}(||\boldsymbol{x} - \boldsymbol{x}_o||_2 \leq \epsilon)$.

likelihood assumed by any ABC algorithm. Recall that the calibration correction network estimates $c_{\zeta^*}(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x}|\boldsymbol{\theta})}[k_\epsilon(\boldsymbol{x} - \boldsymbol{x}_o)]$ and thus

$$p_\epsilon^{ABC}(\boldsymbol{\theta}|\boldsymbol{x}_o) \propto \mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x}|\boldsymbol{\theta})}[k_\epsilon(\boldsymbol{x} - \boldsymbol{x}_o)]p(\boldsymbol{\theta}) = c_{\zeta^*}(\boldsymbol{\theta})p(\boldsymbol{\theta})$$

The calibrated likelihood model $\ell_\psi(\boldsymbol{x}|\boldsymbol{\theta})$ learns $\ell_{\psi^*}(\boldsymbol{x}|\boldsymbol{\theta}) \propto k_\epsilon(\boldsymbol{x}_o - \boldsymbol{x})p(\boldsymbol{x}|\boldsymbol{\theta})$. Thus instead of adjusting samples, we can adjust the ABC posterior $p_\epsilon^{ABC}(\boldsymbol{\theta}|\boldsymbol{x}_o)$ towards the true posterior by multiplying $\ell_{\psi^*}(\boldsymbol{x}_o|\boldsymbol{\theta})$ as shown in Theorem A.1.

This relates our approach to Wilkinson (2014) method to accelerate ABC using a Gaussian process. He proposes to learn the ABC likelihood using a Gaussian process. This can be beneficial as one additionally obtain an uncertainty estimate. We choose a different model e.g. we estimate the exact same quantity using a neural network c_{ζ^*}. Similar to sequential neural methods this estimate can be used to rule out proportions of the input space which are unlikely to produce the observation (yet Wilkinson (2014) approach does not use posterior estimates to do this). We can then use MCMC to sample from the ABC-posterior distribution and thus can interpret Wilkinson (2014) approach as ABC equivalent to SNLE (Papamakarios et al., 2019). The corresponding SNVI version would instead obtain samples using variational inference. Whereas this approach can be significantly more efficient than naive ABC approaches, it does not correct for non-zero tolerance level ϵ within the ABC kernel k_ϵ.

Thus there is a dual interpretation for calibration adjustment using ABC kernels. We can either interpret c_{ζ^*} as correction factor for ℓ_{ψ^*} or ℓ_{ψ^*} as correction factor for c_{ζ^*}. Thereby ϵ mediates which parts of the likelihood are learned by ℓ_ψ and which parts are learned by c_ζ. As $\epsilon \to 0$, clearly $c_{\zeta^*}(\boldsymbol{\theta}) \to p(\boldsymbol{x}_o|\boldsymbol{\theta})$ and $\ell_{\psi^*}(\boldsymbol{x}|\boldsymbol{\theta}) \to \delta(\boldsymbol{x}_o - \boldsymbol{x})$ i.e. ℓ_ψ becomes increasingly irrelevant. Respectively as $\epsilon \to \infty$, clearly $c_{\zeta^*}(\boldsymbol{\theta}) \to K$ (where K is constant i.e. if k_ϵ is an indicator then $K = 1$) and $\ell_\psi(\boldsymbol{x}|\boldsymbol{\theta}) \to p(\boldsymbol{x}|\boldsymbol{\theta})$.

In Fig. A.3 we show results obtained for a classical ABC calibration kernel. It did improve performance in some cases, but not in all. Additionally, we obtain quite different results for different choices of ϵ. In fact, it may be better to adaptively choose ϵ and gradually decrease it. Within the above experiments, almost all samples are 'accepted' after the

first round. Further in such a scenario, we may also drop ℓ_ψ completely and instead infer the ABC-posterior. This may be beneficial in high dimension, as the architecture of c_ζ is unrestricted. Further investigation is left to future research.

A.6. SNPLA*

In Fig. 4.1 and Fig. 4.2, we compared SNVI to SNPLA (Wiqvist et al., 2021). To ensure comparability between SNPLA and SNVI, we implemented SNPLA ourselves and used the same likelihood- and posterior-model for both methods. The main difference between our implementation and the original implementation of SNPLA are:

1. We do not use the proposal $\hat{p}_r(\boldsymbol{\theta}) = \alpha p(\boldsymbol{\theta}) + (1-\alpha) q_\phi(\boldsymbol{\theta})$ for $\alpha \in [0, 1]$, instead we use $\alpha = 0$, i.e. we use the current posterior estimate as proposal.
2. Secondly, we use a Rational Linear Spline Flow (RSF) based on pyro (Bingham et al., 2019), whereas Wiqvist et al. (2021) uses a Masked Autoregressive Flow based on nflows (Durkan et al., 2019b).

Fig. A.4 compares the performance of our SNPLA implementation to the original implementation. Our implementation performs slightly better, likely due to the use of more expressive normalizing flows. We used our implementation for all experiments and nonetheless refer to the method with the name 'SNPLA'.

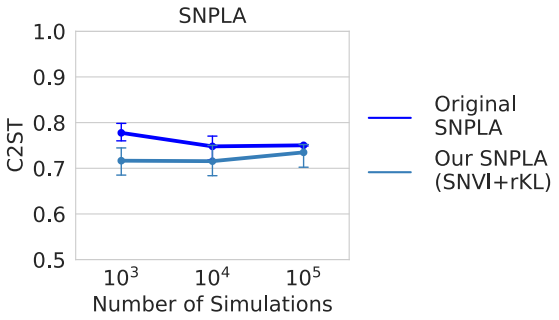

Figure A.4.: Comparison between SNPLA implementation of (Wiqvist et al., 2021) and SNVI with rKL.

