LEARNING IMPLICIT HIDDEN MARKOV MODELS USING NEURAL LIKELIHOOD-FREE INFERENCE

Anonymous authors
Paper under double-blind review

ABSTRACT

likelihood-free inference methods based on neural conditional density estimation were shown to drastically reduce the simulation burden in comparison to classical methods such as ABC. However, when applied in the context of any latent variable model, such as a Hidden Markov model (HMM), these methods are designed to only estimate the parameters rather than the joint posterior distribution of both the parameters and the hidden states. Naive application of these methods to a HMM, ignoring the inference of this joint posterior distribution, will result in overestimation of uncertainty of the posterior predictive. We propose a postprocessing step that can rectify this problem. Our approach relies on learning directly the intractable posterior distribution of the hidden states, using an autoregressive-flow, by exploiting the Markov property. Upon evaluating our approach on some intractable HMMs, we found that the quality of the estimates retrieved using our postprocessing is comparable to what can be achieved using a computationally expensive particle-filtering which additionally requires a tractable data distribution.

1 INTRODUCTION

We consider the task of Bayesian inference of a Hidden Markov model whose likelihood is analytically intractable and the model is only available as a simulator. Due to the unavailability of the likelihood standard Bayesian inference methods cannot be applied to such a model. Inference of such a model is generally carried out using approximate Bayesian computation (ABC) (Sisson et al., 2018), which only require forward simulations from the model, see for example Martin et al. (2019); Picchini (2014); Toni et al. (2009).

Recently, a new class of likelihood-free inference methods (Cranmer et al., 2020) were developed that use a neural network based emulator of the (unnormalised) posterior density, the likelihood density and the likelihood ratio. Such methods were empirically shown to be much more sample efficient in comparison to ABC. Additionally, these methods do not require the user to specify difficult-to-choose algorithmic parameters and they perform equally well across different models without (much) problem specific choice of the neural network's architecture. Naturally, these methods appear as more preferable algorithmic choices for carrying out inference in an intractable HMM, in comparison to ABC. We like to point out that these *neural likelihood-free* approaches (NLFI), by relying on emulation of an intractable density, are designed to estimate the posterior of the parameters only. However, in any HMM the unknown quantities include both the parameters as well as the hidden states, the latter being potentially high-dimensional. Estimation of the hidden states may or may not be of interest within a particular application domain. However, without estimating the joint posterior of the parameters and the hidden states the goodness-of-fit cannot be correctly assessed. This is indeed a severe limitation. Note that although ABC theoretically targets the joint distribution it fails to estimate (we will demonstrate this later in the experiments) the hidden states adequately within a reasonable simulation budget.

In this paper we present a novel technique to estimate the hidden states by learning an approximation of the incremental posterior distribution of the states using a neural density estimator. After learning the incremental posterior density, the density estimator can be used to draw the full path of the hidden states recursively. Following are our salient contributions:

- We highlight the problem of neural likelihood-free methods in estimating the joint density of the unknown variables within a HMM. We then propose a postprocessing technique to mitigate this limitation.
- Our approach can be used as a postprocessing technique for any likelihood-free method, in order to avoid overestimation of uncertainty.
- We develop a method to obtain an amortised approximation of the posterior of a latent Markov process, with intractable transition and observation densities, without using ABC.

2 BACKGROUND

We begin by first introducing the intractable HMM and then we will discuss the challenges of carrying out Bayesian inference. We can describe a HMM, for a latent Markov process $\boldsymbol{X}_t \in \mathbb{R}^K$ with a K-dimensional state-space, as follows:

$$\begin{aligned} \boldsymbol{X}_t &\sim f(\boldsymbol{X}_t|\boldsymbol{X}_{t-1}, \boldsymbol{\theta}) \\ \boldsymbol{y}_t &\sim g(\boldsymbol{y}_t|\boldsymbol{X}_t, \boldsymbol{\theta}) \end{aligned} \tag{1}$$

where $\boldsymbol{\theta}$ parameterise the transition $f(\boldsymbol{X}_t|\boldsymbol{X}_{t-1}, \boldsymbol{\theta})$ and the observation $g(\boldsymbol{y}_t|\boldsymbol{X}_t, \boldsymbol{\theta})$ densities respectively. Given a set of noisy observations of L out of the K states $\boldsymbol{y} \in \mathbb{R}^{M\times L}$ at M experimental time points, $\{t_i\}_{i=0}^{M-1}$, of the latent process, our goal is to infer the joint posterior distribution $p(\boldsymbol{\theta}, \boldsymbol{x}|\boldsymbol{y})$, where $\boldsymbol{x} = (\boldsymbol{X}_{t_0}, \ldots, \boldsymbol{X}_{t_{M-1}})$ is the unobserved sample path of the process – the hidden states. The expression for the unnormalised posterior is given by

$$p(\boldsymbol{\theta}, \boldsymbol{x}|\boldsymbol{y}) \propto p(\boldsymbol{\theta})p(\boldsymbol{X}_{t_0})\left(\prod_{i=0}^{M-1} g(\boldsymbol{y}_{t_i}|\boldsymbol{X}_{t_i}, \boldsymbol{\theta})\right)\left(\prod_{i=1}^{M-1} f(\boldsymbol{X}_{t_i}|\boldsymbol{X}_{t_{i-1}}, \boldsymbol{\theta})\right), \tag{2}$$

where $p(\boldsymbol{\theta})$ and $p(\boldsymbol{X}_{t_0})$ are the prior distributions over the parameters and the initial values. The trickier part is the inference of the hidden states. We are interested in the case where either $f(\cdot)$ or $g(\cdot)$ or both is unavailable analytically. Note that indeed when these densities are known we can apply classical state-space techniques (Särkkä, 2013) such as Kalman filtering, for linear-Gaussian models, extended/unscented Kalman filtering for nonlinear ones, and particle-filtering for nonlinear non-Gaussian ones. However, when these densities are analytically intractable then we have to resort to likelihood-free methods.

3 NEURAL LIKELIHOOD-FREE INFERENCE

If instead of the joint $p(\boldsymbol{\theta}, \boldsymbol{x}|\boldsymbol{y})$ we only wish to estimate the marginal $p(\boldsymbol{\theta}|\boldsymbol{y})$, then a number of strategies based on conditional density estimation can be employed. For example, we can simulate pairs of $\boldsymbol{\theta}, \boldsymbol{y}$ from their joint distribution and then subsequently create a training dataset, of N samples $\{\boldsymbol{\theta}^n, \boldsymbol{y}^n\}_{n=1}^N$, which can be utilised to train a conditional density estimator that can approximate the posterior (Papamakarios & Murray, 2016) $p(\boldsymbol{\theta}|\boldsymbol{y}) \approx q_{\boldsymbol{\psi}}(\boldsymbol{\theta}|\boldsymbol{y})$ or the likelihood Papamakarios et al. (2019) $p(\boldsymbol{y}|\boldsymbol{\theta}) \approx q_{\boldsymbol{\psi}}(\boldsymbol{y}|\boldsymbol{\theta})$. In the former case once we have trained an approximation to the posterior, using a density estimator, we can directly draw samples $\boldsymbol{\theta} \sim q_{\boldsymbol{\psi}}(\boldsymbol{\theta}|\boldsymbol{y}_o)$ by conditioning on a particular dataset $\boldsymbol{y}_o$. In the latter case we can use the trained density estimator to approximate the unnormalised posterior $p(\boldsymbol{\theta}|\boldsymbol{y}) \propto q_{\boldsymbol{\phi}}(\boldsymbol{y}_o|\boldsymbol{\theta})p(\boldsymbol{\theta})$ and then draw samples from it using Markov chain Monte Carlo (MCMC). In principle we can choose any density estimator for $q_{\boldsymbol{\psi}}(\cdot|\cdot)$. However, in the literature surrounding NLFI, a neural network based density estimator is generally used. A neural network is used in this context either as a nonlinear transformation of the conditioning variables, within a mixture-of-Gaussian density as was proposed in Bishop (1994), or as a *normalizing-flow* (Rezende & Mohamed, 2015; Papamakarios et al., 2021) that builds a transport map (Parno, 2015) between a simple distribution (such as a standard Gaussian) and a complex one such as the likelihood/posterior density. Following the seminal work of Tabak & Turner (2013) a large amount of research is undertaken to build such transport maps using samples from the respective measures. An alternative formulation of NLFI utilises the duality (Cranmer et al., 2015) between the optimal decision function of a probabilistic classifier and the likelihood ratio, $r(\boldsymbol{\theta}^a, \boldsymbol{\theta}^b) = \frac{p(\boldsymbol{y}|\boldsymbol{\theta}^a)}{p(\boldsymbol{y}|\boldsymbol{\theta}^b)}$ evaluated using two samples $\boldsymbol{\theta}^a$ and $\boldsymbol{\theta}^b$, to approximate the latter through training a binary classifier using samples from $p(\boldsymbol{y}, \boldsymbol{\theta})$.

This likelihood ratio can then be used as proxy within a MCMC scheme as follows:

$$\min\left\{1, \frac{p(\boldsymbol{y}_o|\boldsymbol{\theta}^*)k_{\boldsymbol{\theta}}(\boldsymbol{\theta}|\boldsymbol{\theta}^*)p(\boldsymbol{\theta}^*)}{p(\boldsymbol{y}_o|\boldsymbol{\theta})k_{\boldsymbol{\theta}}(\boldsymbol{\theta}^*|\boldsymbol{\theta})p(\boldsymbol{\theta})}\right\} \approx \min\left\{1, r(\boldsymbol{\theta}^*, \boldsymbol{\theta})\frac{k_{\boldsymbol{\theta}}(\boldsymbol{\theta}|\boldsymbol{\theta}^*)p(\boldsymbol{\theta}^*)}{k_{\boldsymbol{\theta}}(\boldsymbol{\theta}^*|\boldsymbol{\theta})p(\boldsymbol{\theta})}\right\}, \tag{3}$$

where $k_{\boldsymbol{\theta}}(\cdot)$ is a proposal density. Note that these NLFI methods carry out *amortised inference* that is there is no need to re-learn the density/density-ratio estimator for every new instances of the observations.

To increase sample efficiency of these methods one can use them in a sequential manner (Durkan et al., 2018). After an initial round of NLFI, we are left with samples of $\boldsymbol{\theta}$ from its posterior distribution. We can subsequently use these samples to generate further simulated data concentrated around the given observations $\boldsymbol{y}_o$. This constitute a new training set on which a second round of NLFI can be applied to further refine the approximations. This process can be repeated for a number of rounds. Note that when a sequential process is used in conjunction with a density estimator for the posterior then the parameter samples from the second round are no longer drawn from the prior. Thus, different adjustments had been proposed, leading to different algorithms (Greenberg et al., 2019; Lueckmann et al., 2017), to overcome this issue.

