Journal of Economic Dynamics & Control 161 (2024) 104827

Contents lists available at ScienceDirect

Journal of Economic Dynamics and Control

journal homepage: www.elsevier.com/locate/jedc

Black-box Bayesian inference for agent-based models

Joel Dyer [a,b,*], Patrick Cannon, J. Doyne Farmer [a,c,d], Sebastian M. Schmon

[a] Institute for New Economic Thinking, Manor Road Building, Manor Road, Oxford, OX1 3UQ, United Kingdom
[b] Department of Computer Science, University of Oxford, 7 Parks Rd, Oxford, OX1 3QG, United Kingdom
[c] Smith School of Enterprise and the Environment, School of Geography and the Environment, University of Oxford, Oxford, OX1 3QY, United Kingdom
[d] Santa Fe Institute, 1399 Hyde Park Rd, Santa Fe, NM 87501, United States

ARTICLE INFO

Dataset link: https://github.com/joelnmdyer/sbi4abm

Keywords:
Agent-based models
Bayesian inference
Neural networks
Parameter estimation
Simulation-based inference
Time series

ABSTRACT

Simulation models, in particular agent-based models, are gaining popularity in economics and the social sciences. The considerable flexibility they offer, as well as their capacity to reproduce a variety of empirically observed behaviours of complex systems, give them broad appeal, and the increasing availability of cheap computing power has made their use feasible. Yet a widespread adoption in real-world modelling and decision-making scenarios has been hindered by the difficulty of performing parameter estimation for such models. In general, simulation models lack a tractable likelihood function, which precludes a straightforward application of standard statistical inference techniques. A number of recent works have sought to address this problem through the application of *likelihood-free* inference techniques, in which parameter estimates are determined by performing some form of comparison between the observed data and simulation output. However, these approaches are (a) founded on restrictive assumptions, and/or (b) typically require many hundreds of thousands of simulations. These qualities make them unsuitable for large-scale simulations in economics and the social sciences, and can cast doubt on the validity of these inference methods in such scenarios. In this paper, we investigate the efficacy of two classes of simulation-efficient black-box approximate Bayesian inference methods that have recently drawn significant attention within the probabilistic machine learning community: neural posterior estimation and neural density ratio estimation. We present a number of benchmarking experiments in which we demonstrate that neural network-based black-box methods provide state of the art parameter inference for economic simulation models, and crucially are compatible with generic multivariate or even non-Euclidean time-series data. In addition, we suggest appropriate assessment criteria for use in future benchmarking of approximate Bayesian inference procedures for simulation models in economics and the social sciences.

1. Introduction

Simulation models are increasingly employed across the sciences. Their primary advantage is the flexibility they afford modellers: models may be specified mechanistically and at the microscopic level without concern for analytic tractability of the full system behaviour. This flexibility is embraced in the *agent-based modelling* paradigm wherein individual agents can be described by realistic

* Corresponding author at: Department of Computer Science, University of Oxford, 7 Parks Rd, Oxford, OX1 3QG, United Kingdom.
E-mail address: joel.dyer@cs.ox.ac.uk (J. Dyer).

https://doi.org/10.1016/j.jedc.2024.104827
Received 5 November 2022; Received in revised form 7 November 2023; Accepted 29 January 2024

Available online 5 February 2024
0165-1889/© 2024 The Author(s). Published by Elsevier B.V. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

and heterogeneous decision and interaction rules. Examining the macroscopic properties of the system is then done simply through simulation, freeing the modeller from the burden of reasoning about the global behaviour of the system themselves and facilitating the study of radically more complex systems and emergent phenomena than is otherwise possible.

In general, agent-based models (ABMs) take as input some parameter vector $\boldsymbol{\theta}$, and return a (possibly multivariate) time-series $\mathbf{x}$. Typically, ABMs are *stochastic* simulators, with the property that running the simulator repeatedly with a fixed parameter $\boldsymbol{\theta}$ will produce different outputs with each run. The probability (density) with which the simulator produces a particular output $\mathbf{x}$, conditional on an input parameter vector $\boldsymbol{\theta}$, is given by its likelihood function, written $p(\mathbf{x} \mid \boldsymbol{\theta})$, which in some sense encapsulates the ABM's statistical properties. We denote simulation of a dataset $\mathbf{x}$ from the ABM as a draw from its probability density function, $\mathbf{x} \sim p(\mathbf{x} \mid \boldsymbol{\theta})$.

An essential prerequisite for the successful application of ABMs in real-world decision making scenarios is the ability to perform parameter estimation, that is, to tune $\boldsymbol{\theta}$ such that the simulated data are good matches to observed data, which we denote with $\mathbf{y}$. Traditional statistical workhorses like maximum likelihood estimation or Bayesian inference rely on pointwise evaluations of the likelihood function. In practice, $p(\mathbf{x} \mid \boldsymbol{\theta})$ cannot be obtained or evaluated in a reasonable time for arbitrary simulation models, and is thus only implicitly defined by the software specifying the behaviour of the simulator. This presents a barrier to the use of ABMs in real-world use.

To overcome this, a number of approaches to performing statistical inference have been developed in which exact density evaluations are replaced by evaluations of approximate densities or cost functions that are constructed using simulations from the model. Some of these simulation-based inference (SBI) methods have been explored within the ABM community. Perhaps the most prevalent of them is simulated minimum distance (SMD), in which an estimate $\hat{\boldsymbol{\theta}}$ is obtained by minimising some loss function $f(\mathbf{y}, \boldsymbol{\theta})$ between the observed data $\mathbf{y}$ and simulated data $\mathbf{x} \sim p(\cdot \mid \boldsymbol{\theta})$ over some search space $\boldsymbol{\Theta}$:

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} f(\mathbf{y}, \boldsymbol{\theta}). \tag{1}$$

This general class of estimators includes the maximum likelihood estimator – corresponding to choosing, for example, $f(\mathbf{y}, \boldsymbol{\theta}) = -\log p(\mathbf{y} \mid \boldsymbol{\theta})$ or, where the likelihood is intractable, some estimate thereof (see e.g. Diggle and Gratton, 1984; Kukacka and Barunik, 2017). It also includes the Method of Simulated Moments (MSM) (Franke, 2009) – in which f takes the form

$$f(\mathbf{y}, \boldsymbol{\theta}) = \left(g(\mathbf{y}) - \hat{g}_{\boldsymbol{\theta}}\right)' W \left(g(\mathbf{y}) - \hat{g}_{\boldsymbol{\theta}}\right), \tag{2}$$

where $g(\mathbf{y})$ denotes a set of moments derived from $\mathbf{y}$, $\hat{g}_{\boldsymbol{\theta}}$ denotes the same set of moments derived from $R \geq 1$ simulations at $\boldsymbol{\theta}$, W is a suitably chosen weight matrix, and $'$ denotes the transpose. As a final example, it includes Indirect Inference (II) (Gourieroux et al., 1993), which follows a similar approach to MSM but replaces the moments with estimated parameters of a tractable auxiliary model. Some further loss functions have been proposed more recently in the context of ABM estimation (such as e.g. the Markov Information Criterion Barde (2017) and GSL-div Lamperti (2018b); see Platt (2020) for a recent review).

However, over recent years there has been a growing interest in Bayesian approaches to parameter inference for ABMs (see e.g. Grazzini et al., 2017; Lux, 2018; Platt, 2020, 2021; Lux, 2021), with proponents of the Bayesian paradigm arguing in favour of its alternative approach to quantifying uncertainty and it ability to incorporate expert prior knowledge through the specification of prior distributions. Here, the primary object of interest is the parameter posterior distribution $p(\boldsymbol{\theta} \mid \mathbf{y})$, obtained via Bayes theorem:

$$p(\boldsymbol{\theta} \mid \mathbf{y}) = \frac{p(\mathbf{y} \mid \boldsymbol{\theta})}{p(\mathbf{y})} p(\boldsymbol{\theta}). \tag{3}$$

In Equation (3), $p(\boldsymbol{\theta})$ is referred to as the *prior distribution* over parameters $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, which encodes the experimenter's initial beliefs regarding appropriate values for $\boldsymbol{\theta}$ before data is observed, while the simulation model itself appears via its associated data likelihood function $p(\mathbf{y} \mid \boldsymbol{\theta})$. Importantly, Bayesian inference procedures seek to accurately approximate the entire distribution $p(\boldsymbol{\theta} \mid \mathbf{y})$, rather than any single point estimate derived from this distribution. In this way, accurate and meaningful uncertainty estimates may also be obtained which correctly account for the experimenter's initial beliefs, as well as the evidence provided by the data $\mathbf{y}$.

A well-known investigation into Bayesian estimation methods for ABMs is Grazzini et al. (2017), which explores various means of constructing a surrogate likelihood $\tilde{p}(\mathbf{y} \mid \boldsymbol{\theta})$ to account for the intractability of the true likelihood $p(\mathbf{y} \mid \boldsymbol{\theta})$. Two immediate drawbacks of the methods Grazzini et al. (2017) discuss are that they (a) are difficult to reconcile with the fact that ABMs are frequently dynamical models generating time-series output, and (b) entail a huge computational burden, since each likelihood evaluation in the employed posterior sampling algorithm requires at least one simulation from the model, and it is typically necessary to make many hundreds of thousands of such evaluations. More recent work by Platt (2021) suggests estimating the model's transition density via a mixture density network. While the approach offers some improvements through the use of a more flexible density estimator and the incorporation of some temporal dependencies, it suffers from similar drawbacks, assuming time-homogeneity and requiring a computationally expensive estimation step for every single sample from the approximate posterior distribution. In summary, there is a need for parameter inference methods that fulfil two main criteria:

1. they must be simulation-efficient in order to remain applicable to large-scale simulation models such as ABMs;
2. and they must be able to deal with non-homogenous/non-stationary temporal data, both simulated and observed.

To this end, we seek with this paper to analyse the utility of two classes of parameter inference methods that have seen significant activity within the computational statistics and probabilistic machine learning literature in recent years: neural posterior estimation

(Papamakarios and Murray, 2016; Lueckmann et al., 2017; Greenberg et al., 2019), and density ratio estimation (Thomas et al., 2021; Hermans et al., 2020; Durkan et al., 2020; Dyer et al., 2022b). Such methods have been employed successfully in a variety of applied domains, including high-energy physics (Brehmer et al., 2018), cosmology (Alsing et al., 2019), and neuroscience (Gonçalves et al., 2020). In addition, as we will demonstrate below, they work flexibly with potentially multivariate time-series data, requiring minimal model assumptions, and typically generate more accurate parameter inferences with a significantly reduced simulation budget in comparison to common alternatives, such as those reported in Grazzini et al. (2017).

We further seek to establish a higher standard of assessment criteria to be used in future benchmarking experiments for approximate Bayesian parameter inference for simulation models in economics and the social sciences through the use of common integral probability metrics and inference validation procedures used widely in statistics and machine learning. In Bayesian inference, the object of interest is the posterior density, whose entire shape is important since this captures degrees of beliefs regarding the unknown $\boldsymbol{\theta}$. However, the literature on Bayesian estimation of ABMs has largely forgone attempts to examine the degree of overlap between the estimated and ground-truth posteriors or the extent to which the inference procedure has learnt the dynamics of the data, often in favour of a simple comparison of certain point estimates of $\boldsymbol{\theta}$ (Grazzini et al., 2017; Platt, 2021; Shiono, 2021). Since one of the primary arguments put forward in favour of Bayesian inference is its ability to naturally quantify uncertainty in comparison to other methods, it is important to adopt assessment criteria that quantify the degree of closeness between the estimated and target posterior densities in benchmarking experiments.

In summary, our contributions with this article are to provide:

1. an overview of recent advances in black-box simulation-based Bayesian inference for simulation models, and a motivation for their use in the specific case of agent-based models;
2. a systematic benchmarking of these state-of-the-art methods against popular alternatives within the literature on Bayesian parameter inference for agent-based models in economics and the social sciences, in terms of their ability to recover the full posterior distribution;
3. a novel extension of the methods we present, permitting them to generate parameter posteriors for agent-based models when fine-grained, granular microdata is available on the modelled system. We demonstrate that this contribution enables agent-based modellers to calibrate their simulation models to graph- or graph-sequence-valued data, which arise ubiquitously when modelling a broad variety of complex social and economic systems such as social networks, labour markets, and production networks and supply chains.

2. An overview of approximate Bayesian inference methods for agent-based models

Bayesian inference has seen relatively little attention from the economic ABM community as a means to parameter estimation, but has gained popularity with the work of Grazzini et al. (2017). Here, the authors discuss three approaches to performing parameter inference with approximate Bayesian computation (ABC) (see e.g. Tavaré et al., 1997; Pritchard et al., 1999; Beaumont et al., 2002; Sunnåker et al., 2013; Schmon et al., 2020). One such proposal is the classical approach to ABC, in which samples are accepted or rejected from some proposal distribution according to some notion of discrepancy between the observed data and the output of the ABM at those parameter values. The remaining two entail the use of an explicit generative model that approximates the distribution of the output of the ABM, either through the use of a parametric density function – akin to synthetic likelihood (Wood, 2010; Price et al., 2018) – or through the use of a non-parametric kernel density estimation trained on the output of the ABM. Both approaches require the time-series generated by the ABM to be stationary. Other works, such as Lux (2018), Lux (2021), and Platt (2021), similarly perform a procedure that can be subsumed under ABC using different methods for approximating the likelihood function. For example, Lux (2018, 2021) use particle filters, while Platt (2021) assumes time-homogeneity of the transition density associated with the ABM, which is approximated with a mixture density network (Bishop, 1994) at each parameter value visited in their sampling procedure. This feature of Platt (2021) makes its application prohibitively expensive in many cases, since each step of the posterior sampling procedure – which must typically be at least of order 10^5 even in low-dimensional cases to achieve a low Monte Carlo error on the resultant posterior estimate – requires multiple simulations in addition to training of a neural network from scratch, which can quickly lead to simulation budgets of order 10^7 or greater. Lux and Zwinkels (2018) further discusses the use of ABC and particle filters for Bayesian inference, in addition to other non-Bayesian approaches such as maximum likelihood and MSM, discussed in the previous section. We provide further technical details on these approaches, and unify many of them under the umbrella of ABC, in Appendix A.

A common feature of these approaches is that the task of inference – that is, of constructing the posterior distribution – is inherently linked to the act of simulating from the ABM and, in most cases, building an additional approximate *generative* model (sometimes also called an *emulator*), in the sense that an approximation to the likelihood function is constructed. We will see in the following chapters that this contrasts with a new generation of Bayesian estimation methods in which inference is decoupled from the act of simulating and can be performed in a *discriminative* manner, typically resulting in more efficient inferences.

3. A new generation of simulation-based inference methods

In this section, we provide an overview of the two main simulation-based inference (SBI) methods we make use of in this paper: neural posterior estimation (NPE) and (neural) ratio estimation (NRE), both of which – as their names suggest – employ

neural networks[1] to obtain an estimate of the posterior density $p(\boldsymbol{\theta} \mid \mathbf{y})$. We motivate the use of these methods for agent-based models (ABMs) by framing them as *discriminative* approaches to simulation-based parameter inference, and contrasting them with the *generative* approaches described in Section 2 which seek to approximate the model's distribution (likelihood) using some form of probabilistic model. We then present the core elements of these methods, and further provide a discussion on how they may be leveraged flexibly and automatically to accommodate inference tasks involving high-dimensional and potentially multivariate time-series data, as is often the case in applications in economics and the social sciences.

3.1. Motivating black-box, discriminative approaches to parameter inference

The methods we endorse here – NPE and NRE – can be motivated by the fact that they are:

1. simulation-efficient alternatives to more traditional approaches to SBI, such as the approximate Bayesian computation (ABC) approaches described in Section 2;
2. generic SBI methods that treat the simulator as a black-box, thus making minimal assumptions about the structure and output of the simulation model;
3. discriminative approaches to SBI, in the sense that inference does not require a probabilistic model for the data $\mathbf{x}$: instead of *explaining* the mechanisms that create the data, we only need to *distinguish* between realisations that arise from different parameter values.

Simulation efficiency The algorithms described in Section 2 share a common pattern. For a fixed parameter $\boldsymbol{\theta}$, *iid* simulations $\mathbf{x}^{(r)} \sim p(\mathbf{x} \mid \boldsymbol{\theta})$, $r = 1, \ldots, R$, are sampled to produce a proxy likelihood $\hat{p}(\mathbf{x} \mid \boldsymbol{\theta})$. Then, to generate n approximate posterior samples via Markov chain Monte Carlo (MCMC), this procedure is performed at n different values for $\boldsymbol{\theta}$, where n must typically be a large number – often a few hundred thousand, if not orders or magnitude larger – to ensure a low Monte Carlo error. Consequently, at least nR simulations from the ABM are required in total for inference under these algorithms. Since ABMs can be very expensive to simulate, and the act of simulating from the ABM remains the primary bottleneck in Bayesian estimation procedures for ABMs, this simulation demand can quickly become infeasible.

In contrast, the methods for which we advocate here differ by eliminating the need to simulate when sampling from the posterior. Instead, they decouple the act of simulating from the task of constructing the posterior by employing powerful function approximators to learn – in essence – *global* posterior density estimators on the basis of a limited number of simulations from across the parameter space of the ABM; that is, they learn functions $h : \mathcal{Y} \times \boldsymbol{\Theta} \to \mathbb{R}$ which approximate the posterior density $p(\boldsymbol{\theta} \mid \mathbf{y})$ across the space of all possible values for $\boldsymbol{\theta}$ and $\mathbf{y}$. This has the potential to significantly reduce the simulation burden associated with approximate Bayesian inference procedures, because the pointwise estimates of $p(\boldsymbol{\theta} \mid \mathbf{y})$ can borrow strength from, and share information between, one another; in contrast, the algorithms described in Section 2 consider each pointwise evaluation of $p(\boldsymbol{\theta} \mid \mathbf{y})$ as standalone density estimation tasks, such that no information can be shared between them. In this way, a large number of approximate posterior samples can be generated from an algorithm trained on what can in practice be a far smaller number of model samples than is required for the algorithms described in Section 2.

Finally, the approaches we endorse here are equipped with a further efficiency benefit: *amortisation*. This phrase captures the fact that the global density estimators can generate samples from an approximate posterior $\hat{p}(\boldsymbol{\theta} \mid \mathbf{y})$ for any data $\mathbf{y}$ without the need to further simulate from the ABM, which results from the fact that they are trained on the full space of possible values for $\boldsymbol{\theta}$ *and* $\mathbf{y}$. In contrast, applying the methods described in Section 2 to a new dataset $\mathbf{y}'$ would entail nR further simulations from the ABM, multiplying the already high simulation costs.

Black-box inference methods We discussed in Section 2 that many Bayesian inference approaches for ABMs come with restrictive assumptions, for instance, stationarity of the simulated and observed time-series. In general, it is useful to dispense with these assumptions, since they can be difficult to verify and limit the applicability of these methods for arbitrary simulators. Instead, it can be preferable to employ black-box methods that make minimal assumptions about the model and are therefore generically applicable to arbitrary simulators. Doing so enables the modeller to concentrate resources on model design and implementation, rather than on developing bespoke inference algorithms for each new simulator. Furthermore, assumptions such as stationarity are known to be particularly poorly suited to certain simulation models such as ABMs, since these models are known and are even designed to produce non-equilibrium dynamics. Employing inference procedures that are able to handle such dynamics is therefore essential to the task of estimating generic ABMs.

Discriminative approaches to parameter inference Discriminative tasks in machine learning are typically simpler than generative tasks, since generative tasks address larger problems than pure discrimination. This is intuitive: for example, it is typically easier for humans and computers alike to distinguish between images of cats and dogs than to generate them. Analogously, it is generally a simpler task to discriminate between complex time-series data than it is to generate such time-series. The approaches described in Section 2 adopt a generative approach: they each – either explicitly or implicitly – seek to derive an approximation to the simulator's

[1] Note that neural networks are not essential here – the procedures we describe require only flexible function approximators. We frame the discussion around neural networks primarily because they are a convenient choice of flexible function approximators in many settings.

likelihood function using a probabilistic model, and thus seek to model the (probability density function of the) simulation output itself. Formally, this may be understood as learning the (stochastic) map $\boldsymbol{\theta} \mapsto \mathbf{x}$. NPE and NRE, in contrast, do not seek to model the simulation output: instead, they map from instances of the simulation output to certain target values which we will describe in more detail below. This can be formally thought of as learning the map $\mathbf{x} \mapsto \boldsymbol{\theta}$. Such an approach to parameter estimation therefore embodies a fundamental departure from the approaches described in Section 2. It has the potential to be particularly beneficial for ABMs, which are known to be able to produce complex, non-equilibrium dynamics that are especially difficult to model and generate.

3.2. Neural posterior estimation

In this section, we introduce neural conditional density estimators and their use in posterior estimation tasks. The core idea underlying this class of simulation-based Bayesian inference techniques is to use such conditional density estimators to model the parameter posterior density directly. While various conditional density estimators can be used, e.g. mixture density networks (Bishop, 1994; Papamakarios and Murray, 2016), we focus here on the case of **normalising flows** (Tabak and Vanden-Eijnden, 2010; Tabak and Turner, 2013; Rezende and Mohamed, 2015) due to their widespread use in various density estimation tasks, including in the areas of image generation (e.g. Kingma and Dhariwal, 2018) and physics (e.g. Noé et al., 2019).

3.2.1. Normalising flows

Normalising flows involve transforming a simple base distribution into a more complicated one, often with the use of neural networks. Consider a random variable $\mathbf{U} \sim p_{\mathbf{U}}$, where $p_{\mathbf{U}}$ is a probability distribution chosen to be a "simple" base distribution, in the sense that it is easy to both generate samples from $p_{\mathbf{U}}$ and to evaluate $p_{\mathbf{U}}(\mathbf{u})$ for any $\mathbf{u}$. Now consider the transformed random variable $\mathbf{X} = g(\mathbf{U})$, where g is a differentiable, invertible function with differentiable inverse $f := g^{-1}$. Then, by the change of variables formula, we have

$$p_{\mathbf{X}}(\mathbf{x}) = p_{\mathbf{U}}(f(\mathbf{x}))\left|\det J_f(\mathbf{x})\right| \tag{4}$$

where J_f is the Jacobian of f. Sampling from $p_{\mathbf{X}}$ then simply involves sampling a value $\mathbf{u}$ from the base distribution $p_{\mathbf{U}}$ and immediately obtaining $\mathbf{x} = g(\mathbf{u})$. Evaluating $p_{\mathbf{X}}(\mathbf{x})$ also then simply involves evaluation of the right-hand side of Equation (4). The above may be extended easily to a composition, or *flow*, $g = g_n \circ g_{n-1} \circ \ldots \circ g_1$ of differentiable and invertible functions g_i with inverses f_i, for which we now have that

$$p_{\mathbf{X}}(\mathbf{x}) = p_{\mathbf{U}}\left(f_1\left(\ldots f_{n-1}\left(f_n(\mathbf{x})\right)\ldots\right)\right)\left|\prod_{i=1}^{n}\det J_{f_i}(\mathbf{x})\right|. \tag{5}$$

Given samples from $p_{\mathbf{X}}$, the question of how to choose a base distribution $p_{\mathbf{U}}$ and a series of functions $g_1, \ldots, g_n$ such that the distribution of the samples is modelled accurately arises. The idea behind normalising flows is to *learn* this sequence of transformations, that is, to have each f_i be a flexible transformation f_{ϕ_i} with trainable parameters ϕ_i. The resulting density estimator q_ϕ can be written

$$q_\phi(\mathbf{x}) = p_{\mathbf{U}}\left(f_{\phi_1}\left(\ldots f_{\phi_{n-1}}\left(f_{\phi_n}(\mathbf{x})\right)\ldots\right)\right)\left|\prod_{i=1}^{n}\det J_{f_{\phi_i}}(\mathbf{x})\right|, \quad \text{where} \quad \phi = (\phi_n, \ldots, \phi_1). \tag{6}$$

The simple base distribution is then often taken to be a standard normal distribution[2] of the same dimension as the target distribution, and the flexible transformations f_{ϕ_i} are typically neural networks with a structure designed to guarantee the required invertibility and differentiability. Since the computation of the determinant of the Jacobian, which has appeared in the expressions written above, can in general be an intensive task, these neural networks are also designed such that obtaining the determinant is fast and straightforward. This can be achieved, for example, by using an architecture for the normalising flow that results in a triangular Jacobian, such that the determinant is simply the product of the main diagonal elements, as is the approach taken in e.g. Papamakarios et al. (2017).

The trainable parameters $\phi = \{\phi_i : 1 \le i \le n\}$ are optimised by maximising the log-likelihood of the data $\mathbf{x}^{(r)} \overset{iid}{\sim} p_{\mathbf{X}}, r = 1, \ldots, R$:

$$\hat{\phi} = \arg\max_{\phi} \sum_{r=1}^{R} \log q_\phi\left(\mathbf{x}^{(r)}\right). \tag{7}$$

Normalizing flows are typically implemented in machine learning libraries that support automatic differentiation such as `pytorch` (Paszke et al., 2019) or `TensorFlow` (Abadi et al., 2016) allowing the effective use of gradient-based optimisation techniques, such as Adam (Kingma and Ba, 2014). We provide a schematic of the sampling (flow, right-pointing arrows) and density evaluation (normalising flow, left-pointing arrows) processes in Fig. 1, in which the simple, left-most distribution is morphed successively into the more complex, right-most distribution.

[2] For this reason, the composition $f = f_1 \circ \ldots \circ f_{n-1} \circ f_n$ is often referred to as the *normalising* flow, since it is a flow of transformations resulting (typically) in a Gaussian base distribution.