A.7. Experiments: Benchmark*

All tasks were taken from an sbi benchmark (Lueckmann et al., 2021). For a description of the simulators, summary statistics, and prior distributions, we refer the reader to that paper.

*Parts of this section are currently under review for publication and hence revised by Michael Deistler and Jakob Macke (see `https://openreview.net/forum?id=kZOUYdhqkNY`).

We use the SNLE and SNRE as implemented in the sbi package (Tejero-Cantero et al., 2020). In all experiments, we learn the likelihood with a Masked Autoregressive Flow (MAF) with five autoregressive layers each with two hidden layers and 50 hidden units (Tejero-Cantero et al., 2020; Durkan et al., 2019b). For SNRE we use a two block residual network with 50 hidden units. Just as in Lueckmann et al. (2021), we implement SNRE with the loss described in Durkan et al. (2020a).

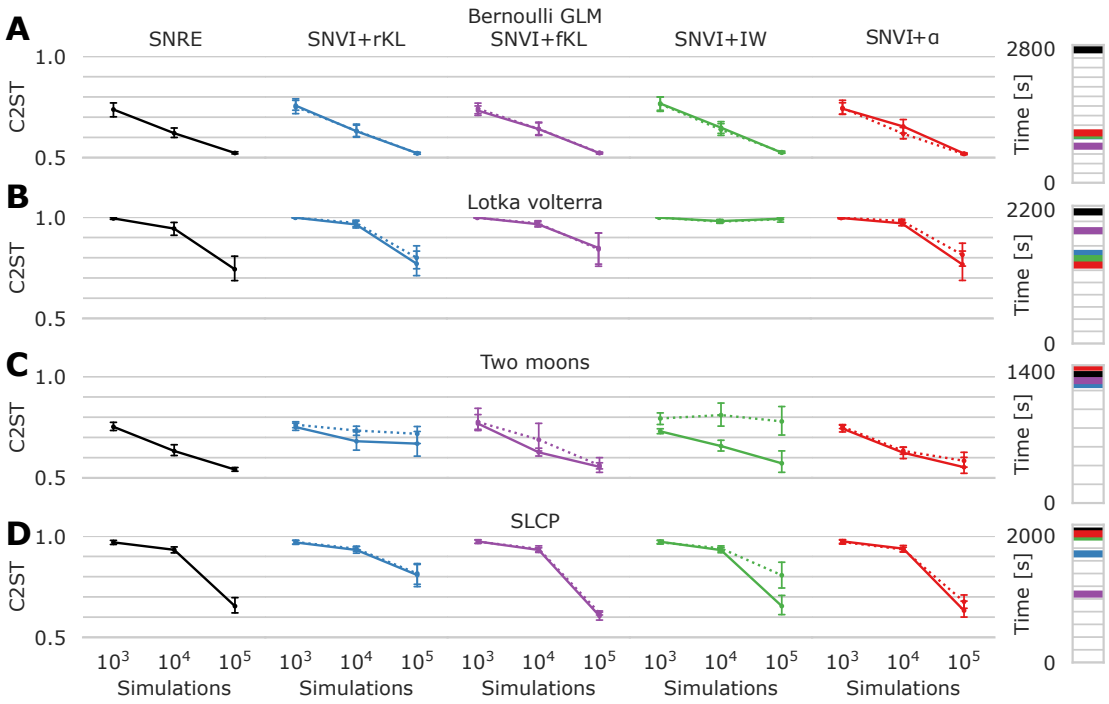

Figure A.5.: C2ST benchmark results for SNVI with ratio estimation (SNRVI) for four models, Bernoulli GLM (**A**), Lotka volterra (**B**), Two moons (**C**) and SLCP (**D**). Each point represents the average metric value for ten different observations, as well as the confidence intervals. Bars on the right indicate the average runtime. Two reference methods: SNRE with MCMC sampling and the rKL, as well as three variants of SNVI, with forward KL (SNVI+fKL), importance-weighted ELBO (SNVI+IW) and α-divergence (SNVI+α). Dotted lines: performance when not using SIR.

The implementation of the posterior normalizing flows is based on pyro (Bingham et al., 2019), as pyro caches intermediate values during sampling and thus allow cheap density evaluation on obtained samples. We use MAFs for higher dimensional problems and Rational Linear Spline Flows (RSF) for low dimensional but complex problems (SLCP, Two moons). We always use a standard Gaussian base distribution and five autoregressive layers with a hidden size depending on input dimension ($[\dim \cdot 10, \dim \cdot 10]$ for spline autoregressive nets and $[\dim \cdot 5 + 5]$ for affine autoregressive nets, each with ReLU activations). As the posterior support must match that of the prior, we add a bijective

mapping that maps the support to that of the prior. This allows training the normalizing flows directly on the constrained domain.

We used a total sampling budget of $N = 256$ for any VI loss. To estimate the IW-ELBO we use $N = 32$ to estimate $\mathcal{L}_{IW}^{(K=8)}(\phi)$ (Rainforth et al., 2018). Additionally, we use the STL estimator (Roeder et al., 2017). An alternative would be the doubly reparameterized gradient estimator, which is unbiased. We choose the STL estimator as it admits larger SNRs at the cost of introducing some bias (Tucker et al., 2018). Because for $\alpha \to 0$ we have that $\mathcal{L}_\alpha \to \mathcal{L}_{IW}^{(K=1)}$ we use this estimator also to estimate $\mathcal{L}_{\alpha=0.1}(\phi)$. While the estimator can also be used for the ELBO, it requires additional computational cost i.e. we additionally need to calculate the inverse transformation, which is costly for autoregressive flows. Note that the fKL estimator also requires the inverse transform, thus we recommend using a normalizing flow with fast forward and inverse passes in problems with many parameters, e.g. normalizing flows based on coupling layers (Dinh et al., 2017; Durkan et al., 2019a).