We like to further point out that NLFI is applied using some summary statistic $s(\boldsymbol{y})$ of the data $\boldsymbol{y}$, a practise carried over from the usage of ABC methods. There have been recent work on using a neural network to generate the summary, trained simultaneously with the network used for density/density-ratio estimation.

Can we ignore the joint distribution? The various NLFI techniques discussed so far are designed to solve the marginal problem of estimating $p(\boldsymbol{\theta}|\boldsymbol{y})$. The necessity of the estimation of the hidden states $\boldsymbol{x}$ is problem specific. In some applications estimation of the hidden states is of paramount importance. For example, in epidemiology the hidden states reflect how an infectious disease has progressed and thus inference of both the parameters and the hidden states is necessary. However, in certain applications one may wish to ignore the hidden states. Nonetheless, it is helpful if the methodology can accommodate estimation of the joint distribution $p(\boldsymbol{\theta}, \boldsymbol{x}|\boldsymbol{y})$, and the choice of estimating just $\boldsymbol{\theta}$ or $\boldsymbol{x}$ or both is left to the practitioner's discretion. However, in the process of modelling a physical phenomenon it is often necessary to asses the goodness-of-fit. Within the Bayesian context this is carried out by inspection of the posterior predictive distribution $p(\boldsymbol{y}^*|\boldsymbol{y})$ given by

$$p(\boldsymbol{y}^*|\boldsymbol{y}) = \int p(\boldsymbol{y}^*|\boldsymbol{x}, \boldsymbol{\theta})p(\boldsymbol{x}, \boldsymbol{\theta}|\boldsymbol{y})d\boldsymbol{x}d\boldsymbol{\theta}. \tag{4}$$

But when instead of the joint distribution we only have access to the marginal distribution (that is access to only samples of $\boldsymbol{\theta}$, the output of any NLFI method) then the above can only be obtained as follows

$$p(\boldsymbol{y}^*|\boldsymbol{y}) \approx \hat{p}(\boldsymbol{y}^*|\boldsymbol{y}) = \int p(\boldsymbol{y}^*|\boldsymbol{x}, \boldsymbol{\theta})p(\boldsymbol{x}|\boldsymbol{\theta})p(\boldsymbol{\theta}|\boldsymbol{y})d\boldsymbol{x}d\boldsymbol{\theta}, \tag{5}$$

where the joint posterior of $\boldsymbol{x}, \boldsymbol{\theta}$ is approximated as $p(\boldsymbol{x}, \boldsymbol{\theta}|\boldsymbol{y}) \approx p(\boldsymbol{x}|\boldsymbol{\theta})p(\boldsymbol{\theta}|\boldsymbol{y})$, which is akin to drawing $\boldsymbol{x}$ from the prior $p(\boldsymbol{x}|\boldsymbol{\theta}) = p(\boldsymbol{X}_{t_0})\prod_{i=1}^{M-1} f(\boldsymbol{X}_{t_i}|\boldsymbol{X}_{t_{i-1}}, \boldsymbol{\theta})$, of the latent Markov process. As a result the credible intervals of $\hat{p}(\boldsymbol{y}^*|\boldsymbol{y})$ would be erroneously inflated, since in this case the latent sample path $\boldsymbol{x}$ is not correctly constrained by the data, leading to an incorrect assessment of the goodness-of-fit. This is a severe problem that needs to be addressed even in the case where we wish to estimate just the marginal $p(\boldsymbol{\theta}|\boldsymbol{y})$.

limitations of NFLI for inferring the joint: let us now consider the task of estimating the joint posterior $p(\boldsymbol{\theta}, \boldsymbol{x}|\boldsymbol{y})$ using a NLFI method. If we want to approximate the posterior then we have to extend any chosen density estimator to target a high-dimensional vector $(\boldsymbol{\theta}, \mathrm{vec}(\boldsymbol{x}))$, where $\mathrm{vec} : \mathbb{R}^{K\times M} \to \mathbb{R}^{KM}$, which would invariably require a larger training set, and thus more simulations, in comparison to estimating only $\boldsymbol{\theta}$ (see Appendix E for an example). Alternatively, if we choose to approximate the likelihood density, then note that the acceptance ratio of the MCMC step will be of the following form:

$$\min\left\{1, \frac{q_{\boldsymbol{\psi}}(\boldsymbol{y}_o|\boldsymbol{x}^*, \boldsymbol{\theta}^*)p(\boldsymbol{x}^*, \boldsymbol{\theta}^*)k_{\boldsymbol{x}}(\boldsymbol{x}|\boldsymbol{x}^*)k_{\boldsymbol{\theta}}(\boldsymbol{\theta}|\boldsymbol{\theta}^*)}{q_{\boldsymbol{\psi}}(\boldsymbol{y}_o|\boldsymbol{x}\cdot\boldsymbol{\theta})p(\boldsymbol{x}, \boldsymbol{\theta})k_{\boldsymbol{x}}(\boldsymbol{x}^*|\boldsymbol{x})k_{\boldsymbol{\theta}}(\boldsymbol{\theta}^*|\boldsymbol{\theta})}\right\}, \tag{6}$$

where $k_{\boldsymbol{x}}(\cdot)$, $k_{\boldsymbol{\theta}}(\cdot)$ are the proposal densities. Due to the intractability of $p(\boldsymbol{x}, \boldsymbol{\theta})$ our only option as a proposal $k_{\boldsymbol{x}}(\cdot)$ is its prior that is the transition density in equation 1. This will jeopardise the mixing of the MCMC sampler which in turn would require excessive simulation from the model. We would face the same limitation if we had chosen to emulate the likelihood ratio.

We like to point out that ABC indeed jointly samples the hidden states and the parameters. However, the states are updated using the prior of the Markov process as the proposal (see Appendix A for details) resulting in a poor estimation of $\boldsymbol{x}$. In what follows we develop a novel technique to estimate $\boldsymbol{x}$ given samples of $\boldsymbol{\theta}$.

4 METHODS

4.1 INFERRING BOTH THE PARAMETERS AND HIDDEN STATES

We can decompose the joint density using the product rule as follows:

$$p(\boldsymbol{x}, \boldsymbol{\theta}|\boldsymbol{y}) = p(\boldsymbol{x}|\boldsymbol{\theta}, \boldsymbol{y})p(\boldsymbol{\theta}|\boldsymbol{y}). \tag{7}$$

With the above decomposition we can break down the task of inferring the joint distribution of $\boldsymbol{x}, \boldsymbol{\theta}$ into two sub-tasks of inferring separately the distributions $p(\boldsymbol{\theta}|\boldsymbol{y})$ and $p(\boldsymbol{x}|\boldsymbol{\theta}, \boldsymbol{y})$. Samples of $\boldsymbol{x}$ can then be drawn given samples of $\boldsymbol{\theta}$. Note that the task of inferring $p(\boldsymbol{\theta}|\boldsymbol{y})$ can be carried out, sample efficiently, using any off-the-shelf NLFI method. Thus, the challenge is to develop a NLFI strategy for inferring $p(\boldsymbol{x}|\boldsymbol{\theta}, \boldsymbol{y})$. The advantage of applying NLFI to infer $p(\boldsymbol{x}|\boldsymbol{\theta}, \boldsymbol{y})$ rather than $p(\boldsymbol{x}, \boldsymbol{\theta}|\boldsymbol{y})$ is that for the former case we can exploit the Markov property to escape the curse-of-dimensionality.

This is a non-standard approach of drawing from the joint distribution, simply because there did not exist any method, before the advent of NLFI, which can target $p(\boldsymbol{\theta}|\boldsymbol{y})$ without drawing $\boldsymbol{x}$. Any classical pseudo-marginal sampling method, targeting $\boldsymbol{\theta}$ (Beaumont, 2003; Andrieu & Roberts, 2009), that uses the marginal likelihood $p(\boldsymbol{y}|\boldsymbol{\theta})$, or its unbiased estimate, in the MCMC acceptance step invariably also draws $\boldsymbol{x}$.

Note that if the observation density is known analytically, then a standard Bootstrap (Gordon et al., 1995) SMC algorithm can be used, since it does not require a tractable transition density, to infer $p(\boldsymbol{x}|\boldsymbol{\theta}, \boldsymbol{y}) \approx p_{smc}(\boldsymbol{x}|\boldsymbol{y}, \boldsymbol{\theta})$, given an estimate of $p(\boldsymbol{\theta}|\boldsymbol{y})$ obtained using NLFI. The posterior predictive distribution can then be evaluated as follows:

$$\begin{aligned} p(\boldsymbol{y}^*|\boldsymbol{y}) &= \int p(\boldsymbol{y}^*|\boldsymbol{x}, \boldsymbol{\theta})p(\boldsymbol{x}, \boldsymbol{\theta}|\boldsymbol{y})d\boldsymbol{x}d\boldsymbol{\theta} \\ &\approx \int p(\boldsymbol{y}^*|\boldsymbol{x}, \boldsymbol{\theta})p_{smc}(\boldsymbol{x}|\boldsymbol{y}, \boldsymbol{\theta})p(\boldsymbol{\theta}|\boldsymbol{y})d\boldsymbol{x}d\boldsymbol{\theta}. \end{aligned} \tag{8}$$

Although in certain practical applications the observation density remains tractable, the SMC algorithm however would require a large number of simulations from the model, thus defeating the purpose of NLFI. This is the reason we consider learning an approximation of $p(\boldsymbol{x}|\boldsymbol{\theta}, \boldsymbol{y})$ using a neural density estimator, which we turn to next.