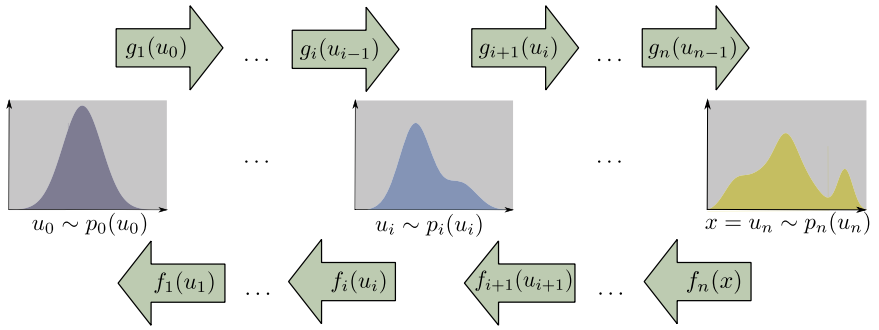


Fig. 1. Schematic of a normalising flow. Sampling (top) entails sampling u_0 from the base density p_0 and iteratively finding $u_i = g_i(u_{i-1})$. Density evaluation (bottom) entails sampling x from the true density of the data and finding the corresponding point u_0 in the base density through successive application of the $f_i = g_i^{-1}, i = n, \dots, 1$ and applying Equation (5).

3.2.2. Normalising flows for neural posterior estimation

Normalising flows may also be extended to the case of *conditional* density estimation (see e.g. Papamakarios and Murray, 2016; Lueckmann et al., 2017; Papamakarios et al., 2019; Greenberg et al., 2019). In this way, we can use a normalising flow to estimate the posterior density $p(\boldsymbol{\theta} \mid \mathbf{x})$ associated with a simulation model by following the same procedure as above and finding the neural network parameters as

$$\hat{\phi} = \arg\max_{\phi \in \Phi} \sum_{r=1}^{R} \log q_\phi \left(\boldsymbol{\theta}^{(r)} \mid \mathbf{x}^{(r)} \right), \tag{8}$$

where $(\mathbf{x}^{(r)}, \boldsymbol{\theta}^{(r)}) \overset{iid}{\sim} p(\mathbf{x}, \boldsymbol{\theta}), r = 1, \dots, R$. Importantly, this framework allows us to learn a *single* conditional density estimator for the posterior density function across data $\mathbf{x}$, rather than having to undergo the expensive procedure of training a separate density estimator for each point evaluation of the posterior density as in e.g. Platt (2021).

3.3. Density ratio estimation and neural density ratio estimation

An alternative to density estimation can be found leveraging the connection between supervised learning and unsupervised learning, using the often-termed *likelihood-ratio trick*. Given independent samples $x^{(r)} \sim f_0(\cdot), r = 1, \dots, R$ and $x^{(r)} \sim f_1(\cdot), r = R+1, \dots, 2R$, where we assign a classification label $c = 1$ to samples from f_1 and $c = 0$ to samples from f_0, then

$$\mathbb{P}(c = 1 \mid x) = \frac{f_1(x)}{f_0(x) + f_1(x)} = \frac{f_1(x)/f_0(x)}{1 + f_1(x)/f_0(x)},$$

where $\mathbb{P}$ is a probability measure over class labels. Thus, if we can find an estimate for the class membership probabilities, $\hat{g}(x) \approx \mathbb{P}(c = 1 \mid x)$, we can in turn find an estimate of the ratio:

$$\hat{r}(x) = \log \left(\frac{\hat{g}(x)}{1 - \hat{g}(x)} \right) \approx \frac{f_1(x)}{f_0(x)}.$$

While this is a relatively well known approach, it has only recently found use in SBI tasks. Pham et al. (2014) propose to use classifiers such as random forests to estimate $p(\mathbf{x} \mid \boldsymbol{\theta}_1)/p(\mathbf{x} \mid \boldsymbol{\theta}_0)$, in order to use this ratio in MCMC. Similarly, Cranmer et al. (2015) suggest the use of the same quantity for frequentist likelihood ratio approaches.

3.3.1. Likelihood-to-evidence ratio via binary classification

The idea developed above may be applied to perform **likelihood-to-evidence ratio estimation** (Thomas et al., 2021; Hermans et al., 2020), which can improve the simulation efficiency of the idea underlying the procedures described above. The core idea underlying this class of methods contrasts with NPE in that while NPE seeks to model the posterior density directly, (neural) ratio estimation (NRE) instead seeks to estimate the following ratio:

$$\frac{p(\mathbf{x}, \boldsymbol{\theta})}{p(\mathbf{x})\, p(\boldsymbol{\theta})} = \frac{p(\mathbf{x} \mid \boldsymbol{\theta})}{p(\mathbf{x})} = \frac{p(\boldsymbol{\theta} \mid \mathbf{x})}{p(\boldsymbol{\theta})} =: r(\mathbf{x}, \boldsymbol{\theta}). \tag{9}$$

This ratio is frequently referred to as the **likelihood-to-evidence ratio**. In the event that Bayesian inference is the goal, this then permits evaluation of the posterior density as

$$p(\boldsymbol{\theta} \mid \mathbf{x}) = r(\mathbf{x}, \boldsymbol{\theta}) p(\boldsymbol{\theta}). \tag{10}$$

Thomas et al. (2021) suggest to make use of this observation by estimating the likelihood-to-evidence ratio $p(\mathbf{x} \mid \boldsymbol{\theta})/p(\mathbf{x})$ at $\boldsymbol{\theta}$ by training a probabilistic classifier to distinguish between $\mathbf{x}^{(n)} \overset{iid}{\sim} p(\mathbf{x} \mid \boldsymbol{\theta}), n = 1, \dots, N$ and $\tilde{\mathbf{x}}^{(m)} \overset{iid}{\sim} p(\mathbf{x}), m = N+1, \dots, 2N$ at each proposed parameter value $\boldsymbol{\theta}$. The authors show that the optimal estimate of the class probability is

$$\mathbb{P}(c = 1 \mid \mathbf{y}) = \frac{p(\mathbf{y} \mid \boldsymbol{\theta})}{p(\mathbf{y} \mid \boldsymbol{\theta}) + p(\mathbf{y})}, \tag{11}$$

permitting an estimate of $r(\mathbf{y}, \boldsymbol{\theta}) := p(\mathbf{y} \mid \boldsymbol{\theta})/p(\mathbf{y})$ as $\mathbb{P}(c = 1 \mid \mathbf{y})/\mathbb{P}(c = 0 \mid \mathbf{y})$. This approach was later extended by Hermans et al. (2020), in which the authors propose to train a probabilistic classifier to instead distinguish between the following two sets of training examples:

1. a set of "genuine" examples drawn from the joint distribution, $(\mathbf{x}, \boldsymbol{\theta}) \overset{iid}{\sim} p(\mathbf{x} \mid \boldsymbol{\theta})p(\boldsymbol{\theta})$, which are assigned a class label $c = 1$;
2. a set of "false" examples drawn from the product of the marginals, $(\mathbf{x}, \boldsymbol{\theta}) \overset{iid}{\sim} p(\mathbf{x})p(\boldsymbol{\theta})$, with class label $c = 0$.

The difference between the two sets of examples is that the $\mathbf{x}$ are generated by the $\boldsymbol{\theta}$ to which they are paired in the former set of examples, while in the latter set of examples the $\mathbf{x}$ bear no relation to the $\boldsymbol{\theta}$ they are paired with.

The function of a probabilistic binary classifier trained on such data is to model the class probability $d(\mathbf{x}, \boldsymbol{\theta}) := \mathbb{P}(c = 1 \mid \mathbf{x}, \boldsymbol{\theta}) \in [0, 1]$; hard classification labels (i.e. decisions regarding the predicted value of c) are obtained when the continuous-valued $d(\mathbf{x}, \boldsymbol{\theta})$ is combined with a decision rule, for example the rule that $c = 1$ should be predicted whenever $d(\mathbf{x}, \boldsymbol{\theta}) > 0.5$. One can show (see Appendix B of Hermans et al., 2020) that the *optimal* estimate of $d(\mathbf{x}, \boldsymbol{\theta})$ is the value

$$d^*(\mathbf{x}, \boldsymbol{\theta}) := \frac{p(\mathbf{x} \mid \boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathbf{x} \mid \boldsymbol{\theta})p(\boldsymbol{\theta}) + p(\mathbf{x})p(\boldsymbol{\theta})}. \tag{12}$$

Thus, a good probabilistic classifier trained to distinguish between the two possible types of pairs $(\mathbf{x}, \boldsymbol{\theta})$ will learn a good estimate $\hat{d}(\mathbf{x}, \boldsymbol{\theta})$ of this ratio. Such a probabilistic classifier will allow us to evaluate the posterior density in this way: by noticing that one can rearrange Equation (12) as

$$p(\boldsymbol{\theta} \mid \mathbf{x}) = \frac{d^*(\mathbf{x}, \boldsymbol{\theta})}{1 - d^*(\mathbf{x}, \boldsymbol{\theta})} p(\boldsymbol{\theta}) =: r^*(\mathbf{x}, \boldsymbol{\theta})p(\boldsymbol{\theta}), \tag{13}$$

where $r^*(\mathbf{x}, \boldsymbol{\theta})$ is the corresponding estimate of the likelihood-to-evidence ratio $p(\mathbf{x} \mid \boldsymbol{\theta})/p(\mathbf{x})$. In practice, of course, only an approximation $\hat{d}(\mathbf{x}, \boldsymbol{\theta})$ will be obtained to the ratio in Equation (12), yielding a correspondingly imperfect estimate $\hat{r}(\mathbf{x}, \boldsymbol{\theta})$ of the likelihood-to-evidence ratio. Nonetheless, training high-capacity neural network classifiers on the cross-entropy loss

$$\ell\left(\left\{c^{(i)}, \mathbf{x}^{(i)}, \boldsymbol{\theta}^{(i)}\right\}_{i=1}^{N}\right) = -\frac{1}{N}\sum_{i=1}^{N}\left[c^{(i)} \log \hat{d}\left(\mathbf{x}^{(i)}, \boldsymbol{\theta}^{(i)}\right) + \left(1 - c^{(i)}\right) \log\left(1 - \hat{d}\left(\mathbf{x}^{(i)}, \boldsymbol{\theta}^{(i)}\right)\right)\right]$$

can yield classifiers with good probability estimates, and thus good estimates of $r^*(\mathbf{x}, \boldsymbol{\theta})$.

3.3.2. A generalisation to multi-class classification

In more recent work, Durkan et al. (2020) demonstrate that the above approach to density ratio estimation can be generalised to the problem of training a probabilistic classifier to identify the *correct* $(\mathbf{x}, \boldsymbol{\theta}^{(i)})$ pair from a batch $\{(\mathbf{x}, \boldsymbol{\theta}^{(b)})\}_{b=1}^{B}$ of B pairs that otherwise contain only "incorrect" pairs, where "correct" and "incorrect", respectively, correspond to having been drawn from the joint distribution $p(\mathbf{x}, \boldsymbol{\theta})$ and from the product of the marginal distributions $p(\mathbf{x})p(\boldsymbol{\theta})$. In this case, the optimal estimate of the correct class probability is

$$p\left(c = i \mid \mathbf{x}, \{\boldsymbol{\theta}^{(b)}\}_{b=1}^{B}\right) = \frac{p(\boldsymbol{\theta}^{(i)} \mid \mathbf{x}) \prod_{b \neq i} p(\boldsymbol{\theta}^{(b)})}{\sum_{b=1}^{B} p(\boldsymbol{\theta}^{(b)} \mid \mathbf{x}) \prod_{b' \neq b} p(\boldsymbol{\theta}^{(b')})} = \frac{p(\boldsymbol{\theta}^{(i)} \mid \mathbf{x})/p(\boldsymbol{\theta}^{(i)})}{\sum_{b=1}^{B} p(\boldsymbol{\theta}^{(b)} \mid \mathbf{x})/p(\boldsymbol{\theta}^{(b)})}. \tag{14}$$

Thus, by comparison with Equation (14), training a neural network f_ϕ on the loss function

$$\ell(\phi) = -\log \frac{\exp f_\phi(\mathbf{x}, \boldsymbol{\theta}^{(i)})}{\sum_{b=1}^{B} \exp f_\phi(\mathbf{x}, \boldsymbol{\theta}^{(b)})} \tag{15}$$

for each $(\mathbf{x}, \boldsymbol{\theta}) \sim p(\mathbf{x} \mid \boldsymbol{\theta})p(\boldsymbol{\theta})$ will induce $f_\phi(\mathbf{x}, \boldsymbol{\theta})$ to learn the value $f_\phi(\mathbf{x}, \boldsymbol{\theta}) = \log(p(\boldsymbol{\theta} \mid \mathbf{x})/p(\boldsymbol{\theta}))$, thus recovering an estimate of the desired density ratio. This can be extended to a batch of R "correct" data-parameter pairs as

$$\mathcal{L}(\phi) = -\frac{1}{R}\sum_{r=1}^{R} \log \frac{\exp f_\phi(\mathbf{x}^{(r)}, \boldsymbol{\theta}^{(r)})}{\sum_{b_r=1}^{B_r} \exp f_\phi(\mathbf{x}^{(r)}, \boldsymbol{\theta}^{(b_r)})}, \tag{16}$$

where the terms in the denominator are labelled with b_r to account for the possibility that the contrasting ("incorrect") set of parameters may be different for different "correct" pairs $(\mathbf{x}^{(r)}, \boldsymbol{\theta}^{(r)})$. The authors further demonstrate that this can yield more accurate density ratio estimators. For this reason, we adopt this approach throughout this article.

3.4. Sampling from the neural posterior

After successful training, the posterior density estimator $q_{\hat{\phi}}(\boldsymbol{\theta} \mid \mathbf{x})$ obtained from NPE is a parametric approximation of the true posterior distribution, which can then be used to generate *iid* samples $\boldsymbol{\theta} \sim q_{\hat{\phi}}(\boldsymbol{\theta} \mid \mathbf{x})$, as described in Section 3.2.1, or to evaluate

the posterior density on a pointwise basis, as also described in Section 3.2.1. While ratio estimation similarly permits a pointwise evaluation of the posterior distribution as $p(\boldsymbol{\theta})\exp(f_{\hat{\phi}}(\mathbf{x},\boldsymbol{\theta}))$, it requires a further inference step using, for example, Metropolis–Hastings (see Appendix E.1 for details) to generate samples from the same posterior. However, the upfront training of the ratio estimator eliminates the need for expensive sampling from the ABM, reducing the run-time significantly in comparison to alternative approaches such as those described in Section 2.

3.5. Round-based training

The procedures described in Sections 3.2 and 3.3 are framed as single density (ratio) estimation tasks. This means that a single dataset of R data-parameter pairs $(\mathbf{x}^{(r)},\boldsymbol{\theta}^{(r)}), r=1,\dots,R$ is created in the following way:

$$\boldsymbol{\theta}^{(r)} \sim p(\boldsymbol{\theta}), \qquad \text{sample from the prior;}$$
$$\mathbf{x}^{(r)} \sim p(\mathbf{x}\mid\boldsymbol{\theta}^{(r)}), \qquad \text{sample from the model using the draws from the prior.}$$

Subsequently, NRE and NPE algorithms are trained on the *whole dataset* to find either the likelihood-to-evidence ratio or the posterior directly following, respectively, (8) or (16). Density estimators trained in this way are referred to as *amortised* density estimators, which reflects the fact that they may be used to construct posterior distributions for any data $\mathbf{x}$ without further simulation from the ABM or the need to train multiple density estimators.

The disadvantage of this one-stage approach is that all training simulations from the ABM are generated by parameters drawn from the prior distribution. In many cases, however, interest lies only in the posterior for the *particular observation* $\mathbf{y}$, which can be concentrated on specific subregions of the entire space covered by the prior density. It can then be preferable to narrow down the search space of good candidate values for the parameter $\boldsymbol{\theta}$ to subregions of the parameter space that could most plausibly have generated $\mathbf{y}$. By doing so, the density (ratio) estimator will be presented with a less varied range of dynamics between which it must learn to distinguish, allowing it to develop a more refined approximation of the density (ratio) in regions of high posterior density. This can facilitate more rapid learning and potentially reduce the number of training examples that must be simulated by the ABM.

Significant effort has thus been extended towards the constructions of "round"-based approaches to training density (ratio) estimators for SBI (see e.g. Papamakarios et al., 2019; Greenberg et al., 2019; Hermans et al., 2020; Durkan et al., 2020). The idea is to split the total budget of R simulations into subsets of size $N_1, N_2, \dots$ such that $\sum_i N_i = R$. In the first step, N_1 data points are created as before and a posterior approximation $q_{\hat{\phi}}\left(\boldsymbol{\theta}\mid\mathbf{x}\right)$ is constructed. Subsequently, in round $i \geq 2$, we create new data $(\mathbf{x}^{(r)},\boldsymbol{\theta}^{(r)}), r=1,\dots,N_i$ using

$$\boldsymbol{\theta}^{(r)} \sim q_{\hat{\phi}}^{(i-1)}\left(\boldsymbol{\theta}\mid\mathbf{x}\right), \qquad \text{sample from round-}(i-1)\text{ posterior;}$$
$$\mathbf{x}^{(r)} \sim p(\mathbf{x}\mid\boldsymbol{\theta}^{(r)}), \qquad \text{sample from the model using the draws from the round-}(i-1)\text{ posterior.}$$

The posterior $q_{\phi}\left(\boldsymbol{\theta}\mid\mathbf{x}\right)$ may now be retrained using (a combination of the old and) the new samples, and the process can be repeated for as many rounds as necessary. An identical process exists for density ratio estimation, with the exception that parameters in round i are drawn using the likelihood-to-evidence ratio obtained from round $i-1$ and, for example, Metropolis–Hastings.

While the round-based approach is appealing, it does not come without additional challenges. In particular, the joint distribution of the example pairs $(\mathbf{x}^{(r)},\boldsymbol{\theta}^{(r)})$ is no longer $p(\mathbf{x},\boldsymbol{\theta})$ for data sampled after the first round. In practice, this needs to be accounted for during training by the use of additional weighing factors, such as those appearing in Equation (17) (cf. Equation (8)). We refer the interested reader to the following papers for further details on this matter: Papamakarios and Murray (2016); Lueckmann et al. (2017); Greenberg et al. (2019).

Training schemes for NPE and NRE which make use of this round-based training approach are referred to as *sequential* neural posterior estimation (SNPE) and *sequential* neural ratio estimation (SNRE), respectively, in which the prefix "sequential" reflects the round-based training design of the now non-amortised density (ratio) estimator.

3.6. Summarising the simulation output

As described above, NPE and NRE take as input either $\mathbf{x}$ alone or $(\mathbf{x},\boldsymbol{\theta})$ pairs. Given that $\mathbf{x}$ is a (possibly multivariate) time-series for many ABMs and is thus a high-dimensional object, it can be beneficial to incorporate useful inductive biases[3] into the network architecture that account for the sequential nature of $\mathbf{x}$ and reduce its dimensionality. To this end, the density (ratio) estimator can be prefixed with a so-called embedding network with trainable parameters φ, whose function is to consume the original high-dimensional dataset $\mathbf{x}$ and express this as a lower-dimensional summary statistic vector $\mathbf{s}_{\varphi}(\mathbf{x})$. The parameters of the embedding network and of the density (ratio) estimator may then be learned concurrently using the same loss function, offering a means to automatically learn descriptive low-dimensional features of the raw input data during the *same* posterior or density ratio estimation task in an end-to-end fashion.

[3] Inductive biases are usually realised as constraints on the form of the network architecture in the case of neural networks. For example, when certain symmetries are known to be present in the data, this inductive bias may be incorporated with the use of (partially) exchangeable neural networks (Zaheer et al., 2017; Wiqvist et al., 2019). Imposing such constraints helps to restrict the space over which appropriate functions are searched for and facilitates effective learning and generalisation from training data.

This quality is a further attractive feature of NPE and NRE. Finding low-dimensional summary statistics of the data is necessary in many approaches to SBI,[4] and the question of how to summarise data naturally arises in response to this. Popular solutions that have been explored to-date include: the experimenter themselves devising a collection of hand-crafted summary statistics that they believe adequately describes the data, which is an arduous task that can lead to a difficult-to-quantify loss of information; or learning summary statistics by constructing a separate learning task to the existing problem of density (ratio) estimation (see e.g. Fearnhead and Prangle, 2012; Chen et al., 2020), imposing an additional burden on the experimenter. NPE and NRE, in contrast, offer a convenient approach to incorporating inductive biases and to learning summary statistics in an end-to-end fashion without the additional complications associated with alternative SBI techniques. However, when trusted hand-crafted summary statistics $\mathbf{s}(\mathbf{x})$ are available, these can either be used in place of, or in addition to, learned summary statistics. In the latter case, this is easily achieved by either (a) concatenating the two kinds of summary statistics into a single vector $\mathbf{v}_\varphi(\mathbf{x}) = (\mathbf{s}(\mathbf{x}), \mathbf{s}_\varphi(\mathbf{x}))$, or (b) learning lower-dimensional summary statistics from the hand-crafted summary statistics as $\mathbf{v}(\mathbf{x}) = \mathbf{s}_\varphi(\mathbf{s}(\mathbf{x}))$, enabling the user to flexibly make use of both trusted user expertise and learned features.

3.7. Training procedures for density (ratio) estimators

To summarise and conclude this section, we follow Greenberg et al. (2019) and Durkan et al. (2020) and provide in Algorithm 1 procedures for training neural posterior and density ratio estimators for SBI over multiple rounds. In both cases, we assume that the density (ratio) estimator includes any embedding network used to summarise data concurrently with the density (ratio) estimate, such that ϕ contains the parameters of the estimator and the embedding network.

Algorithm 1: Training SNPE (see Greenberg et al., 2019, Algorithm 1) and SNRE (see Durkan et al., 2020, Algorithm 1).

Input: prior distribution $p(\boldsymbol{\theta})$, simulator $p(\mathbf{x} \mid \boldsymbol{\theta})$, observation $\mathbf{y}$, conditional density estimator q_ϕ/density ratio estimator f_ϕ, number of rounds M, number of simulations per round N, minibatch size B, contrasting set size K;
Result: Trained conditional density/density ratio estimator
Set $\tilde{p}_0(\boldsymbol{\theta}) = p(\boldsymbol{\theta})$, dataset $\mathcal{D} = \{\}$;
for $m = 0, \dots, M-1$ **do**
  Sample $\boldsymbol{\theta}^{(n)} \overset{iid}{\sim} \tilde{p}_m(\boldsymbol{\theta})$, $n = 1, \dots, N$;
  Simulate $\mathbf{x}^{(n)} \sim p\left(\mathbf{x} \mid \boldsymbol{\theta}^{(n)}\right)$, $n = 1, \dots, N$;
  Append the dataset:

$$\mathcal{D} := \mathcal{D} \cup \bigcup_{n=1}^{N} \left\{ \left(\mathbf{x}^{(n)}, \boldsymbol{\theta}^{(n)}\right) \right\};$$

  if SNPE **then**
    until *convergence* **do**
      Evaluate the loss function

$$\mathcal{L}(\phi) = -\sum_{(\mathbf{x},\boldsymbol{\theta}) \in \mathcal{D}} \log\left(q_\phi(\boldsymbol{\theta} \mid \mathbf{x}) \frac{\tilde{p}_m(\boldsymbol{\theta})}{p(\boldsymbol{\theta})} \frac{1}{Z(\mathbf{x}, \phi)} \right); \tag{17}$$

      Update trainable parameters ϕ on the basis of $\mathcal{L}(\phi)$
    end
    Set $\tilde{p}_{m+1}(\boldsymbol{\theta}) := q_\phi(\mathbf{y}, \boldsymbol{\theta})$.
  end
  if SNRE **then**
    until *convergence* **do**
      Sample minibatch $\left\{ \left(\mathbf{x}^{(b)}, \boldsymbol{\theta}^{(b)}\right) \right\}_{b=1}^{B}$ uniformly from $\mathcal{D}$;
      For each minibatch pair, draw $0 < K < B$ parameters $\tilde{\boldsymbol{\theta}}^{(k)}$, $k = 1, \dots, K$, from elsewhere in the training data $\mathcal{D}$;
      Evaluate the finite-sample multinomial logistic loss

$$\mathcal{L}(\phi) = -\frac{1}{B} \sum_{b=1}^{B} \log \frac{\exp\left(f_\phi\left(\mathbf{x}^{(b)}, \boldsymbol{\theta}^{(b)}\right)\right)}{\exp\left(f_\phi\left(\mathbf{x}^{(b)}, \boldsymbol{\theta}^{(b)}\right)\right) + \sum_{k=1}^{K} \exp\left(f_\phi\left(\mathbf{x}^{(b)}, \tilde{\boldsymbol{\theta}}^{(k)}\right)\right)}; \tag{18}$$

      Update trainable parameters ϕ on the basis of $\mathcal{L}(\phi)$
    end
    Set $\tilde{p}_{m+1}(\boldsymbol{\theta}) \propto \exp\left(f_\phi(\mathbf{y}, \boldsymbol{\theta})\right)$.
  end
end

[4] This is the prototypical approach to ABC, for example, although methods that obviate the need to summarise high-dimensional data have been proposed and explored in recent years (see e.g. Park et al., 2016; Bernton et al., 2019; Dyer et al., 2021a).

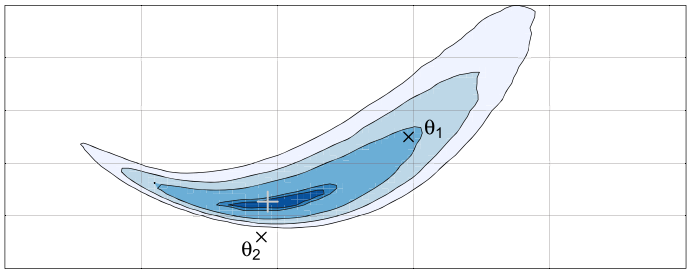


Fig. 2. Visualisation: why standard Euclidean distances are misleading when assessing Bayesian inference algorithms. The parameter θ_2 is ostensibly closer to the "true parameter" (grey) in this toy posterior than θ_1. However, θ_1 has higher posterior density, i.e. it is more credible given the data.

