Simulation-Based Inference for Neuroscience and Beyond

Dissertation
der Mathematisch-Naturwissenschaftlichen Fakultät
der Eberhard Karls Universität Tübingen
zur Erlangung des Grades eines
Doktors der Naturwissenschaften
(Dr. rer. nat.)

vorgelegt von
Jan-Matthis Lückmann
aus Hamburg

Tübingen
2021

Gedruckt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Eberhard Karls Universität Tübingen.

Tag der mündlichen Qualifikation:	11.07.2022
Dekan:	Prof. Dr. Thilo Stehle
1. Berichterstatter:	Prof. Dr. Jakob H. Macke
2. Berichterstatter:	Prof. Dr. Philipp Berens
3. Berichterstatter:	Prof. Dr. Luigi Acerbi

ABSTRACT

Science makes extensive use of simulations to model the world. Statistical inference identifies which models are consistent with observed phenomena, thus bridging the gap between theory and reality. However, conventional statistical inference is often inapplicable to detailed simulation models because their associated likelihood functions are intractable. Simulation-based inference (SBI) addresses this problem: It allows statistical inference from simulations alone and can thus be used with implicit models, which lack evaluable likelihoods. This thesis consists of four publications that draw on advances in machine learning to contribute to the transition away from heuristic approaches towards principled statistical inference with SBI, which allows to identify data-consistent models. To this end, this thesis proposes new algorithms, applications to neuroscience, and the first unified benchmark for SBI. Overall, it shows the potential for fast and flexible likelihood-free algorithms to facilitate scientific discovery in neuroscience and beyond.

The trade-off between models of neural dynamics that are statistically amenable or mechanistically plausible was the starting point for the work presented in this thesis. In the first publication, we introduce an SBI algorithm for sequential neural posterior estimation, which overcomes the drawbacks of an earlier method. We provide several extensions motivated by challenging problems in neuroscience, including end-to-end learning of summary statistics for high-dimensional time series data. In the second publication, we demonstrate its broad applicability to mechanistic models in neuroscience—from the scale of ion channels, which are the basic building blocks of biophysical neuron models, to network models of neural dynamics. Our approach overcomes the limitations of heuristic alternatives and narrows the divide between statistical and mechanistic models. The third publication proposes a novel SBI algorithm that proceeds by learning an emulator of the simulator. This approach enables the use of active learning schemes to adaptively acquire new simulations, which allows scaling to problems that are computationally highly expensive. With rapid progress in SBI, the need for a unified benchmark became apparent: In the fourth publication, we propose the first benchmark for the field to transparently evaluate progress and to contribute to more efficient and reproducible science.

ZUSAMMENFASSUNG

Die Wissenschaft macht ausgiebig Gebrauch von Simulationen, um die Welt zu modellieren. Statistische Inferenz ermittelt, welche Modelle mit beobachteten Phänomenen übereinstimmen, wodurch die Kluft zwischen Theorie und Realität überbrückt wird. Konventionelle statistische Inferenz ist jedoch oft nicht auf detaillierte Simulationsmodelle anwendbar, da die dazugehörigen Likelihood-Funktionen unauswertbar sind. Simulations-basierte Inferenz (SBI) adressiert dieses Problem: SBI ermöglicht statistische Inferenz alleinig auf Basis von Simulationen und kann daher für implizite Modelle verwendet werden, für die es keine auswertbaren Likelihoods gibt. Diese Arbeit besteht aus vier Veröffentlichungen, die sich auf Fortschritte im maschinellen Lernen stützen, um zum Übergang von heuristischen Ansätzen zu fundierter statistischer Inferenz mit SBI beizutragen. Damit wird ermöglicht, datenkonsistente Modelle zu identifizieren. Zu diesem Zweck werden neue Algorithmen, Anwendungen in den Neurowissenschaften und der erste einheitliche Benchmark für SBI beigetragen. Insgesamt wird das Potenzial für schnelle und flexible likelihood-freie Algorithmen aufgezeigt, die wissenschaftliche Entdeckung in den Neurowissenschaften und darüber hinaus ermöglichen.

Die Abwägung zwischen statistisch angemessenen und mechanistisch plausiblen Modellen der neuronalen Dynamik war der Ausgangspunkt für die hier vorgestellte Arbeit. In der ersten Veröffentlichung wird ein SBI-Algorithmus für sequenzielle neuronale Posteriorschätzung vorgestellt, der die Nachteile einer früheren Methode überwindet. Wir führen mehrere Erweiterungen ein, die durch schwierige Probleme in der Neurowissenschaft motiviert sind, einschließlich des Ende-zu-Ende-Lernens von zusammenfassenden Statistiken für hochdimensionale Zeitreihendaten. In der zweiten Veröffentlichung demonstrieren wir die breite Anwendbarkeit der Methode auf mechanistische Modelle in den Neurowissenschaften — von der Größenordnung von Ionenkanälen, die grundlegenden Bausteine biophysikalischer Neuronenmodelle, bis hin zu Netzwerkmodellen neuronaler Dynamik. Unser Ansatz überwindet die Grenzen heuristischer Alternativen und verringert die Kluft zwischen statistischen und mechanistischen Modellen. In der dritten Veröffentlichung wird ein neuartiger SBI-Algorithmus vorgeschlagen, in welchem ein Emulator des Simulators gelernt wird. Dieser Ansatz ermöglicht den Einsatz aktiver Lernverfahren zur adaptiven Durchführung neuer Simulationen, was Skalierung auf Probleme, die hohen Rechenaufwand erfordern, ermöglicht. Mit den raschen Fortschritten in SBI wurde der Bedarf an einem einheitlichen Benchmark deutlich: In der vierten Veröffentlichung schlagen wir den ersten Benchmark für das Feld vor, um Fortschritte transparent zu bewerten, und um zu mehr Effizienz und Reproduzierbarkeit in der Wissenschaft beizutragen.

ACKNOWLEDGMENTS

And you may ask yourself, "Well... how did I get here?"

Talking Heads — Once In A Lifetime

Thinking about this question makes me realize how unpredictable life is and how lucky I am for things turning out the way they did. It makes me tremendously grateful to many people without whom I certainly would not have gotten here.

I want to express my deepest gratitude to my advisor, Jakob Macke, who enormously impacted my thinking. I thank him for his trust, guidance, and support throughout the years. He created an environment to grow and enjoy doing science—working together has been a great privilege.

I would like to extend my gratitude to all former and present members of his group, particularly Pedro Gonçalves, David Greenberg, Álvaro Tejero-Cantero, Artur Speiser, Auguste Schulz, Florian Franzen, Giacomo Bassetto, Jan Bölts, Kaan Öcal, Marcel Nonnenmacher, Michael Deistler, Poornima Ramesh, and Richard Gao. I thank the entire group for the shared time in Tübingen, Bonn, and Munich over the past years—for discussing science and life, good laughs and support, joint projects and pushing through deadlines, traveling together, and—when times changed—virtual tea times. I am very thankful for administrative support by Alexandra Petelski, Bettina Bosse, Brigitte Bauer, and Franziska Weiler.

I would like to thank Theofanis Karaletsos, who supervised me during an internship at Uber AI labs, for his enthusiasm and support. I also thank the Pyro team and fellow interns for our discussions and the time in San Francisco and at Lake Tahoe. I thank my external collaborators Tim Vogels, Chaitanya Chintaluri, and Bill Podlaski for our project on inference for ion channel models and for hosting Pedro and me in Oxford. I would like to extend a special thank you to all participants and organizers of workshops and summer schools that I had the fortune to meet, as well as to the contributors and developers of the many open-source projects on which this thesis rests.

I am honored that Philipp Berens, Luigi Acerbi, Anna Levina, and Peter Dayan unhesitatingly agreed to referee this dissertation and would like to thank them for their valuable time. For feedback on this thesis, I am grateful to Auguste Schulz, Jan Bölts, Leandra White, Michael Deistler, Nikolas Karalis, Pedro Gonçalves, Poornima Ramesh, and Richard Gao.

Finally, with all my heart, I thank my friends, partner, and family for their encouragement and love.

CONTENTS

LIST OF PUBLICATIONS

Lueckmann*, J.-M., Gonçalves*, P. J., Bassetto, G., Öcal, K., Nonnenmacher, M., & Macke, J. H. (2017). Flexible statistical inference for mechanistic models of neural dynamics. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems 30* (pp. 1289–1299). Curran Associates, Inc.
* Equal contribution

Gonçalves*, P. J., Lueckmann*, J.-M., Deistler*, M., Nonnenmacher, M., Öcal, K., Bassetto, G., Chintaluri, C., Podlaski, W. F., Haddad, S. A., Vogels, T. P., & Macke, J. H. (2020). Training deep neural density estimators to identify mechanistic models of neural dynamics. *Elife*, 9, e56261.
* Equal contribution

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

Lueckmann, J.-M., Boelts, J., Greenberg, D. S., Gonçalves, P. J., & Macke, J. H. (2021). Benchmarking Simulation-Based Inference. In A. Banerjee & K. Fukumizu (Eds.), *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research* (pp. 343–351). PMLR.

CHAPTER 1

INTRODUCTION

1.1 OBSERVATIONS AND MODELS IN NEUROSCIENCE: PAST AND PRESENT

In 1939, Alan Hodgkin and Andrew Huxley made a historical *observation*: Using an ingenious system of mirrors, they inserted a micro-electrode made from a fine glass tube and silver wire for electric contact into the giant nerve fiber of a squid (*Loligo forbesii*). Placing another electrode in the seawater outside the nerve, they measured the potential difference across the membrane, obtaining the first intracellular recording of an action potential (Figure 1.1A).[1] Across species, information processing in neural systems relies on transmission of such short, stereotypical 'spikes' of electric activity along nerve fibers.[2] To put it figuratively, spikes are the language of our brains, which are composed of a staggering number of approximately 86 billion neurons connected through ten million kilometers of nerve fibers.[3] Ultimately, all our perception, memory, thoughts, and behavior arise from neurons talking to other neurons.

Following wartime duties, Hodgkin and Huxley turned towards creating a *model* of their observation, aiming to explain the shape of action potentials. They proposed an equivalent circuit model as an analogy for nerve cells. In the model, a capacitor serves as a stand-in for the membrane, and three parallel resistors account for ions flowing across different types of hypothesized channels (Figure 1.1B).[4] By 1951, they had settled on the equations and constants associated with their circuit model and were hoping to *simulate* them on one of the earliest digital computers (EDSAC 1) at the University of Cambridge. Unfortunately, the computer was down for six-month maintenance, and so the differential equations underlying the model were laboriously computed using a hand-operated mechanical calculator (Brunsviga 20) over the course of three weeks.[5] This was well worth the effort: The Hodgkin-Huxley model, published in a series of consecutive papers (Hodgkin & Huxley, 1952a,b,c,d,e) provided the first quantitative account of the observed action potentials, faithfully reproducing biophysical properties. It is one of the fundamental breakthroughs in neuroscience, earning its inventors a Nobel Prize in 1963, forever shaping our mechanistic understanding of the biophysics of neurons.[6]

Fast-forward to today: Access to digital computers and the use of numerical simulations in neuroscience are ubiquitous.[7] Special-purpose simulation software and databases allow detailed specification of biophysical models, for example, incorporating today's knowledge of ion channels, the stunning diversity of neural morphologies and cell types, connectomics, and plasticity.[8] Progressively, more and more realistic models of mechanisms underlying neural dynamics, on a single neuron and population level, are formulated. Apart from aiming for such micro- or macroscopic biophysical realism, computer simulations in neuroscience are performed with many different goals in mind. This also includes striving for realistic behavior, for example, imitating how we remember, learn and make decisions, or realism regarding representations.[9] Along with this development, neuroscience is undergoing a data-driven revolution, with tools of unprecedented resolution for measuring brain dynamics and complex naturalistic behavior at its disposal: High-density probes and optical tools of today allow parallel recordings of thousands of neurons. Simultaneously, video and sensor data of behavior can be acquired with extreme precision.[10]