4.2 LEARNING AN INCREMENTAL POSTERIOR

We can decompose the posterior of $\boldsymbol{x}$, again using the product rule, as follows:

$$p(\boldsymbol{x}|\boldsymbol{\theta}, \boldsymbol{y}) = p(\boldsymbol{X}_{t_{M-1}}|\boldsymbol{X}_{t_{M-2}}, \ldots, \boldsymbol{X}_{t_0}, \boldsymbol{\theta}, \boldsymbol{y})p(\boldsymbol{X}_{t_{M-2}}, \ldots, \boldsymbol{X}_{t_0}|\boldsymbol{\theta}, \boldsymbol{y}). \tag{9}$$

It can be easily verified (see Appendix A) that the posterior distribution of the last sample point $\boldsymbol{X}_{t_{M-1}}$, due to the Markov property, is independent of the past history:

$$p(\boldsymbol{X}_{t_{M-1}}|\boldsymbol{X}_{t_{M-2}}, \ldots, \boldsymbol{X}_{t_0}, \boldsymbol{\theta}, \boldsymbol{y}) = p(\boldsymbol{X}_{t_{M-1}}|\boldsymbol{X}_{t_{M-2}}, \boldsymbol{y}_{t_{M-1}}, \boldsymbol{\theta}). \tag{10}$$

For each of the remaining sample points $\{\boldsymbol{X}_{t_k}\}_{k=1}^{M-2}$ if we drop the conditioning of future datapoints $\{\boldsymbol{y}_{t_l}\}_{l>k}$, then, using the above property, equation 9 can be approximated as follows:

$$p(\boldsymbol{x}|\boldsymbol{\theta}, \boldsymbol{y}) \approx \prod_{i=1}^{M-1} p(\boldsymbol{X}_{t_i}|\boldsymbol{X}_{t_{i-1}}, \boldsymbol{y}_{t_i}, \boldsymbol{\theta}). \tag{11}$$

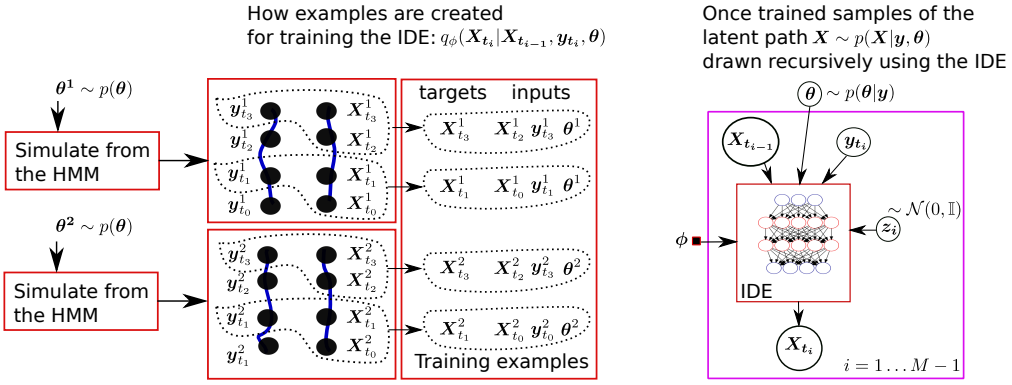


Figure 1: (**Left**) Process of creation of a dataset for training the IDE: $q_{\boldsymbol{\phi}}(\boldsymbol{X}_{t_i}|\boldsymbol{X}_{t_{i-1}}, \boldsymbol{y}_{t_i}, \boldsymbol{\theta})$. We demonstrate this using $M = 4$ time points. (**Right**) Graphical model of the IDE showing how the sample path of the hidden states $\boldsymbol{x}$ is drawn. Note that $p(\boldsymbol{\theta}|\boldsymbol{y})$ can be obtained using any off-the-shelf likelihood-free method. We build the IDE using a masked autoregressive flow.

We can now emulate the incremental posterior $p(\boldsymbol{X}_{t_i}|\boldsymbol{X}_{t_{i-1}}, \boldsymbol{y}_{t_i}, \boldsymbol{\theta}) \approx q_{\boldsymbol{\phi}}(\boldsymbol{X}_{t_i}|\boldsymbol{X}_{t_{i-1}}, \boldsymbol{y}_{t_i}, \boldsymbol{\theta})$, between any two consecutive time points t_i, t_{i-1}, using a neural density estimator $q_{\boldsymbol{\phi}}(\cdot)$. We can then recursively sample the full path using the corresponding incremental density estimator (IDE) $q_{\boldsymbol{\phi}}(\boldsymbol{X}_{t_i}|\boldsymbol{X}_{t_{i-1}}, \boldsymbol{y}_{t_i}, \boldsymbol{\theta})$ (see Figure 1). We can approximate the posterior predictive in equation 4 now as follows:

$$p(\boldsymbol{y}^*|\boldsymbol{y}) \approx \int p(\boldsymbol{y}^*|\boldsymbol{x}, \boldsymbol{\theta}) \prod_{i=1}^{M-1} q_{\boldsymbol{\phi}}(\boldsymbol{X}_{t_i}|\boldsymbol{X}_{t_{i-1}}, \boldsymbol{y}_{t_i}, \boldsymbol{\theta}) p(\boldsymbol{\theta}|\boldsymbol{y}) d\boldsymbol{x} d\boldsymbol{\theta}. \tag{12}$$

In summary our strategy for drawing samples $\boldsymbol{x}, \boldsymbol{\theta}$ from an approximation of their joint posterior is as follows. We first infer the parameter marginal $p(\boldsymbol{\theta}|\boldsymbol{y})$ using any chosen NLFI method and draw samples of $\boldsymbol{\theta}$. In parallel we train an IDE using a subset of simulations used in inferring $p(\boldsymbol{\theta}|\boldsymbol{y})$. Then for each sample $\boldsymbol{\theta}$, and an observed dataset $\boldsymbol{y}_o$, we can use the trained IDE recursively to obtain the sample path $\boldsymbol{x}$ conditioned on the sample $\boldsymbol{\theta}$ and the dataset $\boldsymbol{y}_o$.

Limitations: There are two fundamental assumptions behind our approach. Firstly, we assume that samples of $\boldsymbol{\theta}$ are drawn from a good approximation to the true unknown posterior. However, this may not be true when inference of $\boldsymbol{\theta}$ is done on a very limited simulation budget. Secondly, we assume that there is no model miss-specification and the observations come from the joint distribution $p(\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{y})$ used for learning the IDE. This can be a strong assumption when modelling a new phenomenon.

4.3 Training the incremental density estimator

We chose a *masked autoregressive flow* (MAF) as the incremental density estimator. MAF is built upon the idea of chaining together a series of autoregressive functions, and can be interpreted as a normalizing-flow (Papamakarios et al., 2017). That is we can represent $q_{\boldsymbol{\phi}}(\boldsymbol{X}_{t_i}|\boldsymbol{X}_{t_{i-1}}, \boldsymbol{y}_{t_i}, \boldsymbol{\theta})$ as a transformation of a standard Gaussian $\mathcal{N}(\mathbf{0}, \mathbb{I})$ (or another simple distribution) through a series of J autoregressive functions $h_1, \dots, h_J$, parameterised by $\boldsymbol{\phi}$, each of which is dependent on the triplet $(\boldsymbol{X}_{t_{i-1}}, \boldsymbol{y}_{t_i}, \boldsymbol{\theta})$:

$$\boldsymbol{X}_{t_i} = \boldsymbol{z}_J, \quad \text{where} \quad \begin{array}{l} \boldsymbol{z}_0 = \mathcal{N}(\mathbf{0}, \mathbb{I}) \\ \boldsymbol{z}_j = h_j(\boldsymbol{z}_{j-1}, \boldsymbol{X}_{t_{i-1}}, \boldsymbol{y}_{t_i}, \boldsymbol{\theta}) \end{array} . \tag{13}$$

Each h_j is a bijection with a lower-triangular Jacobian matrix, implemented by a *Masked Autoencoder for Distribution Estimation* (MADE) (Germain et al., 2015), and is conditioned on $(\boldsymbol{X}_{t_{i-1}}, \boldsymbol{y}_{t_i}, \boldsymbol{\theta})$. Using the formula for change of variable the density is given by

$$q_{\boldsymbol{\phi}}(\boldsymbol{X}_{t_i}|\boldsymbol{X}_{t_{i-1}}, \boldsymbol{y}_{t_i}, \boldsymbol{\theta}) = \mathcal{N}(\mathbf{0}, \mathbb{I}) \prod_{j=1}^{J} \left| \det \left(\frac{\partial h_j}{\partial \boldsymbol{z}_{j-1}} \right) \right|^{-1} . \tag{14}$$

We can learn the parameters $\boldsymbol{\phi}$ by maximising the likelihood. To do this we create a training dataset consisting of N examples. We first sample N values of the parameter $\{\boldsymbol{\theta}^n\}_{n=1}^N$ from its prior and

for each $\boldsymbol{\theta}^n$ we simulate (see Appendix C) the sample path of the states and the observations using equation 1. Each training example is then created by collecting the random variables: $(\boldsymbol{X}_{t_i}^n, \boldsymbol{y}_{t_j}^n, \boldsymbol{\theta}^n)$, and $\boldsymbol{X}_{t_j}^n$, as the input-target pair, where $(i, j) = (0, 1), (2, 3), \ldots, (M-2, M-1)$. Clearly, even with a small number of model simulations (a few thousands) we can create a large training dataset to learn an expressive neural density estimator. In Figure 1 we outline the process of creating this training dataset. Given these training examples $\boldsymbol{\phi}$ can be learnt, using gradient descent, through maximising the total likelihood: $\sum_{n=1}^{N} q_{\boldsymbol{\phi}}(\boldsymbol{X}_{t_i}^n | \boldsymbol{X}_{t_{i-1}}^n, \boldsymbol{y}_{t_i}^n, \boldsymbol{\theta}^n)$, which is equivalent to minimising the Kullback–Leibler divergence $\mathrm{KL}\left(p(\boldsymbol{X}_{t_i}|\boldsymbol{X}_{t_{i-1}}, \boldsymbol{y}_{t_i}, \boldsymbol{\theta})||q_{\boldsymbol{\phi}}(\boldsymbol{X}_{t_i}|\boldsymbol{X}_{t_{i-1}}, \boldsymbol{y}_{t_i}, \boldsymbol{\theta})\right)$ (Papamakarios et al., 2019).

5 Related work in likelihood-free inference of HMMs

The most common approaches to tackle the inference of an intractable HMM consist largely of ABC methods (Dean et al., 2014; Martin et al., 2019; Picchini, 2014). Note that when the observational density is known analytically then the particle-MCMC (Andrieu et al., 2010) method can be used to carry out exact inference. However, the computational cost of this method is prohibitive, as in each step of MCMC a particle filter with a large number of particles is run to calculate an unbiased estimate of the marginal likelihood. Interestingly, a new avenue of research can be of combining our proposed IDE as an importance density within a particle-MCMC scheme. An alternative approach which combines SMC with ABC was proposed in (Drovandi et al., 2016). However, this approach requires the problematic choices of ABC algorithmic parameters.

The incremental posterior density is also the optimal importance density for a SMC sampler (Doucet et al., 2009; Creal, 2012). This quantity is only available in closed form for simple linear Gaussian models only. For complex models with a nonlinear transition this is mostly approximated sub-optimally using either linearisation techniques (Creal, 2012) or model approximations (Golightly & Wilkinson, 2011). The proposed IDE, being a normalising-flow, is a closed form approximation of this quantity. Thus, the IDE can be applied for novel algorithm design within the context of SMC.