4. Experiments for tractable examples

In this section, we present experiments in which we compare the ability of NPE and NRE to estimate parameter posterior distributions for economic simulation models with tractable likelihood functions against common alternatives in the ABM literature. We also provide details of the neural network architecture and training hyperparameters in Appendix G. All code for these experiments is available at https://github.com/joelnmdyer/sbi4abm.

4.1. Performance metrics

Each example presented in this section will be equipped with a tractable transition density, and therefore a tractable likelihood function. Such models are thus amenable to Bayesian inference via standard means, such as MCMC, to obtain an approximate ground-truth posterior density.[5]

To assess the accuracy of the estimated posteriors in this case, we compare the full ground-truth posterior density with the full posterior estimated with each of the implemented simulation-based approaches. This contrasts with previous studies of Bayesian parameter estimation for ABMs, in which point estimates alone are often used to assess the quality of the tested inference procedures. For example, Platt (2021) uses a prior-weighted Euclidean distance between the estimated posterior mean and generating parameters (that is, the $\boldsymbol{\theta}$ that generated the pseudo-observation $\mathbf{y}$) as a performance metric. However, such metrics provide a very limited and at times misleading view on the outcome of the inference process, because (a) "closeness" is not measured in the geometry of the target distribution, as visualised in Fig. 2; (b) it is possible that an approximate posterior can produce a posterior mean close to the generating parameter but simultaneously under- or over-estimate the width of the distribution or otherwise yield poor uncertainty quantification, and (c) finite datasets are not guaranteed to yield posterior densities that concentrate on the generating parameter.

To compare the ground-truth and simulation-based posteriors, we compute two integral probability metrics which each correspond to notions of *dissimilarity* between these posteriors:

Wasserstein distance This is a distance measure derived from optimal transport theory (Kantorovich, 1960) and used widely in various machine learning contexts, e.g. generative adversarial networks (Arjovsky et al., 2017). For some distance ρ_0 on $\boldsymbol{\Theta}$ and two probability measures μ and μ', the p-Wasserstein metric between two sets of samples $\{\boldsymbol{\theta}_i\}_{i=1}^n \overset{iid}{\sim} \mu$ and $\{\boldsymbol{\theta}'_i\}_{i=1}^m \overset{iid}{\sim} \mu'$ is computed as

$$\mathcal{W}_p\left(\{\boldsymbol{\theta}_i\}_{i=1}^n, \{\boldsymbol{\theta}'_j\}_{j=1}^m\right)^p = \inf_{\gamma\in\Gamma_{n,m}} \sum_{i=1}^n \sum_{j=1}^m \rho_0(\boldsymbol{\theta}_i, \boldsymbol{\theta}'_j)^p \gamma_{ij} \tag{19}$$

where $\Gamma_{n,m}$ is the set of non-negative $n\times m$ matrices with columns (resp. rows) summing to m^{-1} (resp. n^{-1}). Throughout, we use the Euclidean distance for ρ_0 and term this metric WASS.

Maximum mean discrepancy (MMD) This metric is once again a metric on probability distributions that draws from the theory of reproducing kernel Hilbert spaces (Gretton et al., 2006, 2012) and is used widely within the machine learning and simulation-based inference community to assess the dissimilarity between probability distributions (Papamakarios et al., 2019; Lueckmann et al., 2021). Here, under a suitable choice of kernel[6] k chosen by the experimenter, the discrepancy between two probability distributions P and Q is taken to be

$$\text{MMD}(P,Q) = \left\| \mathbb{E}_{\theta\sim P}\left[k(\boldsymbol{\theta},\cdot)\right] - \mathbb{E}_{\theta'\sim Q}\left[k(\boldsymbol{\theta}',\cdot)\right] \right\|_{\mathcal{H}}^2, \tag{20}$$

[5] While the true posterior density is targeted with such a sampling scheme, the ground-truth obtained in this fashion remains an approximation due to the Monte Carlo error associated with the finite sample size.

[6] That is: a symmetric, positive semi-definite function $\boldsymbol{\Theta}\times\boldsymbol{\Theta}\to\mathbb{R}$.

where $\mathcal{H}$ is the reproducing kernel Hilbert space (RKHS) associated with k and $\mathbb{E}_{\theta\sim P}[k(\theta,\cdot)]$ is the so-called *mean embedding* of P via kernel k. When only samples $\{\theta_i\}_{i=1}^n \overset{iid}{\sim} P$ and $\{\theta'_i\}_{i=1}^m \overset{iid}{\sim} Q$ are available, an unbiased estimator of this metric can be computed as

$$\widehat{\mathrm{MMD}}(P,Q) = \frac{1}{m(m-1)}\sum_{\substack{i,j\\ j\neq i}} k(\theta'_i,\theta'_j) + \frac{1}{n(n-1)}\sum_{\substack{i,j\\ i\neq j}} k(\theta_i,\theta_j) - \frac{2}{nm}\sum_{i,j} k(\theta_i,\theta'_j). \tag{21}$$

Throughout, we use a Gaussian kernel as k,

$$k(\theta,\theta') = \exp\left(-\frac{\|\theta-\theta'\|_2^2}{2\sigma^2}\right), \tag{22}$$

where $\sigma^2 = \mathrm{median}\{\|\tilde{\theta}_i - \tilde{\theta}_j\|_2^2\}$ following Briol et al. (2019), and $\{\tilde{\theta}_i\}_{i=1}^n$ are samples from the ground-truth posteriors.

4.2. Example 1: the Brock and Hommes model (1998)

We consider a variant of the model proposed by (Brock and Hommes, 1998), which has previously been used in ABM calibration experiments (Platt, 2020). The model dynamics can be expressed as the following system of coupled equations:

$$\begin{aligned} x_{t+1} &= \frac{1}{R}\left[\sum_{h=1}^{H} n_{h,t+1}\left(g_h x_t + b_h\right) + \epsilon_{t+1}\right], \qquad \epsilon_t \sim \mathcal{N}(0,\sigma^2),\\ n_{h,t+1} &= \frac{\exp\left(\beta\left(x_t - Rx_{t-1}\right)\left(g_h x_{t-2} + b_h - Rx_{t-1}\right)\right)}{\sum_{h'=1}^{H}\exp\left(\beta\left(x_t - Rx_{t-1}\right)\left(g_{h'} x_{t-2} + b_{h'} - Rx_{t-1}\right)\right)}, \end{aligned} \tag{23}$$

where R,β,σ are parameters. Following Platt (2020), we take $\beta = 120$, $H = 4, R = 1.01, \sigma = 0.04, g_1 = b_1 = b_4 = 0$ and $g_4 = 1.01$. We consider the task of estimating the posterior $p(\theta \mid \mathbf{y})$, where $\theta = (g_2, b_2, g_3, b_3)$, $\mathbf{y} := (y_1, y_2, \ldots, y_T) \sim p(\mathbf{x} \mid \theta^*)$ is the pseudo-observation, $T = 100$, and $\theta^* = (g_2^*, b_2^*, g_3^*, b_3^*) = (0.9, 0.2, 0.9, -0.2)$ is the parameter used to generate $\mathbf{y}$. To do so, we use the following uniform priors: $g_2, b_2, g_3 \sim \mathcal{U}(0,1)$, while $b_3 \sim \mathcal{U}(-1,0)$. In what follows, we provide further details on each method we compare, before discussing experimental results.

4.2.1. Neural posterior estimation and neural ratio estimation

For NPE and NRE, we use a round-based training approach i.e. SNPE and SNRE: we train over 10 rounds and generate 1000 simulations per round. We consider two versions of NPE and NRE:

1. for the first, we summarise manually with the mean value, variance, maximum, minimum, median, 25th quantile, 75th quantile, and the autocorrelations of the x_t to lags 1, 2, and 3. We denote these statistics with $\tilde{\mathbf{s}}$ and term them *naive* or *hand-crafted* summary statistics;
2. for the second, we learn them as part of the training process with an embedding network $\mathbf{s}_\varphi$ with trainable parameters φ (see Section 3.6). While a plethora of candidate network architectures can be used that incorporate useful inductive biases for time-series data (e.g. Wong et al., 2018; Dyer et al., 2021b; Kidger et al., 2020), we use an embedding network consisting of two stacked Elman recurrent units with hidden state of size 32, followed by a single linear layer of size 16. Below, we refer to summary statistics obtained in this fashion as *learned* summary statistics.

4.2.2. Approximate Bayesian computation

We compare NPE and NRE to common alternatives to approximate Bayesian inference in the economic ABM literature, many of which – as discussed in Section 2 (and expanded upon in Appendix A) – can be seen as instances of approximate Bayesian computation (ABC). In ABC, the ABM's unknown likelihood function is approximated in the following way:

$$\hat{p}(\mathbf{y} \mid \theta) \propto \int K_\epsilon(\mathbf{y},\mathbf{x})\, p(\mathbf{x} \mid \theta)\, \mathrm{d}\mathbf{x}, \tag{24}$$

where K_ϵ is a (possibly unnormalised) kernel function, with parameter(s) ϵ, providing a measure of "distance" between observed data $\mathbf{y}$ and simulated data $\mathbf{x}$. This permits an approximation $\hat{p}(\theta \mid \mathbf{y})$ of the posterior density by using $\hat{p}(\mathbf{y} \mid \theta)$ in place of $p(\mathbf{y} \mid \theta)$ in Bayes's theorem, Equation (3).

As a first comparison, we consider the non-parametric density estimation approach discussed in Grazzini et al. (2017), which we term KDE-ABC. Here, the choice

$$\hat{p}(\mathbf{y} \mid \theta) \approx K_\epsilon(\mathbf{x},\mathbf{y}), \quad \text{where} \quad K_\epsilon(\mathbf{x},\mathbf{y}) = \prod_{t=1}^{T} \hat{p}_\epsilon\left(y_t \mid \theta, \mathbf{x}\right) \quad \text{and} \quad \mathbf{x} \sim p(\mathbf{x} \mid \theta) \tag{25}$$

is used, where $\hat{p}_\epsilon\left(y_t \mid \theta, \mathbf{x}\right)$ estimates $p\left(y_t \mid \theta, \mathbf{x}\right)$ with kernel density estimation, using a Gaussian kernel κ_ϵ and bandwidth ϵ chosen using Silverman's method (Silverman, 1986):

$$\hat{p}_\epsilon(y_t \mid \boldsymbol{\theta}, \mathbf{x}) = \frac{1}{T}\sum_{s=1}^{T} \kappa_\epsilon\left(y_t - x_s\right), \qquad \text{where} \qquad \kappa_\epsilon\left(y_t - x_s\right) = \frac{1}{\epsilon}\exp\left(-\frac{\|y_t - x_s\|_2^2}{\epsilon^2}\right). \tag{26}$$

To target the ABC posterior corresponding to this choice of kernel function K_ϵ, we sample with Metropolis-Hastings (MH) (see Appendix E.1 for further details).

We consider two further variants of ABC in addition to KDE-ABC, both of which correspond to different choices for the kernel K_ϵ in Equation (24). The first approach is based on the generalised subtracted l-divergence (GSL)[7] (Lamperti, 2018b), denoted as d_{GSL}, which is a popular approach within the economic ABM calibration literature for assessing the discrepancy between two univariate time series by comparing the temporal patterns that occur in the different series. It was previously used in optimisation-centric calibration procedures in Lamperti (2018b) and Platt (2020) to obtain point estimates of $\boldsymbol{\theta}^*$. In our setting, however, we use the discrepancy measure in a Bayesian fashion by taking

$$K_\epsilon^{\text{GSL}}(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{d_{\text{GSL}}(\mathbf{x}, \mathbf{y})}{\epsilon}\right) \tag{27}$$

in Equation (24), resulting in an ABC posterior that uses the GSL as a distance measure. Additionally, we consider a Bayesian version of Method of Simulated Moments (MSM), in which the Euclidean distance d_{MSM} between the summary statistics described above in Section 4.2 is used in place of d_{GSL} above, giving

$$K_\epsilon^{\text{MSM}}(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{d_{\text{MSM}}(\mathbf{x}, \mathbf{y})}{\epsilon}\right) = \exp\left(-\frac{\|\tilde{\mathbf{s}}(\mathbf{x}) - \tilde{\mathbf{s}}(\mathbf{y})\|}{\epsilon}\right). \tag{28}$$

This latter choice closely aligns to classical ABC (Tavaré et al., 1997), which computes distances between hand-crafted summary statistics and was also discussed in Grazzini et al. (2017). We sample from the ABC posteriors implied by the kernels in Equations (27) and (28) with sequential Monte Carlo approximate Bayesian computation (SMC-ABC) (Beaumont et al., 2009), a state-of-the-art approach to sampling from ABC posteriors. To perform SMC-ABC, we use an initial population of 10^3 particles and decay the value of the ABC tolerance hyperparameter ϵ by a factor of 0.8 at each step. We term these two approaches GSL-ABC and MSM-ABC, respectively.

4.2.3. The approximate ground-truth posterior

By rewriting the system in Equation (23) defining the Brock & Hommes model, we are able to find the transition density for observation y_{t+1} as

$$\begin{aligned} p(y_{t+1} \mid y_{1:t}, \boldsymbol{\theta}) &= \mathcal{N}\left(y_{t+1}; f(y_{t-2:t}, \boldsymbol{\theta}), \frac{\sigma^2}{R^2}\right), \quad \text{with} \\ f\left(y_{t-2:t}, \boldsymbol{\theta}\right) &= \frac{1}{R}\sum_{h=1}^{H} \frac{\exp\left[\beta\left(y_t - Ry_{t-1}\right)\left(g_h y_{t-2} + b_h - Ry_{t-1}\right)\right]}{\sum_{h'=1}^{H}\exp\left[\beta\left(y_t - Ry_{t-1}\right)\left(g_{h'} y_{t-2} + b_{h'} - Ry_{t-1}\right)\right]}\left(g_h y_t + b_h\right). \end{aligned} \tag{29}$$

Using this, we can evaluate the model's likelihood function numerically and obtain samples from an approximate ground truth posterior using MH.

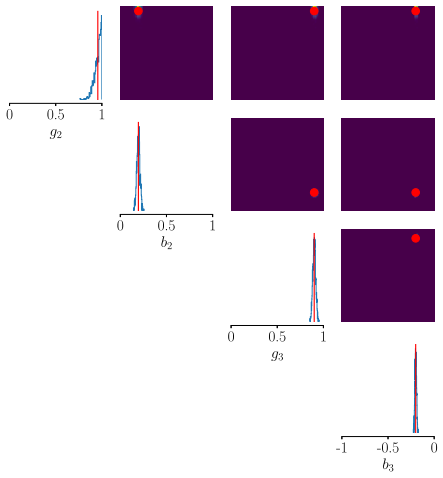
4.2.4. Results

In Figs. 3 and 4, we show the posteriors obtained using different posterior estimation methods: Fig. 3a shows the approximate ground-truth posterior; Figs. 3b, 3c, and 3d show the posteriors obtained with KDE-ABC, GSL-ABC, and MSM-ABC, respectively; Figs. 4a and 4b show the posterior estimated via SNPE with naive and learned summary statistics, respectively; and Figs. 4c and 4d show the posterior obtained via SNRE and naive and learned summary statistics, respectively, which were also sampled using MH. In each figure, the marginals and joint bivariate densities are located on the diagonal and upper diagonal, respectively, while the red lines/dots locate the mean of the true posterior.

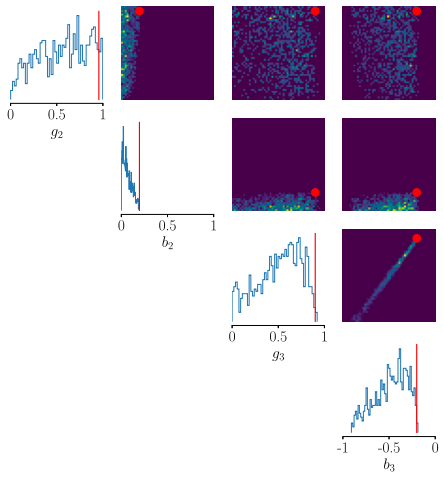
We see from the approximate ground-truth in Fig. 3a that the marginal posteriors are sharply peaked on the generating parameters for b_2, g_3, and b_3, while the ground truth marginal for g_2 is shifted towards higher values. While these features are recovered well for Figs. 4a–4d (the most notable exception being the posteriors for g_2), they are recovered poorly with KDE-ABC, GSL-ABC, and MSM-ABC. This performance gap is also manifested in the WASS and MMD metrics reported in Table 1, where lower values indicate better estimates of the posterior. In summary, KDE-ABC both requires a far larger simulation budget to estimate a single posterior, while simultaneously generating worse estimates of that posterior. In contrast, SNPE and SNRE achieve superior posterior approximations, despite the 10-fold reduction in the simulation budget they are afforded.

We further report on the robustness of these results over multiple trials and to the expressivity of the networks comprising NPE and NRE. Specifically, we decrease the capacity of the neural networks comprising NPE and NRE and repeat the entire inference procedure 20 times at different seeds for the same fixed $\mathbf{y}$ (see Appendix C.1 for further details). Fig. 5 shows boxplots for the WASS distance and MMDs resulting from this exercise. Numbers appearing in parentheses in the vertical axes indicate the number of simulations used by the method. Importantly, NPE and NRE – shown with dark blue and medium blue boxplots – used 10^4 simulations, while KDE-ABC, GSL-ABC, MSM-ABC used 10^5 simulations.

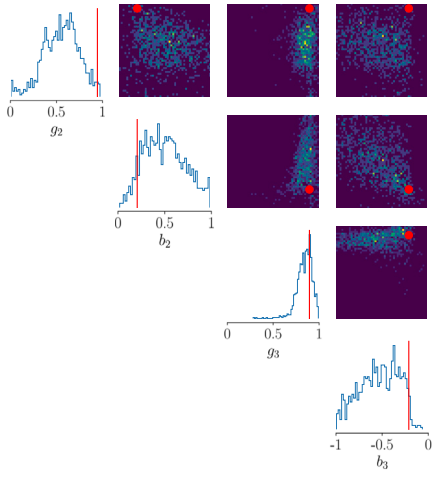
[7] This is implemented in the `black-it` software package (Benedetti et al., 2022).



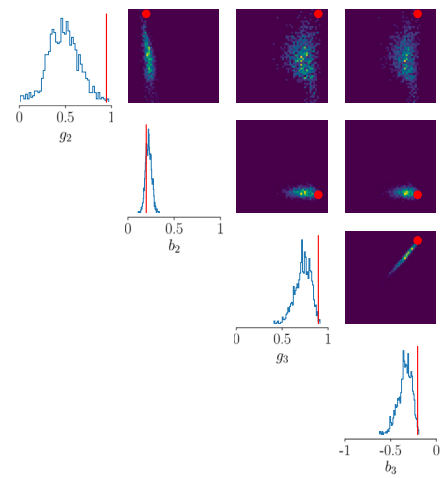
(a) True posterior



(b) KDE posterior, 1.5×10^5 simulations



(c) GSL-ABC posterior, 10^5 simulations



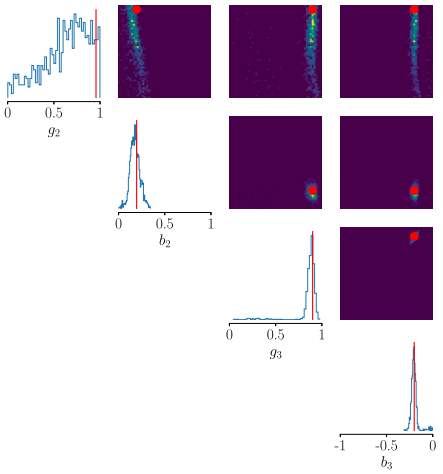
(d) MSM-ABC, 10^5 simulations

Fig. 3. (**Brock & Hommes**) Posteriors obtained with the ABC methods from Section 4.2.2. Marginal posterior distributions (blue curves) are located on the diagonals; bivariate joint distributions for each parameter pair are located on the off-diagonal. Darker (resp. lighter) colours indicate lower (resp. higher) posterior density. Red lines/dots show the mean of the true posterior. (For interpretation of the colours in the figure(s), the reader is referred to the web version of this article.)

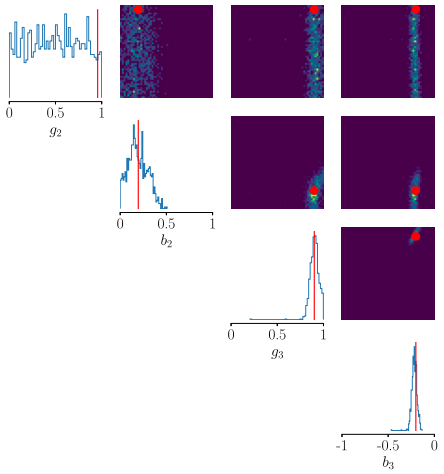
Table 1
(Brock & Hommes) Discrepancies between the approximate ground-truth posterior and the posteriors estimated with the methods described in Section 4.2. **Budget** denotes the total number of simulations used by each posterior estimation method. **Bold** and *italics* indicate best and second-best, respectively. For the neural methods, * indicates that the naive hand-crafted summary statistics described in the main text were used, otherwise summary statistics are learned from the simulated data.

Metric	Estimation method						
	KDE-ABC	GSL-ABC	MSM-ABC	SNPE*	SNPE	SNRE*	SNRE
WASS	0.690	0.695	0.541	0.336	0.477	*0.299*	**0.241**
MMD	1.015	1.024	1.268	*0.552*	0.789	0.781	**0.451**
Budget	1.5×10^5	10^5	10^5	10^4	10^4	10^4	10^4

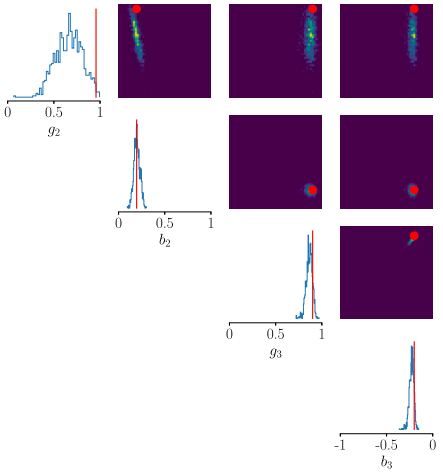
From Fig. 5, we see that KDE-ABC, GSL-ABC, and MSM-ABC are consistently outperformed by NPE and NRE. The distributions of WASS distances and MMDs obtained with NPE with either learned or hand-crafted summary statistics are entirely non-overlapping with those of KDE-ABC, GSL-ABC, and MSM-ABC. Additionally, the WASS distances and MMDs for NRE with hand-crafted and learned summary statistics are noticeably shifted towards lower values: all 20 runs of NRE with hand-crafted summary statistics produced



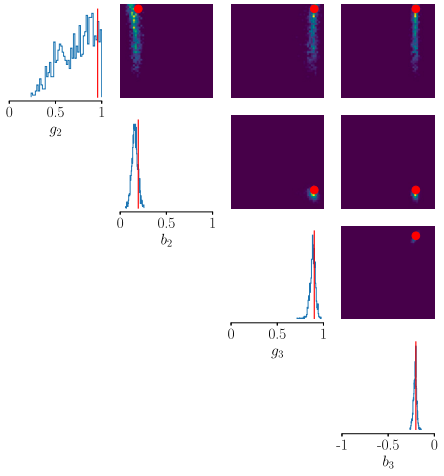
(a) Sequential neural posterior estimation with naive summary statistics, 10^4 simulations



(b) Sequential neural posterior estimation with learned summary statistics, 10^4 simulations

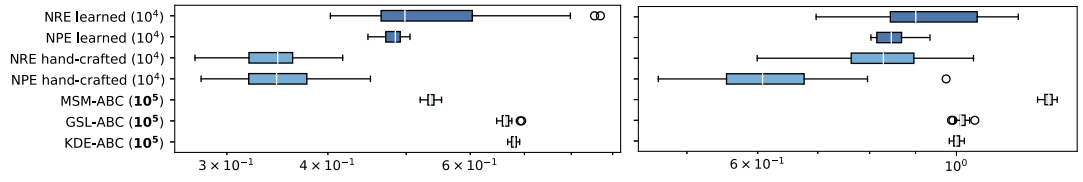


(c) Sequential neural ratio estimation with naive summary statistics, 10^4 simulations



(d) Sequential neural ratio estimation with learned summary statistics, 10^4 simulations

Fig. 4. (**Brock & Hommes**) Posteriors obtained with NPE and NRE. Marginal posterior distributions (blue curves) are on the diagonals; bivariate joint distributions for each parameter pair are on the off-diagonals. Darker (resp. lighter) colours indicate lower (resp. higher) posterior density. Red lines/dots show the mean of the true posterior.



(a) Wasserstein distances **(b)** Maximum mean discrepancies

Fig. 5. (**Brock & Hommes**) Boxplots for the (a) WASS distances and (b) MMDs for each method, obtained by running the entire inference procedure 10 times for the same fixed **y**.

WASS distances better than the best WASS metrics from each of the ABC methods, and 18 out of 20 runs produced better MMDs than the best MMD score produced by the ABC methods. Thus, while NRE with learned summary statistics has occasionally produced worse WASS distances and MMDs at 10^4 simulations than KDE-ABC, GSL-ABC, and MSM-ABC have done at 10^5 simulations, it is more likely than not that NRE will improve significantly on KDE-ABC, GSL-ABC, and MSM-ABC even when summary statistics are learned from scratch. In addition, techniques such as ensembling (see e.g. Cannon et al., 2022) and methods for checking the quality of inferences

(see Section 5.2 below and Appendix D.1) can be used to detect and guard against unusually poor performance resulting from the marginally greater variability observed in NPE and NRE, which we explain in terms of the increased stochasticity of the training procedures for the neural networks relatively to the sampling procedures for KDE-ABC, GSL-ABC, and MSM-ABC.