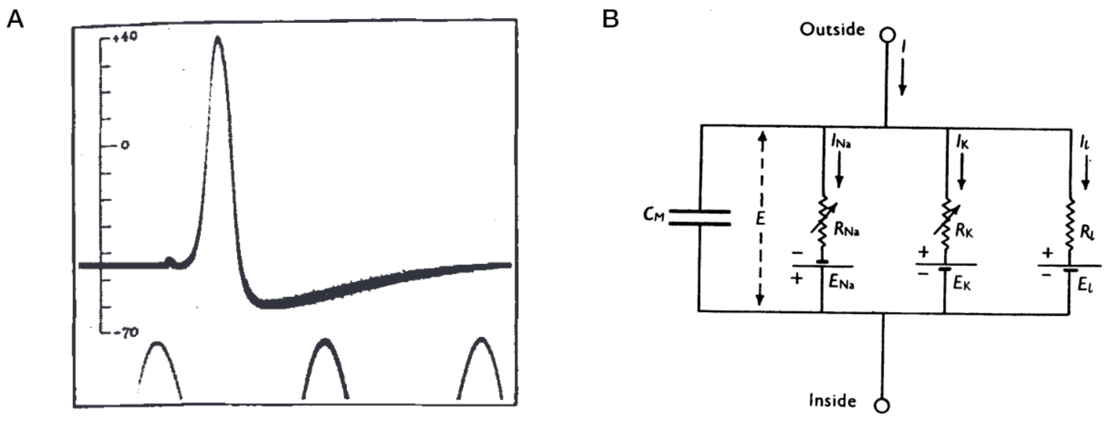


Figure 1.1: Observation and model by Hodgkin and Huxley. **A.** Hodgkin & Huxley (1939) observed the first intracellular action potentials, which they measured from the squid giant axon: Time, with a 500 Hz signal included for reference, plotted against the measured potential in mV. Reprinted by permission from Springer Nature Customer Service Centre GmbH: Springer Nature, Nature, Action potentials recorded from inside a nerve fibre. Hodgkin, A. L. and Huxley, A. F., 1939. **B.** In later work, they formulated a model representing the membrane as a electrical circuit (Hodgkin & Huxley, 1952e): Sodium (*Na*), potassium (*K*), and leak (*L*) conductances $\bar{g}$ are modeled as inverses of corresponding resistances (*R*). R_{Na} and R_K vary with membrane potential and time. Their model provided the first quantitative account of the time-course of action potential generation, capturing the essence of the underlying mechanism, which involves a short influx of sodium ions into the axon, followed by an efflux of potassium. Reprinted by permission from John Wiley and Sons: Wiley-Blackwell, The Journal of Physiology, A quantitative description of membrane current and its application to conduction and excitation in nerve. Hodgkin, A. L. and Huxley, A. F., 1952.

1.2 BRIDGING THE GAP BETWEEN REALITY AND THEORY

With these exciting advances comes a significant challenge at the heart of scientific progress: Given detailed, interpretable models that specify hypothesized cause-effect relationships unfolding over time in simulations on the one hand, and evermore complex observations on the other—how can we identify which models are consistent with observed data and knowledge?

Answering this question requires bridging the gap between observations and models. In the big picture, statistical inference is uniquely poised to serve as the bridge that connects the real world, in which observations are made, and the theoretical world, of which models are part of (Figure 1.2).[11] This way, statistical inference enables the *iteration* between theory and practice on which scientific learning and insight rely.[12] As its language, it uses probability, "the logic of uncertainty".[13] Variability and limited information are inherent to all scientific problems: Fundamentally, we never observe a *phenomenon* directly, but only through *probes* used for measurements, such as a fine glass tube and silver wire in the case of Hodgkin and Huxley. The phenomenon is embedded in its *environment*, for example, involving the experimental preparation. Phenomenon, probe, and environment all interact and affect the observation, inevitably leading to variability.[14] Similarly, we face incomplete knowledge when specifying models and are commonly confronted with systems that appear stochastic and noisy when observed on a microscopic scale. For example, for most synapses, a spike arriving at the terminal of an axon triggers neurotransmitter release only about half of the time, and spontaneous release is frequent.[15] Statistics offers a way to make principled decisions in light of uncertainty, no matter what the exact source of uncertainty is.[16]

Unfortunately, conventional statistical inference often poses prohibitively strict restrictions on which models can be used. Broadly speaking, identifying data-consistent model configurations requires numerical evaluation of the so-called *likelihood function*—which captures the *relative consistency* of any given model configuration with observations.[17]Commonly, however, this is infeasible for practical, computational, or mathematical reasons, which are detailed in Chapter 2.

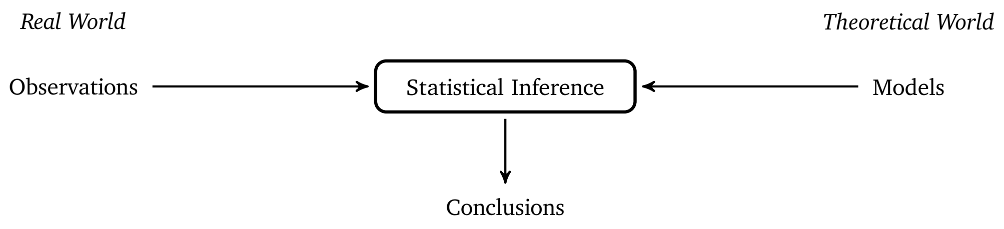


Figure 1.2: Statistical inference bridges the real world, one in which observations are made, with the theoretical world, inhabited by models. Bridging this gap allows us to draw conclusions, in particular, to answer the question which models are consistent with observed data and knowledge. Inspired by Kass (2011, Figure 1).

This creates a dilemma: When we design models with statistical considerations in mind, we have a close feedback loop between data and theory. However, this poses tight constraints on what models can be used. This strategy often only yields limited insights about underlying causal mechanisms. In contrast, when we design interpretable mechanistic models predestined for this purpose, for example, relying on high-fidelity computer simulations, we lose the ability to perform (likelihood-based) statistical inference. Constraining these models by observed data often poses an extremely difficult challenge.

1.3 DOWNSIDES OF HEURISTIC ALTERNATIVES TO STATISTICAL INFERENCE

We can illustrate a number of different existing approaches to this problem, and their respective limitations, by virtue of a simple but insightful example. Relating back to the beginning, let us assume we stimulated an axon by supplying a small fluctuating current and counted the resulting number of spikes in a short time window. We might then ask the question which configurations of a Hodgkin-Huxley model (Figure 1.1B) are consistent with what we observed. To keep this example simple, we only consider models which differ in sodium and potassium conductances. The standard toolkit of statistical inference cannot be applied to this example for lack of a tractable likelihood function.[18]

An ideal answer to our question is presented in Figure 1.3, showing the probability of obtaining a given number of spikes when simulating a Hodgkin-Huxley model as the conductances are varied.[19] The answer in this figure was obtained by brute force, that is, the model was simulated over a fine grid of combinations of the two conductances.[20] While this is feasible thanks to modern digital computers, this strategy quickly reaches its limits. In particular, it is only possible because the number of parameters that are varied is so low (only two). With each additional free parameter, the computational cost increases exponentially, rapidly exceeding feasible limits. Computing the results shown in the figure took about 70 hours (about 0.008 years) to complete; varying just one additional parameter at the same resolution would already take about 8 years to finish, two more 8 000 years, and so forth.[21] Note that models in neuroscience can have tens to hundreds of parameters.[22] In addition, individual simulations can take significantly longer than the ones considered here. This illustrates why brute-force exploration of the entire model configuration space is usually impossible in practice, and, more generally, how the cost of simulations places tight constraints on parameter exploration.

Instead, consistent model configurations are commonly searched for either manually, which is highly laborious and subjective, or using an automated search method.[23] Automated search procedures critically rely on the definition of their objective function, which measures the agreement between observed data and simulations.[24] In practice, high-dimensional data (for example, acquired by modern electrophysiological or optical tools) is often reduced to a low number of heuristically chosen features for the objective. The outcomes of any automatic search procedure will be highly sensitive to these choices. In addition to purposefully keeping the number of free parameters very low and choosing a fast simulator,

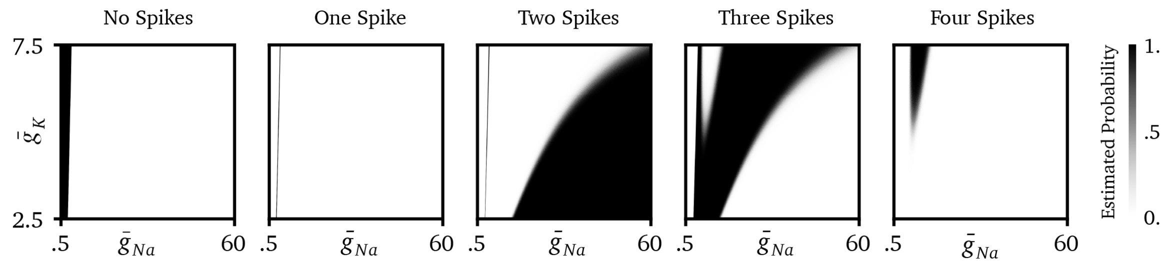


Figure 1.3: From left to right, this figure shows the estimated probability of a Hodgkin-Huxley model yielding a given spike count within a short time window with different configurations of potassium and sodium conductances, $\bar{g}_K$ and $\bar{g}_{Na}$ (both in mS/cm^2). The results were obtained by simulating over a finely-spaced grid of combinations of these two parameters. For any given spike count, there are many consistent solutions.

our simple example also side-steps this issue: By specifying upfront that we are only interested in spike counts, data is one-dimensional, and specifying an objective is straightforward.[25]

We can illustrate a final issue on this example: Search methods, unlike statistical procedures, do not quantify the uncertainty in returned solutions and are typically ill-suited to deal with non-uniqueness. Assume we run a common search algorithm to answer the question which model configurations are consistent with observing a given number of spikes. As a result, we obtain a 'best' solution in the form of a single pair of values for $\bar{g}_{Na}$, $\bar{g}_K$. Repeating the search, however, we might obtain an entirely different 'best' solution.[26] Considering Figure 1.3, we see why: There is a large number of consistent model configurations for any given spike count.

Since we only considered a single feature in our example, this might come as no surprise. However, reducing the dimensionality of data to few features is not the only way in which non-uniqueness arises: Having multiple solutions is closely associated with robustness. In neuroscience, this has been widely acknowledged through the work of Eve Marder and colleagues: Studying the stomatogastric nervous systems of lobsters (*Homarus americanus*), they found that the same functional output (a triphasic motor pattern) can be generated by a large variety of different synaptic strengths and intrinsic neuronal properties.[27] Similar behavior on the level of individual neurons and networks can be achieved by substantially different parameters through compensatory mechanisms. This ensures robustness, and ultimately survival, in the face of changing environmental conditions. This deep insight about biology is referred to as *degeneracy*.[28] The fundamental lesson is that studying a single 'best' model to understand the behavior of a system can be ill-conceived.[29]

1.4 STATISTICAL INFERENCE FOR IMPLICIT MODELS: CLASSICAL TECHNIQUES

These considerations and issues bring us back to statistical inference. Generally, statistical inference offers a principled alternative to the strategies discussed above and allows to infer the entire space of data-consistent models.[30] How can it be applied to mechanistic models with intractable likelihood functions?