We trained for 10 rounds of simulations. In each round, we initialize the likelihood- and the posterior-model as their respective last estimates from the previous round. We train the posterior model for each round for at least 100 iterations and at most 1000 iterations. We evaluate convergence by tracking the decrease within the loss. For this automated benchmark, the convergence criteria are chosen conservative to avoid early stopping. More elaborate convergence criteria may improve runtime.

As metrics, we used classifier 2-sample tests (C2ST). C2ST trains a classifier to distinguish posterior samples produced by a specific method to ground truth posterior samples. Thus, a value of 0.5 means that the distributions are identical, whereas higher values indicate a mismatch between the distributions. As in Lueckmann et al. (2021), we computed the C2ST using 10,000 samples. Each figure shows the average metric value over 10 different observations, as well as the corresponding 95% confidence interval.

Fig. A.6 shows results for further variational objectives on the two moons (top) and on the SLCP task (bottom). The self-normalization used for the fKL estimator improves the approximation quality (A.6, left, dark vs light purple). For the IW-ELBO (middle) as well as for the α-divergences (right), the STL estimator improves performance (Rainforth et al., 2018). The gains from the STL are stronger for α-divergences than for the IW-ELBO (especially when using SIR). The STL particularly improves the estimate for low values of *alpha* (which are more support-covering). This is because they are also tighter evidence lower bounds.

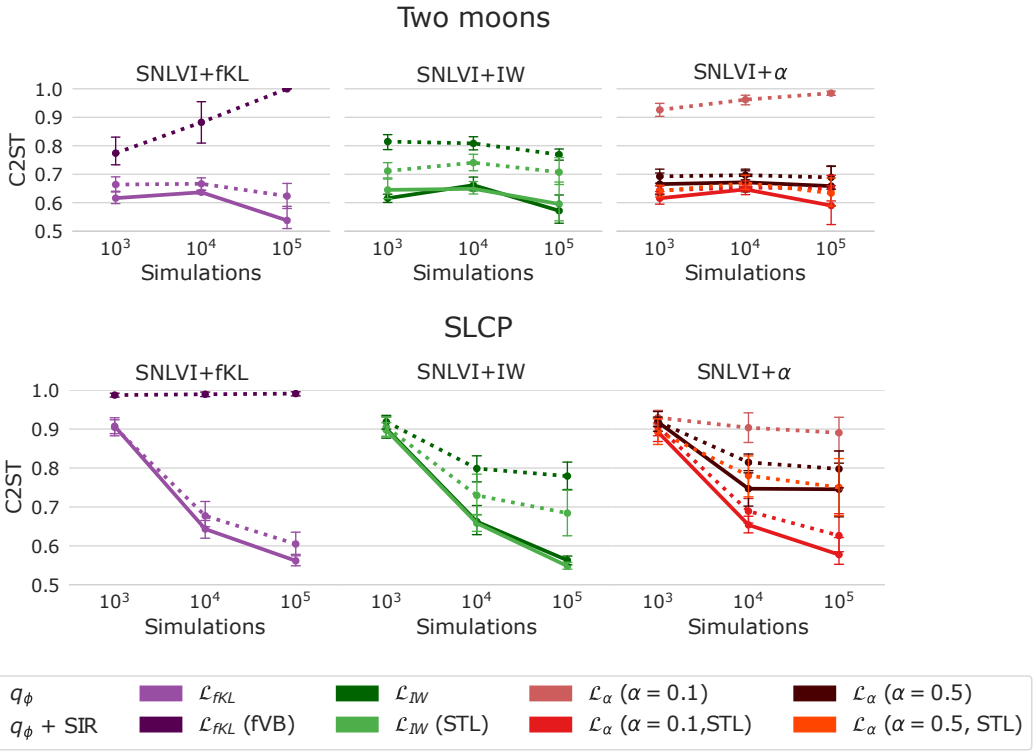

Figure A.6.: Evaluation of further variational objectives for the two moons (top) and the SLCP (bottom) task. Left: Variations of the forward-KL (with and without self-normalized weights). Middle: Variations of the importance-weighted objective (with and without STL). Right: Variations of the α-divergence (with and without STL as well as for different values of α.