4.3. Example 2: multivariate geometric Brownian motion

In Section 4.2, we demonstrated that NPE and NRE can be used to generate Bayesian parameter posteriors that better match the ground truth posterior than KDE-ABC for the univariate Brock & Hommes model, despite the fact that the latter method entailed 10 times as many simulations as the former two methods based on neural networks. In this section, we seek to demonstrate that this remains the case even for models that generate multivariate time-series as output.

To this end, we consider a multivariate geometric Brownian motion (MVGBM), which is used in a variety of applications in financial time-series modelling. The model is a stochastic differential equation in which each of the d components, labelled $i \in \{1, 2, \dots, d\}$, of the path $\mathbf{X}_t = \left(X_t^1, X_t^2, \dots, X_t^d\right) \in \mathbb{R}_+^d$ evolve according to

$$\mathrm{d}X_t^i = X_t^i \left(\left[b_i - \frac{1}{2} \sum_{j=1}^{d} \sigma_{ij}^2 \right] \mathrm{d}t + \sum_{j=1}^{d} \sigma_{ij} \, \mathrm{d}W_t^j \right), \tag{30}$$

where the b_i are drift coefficients, σ_{ij} are volatility coefficients, and W_t^i is a Brownian motion. This model is generally non-stationary and non-ergodic (Peters and Klein, 2013), which makes inference a challenging task and highlights the applicability of NPE and NRE to such data.

We consider the case of $d = 3$ and the task of estimating the posterior for the parameters $\boldsymbol{\theta} = (b_1, b_2, b_3)$ given an observation $\mathbf{y} \sim p\left(\mathbf{x} \mid \boldsymbol{\theta}^*\right)$ of $T = 100$ points spaced equally with spacing $\Delta t = 1/(T-1)$, where $\boldsymbol{\theta}^* = (0.2, -0.5, -0.1)$. We take priors $b_i \sim \mathcal{U}(-1, 1)$ for each $i = 1, 2, 3$. We note that this model once again permits both exact simulations and samples from the exact posterior since the transition density is once again tractable[8] and can be written as

$$\mathbf{X}_{t+\Delta t} = \exp \mathbf{Z}_{t+\Delta t}, \quad \text{with} \quad \mathbf{Z}_{t+\Delta t} \sim \mathcal{N}\left(\log \mathbf{X}_t + (\boldsymbol{\theta} - \boldsymbol{\gamma})\,\Delta t,\ \sigma\sigma^T \Delta t\right) \tag{31}$$

and where $\exp \mathbf{Z}_{t+\Delta t}$ denotes exponentiation of the elements of $\mathbf{Z}_{t+\Delta t}$ and

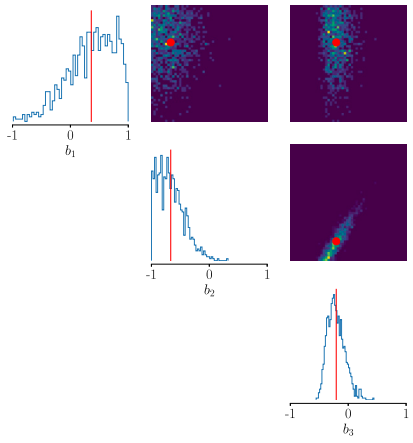
$$\boldsymbol{\gamma} = \frac{1}{2}\left[\sum_{j=1}^{d} \sigma_{1j}^2,\ \sum_{j=1}^{d} \sigma_{2j}^2,\ \sum_{j=1}^{d} \sigma_{3j}^2\right]', \qquad \text{where we fix} \qquad \sigma = \begin{pmatrix} 0.5 & 0.1 & 0.0 \\ 0.0 & 0.1 & 0.3 \\ 0.0 & 0.0 & 0.2 \end{pmatrix}. \tag{32}$$

In Fig. 6a we show the approximate ground truth posterior obtained with MH and the transition density described above, while in Figs. 6b, 6c, and 6d we show the posteriors obtained with KDE-ABC and MH, NPE with learned summary statistics, and NRE with MH also with learned summary statistics, respectively. The corresponding simulation budgets are 10^6, 10^3, and 10^3, respectively. To learn the summary statistics, we use two stacked gated recurrent units (GRUs) with hidden state size 32 followed by a feedforward network with layer sizes 32, 16 as the embedding network, and train as in Algorithm 1. Since KDE-ABC requires stationarity of the time-series, we take first-differences of both $\mathbf{y}$ and $\mathbf{x}$ for this method. No requirements on stationarity of ergodicity are imposed by NPE and NRE, however; we therefore leave $\mathbf{y}$ and $\mathbf{x}$ unaltered for NPE and NRE. Note that GSL-ABC cannot be applied in this experiment since the GSL is undefined for multivariate time series (Lamperti, 2018b). We see that the approximate ground-truth posteriors are relatively diffuse, with peaks approximately coinciding with the true posterior mean, shown with red lines/dots. The shape and degree of diffuseness is captured accurately by NPE and NRE. In contrast, the posterior obtained with KDE-ABC is insufficiently diffuse and biased, and thus a significantly worse estimate of the ground-truth posterior.

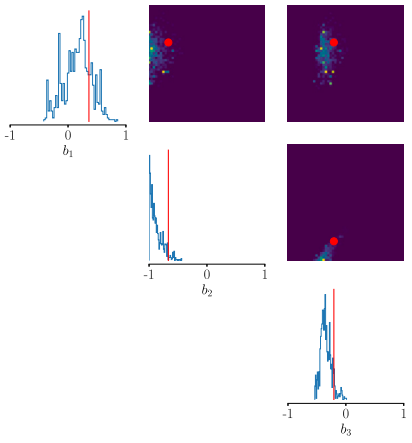
In Fig. 7, we show boxplots for the distribution of WASS distances and MMDs obtained by repeating the inference procedure for NPE, NRE, and KDE-ABC 10 times for different seeds and using the same fixed observed data set $\mathbf{y}$. Details on the network architectures are provided in Appendix C.2. Numbers in parentheses in the vertical axes in Fig. 7 show the number of simulations used by each method. From Figs. 7a and 7b, we see that NPE and NRE improve significantly on KDE-ABC with 1/1000th of the simulation budget: the distributions of WASS distances and MMDs for NPE and NRE with 10^3 simulations do not overlap at all with the distributions of WASS distances and MMDs generated by KDE-ABC when this uses 10^6 simulations from the model. This demonstrates that NPE and NRE can immediately perform well when the model is non-stationary and non-ergodic, while KDE-ABC fails to perform when the model is multivariate and/or non-stationary, even when first-differences of the simulated and observed time series has been applied in an attempt to eliminate the non-stationarity of the output series.[9] While the variance in these metrics is slightly (approximately two times) larger for NPE and NRE than for KDE-ABC – which again results from both the stochasticity of the neural network training procedure and the more limited training data with which NPE and NRE are provided in comparison to KDE-ABC – NPE and NRE continue to uniformly outperform KDE-ABC. In summary, NPE and NRE achieve significantly more accurate posterior estimates here with a 1000-fold decrease in the simulation budget, and are able to flexibly accommodate multivariate, non-ergodic, non-stationary time-series as input for the inference problem.

[8] Assuming that σ is of full rank.

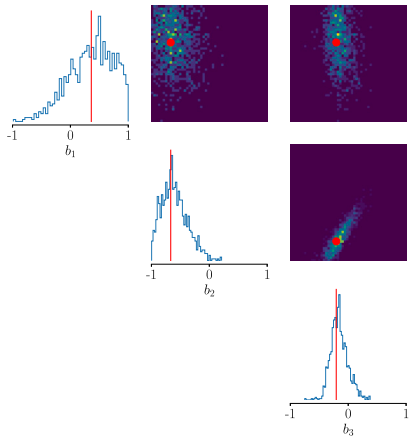
[9] Equally poor performance was observed from KDE-ABC without application of this first-differencing.



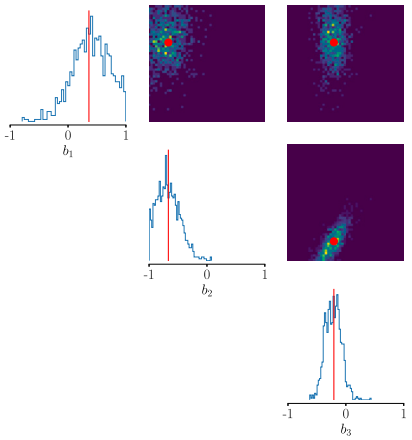
(a) True posterior



(b) KDE-ABC posterior, 10^6 simulations

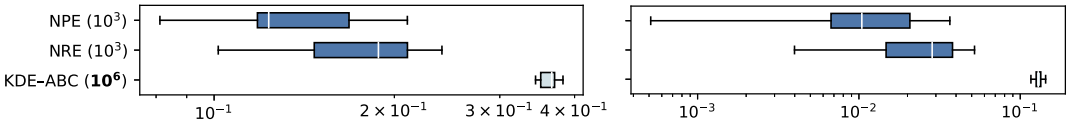


(c) Neural posterior estimation with automatically learned summary statistics, 10^3 simulations



(d) Neural ratio estimation with automatically learned summary statistics, 10^3 simulations

Fig. 6. (**Multivariate geometric Brownian motion**) Posteriors obtained with each estimation method. Marginal posterior distributions (blue curves) are on the diagonals; bivariate joint distributions for each parameter pair are on the off-diagonal. Darker (resp. lighter) colours indicate lower (resp. higher) posterior density. Red lines/dots show the mean of the true posterior.



(a) Wasserstein distances (b) Maximum mean discrepancies

Fig. 7. (**Multivariate geometric Brownian motion**) Boxplots for the (a) WASS distances and (b) MMDs for each method, obtained by running the entire inference procedure 10 times for the same fixed $\mathbf{y}$.

5. Approximate Bayesian inference from network data

In the previous experiments, we demonstrated that NPE and NRE can result in more accurate and simulation-efficient Bayesian parameter inferences than common alternatives in the ABM literature. We saw that this improvement in performance could be realised in a variety of settings, where the time series generated by the simulation model is stationary and univariate, or non-stationary, non-ergodic, and vector-valued (i.e. where $\mathbf{y}_t, \mathbf{x}_t \in \mathbb{R}^d$ for $d > 1$). We described and demonstrated that the modeller may draw inferences from such high-dimensional time series data by either (a) reducing such data to a comparatively low-dimensional vector of hand-crafted summary statistics that the modeller believes to be informative about the parameters to be inferred, (b) automatically finding such summary statistics through the use of an embedding network, or (c) incorporating both expert knowledge and learned knowledge from the embedding network by using a combination of the two approaches.

However, in many modelling problems of relevance to ABMs, and as ABMs become increasingly data-driven, the data that is available on the complex networked system being modelled by the ABM can be very granular, and may not consist only of a sequence of aggregate, vector-valued statistics describing the macroscopic state of the system as it evolves over time. Instead, the data that is available from the real world system may be a *graph*, or a *sequence of* graphs. This might be the case, for example, when modelling social networks (Leskovec and Krevl, 2014), patent networks and innovation (Pichler et al., 2020), or supply chain and production networks, either through direct access to such data or as the state of the art in production network reconstruction improves (Mungo et al., 2023).

As discussed in Fagiolo et al. (2019) and Axtell and Farmer (2022), the need for methods for systematically including such microlevel data into agent-based modelling accompanies the increased availability of this data. Yet, approaches to performing parameter inference for ABMs when fine-grained graph- or graph-sequence-valued data is available from the real world system are currently lacking in the ABM calibration literature. Importantly, modellers lack approaches to parameter inference that incorporate useful inductive biases that reflect the natural (dynamic) graph structure of this data, and this absence of methods can prevent agent-based modellers from properly capitalising on the availability and full information content of granular data of this kind. In this section,[10] we begin to remedy this issue by proposing a novel approach to approximate Bayesian parameter inference for ABMs and other complex networked systems, based on NPE and NRE, that operates directly on graph- or graph-sequence-valued data of the kind that can arise when modelling complex networked systems. Our proposal represents an initial step towards addressing the shortcomings raised in Fagiolo et al. (2019) and Axtell and Farmer (2022) by integrating independent software ecosystems that address (social) network analysis on the one hand, and interfacing data with agent-based modelling on the other.

5.1. Method

As described above, we consider data from both the real world and the ABM that takes the form of a graph or sequence of graphs. That is, the data is of the form $\mathbf{y} = (\mathbf{z}, \mathbf{w})$, where $\mathbf{z} = (\mathbf{z}_1, \dots, \mathbf{z}_T), \mathbf{z}_t \in \mathcal{Z} \subseteq \mathbb{R}^{N \times K}$ are the K-dimensional states of a set of N nodes over time $1 \leq t \leq T$, and $\mathbf{w} = (\mathbf{w}_1, \dots, \mathbf{w}_T), \mathbf{w}_t \in \mathcal{W} \subseteq \mathbb{R}^{N \times N}$ are the adjacency matrices corresponding to the potentially evolving graph, which may in general be weighted and directed. $\mathcal{Z}$ and $\mathcal{W}$ are the set of values the $\mathbf{z}_t$ and $\mathbf{w}_t$ can assume. To be able to derive a parameter posterior directly from such graph- or graph-sequence-valued data – rather than from an aggregate time-series that summarises this fine, granular data – we propose to construct a neural posterior estimator in which a graph neural network (GNN) g_ϕ – or a *recurrent* GNN in the case of $T > 1$ – and a neural conditional density estimator q_φ are paired to approximate the map $(\mathbf{z}, \mathbf{w}) \mapsto p(\cdot \mid \mathbf{z}, \mathbf{w})$, where ϕ and φ are the parameters of the respective neural networks and $p(\cdot \mid \mathbf{z}, \mathbf{w})$ is the posterior density function over parameters $\boldsymbol{\theta}$ obtained when conditioning on the graph-(sequence-)valued data, $(\mathbf{z}, \mathbf{w})$. The posterior $p(\cdot \mid \mathbf{z}, \mathbf{w})$ may then be approximated directly from the high-dimensional sequence of graphs as $q_\varphi(\cdot \mid g_\phi(\mathbf{z}, \mathbf{w}))$, where the parameters φ and ϕ are trained simultaneously with Algorithm 1. We provide an overview of GNNs and recurrent GNNs – which are neural network architectures that are designed to operate on graphs and sequences of graphs, respectively – in Appendix F.

5.2. Example: the Hopfield model of social dynamics

To illustrate the proposed method, we consider an inference task based on the Hopfield model of social dynamics proposed by Macy et al. (2003), which describes the coevolution of opinions and the social network structure, and the emergence of polarisation, in a population of N agents. At each time step $t = 1, \dots, T$, each agent is equipped with $N - 1$ undirected ties to the remaining agents in the population, and the strength and valence of the tie between agents i and j is characterised by $w_{tij} \in [-1, 1]$. Each agent is also equipped with a state vector $\mathbf{z}_{ti} = (z_{ti1}, \dots, z_{tiK}) \in \{-1, 1\}^K, i = 1, \dots, N$, which represent the opinion of agent i on each of a number $K \geq 1$ of topics at time t. The model proceeds by setting $z_{tik} = 1$ if

$$\frac{1}{1 + e^{-\rho \cdot P_{tik}}} > 0.5 + \epsilon U_{ti}, \quad \text{where} \quad P_{tik} = \frac{1}{N-1} \sum_{j \neq i} w_{tij} z_{tik} \quad \text{and} \quad U_{ti} \sim \mathcal{U}(-0.5, 0.5), \tag{33}$$

and $\epsilon \in [0, 1]$ and $\rho > 0$ are two free parameters of the model. Denoting with $\lambda \in [0, 1]$ a third free parameter, the ties between agents evolve as

$$w_{(t+1)ij} = (1 - \lambda) w_{tij} + \frac{\lambda}{K} \sum_{k=1}^{K} z_{(t+1)ik} z_{(t+1)jk}. \tag{34}$$

In this inference task, we assume the goal of approximating a posterior density for $\boldsymbol{\theta} = (\rho, \epsilon, \lambda)$ having observed the agent-based model *in its entirety*; that is, we take both the agent states $\mathbf{z} = (\mathbf{z}_1, \dots, \mathbf{z}_T)$ and the inter-agent tie-strengths $\mathbf{w} = (\mathbf{w}_1, \dots, \mathbf{w}_T)$ over all $T = 50$ time steps – rather than some aggregate time-series derived from these microscopic details – as the output of the model, such that $\mathbf{x} = (\mathbf{z}, \mathbf{w}) \sim p(\cdot \mid \boldsymbol{\theta})$. In this form, the output data $\mathbf{x}$, as well as the pseudo-true dataset $\mathbf{y}$, takes the form of a sequence of weighted, undirected snapshots of an evolving graph, in which the nodes have a set of $K \geq 1$ attributes contained in the state vectors $\mathbf{z}_{ti}$. As described previously, this inference problem mirrors inference problems faced in real life when attempting to model social interaction data (see e.g. Leskovec and Krevl, 2014) with ABMs or other stochastic simulation models.

[10] The work we present in this section extends our previous unpublished work Dyer et al. (2022a), which appeared as a Spotlight Paper in the inaugural *Artificial Intelligence for Agent-based Modelling* Workshop at the 2022 International Conference on Machine Learning, where it won one of three Best Paper Awards.

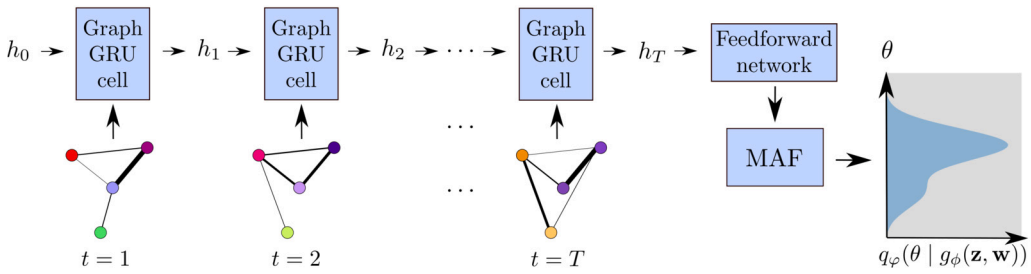


Fig. 8. A schematic of our posterior estimation pipeline. The data/output of the ABM – shown as the dynamic graph with evolving node states (node colours) $\mathbf{z}$ and edge weights (line widths) $\mathbf{w}$ – is embedded into a low-dimensional space with a graph convolutional GRU and a feedforward network applied to the GRU's final hidden state, h_T. This representation, $g_\phi(\mathbf{z}, \mathbf{w})$, is fed to the masked autoregressive flow to estimate the posterior as $q_\varphi(\boldsymbol{\theta} \mid g_\phi(\mathbf{z}, \mathbf{w}))$.

The specific implementation of the general method we propose in Section 5.1 that we use in this experiment is as follows: we take g_ϕ to be a feedforward network applied to the final hidden state of a graph convolutional GRU (Seo et al., 2018) – which is a graph generalisation of the GRU network that we have used earlier in this paper – and q_φ to be a masked autoregressive flow (Papamakarios et al., 2017). Our implementation is depicted graphically in Fig. 8, and further details on the network architectures and training procedure are provided in Appendix G. We train this composite network using a budget of 1000 simulations from the ABM described above, and assume prior densities $\rho \sim \mathcal{U}(0, 5)$, $\epsilon \sim \mathcal{U}(0, 1)$, and $\lambda \sim \mathcal{U}(0, 1)$. The inference task is performed for a pseudo-true dataset $\mathbf{y} \sim p(\mathbf{y} \mid \boldsymbol{\theta}^*)$ generated at ground-truth parameter values $\boldsymbol{\theta}^* = (1, 0.8, 0.5)$, and the number of agents in the system is taken to be $N = 25$.

5.2.1. Assessing the quality of the approximate inference procedure: posterior predictive checks

Having trained the posterior estimator described in the previous section, we proceed to assess the quality of the resultant posterior, which is an important step in approximate Bayesian inference pipelines. While in previous experiments we have assessed the quality of posteriors by comparing them to approximate ground-truth values – obtained from MCMC schemes that target the true posterior – we consider here approaches to assessing the quality of the posterior by the practitioner in practice, when there is of course no ground-truth posterior available. Although there exist multiple approaches to doing so (such as through *simulation-based calibration* (Talts et al., 2020); we refer the interested reader to Appendix D.1 for further details and a demonstration of this procedure), here we assess the quality of the recovered posterior with the use of posterior predictive checks (PPCs) (see e.g. Section 6.3, Gelman et al., 1995).

PPCs are a tool for validating Bayesian inference procedures that captures the notion that, if the inferences that have been drawn about plausible parameter values $\boldsymbol{\theta}$ based on the observed data $\mathbf{y}$ are accurate, then the posterior distribution $p(\boldsymbol{\theta} \mid \mathbf{y})$ should assign high probability mass to regions of the parameter space that tend to generate dynamics similar to $\mathbf{y}$. In other words, samples from the posterior predictive distribution,

$$p(\tilde{\mathbf{y}} \mid \mathbf{y}) = \int_{\boldsymbol{\Theta}} p(\tilde{\mathbf{y}} \mid \boldsymbol{\theta})\, p(\boldsymbol{\theta} \mid \mathbf{y})\, \mathrm{d}\boldsymbol{\theta}, \tag{35}$$

should be in some sense similar to $\mathbf{y}$, and $\mathbf{y}$ should look like a "typical" data point amongst samples from (35). Here, $\tilde{\mathbf{y}}$ may denote either future values obtained by simulating forward from the final value in $\mathbf{y}$, or may be taken as hypothetical repetitions of $\mathbf{y}$. Validating approximate Bayesian inference in this way then corresponds simply to repeatedly performing the following set of actions:

1. Sample a parameter from the approximate posterior, $\boldsymbol{\theta} \sim p(\boldsymbol{\theta} \mid \mathbf{y})$;
2. Simulate a dataset from the model at that parameter value, $\tilde{\mathbf{y}} \sim p(\tilde{\mathbf{y}} \mid \boldsymbol{\theta})$;
3. Store $\tilde{\mathbf{y}}$ for later comparison with $\mathbf{y}$.

Once the $J \geq 1$ posterior predictive samples $\mathcal{S} := \{\tilde{\mathbf{y}}^{(j)}\}_{j=1}^{J}$ have been generated, a comparison between $\mathbf{y}$ and $\mathcal{S}$ may be performed through, for example, a visual inspection by plotting (summary statistics of) $\mathbf{y}$ and the $\tilde{\mathbf{y}}^{(j)}$, or by identifying appropriate scoring rules or test statistics for hypothesis testing (Section 6.3, Gelman et al., 1995).

In our experiment, we take the former approach and perform PPCs by visualising the distribution of two simple statistics of posterior predictive samples $\tilde{\mathbf{y}} = (\tilde{\mathbf{z}}, \tilde{\mathbf{w}})$:

1. the number of balanced triads in the (signed) adjacency matrix $\boldsymbol{\omega} \in \{-1, 1\}^{N \times N}$ with elements $\boldsymbol{\omega}_{ij} = \mathrm{sign}(\tilde{\mathbf{w}}_{Tij})$, capturing the degree to which the adage "the enemy of my enemy is my friend" is reflected in the final network $\mathbf{w}_T$ and computed as

$$C = \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{k=1}^{N} \boldsymbol{\omega}_{ik} \boldsymbol{\omega}_{kj} \boldsymbol{\omega}_{ki}.$$

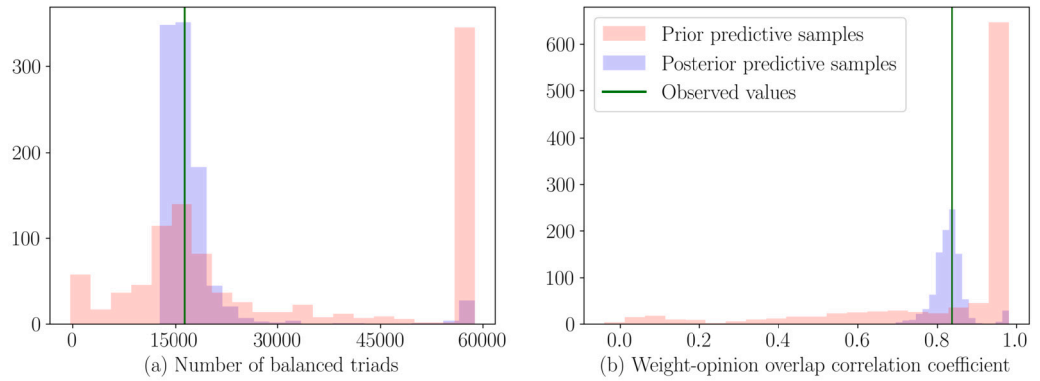


Fig. 9. (**Social dynamics model**) Distributions of statistics of prior (red) and posterior (blue) predictive samples. Observed statistics are shown with the vertical green line.

2. the linear correlation coefficient between the final tie-strengths $\tilde{w}_{Tij}$ and the final opinion overlap value, $\sum_{k=1}^{K} \tilde{z}_{Tik}\tilde{z}_{Tjk}$.

We plot the first and second of these two statistics in the left and right subplots, respectively, of Fig. 9 for samples from both the posterior predictive and the prior predictive distributions,

$$p(\tilde{\mathbf{y}}) = \int_{\boldsymbol{\Theta}} p(\tilde{\mathbf{y}} \mid \boldsymbol{\theta})\, p(\boldsymbol{\theta})\, \mathrm{d}\boldsymbol{\theta}, \tag{36}$$

where $p(\boldsymbol{\theta})$ is the prior distribution described in Section 5.2. We also show the values of these statistics obtained from the pseudo-true dataset $\mathbf{y}$. From this, we see that the observed value of these two statistics is more typical under the posterior predictive samples than it is under the prior predictive samples, suggesting that the posterior estimator has accurately learned how to assign probability mass to appropriate regions of the parameter space.