As a matter of fact, some statisticians recognized the potential of extending statistical inference to models without tractable likelihoods decades ago—although they were yet significantly limited by the speed of computers.[31] For example, in 1984, Peter Diggle and Richard Gratton proposed an inference algorithm for *implicit models*, which is the name they gave to models with intractable likelihoods.[32] They refer to their algorithm as *simulation-based*, since it proceeds by simulating outcomes from the model, which are then used to obtain an estimate of the likelihood. Discussing the potential merits, they write:

> Simulation-based methodology for parameter estimation extends the range of models which can be fitted to a set of data. An immediate consequence is that a scientifically plausible

> implicit statistical model need not be ruled out on the grounds of mathematical intractability.
> (Diggle & Gratton, 1984, p. 210)

A likewise prescient paper by statistician Donald Rubin published in the same year (Rubin, 1984) includes a thought experiment describing a rejection-based algorithm which is part of the pre-history of a family of approaches referred to as Approximate Bayesian Computation (ABC).[33] ABC was developed in population genetics around 15 years later and has henceforth found many applications beyond it.[34] Ideas to make statistical inference possible for implicit models have been actively developed over the past decades. Chapter 2 will go into more detail about some classical techniques and their limitations. For example, rejection-based statistical methods suffer from similar issues as the brute-force approach—the so-called curse of dimensionality. This inherently limits their efficiency and scalability—and generally requires models to only have few free parameters and low-dimensional data. This explains why these approaches have not found widespread application in neuroscience.

1.5 ADVANCES IN SIMULATION-BASED INFERENCE THROUGH MACHINE LEARNING

So far, we coarsely sketched some progress and developments in neuroscience and statistics. A third and final area of progress is one that is beginning to increasingly change aspects of our daily lives: machine learning.[35] On a technical level, many of its recent advances are based on deep artificial neural networks, which have yielded impressive improvements on a huge variety of problems.[36] The starting point for this thesis was to ask how we can draw on advances in machine learning in order to develop efficient and scalable algorithms for statistical inference with implicit models. This was motivated by the dilemma we faced regarding the choice between statistically convenient and mechanistic models in neuroscience.

What is particularly exciting about this research question is that the challenges discussed in a neuroscience context thus far are by no means specific to it. *Simulation-based inference* is a recent encompassing term for algorithms performing statistical inference with implicit models.[37] One way to appreciate the breadth of its potential applications is to realize that implicit models are used to simulate processes from the smallest to the largest scales of what we humans have measured: At the extremes, computerized simulations are used from particle physics, studying events at subatomic scales, up to simulations of universe evolution.[38] Drug discovery, epidemiology, robotics, economics, and climate science all make use of complex simulations, to give a few examples beyond my own field of scientific expertise.[39] Advances in simulation-based inference can benefit many domains of science and industry at once—by offering a principled way to bridge the gap between observations and models.

Since I started working on this question in mid-2016, this research endeavour has gained a lot of traction and momentum. Retrospecting the progress made over the last couple of years, a recent review titled “The frontier of simulation-based inference” predicts:

> The rapidly advancing frontier means that several domains of science should expect either a significant improvement in inference quality or the transition from heuristic approaches to those grounded in statistical terms tied to the underlying mechanistic model. It is not unreasonable to expect that this transition may have a profound impact on science.
> Cranmer et al. (2020, p. 8)

This thesis contributes towards this transition: It is based on four publications, proposing new algorithms for simulation-based inference, applications to neuroscience, and a first unified benchmark. Overall, it demonstrates the potential of simulation-based inference to facilitate scientific discovery in neuroscience and beyond.

1.6 OUTLINE OF THIS THESIS

This introduction provides the overall context and motivation for the work presented in this thesis. Chapter 2 complements it with technical background: I briefly review 'likelihood-based' statistical inference, different reasons for intractable likelihoods, some classical simulation-based approaches, as well as recent neural network-based inference algorithms. In Chapter 3, each publication is briefly summarized and discussed. Finally, in Chapter 4, I provide a broader reflection and outlook. Comments and pointers to the literature are collected in endnotes appearing before the reference section. Publications are included in full in Appendix I, II, III, and IV. Appendix V delineates my individual author contributions.

CHAPTER 2

BACKGROUND

This chapter complements the introduction. It provides a concise summary of statistical inference and introduces relevant notation. Simulators and reasons for the common intractability of likelihoods are discussed, distinguishing practical, computational, and mathematical reasons. Finally, some classical approaches to simulation-based inference, their respective drawbacks, and how these are addressed by recent methods drawing on machine learning are reviewed.

2.1 STATISTICAL INFERENCE

The introduction motivated statistical inference and framed it as bridging two worlds, the real one, in which data is observed, and a theoretical one, which models are part of (Figure 1.2). Next, we will introduce relevant mathematical notation for both. The overall perspective on statistical inference in this section draws on Betancourt (2018, 2019).[40]

2.1.1 OBSERVATIONS

Recall from the introduction that we never observe a phenomenon directly, but only through probes used for measurement. The process of making observations is inherently variable since phenomenon, probe, and environment all interact. To deal with this stochasticity, we assume probability distributions over the space of all possible observations, $\mathcal{X}$. We refer to any particular probability distribution generating observations as a data generating process. The collection of all possible data generating processes is $\mathcal{P}_X$. When we study a given phenomenon, it is often helpful to think that a particular true (but unknown) data generating process p_o^* exists. We refer to a realization from p_o^* as an observation $\mathbf{x_o} \in \mathcal{X}$.

2.1.2 STATISTICAL MODELS

The true data generating process is unknown to us. We would like to identify data generating processes that are consistent with observation(s) realized from p_o^*. Searching through the entire space of all data generating processes $\mathcal{P}_X$ is infeasible. In practice, we thus restrict ourselves to a subspace $\mathcal{M} \subset \mathcal{P}_X$ (Figure 2.1). We equivalently refer to $\mathcal{M}$ as the model configuration space, the observational model, or the statistical model. We associate each data generating process (model configuration) in $\mathcal{M}$ with a vector of parameters $\boldsymbol{\theta} \in \boldsymbol{\Theta}$. Any data generating process in $\mathcal{M}_{\boldsymbol{\Theta}}$ is a conditional probability distribution $p(\mathbf{x}|\boldsymbol{\theta})$ that we consider as a possible stand-in for the true data generating process.[41]

In the context of this thesis, we are concerned with models that are generative, i.e., models for which we can generate samples $\mathbf{x} \sim p(\mathbf{x}|\boldsymbol{\theta})$. The dependency on $\boldsymbol{\theta}$ jointly describes phenomenological and non-phenomenological variables, e.g., biophysical parameters as well as ones related to the measurement process and environment. When we write $\mathbf{x}$, we refer to a sample generated by a model. When we write $\mathbf{x_o}$, we instead refer to observed data, e.g., data realized from p_o^* for which we would like to identify data-consistent model configurations.

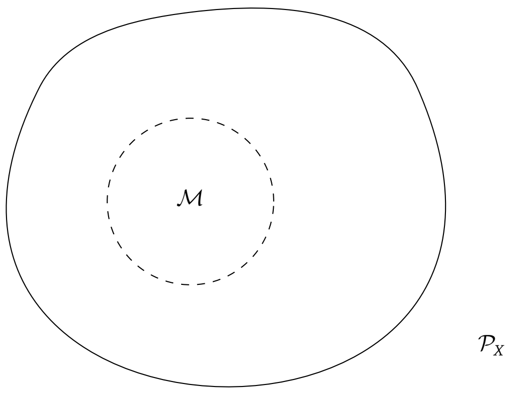


Figure 2.1: The model configuration space $\mathcal{M}$ contains a subset of all data generating processes $\mathcal{P}_X$ that are considered for inference. After Betancourt (2019, Figure 2).

$\mathcal{M}$ should be rich enough as to contain data generating processes that are sufficiently similar to the true data generating process for our purposes. Ideally, the true data generating would lie inside $\mathcal{M}$, but in practice, it usually won't be contained in it: We cannot capture the entire complexity of the natural world with our models of it. Consequently, when dealing with observations made in the real world, we should expect a mismatch relative to the true data generating process. We will return to discussing this important point in the last chapter.

2.1.3 FREQUENTIST AND BAYESIAN INFERENCE

Statistical inference allows us to answer the question which model configurations are consistent with observed data. As traditionally conceived, statistical inferences of all kinds rely on numerical evaluation of the likelihood function. The likelihood function returns a value that increases for consistent model configurations and decreases for inconsistent ones, i.e., the likelihood function expresses the *relative consistency* of each model configuration with the observation $\mathbf{x_o}$. Given parameters $\boldsymbol{\theta}$, the likelihood function evaluates the model $p(\mathbf{x}|\boldsymbol{\theta})$ at $\mathbf{x} = \mathbf{x_o}$. It is thus a function over the parameter space, specifically,

$$\begin{aligned}\mathcal{L}_{\mathbf{x_o}}(\boldsymbol{\theta}) : \boldsymbol{\Theta} &\rightarrow \mathbb{R}^+ \\ \boldsymbol{\theta} &\mapsto p(\mathbf{x} = \mathbf{x_o}|\boldsymbol{\theta}).\end{aligned}$$

Here, whenever we write likelihood, it is short for likelihood function, $\mathcal{L}_{\mathbf{x_o}}(\boldsymbol{\theta})$, as defined above. This is worth clarifying since the term likelihood (omitting function) is sometimes used more loosely to refer to the conditional density $p(\mathbf{x}|\boldsymbol{\theta})$ without fixing $\mathbf{x} = \mathbf{x_o}$.

There are two major paradigms in statistics—frequentist and Bayesian inference—both of which typically rely on numerical evaluation of the likelihood. We will first turn towards discussing the latter, which rests on Bayes' theorem.[42]

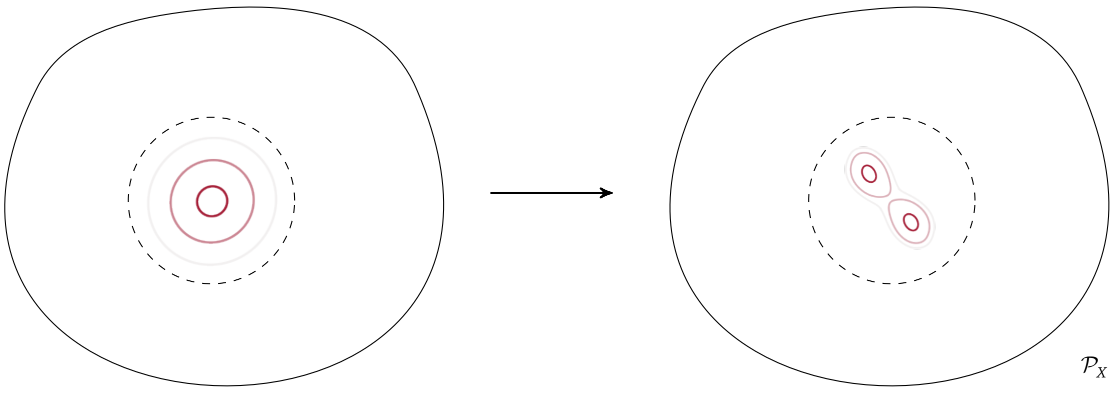


Figure 2.2: Through statistical inference, we identify which model configurations are consistent given observed data. In a Bayesian approach, which is illustrated in this figure, consistency is expressed in terms of probability distributions. The information contained in the observation updates the prior over model configurations (left), resulting in the posterior (right). After Betancourt (2019, Figure 4).