6 Evaluations

We evaluate our proposed approach to recover the hidden states, through learning the incremental posterior density, using three biological HMMs: (i) the Lotka-Volterra (LV) model (Wilkinson, 2018), (ii) the prokaryotic auto-regulator (PKY) model (Golightly & Wilkinson, 2011) and (iii) a SIR epidemic model (Anderson et al., 1992). For the LV and PKY models the hidden states evolve as a pure Markov jump process (MJP) whereas for the SIR model we consider a stochastic differential equation (SDE). For all of the models we used simulated data, so that we are cognisant of the ground truth. We also used a tractable observation density $g(\cdot)$ to facilitate SMC.

Our goal is to evaluate how well the proposed approach can estimate the posterior predictive distribution $p(\boldsymbol{y}^*|\boldsymbol{y})$, in comparison to other approaches. We chose the following competing approaches. First is the ABC-SMC algorithm (Toni et al., 2009), used only for the PKY and LV model, which produces samples from the joint distribution $p(\boldsymbol{x}, \boldsymbol{\theta}|\boldsymbol{y})$, that can be used to evaluate the posterior predictive in equation 4. Except ABC-SMC all other approaches, including the proposed one, rely on the availability of parameters samples from the marginal $p(\boldsymbol{\theta}|\boldsymbol{y})$, which can be obtained using any off-the-shelf NLFI method such as the ones discussed in section 3. Once samples of $\boldsymbol{\theta}$ become available then samples of $\boldsymbol{x}$ can be drawn from its posterior using the IDE, the SMC (since $g(\cdot)$ is available), or simply from the prior transition $p(\boldsymbol{x}|\boldsymbol{\theta})$ (which we denote as PrDyn). Samples from $p(\boldsymbol{y}^*|\boldsymbol{y})$ can then be drawn using equation 12 for IDE, equation 8 for SMC and equation 5 for PrDyn.

To estimate $p(\boldsymbol{\theta}|\boldsymbol{y})$, required for IDE, SMC and PrDyn approaches, we used two sequential NLFI methods. One based on learning the likelihood density (SNLE) and the other one based on learning the likelihood-ratio (SRE). It was recently shown in (Durkan et al., 2020) that SRE (Hermans et al., 2020) is equivalent to a certain form of sequential learning of the posterior density (SNPE-C) (Greenberg et al., 2019) and both can be unified under a common framework on contrastive learning (Gutmann & Hyvärinen, 2010). Thus, by using SNLE and SRE we can cover the general ambit of sequential NLFI approaches. For SNLE we again used a MAF as the likelihood density estimator $q_{\psi}(s(\boldsymbol{y})|\boldsymbol{\theta})$ and for SRE we used a mlp classifier. For both the MAFs, $q_{\boldsymbol{\phi}}(\boldsymbol{X}_{t_i}|\boldsymbol{X}_{t_{i-1}}, \boldsymbol{y}_{t_i}, \boldsymbol{\theta})$ for the IDE and

$q_\psi(\boldsymbol{\theta}|s(\boldsymbol{y}))$ for SNLE, we used the same architecture. That is $J = 5$ layers each of which has two hidden layers of 50 units each and ReLU nonlinearities. For SRE we used a residual network based classifier with two residual layers of 50 units each and ReLU nonlinearities. For training all the neural networks we used ADAM (Kingma & Ba, 2015) with a minibatch size of 256, and a learning rate of 0.0005. Following, Papamakarios et al. (2019) we used 10% of the training data as a validation set, and stopped training if validation log likelihood did not improve after 20 epochs. Following Papamakarios et al. (2019), we used the Slice Sampling algorithm (Neal, 2003) to draw samples from the posterior while using SNLE and SRE.

Since, $p(\boldsymbol{y}^*|\boldsymbol{y})$ is obtained by jointly marginalising out $\boldsymbol{\theta}$ and $\boldsymbol{x}$, and since the IDE is only useful in estimating $p(\boldsymbol{x}|\boldsymbol{\theta}, \boldsymbol{y})$, thus we want to rule out major differences in the estimates of $\boldsymbol{\theta}$ among different methods. For this reason we used a fixed budget of simulations respectively for ABC-SMC (which jointly estimates $\boldsymbol{\theta}, \boldsymbol{x}$) and SNLE/SRE (used for estimating $\boldsymbol{\theta}$). These simulation budgets were determined based on previous studies such as Lueckmann et al. (2021), on models such as the ones we used, that compared the sample efficiency between ABC-SMC and NLFI methods for estimating $p(\boldsymbol{\theta}|\boldsymbol{y})$. Thus, for all three models, while using SNLE/SNRE, we used 30 rounds and the posterior samples from the final round were collected. For each model we generated 5000 training examples in the first round and in the subsequent rounds we generated 1000 examples. Furthermore, we used the 5000 simulations generated in the first round to learn the parameters $\boldsymbol{\phi}$ of the IDE. We limited the ABC-SMC to use no more than 10^7 simulations from the model. ABC-SMC is far less sample efficient in comparison to SNLE/SRE (Lueckmann et al., 2021). Hence, we used considerably more simulations, in case of ABC-SMC, to ensure that the parameter estimates are as close as possible to SNLE/SRE so that the differences in estimates of $\boldsymbol{x}$, and subsequently $\boldsymbol{y}^*$, cannot be attributed to differences in parameter estimates. Further details of ABC-SMC settings are given in Appendix B. Note that for inferring the parameters using ABC, SNLE and SRE we used summary statistics (hand-crafted). Training of the IDE requires the full data.

Note that since the SMC algorithm is (asymptotically) exact for the task of inferring $p(\boldsymbol{x}|\boldsymbol{y}, \boldsymbol{\theta})$, thus we considered the estimates of $p(\boldsymbol{y}^*|\boldsymbol{y})$ obtained using SMC to be the baseline. Hence, we compared different approaches by measuring the maximum mean discrepancy (MMD) (Gretton et al., 2012) between an estimate obtained by SMC and that obtained using other competing approaches. In particular we considered the sum of the pathwise MMDs for comparison:

$$\text{MMD}_{ppc} = \sum_{i=0}^{M-1} MMD^2(\boldsymbol{y}_{t_i}^{*^{smc}}, \boldsymbol{y}_{t_i}^{*^{a}}), \tag{15}$$

where $\boldsymbol{y}^{*^{smc}}$ is obtained using SMC and $\boldsymbol{y}^{*^{a}}$ using approach "a" (IDE, PrDyn, ABC-SMC). We used a rbf kernel and 500 samples, from the posterior predictive distribution, for calculating the MMD^2 using the formula in Gretton et al. (2012). For running the SMC algorithm we used a Bootstrap particle-filter with 100 particles for all models, shown to be enough for these models in Golightly & Wilkinson (2011). We repeated this evaluation for 10 different simulated datasets.

Additionally, we have also run an experiment with the LV model to show the perils of trying to estimate $\boldsymbol{x}, \boldsymbol{\theta}$ jointly using a neural density estimator. See Appendix E for details.

The NLFI approaches were implemented using the `sbi`[1] package (Tejero-Cantero et al., 2020). We implemented the stochastic simulation algorithm (Gillespie, 1977) in `C++` to simulate the LV and PKY models, and an Euler-Maruyama solver (Kloeden & Jentzen, 2007), implemented in `Python`, was used to simulate the SIR. All experiments were run on a high performance computing cluster. Our code is available at `retracted`.

6.1 Models

The stochastic Lotka–Volterra model: This model (Wilkinson, 2018) has been widely used for benchmarking (see Fearnhead et al. (2014); Giagos (2010)) a variety of inference algorithms including ones that use NLFI. This model describes a population comprising of two competing species: *predators* which die with rate c_2 and reproduce with rate c_1 by consuming prey, which in turn reproduce with rate c_3. The dynamics of the states $\boldsymbol{X}_t = (X_t^{pred}, X_t^{prey})$ is governed by a MJP

[1] https://www.mackelab.org/sbi/

and the objective here is to recover the posterior of the parameters $\boldsymbol{\theta} = (c_1, c_2, c_3)$ and the hidden states given noisy observations of the state $\boldsymbol{X}_t$.

Prokaryotic autoregulatory gene network: Golightly & Wilkinson (2011) considered the inference of an autoregulatory gene network that consists of four species $\boldsymbol{X}_t = (RNA_t, P_t, P2_t, DNA_t)$ evolving according to a MJP. The parameter vector $\boldsymbol{\theta} = (c_1, c_2, c_3, c_4, c_5, c_6, c_7, c_8)$ consists of eight reaction constants. The latent MJP is observed as a sum of the proteins $P, P2$ at each time-point. This model has traits of un-identifiability and had been used for benchmarking various inference methods.

The SIR Compartmental Model: The SIR model (Anderson et al., 1992) of infectious disease models the number of susceptible (S), infected (I), and recovered (R) people in a population subjected to an epidemic. The stochastic version of the SIR model, for a population of N_{pop} people, can be described as a SDE consisting the state vector $\boldsymbol{X}_t = (S_t, I_t)$. Note that for the SIR model we have $N_{pop} = S_t + I_t + R_t$. The SDE is governed by two parameters that control the infection β and recovery γ rates. In addition to β, γ we also estimated the fractional initial susceptibility, $s_0 = S_0/N$, assuming the initial recovered fraction $r_0 = 0$ and thus $i_0 = 1 - s_0$. Hence, the parameter vector is $\boldsymbol{\theta} = (\beta, \gamma, s_0)$.

See Appendix C for further details such as priors, data-generation and summary statistics for the models introduced above.

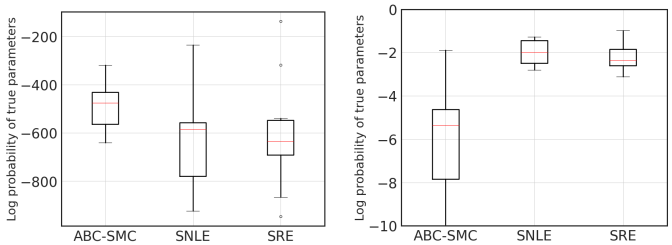
6.2 Results

In Table 1 we furnished the MMD estimates for all the models, which show that our proposed method produces an estimate of the posterior predictive that is closer to what can be achieved using SMC. In Figure 2 we have compared the accuracy of parameters estimates obtained using ABC-SMC and the NLFI methods for the LV and PKY models. Accuracy of the estimates were evaluated as the log probability of the true parameter vector under a mixture of multivariate Gaussian densities fitted to 500 samples drawn from an estimate of $p(\boldsymbol{\theta}|\boldsymbol{y})$ obtained using each method. We did not notice any drastic difference in accuracy and thus the difference in the MMD estimates were largely influenced by the estimates of the hidden states. The marginal densities of the parameter posteriors, for one dataset, are shown in Appendix D.

Posterior sample paths obtained by the IDE are almost indistinguishable from a SMC estimate (shown for one dataset in Figure 3). Notice the overestimation of uncertainty in case of PrDyn and ABC-SMC, for all models.