6. Conclusion

In this paper, we investigated the use of two recently developed approaches to Bayesian parameter estimation for intractable simulation models: neural posterior estimation (NPE) for approximating the posterior density directly, and (neural) ratio estimation (NRE) for approximating the likelihood-to-evidence ratio. We motivated the use of these parameter inference methods as simulation-efficient black-box discriminative approaches to Bayesian estimation for complex time-series simulators such as agent-based models in economics and the social sciences, which contrast with existing methods that (a) entail an often prohibitively large simulation burden, (b) can make restrictive assumptions about the form of the simulator, for example by ignoring important structural features such as temporal dependencies, and (c) are inherently generative, and thus assume a task that is typically more difficult than discrimination alone. We argue that this latter point is likely to be particularly pertinent for the case of agent-based models in economics and the social sciences, since they are known to be able to generate complex and stochastic non-equilibrium dynamics that can be difficult to model and generate.

We further reviewed existing alternatives, and benchmarked[11] NPE and NRE against one of the most popular of these. In these examples, we see that NPE and NRE generally achieve superior recovery of the approximate ground-truth posterior distributions, despite requiring simulation budgets that are orders of magnitude lower than traditional alternatives. In addition, we demonstrated that a verification of the accuracy of the density (ratio) estimators is possible with the introduced methods via simulation-based calibration (SBC) (see Appendix D.1), due to the amortisation of the estimators, and furthermore by other well-known approaches such as posterior predictive checks (PPCs). In contrast to NPE and NRE, the former of these – SBC – would necessitate an unreasonably large number of simulations when existing methods for Bayesian estimation of large-scale simulation models in economics and the social sciences are used, for example the methods discussed by Grazzini et al. (2017). For these reasons, we argue that such simulation-efficient black-box Bayesian inference techniques as NPE and NRE may enable economists and social scientists alike to more readily exploit the potential benefits of the agent-based modelling paradigm, due to the dramatic increases in accuracy and correspondingly dramatic decreases in the simulation burden seen with these methods.

6.1. Limitations, challenges, and future work

We consider some limitations and challenges faced in simulation-based Bayesian parameter inference, and how these challenges relate to the methods we discuss in this article.

[11] Code available at https://github.com/joelnmdyer/sbi4abm.

Choice of neural network architecture Previously, we have discussed how the methods we endorse in this article are able to flexibly accommodate different kinds of data and simulators by choosing a network architecture that incorporates appropriate inductive biases – network architectures that reflect the specific structure of the simulator, or the data generated by the simulator. While this offers considerable flexibility to the practitioner, and can result in powerful posterior estimators and eliminate the need to manually construct summary statistics of the data, it nonetheless leaves open the precise choice of architecture and network design. It may be argued, therefore, that a degree of arbitrariness remains, and that the problem of designing an appropriate set of summary statistics that describe the data has simply been replaced with the similarly difficult task of designing an appropriate neural architecture. We submit, however, that (a) the flexibility and (b) the reduction in the required simulation burden that results from the use of NPE and NRE outweighs this drawback:

(a) there exists a vast and rapidly growing literature in which the performance of different neural network architectures is thoroughly tested and compared, which will provide the practitioner with expert guidance on appropriate network architecture choices. Furthermore, there is significant interest and progress being made in developing neural network architectures that are robust to complex and messy data settings – such as neural controlled differential equations (Kidger et al., 2020) for irregularly sampled data with missing values, neural rough differential equations (Morrill et al., 2021) for extremely long time-series, or temporal graph networks (Rossi et al., 2020) for streams of edges in social networks – which are of great practical relevance to economists and computational social scientists seeking to use such data to calibrate agent-based models (ABMs);
(b) in all methods, experimentation with hyperparameters will be required (e.g. network architecture for NPE and NRE; choice of distance function in approximate Bayesian computation (ABC); choice of kernel in KDE-ABC etc.). The key point is that, when more traditional methods such as KDE-ABC are used to approximate the parameter posterior, such experiments will entail a number of simulations that is orders of magnitude larger than the number of simulations that will likely be required for corresponding experiments with NPE and NRE. When the cost of simulating is large – as it typically will be for large-scale ABMs in economics and the social sciences – this will allow more rapid iterations and an ability to appropriately assess the posteriors/posterior estimators.

Robustness to misspecification Throughout this article, we have implicitly assumed that the simulator accurately describes reality. More precisely, we have assumed throughout that the simulator is a *well-specified* model, meaning that there exists some $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ such that $p(\cdot \mid \boldsymbol{\theta}) = p_*$, where p_* is the density from which the observed data $\mathbf{y}$ was drawn, $\mathbf{y} \sim p_*$. In practice, however, models will by definition be misspecified – every model necessarily omits some aspect of reality – such that for any $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, $p(\cdot \mid \boldsymbol{\theta}) \neq p_*$. For the modeller, it is then a question of how poorly specified the model is, and whether any inference procedure employed for e.g. parameter calibration can appropriately handle this misspecification. This is an active area of research for inference procedures in general (see e.g. Miller and Dunson, 2018; Knoblauch et al., 2019; Schmon et al., 2020), and is a problem relevant to all inference procedures (i.e. not only NPE and NRE). There is currently some work on assessing or improving the robustness of NPE and NRE to model misspecification – see Cannon et al. (2022) and Ward et al. (2022) for recent examples – but further work is needed to understand how well these methods work to misspecified settings.

Expense of training The methods we endorse in this article – NPE and NRE – have been seen to entail simulation burdens that are typically orders of magnitude smaller than more traditional parameter inference methods for ABMs. However, NPE and NRE entail additional training times that are not shared by some other techniques. In general, we submit that, for large-scale ABMs, this training time will likely often be a small proportion of the time spent simulating; this is particularly true when the practitioner has access to specialised hardware such as GPUs, which are by now taken for granted by computational scientists and access to which is relatively straightforward through e.g. cloud computing platforms. Further, this cost is unlikely to scale dramatically with the dimensionality of the time-series: each new channel in the time-series increases the dimensionality of only the first affine transformation of the embedding network, which will typically amount to only a handful of new trainable parameters in the network each time the dimensionality of the time-series increases by one.

6.2. Additional practical guidance for the practitioner

We conclude by providing some final practical guidance to the practitioner on how to apply NPE and NRE. This guidance does not constitute a definitive or immutable set of rules – it is difficult to know *a priori* how and when these and other methods will perform best – but it is provided only to supply the practitioner with a useful starting point for their own experiments:

Neural posterior estimation vs. neural ratio estimation The question of which approach to consider in real-world use cases – NPE and NRE – arises naturally. While in general it can be useful to try both approaches, we suggest that, in the event that the practitioner must choose between the two, the practitioner may more readily achieve high quality results with NRE; our reasoning for this suggestion is that the expressivity of the neural networks comprising NRE can be more easily enhanced than can the networks comprising NPE, due to the latter's requirement of a bijective/invertible neural network (see Section 3.2.1).

Number of simulations to generate It is extremely difficult to provide useful general guidance on the question of how many simulations the experimenter should generate from the ABM when applying NPE and/or NRE: since the ABM's simulations constitute a training dataset for the function approximator comprising NPE/NRE, this question is intimately linked to the question of how much

data is needed to train these function approximators well enough that they generalise successfully to data not seen during training, such as the real-world dataset for which a posterior distribution is desired. This is an extremely challenging theoretical question in computational learning theory, and one of the biggest open problems in machine learning is understanding the generalisation properties of neural networks. However, in the interest of offering some (albeit imperfect) guidance, we suggest to aim to generate 10^3 $(\mathbf{x}, \boldsymbol{\theta})$ pairs from the joint distribution $p(\mathbf{x} \mid \boldsymbol{\theta})\, p(\boldsymbol{\theta})$ in the first instance, since in many cases this offers a good tradeoff between (a) having sufficient data to train neural networks reasonably well, and (b) the computational constraints encountered while running expensive ABMs. There is also evidence to suggest that using a relatively small dataset to train NPE and NRE can improve their robustness in misspecified settings (Cannon et al., 2022). However, we stress again that the number of training points required to achieve good performance will generally depend strongly on the given problem.

Data availability statement

All data and code are available on GitHub at https://github.com/joelnmdyer/sbi4abm.

Acknowledgements

We thank Robert Axtell, Nicholas Bishop, François Lafond, Marco Pangallo, R. Maria del Rio-Chanona, and the anonymous reviewers for their helpful comments. The majority of this work was completed while JD was supported by the EPSRC Centre For Doctoral Training in Industrially Focused Mathematical Modelling (EP/L015803/1) in collaboration with Improbable, and by The Alan Turing Institute via The PhD Enrichment Scheme under the EPSRC grant EP/N510129/1. JD acknowledges further support from grant number EP/W002949/1 (the UKRI AI World Leading Researcher Fellowship awarded to Wooldridge). JDF acknowledges funding from Baillie Gifford.

Appendix A. Further details on the literature on Bayesian parameter inference for agent-based models

A.1. Approximate Bayesian computation

Bayesian estimation of economic ABMs has gained popularity with the work of Grazzini et al. (2017), in which the authors discuss three approaches to approximate Bayesian parameter inference. In this section, we briefly outline the three approaches, and show that each method can be seen as explicitly or implicitly approximating the unknown likelihood function associated with the ABM by

$$\hat{p}(\mathbf{y} \mid \boldsymbol{\theta}) \propto \int K_\epsilon(\mathbf{y}, \mathbf{x})\, p(\mathbf{x} \mid \boldsymbol{\theta})\, \mathrm{d}\mathbf{x}, \tag{37}$$

where K_ϵ is a (possibly unnormalised) kernel function, with parameter(s) ϵ, providing a measure of "distance" between observed data $\mathbf{y}$ and simulated data $\mathbf{x}$. This permits an approximation $\hat{p}(\boldsymbol{\theta} \mid \mathbf{y})$ of the posterior density as

$$\hat{p}(\boldsymbol{\theta} \mid \mathbf{y}) \propto \int K_\epsilon(\mathbf{y}, \mathbf{x})\, p(\mathbf{x} \mid \boldsymbol{\theta})\, p(\boldsymbol{\theta})\, \mathrm{d}\mathbf{x}. \tag{38}$$

Such approaches – targeting (38) – are comprehensively subsumed under "ABC" which, as noted by Grazzini et al. (2017), is a technique that has received significant attention within the computational statistics community over the last two decades as a means to constructing approximate posterior densities for models with intractable likelihood functions (see e.g. Tavaré et al., 1997; Pritchard et al., 1999; Beaumont et al., 2002; Sunnåker et al., 2013; Schmon et al., 2020). The approach has been successfully applied within a variety of fields, most prominently in ecology, epidemiology, and systems biology (Toni et al., 2009; Liepe et al., 2014; Beaumont, 2010; Ju et al., 2021). While the approaches discussed by Grazzini et al. (2017) can be cast in this general form, a judicious choice of a *particular* kernel K_ϵ based on reasonable assumptions is crucial. In particular, different kernel choices are called for in different scenarios. Some possibilities are outlined below.

A.1.1. Parametric density estimation

The simplest approach discussed by Grazzini et al. (2017) involves assuming that the ABM has entered a statistical equilibrium, such that the observations $y_t \in \mathbb{R}^d$ in the time-series $\mathbf{y} := (y_1, y_2, \ldots, y_T) \in \mathbb{R}^{d \times T}$ are fluctuations around some stationary value m^*:

$$y_t = m^* + \varepsilon_t, \tag{39}$$

where (ε_t) are *iid* noise terms with density g_ϵ and parameters ϵ. Although more complex distributions are available, g_ϵ may for example be a zero-mean Gaussian distribution, and ϵ the elements of the covariance matrix.

Under these assumptions, recalling that the elements of the time-series are assumed to be independent, the parameters m^* and ϵ can be estimated by the means and covariances of the elements of the time-series to give $\hat{m}^*$ and $\hat{\epsilon}$. More precisely, first fixing $\boldsymbol{\theta}$ and drawing a sample $\mathbf{x} \sim p(\mathbf{x} \mid \boldsymbol{\theta})$, one can calculate the estimates $\hat{m}^*(\mathbf{x})$ and $\hat{\epsilon}(\mathbf{x})$ using, for example, maximum likelihood estimation. Adopting as the measure of distance the probability of the true data being observed under this approximated process yields the choice of kernel

$$K_\epsilon(\mathbf{y}, \mathbf{x}) = \prod_{t=1}^{T} g_{\hat{\epsilon}(\mathbf{x})}\left(y_t - \hat{m}^*(\mathbf{x})\right). \tag{40}$$

Finally by taking $R \geq 1$ *iid* simulated datasets[12] $\mathbf{x}^{(r)} \sim p(\mathbf{x} \mid \boldsymbol{\theta})$, $r = 1, \ldots, R$, and calculating their associated estimators $\hat{m}^*\left(\mathbf{x}^{(r)}\right)$ and $\hat{\epsilon}\left(\mathbf{x}^{(r)}\right)$, the likelihood at $\boldsymbol{\theta}$, i.e. Equation (37), is approximated with the Monte Carlo average

$$\hat{p}_{\boldsymbol{\theta}}(\mathbf{y}) \approx \frac{1}{R} \sum_{r=1}^{R} \prod_{t=1}^{T} g_{\epsilon\left(\mathbf{x}^{(r)}\right)}\left(y_t - m^*\left(\mathbf{x}^{(r)}\right)\right), \tag{41}$$

where we write $\hat{p}_{\boldsymbol{\theta}}$ instead of $\hat{p}(\cdot \mid \boldsymbol{\theta})$. Alternatively, the R simulations may be pooled to generate single estimates $\hat{m}^*\left(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(R)}\right)$ and $\hat{\epsilon}\left(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(R)}\right)$ at $\boldsymbol{\theta}$. This is closely related to *synthetic likelihood* approaches (Wood, 2010; Price et al., 2018) which use similar Gaussian distributions, but usually rely on summary statistics.

In either case, the resulting approximate likelihood $\hat{p}_{\boldsymbol{\theta}}$ from Equation (41) can then be used downstream for parameter estimation, either directly via e.g. maximum likelihood or through the corresponding approximate posterior, simulated e.g. through Markov chain Monte Carlo (MCMC). This approach suffers from clear limitations, however. Firstly, it treats the data points in the observed time-series as independent – that is, as lacking a natural ordering – which destroys important information when the observed $\mathbf{y}$ and simulated $\mathbf{x}$ are time-series. Secondly, choosing an appropriate and sufficiently flexible family of parametric densities g_ϵ to construct the likelihood approximation is non-trivial, with poor choices leading to erroneous Bayesian inference.

A.1.2. Non-parametric density estimation

An alternative approach, which partially addresses the second limitation described in Section A.1.1, is to forgo the assumption of a parametric family of densities and instead use a non-parametric method for density estimation. Grazzini et al. (2017) describe the use of KDE-ABC for this purpose. Here, the data points in the time-series are once again assumed to be independent and fluctuating about some stationary value m^* as in the parametric approach described above. Then, an estimate of the likelihood function is obtained by applying KDE-ABC to $R \geq 1$ *iid* simulations of length S, $\mathbf{x}^{(r)} := (x_1^{(r)}, \ldots, x_S^{(r)}) \sim p(\mathbf{x} \mid \boldsymbol{\theta})$, $r = 1, \ldots, R$, providing an unbiased estimate of the approximated likelihood function in Equation (37) as

$$\hat{p}_{\boldsymbol{\theta}}(\mathbf{y}) \approx \frac{1}{R} \sum_{r=1}^{R} K_\epsilon\left(\mathbf{y}, \mathbf{x}^{(r)}\right) := \frac{1}{R} \sum_{r=1}^{R} \prod_{t=1}^{T} \hat{p}_\epsilon\left(y_t \mid \boldsymbol{\theta}, \mathbf{x}^{(r)}\right), \tag{42}$$

where $\hat{p}_\epsilon\left(y_t \mid \boldsymbol{\theta}, \mathbf{x}^{(r)}\right)$ is the estimate of the conditional density $p\left(y_t \mid \boldsymbol{\theta}, \mathbf{x}^{(r)}\right)$ obtained via KDE-ABC:

$$\hat{p}_\epsilon(y_t \mid \boldsymbol{\theta}, \mathbf{x}) = \frac{1}{S} \sum_{s=1}^{S} \kappa_\epsilon\left(y_t - x_s\right), \tag{43}$$

for some probability kernel κ_ϵ with bandwidth parameter(s) ϵ. In particular, the authors employ a Gaussian kernel with bandwidth chosen using Silverman's method (Silverman, 1986), such that

$$\kappa_\epsilon\left(y_t - x_s\right) = \frac{1}{\epsilon} \kappa\left(\frac{\|y_t - x_s\|_2}{\epsilon}\right). \tag{44}$$

Alternatively, as in Section A.1.1, the R simulations may be pooled and a single KDE-ABC model fit to the combined dataset to obtain $\hat{p}(\mathbf{y} \mid \boldsymbol{\theta})$. While these non-parametric approaches to density estimation are arguably more flexible than the assumption of a parametric family, they are well known to suffer from the *curse of dimensionality*, limiting their applicability to low dimensional data only, even under the unrealistic assumption that the y_t are assumed independent.

A.1.3. Classical approximate Bayesian computation

The authors discuss a third common approach to simulation-based Bayesian inference for intractable simulation models, namely classical ABC. A typical ABC algorithm proceeds by proposing a parameter value $\boldsymbol{\theta}$ from some proposal distribution, for example the prior density, and determining whether to accept or reject $\boldsymbol{\theta}$ on the basis of some meaningful notion of distance $d(\mathbf{x}, \mathbf{y})$ between a simulation $\mathbf{x} \sim p(\mathbf{x} \mid \boldsymbol{\theta})$ and the observed data $\mathbf{y}$. A common choice for the distance d is the Euclidean distance between suitable summary statistics of the data; that is, $d(\mathbf{x}, \mathbf{y}) := \|\mathbf{s}(\mathbf{x}) - \mathbf{s}(\mathbf{y})\|$ where $\mathbf{s}$ is a vector of, for example, the mean, standard deviation, and lag-1 autocorrelation of the data. While many versions of ABC exist, the simplest – Rejection ABC – involves repeatedly proposing candidate parameter values $\boldsymbol{\theta} \sim p(\boldsymbol{\theta})$ and rejecting in the event that $d(\mathbf{x}, \mathbf{y}) > \epsilon$, where $\mathbf{x} \sim p(\mathbf{x} \mid \boldsymbol{\theta})$. Rejection ABC can therefore be cast in the form of Equation (38) by taking

$$K_\epsilon(\mathbf{y}, \mathbf{x}) = \mathbb{1}\left[d(\mathbf{x}, \mathbf{y}) \leq \epsilon\right], \tag{45}$$

providing an estimate of the (unnormalised) approximate likelihood function in Equation (37) as

$$\hat{p}_{\boldsymbol{\theta}}(\mathbf{y}) \propto \frac{1}{R} \sum_{r=1}^{R} \mathbb{1}\left[d\left(\mathbf{x}^{(r)}, \mathbf{y}\right) \leq \epsilon\right]. \tag{46}$$

[12] While R can be chosen to be any natural number, it is usually efficient to take $R = 1$ when averages of estimators are concerned and run the Markov chain for an accordingly higher number of iterations (Bornn et al., 2017; Sherlock et al., 2017).

Algorithm 2: Metropolis-Hastings sampling scheme for ABC.

Input: Prior distribution $p(\cdot)$, observation $\mathbf{y}$, proposal distribution $q(\cdot \mid \boldsymbol{\theta})$, initial value $\boldsymbol{\theta}_0$, number of iterations n;
Result: Empirical posterior $\sum_{i=1}^{n} \delta_{\boldsymbol{\theta}_i}$
for $r = 1, \dots, R$ **do**
  Simulate $\mathbf{x}^{(r)} \sim p\left(\cdot \mid \boldsymbol{\theta}_0\right)$;
end
for $i = 1, \dots, n$ **do**
  Propose a parameter by sampling $\boldsymbol{\theta} \sim q\left(\cdot \mid \boldsymbol{\theta}_{i-1}\right)$;
  for $r = 1, \dots, R$ **do**
    Simulate $\tilde{\mathbf{x}}^{(r)} \sim p(\cdot \mid \boldsymbol{\theta})$;
  end
  Evaluate $\hat{p}_{\boldsymbol{\theta}}(\mathbf{y})$ using $\{\tilde{\mathbf{x}}^{(r)}\}_{r=1}^{R}$ according to either Equation (41), (42), or (46) as desired;
  Accepted the proposed parameter, by setting $\hat{p}_{\boldsymbol{\theta}_i}(\mathbf{y}) = \hat{p}_{\boldsymbol{\theta}}(\mathbf{y})$, $\boldsymbol{\theta}_i = \boldsymbol{\theta}$ and $\{\mathbf{x}^{(r)}\}_{r=1}^{R} = \{\tilde{\mathbf{x}}^{(r)}\}_{r=1}^{R}$, with probability

$$\alpha = \min\left\{1, \frac{\hat{p}_{\boldsymbol{\theta}}(\mathbf{y}) p(\boldsymbol{\theta}) q(\boldsymbol{\theta}_{i-1} \mid \boldsymbol{\theta})}{\hat{p}_{\boldsymbol{\theta}_{i-1}}(\mathbf{y}) p(\boldsymbol{\theta}_{i-1}) q(\boldsymbol{\theta} \mid \boldsymbol{\theta}_{i-1})}\right\},$$

  otherwise reject the proposed parameter by setting $\hat{p}_{\boldsymbol{\theta}_i}(\mathbf{y}) = \hat{p}_{\boldsymbol{\theta}_{i-1}}(\mathbf{y})$ and $\boldsymbol{\theta}_i = \boldsymbol{\theta}_{i-1}$.
end

As discussed by Grazzini et al. (2017), ABC enjoys a number of desirable properties, in particular (under mild regularity conditions) consistency with the true posterior distribution as $\epsilon \to 0$, provided no information loss is incurred by the choice of distance d. However, a major challenge to its use in practice is designing such a distance function: the success or failure of the approach often hinges on the engineering of summary statistics that are close to sufficient.[13] A number of works have investigated strategies for summary statistic selection in ABC (e.g. Fearnhead and Prangle, 2012; Blum et al., 2013; Wiqvist et al., 2019; Chen et al., 2020) and there has been some success in alternative, summary-free distances (see e.g. Park et al., 2016; Bernton et al., 2019; Dyer et al., 2021a). It is however in general not possible to derive a low-dimensional sufficient statistic for arbitrary simulation models, and a poor choice of summary statistics can lead to an unacceptable loss of information from the original data, resulting in a posterior approximation of little value.

A.1.4. Unifying the approaches

In Algorithm 2, we show how the approximate posterior (38), with any of the three possibilities for K_ϵ discussed by Grazzini et al. (2017), may be targeted using a variant of the popular Metropolis-Hastings (MH) algorithm (e.g. Hastings, 1970). If the likelihood estimate $\hat{p}_{\boldsymbol{\theta}}(\mathbf{y})$ is an unbiased, non-negative estimate of some desired target $p_{\boldsymbol{\theta}}(\mathbf{y})$, then Algorithm 2 is an example of a pseudo-marginal MH algorithm (Andrieu and Roberts, 2009). Note that while the sampling procedure in Algorithm 2 exactly targets the posterior distributions described in Grazzini et al. (2017), many alternative posterior sampling algorithms exist (see e.g. Beaumont et al., 2009).

We once again emphasise that a major disadvantage to applying the methods described by Grazzini et al. (2017) to ABMs is that estimating the posterior involves simulating $R \geq 1$ times from the ABM at each proposed parameter value $(\boldsymbol{\theta}_i)_{i=1,\dots,n}$. Since ABMs are typically expensive to simulate, and n is required to be many hundreds of thousands in order to accurately estimate the targeted posterior, such approaches can rapidly lead to a prohibitively large number of required simulations.

A.2. Latent variable models

A class of models closely related to approximate Bayesian computation is the class of so-called *latent variable models*. Here, the real data, $\mathbf{y}$, is assumed to be a noisy observation of an unobserved (latent) process $\mathbf{x}$. Thus, in contrast to previous approaches, some discrepancy between $\mathbf{y}$ and $\mathbf{x}$ is to be expected.

Under the most basic scenario, the data obtained from the simulator are identically $x_t \sim p(x \mid \boldsymbol{\theta})$ and $p(\mathbf{x} \mid \boldsymbol{\theta}) = \prod_{t=1}^{T} p(x_t \mid \boldsymbol{\theta})$. The observed data is then assumed to have an error distribution, g_ϵ, say, such that the contribution made by one observation y_t to the likelihood function is thus

$$p(y_t \mid \boldsymbol{\theta}) = \int g_\epsilon(y_t \mid x_t) p(x_t \mid \boldsymbol{\theta}) \mathrm{d}x_t \tag{47}$$

and $p(\mathbf{y} \mid \boldsymbol{\theta}) = \prod_{t=1}^{T} p(y_t \mid \boldsymbol{\theta})$. We draw the attention of the reader to the conceptual similarity to (37); however, (47) is no longer an approximation but is the exact likelihood under the assumption of measurement error. (Similarly, ABC can be seen as exact if K_ϵ describes the observational error, see e.g. Wilkinson 2013.) A simple unbiased estimator of (47) is

$$\hat{p}(\mathbf{y} \mid \boldsymbol{\theta}) = \prod_{t=1}^{T} \frac{1}{R} \sum_{r=1}^{R} g_\epsilon\left(y_t \mid x_t^{(r)}\right), \qquad x_t^{(r)} \sim p(x \mid \boldsymbol{\theta})$$

[13] A Bayesian sufficient statistic $\mathbf{s}(\mathbf{x})$ for a model $p(\mathbf{x} \mid \boldsymbol{\theta})$ is one for which $p(\boldsymbol{\theta} \mid \mathbf{s}(\mathbf{x})) = p(\boldsymbol{\theta} \mid \mathbf{x})$. Low-dimensional sufficient statistics for arbitrary probabilistic models are generally unobtainable (Pitman, 1936; Koopman, 1936; Darmois, 1935).

for some $R \geq 1$,[14] which can then also be used in Algorithm 2, for example, to target the associated approximate posterior. The independence assumption underlying the simulator data is, however, not realistic for ABMs and other time-series models, and incorporating the temporal aspect of data requires further structural assumptions, such as those of hidden Markov models.