Applied to the random variables we introduced, the theorem states that:

$$\underbrace{p(\boldsymbol{\theta}|\mathbf{x}_o)}_{\text{posterior}} = \frac{\overbrace{p(\mathbf{x}_o|\boldsymbol{\theta})}^{\text{likelihood}}\,\overbrace{p(\boldsymbol{\theta})}^{\text{prior}}}{\underbrace{\int \mathrm{d}\boldsymbol{\theta}\, p(\mathbf{x}_o|\boldsymbol{\theta})\, p(\boldsymbol{\theta})}_{\text{marginal likelihood}}}.$$

We have labeled parts of the equations to refer to them. The posterior distribution, on the left-hand side, is the inferential outcome—it is the probability distribution over model configurations consistent with our observation. The right-hand side states that the posterior can be obtained through multiplication of likelihood and prior terms, divided by the marginal likelihood (also referred to as evidence).[43] Conceptually, we can think of Bayesian inference as a learning process, as illustrated in Figure 2.2: Our prior beliefs about plausible model configurations are expressed as $p(\boldsymbol{\theta})$.[44] Bayes' theorem tells us how to update our beliefs about plausible models in light of observations. Combining prior and likelihood, through which observations enter the equation, posterior beliefs are obtained.

Next, we will briefly discuss frequentist inference, which is the classical, i.e., most widely used approach for parameter estimation.[45] When faced with statistical models that do not allow analytic treatment, a frequentist approach would typically use the maximum likelihood (ML) point estimator,

$$\hat{\boldsymbol{\theta}}_{ML} = \arg\max_{\boldsymbol{\theta}} \log \mathcal{L}_{\mathbf{x}_o}(\boldsymbol{\theta}),$$

where the log of the likelihood is taken for numerical reasons.[46] Solving for $\hat{\boldsymbol{\theta}}_{ML}$ identifies a single model configuration that is consistent with the observed data, i.e., a 'best fit'. While frequentist approaches do allow expressing uncertainty in inferential outcomes, this perspective has a different notion of what uncertainty means.[47] Such differences have caused significant and long-running debates.[48] Arguably, the problem at hand and practical successes should pragmatically guide the choice of inference framework.[49] In this thesis, we focus on Bayesian approaches for reasons discussed in the introduction; to explore the space of all feasible solutions in the presence of degeneracy.

2.2 IMPLICIT MODELS: REASONS FOR INTRACTABLE LIKELIHOODS

Statistical inferences of all kinds usually rely on numerical evaluation of the likelihood function. However, in many cases of practical interest, the likelihood either cannot be evaluated numerically at all or doing so would be prohibitively expensive. Models for which the likelihood is intractable have been dubbed implicit models; as opposed to prescribed models for which evaluation of the likelihood is possible.[50]

As discussed in the introduction, implicit models are ubiquitously used to simulate processes in many domains of science and industry. In this thesis, we focus on models of the following general mathematical form, to which we also refer to as simulators (Figure 2.3): A simulator takes in parameters $\boldsymbol{\theta}$ defining the model configuration. Depending on $\boldsymbol{\theta}$ a series of latent variables $\mathbf{z}$ (internal states) are generated. This can involve a succession, i.e. $\mathbf{z}_1, \dots, \mathbf{z}_N$, are sampled from distributions that can depend on $\boldsymbol{\theta}$ as well as previously generated $\mathbf{z}_n$. When the simulator terminates, it returns outcomes $\mathbf{x} \sim p(\mathbf{x}|\boldsymbol{\theta}, \mathbf{z}_{1:N})$.[51] The likelihood of such a model given an observation $\mathbf{x}_\text{o}$ is:

$$\mathcal{L}_{\mathbf{x}_\text{o}}(\boldsymbol{\theta}) = \int \mathrm{d}\mathbf{z}_{1:N}\, p(\mathbf{x} = \mathbf{x}_\text{o}, \mathbf{z}_{1:N}|\boldsymbol{\theta}). \tag{1}$$

We also consider a second scenario, in which the simulator output $\mathbf{x}$ is a deterministic transformation of latent variables and parameters, i.e., $\mathbf{x} = f(\mathbf{z}_{1:N}, \boldsymbol{\theta})$. In this case, we can write the likelihood using a Dirac delta function as:

$$\mathcal{L}_{\mathbf{x}_\text{o}}(\boldsymbol{\theta}) = \int \mathrm{d}\mathbf{z}_{1:N}\, \delta(\mathbf{x}_\text{o} - f(\mathbf{z}_{1:N}, \boldsymbol{\theta}))p(\mathbf{z}_{1:N}|\boldsymbol{\theta}). \tag{2}$$

We can distinguish three reasons why above likelihoods may be intractable for a simulator of interest.[52] Jointly, all of these reasons motivate the need for methods that can perform inference without requiring numerical evaluation of the likelihood function, i.e., simulation-based inference.

PRACTICAL REASONS: INACCESSIBLE LATENT VARIABLES

Calculating above likelihoods requires that latent variables of the simulator can be accessed. From a practical standpoint, this may not the case, e.g., because simulators are implemented in codebases hard or impossible to interface with (such as legacy or commercial simulators), require specialized hardware, or are actual physical simulators rather than implemented in code.

COMPUTATIONAL REASONS: INFEASIBLE INTEGRATION DUE TO LATENTS

Considering the scenario where the latents $\mathbf{z}$ of the simulator can be accessed, we still face the need to integrate them out. In other words, given parameters $\boldsymbol{\theta}$ all possible execution traces would be required for exact evaluation of the likelihood. This commonly renders the likelihood intractable for computational reasons.[53] Note that, in principle, standard inference techniques over the joint posterior $p(\boldsymbol{\theta}, \mathbf{z}|\mathbf{x}_\text{o})$ could be performed, followed by marginalizing out $\mathbf{z}$. For example, one might use MCMC to obtain samples $\mathbf{x}, \mathbf{z} \sim p(\boldsymbol{\theta}, \mathbf{z}|\mathbf{x}_\text{o})$, discarding samples of $\mathbf{z}$. However, the joint space is often high-dimensional, making standard techniques likewise inapplicable.

MATHEMATICAL REASONS: DETERMINISTIC TRANSFORMATIONS

The likelihood in equation (2) poses a third source for intractability, which can occur even if latent variables are accessible and low-dimensional. For a minimal example, consider the case in which the outputs of the simulator are a deterministic transformation $\mathbf{x} = f(\mathbf{z})$, where $\mathbf{z} \sim p(\mathbf{z}|\boldsymbol{\theta})$, and $\mathbf{z}$ is low-dimensional. The nature of f, rather than the dimensionality of $\mathbf{z}$, may cause intractability: Calculating

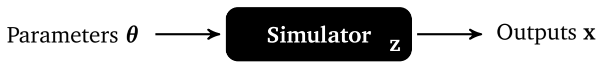


Figure 2.3: A simulator maps from parameters $\boldsymbol{\theta}$ to outputs $\mathbf{x}$. Outputs can depend both on $\boldsymbol{\theta}$ as well as a sequence of latent random variables $\mathbf{z}$. This renders the simulator stochastic: even for constant inputs, outputs vary.

the likelihood requires a change of variables from $\mathbf{z}$ to $\mathbf{x}$, the tractability of which depends on f. In other words, the likelihood is not tractable in general in this case.

In the Hodgkin-Huxley example provided in the introduction, we reduced the membrane potential down to a single number, counting the number of spikes. This f is a non-invertible many-to-one transformation for which the required change of variable cannot be performed; thus the likelihood is intractable. In addition, intrinsic noise is added at every time step, so that the latent space $\mathbf{z}$ is also high-dimensional in this example.

As discussed in Papamakarios (2019), one could again attempt to sample from the joint posterior, e.g., with MCMC methods for constrained manifolds (Graham & Storkey, 2017), however, the assumptions that such alternatives make do not hold in general. For example, the MCMC method by Graham & Storkey (2017) presupposes differentiability.

2.3 SIMULATION-BASED INFERENCE

Simulation-based inference (SBI) allows statistical inference in light of intractable likelihoods. Over the last few years, advances in machine learning have sparked rapid developments in the field of simulation-based inference which attempts to overcome limitations of classical algorithms. First, we discuss two classical approaches and their respective limitations, rejection-based Approximate Bayesian Computation and classical Synthetic Likelihood. Next, we will review a subset of new approaches, in particular ones using neural networks, e.g., for conditional density estimation or classification.

Note that this discussion is deliberately kept short and focused—it is meant to provide relevant context for understanding the publications in this thesis, which propose and benchmark algorithms for simulation-based inference. A more comprehensive review of SBI is provided by Cranmer et al. (2020), and details can be found in the respective publications.

2.3.1 REJECTION-BASED APPROXIMATE BAYESIAN COMPUTATION

Approximate Bayesian Computation (ABC) has its roots in population genetics, in the context of which the classical ABC rejection algorithm has been developed by Tavaré et al. (1997) and Pritchard et al. (1999), the latter extending it to continuous observations. The classical ABC algorithm is a special case of an accept-reject method, which proceeds as outlined in Algorithm 1: Parameters are proposed from the prior and accepted if simulation outcomes fall within a predefined distance to the observation, e.g., using the l_2-norm $\left\|\mathbf{x}-\mathbf{x_o}\right\|_2 \leq \epsilon$. The choice of ϵ trades off posterior accuracy and computational costs. The posterior is exact only in the limit $\epsilon \to 0$, however, the lower ϵ, the lower the acceptance rate, i.e., the larger the average amount of simulations needed per accepted sample.

Algorithm 1: Rejection ABC (REJ-ABC)

```
while in simulation budget do
    Sample $\boldsymbol{\theta}'$ from $p(\boldsymbol{\theta})$
    Simulate data $\mathbf{x}'$ from $p(\mathbf{x}|\boldsymbol{\theta}')$
    if $\left\|\mathbf{x}'-\mathbf{x}_o\right\| \leq \epsilon$ then
        Accept $\boldsymbol{\theta}'$
    else
        Reject $\boldsymbol{\theta}'$
    end
end
return Accepted samples $\{\boldsymbol{\theta}'\}$ from $\hat{p}(\boldsymbol{\theta} \mid \left\|\mathbf{x}'-\mathbf{x}_o\right\| \leq \epsilon)$
```

This is the simplest of all REJ-ABC schemes, which proposes all parameters from the prior, ignoring any information about which samples were rejected/accepted. There are many REJ-ABC schemes that make use of more sophisticated sampling schemes. For instance, by using a sequentially refined proposal distribution, illustrated in Figure 2.4. Such algorithms were, for example, proposed in Beaumont et al. (2002); Simola et al. (2021); Sisson et al. (2007); Toni et al. (2009)—in these publications, the proposal is constructed by perturbing previously accepted particles in an importance sampling scheme. These algorithms can improve acceptance rates but come with additional hyper-parameter choices that their performance can be sensitive to (number of rounds, ϵ schedule, perturbation kernel).