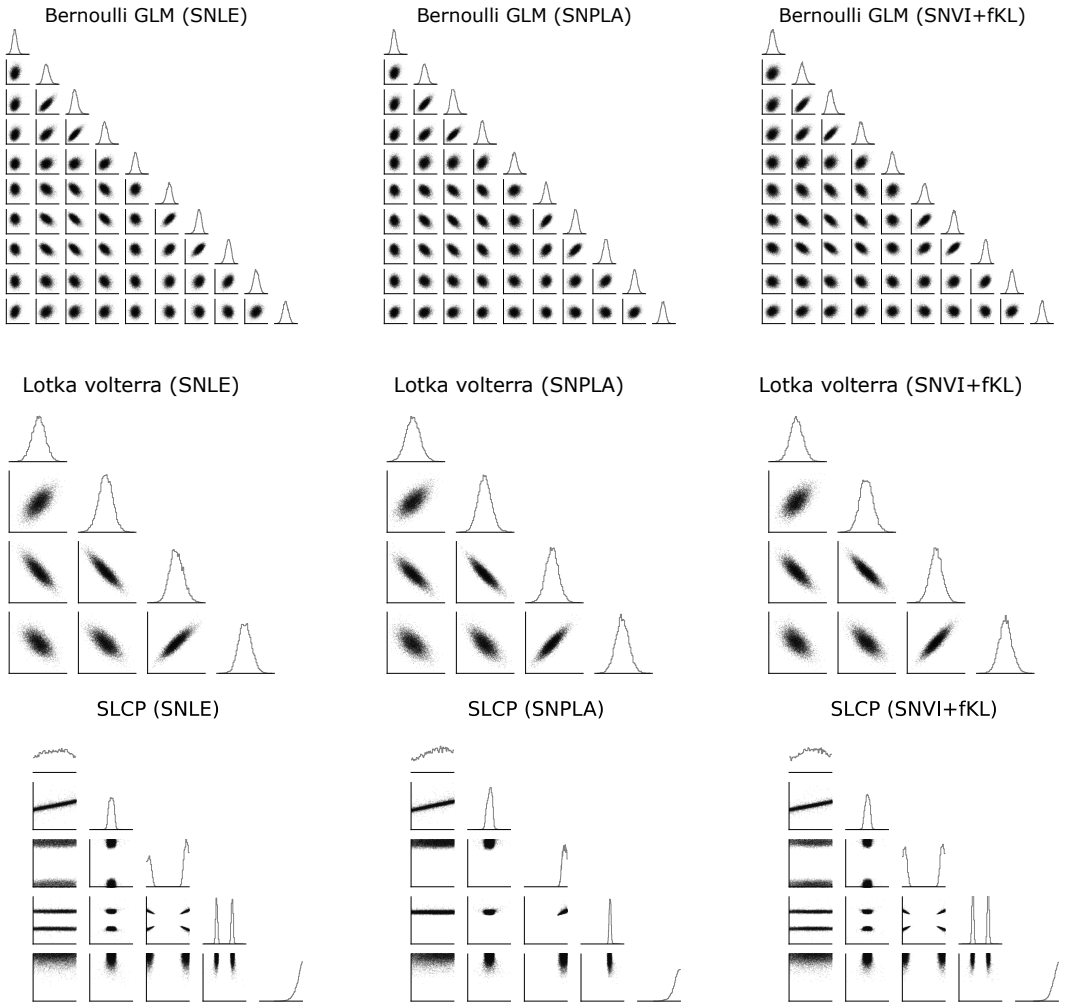

Figure A.7.: Samples from the posterior distributions for SNLE with MCMC, SNVI + fKL, SNVI + rKL. First row: results for SLCP. Second row: Lotka-Volterra. Third row: Bernoulli GLM.

A.8. Experiments: Alternative variational family

We investigated several variational objectives for SNVI. Yet, another major factor that can strongly affect the accuracy in VI is the variational family. We mainly focus on normalizing flows because of their flexibility to approximate most distributions very well. Another similarly flexible family are mixture distributions with parameters $\Phi = \{\pi_1, \dots, \pi_K, \phi_1, \dots, \phi_K\}$ given by

$$q_\Phi(\boldsymbol{\theta}) = \sum_{k=1}^{K} \pi_k q_{\phi_k}(\boldsymbol{\theta})$$

with $\sum_{k=1}^{K} \pi_k = 1$. Choosing $q_{\phi_k}(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}; \mu_k, \Sigma_K)$ leads to a Gaussian mixture (MoG), which is a universal density approximator given enough components (Nguyen, 2017). In general q_{ϕ_k} can be an arbitrary distribution. We also consider a mixture of normalizing flows (MoF), which can be highly expressive even with a few components (Pires and Figueiredo, 2020).

Problematically this family is less suitable for BBVI as e.g. the reparameterization trick cannot be applied directly. Alternatives as the 'score function' (also called REINFORCE) gradient estimate typically have larger variance (Mohamed et al., 2020; Morningstar et al., 2021), hence we consider two approaches for reparameterization.

Gradient estimate for reparameterizable components For BBVI we often need an estimate for $\nabla_\Phi \mathbb{E}_{\boldsymbol{\theta} \sim q_\Phi}[f(\boldsymbol{\theta})]$ e.g. if we minimize the KL divergence we have $f(\boldsymbol{\theta}) = \log(p(\boldsymbol{x}_o, \boldsymbol{\theta}) / \log q_\Phi(\boldsymbol{\theta}))$. Mixture distributions possess the property that any expectation can be written as follows:

$$\mathbb{E}_{\boldsymbol{\theta} \sim q_\Phi}[f(\boldsymbol{\theta})] = \int f(\boldsymbol{\theta}) \sum_{k=1}^{K} \pi_k q_{\phi_k}(\boldsymbol{\theta}) d\boldsymbol{\theta} = \sum_{k=1}^{K} \pi_k \mathbb{E}_{\boldsymbol{\theta} \sim q_{\phi_k}}[f(\boldsymbol{\theta})]$$

If the components q_{ϕ_k} are reparameterizable i.e. $\boldsymbol{\theta} \sim q_{\phi_k} \iff \boldsymbol{\theta} = T_{\phi_k}(\boldsymbol{\theta}_0)$ with $\boldsymbol{\theta}_0 \sim q_k(\boldsymbol{\theta}_0)$, then we can estimate the gradient of the full expectation, by using reparameterized samples from each of it's components i.e

$$\nabla_\Phi \mathbb{E}_{\boldsymbol{\theta} \sim q_\Phi}[f(\boldsymbol{\theta})] = \nabla_\Phi \sum_{k=1}^{K} \pi_k \mathbb{E}_{\boldsymbol{\theta}_0 \sim q_k}[f(T_{\phi_k}(\boldsymbol{\theta}_0))]$$

Morningstar et al. (2021) demonstrated that this approach can perform better than a 'score function' estimator. They further investigated the importance weighted ELBO, which showed a mass-covering property similar to our results using normalizing flows. Yet this approach scales badly with the number of mixture components. If we estimate any expectation with N samples we require $N \cdot K$ samples in total. Hence this approach is inefficient for mixture distributions with many components. Problematically if we for example use MoGs we require many components to be expressive.