A.2.1. Hidden Markov models and particle filters

Particle filters, an instance of a broader class of sequential Monte Carlo (SMC) methods (see Ju et al., 2021, for a recent overview of SMC and their application to epidemiological ABMs) relax the independence assumption imposed on the output of the simulation model and have previously been employed for Bayesian parameter inference for economic ABMs (Lux, 2018, 2021). However, such methods also involve making assumptions about the structure of the model – specifically, the assumption of a state space structure with a particular error distribution. Furthermore, previous works have noted the significant computational burden required for SMC methods (Malleson et al., 2020; Lux, 2021), which can arise due to the fact that a number of particles that grows exponentially with the model dimensions is often required to avoid failure modes such as particle collapse.

A.3. Neural methods

As an alternative to kernel density estimation, neural density estimators have previously been employed to perform Bayesian estimation of ABMs. Platt (2021) presents a method which diverges from the approaches of Grazzini et al. (2017) in two main regards. Firstly, in contrast to the previous independence assumption, it assumes an autoregressive structure for the time-series model to better capture temporal correlations in the data. In particular, the approach assumes a time-series model that is Markov of order L, that is, $p(x_t \mid x_{1:t-1}, \boldsymbol{\theta}) = p(x_t \mid x_{t-L:t-1}, \boldsymbol{\theta})$, where $x_{1:t} = (x_1, \dots, x_t)$. In words, the distribution of the state at time t depends only on the previous L states (and $\boldsymbol{\theta}$). In light of this assumption, the full likelihood can be factorised as

$$p_{\boldsymbol{\theta}}(\mathbf{y}) = p(y_{1:L} \mid \boldsymbol{\theta}) \prod_{t=L+1}^{T} p(y_t \mid y_{t-1}, \dots, y_{t-L}, \boldsymbol{\theta}). \tag{48}$$

Secondly, as a consequence of the autoregressive model structure, the method employs a *conditional* density estimator $q_{\boldsymbol{\phi}}$ with trainable parameters $\boldsymbol{\phi}$ to approximate the transition density function $p(x_t \mid x_{t-L:t-1}, \boldsymbol{\theta})$. The resulting likelihood approximation can be written

$$\hat{p}_{\boldsymbol{\theta}}(\mathbf{y}) = \prod_{t=L+1}^{T} q_{\boldsymbol{\phi}(\boldsymbol{\theta})}(y_t \mid y_{t-1}, \dots, y_{t-L}), \tag{49}$$

where the contribution of the first L terms is ignored. A mixture density network (Bishop, 1994) assumes the role of the conditional density estimator $q_{\boldsymbol{\phi}(\boldsymbol{\theta})}$ with parameters $\boldsymbol{\phi}(\boldsymbol{\theta})$ trained for each $\boldsymbol{\theta}$. Denoting the L states preceding y_t as $\mathbf{y}_{<t}$ for brevity, the particular form of the mixture density network is

$$q_{\boldsymbol{\phi}}(y_t \mid \mathbf{y}_{<t}) = \sum_{k=1}^{K} \alpha_k(\mathbf{y}_{<t}) \, \mathcal{N}(y_t \mid \mu_k(\mathbf{y}_{<t}), \Sigma_k(\mathbf{y}_{<t})). \tag{50}$$

In the above, α_k are the mixing coefficients of the K Gaussian mixture components each with Gaussian density $\mathcal{N}(\cdot \mid \mu_k, \Sigma_k)$, and μ_k, Σ_k are respectively the means and covariance matrices of these Gaussian mixture components, all of which are parameterised by neural networks and trained via maximum likelihood on R *iid* model samples $\mathbf{x}^{(r)} \sim p(\mathbf{x} \mid \boldsymbol{\theta})$, $r = 1, \dots, R$. The resulting likelihood estimate can then once again be used also with Algorithm 2, for example, to target the associated approximate posterior distribution.

While the model structure described above accounts for the sequential nature of simulations generated by ABMs to some extent, the computational burden associated with the simulation of sufficient training data and the training of a new conditional density estimator at each $\boldsymbol{\theta}$ renders it largely infeasible (Shiono, 2021). This is because each of the N steps of the MCMC procedure the author employs to construct a posterior distribution over parameters entails (a) $R \geq 1$ simulations from the ABM, in addition to (b) training an entirely new neural network on the resulting $R(T - L)$ training examples generated at each parameter setting. Since N will need to be large – often many millions – in order to achieve a low Monte Carlo error on the resulting posterior, and $R \gg 1$ will generally be needed to avoid overfitting to any single trajectory generated by the ABM (Platt (2021) uses $R = 100$ in their studies), the simulation burden required by this method to achieve a reasonable performance can easily reach magnitudes of order 10^8 or larger. For computationally intensive simulation models, such as many ABMs, this can be prohibitively large, and methods that yield good quality posteriors at far lower simulation budgets are required.

In other work, Shiono (2021) examines a particular neural approach, with an invertible architecture and the ability to accommodate time-series data, for the specific purpose of estimating the posterior density for a New-Keyensian ABM. In contrast to both Platt (2021) and Shiono (2021), we take in this work a broad perspective on the subject of neural simulation-based inference for ABMs, by thoroughly contextualising the problem of Bayesian parameter estimation for ABMs in economics and the social sciences and by incorporating both benchmarking tasks and an exploration of suitable assessment criteria.

[14] Note that opposed to earlier cases the choice $R = 1$ is generally not optimal for products of unbiased estimators, see Doucet et al. (2015); Sherlock et al. (2015); Schmon et al. (2021).

Table 2
(Brock & Hommes) Discrepancies between the approximate ground-truth posterior and the posteriors estimated with KDE-ABC, GSL-ABC, MSM-ABC, SNPE, and SNRE. **Bold** and *italics* indicate best and second-best, respectively. For SNPE and SNRE, * indicates that the hand-crafted summary statistics described in Section 4.2 were used; otherwise, summary statistics are learned from the simulated data.

Metric	Estimation method						
	KDE-ABC	GSL-ABC	MSM-ABC	SNPE*	SNPE	SNRE*	SNRE
WASS	0.304	0.321	0.312	0.306	*0.154*	0.291	**0.132**
MMD	0.127	0.100	0.141	0.133	*0.036*	0.118	**0.025**
Budget	1.5×10^5	10^5	10^5	10^4	10^4	10^4	10^4

Appendix B. Further experimental results

In this section, we provide further experimental results to supplement those presented in the main body of the article.

B.1. The Brock & Hommes model: an additional parameter set

We now take $\beta = 10$ and consider the task of estimating the posterior density $p(\boldsymbol{\theta} \mid \mathbf{y})$, where $\mathbf{y} := (y_1, y_2, \dots, y_T) \sim p(\mathbf{x} \mid \boldsymbol{\theta}^*)$, $T = 100$, and $\boldsymbol{\theta}^* := (g_2^*, b_2^*, g_3^*, b_3^*) = (-0.7, -0.4, 0.5, 0.3)$. In this case, we use the following priors: $g_2, b_2 \sim \mathcal{U}(-1, 0)$ and $g_3, b_3 \sim \mathcal{U}(0, 1)$.

In Fig. 10, we show the posteriors obtained using different posterior estimation methods: Fig. 10a shows the approximate ground-truth posterior density obtained with the true likelihood function and MH; Fig. 10b shows the posterior obtained with KDE-ABC and MH; Figs. 10c and 10d show the posterior estimated via *sequential* neural posterior estimation (SNPE) with naive and learned summary statistics, respectively; and Figs. 10e and 10f show the posterior obtained via *sequential* neural ratio estimation (SNRE) and naive and learned summary statistics, respectively, which were also sampled with MH. In this experiment, we use an embedding network consisting of two stacked gated recurrent units (GRUs) with hidden state of size 32, followed by a single linear layer of size 16. This results in an embedding network with approximately 10,000 trainable parameters. Finally, Fig. 11 shows the posteriors obtained with GSL-ABC and MSM-ABC as described in Section 4.2.

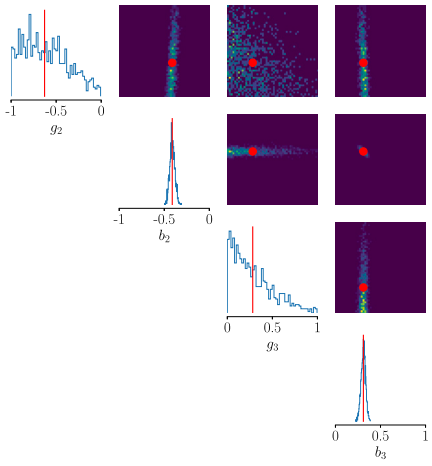
We see from the approximate ground-truth in Fig. 10a that the marginal posteriors for b_2 and b_3 remain sharply peaked on the generating parameters, but that the ground truth marginals for g_2 and g_3 are diffuse and sloped. The diffuseness of the posterior with respect to these two dimensions is also visually apparent in the joint density plots in the upper diagonal. This observation is consistent with previous studies, in which difficulty in accurately estimating the g parameters has been reported when the model exhibits random walk-like behaviour (see e.g. Lamperti, 2018a). Upon inspection of the estimated posteriors, we see that the overall shape is recovered well by SNPE and SNRE with learned summary statistics, but less accurately by KDE-ABC, GSL-ABC, MSM-ABC, and SNPE and SNRE when these use the naive hand-crafted summary statistics described in Section 4.2. This is once again reflected by the WASS and MMD scores, reported in Table 2, in which we see significantly decreased values for SNPE and SNRE with learned summary statistics with respect to the alternatives, despite a 10-fold decrease in the simulation budget they have been afforded. It is nonetheless noteworthy that even with the naive hand-crafted summary statistics, SNPE and SNRE achieve comparable and slightly favourable performance than KDE-ABC, GSL-ABC, and MSM-ABC again with a 10-fold decrease in the allotted simulation budget.

B.2. Robustness results for Appendix B.1

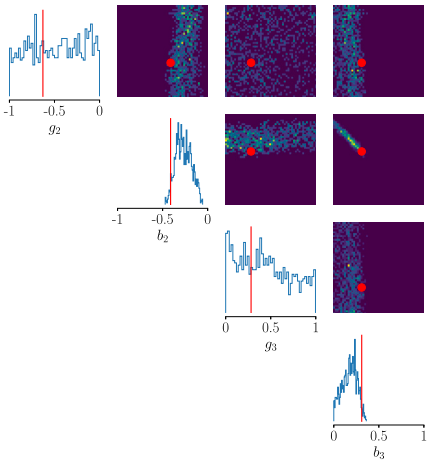
In this section, we detail additional experiments whose primary aim is to determine whether the performance of NPE and NRE is statistically different to that of KDE-ABC. As a secondary purpose, however, some of these experiments also serve to demonstrate the robustness of the performance of NPE and NRE to changes in the simulation budget they are afforded and/or to the capacity of the neural networks with which they obtain approximate posterior densities. Our secondary aim is therefore to provide further evidence to the reader of the broad scope of NPE and NRE in comparison to competing methods. Finally, the further experimental results we present on the Brock & Hommes model studied in Section 4.2 demonstrate the performance of NPE and NRE against additional Bayesian methods derived from popular methods in the ABM calibration literature, also discussed in Section 4.2 of the main text.

To perform NPE, we use a masked autoregressive flow (Papamakarios et al., 2017) with 5 transforms each with 50 hidden units. For NRE, we use a residual network with two layers of size 50. For both NPE and NRE, we use either the hand-crafted summary statistics described in Section 4.2, or a one-layer GRU, with hidden size 32 followed by a two-layer feedforward network with layer sizes 32 and 16, as the embedding network to learn summary statistics. This embedding net has approximately 4000 trainable parameters. For KDE-ABC, we use the same setup as in the main text for constructing a likelihood approximation: we simulate $R = 1$ times from the ABM at each $\boldsymbol{\theta}$ to estimate the likelihood function $p(\cdot \mid \boldsymbol{\theta})$, and use Silverman's method to select a value for the bandwidth (Silverman, 1986). Two further baselines, GSL-ABC and MSM-ABC, are also applied as discussed in Section 4.2.

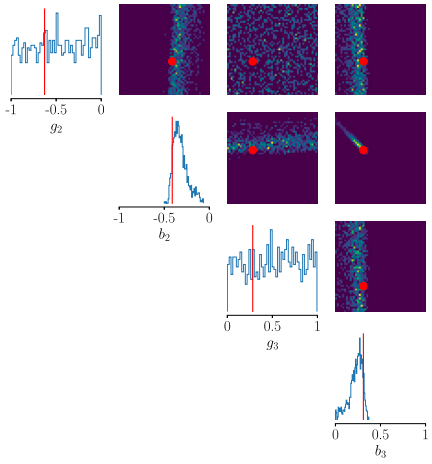
To sample from the posterior distribution for NRE and KDE-ABC, we continue to use MH, but in this case we apply a further test to the robustness of these methods by starting the MH chain at a randomly drawn parameter from the prior distribution. For both NRE and KDE-ABC, we perform a trial run of the sampling procedure with Gaussian proposals and a diagonal covariance matrix $\Sigma = I/10$, where I is the identity matrix. The factor $1/10$ is to reduce the number of proposals that fall outside of the boundaries of the prior



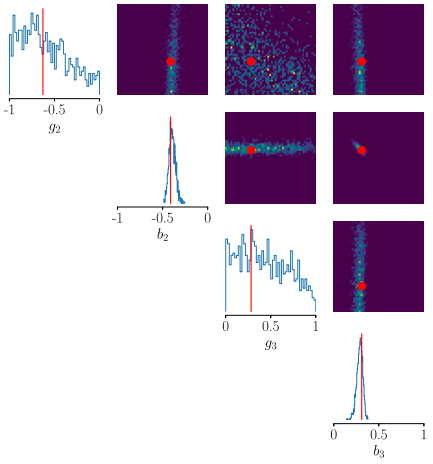
(a) True posterior



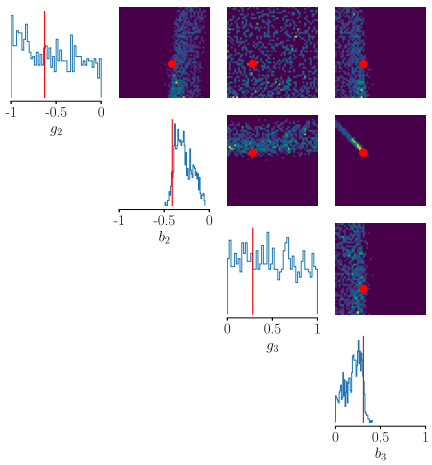
(b) KDE-ABC posterior, 10^5 simulations



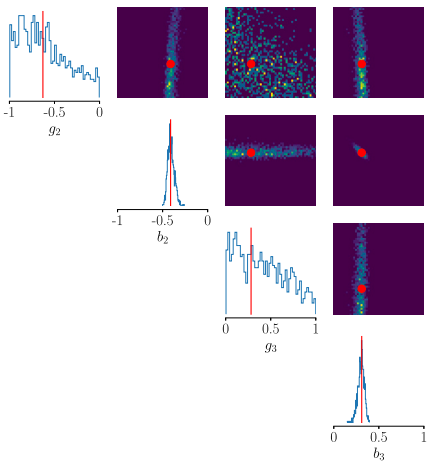
(c) Sequential neural posterior estimation with naive summary statistics, 10^4 simulations



(d) Sequential neural posterior estimation with learned summary statistics, 10^4 simulations

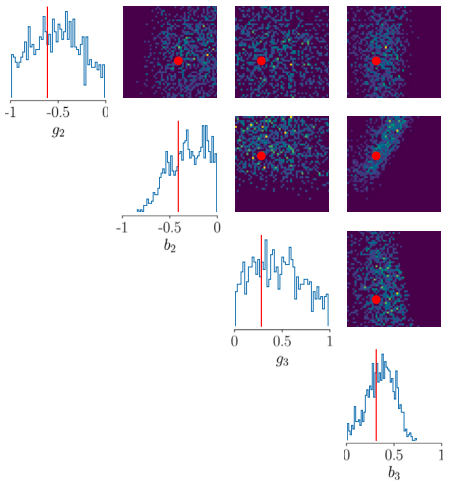


(e) Sequential neural ratio estimation with naive summary statistics, 10^4 simulations

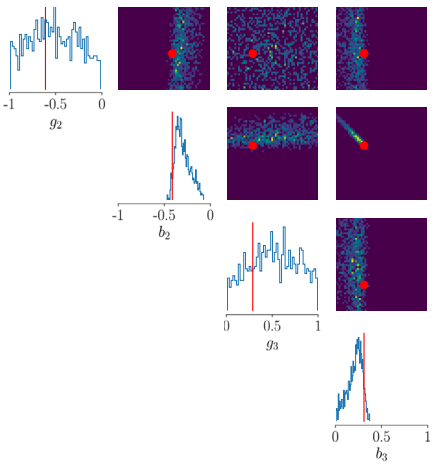


(f) Sequential neural ratio estimation with learned summary statistics, 10^4 simulations

Fig. 10. (**Brock & Hommes, parameter set 2**) Posteriors obtained with KDE-ABC, SNPE, and SNRE. The marginal posterior distributions are located on the diagonals, while the bivariate joint distributions for each parameter pair are located on the upper diagonal. Red lines/dots indicate the mean of the true posterior.



(a) GSL-ABC posterior, 10^5 simulations



(b) MSM-ABC posterior, 10^5 simulations

Fig. 11. (**Brock & Hommes, parameter set 2**) Posteriors obtained with GSL-ABC and MSM-ABC. The marginal posterior distributions are located on the diagonals, while the bivariate joint distributions for each parameter pair are located on the upper diagonal. Red lines/dots indicate the mean of the true posterior.

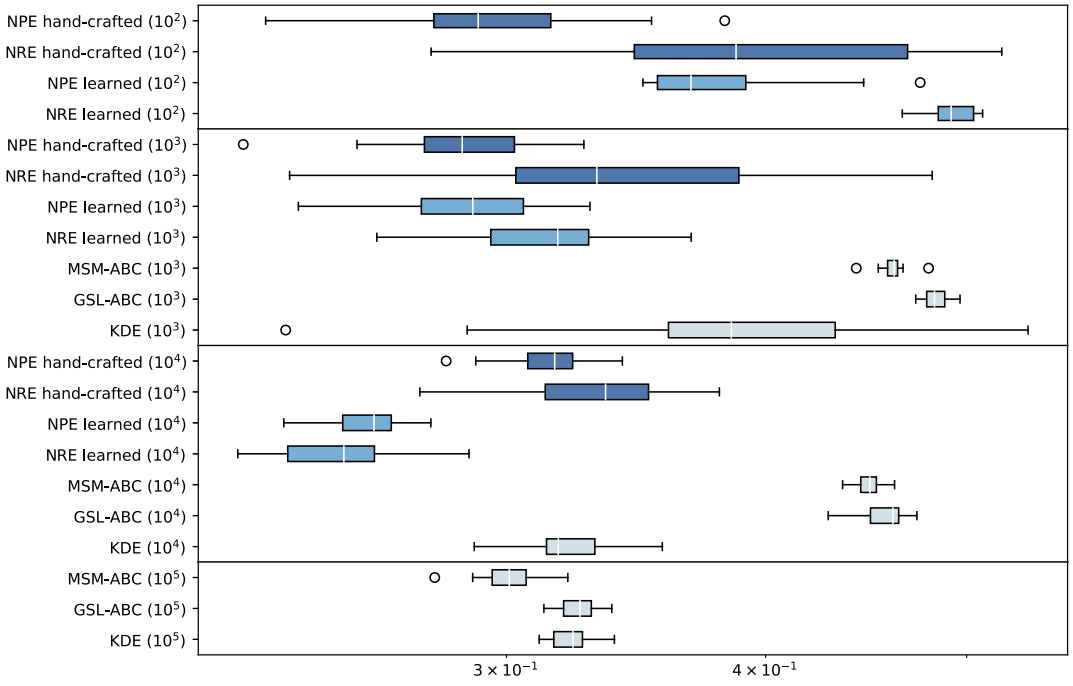
during the trial run. The empirical covariance matrix of the chain generated by the trial run is then used as the covariance matrix in the Gaussian proposal density for the proper MCMC chain.

We show in Fig. 12a (resp. Fig. 12b) boxplots for the WASS distance (resp. MMD) obtained by running the entire inference procedure[15] for 20 different seeds for the same fixed real world data set, $\mathbf{y}$, at different simulation budgets. The boxplots are grouped according to the simulation budget each data point comprising the boxplot was afforded: that is, boxplots within the same panel used the same number of simulations, which is indicated with the number displayed in parentheses following the method name. For example, the methods in the top panel of both subfigures are each afforded 10^2 simulations, while the methods shown in the bottom panel of each subfigure are afforded 10^5 simulations. Our methods are shown in dark and medium blue, while KDE-ABC, GSL-ABC, and MSM-ABC are shown in light blue. Suffixes "hand-crafted" and "learned" indicate, respectively, that NPE and NRE used the hand-crafted summary statistics described in Section 4.2 and used the lower-capacity GRU described earlier in this Section.

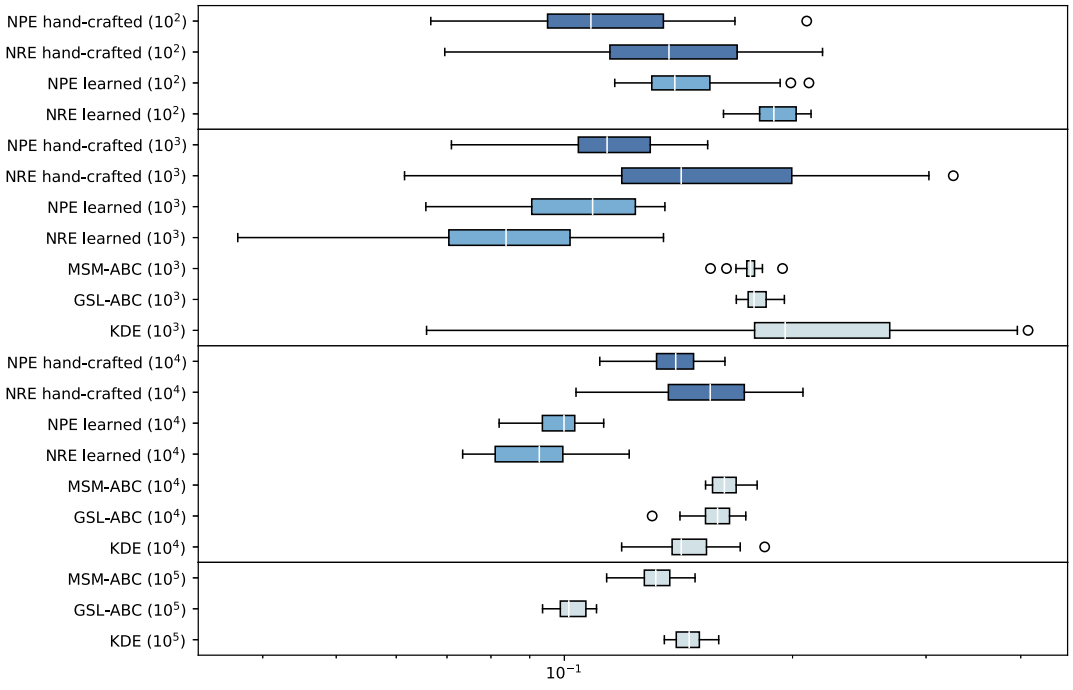
It is immediately apparent from these plots that KDE-ABC consistently underperforms in comparison to NPE and NRE. At a budget of 10^3 simulations, the performance of KDE-ABC varies wildly according to both the WASS distances and MMDs. The reason for this very varied behaviour is that KDE-ABC simply does not have a large enough simulation budget to be able to adequately explore the parameter space, preventing it from finding and generating samples from regions of high posterior density. Therefore, in the unfortunate yet likely event that the MCMC chain is initiated at a region of low posterior density, KDE-ABC will not be able to move away from that irrelevant region of the parameter space when it is only allowed to generate a comparatively small number of simulations from the ABM (recall that KDE-ABC simulates from the ABM at each step in the MCMC chain). Occasionally, KDE-ABC is lucky to have started its random MCMC walk in or near to a region of high posterior density, which is why it occasionally generates low WASS distances and MMDs; however, because the MCMC chain is so short when only 10^3 simulations are permitted, the modeller will not be able to determine whether the chain has converged on a region of high posterior density or whether it has yet to find one. In summary, KDE-ABC is an entirely unreliable Bayesian inference method when afforded 10^3 simulations.

In contrast, with even 10^2 simulations, NPE and NRE can exhibit far more favourable performance compared to KDE-ABC when the latter uses 10^3 simulations: with only 10^2 simulations, NPE with hand-crafted summary statistics, NRE with hand-crafted summary statistics, and NPE with learned summary statistics each produce approximate posteriors with WASS distances and MMDs that are no larger than, or are significantly smaller than, those generated by KDE-ABC when the latter is afforded 10^3 simulations. Even when KDE-ABC uses 10^4 simulations, it is consistently outperformed by NPE when the latter uses the hand-crafted summary statistics described in Section 4.2 and a simulation budget of 10^2 simulations. When NPE and NRE make use of 10^3 simulations, however, they produce a clear improvement in performance over KDE-ABC according to at least one – if not both – performance metrics, whether KDE-ABC uses 10^3, 10^4, or 10^5 simulations. (This is with the exception of NRE with hand-crafted statistics at 10^3 simulations, which only generally outperforms KDE-ABC when the latter also uses 10^3 simulations.) For example, when KDE-ABC is afforded 10^5 simulations, it only begins to compete with NPE's and NRE's performance when they use 10^3 simulations and learn summary statistics; additionally, KDE-ABC's distribution of WASS distances and MMDs at 10^5 simulations are *completely* non-overlapping with those of NPE and NRE when the latter uses learned summary statistics and a simulation budget of 10^4 simulations. In summary, NPE and NRE

[15] For KDE-ABC, this includes both the trial run and proper run of the MCMC chain. For NPE and NRE this includes the simulation of the training set from the joint density $p(\mathbf{x} \mid \theta)p(\theta)$, training of the networks, and construction of the posterior with *iid* sampling in the case of the normalising flow for NPE and by running both the trial run and the proper run of the MCMC chain for NRE.