All REJ-ABC approaches suffer from the curse of dimensionality: With increasing dimensionality of data, $dim(\mathbf{x})$, obtaining a close match to $\mathbf{x_o}$ in all dimensions becomes increasingly unlikely. The posterior approximation gets worse in higher dimensions for fixed simulation budget (even when adjusting ϵ), and computational costs of running REJ-ABC are generally exponential in $dim(\mathbf{x})$.[54] The most common

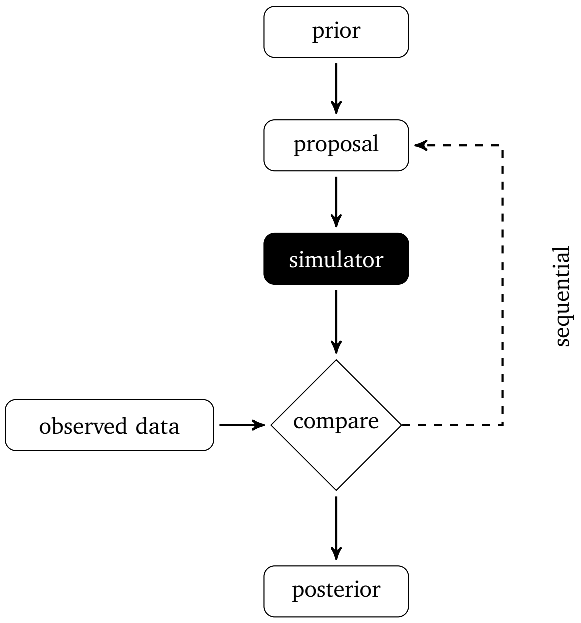


Figure 2.4: General scheme for rejection ABC algorithms, either proposing all parameters from the prior or using sequentially refined proposal distributions. After Cranmer et al. (2020, Figure 1).

way to address this problem is dimensionality reduction, i.e., to work with summary statistics (features) of the data. However, this usually involves a trade-off between dimensionality and information loss, and one infers the posterior given summary statistics, not the posterior given observations.[55]

2.3.2 CLASSICAL SYNTHETIC LIKELIHOOD

Another avenue is to estimate the likelihood function $\mathcal{L}_{\mathbf{x}_o}(\boldsymbol{\theta})$ from simulations, to then perform Bayesian or frequentist inference. An influential variant of such an approach was proposed by Wood (2010), often referred to as 'classical' synthetic likelihood (SL). It assumes that summary statistics of the data, conditional on parameters $\boldsymbol{\theta}$, follow a multivariate normal distribution, the parameters of which are estimated by M simulations, so that $\hat{\mathcal{L}}_{\mathbf{x}_o}(\boldsymbol{\theta}) = \mathcal{N}(\mathbf{x}_o|\hat{\boldsymbol{\mu}}_{\boldsymbol{\theta}}, \hat{\boldsymbol{\Sigma}}_{\boldsymbol{\theta}})$, where $\hat{\boldsymbol{\mu}}_{\boldsymbol{\theta}} = \frac{1}{M}\sum_{i=1}^{M}\mathbf{x}_i$ and $\hat{\boldsymbol{\Sigma}}_{\boldsymbol{\theta}} = \frac{1}{M-1}\sum_{i=1}^{M}(\mathbf{x}_i - \hat{\boldsymbol{\mu}}_{\boldsymbol{\theta}})(\mathbf{x}_i - \hat{\boldsymbol{\mu}}_{\boldsymbol{\theta}})^{\intercal}$.[56] The frequentist approach proposed by Wood (2010) proceeds by finding the optimimum of $\hat{\mathcal{L}}$ via MCMC. Price et al. (2018) extended SL to be used in a Bayesian framework, in which case the approximate posterior distribution is sampled by MCMC.[57] While SL has advantages over ABC methods, e.g., not relying on distance criteria, classical SL is highly simulation inefficient.[58]

2.3.3 NEURAL POSTERIOR ESTIMATION

High-fidelity simulations are often costly to run, can have many parameters, and potentially generate high-dimensional data. For such problems, classical SBI approaches struggle. Next, we turn towards alternative approaches drawing on machine learning to broaden the applicability of SBI.

One such avenue was opened up by the work of Papamakarios & Murray (2016), who proposed a novel approach for SBI which targets the posterior directly through neural network-based conditional density estimation.[59] The general idea of neural posterior estimation (NPE) approaches is as follows: A dataset is generated by proposing parameters from the prior and simulating, yielding a training set $\mathcal{D} = \{(\boldsymbol{\theta}_n, \mathbf{x}_n)\}_{n=1}^{N}$. This dataset is used to optimize the parameters $\boldsymbol{\phi}$ of a neural network-based conditional density estimator $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x})$, which allows to obtain an estimate of the posterior $\hat{p}(\boldsymbol{\theta}|\mathbf{x}_o) = q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x}_o)$.[60] This strategy was found to be more simulation efficient than the classical approaches discussed and has

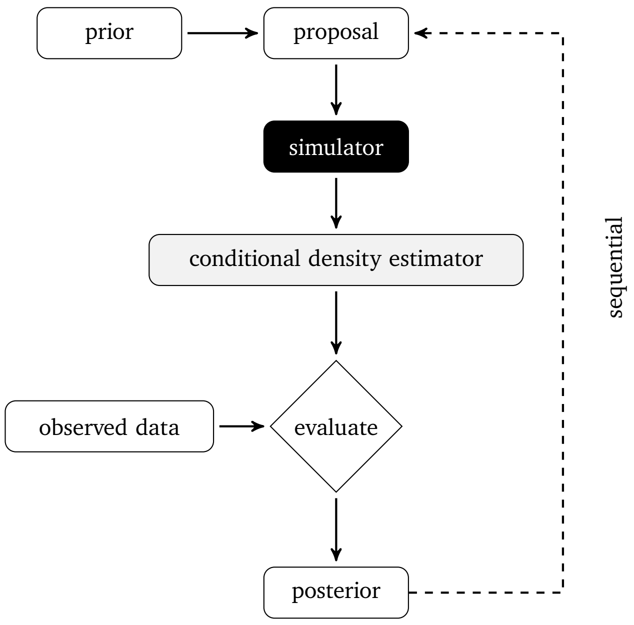


Figure 2.5: General scheme for posterior estimation algorithms, either proposing all parameters from the prior or using sequentially refined proposal distributions. After Cranmer et al. (2020, Figure 1).

the advantage of neither relying on user-defined distance criteria (REJ-ABC, SMC-ABC), nor requiring MCMC sampling (SL).[61]

Simulation efficiency can be further improved by using a sequential version of NPE (SNPE) if we are interested in performing inference for a given observation $\mathbf{x}_o$.[62] This involves R training datasets $\mathcal{D}^{(r)}$ containing parameters from proposals $\tilde{p}^{(r)}(\boldsymbol{\theta})$ and respective simulation outcomes, with $\tilde{p}^{(1)}(\boldsymbol{\theta}) = p(\boldsymbol{\theta})$. In other words, the initial parameters are proposed from the prior, and in following rounds, sequentially refined proposal distributions are used. As a proposal distribution, commonly the current posterior estimate is used, i.e., $\tilde{p}^{(r)}(\boldsymbol{\theta}) = \hat{p}^{(r)}(\boldsymbol{\theta}|\mathbf{x}_o)$. This heuristic scheme has been found to be able to further improve simulation efficiency.[63] Importantly, if we train a q_ϕ using a maximum likelihood loss and a proposal distribution not equal to the prior, the resulting approximate posterior converges to a distribution that is proportional to likelihood times proposal distribution. In other words, it converges to the *posterior had the proposal been the prior*, also known as the proposal posterior. In order to obtain the *posterior under the prior*, different strategies for SNPE have been proposed by Papamakarios & Murray (2016, SNPE-A), Lueckmann et al. (2017, SNPE-B), and most recently Greenberg et al. (2019, SNPE-C or APT).

Note that the choice between NPE and SNPE is not just one about simulation efficiency: If we are interested in performing fast inference for many different observations, we might prefer a single round algorithm (NPE). After a potentially large one-off cost of generating lots of training data and learning q_ϕ, inference can be performed for different $\mathbf{x}_o$ at the speed of milliseconds (a single forward pass through the neural network)—this is also known as amortized inference.[64] The general approach of targeting the posterior, either in a single round or sequentially, is illustrated in Figure 2.5. In the next chapter we will discuss (S)NPE in more detail, and consider extensions for and applications to mechanistic models in neuroscience.

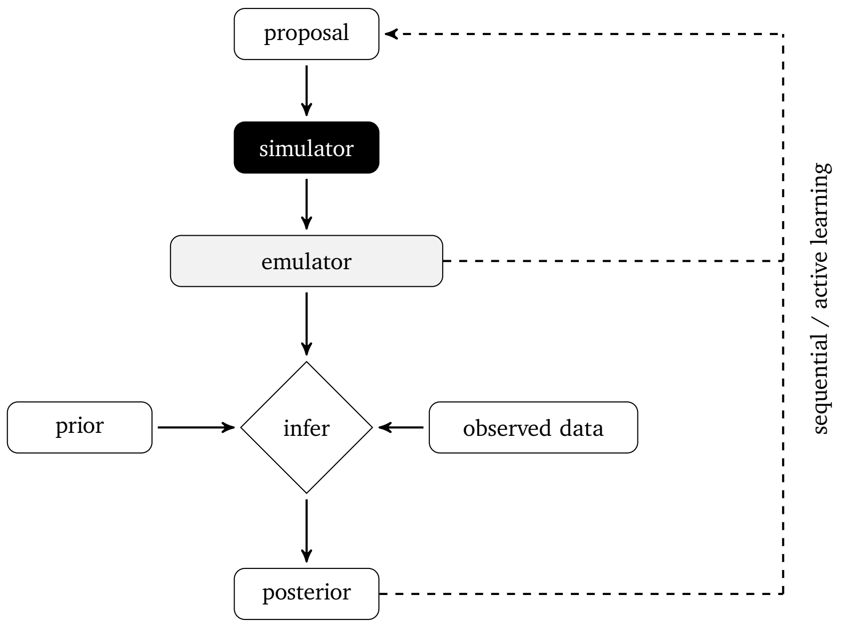


Figure 2.6: Likelihood estimation approaches that proceed by learning a surrogate or emulator, which is then used as an estimate of the likelihood function. They can be used in single round, in a sequential manner akin to SNPE, or with active learning schemes. Active learning can be used in different ways: To target uncertainty in emulator estimates (global emulators, e.g., for amortized inference) or in posterior estimates (local emulators, for efficient inference given specific observations). After Cranmer et al. (2020, Figure 1).

2.3.4 NEURAL LIKELIHOOD ESTIMATION

Neural likelihood estimation (NLE) algorithms learn a synthetic likelihood using neural networks. The general strategy is as follows: A dataset consisting of parameters—proposed from any proposal distribution, for example, the prior—and associated simulation outcomes is used as the training set $\mathcal{D} = \{(\boldsymbol{\theta}_n, \mathbf{x}_n)\}_{n=1}^N$, similar to posterior estimation approaches. It is used to optimize the parameters $\boldsymbol{\phi}$ of a neural network-based conditional density estimator $q_{\boldsymbol{\phi}}(\mathbf{x}|\boldsymbol{\theta})$. Note that the conditioning is opposite relative to SNPE, a surrogate or emulator of the simulator is learned. An estimate of the likelihood can be obtained by evaluating at the observation $\mathbf{x}_o$, $\hat{\mathcal{L}}_{\mathbf{x}_o}(\boldsymbol{\theta}) = q_{\boldsymbol{\phi}}(\mathbf{x} = \mathbf{x}_o|\boldsymbol{\theta})$. With this synthetic likelihood, samples from the approximate posterior $\hat{p}(\boldsymbol{\theta}|\mathbf{x}_o) \propto \hat{\mathcal{L}}_{\mathbf{x}_o}(\boldsymbol{\theta})p(\boldsymbol{\theta})$ can then for instance be obtained by MCMC.

In order to improve simulation efficiency, this scheme can likewise be used with a training dataset that is adaptively built. Unlike sequential neural posterior estimation algorithms, which require taking the proposal distribution into account, this is no concern for NLE. In Chapter 3, we will discuss two concurrently published strategies that use NLE in such a way: Papamakarios (2019) proposed a sequential algorithm in which the current posterior estimate is used in a multi-round algorithm (i.e., similar to how sequential acquisitions are done for SNPE). In Lueckmann et al. (2019), we used the flexibility in proposals for actively learning local emulators (for specific $\mathbf{x}_o$) and global emulators (allowing amortization) for inference. Both strategies are illustrated in Figure 2.6 and we discuss them in more detail in the next chapter.