Implicit reparamerization Direct parameterization is difficult for the whole mixture distribution since finding a suitable transformation is difficult. It is typically easier to find a standardizing function $\boldsymbol{\theta}_0 = S_\phi(\boldsymbol{\theta})$ that when applied remove the dependence on ϕ such that $\boldsymbol{\theta}_0 \sim q(\boldsymbol{\theta}_0)$. We can apply reparameterization if the inverse $T_\phi = S_\phi^{-1}$ is tractable. A standardizing function exists for a wide variety of continuous distributions i.e. the CDF of a univariate distribution maps samples from it to samples from a uniform distribution over $[0, 1]$. Whereas the CDF is often available in closed-form, inversion is often complicated or expensive.

Figurnov et al. (2018) proposed an alternative way to compute the reparameterization gradient which avoids inversion of S_ϕ. Be $\boldsymbol{\theta} = S_\phi^{-1}(\boldsymbol{\theta}_0)$, then the reparameterization gradient is given by

$$\nabla_\phi \mathbb{E}_{\boldsymbol{\theta} \sim q_\phi}[f(\boldsymbol{\theta})] = \mathbb{E}_{\boldsymbol{\theta}_0 \sim q_0}[\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}) \nabla_\phi \boldsymbol{\theta}] \text{ with } \nabla_\phi \boldsymbol{\theta} = \nabla_\phi S_\phi^{-1}(\boldsymbol{\theta}_0)$$

The key trick to avoid the inversion in $\nabla_\phi \boldsymbol{\theta}$ is to instead apply the total gradient to the equality $S_\phi(\boldsymbol{\theta}) = \boldsymbol{\theta}_0$. We obtain

$$\nabla_{\boldsymbol{\theta}} S_\phi(\boldsymbol{\theta}) \nabla_\phi \boldsymbol{\theta} + \nabla_\phi S_\phi(\boldsymbol{\theta}) = 0 \iff \nabla_\phi \boldsymbol{\theta} = -(\nabla_\theta S_\phi(\boldsymbol{\theta}))^{-1} \nabla_\phi S_\phi(\boldsymbol{\theta}).$$

This implicit reparameterization only requires differentiating S_ϕ, hence can be computed analytically or using automatic differentiation. However it requires solving a linear system of d equations were d is the dimension of $\boldsymbol{\theta}$.

Figurnov et al. (2018) also proposed a universal standardizing function for the multivariate case using conditional CDFs:

$$S_\phi(\boldsymbol{\theta}) = (F_\phi(\theta_1), F_\phi(\theta_2|\theta_1), \ldots, F_\phi(\theta_d|\theta_1, \ldots, \theta_{d-1})) = \boldsymbol{\theta}_0 \sim \text{Unif}(\boldsymbol{\theta}_0; 0, 1).$$

For mixture distributions the conditional CDFs can be obtained by

$$F_\Phi(\theta_d|\theta_1, \ldots \theta_{d-1}) = \sum_{k=1}^{K} \tilde{\pi}_k F_{\phi_k}(\theta_d|\theta_1, \ldots, \theta_{d-1}) \text{ with } \tilde{\pi}_k = \frac{\pi_k q_{\phi_k}(\theta_1, \ldots, \theta_d)}{\sum_{j=1}^{K} \pi_j q_{\phi_j}(\theta_1, \ldots, \theta_{d-1})}$$

making implicit reparameterization tractable for mixture models if the conditional CDFs and marginal densities for each component are tractable (Figurnov et al., 2018). If we assume the components factorize than this simplifies. Graves (2016) already derived gradients for Gaussian mixtures with diagonal covariance.

We show that S_ϕ can be efficiently computed if the component's of the mixture are defined through an invertible autoregressive transform of a simple factorized base density with tractable CDF. Examples contain all autoregressive normalizing flows without permutations (Papamakarios et al., 2017; Durkan et al., 2019a), but also the Gaussian distribution as $\boldsymbol{\theta} \sim \mathcal{N}(\boldsymbol{\theta}; \mu, \Sigma) \iff \boldsymbol{\theta} = L\boldsymbol{\theta}_0 + \mu$ with $\boldsymbol{\theta}_0 \sim \mathcal{N}(\boldsymbol{\theta}_0; 0, I)$ and L a lower triangular (thus 'autoregressive') matrix with $L^T L = \Sigma$ which exists as Σ is positive definite.

For simplicity lets just consider a single component, thus we will drop the index k. Be $\boldsymbol{\theta}^0 \sim \pi(\boldsymbol{\theta}^0) = \prod_{j=1}^{d} \pi(\theta_j^0)$ the base density. In this model a variable θ_j is obtained through an invertible transformation $\theta_j = T_{\phi_j}(\theta_j^0)$ and $\phi_j = f(\theta_{1:j-1})$.

This property ensures that we can easily obtain the conditional CDFs by inversion. We have that for some c the conditonal CDF is given by

$$F_j(c|\theta_{1:j-1}) = P(\theta_j \leq c|\theta_{1:j-1}) = P(\theta_j^0 \leq T_{\phi_j}^{-1}(c)|\theta_{1:j-1}) = F_j^0(T_{\phi_j}^{-1}(c))$$

where F_j^0 is the marginal CDF of the base distribution's marginal $\pi(\theta_j^0)$ which we assumed to be tractable. As the base distribution factorized we also can drop the conditioning on $\theta_{1:j-1}$. Consequently we can compute the vector of all conditional CDFs by simply inverting the full autoregressive transform to obtain $\boldsymbol{\theta}_0$, then compute all marginal CDFs of the base density.