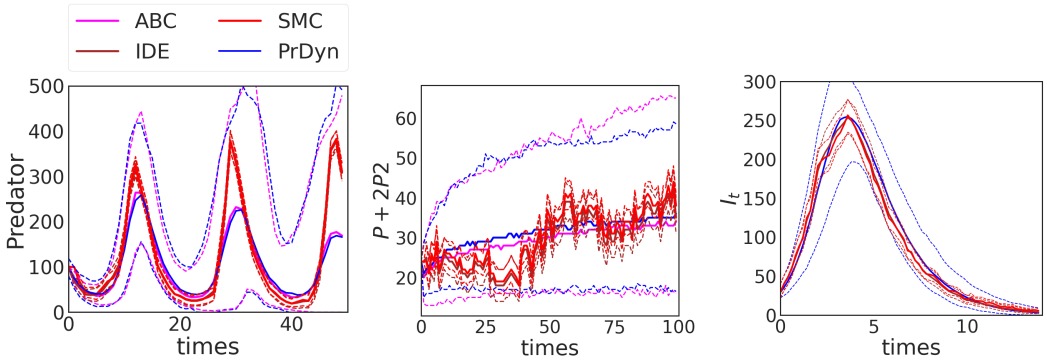
Table 1: The sum of pathwise MMD (smaller the better, given by equation 15) between estimates of the posterior predictive distribution obtained using **SMC** and other competing approaches: **IDE** (proposed), **PrDyn** and **ABC-SMC** (used for LV and PKY model only). These MMDs are summarised by the **mean** $\pm$ **standard deviation** across 10 different simulated datasests.

Comparison of the estimate of $p(\boldsymbol{y}^*\|\boldsymbol{y})$ with **SMC** (baseline)			
(When $p(\boldsymbol{\theta}\|\boldsymbol{y})$ for **SMC**, **IDE** and **PrDyn** obtained using **SNLE**)			
Model	**IDE**	**PrDyn**	**ABC-SMC**
LV	**1.8965** $\pm$ **0.2432**	17.2093 $\pm$ 1.1659	15.4923 $\pm$ 0.8170
PKY	**4.4448** $\pm$ **1.3243**	20.9553 $\pm$ 2.7296	53.2612 $\pm$ 4.9731
SIR	**1.1372** $\pm$ **0.5208**	5.0773 $\pm$ 0.2570	–
(When $p(\boldsymbol{\theta}\|\boldsymbol{y})$ for **SMC**, **IDE** and **PrDyn** obtained using **SRE**)			
Model	**IDE**	**PrDyn**	**ABC-SMC**
LV	**1.8998** $\pm$ **0.2491**	17.1193 $\pm$ 1.1592	15.4705 $\pm$ 0.8785795
PKY	**5.5491** $\pm$ **1.4890**	21.1983 $\pm$ 2.0962	53.6201 $\pm$ 0.0278
SIR	**1.1922** $\pm$ **0.5415**	4.7694 $\pm$ 0.2298	–



(a) (b)

Figure 2: Comparison of the accuracy of parameter estimates for the **Lotka-Volterra** (a) and **Prokaryotic autoregulator** (b) models, using the log probability of the true generative parameter vector, summarised across the 10 datasets.



(a) (b) (c)

Figure 3: Posterior distributions of one component of the latent sample path $\boldsymbol{x}$ summarised by the mean (solid lines) and 95% credible intervals (broken lines), for the **Lotka-Volterra** (a), **Prokaryotic autoregulator** (b) and **SIR** (c) model. The baseline is **SMC**.

6.3 Choice between SMC and IDE in terms of simulation budget

From the experiments it became clear that the proposed IDE is the only approach comparable to SMC in terms of the quality of estimate of the posterior predictive. However, note that the total number of simulations required for SMC is the size of the particle population times the number of parameter samples, which for the experiments was 100×500. These are simulations required in addition to the ones needed to learn $\boldsymbol{\theta}$. In comparison the IDE re-used the simulation needed to learn $\boldsymbol{\theta}$. Thus, IDE can be preferred over SMC, even when $g(\cdot)$ is available.

7 Conclusion

Neural likelihood-free methods have been previously benchmarked using intractable HMMs and are proposed as a computationally cheaper alternative to classical methods such as ABC-SMC in surrounding literature. We argued (and have shown empirically) that both classical as well as neural likelihood-free methods, by ignoring the state estimation, can lead to a grossly incorrect assessment of the goodness-of-fit. We thus proposed a novel postprocessing technique to approximately estimate the hidden states once samples from the posterior (of the parameters) have been obtained, by any likelihood-free method. Our technique, based on learning the posterior Markov process, using an autoregressive flow, produced estimate of the hidden states closer to what can be obtained using a SMC based postprocessing that can be only used for tractable observational densities.

REFERENCES

Roy M Anderson, B Anderson, and Robert M May. *Infectious diseases of humans: dynamics and control*. Oxford university press, 1992.

Christophe Andrieu and Gareth O Roberts. The pseudo-marginal approach for efficient monte carlo computations. *The Annals of Statistics*, 37(2):697–725, 2009.

Christophe Andrieu, Arnaud Doucet, and Roman Holenstein. Particle markov chain monte carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(3): 269–342, 2010.

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

Christopher M Bishop. Mixture density networks. 1994.

Kyle Cranmer, Juan Pavez, and Gilles Louppe. Approximating likelihood ratios with calibrated discriminative classifiers. *arXiv preprint arXiv:1506.02169*, 2015.

Kyle Cranmer, Johann Brehmer, and Gilles Louppe. The frontier of simulation-based inference. *Proceedings of the National Academy of Sciences*, 117(48):30055–30062, 2020.

Drew Creal. A survey of sequential monte carlo methods for economics and finance. *Econometric reviews*, 31(3):245–296, 2012.

Thomas A Dean, Sumeetpal S Singh, Ajay Jasra, and Gareth W Peters. Parameter estimation for hidden markov models with intractable likelihoods. *Scandinavian Journal of Statistics*, 41(4): 970–987, 2014.

Pierre Del Moral, Arnaud Doucet, and Ajay Jasra. An adaptive sequential monte carlo method for approximate bayesian computation. *Statistics and computing*, 22(5):1009–1020, 2012.

Arnaud Doucet, Adam M Johansen, et al. A tutorial on particle filtering and smoothing: Fifteen years later. *Handbook of nonlinear filtering*, 12(656-704):3, 2009.

Christopher C Drovandi, Anthony N Pettitt, and Roy A McCutchan. Exact and approximate bayesian inference for low integer-valued time series models with intractable likelihoods. *Bayesian Analysis*, 11(2):325–352, 2016.

Conor Durkan, George Papamakarios, and Iain Murray. Sequential neural methods for likelihood-free inference. *arXiv preprint arXiv:1811.08723*, 2018.

Conor Durkan, Iain Murray, and George Papamakarios. On contrastive learning for likelihood-free inference. In *International Conference on Machine Learning*, pp. 2771–2781. PMLR, 2020.

Paul Fearnhead, Vasilieos Giagos, and Chris Sherlock. Inference for reaction networks using the linear noise approximation. *Biometrics*, 70(2):457–466, 2014.

Sarah Filippi, Chris P Barnes, Julien Cornebise, and Michael PH Stumpf. On optimality of kernels for approximate bayesian computation using sequential monte carlo. *Statistical applications in genetics and molecular biology*, 12(1):87–107, 2013.

Christiane Fuchs. *Inference for diffusion processes: with applications in life sciences*. Springer Science & Business Media, 2013.

Mathieu Germain, Karol Gregor, Iain Murray, and Hugo Larochelle. Made: Masked autoencoder for distribution estimation. In *International Conference on Machine Learning*, pp. 881–889. PMLR, 2015.

Vasileios Giagos. *Inference for auto-regulatory genetic networks using diffusion process approximations*. PhD thesis, Lancaster University, 2010.

Daniel T Gillespie. Exact stochastic simulation of coupled chemical reactions. *The journal of physical chemistry*, 81(25):2340–2361, 1977.

Andrew Golightly and Colin S Gillespie. Simulation of stochastic kinetic models. In *In Silico Systems Biology*, pp. 169–187. Springer, 2013.

Andrew Golightly and Darren J Wilkinson. Bayesian parameter inference for stochastic biochemical network models using particle markov chain monte carlo. *Interface focus*, 1(6):807–820, 2011.

Neil Gordon, David Salmond, and Craig Ewing. Bayesian state estimation for tracking and guidance using the bootstrap filter. *Journal of Guidance, Control, and Dynamics*, 18(6):1434–1443, 1995.

David Greenberg, Marcel Nonnenmacher, and Jakob Macke. Automatic posterior transformation for likelihood-free inference. In *International Conference on Machine Learning*, pp. 2404–2414. PMLR, 2019.

Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(Mar):723–773, 2012.

Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 297–304. JMLR Workshop and Conference Proceedings, 2010.

Joeri Hermans, Volodimir Begy, and Gilles Louppe. Likelihood-free mcmc with amortized approximate ratio estimators. In *International Conference on Machine Learning*, pp. 4239–4248. PMLR, 2020.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

Peter E Kloeden and Arnulf Jentzen. Pathwise convergent higher order numerical schemes for random ordinary differential equations. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 463(2087):2929–2944, 2007.

Peter Eris Kloeden, Eckhard Platen, and Henri Schurz. *Numerical solution of SDE through computer experiments*. Springer Science & Business Media, 2012.

Jan-Matthis Lueckmann, Pedro J Goncalves, Giacomo Bassetto, Kaan Öcal, Marcel Nonnenmacher, and Jakob H Macke. Flexible statistical inference for mechanistic models of neural dynamics. *Advances in neural information processing systems*, 30, 2017.

Jan-Matthis Lueckmann, Jan Boelts, David Greenberg, Pedro Goncalves, and Jakob Macke. Benchmarking simulation-based inference. In Arindam Banerjee and Kenji Fukumizu (eds.), *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pp. 343–351. PMLR, 13–15 Apr 2021.

Jean-Michel Marin, Pierre Pudlo, Christian P Robert, and Robin J Ryder. Approximate bayesian computational methods. *Statistics and Computing*, 22(6):1167–1180, 2012.

Paul Marjoram, John Molitor, Vincent Plagnol, and Simon Tavaré. Markov chain monte carlo without likelihoods. *Proceedings of the National Academy of Sciences*, 100(26):15324–15328, 2003.

Gael M Martin, Brendan PM McCabe, David T Frazier, Worapree Maneesoonthorn, and Christian P Robert. Auxiliary likelihood-based approximate bayesian computation in state space models. *Journal of Computational and Graphical Statistics*, 28(3):508–522, 2019.