2.3.5 NEURAL RATIO ESTIMATION

As a final category of approaches, we will briefly discuss recent ratio estimation approaches for simulation-based inference. The algorithm proposed by Hermans et al. (2020) falls into this category: It uses classifiers to estimate ratios of likelihoods which can subsequently be used within an MCMC sampler.[65] In particular, they propose to train a neural network-based classifier $\hat{r}$ to distinguish between samples $(\boldsymbol{\theta}, \mathbf{x}) \sim p(\mathbf{x}|\boldsymbol{\theta})p(\boldsymbol{\theta})$ and ones from the marginal model $(\boldsymbol{\theta}, \mathbf{x}) \sim p(\boldsymbol{\theta})p(\mathbf{x})$. An optimal classifier r approximates the ratio

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

so that the posterior can be recovered as $\hat{p}(\boldsymbol{\theta}|\mathbf{x}_o) = \hat{r}(\boldsymbol{\theta}, \mathbf{x}_o)p(\boldsymbol{\theta})$. Such neural ratio estimation (NRE) approaches can likewise be employed sequentially, utilizing proposal distributions for simulation efficiency, see Durkan et al. (2020); Hermans et al. (2020).[66] Note that in contrast to the two previous approaches that we discussed, no conditional density estimator but a classifier is trained. In the next chapter, we provide quantitative comparisons of classical approaches as well as neural posterior, likelihood, and ratio estimation strategies.[67]

CHAPTER 3

PUBLICATIONS

This thesis develops and applies new techniques for simulation-based inference. As detailed in the introduction, it was motivated by wanting to perform statistical inference for mechanistic models in neuroscience—however, the resulting techniques are generally applicable to any implicit model. In the following, each of the four publications that are part of this thesis is briefly summarized and discussed. Full publications and my individual author contributions are in the Appendix.

3.1 FLEXIBLE STATISTICAL INFERENCE FOR MECHANISTIC MODELS OF NEURAL DYNAMICS

Our paper “Flexible statistical inference for mechanistic models of neural dynamics” was presented at the 30th Conference on Neural Information Processing Systems (NeurIPS) and published in the conference proceedings (Lueckmann et al., 2017). Its starting point was the work by Papamakarios & Murray (2016), who proposed a neural network-based algorithm that casts simulation-based inference into a problem of posterior density estimation, overcoming limitations of classical approaches (for more background, see Section 2.3.3). We found their algorithm to yield highly promising initial results (Gonçalves et al., 2017) but discovered critical drawbacks in their sequential estimation strategy, which we addressed in this publication. Simultaneously, we proposed several extensions for neuroscience problems to avoid unstable regimes of the simulator and to deal with missing data and high-dimensional time series. We demonstrated the potential of neural posterior estimation to automatically learn features of data rather than relying on hand-crafted summaries through problem-specific inductive biases.

Specifically, the algorithm we proposed overcomes two limitations of the one by Papamakarios & Murray (2016). Their sequential algorithm relies on an analytical correction step to account for using proposal distributions, i.e., the posterior estimate is obtained as $\hat{p}(\boldsymbol{\theta}|\mathbf{x}_o) \propto p(\boldsymbol{\theta})/\tilde{p}(\boldsymbol{\theta})q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x}_o)$. However, this has two drawbacks: Firstly, using an analytical correction poses constraints on the types of conditional density estimators and proposals that can be used. Secondly, the analytical correction step can not necessarily be carried out even if choices are made to make it tractable. In particular, assuming a uniform prior $p(\boldsymbol{\theta}) \propto 1$, a Gaussian proposal $\tilde{p}(\boldsymbol{\theta})$, and Mixture of Gaussians $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x})$, the division requires that the covariance of the proposal is greater than each of the component covariances of $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x})$, i.e., the proposal needs to be wider. Empirically, we found that the post-hoc analytical correction step can fail in practice, leading the algorithm to terminate early, approximating the posterior with a single Gaussian from the previous round. In order to address both of these issues we introduced an importance weighting factor $p(\boldsymbol{\theta})/\tilde{p}(\boldsymbol{\theta})$ into the loss function, so that no analytical correction is necessary. This removes constraints on which combinations of proposals, conditional density estimators, and priors can be used and addresses the issue of early termination.

We contributed a number of extensions motivated by problems we encountered when applying the algorithm to mechanistic models in neuroscience: 1) We introduced a scheme that can avoid parameter regimes that yield bad simulation outcomes, e.g., the simulator being numerically unstable and thus returning meaningless results. Our scheme uses a neural network classifier that can predict bad simulation outcomes and thus avoids expensive, unnecessary simulations. 2) We proposed a scheme for dealing with missing features that automatically imputes values as part of the neural network architecture. For example, a summary feature like spike latency cannot be computed if there are no spikes at all in a time series. 3) We demonstrated that this approach can be used to automatically learn suitable summary statistics. For this, we made use of an embedding network prepended to the density estimator. In particular, we used an RNN architecture with GRU-cells to reduce the dimensionality of time series data. We applied our algorithms to synthetic data and in-vitro recordings of membrane voltages and compared our method against classical rejection-based algorithms (SMC-ABC), the approach by Papamakarios & Murray (2016), as well as a popular genetic algorithm (IBEA, as implemented in Van Geit et al., 2016).

In retrospect, our paper contributed to the lineage of approaches addressing SBI through neural density estimation of the posterior and provided inspiration and basis for much future work, including the second publication of this thesis. We originally named the algorithm in Lueckmann et al. (2017) SNPE for sequential neural posterior estimation. Greenberg et al. (2019) subsequently improved upon our importance weighting-based loss which can have high variance for some problems, elegantly reparameterizing it to obtain the posterior via maximization of the estimated proposal posterior. Following this publication, we decided to jointly refer to sequential neural network-based approaches for posterior estimation as SNPE approaches (Figure 2.5) and refer to Papamakarios & Murray (2016) as SNPE-A, Lueckmann et al. (2017) as SNPE-B, and Greenberg et al. (2019) as SNPE-C respectively, when a distinction is necessary (Lueckmann et al., 2021; Papamakarios, 2019). The code for SNPE-B was published as the first release of the inference toolbox `delfi` (github.com/mackelab/delfi), which provided the basis for future research projects, e.g., Gonçalves et al. (2020); Greenberg et al. (2019). `delfi` was the predecessor to the `sbi` toolbox (github.com/mackelab/sbi) we introduced in Tejero-Cantero et al. (2020).

3.2 TRAINING DEEP NEURAL DENSITY ESTIMATORS TO IDENTIFY MECHANISTIC MODELS OF NEURAL DYNAMICS

Our paper “Training deep neural density estimators to identify mechanistic models of neural dynamics” was published in eLife (Gonçalves et al., 2020). An earlier version of this work was made available as a preprint (Gonçalves et al., 2019). Simulations of interpretable mechanistic models, relating causes to effects, are a core ingredient in the toolkit of computational neuroscience. The status quo in the field is to use ad-hoc strategies instead of statistical inference when fitting these models to data, e.g., tuning parameters by hand, by exhaustive or grid search, or genetic algorithms based on heuristics score functions (see Chapter 1). Simulation-based inference, called ‘likelihood-free’ inference in the paper, offers a principled alternative: Using neural conditional density estimation, we can identify the entire space of data-consistent models. Following our work on SNPE (Lueckmann et al., 2017), this publication demonstrated the usefulness of (sequential) neural posterior estimation for neuroscience on a diverse set of mechanistic models, including receptive fields, ion channels, single neuron dynamics, and the crustacean stomatogastric ganglion (STG). This project, which was a large collaborative effort, revealed the flexibility and potential of using (S)NPE in neuroscience and overall helped narrow the gap between mechanistic and statistical approaches.

The subproject I focused on was about ion channel models: Ion channels are the basic building blocks of many mechanistic models. A huge diversity of different channels types has been described, which has been analysed and categorized by Podlaski et al. (2017) in the ion channel genealogy (ICG) project. In Gonçalves et al. (2020), we trained a neural posterior estimator to identify consistent models for many different observations available in the ICG database. We trained a neural posterior estimator on a subset of 350 channel types, with a million simulations, to infer parameters of a flexible ion channel model. Simulations took around 1000 CPU-hours and training the network around 150 CPU-hours. Once simulation and training finished, the network was able to infer consistent models at the speed of milliseconds, returning a full posterior in less than 10 ms. This makes it possible to rapidly perform inference for many different observations once the upfront training cost is paid (*amortized inference*).

Amortized inference with neural posterior estimation is simple yet highly effective. While our application in Gonçalves et al. (2020) was intended as a proof-of-principle, it could be extended to the entire database and made available through an online interface. Similar use-cases are widely found across sciences—in fact, amortized inference with neural posterior estimation has, e.g., recently been applied in gravitational physics (Dax et al., 2021).

3.3 LIKELIHOOD-FREE INFERENCE WITH EMULATOR NETWORKS

Our paper "Likelihood-free inference with emulator networks" was presented at the first Symposium on Advances in Approximate Bayesian Inference (AABI), a continuation of the NeurIPS 2015–17 workshop series of the same name, and published in its proceedings (Lueckmann et al., 2019). An earlier version of this work was made available as a preprint (Lueckmann et al., 2018). In it, we proposed a novel, simulation-efficient algorithm for simulation-based inference. Our algorithm is based on learning an emulator of the simulator: The emulator can then be used as an evaluable and differentiable stand-in for the intractable likelihood during inference, or as a fast surrogate model.

In the paper, we demonstrated how our approach of learning emulators allows for highly simulation-efficient algorithms. We used two different acquisition rules for new simulations that target maximum variance of the posterior estimate or epistemic uncertainty in the emulator parameters. Depending on the rule, either a local emulator, which can be used for inference given a fixed observation or a global emulator for amortized inference is learned. We demonstrated this approach on synthetic examples and a biophysical neuron model. We found it to allow for efficient inference even on high-dimensional problems and to compare favorably against BOLFI (Gutmann & Corander, 2016).

In a concurrent publication, Papamakarios et al. (2019) adopted a similar strategy, likewise learning a conditional density $q_{\phi}(\mathbf{x}|\boldsymbol{\theta})$ to perform simulation-based inference. In contrast to our approach, they focused on having a flexible emulator, using normalizing flows for q, but acquired new simulations using the current posterior estimate, as, e.g., done for SNPE. This strategy is not guaranteed to be optimal but rather a heuristic choice. In addition, it limits inference to a particular observation. On the other hand, we focused our efforts on acquisition rules which are highly simulation-efficient and also considered a strategy that can be used for amortized inference. Since our proposed schemes are computationally expensive, they should be traded-off against the runtime cost of the simulator—whether the use of active learning is warranted will strongly depend on the problem at hand (see also Durkan et al., 2018).

3.4 BENCHMARKING SIMULATION-BASED INFERENCE

Our paper "Benchmarking Simulation-Based Inference" was presented at The 24th International Conference on Artificial Intelligence and Statistics (AISTATS) and published in the conference proceedings (Lueckmann et al., 2021). Advances in machine learning have sparked rapid and exciting developments in the field of simulation-based inference, as discussed in the first two chapters of this thesis. However, publications introducing new algorithms often compare against a limited subset of other algorithms, use different tasks, and report different performance metrics. This has made it hard to thoroughly judge the strengths and weaknesses of new algorithms.

In the paper, we introduced a unified benchmark for quantitative, transparent comparison between SBI algorithms on the same set of tasks and performance metrics. More specifically, our initial selection of algorithms contained classical approaches (REJ/SMC-ABC and variants using linear regression adjustments and semi-automatic summary statistics, classical synthetic likelihood), as well as more recent approaches based on neural networks (neural posterior, likelihood, and ratio estimation approaches in sequential and non-sequential variants). Our results showed that the choice of performance metric is critical, that sequential approaches have higher sample efficiency, and that there is substantial room for improvement even for state-of-the-art algorithms. Neural network-based approaches generally performed better than classical ones, but there was no uniformly best algorithm across all tasks. Full results are available through an interactive website at sbi-benchmark.github.io.