We also require all marginal distributions $q(\theta_{1:j})$ to compute S_ϕ. By basic rules of probability we can compute them using all conditionals $q(\theta_{1:j}) = \prod_{i=1}^{j} q(\theta_i|\theta_{1:i-1})$. Be $J^{-1} = \frac{\partial T_\phi^{-1}}{\partial \boldsymbol{\theta}}$ the Jacobian matrix. Then J^{-1} is lower triangular as the transform is autoregressive and the absolute determinant is given by $|\det J|^{-1} = \prod_{i=1}^{d} |J_{ii}^{-1}|$ (Papamakarios et al., 2017; Papamakarios, 2019). By the change of variable theorem we can hence write

$$q(\boldsymbol{\theta}) = \pi(T_\phi^{-1}(\boldsymbol{\theta}))|\det J|^{-1} = \prod_{i=1}^{d} \pi(\theta_i^0)|J_{ii}^{-1}| = \prod_{i=1}^{d} q(\theta_i|\theta_{1:i-1}).$$

Thus we can compute the vector of all conditional distributions by the elementwise product of $\pi(\boldsymbol{\theta}_0)$ and the diagonal elements of J^{-1}. Each joint distributions $q(\theta_1, \ldots, \theta_j)$ can thus be expressed as a cumulative product of this vector up to variable θ_j.

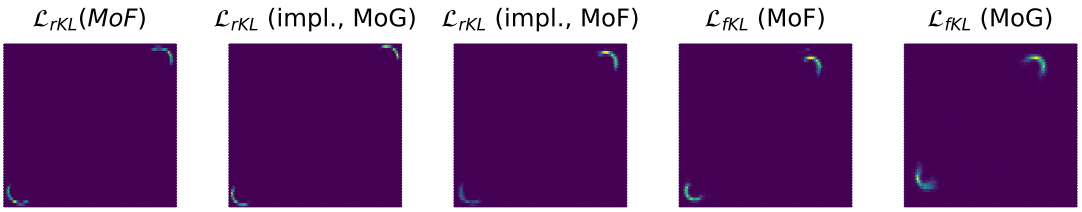

Figure A.8.: Two moons posterior distribution using mixture variational families.

Experiments: On multimodal targets, variational objectives different from the reverse KL divergence can provide better approximation quality using normalizing flows. Yet also the variational family itself can have a large influence on the resulting posterior.

We run SNVI with a mixture distribution variational family. Mixture distributions possess the favorable property that different components naturally cover different modes. For instance, consider a Gaussian mixture in which we initialize each component randomly.

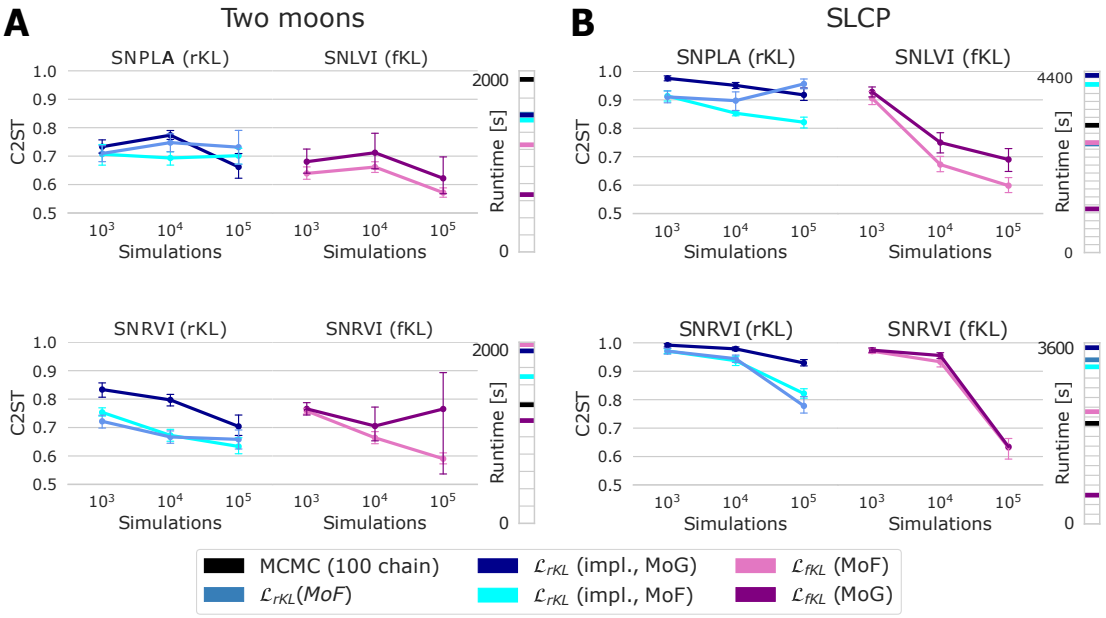

Figure A.9.: Results of SNLVI and SNRVI using a mixture variational family. We investigate mixture of spline autoregresive flows (MoF, 4 components) and mixture of Gaussians (MoG, 50 components). **A** shows results on two moons using likelihood and likelihood-ratio estimation. **B** shows results on slcp using likelihood and likelihood-ration estimation.

Similar to multi-chain MCMC, the different components will approach the nearest uncovered mode of the posterior. As a result, the variational posterior will typically cover all modes given enough components and a sufficiently diverse initialization. This is demonstrated in figure A.8, even for the mode-seeking reverse KL divergence the variational posterior covers both modes.

We compare a mixture of four two-layer spline autoregressive flows against a fifty-component Gaussian mixture distribution. The results for SNLVI and SNRVI are shown in figure A.9. We compare the reverse KL and the forward KL objectives, as well as the implicit and explicit reparameterization on the mixture of flows on the reverse KL divergence. For explicit reparameterization, we use 32 samples per component (i.e. in total 128), whereas for implicit reparameterization we use 128 in total.

When using the reveres KL divergence, we observe a small benefit in comparison to flows. This is because the corresponding variational posterior encompasses a broader range of modes. In SLCP with four modes, however, this effect is insufficient to cover all modes and would require more components (or a more sophisticated initialization). Even though implicit reparameterization is often better than the explicit version, it is inefficient when solving large dimensional problems (see SLCP). A major factor is that it requires the computation of the Jacobian of the standardizing function, which is currently done by

automatic differentiation, and thus requires d backward passes.