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

George Papamakarios and Iain Murray. Fast ε-free inference of simulation models with bayesian conditional density estimation. *Advances in neural information processing systems*, 29, 2016.

George Papamakarios, Theo Pavlakou, and Iain Murray. Masked autoregressive flow for density estimation. *Advances in neural information processing systems*, 30, 2017.

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

George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 22(57):1–64, 2021.

Matthew David Parno. *Transport maps for accelerated Bayesian computation*. PhD thesis, Massachusetts Institute of Technology, 2015.

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

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

Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pp. 1530–1538. PMLR, 2015.

Simo Särkkä. *Bayesian filtering and smoothing*. Number 3. Cambridge university press, 2013.

Scott A Sisson, Yanan Fan, and Mark Beaumont. *Handbook of approximate Bayesian computation*. CRC Press, 2018.

Esteban G Tabak and Cristina V Turner. A family of nonparametric density estimation algorithms. *Communications on Pure and Applied Mathematics*, 66(2):145–164, 2013.

Alvaro Tejero-Cantero, Jan Boelts, Michael Deistler, Jan-Matthis Lueckmann, Conor Durkan, Pedro J. Gonçalves, David S. Greenberg, and Jakob H. Macke. sbi: A toolkit for simulation-based inference. *Journal of Open Source Software*, 5(52):2505, 2020. doi: 10.21105/joss.02505. URL https://doi.org/10.21105/joss.02505.

T. Toni, D. Welch, N. Strelkowa, A. Ipsen, and M. P.H Stumpf. Approximate Bayesian computation scheme for parameter inference and model selection in dynamical systems. *Journal of the Royal Society Interface*, 6(31):187–202, February 2009.

Darren J Wilkinson. *Stochastic modelling for systems biology*. CRC press, 2018.

A Derivations of ABC and incremental posteriors of HMM

A.1 Joint distribution for HMM using ABC

NLFI methods are designed to efficiently sample from the marginal distribution $p(\boldsymbol{\theta}|\boldsymbol{y})$. In ABC although the desired outcome often is the marginal distribution, however it is easy to show that for a latent variable model, such as an intractable HMM, ABC does indeed target an approximation of the joint distribution $p(\boldsymbol{\theta}, \boldsymbol{x}|\boldsymbol{y})$.

In ABC we rely upon simulation of a pseudo-data $\hat{\boldsymbol{y}}$, when the likelihood $p(\boldsymbol{y}|\boldsymbol{\theta})$ is intractable. The operating principle of any standard ABC algorithm, based on rejection sampling (Pritchard et al., 1999), MCMC (Marjoram et al., 2003) or SMC (Toni et al., 2009; Del Moral et al., 2012), is to jointly sample the parameters $\boldsymbol{\theta}$ and the pseudo-data $\hat{\boldsymbol{y}}$ from their posterior density (Marin et al., 2012)

$$p_\epsilon(\boldsymbol{\theta}, \hat{\boldsymbol{y}}|\boldsymbol{y}) = \frac{\mathbb{1}_\epsilon \left\{ d(s(\hat{\boldsymbol{y}}), s(\boldsymbol{y}) < \epsilon) \right\} p(\hat{\boldsymbol{y}}|\boldsymbol{\theta}) p(\boldsymbol{\theta})}{\int \mathbb{1}_\epsilon \left\{ d(s(\hat{\boldsymbol{y}}), s(\boldsymbol{y}) < \epsilon) \right\} p(\hat{\boldsymbol{y}}|\boldsymbol{\theta}) p(\boldsymbol{\theta}) d\boldsymbol{\theta}}, \tag{16}$$

where $\mathbb{1}_\epsilon(\cdot)$ is the indicator function, $d(\cdot)$ is a chosen distance metric, $\epsilon > 0$ and we consider the summary $s(\cdot)$ to be sufficient. The desired marginal posterior then follows as

$$p_\epsilon(\boldsymbol{\theta}|\boldsymbol{y}) = \int p_\epsilon(\boldsymbol{\theta}, \hat{\boldsymbol{y}}|\boldsymbol{y}) d\hat{\boldsymbol{y}}. \tag{17}$$

Note that the pseudo-data distribution $p(\hat{\boldsymbol{y}}|\boldsymbol{\theta})$ appearing in equation 16 is not required analytically in any of the ABC algorithms. This distribution is essentially the generative model under consideration.

For the HMM such a pseudo data is sampled from the distribution

$$p(\hat{\boldsymbol{y}}, \boldsymbol{x}|\boldsymbol{\theta}) = \left(\prod_{i=0}^{M-1} g(\hat{\boldsymbol{y}}_{t_i}|\boldsymbol{X}_{t_i}, \boldsymbol{\theta}) \right) \left(\prod_{i=1}^{M-1} f(\boldsymbol{X}_{t_i}|\boldsymbol{X}_{t_{i-1}}, \boldsymbol{\theta}) \right), \tag{18}$$

where $f(\cdot)$, $g(\cdot)$ and thus $p(\hat{\boldsymbol{y}}, \boldsymbol{x}|\boldsymbol{\theta})$ need not be analytically tractable, just a sample $\hat{\boldsymbol{y}}$ of the pseudo-data from this distribution is required. Sampling from this distribution is essentially the process of forward sampling from the generative model of the HMM given by equation 1 (see main text). Considering $\hat{\boldsymbol{y}}$ alone from the pair $(\hat{\boldsymbol{y}}, \boldsymbol{x})$ we have a sample of the pseudo-data drawn from its marginal $p(\hat{\boldsymbol{y}}|\boldsymbol{\theta})$. Thus, when ABC is applied to the HMM in equation 1 the joint density in equation 16 is replaced by a density over the triplet $(\boldsymbol{\theta}, \boldsymbol{x}, \hat{\boldsymbol{y}})$ given by

$$p_\epsilon(\boldsymbol{\theta}, \boldsymbol{x}, \hat{\boldsymbol{y}}|\boldsymbol{y}) = \frac{\mathbb{1}_\epsilon \left\{ d(s(\hat{\boldsymbol{y}}), s(\boldsymbol{y}) < \epsilon) \right\} p(\hat{\boldsymbol{y}}, \boldsymbol{x}|\boldsymbol{\theta}) p(\boldsymbol{\theta})}{\int \mathbb{1}_\epsilon \left\{ d(s(\hat{\boldsymbol{y}}), s(\boldsymbol{y}) < \epsilon) \right\} p(\hat{\boldsymbol{y}}, \boldsymbol{x}|\boldsymbol{\theta}) p(\boldsymbol{\theta}) d\boldsymbol{\theta}}, \tag{19}$$

from which samples of the pair $(\boldsymbol{\theta}, \boldsymbol{x})$ is distributed from $p_\epsilon(\boldsymbol{\theta}, \boldsymbol{x}|\boldsymbol{y})$. And the corresponding ABC marginal posterior is given by

$$p_\epsilon(\boldsymbol{\theta}|\boldsymbol{y}) = \int p_\epsilon(\boldsymbol{\theta}, \boldsymbol{x}, \hat{\boldsymbol{y}}|\boldsymbol{y}) d\hat{\boldsymbol{y}} d\boldsymbol{x}. \tag{20}$$

From equation 19 it is evident that any ABC algorithm applied to the HMM will target the joint distribution $p_\epsilon(\boldsymbol{\theta}, \boldsymbol{x}|\boldsymbol{y})$. However, this distribution will only be an approximation to the true posterior $p(\boldsymbol{\theta}, \boldsymbol{x}|\boldsymbol{y})$, since $\epsilon \neq 0$ (considering $s(\cdot)$ to be sufficient). Note that since $\boldsymbol{x}$ is sampled from its prior thus if ϵ is set to zero (or a small value) then a practically infeasible amount of simulations is required to produce an ABC posterior $p(\boldsymbol{\theta}, \boldsymbol{x}|\boldsymbol{y})$ that can approximate closely the true posterior.

A.2 Deriving the incremental posterior

Consider the posterior distribution of the sample path and the parameters conditioned on the observations. We can write this density, upto a normalising constant, as follows

$$\begin{aligned} &p(\boldsymbol{X}_{t_{M-1}}, \ldots, \boldsymbol{X}_{t_0}, \boldsymbol{\theta}|\boldsymbol{y}_{t_{M-1}}, \ldots, \boldsymbol{y}_{t_0}) \propto \\ &p(\boldsymbol{\theta}) p(\boldsymbol{X}_{t_0}) \left(\prod_{i=0}^{M-1} g(\boldsymbol{y}_{t_i}|\boldsymbol{X}_{t_i}, \boldsymbol{\theta}) \right) \left(\prod_{i=1}^{M-1} f(\boldsymbol{X}_{t_i}|\boldsymbol{X}_{t_{i-1}}, \boldsymbol{\theta}) \right). \end{aligned} \tag{21}$$

We can obtain from this the density of $\boldsymbol{X}_{t_{M-1}}$ conditioned on all other random variables by only retaining the terms that involve it. So we have this conditional density, upto a normalising constant, given by

$$\text{p}(\text{X}_{t_{M-1}}|\boldsymbol{X}_{t_{M-2}}, \ldots, \boldsymbol{X}_{t_0}, \boldsymbol{\theta}, \boldsymbol{y}_{t_{M-1}}, \ldots, \boldsymbol{y}_{t_0}) \propto g(\boldsymbol{y}_{t_{M-1}}|\boldsymbol{X}_{t_{M-1}}, \boldsymbol{\theta}) f(\boldsymbol{X}_{t_{M-1}}|\boldsymbol{X}_{t_{M-2}}, \boldsymbol{\theta}) p(\boldsymbol{\theta}), \tag{22}$$

which is simply the density $p(\boldsymbol{X}_{t_{M-1}}|\boldsymbol{X}_{t_{M-2}}, \boldsymbol{y}_{t_{M-1}}, \boldsymbol{\theta})$.

B ABC-SMC implementation details

We applied the particular version of ABC-SMC algorithm, that was proposed in Toni et al. (2009), using 1000 particles. Furthermore, we used an adaptive tolerance sequence where the tolerance ϵ_τ at the τ-th step of the algorithm is selected as the 0.1-quantile of the distances of the accepted particles in the $\tau - 1$-th step. Moreover, we chose the perturbation kernel of ABC-SMC (see Toni et al. (2009)) as a multivariate Gaussian whose covariance is based on a *k-nearest neighbours* strategy, with $k = 15$, proposed in Filippi et al. (2013). We terminated the ABC-SMC algorithm when a predetermined number of simulations has been carried out. If that number is exceeded within the τ-th step, we then considered the weighted particle system at the $\tau - 1$-th step as the desired ABC posterior.