Benchmarking is crucial for communicating the state-of-the-art and limitations of algorithms, both within machine learning and more broadly. It can also contribute to more efficient and reproducible science, as it allows researchers to directly use reference results and algorithms rather than having to implement and compute them anew. However, benchmarking in itself has limitations and can lead to unwanted outcomes. For example, a too narrow focus on scoring good performance on a small set of tasks and quantitative metrics can bias and skew research. With that in mind, we designed the benchmark to be extensible with respect to algorithms, tasks, and metrics: we invite researchers to collaborate and help ensure that strengths and weaknesses of existing and future approaches to SBI are appropriately evaluated and reported. Our open and extensible framework to facilitate this process is available at github.com/sbi-benchmark/sbibm.

CHAPTER 4

CONCLUSION

Simulations, as traditionally conceived, are used in a *forward direction*—we let them run and see what happens.[68] This narrow use was criticized decades ago:

> Having gone to the trouble of encoding the requisite knowledge for building a simulation, one should attempt to derive the maximum benefit from this knowledge. That is, in addition to "running" a simulation to answer what-if questions, one should be able to utilise the full range of inferencing, reasoning, and search methods that are available in AI. [...] This natural, though long-overdue, extension of simulation can be referred to as **beyond "what-if."** (Rothenberg, 1989, p. 11)

Despite realizing that such extended usage of simulations would be highly useful—and calling this long-overdue back in 1989—widely applicable *inferencing methods* for detailed simulation models have only been proposed very recently, driven in part by advances in machine learning, statistics, and high-performance computing: Simulation-based inference enables principled reasoning in the *backward* or *inverse direction*, by virtue of enabling statistical inference for implicit models. This allows us to infer causes from effects, to reason about grounds on the basis of outcomes (Figure 4.1).[69]

Overall, this thesis demonstrates the potential of SBI to greatly improve the quality of insight we can gain from simulations. This way, SBI might become an essential part of the scientists' toolkit to facilitate scientific discovery. What is still needed to fully realize this potential?

Our benchmark empirically shows that impressive progress has been made in the field. However, it also reveals that scalability towards higher-dimensional problems and the sample efficiency remain important areas for further improvements. This is also echoed in the recent review by Cranmer et al. (2020), in which they identify three main directions for recent and future progress: 1) Advances in machine learning, e.g., density estimation in high dimensions and the incorporation of problem-specific inductive biases, 2) active learning, to guide where simulations are acquired and thus improve sampling efficiency, and 3) approaches turning black-box inference problems into gray-box ones, i.e., using additional information about the simulator.

In addition to these central directions for future progress, I want to highlight that we do not only need scalable and efficient methods but ones that are reliable, robust, and user-friendly. Inference should be seen as only one step in a workflow, and it should always be followed by critique and refinement.[70] In the problem setting of simulation-based inference, only a subset of established procedures to critique inferential outcomes is applicable, and developing new diagnostics for SBI is an important avenue for future work.[71] In virtually all practical settings, the true data generating process will not be among the models we consider.[72] How sensitive different SBI algorithms are to model misspecification and how to thoroughly detect it, is a challenging but important broader issue.[73] From a practical perspective, it will be essential to provide guidance, which algorithm to use when, as well as user-friendly software that includes informative and easily-interpretable diagnostics.

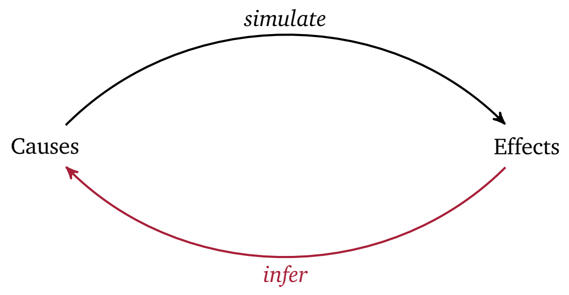


Figure 4.1: Simulations, as classically conceived, are used to reason in the *forward direction*. When we simulate, for instance, using a computer program, we observe effects given underlying causes. Simulation-based inference enables principled reasoning in the *backward* or *inverse direction*: Given outcomes, it allows to infer underlying grounds to solve problems.

SBI is already increasingly applied in neuroscience. For example, recent uses include applications to mechanistic models of synaptic release (Schröder et al., 2019), stimulus optimization for retinal neuroprosthetics (Oesterle et al., 2020), pyloric network models to study metabolic efficiency (Deistler et al., 2021), jumping evidence accumulation models for decision-making (Wieschen et al., 2020), neural mass models of cortical columns (Rodrigues et al., 2021), and cytoarchitecture measurements with diffusion MRI (Jallais et al., 2021). Beyond neuroscience, SBI finds applications in diverse scientific fields. To name a few recent examples out of many, this includes using SBI to study particle collisions (Brehmer & Cranmer, 2020), bacterial colonization (Järvenpää et al., 2019), biochemical reaction networks (Mikelson & Khammash, 2020), epidemics and pandemics (de Witt et al., 2020; Wood et al., 2020), volcanic eruptions (Pacchiardi et al., 2021), flash flooding (Nourali, 2021), paleodemographic patterns (DiNapoli et al., 2021), transfer learning in robotics (Marlier et al., 2021; Muratore et al., 2021), cosmological parameters with weak lensing (Jeffrey et al., 2021), gravitational waves (Dax et al., 2021; Delaunoy et al., 2020), warm dark matter (Hermans et al., 2021), and X-ray binaries (Huppenkothen & Bachetti, 2021). I hope that we will see progress in simulation-based inference that is driven hand in hand by problems in a wide range of natural, engineering, and social sciences. I am looking forward with excitement to the scientific discoveries that this will enable.

NOTES

1. Their findings were published in Hodgkin & Huxley (1939). The use of a system of mirrors invented by Huxley allowed them to simultaneously view the $\sim 500\,\mu\text{m}$ wide nerve fiber (axon) from both front and side through a microscope, which was essential to avoid damaging the axon (recounted in Schwiening, 2012). What was particularly remarkable about the shape of action potentials, and counter to existing theories at the time, was the observed 'overshoot' of the action potential, i.e., the crossing of the zero potential.

2. In addition to numerous neuroscience textbooks, a reader interested in the theory of action potentials is referred to Brette (2016): This chapter includes a brief history of 'excitability theories', discussing findings and ideas of the many predecessors to Hodgkin and Huxley that I omitted for brevity (e.g., Galvani, Matteucci, Du Bois-Reymond, Helmholtz, Nernst and Bernstein). It considers the question 'Why do cells spike?' from a biophysical, functional, and epistemological standpoint.

3. The estimate of 86 billion (86×10^9) neurons is provided in Azevedo et al. (2009), downward correcting earlier estimates of 100 billion neurons. The length of 10 million kilometers of fibers in the human brain is taken from Murre & Sturdy (1995). Equally impressive is the number of synaptic connections, which is sometimes stated to be 100 trillion (100×10^{12}). However, this number (which assumes 100 billion neurons with an average of 1 000 connections each) is not typically found in peer-reviewed literature, see Rose (2018) for discussion. Detailed estimates of densities of neurons, synapses and axons for the mouse cortex can be found in Braitenberg & Schüz (1991).

4. An earlier equivalent circuit model of the membrane was proposed in 1907 by Louis Lapicques (Abbott, 1999; Brunel & Van Rossum, 2007; Lapicque, 1907). Compared to Lapicques' model, Hodgkin and Huxley included two channels respectively selective for sodium and potassium ions. In addition, each channel's conductance was chosen to depend on the potential difference across the membrane. A broad perspective on the role of mathematical models in neuroscience, which includes both these equivalent circuit models and historical context, can be found in Lindsay (2021).

5. These events are recounted in Schwiening (2012). Additional details can, for example, be found in personal recollections by Hodgkin (1976), or Huxley's Nobel Lecture, in which he said:

   > The computations so far described were done by hand. This was a laborious business: a membrane action took a matter of days to compute, and a propagated action potential took a matter of weeks. But it was often quite exciting. [...] Very often my expectations turned out to be wrong, and an important lesson I learnt from these manual computations was the complete inadequacy of one's intuition in trying to deal with a system of this degree of complexity. (Huxley, 1972, pp. 61–64)

6. The mathematical model by Hodgkin & Huxley (1952a,b,c,d,e) provided the first quantitative account of action potentials. Their work paved the way for many future discoveries, and the underlying

equations are still used as standard building blocks in simulation software for biophysical models in neuroscience today. Together with John Eccles, they shared the 1963 Nobel Prize for Physiology or Medicine "for their discoveries concerning the ionic mechanisms involved in excitation and inhibition in the peripheral and central portions of the nerve cell membrane" (Nobel Prize Outreach AB, 2021). A brief historical account can be found in Schwiening (2012). The legacy of their work is, for example, discussed in Catterall et al. (2012); Meunier & Segev (2002) from the perspective of neuroscience, and in Craver (2008); Levy (2014) from the perspective of philosophy of science. The latter perspective provides a discussion centered around the nature of mechanistic explanation.

7. In-depth discussions about the role of simulation in neuroscience can, for example, be found in Dudai & Evers (2014); Einevoll et al. (2019); Gerstner et al. (2012). A review of common software and databases used for simulations is provided in Blundell et al. (2018).

8. At the time Hodgkin and Huxley published their model, little was known about ion channels. That sodium and potassium selective channels exist on a molecular level was only shown in the decades to follow, through the work Bertil Hille, Clay Armstrong, Roderick MacKinnon (2003 Nobel Laureate) and others (for a retrospective, see Hille et al., 1999). The patch recording technique by Erwin Neher and Bert Sakmann (Neher & Sakmann, 1976, 1991 Nobel Laureates) allowed measurements of currents through single ion channels and revealed their extraordinary diversity. Today, large-scale databases of ion channel models exist, containing over a thousand unique models (Podlaski et al., 2017). Kandel et al. (2021) can serve as a starting point to read more about the topics that are cited as having made significant advances.

9. In a recent commentary, Kording et al. (2020) characterize computational neuroscience, as the sub-field of neuroscience that is united by combining mathematical reasoning with computer simulations, whilst highlighting the diversity of underlying goals. In particular, they conducted a survey asking scientists to specify their modeling goals. Apart from rating their goals in terms of biological realism (microscopic, macroscopic, behavioral, representational), participants were asked about desired scientific impact (useful, normative, clinically relevant, inspiring experiments) and preferred style of models (compact, tractable, interpretable, beauty). Interpretability was rated as the most important goal on average, but overall the survey shows just how variable underlying modeling goals in computational neuroscience are. I agree with the authors that this diversity should be embraced and seen as a strength.

10. For perspectives, see, for example, Gomez-Marin et al. (2014); Paninski & Cunningham (2018); Urai et al. (2021).

11. This perspective on statistical inference is laid out in *Statistical Inference: The Big Picture* (Kass, 2011). Figure 1.2 is inspired by Kass' big picture, but simplifies it for ease of presentation. One notable difference is that Kass draws a distinction between scientific and statistical models (on which he comments in the end of article). For ease of presentation, I avoid this distinction in the introduction and overall explain concepts such as 'model' through examples rather than by definitions. Acknowledging that defining the term might be useful, the following characterizations can serve as a working definition: Broadly speaking, a model represents selected parts or aspects of a target system (Koperski, 2021) and is used as "a stand-in, an imposter, an artificial construct designed to respond in the same way as the system you would like to understand" (Meent et al., 2018, p. 10). A concise definition can also be found in Rothenberg (1989), who defines modeling as "the cost-effective use of something in place of something else for some cognitive purpose" (p. 1).