Consistent with our results on flows, the forward KL divergence outperforms the reverse KL. Additionally, it does not require any form of reparameterization and is thus typically more efficient. This is especially relevant using a mixture of Gaussians. The major computational bottleneck of the forward KL divergence is that it requires the inverse transform which is expensive for deep autoregressive normalizing flows (Papamakarios et al., 2017). Yet for Gaussian distributions, it is comparably cheap, which explains the big runtime improvements. In comparison to MoF's (or a single normalizing flow), they are less accurate.

We do jointly optimize all parameters of the mixture distribution. This approach is typically known to be less stable to optimize (Jerfel et al., 2021), which we also observed in our experiments. An alternative is known as variational boosting i.e. instead sequentially add new components (Miller et al., 2017; Jerfel et al., 2021). At the first glance, this may fit perfectly with SNVI i.e. adding each round a new component. Problematically, current implementations can only "remove" bad components, by removing all previously added components (as the parameter of all previous components are fixed). Yet in SNVI components added in early rounds are by construction less accurate (as the likelihood is less accurate). Investigation of more sophisticated methods for mixture distribution may be worth, but is not within the scope of this work.

A.9. Experiments: Inference in a neuroscience model of the pyloric network[¶]

We used the same simulator as in Gonçalves et al. (2020); Deistler et al. (2021) and the 15 summary statistics originally described in Prinz et al. (2004) and also used in Gonçalves et al. (2020); Deistler et al. (2021) (notably, Gonçalves et al. (2020); Deistler et al. (2021) used 3 additional features). Below, we describe the simulator briefly, for a full description we refer the reader to Prinz et al. (2004); Gonçalves et al. (2020); Deistler et al. (2021).

The model is composed of three single-compartment neurons, AB/PD, LP, and PY, where the electrically coupled AB and PD neurons are modeled as a single neuron. Each of the model neurons contains 8 currents. In addition, the model contains 7 synapses. As in Prinz et al. (2004), these synapses are simulated using a standard model of synaptic dynamics (Abbott and Marder, 1998).

For each set of membrane and synaptic conductances, we numerically simulate the circuit for 10 seconds with a step size of 0.025 ms. At each time step, each neuron receives Gaussian noise with mean zero and standard deviation 0.001 $\text{mV}\cdot\text{ms}^{-0.5}$.

We applied SNVI to infer the posterior over 24 membrane parameters and 7 synaptic parameters, i.e. 31 parameters in total. The 7 synaptic parameters are the maximal

[¶]Parts of this section are currently under review for publication and hence revised by Michael Deistler and Jakob Macke (see `https://openreview.net/forum?id=kZOUYdhqkNY`).

conductances of all synapses in the circuit, each of which is varied uniformly in the logarithmic domain and the membrane parameters are the maximal membrane conductances for each neuron. All membrane and synaptic conductances are varied over the same range as in Gonçalves et al. (2020); Deistler et al. (2021).

The 15 summary features proposed by Prinz et al. (2004) are salient features of the pyloric rhythm: Cycle period (s), three burst durations (s), two gap durations between bursts, two phase delays, three duty cycles, two phase gaps, and two phases of burst onsets. Note that several of these values are only defined if each neuron produces rhythmic bursting behavior. In particular, we call any simulations invalid if at least one of the summary features is undefined.

The experimental data is taken from file 845_082_0044 in a publicly available dataset (Haddad and Marder, 2021).

For the likelihood-model, we use a Neural Spline Flow (NSF) with five autoregressive layers. Each layer has two hidden layers and 50 hidden neurons, as implemented in the sbi package (Tejero-Cantero et al., 2020; Durkan et al., 2019b). The posterior model is a Masked autoregressive flow (MAF) with five autoregressive layers each with one hidden layer and 160 hidden units.

We train a total of 31 rounds. In the first round we use 50000 simulations from which only 492 are valid, and thus used to estimate the likelihood. For all other rounds we each simulated 10000 samples. To account for invalid summary features, we use the calibration kernel $K(\boldsymbol{x}, \boldsymbol{x}_o) = I(\boldsymbol{x} \text{ is valid})$, hence can simply exclude any invalid simulations from training the likelihood-model. By Theorem A.1 we have to correct the likelihood by multiplication of $\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x}|\boldsymbol{\theta})}[I(\boldsymbol{x} \text{ is valid})] = P(\boldsymbol{x} \text{ is valid}|\boldsymbol{\theta})$. To estimate this probability we use a deep logistic regression net with 3 hidden layers each with 50 neurons and ReLU activations. We train this classifier simultaneously with the likelihood-model, that is in each round we add new data $\{(\boldsymbol{\theta}_i, I(\boldsymbol{\theta}_i \text{ is valid}))\}_{i=1}^N$ and retrain the classifier using the weighted binary-cross-entropy loss. We weight the loss by the estimated class probabilities to account for class imbalance, especially in early rounds. We fix the number of epochs to 200 per round. We use the fKL loss with $N = 1024$ samples, as well as SIR.

Note that training an additional classifier concurrently is computationally cheap. We demonstrate this in Figure A.11, which becomes clear as training a simple feed-forward neural net is cheap compared to a normalizing flow. This is particularly useful for simulators that produce many invalid simulations, as these can be excluded from the training dataset of the likelihood estimator. We hence increase computational efficiency but not accuracy as invalid simulations are typically irrelevant to estimate the posterior of a valid observation.

In total, the procedure took 27 hours, with the runs of the simulator being parallelized across several nodes. Because of this, the runtime also depends greatly on the availability of computing resources on the cluster.