C MODEL DETAILS

C.1 STOCHASTIC LOTKA-VOLTERRA MODEL

The stochastic Lotka-Volterra model, a stochastic kinetic system, can be defined through the following list of reactions:

$$\begin{aligned} \mathcal{R}_1: &\quad X^{prey} \xrightarrow{c_1} 2X^{prey} \\ \mathcal{R}_2: &\quad X^{prey} + X^{pred} \xrightarrow{c_2} 2X^{pred} \\ \mathcal{R}_3: &\quad X^{pred} \xrightarrow{c_2} \emptyset, \end{aligned} \tag{23}$$

where we denote by X^{prey}, X^{pred} the prey and predator species respectively. We further denote the corresponding numbers of the species as the system state $\boldsymbol{X}_t = (X_t^{prey}, X_t^{pred})$. The hazard vector for this system is $h(\boldsymbol{X}_t, \boldsymbol{c}) = \left(c_1 X_t^{prey}, c_2 X_t^{prey} X_t^{pred}, c_3 X_t^{pred}\right)$. The stoichiometry matrix for this system is given by

$$S = \begin{pmatrix} 1 & -1 & 0 \\ 0 & 1 & -1 \end{pmatrix}. \tag{24}$$

We set the initial values as $\mathcal{X}_0 = (100, 100)$ and consider them known.

A MJP describing a stochastic kinetic system, like the one above or the PKY model, is characterised by the transition probability $p(t_0, \boldsymbol{X}_0, t, \boldsymbol{X}_t) := p(\boldsymbol{X}, t)$ for the process arriving at state $\boldsymbol{X}_t$ at time t conditioned on an initial state $\boldsymbol{X}_0$ at time t_0. This is basically the density $f(\cdot)$ in equation 1 (main text), in continuous time. Now this transition probability is given by the solution of the following differential equation:

$$\frac{\partial p(\boldsymbol{X}, t)}{\partial t} = \sum_{i=1}^{v} = \{h_i(\boldsymbol{X} - S^i, c_i) p(\boldsymbol{X} - S^i, t) - h_i(\boldsymbol{X}, c_i) p(\boldsymbol{X}, t)\}, \tag{25}$$

known as the chemical master equation (Golightly & Gillespie, 2013, and the references therein). The CME only admits an analytical solution for a handful of simple models (not for the ones we have used: LV and PKY). Thus $f(\cdot)$ in equation 1 (main text) cannot be evaluated. However, the seminal work in Gillespie (1977) developed an algorithm, commonly referred to as the *stochastic simulation algorithm*, that can simulate $\boldsymbol{X}$ exactly.

We generated simulated trajectories from this model using the stochastic simulation algorithm and added Gaussian noise corruption, with variance 100, at 50 time points. We used the following generative values of the parameters $\boldsymbol{\theta} = (0.3, 0.0025, 0.5)$ to ensure that the model follows an oscillatory regime. Moreover, following previous studies we considered the initial values to be known and set at $\boldsymbol{X}_{t_0} = (100, 100)$. For running ABC-SMC and all the NLFI methods we downsampled the generated time series by a factor of 5 to create a summary statistic $s(\boldsymbol{y}) \in \mathbb{R}^{20}$ which is used in place of the full data $\boldsymbol{y}$. For further details of the model and simulations see Appendix C. We used the following set of prior distributions: $c_1 \sim \text{Beta}(1, 2)$, $c_2 \times 10^3 \sim \mathcal{U}(15, 50)$ and $c_3 \sim \text{Beta}(2, 1)$.

C.2 PROKARYOTIC AUTOREGULATORY GENE NETWORK

We considered the autoregulatory model used to benchmark the particle MCMC method in Golightly & Wilkinson (2011). This is a simplified model that describes a mechanism for autoregulation in prokaryotes based on a negative feedback mechanism of dimers of a protein coded by a gene repressing its own transcription. Essentially this is a stochastic kinetic model described by the following set of reactions:

$$\begin{aligned} &\mathcal{R}_1: DNA + P2 \rightarrow DNA \cdot P2 \\ &\mathcal{R}_2: DNA \cdot P2 \rightarrow DNA + P2 \\ &\mathcal{R}_3: DNA \rightarrow DNA + RNA \\ &\mathcal{R}_4: RNA \rightarrow RNA + P \\ &\mathcal{R}_5: 2P \rightarrow P2 \\ &\mathcal{R}_6: P2 \rightarrow 2P \\ &\mathcal{R}_7: RNA \rightarrow \emptyset \\ &\mathcal{R}_8: P \rightarrow \emptyset. \end{aligned} \tag{26}$$

We order the variables as $\boldsymbol{X} = (RNA, P, P2, DNA, DNA \cdot P2)$ leading to a stoichiometry matrix for the system:

$$S = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 2 & 2 & 0 & 1 & \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \tag{27}$$

and the associated hazard function is given by

$$h(\boldsymbol{X}, \boldsymbol{c}) = (c_1 DNA \times P2, c_2 DNA \cdot P2, c_3 DNA, c_4 RNA, c_5 P(P-1)/2, c_6 P2, c_7 RNA, c_8 P). \tag{28}$$

This model has one conservation law (Golightly & Wilkinson, 2011)

$$DNA \cdot P2 + DNA = k, \tag{29}$$

where k is the number of copies of this gene in the genome. Following Golightly & Wilkinson (2011) we use this relation to to remove $DNA \cdot P2$ from the model, replacing any occurrences of $DNA \cdot P2$ in rate laws with $k - DNA$. This leads to a reduced full-rank model with species $\boldsymbol{X} = (RNA, P, P2, DNA)$, stoichiometry matrix:

$$S = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 2 & 2 & 0 & 1 & \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \tag{30}$$

and associated hazard function

$$h(\boldsymbol{X}, \boldsymbol{c}) = (c_1 DNA \times P2, c_2(k - DNA), c_3 DNA, c_4 RNA, c_5 P(P-1)/2, c_6 P2, c_7 RNA, c_8 P). \tag{31}$$

We consider k to be known and set to 10. Again we generated simulated trajectories from this model using the stochastic simulation algorithm.

Following Golightly & Wilkinson (2011), we considered the observations as a linear combination of the proteins $P, P2$ as follows:

$$y_t = P_t + 2P2_t + \epsilon_t, \tag{32}$$

where ϵ is assumed to be iid Gaussian noise. We generated 100 simulated observations from this model at times $t = [0 : .5 : 50]$ with generative rate constants $\boldsymbol{\theta} = (0.1, 0.7, 0.35, 0.2, 0.1, 0.9, 0.3, 0.1)$ and $\epsilon \sim \mathcal{N}(0, 4)$. In this case also we consider the initial values $\boldsymbol{X}_{t_0}$ to be known and set to $(8, 8, 8, 5)$. We downsampled the simulated data by a factor of five to obtain the summary statistics $s(\boldsymbol{y}) \in \mathbb{R}^{20}$. Furthermore, we placed a $\mathrm{Gamma}(2, 3)$ prior on all the rate constants.

C.3 The SIR Compartmental Model

The stochastic version of the SIR model, for a population of N_{pop} people, with states variables $\boldsymbol{X}_t = (S_t, I_t)$ can be defined using an Itô SDE:

$$d\boldsymbol{X}_t = (\boldsymbol{X}_t, \boldsymbol{\theta})dt + \sqrt{\boldsymbol{B}(\boldsymbol{X}_t, \boldsymbol{\theta})} d\boldsymbol{W}_t, \tag{33}$$

driven by a two-dimensional Brownian motion $\boldsymbol{W}_t$ with the following drift and diffusion terms (see Fuchs (2013) for derivation):

$$\begin{aligned} (\boldsymbol{X}_t, \boldsymbol{\theta}) &= \begin{bmatrix} -\beta S_t I_t \\ \beta S_t I_t - \gamma I_t \end{bmatrix}, \\ \boldsymbol{B}(\boldsymbol{X}_t, \boldsymbol{\theta}) &= \frac{1}{N_{pop}} \begin{bmatrix} \beta S_t I_t & -\beta S_t I_t \\ -\beta S_t I_t & \beta S_t I_t + \gamma I_t \end{bmatrix}, \end{aligned} \tag{34}$$

where the β and γ are the unknown rate of infection and recovery. We generated simulated trajectories by numerically solving this SDE using the Euler-Maruyama solver (Kloeden et al., 2012).

We simulated the SDE on a time interval $t = [0 : 0.2, 14]$, with generative values of $\boldsymbol{\theta} = (1.7, 0.6, 1 - 0.05)$, to generate the state trajectory. We consider the observations to be the prevalence of the latent infectious state I_t observed through a Poisson counting process. Thus, we have the observations as $p(y_t|\beta, \gamma, s_0) = \mathrm{Poisson}(I_t)$, and placed the following priors: $\beta \sim \mathcal{U}(0.5, 2)$, $\gamma \sim \mathcal{U}(0.1, 1)$ and $1 - s_0 \sim \mathcal{U}(0.001,)$. We then downsampled the observation time series and retain 14 values as the summary statistics $s(\boldsymbol{y}) \in \mathbb{R}^{14}$.

D PLOTS OF PARAMETER POSTERIORS

In the subsequent plots Figure D.1, D.2 and D.3 we compare the parameter estimates of the three models between NLFI based methods, SNLE/SRE, and ABC-SMC. Here we have considered the estimates for one of the 10 different simulated datasets. Note that the parameter estimates are reasonably close to each other and thus the estimate of the posterior predictive distribution is largely influenced by the estimates of the hidden states.

D.1 LOTKA-VOLTERRA

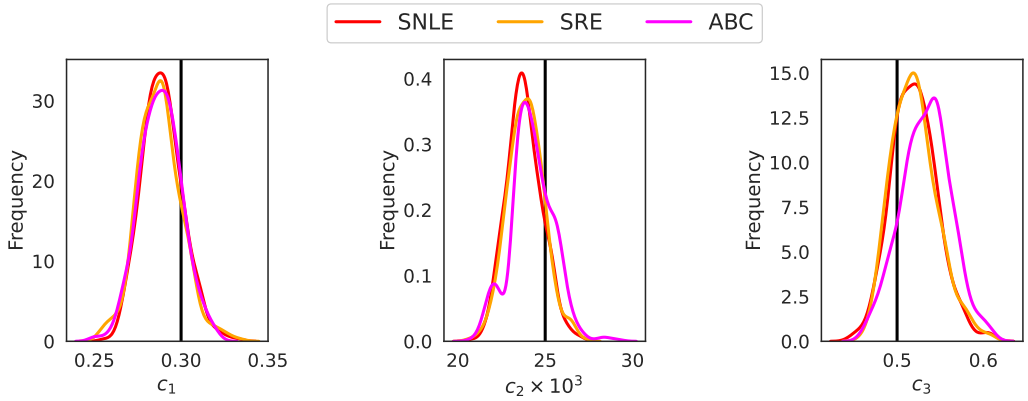


Figure 4: Posterior marginal densities of the parameters of the **Lotka-Volterra** model, inferred from one of the 10 datasets.