(a) Wasserstein distances



(b) Maximum mean discrepancies

Fig. 12. (**Brock & Hommes, Appendix B.1**) Boxplots for the WASS distances (top) and MMDs (bottom) for each method, obtained by running the entire inference procedure 20 times for the same fixed **y** used in Appendix B.1.

generally result in improved performance over KDE-ABC even when the former are afforded a simulation budget that is orders of magnitude smaller.

Much of the discussion that we have provided so far on the performance of KDE-ABC in comparison to NPE and NRE in this experiment also applies to GSL-ABC and MSM-ABC. For example, even when afforded 10^4 simulations, GSL-ABC and MSM-ABC consistently underperform against NPE and NRE when the latter use 10^3 simulations. Interestingly, we see that GSL-ABC and MSM-ABC at 10^5 simulations outperform KDE-ABC at 10^5 simulations, according to the MMD metric, with MSM-ABC outperforming KDE-ABC in the WASS metric also; even then, it can be seen that GSL-ABC and MSM-ABC at 10^5 simulations is outperformed by NPE and NRE when the latter learns summary statistics and uses 10^4 simulations, when taking into account the distribution of both the WASS distances and MMDs.

As we discuss in the main text, there are two main reasons for this improvement in performance by NPE and NRE over KDE-ABC, GSL-ABC, and MSM-ABC:

1. The first is that NPE and NRE perform an upfront, offline training of their neural networks, which is performed separately from the act of drawing inferences – sampling from the posterior density. Simulating from the ABM therefore only takes place to construct a training set for NPE and NRE. This contrasts with KDE-ABC, GSL-ABC, MSM-ABC – along with other methods we have not shown experiments for, such as the methods described in Platt (2021) and Lux (2021) – which simulate from the ABM *within* the posterior sampling procedure. Since NPE and NRE do not simulate from the ABM within the posterior sampling procedure, and instead only simulate from the ABM to construct a training set for the neural networks comprising these methods, this enables NPE and NRE to sample adequately from the posterior, ensuring that there is a minimal Monte Carlo error on the resulting posterior. To be more explicit: the fact that NPE and NRE separate the task of simulating from the ABM from the task of sampling from the posterior enables NPE and NRE to cheaply generate hundreds of thousands of samples from the posteriors they approximate, which is required in order for the posterior density to be explored adequately. In contrast, if existing methods such as KDE-ABC, GSL-ABC, MSM-ABC, and the methods described in Platt (2021) and Lux (2021) wish to sample many hundreds of thousands of times from the posteriors they approximate – in order to minimise the Monte Carlo error on their posteriors – they must also simulate from the ABM many hundreds of thousands of times.
2. The second main reason is that NPE and NRE do not make restrictive assumptions about the data, in contrast to alternative methods. For example, KDE-ABC assumes stationarity and exchangeability of the observation comprising the observed data and simulated output, meaning it is limited in its performance by this assumption. This may explain the plateauing of KDE-ABC's performance as the simulation budget it is afforded increases from 10^3 to 10^5. In contrast, NPE and NRE with learned summary statistics continue to improve when the simulation budget they are afforded increases from 10^2 to 10^4.

Finally, we note that the change in performance of NPE and NRE with hand-crafted summary statistics does not appear to be monotonic in the simulation budget in this inference task. Indeed, the performance of both methods appears to degrade slightly as the simulation budget is increased when the hand-crafted summary statistics described in Section 4.2 are used to summarise the input data. We believe this can be explained by the fact that the summary statistics used are generic and are not necessarily close to sufficient, such that providing the neural networks with more and more training data (i.e. increasing the simulation budgets these methods are afforded) appears to bias these methods away from the true shape of the posterior. Therefore, while we have seen that NPE and NRE with hand-crafted summary statistics can still vastly improve upon alternative methods such as KDE-ABC, GSL-ABC, and MSM-ABC, and with a far smaller simulation budget, this observation highlights that care must be taken when introducing hand-crafted summary statistics to ensure that they properly characterise the data. In failing to do so, there is a danger of guiding NPE and NRE towards regions of the parameter space that are not entirely consistent with the observed data $\mathbf{y}$. (We note, however, that this point holds for other inference procedures, and not only for NPE and NRE.)

Appendix C. Further experimental details

C.1. Further experimental details for the Brock & Hommes model (Section 4.2)

For NPE, NRE with learned summary statistics, we use as an embedding network two stacked Elman recurrent units with hidden state of size 32, followed by a single linear layer of size 16. For KDE-ABC, we take $R = 1$ (see Appendix A.1.2) and sample with MH using 5×10^4 steps in a trial run with which the proposal density's covariance matrix is estimated before running a full chain with 10^5 steps.

C.2. Further experimental details of the multivariate geometric Brownian motion model

Here, NPE and NRE use learned summary statistics only, found using a one-layer GRU with hidden size 32 followed by a two-layer feedforward network with layer sizes 32 and 16, as the embedding network. We note that this GRU has approximately 4000 trainable parameters. We compare against KDE-ABC, which takes 10^5 steps in the MH chain and simulates $R = 10$ times at each step in the chain, amounting to a total simulation budget of 10^6 for KDE-ABC.

Appendix D. Validating approximate Bayesian inference

When performing approximate Bayesian inference in practice, it can be difficult to assess whether a posterior generated by any algorithm is in some sense accurate, informative, or meaningful. This raises the question of whether methods for checking the quality

of an approximate posterior derived from procedures such as NPE and NRE exist. We have already demonstrated one useful tool for validating approximate Bayesian inference in Section 5.2 of the main text, namely posterior predictive checks. In this section, we outline another common approach to validating approximate Bayesian inference procedures, and discuss how this posterior quality check is more readily available to the practitioner through the use of NPE and NRE than they are when using more classical posterior inference procedures such as KDE-ABC.

D.1. Simulation-based calibration

As previously mentioned, a major benefit of the Bayesian inferential paradigm as seen by a practitioners of Bayesian statistics is the fact that uncertainty quantification is a built-in feature captured via the posterior distribution. Its utility relies, however, on the ability to capture the correct degree of diffuseness in the posterior, such that the uncertainty quantification in the recovered posterior is meaningful. This presents a challenge in approximate Bayesian inference, since by definition the experimenter does not know the correct shape of the posterior when exact inference is intractable. Here, we address this issue by outlining a widely used approach to verifying the accuracy of approximate Bayesian inference pipelines in the absence of any ground-truth posterior densities.

Simulation-based calibration is a general purpose method for validating approximate Bayesian inference pipelines. The core idea is to use the fact that the *data-averaged posterior* should be identical to the prior distribution $p(\boldsymbol{\theta})$. That is, a perfect Bayesian inference pipeline should satisfy the following equality:

$$\int_{\mathcal{Y}\times\boldsymbol{\Theta}} p(\boldsymbol{\theta}\mid\mathbf{y})p(\mathbf{y}\mid\tilde{\boldsymbol{\theta}})p(\tilde{\boldsymbol{\theta}})\,\mathrm{d}\tilde{\boldsymbol{\theta}}\,\mathrm{d}\mathbf{y} = p(\boldsymbol{\theta}). \tag{51}$$

A deviation from this equality signifies some error in the posterior sampling procedure; thus, it has been proposed by Talts et al. (2020) that testing how close our Bayesian workflow is to satisfying this equality is a test of the accuracy of the Bayesian pipeline. This may be performed by repeating the following steps P times:

1. Generate $\tilde{\boldsymbol{\theta}} \sim p(\boldsymbol{\theta})$ from the prior distribution
2. Generate $\tilde{\mathbf{y}} \sim p(\mathbf{y}\mid\tilde{\boldsymbol{\theta}})$ from the likelihood function (i.e. the simulator)
3. Generate L uncorrelated posterior samples $\{\boldsymbol{\theta}_i\}_{i=1}^{L} \sim p(\boldsymbol{\theta}\mid\tilde{\mathbf{y}})$
4. Compute and store the rank statistic $r(\{\boldsymbol{\theta}_i\}_{i=1}^{L},\tilde{\boldsymbol{\theta}}) \in \{0,\dots,L\}$ for $\tilde{\boldsymbol{\theta}}$ within $\{\boldsymbol{\theta}_i\}_{i=1}^{L} \cup \{\tilde{\boldsymbol{\theta}}\}$

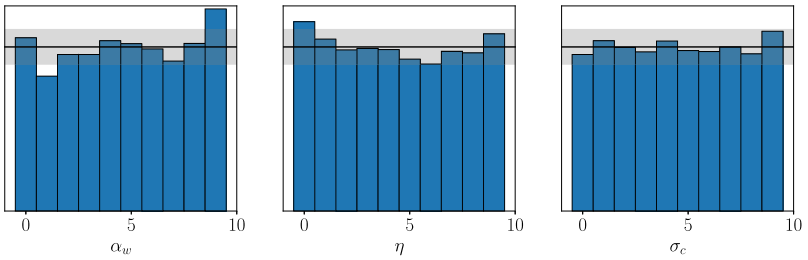
It can be shown (Theorem 1, Talts et al., 2020) that the rank statistics obtained as above will follow a discrete Uniform distribution on $\{0,\dots,L\}$ for any joint distribution $p(\mathbf{y}\mid\boldsymbol{\theta})p(\boldsymbol{\theta})$ for one-dimensional $\boldsymbol{\theta}$. For d-dimensional $\boldsymbol{\theta}$, each of the d components' rank histograms will be uniform. Thus, a deviation from the expected uniform distribution indicates that the four-step procedure described above is not sampling from the prior distribution, meaning that at least one term in Equation (51) is incorrect. In the case of approximate Bayesian inference, the only source of error is in the approximation of the posterior $p(\boldsymbol{\theta}\mid\mathbf{y})$; therefore, inspecting the discrepancy between the true distribution of rank statistics and the desired Uniform distribution thus gives an indication of the accuracy of the posterior approximation used in the Bayesian pipeline. Furthermore and conversely, particular patterns of deviation from the desired uniformity are interpretable and provide insight into the specific way in which the obtained posteriors are inaccurate (Talts et al., 2020). Performing this procedure, and inspecting the resultant rank histograms, is referred to as performing SBC.

D.1.1. Computational expense of simulation-based calibration

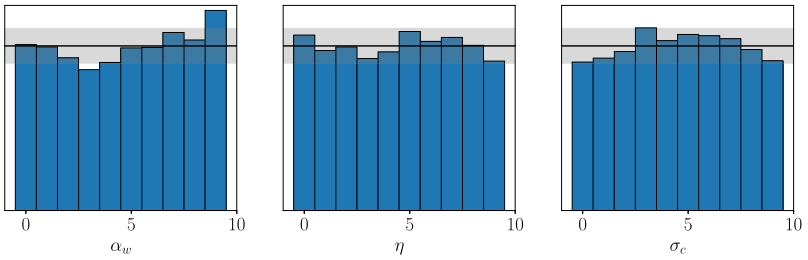
While this procedure is in principle possible for KDE-ABC, the computational burden is immense even for cheap simulation models, and becomes completely infeasible for large-scale ABMs. This is because the number of simulations required to generate n samples from the posterior would, in general, be at least[16] nR, such that the total number of simulations required to perform SBC for KDE-ABC increases to PnR. Even for a moderately sized SBC task with $P \simeq 10^3$ and the most optimistic $R = 1$, the total number of simulations required can easily reach the order of 10^8, since it is typically necessary to take n to be many hundreds of thousands to obtain a reasonably accurate estimate of the posterior. In fact, larger values of n are likely to be necessary, since ABMs are usually highly parameterised and thus can have large parameter spaces.

In contrast, such an approach to verifying the accuracy of posteriors derived from NPE and NRE is feasible due to the fact that such approaches involve the prior training of a global density (ratio) estimator, obviating any simulations during the inference (posterior sampling) phase. Thus the strength of NPE and NRE derives from not only their enhanced performance and improved simulation-efficiency, but also from the fact that such methods permit accuracy checks such as SBC that are otherwise infeasible for alternative parameter estimation techniques when the simulator is expensive, as is often the case for large-scale ABMs.

[16] In practice, this number may be larger due to the necessity of performing trial runs to estimate the parameters of the proposal distribution. For example, a common proposal distribution in an MCMC scheme is the Gaussian proposal, $q(\boldsymbol{\theta}' \mid \boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}' \mid \boldsymbol{\theta}, \Sigma)$, where Σ is some covariance matrix whose form strongly influences the efficiency of the MCMC scheme. Obtaining a useful form for Σ typically entails performing a short trial run of the MCMC scheme with the covariance matrix initially set proportional to the identity matrix, and then a full MCMC procedure is re-run using, for example, the empirical covariance matrix of samples generated by the trial run as Σ.



(a) Simulation-based calibration histogram for neural posterior estimation.



(b) Simulation-based calibration histogram for neural density ratio estimation.

Fig. 13. (**Franke & Westerhoff**) Rank histograms generated according to the simulation-based calibration procedure in Section D.1 using the trained posterior density estimator (top) and density ratio estimator (bottom). The shaded regions indicate the symmetric 95 percent point function of a Binomial random variable with success probability $1/P = 1/10$; that is, the bottom (resp. top) of the shaded region is the inverse of the cumulative distribution function for a Binomial random variable at quantile 0.025 (resp. 0.975). Loosely, this captures the expected deviations from complete uniformity when the histograms are genuinely uniformly distributed.

D.1.2. Example: the Franke & Westerhoff model

To further demonstrate the ability of NPE and NRE to recover accurate posterior estimates and informative summary statistics in a highly automated manner, we perform SBC using a posterior and density ratio estimator trained on the Franke & Westerhoff model (Franke and Westerhoff, 2012), which has previously been used in benchmarking experiments for ABM estimation methods (Platt, 2021). In particular, we consider the model version referred to as the "Wealth & Predisposition" model which, similarly to the Brock & Hommes model (see Section 4.2), may be written as a system of coupled equations which models the asset price dynamics that result from a heterogeneous system of traders:

$$p_t = p_{t-1} + \mu\left(n^f_{t-1} d^f_{t-1} + n^c_{t-1} d^c_{t-1}\right), \tag{52}$$

$$d^f_t = \phi\left(p^* - p_t\right) + \sigma_f \epsilon^f_t, \tag{53}$$

$$d^c_t = \chi\left(p_t - p_{t-1}\right) + \sigma_c \epsilon^c_t, \tag{54}$$

$$n^f_t = 1 - n^c_t = \frac{1}{1 + \exp\left(-\beta a_{t-1}\right)}, \tag{55}$$

$$a_t = \alpha_w\left(w^f_t - w^c_t\right) + \alpha_0, \tag{56}$$

$$w^j_t = \eta w^j_{t-1} + (1-\eta)\, g^j_t, \quad j \in \{f, c\} \tag{57}$$

$$g^j_t = \left[e^{p_t} - e^{p_{t-1}}\right] d^j_{t-2}, \quad j \in \{f, c\} \tag{58}$$

where the $\epsilon^j_t \sim \mathcal{N}(0,1)$ are standard normal random variables and $j = \{f, c\}$ labels fundamentalist and chartist traders, respectively. We take as output from the model the time series of log returns $r_t = p_t - p_{t-1}$, and assume the task of training a global, amortised density and density ratio estimator via NPE and NRE, respectively, using 10^4 simulations of length $T = 100$ from the simulator defined by Equations (52)–(58) with the following parameter settings: $\mu = 0.01$, $\beta = 1$, $\phi = 1$, $\chi = 0.9$, $\alpha_0 = 2.1$, $\sigma_f = 0.752$. The parameter vector for which we seek the posterior and density ratio estimators is taken to be $\boldsymbol{\theta} = \left(\alpha_w, \eta, \sigma_c\right)$, and we obtain *iid* samples from the posterior associated with the density ratio estimator via a sampling-importance-resampling scheme described in Appendix E.2. (Samples may be drawn *iid* from the NPE posterior by construction.)

We show in Fig. 13 the rank histograms obtained via the simulation-based calibration procedure in Section D.1 with NPE (Fig. 13a) and NRE (Fig. 13b). To construct these histograms, we take $P = 5,000$ and the following uniform priors: $\alpha_w \sim \mathcal{U}(0, 15000)$, $\eta \sim \mathcal{U}(0,1)$, and $\sigma_c \sim \mathcal{U}(0,5)$. The black line and gray band denotes the expected height and the expected variation in the heights, respectively, of the bars at this P and with this number of bins. We see that the bars tend to lie within the expected range, although some notable deviations exist: for example, the rank histograms for α_w for both NPE and NRE exhibit a minor bias towards larger

rank values, which suggests that the marginal posteriors for α_w in both cases is slightly biased towards lower values on average than the true posteriors (Talts et al., 2020). However, we emphasise that a major benefit of NPE and NRE is that they allow the modeller to make statements of this sort in the first place, whereas the number of simulations required to make such statements in the case of more traditional techniques such as KDE-ABC would quickly become prohibitively large.

D.1.3. Contrasting the computational expense of simulation-based calibration with that of posterior predictive checks

To simulate $J \geq 1$ realisations $\tilde{\mathbf{y}}^{(j)}, j = 1, \ldots, J$ from the posterior predictive distribution (see Equation (35)), J *iid* parameters must be sampled from the posterior density. The simulation burden for PPCs is therefore the simulation burden associated with construction of the posterior for $\mathbf{y}$ in the first place, in addition to the J further simulations required to perform the PPCs. Since this procedure needs only to be performed for the single dataset $\mathbf{y}$ observed from the real world, the difference in the simulation burden of PPCs when using the methods we endorse – NPE and NRE – and with more classical methods – such as KDE-ABC or other instances of ABC – is likely to be smaller than it is in the case of SBC. However, it remains the case that the simulation burden associated with performing PPCs under NPE and NRE is likely to be orders of magnitude smaller than for more classical approaches to approximate Bayesian parameter inference, due to the greater simulation efficiency of these methods for constructing the posterior for $\mathbf{y}$ in the first place, which we explain in Section 3.

Appendix E. Posterior sampling

E.1. Sampling with Metropolis-Hastings

The Metropolis-Hastings algorithm is a classical algorithm for generating samples from some density $p(\boldsymbol{\theta})$. Starting from $\boldsymbol{\theta}_0$ and given a proposal distribution $q(\cdot \mid \boldsymbol{\theta})$ which proposes successive values in the chain, it entails repeating the following steps: at each step $t \geq 1$,

1. propose $\boldsymbol{\theta} \sim q(\cdot \mid \boldsymbol{\theta}_t)$;
2. accept $\boldsymbol{\theta}$ (i.e. set $\boldsymbol{\theta}_{t+1} := \boldsymbol{\theta}$) with probability

$$\alpha = \min\left\{1, \frac{p(\boldsymbol{\theta})q(\boldsymbol{\theta}_t \mid \boldsymbol{\theta})}{p(\boldsymbol{\theta}_t)q(\boldsymbol{\theta} \mid \boldsymbol{\theta}_t)}\right\},$$

else set $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t$.

Since such a procedure generates correlated samples from the posterior, it is customary to *thin* the samples by retaining every nth sample for some integer $n \geq 1$ chosen as required.

To sample from the posteriors obtained via KDE-ABC and NRE in Section 4, we use MH with a normal proposal distribution $q(\cdot \mid \boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}; \ell^2\Sigma)$, and perform a trial run of 50,000 steps using an isotropic Gaussian to estimate the covariance matrix Σ of q before a further 100,000 steps are run. The use of such pilot runs is long standing-practice for random walk MH algorithms and can be motivated theoretically, for example, using Bayesian asymptotics (Schmon and Gagnon, 2022). In addition, we set $\ell = 2/\sqrt{d}$, where d is the parameter dimension following the guidelines of Gelman et al. (1996); Roberts et al. (1997); Schmon and Gagnon (2022). All chains are initialised at the parameter value which generated the observation. We thin the resultant chains by retaining every 100th value, resulting in 1,000 approximately uncorrelated samples from the respective posteriors.

E.2. Sampling with sampling-importance-resampling

Sampling-importance-resampling (SIR) is an approach to obtaining samples from a target distribution $f(\boldsymbol{\theta})$ given samples from a different distribution $g(\boldsymbol{\theta})$ which proceeds as follows:

1. generate samples $\{\boldsymbol{\theta}_i\}_{i=1}^N \overset{iid}{\sim} g(\boldsymbol{\theta})$;
2. compute weights $w_i = f(\boldsymbol{\theta}_i)/g(\boldsymbol{\theta}_i)$ for all i;
3. resample $\boldsymbol{\theta}_i$ with probability proportional to w_i.

In particular, this may be used in density ratio estimation to approximate the posterior by setting $g(\boldsymbol{\theta}) := p(\boldsymbol{\theta})$, $f(\boldsymbol{\theta}) := p(\boldsymbol{\theta} \mid \mathbf{y})$, and using that the density ratio estimator estimates the ratio

$$\frac{f(\boldsymbol{\theta})}{g(\boldsymbol{\theta})} = \frac{p(\boldsymbol{\theta} \mid \mathbf{y})}{p(\boldsymbol{\theta})} = \frac{p(\mathbf{y} \mid \boldsymbol{\theta})}{p(\mathbf{y})}. \tag{59}$$

Appendix F. Graph neural networks

Recent years have seen considerable progress in the development of graph neural networks (GNNs) in machine learning (e.g. Yao et al., 2019; Zhang et al., 2020; Baek et al., 2021). In many cases, the design of a GNN consists of generalising a convolution operator from regular, Euclidean domains – as appears in convolutional neural networks – to graphs. This has predominantly proceeded by

constructing a convolution in the spatial domain (see e.g. Masci et al., 2015; Niepert et al., 2016) or by exploiting the convolution theorem and performing a multiplication in the graph Fourier domain (see e.g. Bruna et al., 2014). A recent review of GNNs and their design can be found in Zhou et al. (2020).

The problem of extending GNNs to dynamic graphs has also recently received significant attention. In this vein, Li et al. (2017) introduce Diffusion Convolutional Recurrent Neural Networks, with applications to traffic flow prediction. In addition, Seo et al. (2018) propose Graph Convolutional Recurrent Networks, an adaptation of standard recurrent networks to operate on sequences of graphs via graph convolutional operators. Further examples of recurrent graph neural network architectures exist; a broader survey of neural networks for dynamic graphs can be found in Wu et al. (2021, Section 7).

Appendix G. Neural network architectures and training

For the graph embedding network in Section 5.2, the first module is a graph convolutional gated recurrent unit proposed in (Seo et al., 2018). Taking $L \in \mathbb{R}^{N\times N}$ as the normalised graph Laplacian for the order-N graph, this component operates on the sequence $(\mathbf{x}_t)_{t=1}^{T}$ of node states $\mathbf{x}_t \in \mathbb{R}^{N\times K}$ – where $K \geq 1$ is the dimensionality of each node state – to find a running embedding $\mathbf{h}^t \in \mathbb{R}^{N\times d_{\mathbf{h}}}$ of each subsequence $(\mathbf{x}_{t'})_{t'=1}^{t}$, $t = 1, \dots, T$, as follows:

$$
\begin{aligned}
\mathbf{s} &= \sigma\left(W_{\mathbf{sx}}(L)\cdot\mathbf{x}^t + W_{\mathbf{sh}}(L)\cdot\mathbf{h}^{t-1}\right),\\
\mathbf{r} &= \sigma\left(W_{\mathbf{rx}}(L)\cdot\mathbf{x}^t + W_{\mathbf{rh}}(L)\cdot\mathbf{h}^{t-1}\right),\\
\tilde{\mathbf{h}} &= \tanh\left(W_{\mathbf{hx}}(L)\cdot\mathbf{x}^t + W_{\mathbf{hh}}(L)\cdot(\mathbf{r}\odot\mathbf{h}^{t-1})\right),\\
\mathbf{h}^t &= \mathbf{s}\odot\mathbf{h}^{t-1} + (1-\mathbf{s})\odot\tilde{\mathbf{h}}
\end{aligned}
$$

for $t = 1, \dots, T$. Here, $W_{\cdot\mathbf{x}} : \mathbb{R}^{N\times N} \to \mathbb{R}^{Q\times d_{\mathbf{h}}\times K}$ and $W_{\cdot\mathbf{h}} : \mathbb{R}^{N\times N} \to \mathbb{R}^{Q\times d_{\mathbf{h}}\times d_{\mathbf{h}}}$ are graph convolution operators with $d_{\mathbf{h}}K$ filters that are parameterised by Q Chebyshev coefficients, taking the form

$$
G(L) = \sum_{q=1}^{Q} a_q H_q(L), \tag{60}
$$

where $H_q(L)$ is the q-th order Chebyshev polynomial evaluated at L. We use $Q = 3$ Chebyshev coefficients in the graph filtering operation and choose a hidden state size of $d_{\mathbf{h}} = 64$, such that the hidden state of each agent is a 64-dimensional vector. A single linear layer reduces this $N \times 64$ matrix into an N-vector, where N is the number of agents in the system. An embedding of the entire graph then proceeds by passing this N-vector through a feedforward network with layer sizes $32, 16, 16$. In our experiments, we take $N = 20$ and simulate for $T = 25$ time steps.

For all neural posterior estimation tasks, we use a masked autoregressive flow (Papamakarios et al., 2017) with 5 flow transforms, each with 2 blocks and 50 hidden features; for all neural density ratio estimation tasks, we use a residual network with two layers of size 50. To train the network weights, we use Adam (Kingma and Ba, 2014), along with a training batch size of 50 and learning rate of 5×10^{-4}. We furthermore reserve 10% of the data for validation, and stop training when the validation error does not improve over 20 epochs to avoid overfitting. Throughout, we use the `sbi` python package (Tejero-Cantero et al., 2020) and PyTorch Geometric Temporal (Rozemberczki et al., 2021) `python` packages.