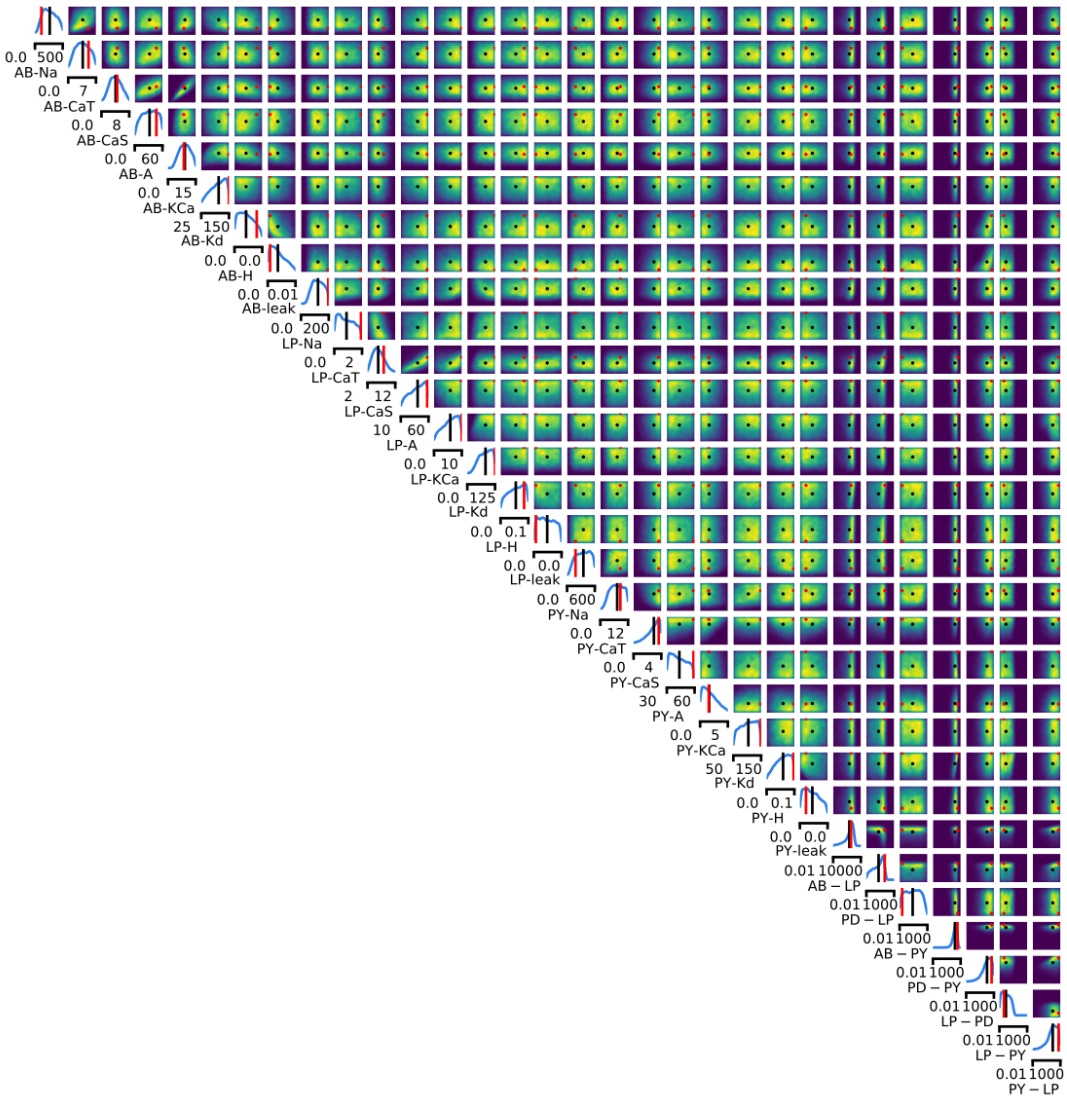

Figure A.10.: Posterior distribution for the neuroscience model of the pyloric network. In Fig. 4.3B we show a subset. The black point is a mean estimate using 10^7 samples. The red point is a maximum a-posterior estimate, obtained by gradient ascent.

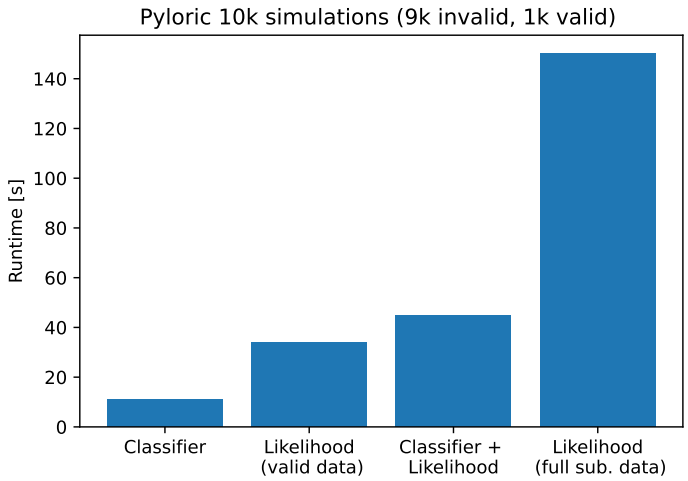

Figure A.11.: Runtime of the classifier $c_\zeta(\boldsymbol{\theta})$ in the model of the pyloric network (90% of simulations are invalid). Training the classifier is approximately three times cheaper than training the likelihood-model (compare left bar to the second left) and thus increases the computational cost only modestly. The likelihood-model is trained only on valid simulations. The combined runtime of classifier and likelihood-model (third bar) is still far less than the time it would take to train the likelihood-model on all simulations (right bar. To estimate the runtime of the likelihood-model on all simulations, we substituted invalid simulation outputs (i.e. NaN) with an unreasonably low value and trained on all simulations).