D.2 SIR MODEL

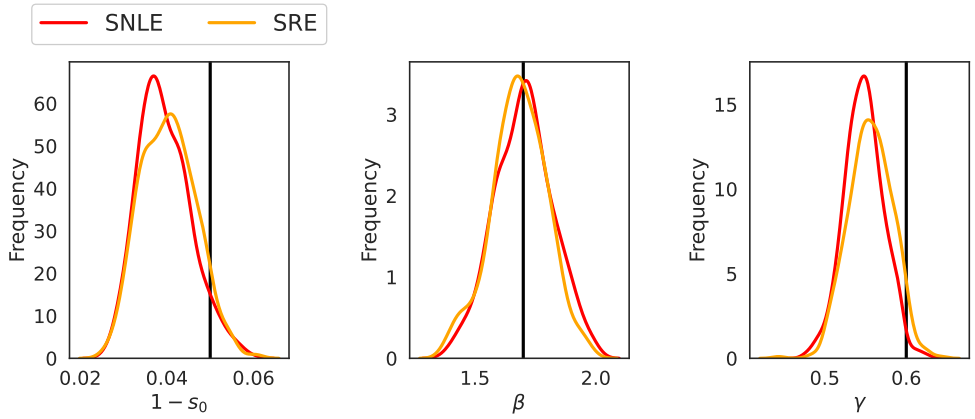


Figure 5: Posterior marginal densities of the parameters of the **SIR** model, inferred from one of the 10 datasets.

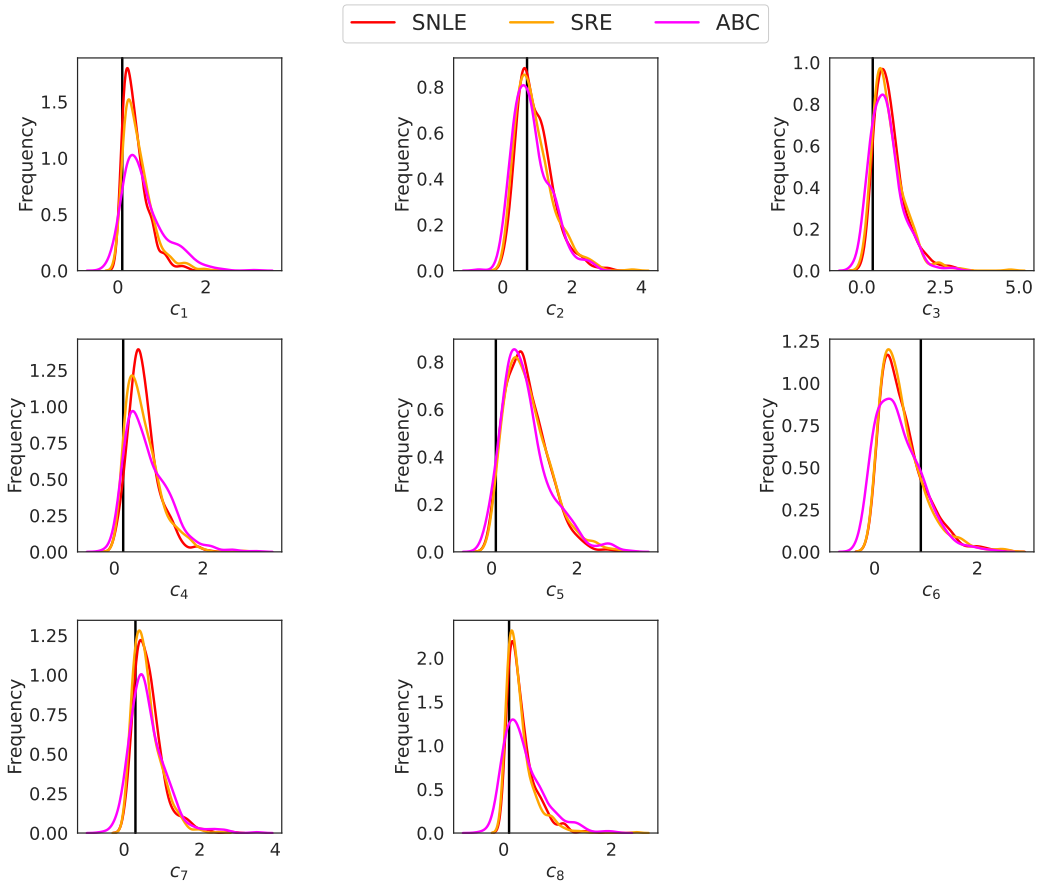


Figure 6: Posterior marginal densities of the parameters of the **Prokaryotic autoregulatory** model, inferred from one of the 10 datasets.

D.3 Prokaryotic autoregulatory gene network

E Joint inference of the sample path and parameters using a MAF

We have argued before (see the last paragraph of section 3) that NLFI methods cannot be used directly for inferring the joint posterior $p(\boldsymbol{x}, \boldsymbol{\theta}|\boldsymbol{y})$. Next, we have shown results for an experiment, using the LV model, that supports our argument. Note that due to the unavailability of $p(\boldsymbol{x}, \boldsymbol{\theta})$, the only strategy that can be applied is of using a normalizing-flow to directly emulate the joint posterior $p(\boldsymbol{x}, \boldsymbol{\theta}|\boldsymbol{y}) \approx q_{\boldsymbol{\psi}}(\boldsymbol{x}, \boldsymbol{\theta}|\boldsymbol{y})$. We denote this approach as neural posterior estimation (NPE). We used 10^6 simulations from the model to train a MAF representing $q_{\boldsymbol{\psi}}(\boldsymbol{x}, \boldsymbol{\theta}|\boldsymbol{y})$. Note that for the proposed IDE approach we used 35×10^3 (including inference of $\boldsymbol{\theta}$). We retained the same architecture and optimisation settings that we used in other experiments. Once trained, we used one of the simulated dataset for the LV model to carry out inference. This is the same dataset corresponding to the plot shown in Figure 3(a).

In Figure E we plot one component of the hidden state estimated by SMC, IDE, ABC-SMC and NPE. Note that SMC, IDE are using same samples of $\boldsymbol{\theta}$ estimated using SNLE. All methods use 500 samples from the posteriors of $\boldsymbol{\theta}, \boldsymbol{x}$. In Figure E we show the corresponding parameter estimates. Although NPE estimates the hidden state better than ABC-SMC, its estimation quality drops at those time points where the concentration reaches a peak before decreasing again. This drop is much more pronounced near the last peak. The parameter estimates are however significantly different than all

the other methods. From which it can be concluded that NPE performs worse than even ABC-SMC to produce the posterior of the parameters when targeting $\boldsymbol{x}, \boldsymbol{\theta}$ jointly.

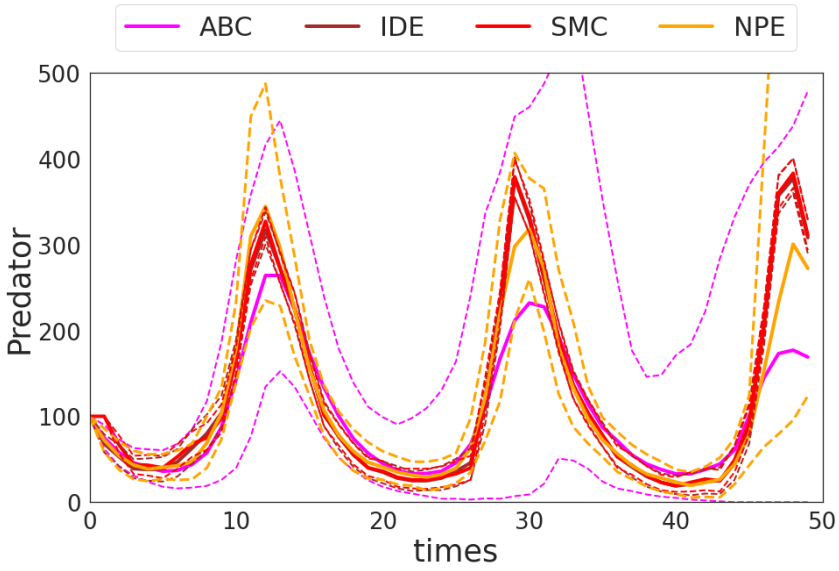


Figure 7: Comparison between methods that estimate jointly the parameters and hidden states of a HMM (in this case the **Lotka-Volterra** model), such as **ABC-SMC** & **NPE**, with those that estimate these quantities separately, such as **SMC** & **IDE**. The plot above shows the posteriors of one component of the hidden state summarised by the mean (solid lines) and 95% credible intervals (broken lines). The proposed method **IDE** reduces the simulation burden by a large factor in comparison to **NPE**. Note that even with a much larger simulation budget **NPE** fails to correctly estimate the hidden states as well as the parameters (see Figure E).

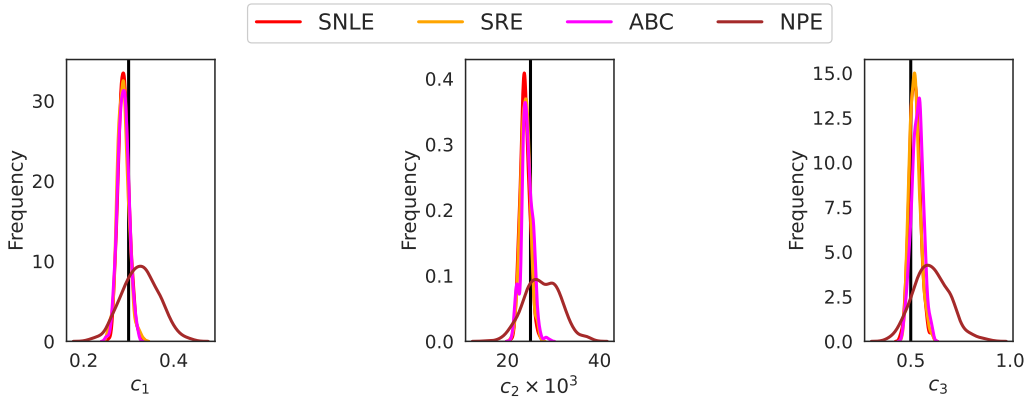


Figure 8: Posterior marginal densities of the parameters of the **Lotka-Volterra** model obtained using SNLE, SRE (both targeting the marginal $p(\boldsymbol{\theta}|\boldsymbol{y})$) with NPE, ABC-SMC (both targeting the joint $p(\boldsymbol{x}, \boldsymbol{\theta}|\boldsymbol{y})$). NPE failed to estimate $\boldsymbol{\theta}$ correctly.