References

Abadi, Martín, Barham, Paul, Chen, Jianmin, Chen, Zhifeng, Davis, Andy, Dean, Jeffrey, Devin, Matthieu, Ghemawat, Sanjay, Irving, Geoffrey, Isard, Michael, et al., 2016. Tensorflow: a system for large-scale machine learning. In: 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16), pp. 265–283.

Alsing, Justin, Charnock, Tom, Feeney, Stephen, Wandelt, Benjamin, 2019. Fast likelihood-free cosmology with neural density estimators and active learning. Mon. Not. R. Astron. Soc. (ISSN 0035-8711) 488 (3), 4440–4458. https://doi.org/10.1093/mnras/stz1960.

Andrieu, Christophe, Roberts, Gareth O., 2009. The pseudo-marginal approach for efficient Monte Carlo computations. Ann. Stat. 37 (2), 697–725.

Arjovsky, Martin, Chintala, Soumith, Bottou, Léon, 2017. Wasserstein generative adversarial networks. In: Precup, Doina, Teh, Yee Whye (Eds.), Proceedings of the 34th International Conference on Machine Learning. In: Proceedings of Machine Learning Research, vol. 70. PMLR, pp. 214–223. https://proceedings.mlr.press/v70/arjovsky17a.html.

Axtell, Robert L., Farmer, J. Doyne, 2022. Agent-based modeling in economics and finance: past, present, and future. J. Econ. Lit.

Baek, Jinheon, Kang, Minki, Hwang, Sung Ju, 2021. Accurate learning of graph representations with graph multiset pooling. In: ICLR. https://openreview.net/forum?id=JHcqXGaqiGn.

Barde, Sylvain, 2017. A practical, accurate, information criterion for nth order Markov processes. Comput. Econ. 50 (2), 281–324.

Beaumont, Mark A., 2010. Approximate Bayesian computation in evolution and ecology. Annu. Rev. Ecol. Evol. Syst. 41, 379–406.

Beaumont, Mark A., Zhang, Wenyang, Balding, David J., 2002. Approximate Bayesian computation in population genetics. Genetics 162 (4), 2025–2035.

Beaumont, Mark A., Cornuet, Jean-Marie, Marin, Jean-Michel, Robert, Christian P., 2009. Adaptive approximate Bayesian computation. Biometrika 96 (4), 983–990.

Benedetti, Marco, Catapano, Gennaro, De Sclavis, Francesco, Favorito, Marco, Glielmo, Aldo, Magnanimi, Davide, Muci, Antonio, 2022. Black-it: a ready-to-use and easy-to-extend calibration kit for agent-based models. J. Open Sour. Softw. 7 (79), 4622. https://doi.org/10.21105/joss.04622.

Bernton, Espen, Jacob, Pierre E., Gerber, Mathieu, Robert, Christian P., 2019. Approximate Bayesian computation with the Wasserstein distance. J. R. Stat. Soc., Ser. B, Stat. Methodol. 81 (2), 235–269. https://doi.org/10.1111/rssb.12312. ISSN 14679868.

Bishop, Christopher M., 1994. Mixture Density Networks.

Blum, Michael G.B., Nunes, Maria Antonieta, Prangle, Dennis, Sisson, Scott A., 2013. A comparative review of dimension reduction methods in approximate Bayesian computation. Stat. Sci. 28 (2), 189–208.

Bornn, Luke, Pillai, Natesh S., Smith, Aaron, Woodard, Dawn, 2017. The use of a single pseudo-sample in approximate Bayesian computation. Stat. Comput. 27 (3), 583–590.
Brehmer, Johann, Cranmer, Kyle, Louppe, Gilles, Pavez, Juan, 2018. Constraining effective field theories with machine learning. Phys. Rev. Lett. 121, 111801. https://doi.org/10.1103/PhysRevLett.121.111801.
Briol, François Xavier, Barp, Alessandro, Duncan, Andrew B., Girolami, Mark, 2019. Statistical inference for generative models with maximum mean discrepancy. arXiv, pp. 1–57.
Brock, William A., Hommes, Cars H., 1998. Heterogeneous beliefs and routes to chaos in a simple asset pricing model. J. Econ. Dyn. Control (ISSN 0165-1889) 22 (8), 1235–1274. https://doi.org/10.1016/S0165-1889(98)00011-6.
Bruna, Joan, Zaremba, Wojciech, Szlam, Arthur, LeCun, Yann, 2014. Spectral networks and deep locally connected networks on graphs. In: 2nd International Conference on Learning Representations, ICLR 2014.
Cannon, Patrick, Ward, Daniel, Schmon, Sebastian M., 2022. Investigating the impact of model misspecification in neural simulation-based inference. https://arxiv.org/abs/2209.01845.
Chen, Yanzhi, Zhang, Dinghuai, Gutmann, Michael, Courville, Aaron, Zhu, Zhanxing, 2020. Neural Approximate Sufficient Statistics for Implicit Models, pp. 1–14. http://arxiv.org/abs/2010.10079.
Cranmer, Kyle, Pavez, Juan, Louppe, Gilles, 2015. Approximating likelihood ratios with calibrated discriminative classifiers. arXiv preprint. arXiv:1506.02169.
Darmois, G., 1935. Sur les lois de probabilites a estimation exhaustive. C. R. Acad. Sci. Paris 200, 1265–1266 (in French).
Diggle, Peter J., Gratton, Richard J., 1984. Monte Carlo methods of inference for implicit statistical models. J. R. Stat. Soc., Ser. B, Methodol. 46 (2), 193–212.
Doucet, Arnaud, Pitt, Michael K., Deligiannidis, George, Kohn, Robert, 2015. Efficient implementation of Markov chain Monte Carlo when using an unbiased likelihood estimator. Biometrika 102 (2), 295–313.
Durkan, Conor, Murray, Iain, Papamakarios, George, 2020. On contrastive learning for likelihood-free inference. In: Daumé III, Hal, Singh, Aarti (Eds.), Proceedings of the 37th International Conference on Machine Learning. In: Proceedings of Machine Learning Research, vol. 119. PMLR, pp. 2771–2781. http://proceedings.mlr.press/v119/durkan20a.html.
Dyer, Joel, Cannon, Patrick, Schmon, Sebastian M., 2021a. Approximate Bayesian computation with path signatures. arXiv preprint. arXiv:2106.12555.
Dyer, Joel, Cannon, Patrick W., Schmon, Sebastian M., 2021b. Deep signature statistics for likelihood-free time-series models. In: ICML Workshop on Invertible Neural Networks, Normalizing Flows, and Explicit Likelihood Models.
Dyer, Joel, Cannon, Patrick, Farmer, J. Doyne, Schmon, Sebastian M., 2022a. Calibrating agent-based models to microdata with graph neural networks. In: ICML 2022 Workshop AI for Agent-Based Modelling.
Dyer, Joel, Cannon, Patrick W., Schmon, Sebastian M., 2022b. Amortised likelihood-free inference for expensive time-series simulators with signatured ratio estimation. In: International Conference on Artificial Intelligence and Statistics. PMLR, pp. 11131–11144.
Fagiolo, Giorgio, Guerini, Mattia, Lamperti, Francesco, Moneta, Alessio, Roventini, Andrea, 2019. Validation of agent-based models in economics and finance. In: Computer Simulation Validation: Fundamental Concepts, Methodological Frameworks, and Philosophical Perspectives, pp. 763–787.
Fearnhead, Paul, Prangle, Dennis, 2012. Constructing summary statistics for approximate Bayesian computation: semi-automatic approximate Bayesian computation. J. R. Stat. Soc., Ser. B, Stat. Methodol. 74 (3), 419–474. https://doi.org/10.1111/j.1467-9868.2011.01010.x. ISSN 13697412.
Franke, Reiner, 2009. Applying the method of simulated moments to estimate a small agent-based asset pricing model. J. Empir. Finance 16 (5), 804–815.
Franke, Reiner, Westerhoff, Frank, 2012. Structural stochastic volatility in asset pricing dynamics: estimation and model contest. J. Econ. Dyn. Control 36 (8), 1193–1211.
Gelman, Andrew, Carlin, John B., Stern, Hal S., Rubin, Donald B., 1995. Bayesian Data Analysis. Chapman and Hall/CRC.
Gelman, Andrew, Roberts, Gareth O., Gilks, Walter R., et al., 1996. Efficient Metropolis jumping rules. Bayesian Stat. 5 (599–608), 42.
Gonçalves, Pedro J., Lueckmann, Jan-Matthis, Deistler, Michael, Nonnenmacher, Marcel, Öcal, Kaan, Bassetto, Giacomo, Chintaluri, Chaitanya, Podlaski, William F., Haddad, Sara A., Vogels, Tim P., et al., 2020. Training deep neural density estimators to identify mechanistic models of neural dynamics. eLife 9, e56261.
Gourieroux, Christian, Monfort, Alain, Renault, Eric, 1993. Indirect inference. J. Appl. Econom. 8 (S1), S85–S118.
Grazzini, Jakob, Richiardi, Matteo G., Tsionas, Mike, 2017. Bayesian estimation of agent-based models. J. Econ. Dyn. Control 77, 26–47.
Greenberg, David S., Nonnenmacher, Marcel, Macke, Jakob H., 2019. Automatic posterior transformation for likelihood-free inference. In: 36th International Conference on Machine Learning, ICML 2019. June 2019, pp. 4288–4304.
Gretton, Arthur, Borgwardt, Karsten, Rasch, Malte, Schölkopf, Bernhard, Smola, Alex, 2006. A kernel method for the two-sample-problem. Adv. Neural Inf. Process. Syst. 19, 513–520.
Gretton, Arthur, Borgwardt, Karsten M., Rasch, Malte J., Schölkopf, Bernhard, Smola, Alexander, 2012. A kernel two-sample test. J. Mach. Learn. Res. 13 (25), 723–773. http://jmlr.org/papers/v13/gretton12a.html.
Hastings, W.K., 1970. Monte Carlo sampling methods using Markov chains and their applications. Biometrika 57, 97–109.
Hermans, Joeri, Begy, Volodimir, Louppe, Gilles, 2020. Likelihood-free MCMC with amortized approximate ratio estimators. In: Daumé III, Hal, Singh, Aarti (Eds.), Proceedings of the 37th International Conference on Machine Learning. In: Proceedings of Machine Learning Research, vol. 119, pp. 4239–4248. https://proceedings.mlr.press/v119/hermans20a.html.
Ju, Nianqiao, Heng, Jeremy, Jacob, Pierre E., 2021. Sequential Monte Carlo algorithms for agent-based models of disease transmission. arXiv preprint. arXiv:2101.12156.
Kantorovich, L.V., 1960. Mathematical methods of organizing and planning production. Manag. Sci. 6 (4), 366–422. ISSN 00251909, 15265501. http://www.jstor.org/stable/2627082.
Kidger, Patrick, Morrill, James, Foster, James, Lyons, Terry, 2020. Neural controlled differential equations for irregular time series. arXiv preprint. arXiv:2005.08926.
Kingma, Diederik P., Ba, Jimmy, 2014. Adam: a method for stochastic optimization. arXiv preprint. arXiv:1412.6980.
Kingma, Diederik P., Dhariwal, Prafulla, 2018. Glow: generative flow with invertible 1x1 convolutions. arXiv preprint. arXiv:1807.03039.
Knoblauch, Jeremias, Jewson, Jack, Damoulas, Theodoros, 2019. Generalized variational inference: three arguments for deriving new posteriors. arXiv preprint. arXiv:1904.02063.
Koopman, B.O., 1936. On distributions admitting a sufficient statistic. Trans. Am. Math. Soc. 39 (3), 399–409. http://www.jstor.org/stable/1989758. ISSN 00029947.
Kukacka, Jiri, Barunik, Jozef, 2017. Estimation of financial agent-based models with simulated maximum likelihood. J. Econ. Dyn. Control 85, 21–45.
Lamperti, Francesco, 2018a. Empirical validation of simulated models through the gsl-div: an illustrative application. J. Econ. Interact. Coord. 13, 143–171.
Lamperti, Francesco, 2018b. An information theoretic criterion for empirical validation of simulation models. Econom. Stat. 5, 83–106.
Leskovec, Jure, Krevl, Andrej, 2014. SNAP datasets: Stanford large network dataset collection. http://snap.stanford.edu/data.
Li, Yaguang, Yu, Rose, Shahabi, Cyrus, Liu, Yan, 2017. Diffusion convolutional recurrent neural network: data-driven traffic forecasting. arXiv preprint. arXiv:1707.01926.
Liepe, Juliane, Kirk, Paul, Filippi, Sarah, Toni, Tina, Barnes, Chris P., Stumpf, Michael P.H., 2014. A framework for parameter estimation and model selection from experimental data in systems biology using approximate Bayesian computation. Nat. Protoc. 9 (2), 439–456.
Lueckmann, Jan-Matthis, Goncalves, Pedro J., Bassetto, Giacomo, Öcal, Kaan, Nonnenmacher, Marcel, Macke, Jakob H., 2017. Flexible statistical inference for mechanistic models of neural dynamics. Adv. Neural Inf. Process. Syst. 30.
Lueckmann, Jan-Matthis, Boelts, Jan, Greenberg, David, Goncalves, Pedro, Macke, Jakob, 2021. Benchmarking simulation-based inference. In: Banerjee, Arindam, Fukumizu, Kenji (Eds.), Proceedings of the 24th International Conference on Artificial Intelligence and Statistics. In: Proceedings of Machine Learning Research, vol. 130. PMLR, pp. 343–351.

Lux, Thomas, 2018. Estimation of agent-based models using sequential Monte Carlo methods. J. Econ. Dyn. Control 91, 391–408.
Lux, Thomas, 2021. Bayesian estimation of agent-based models via adaptive particle Markov chain Monte Carlo. Comput. Econ., 1–27.
Lux, Thomas, Zwinkels, Remco C.J., 2018. Empirical validation of agent-based models. In: Handbook of Computational Economics, vol. 4. Elsevier, pp. 437–488.
Macy, Michael W., Kitts, James A., Flache, Andreas, Benard, Steve, 2003. Polarization in dynamic networks: a Hopfield model of emergent structure. In: Dynamic Social Network Modelling and Analysis, pp. 162–173.
Malleson, Nick, Minors, Kevin, Kieu, Le-Minh, Ward, Jonathan A., West, Andrew, Heppenstall, Alison, 2020. Simulating crowds in real time with agent-based modelling and a particle filter. J. Artif. Soc. Soc. Simul. 23 (3).
Masci, Jonathan, Boscaini, Davide, Bronstein, Michael M., Vandergheynst, Pierre, 2015. Geodesic convolutional neural networks on Riemannian manifolds. In: 2015 IEEE International Conference on Computer Vision Workshop (ICCVW), pp. 832–840.
Miller, Jeffrey W., Dunson, David B., 2018. Robust Bayesian inference via coarsening. J. Am. Stat. Assoc.
Morrill, James, Salvi, Cristopher, Kidger, Patrick, Foster, James, 2021. Neural rough differential equations for long time series. In: International Conference on Machine Learning. PMLR, pp. 7829–7838.
Mungo, Luca, Lafond, François, Astudillo-Estévez, Pablo, Farmer, J. Doyne, 2023. Reconstructing production networks using machine learning. J. Econ. Dyn. Control (ISSN 0165-1889) 148, 104607. https://doi.org/10.1016/j.jedc.2023.104607.
Niepert, Mathias, Ahmed, Mohamed, Kutzkov, Konstantin, 2016. Learning convolutional neural networks for graphs. In: International Conference on Machine Learning. PMLR, pp. 2014–2023.
Noé, Frank, Olsson, Simon, Köhler, Jonas, Wu, Hao, 2019. Boltzmann generators: sampling equilibrium states of many-body systems with deep learning. Science 365 (6457), eaaw1147. https://doi.org/10.1126/science.aaw1147.
Papamakarios, George, Murray, Iain, 2016. Fast ε-free inference of simulation models with Bayesian conditional density estimation. In: Advances in Neural Information Processing Systems, pp. 1028–1036.
Papamakarios, George, Pavlakou, Theo, Murray, Iain, 2017. Masked autoregressive flow for density estimation. In: Proceedings of the 31st International Conference on Neural Information Processing Systems, pp. 2335–2344.
Papamakarios, George, Sterratt, David, Murray, Iain, 2019. Sequential neural likelihood: fast likelihood-free inference with autoregressive flows. In: The 22nd International Conference on Artificial Intelligence and Statistics. PMLR, pp. 837–848.
Park, Mijung, Jitkrittum, Wittawat, Sejdinovic, Dino, 2016. K2-ABC: approximate Bayesian computation with kernel embeddings. In: Proceedings of the 19th International Conference on Artificial Intelligence and Statistics, AISTATS 2016, vol. 41, pp. 398–407.
Paszke, Adam, Gross, Sam, Massa, Francisco, Lerer, Adam, Bradbury, James, Chanan, Gregory, Killeen, Trevor, Lin, Zeming, Gimelshein, Natalia, Antiga, Luca, et al., 2019. Pytorch: an imperative style, high-performance deep learning library. Adv. Neural Inf. Process. Syst. 32, 8026–8037.
Peters, Ole, Klein, William, 2013. Ergodicity breaking in geometric Brownian motion. Phys. Rev. Lett. 110 (10), 100603.
Pham, Kim Cuc, Nott, David J., Chaudhuri, Sanjay, 2014. A note on approximating abc-mcmc using flexible classifiers. Stat 3 (1), 218–227.
Pichler, Anton, Lafond, François, Farmer, J. Doyne, 2020. Technological interdependencies predict innovation dynamics. arXiv preprint. arXiv:2003.00580.
Pitman, Edwin James George, 1936. Sufficient statistics and intrinsic accuracy. In: Mathematical Proceedings of the Cambridge Philosophical Society, vol. 32. Cambridge University Press, pp. 567–579.
Platt, Donovan, 2020. A comparison of economic agent-based model calibration methods. J. Econ. Dyn. Control (ISSN 0165-1889) 113, 103859. https://doi.org/10.1016/j.jedc.2020.103859.
Platt, Donovan, 2021. Bayesian estimation of economic simulation models using neural networks. Comput. Econ., 1–52.
Price, Leah F., Drovandi, Christopher C., Lee, Anthony, Nott, David J., 2018. Bayesian synthetic likelihood. J. Comput. Graph. Stat. 27 (1), 1–11.
Pritchard, Jonathan K., Seielstad, Mark T., Perez-Lezaun, Anna, Feldman, Marcus W., 1999. Population growth of human Y chromosomes: a study of Y chromosome microsatellites. Mol. Biol. Evol. 16 (12), 1791–1798.
Rezende, Danilo, Mohamed, Shakir, 2015. Variational inference with normalizing flows. In: Bach, Francis, Blei, David (Eds.), Proceedings of the 32nd International Conference on Machine Learning. Lille, France. In: Proceedings of Machine Learning Research, vol. 37. PMLR, pp. 1530–1538. https://proceedings.mlr.press/v37/rezende15.html.
Roberts, Gareth O., Gelman, Andrew, Gilks, Walter R., 1997. Weak convergence and optimal scaling of random walk Metropolis algorithms. Ann. Appl. Probab. 7 (1), 110–120.
Rossi, Emanuele, Chamberlain, Ben, Frasca, Fabrizio, Eynard, Davide, Monti, Federico, Bronstein, Michael, 2020. Temporal graph networks for deep learning on dynamic graphs. arXiv preprint. arXiv:2006.10637.
Rozemberczki, Benedek, Scherer, Paul, He, Yixuan, Panagopoulos, George, Riedel, Alexander, Astefanoaei, Maria, Kiss, Oliver, Beres, Ferenc, Lopez, Guzman, Collignon, Nicolas, Sarkar, Rik, 2021. PyTorch geometric temporal: spatiotemporal signal processing with neural machine learning models. In: Proceedings of the 30th ACM International Conference on Information and Knowledge Management, pp. 4564–4573.
Schmon, Sebastian M., Gagnon, Philippe, 2022. Optimal scaling of random walk Metropolis algorithms using Bayesian large-sample asymptotics. Stat. Comput. 32 (2).
Schmon, Sebastian M., Cannon, Patrick W., Knoblauch, Jeremias, 2020. Generalized posteriors in approximate Bayesian computation. In: Third Symposium on Advances in Approximate Bayesian Inference.
Schmon, Sebastian M., Deligiannidis, George, Doucet, Arnaud, Pitt, Michael K., 2021. Large-sample asymptotics of the pseudo-marginal method. Biometrika 108 (1), 37–51.
Seo, Youngjoo, Defferrard, Michaël, Vandergheynst, Pierre, Bresson, Xavier, 2018. Structured sequence modeling with graph convolutional recurrent networks. In: International Conference on Neural Information Processing. Springer, pp. 362–373.
Sherlock, Chris, Thiery, Alexandre H., Roberts, Gareth O., Rosenthal, Jeffrey S., 2015. On the efficiency of pseudo-marginal random walk Metropolis algorithms. Ann. Stat. 43 (1), 238–275.
Sherlock, Chris, Thiery, Alexandre H., Lee, Anthony, 2017. Pseudo-marginal Metropolis–Hastings sampling using averages of unbiased estimators. Biometrika 104 (3), 727–734.
Shiono, Takashi, 2021. Estimation of agent-based models using Bayesian deep learning approach of bayesflow. J. Econ. Dyn. Control 125, 104082.
Silverman, Bernard W., 1986. Density Estimation for Statistics and Data Analysis, vol. 26. CRC Press.
Sunnåker, Mikael, Busetto, Alberto Giovanni, Numminen, Elina, Corander, Jukka, Foll, Matthieu, Dessimoz, Christophe, 2013. Approximate Bayesian computation. PLoS Comput. Biol. 9 (1), e1002803.
Tabak, Esteban G., Turner, Cristina V., 2013. A family of nonparametric density estimation algorithms. Commun. Pure Appl. Math. 66 (2), 145–164.
Tabak, Esteban G., Vanden-Eijnden, Eric, 2010. Density estimation by dual ascent of the log-likelihood. Commun. Math. Sci. 8 (1), 217–233.
Talts, Sean, Betancourt, Michael, Simpson, Daniel, Vehtari, Aki, Gelman, Andrew, 2020. Validating Bayesian Inference Algorithms with Simulation-Based Calibration.
Tavaré, Simon, Balding, David J., Griffiths, Robert C., Donnelly, Peter, 1997. Inferring coalescence times from dna sequence data. Genetics 145 (2), 505–518.
Tejero-Cantero, Alvaro, Boelts, Jan, Deistler, Michael, Lueckmann, Jan-Matthis, Durkan, Conor, Gonçalves, Pedro J., Greenberg, David S., Macke, Jakob H., 2020. sbi: a toolkit for simulation-based inference. J. Open Sour. Softw. 5 (52), 2505. https://doi.org/10.21105/joss.02505.
Thomas, Owen, Dutta, Ritabrata, Corander, Jukka, Kaski, Samuel, Gutmann, Michael U., 2021. Likelihood-free inference by ratio estimation. Bayesian Anal., 1–31. https://doi.org/10.1214/20-BA1238.
Toni, Tina, Welch, David, Strelkowa, Natalja, Ipsen, Andreas, Stumpf, Michael P.H., 2009. Approximate Bayesian computation scheme for parameter inference and model selection in dynamical systems. J. R. Soc. Interface 6 (31), 187–202.

Ward, Daniel, Cannon, Patrick, Beaumont, Mark, Fasiolo, Matteo, Schmon, Sebastian M., 2022. Robust neural posterior estimation and statistical model criticism. Adv. Neural Inf. Process. Syst. 35.

Wilkinson, Richard David, 2013. Approximate Bayesian computation (abc) gives exact results under the assumption of model error. Stat. Appl. Genet. Mol. Biol. 12 (2), 129–141.

Wiqvist, Samuel, Mattei, Pierre-Alexandre, Picchini, Umberto, Frellsen, Jes, 2019. Partially exchangeable networks and architectures for learning summary statistics in approximate Bayesian computation. In: International Conference on Machine Learning. PMLR, pp. 6798–6807.

Wong, Wing, Jiang, Bai, Wu, Tung-yu, Zheng, Charles, 2018. Learning summary statistic for approximate Bayesian computation via deep neural network. Stat. Sin. (ISSN 1017-0405). https://doi.org/10.5705/ss.202015.0340.

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

Wu, Zonghan, Pan, Shirui, Chen, Fengwen, Long, Guodong, Zhang, Chengqi, Yu, Philip S., 2021. A comprehensive survey on graph neural networks. IEEE Trans. Neural Netw. Learn. Syst. 32 (1), 4–24. https://doi.org/10.1109/TNNLS.2020.2978386.

Yao, Liang, Mao, Chengsheng, Luo, Yuan, 2019. Graph convolutional networks for text classification. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 7370–7377.

Zaheer, Manzil, Kottur, Satwik, Ravanbakhsh, Siamak, Poczos, Barnabas, Salakhutdinov, Russ R., Smola, Alexander J., 2017. Deep sets. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (Eds.), Advances in Neural Information Processing Systems, vol. 33. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2017/file/f22e4747da1aa27e363d86d40ff442fe-Paper.pdf.

Zhang, Yuyu, Chen, Xinshi, Yang, Yuan, Ramamurthy, Arun, Li, Bo, Qi, Yuan, Song, Le, 2020. Efficient probabilistic logic reasoning with graph neural networks. In: International Conference on Learning Representations. https://openreview.net/forum?id=rJg76kStwH.

Zhou, Jie, Cui, Ganqu, Hu, Shengding, Zhang, Zhengyan, Yang, Cheng, Liu, Zhiyuan, Wang, Lifeng, Li, Changcheng, Sun, Maosong, 2020. Graph neural networks: a review of methods and applications. AI Open (ISSN 2666-6510) 1, 57–81. https://doi.org/10.1016/j.aiopen.2021.01.001.