12. See, e.g., Box (1976), who discusses the iteration between theory and practice and the need for unhampered feedback.

13. This concise definition of probability is taken from Blitzstein & Hwang (2019, p. 1), who write: "Mathematics is the logic of certainty; probability is the logic of uncertainty."
14. See Betancourt (2018) for more details.
15. Synaptic release probability widely varies between different neuron types (Branco & Staras, 2009). The statement that this probability is around .5 for most synapses is made in Rusakov et al. (2020). Regarding spontaneous release, De Schutter writes:

    > While one needs specific recording methods to study the stochasticity of ionic channels, any recording method will show that synapses in the central nervous system are noisy and stochastic. Their probability of transmission can be as low as 10%, and spontaneous release (i.e., not caused by a spike in the presynaptic neuron) is so frequent that it is often used to study synaptic properties. (De Schutter, 2001, p. 387)

16. One could distinguish between different sources of variation, e.g., variation can be inherent to the phenomenon and be irreducible, which is referred to as ontological or aleatoric uncertainty, or rather be attributable to imprecise measurements and in principle considered reducible, which is referred to as epistemic uncertainty (see Hüllermeier & Waegeman, 2021, for a review). However, here, these distinctions are not going to be primary—no matter what the exact nature of variation is, it will always be present in observations.
17. According to Betancourt (2019), on which the discussion of statistical inference in Chapter 2 rests.
18. The underlying reasons for intractable likelihood functions will be detailed in Chapter 2, once mathematical notation has been introduced.
19. The model used for this example is the same as in Pospischil et al. (2008). In addition to the original model by Hodgkin and Huxley, this formulation also includes a slow voltage-dependent potassium current for spike-frequency adaptation, making this a suitable model for hippocampal and cortical pyramidal cells. Out of 12 total parameters, two were varied. Note that the model is stochastic due to a time-varying noisy input current. For each run of the model, a period of 120 ms was simulated. The resulting time series is reduced to a single feature: The number of spikes that occurred during this time period. See Lueckmann et al. (2019, Appendix F.1) for more details, where we used the same setup.
20. For the brute-force approach, a 2D grid with 1 000 linearly spaced points between upper and lower bounds for each of the two parameters was used, resulting in a million (1000^2) parameter combinations. The lower and upper bounds were 0.5 and 60 (mS/cm^2) for $\bar{g}_{Na}$, and 2.5 and 7.5 (mS/cm^2) for $\bar{g}_K$, respectively. These ranges are roughly in line with the ones used in Pospischil et al. (2008, Table 1). As compared to Lueckmann et al. (2019), the bounds for $\bar{g}_K$ were chosen slightly smaller in order to increase grid resolution. For each parameter combination, 100 simulations (runs) were performed. On the basis of those 100 runs, the probability of obtaining zero, one, two, three, or four spikes was approximated for each parameter combination. No run produced more than four spikes. This resulted in 100 million total runs of the model.
21. Each of the 100 million ($1000^2 \times 100 = 10^8$) runs took about 50 ms on an Intel® Xeon® Gold 6148 CPU (2,40 GHz). The calculation was carried out in parallel on 20 of such cores, thus taking about $10^8 \times 50\,\text{ms} \times 1/20 = 10^8 \times 2.5\,\text{ms} = 69.\bar{4}\,\text{hours} \approx 0.00792\,\text{years}$ to complete. For each additional free parameter at the same resolution, above equation is multiplied by a factor of 1 000.
22. To illustrate this point, consider the class of biophysical single neuron models: For this example, we use the model considered in Pospischil et al. (2008), which has 12 parameters (we only kept two of

them free to make the brute-force strategy viable). This model ignores all spatial structure. Software like NEURON (Hines & Carnevale, 1997) or GENESIS (Bower & Beeman, 2012) allows taking the intricate morphology of neurons into account by dividing them into hundreds of compartments, all of which have their own dynamics. In addition, the classical Hodgkin-Huxley model does not model the synapse. See Gerstner et al. (2014) for an introduction to models of dendrites and synapses. Detailed models taking these aspects into account can thus easily have hundreds of parameters. Since it is very challenging to constrain models with this many parameters by data, the number of free parameters is usually kept significantly lower by making simplifying assumptions. For example, Taylor et al. (2009) consider a multi-compartmental model of lateral pyloric (LP) neurons with 17 free parameters. Hay et al. (2011) and Eyal et al. (2018) use pyramidal cell (PC) models with 22 and 29 free parameters, respectively.

23. Here, we use 'automated search method' to broadly refer to black-box optimization algorithms, i.e., algorithms that do not require differentiability of the simulator. In neuroscience, special purpose search algorithms, e.g., based on genetic algorithms, and software toolboxes for biophysical models have been developed, see Druckmann et al. (2007); Van Geit et al. (2016), for example.

24. Depending on the exact algorithm that is used, the objective may be referred to under a different name, including error, fitness, loss, discrepancy, or distance function.

25. The objective to be minimized could, for example, be specified as the average absolute deviation between simulated and observed spike counts (over multiple repeats per model configuration).

26. For example, we might run an evolutionary/genetic search algorithm, such as covariance matrix adaptation evolution strategy (Hansen, 2006, CMA-ES). The solution obtained in this example will depend strongly on where an algorithm is initialized (as well as on internal randomness of the procedure). Once the objective no longer improves during the exploration of parameter values, the search terminates. A 'best' fit returned by the algorithm corresponds to a point in the solution subspace depicted in Figure 1.3. To some degree, the non-uniqueness of solutions can be studied by running a search procedure repeatedly (potentially in parallel), or by studying the sensitivity of solutions to perturbations. As we will discuss in Chapter 2, statistical inference provides a principled framework to this issue.

27. The highly reliable triphasic motor pattern of the crustacean stomatogastric ganglion is the so-called *pyloric rhythm*. The sequential firing pattern involves lateral pyloric (LP), pyloric (PY), and pyloric dilator (PD) neurons. A seminal study by Prinz et al. (2004) constrained three-cell circuit models by pyloric-rhythm recordings of 99 lobsters (*Homarus americanus*), revealing that many different model configurations can yield similar network activity. Follow-up experimental studies, which considered the same identified neurons across many animals, confirmed that there is significant animal-to-animal variability in individual neuronal and synaptic properties (Hamood & Marder, 2014). Together with many studies on the single neuron and circuit level, this raises fundamental open questions concerning robust neuromodulation in all nervous systems (Marder et al., 2014).

28. Edelman & Gally (2001) discuss the importance of *degeneracy* for biological systems and highlight its centrality for evolution. See Mason (2010) for a broader discussion of the history and definition of the term.

29. See Golowasch et al. (2002) for an example demonstrating that a single conductance-based neuron model constructed by averaging population parameters can fail to accurately characterise a system. Marder & Taylor (2011) propose to study 'population of models' instead. They acknowledge the drawback of grid or random sampling strategies (which get exponentially more difficult

with increasing numbers of parameters) and see promise in techniques that "generate populations of models that conform to a probability distribution that mimics chosen aspects of the biological distribution" (p. 137).

30. Throughout this thesis, we mostly adopt a particular statistical paradigm that is known as Bayesian inference, which is well-suited to infer the entire space of consistent models (the so-called *posterior distribution*). We will discuss differences to frequentist inference, an alternative paradigm, in Chapter 2. This introduction avoids the distinction between the two, emphasizing their commonalities—they are both based on probabilities and usually require computation of likelihood functions—over their differences. This is inspired by the perspective of Kass (2011).

31. For example, the simulation models in Diggle & Gratton (1984) were mostly run on a IBM 370/168 mainframe: Introduced in 1973, it had a clock rate of about 12.5 Mhz and cost 2.8 million US$ (not adjusted for inflation), see McCallum (2021). For years to come, the complexity of simulations and inference procedures was yet significantly limited by the availability of fast and cheap digital computers. Today, even a 4 US$ microcontroller can be ten times faster (and considerably smaller) than the IBM 370/168 mainframe (Adams, 2021).

32. Diggle & Gratton (1984) write "although the idea of using simulation to analyse intractable models is obviously not new, we are aware of no other systematic investigations along the lines of the present paper" (p. 195). Earlier references from the 1970s are included in the passage following the quote.

33. Rubin (1984) includes a thought experiment that describes a likelihood-free rejection sampling algorithm. However, Rubin does not promote using it in a situation where the likelihood is intractable, and his version presupposes discrete sampling spaces, see Marin et al. (2012).

34. Pritchard et al. (1999) developed the first proper ABC algorithm in the context of population genetics which is applicable to continuous spaces. For the history of ABC approaches, see Tavaré (2018) as well.

35. The term *machine learning* was coined by Samuel (1959), who used it in the context of his research on the game of checkers. In it, he predicted: "Programming computers to learn from experience should eventually eliminate the need for much of this detailed programming effort" (p. 535). A concise definition of machine learning can, for example, be found in Mitchell (1997, p. 2). Recent progress in artificial intelligence (AI) is largely based on machine learning algorithms (Winter et al., 2021). For perspectives on the impact of AI, see, e.g., Bughin et al. (2018); Makridakis (2017); Mehrabi et al. (2021); Mohamed et al. (2020); Tomašev et al. (2020).

36. For an overview of deep learning, see LeCun et al. (2015); Schmidhuber (2015).

37. It is only recently that the term simulation-based inference (SBI) is used more widely. In particular, Cranmer et al. (2020) make a case for adopting simulation-based inference as an encompassing name as opposed to using the older and more common one likelihood-free inference (LFI). As reviewed in Chapter 2, many SBI algorithms involve likelihood or likelihood ratio estimation, which is why LFI could be considered "a bit of a misnomer" (Cranmer et al., 2020, p. 1). Approximate Bayesian Computation (ABC) is the name under which some of the first statistical inference algorithms that have been developed for implicit models are known. However, many SBI techniques may be used either in a Bayesian or frequentist paradigm, especially if they involve likelihood or likelihood ratio estimation. Overall, simulation-based inference is meant to be more inclusive, which explains its usage in the title of this thesis as well as in Lueckmann et al. (2021). More generally, readers should be aware of all three acronyms, SBI, LFI, and ABC, since usage varies between authors and may change over time.

38. See Cranmer et al. (2020), or its preprint which includes a figure for illustration (arXiv:1911.01429v3).
39. Regarding the examples cited, see, e.g., Duan et al. (2015); Durrant & McCammon (2011); Edwards (2011); Gourieroux et al. (1993); Mattingly et al. (2012).
40. Note that this perspective on statistical inference is not universal. For pointers to other perspectives the reader is referred to Betancourt (2018, 2019). Relative to these writings, the perspective here is much abridged, and a reader is referred to them for more details. The notation is slightly different—I adopted it to be consistent with the publications that this thesis is based on.
41. Note that the parameterization of model configurations is not unique. In other words, there are generally many equivalent ways of labeling the models in $\mathcal{M}$.
42. A specific case of Bayes' theorem was presented in Bayes (1763). Its general form follows from two fundamental rules of probability, the sum and product rules.
43. Note that the marginal likelihood is a single number, so that posterior is proportional to likelihood times prior. This is a fact that many approximate inference procedures exploit since the integral over all possible parameter configurations is difficult to compute in general. Common approximate Bayesian inference methods include variational inference and MCMC, see, e.g., Andrieu et al. (2003); Blei et al. (2017).
44. The choice of prior distribution is an important point in Bayesian inference, see, e.g., Gelman et al. (2017); Wagenmakers (2007) for discussion. For practical tips on choosing a prior, see, e.g., Stan Developers (2021).
45. Murphy (2007, p.1) writes, for example: "The frequentist approach to statistics is the most widely used, and hence is sometimes called the orthodox approach or classical approach."
46. In practice, finding $\hat{\boldsymbol{\theta}}_{ML}$ usually involves optimization. Since the log function is monotone it does not change the maximum.
47. From a Bayesian perspective, uncertainty is expressed in terms of beliefs given *the current data* in terms of the posterior distribution. In turn, a frequentist perspective on uncertainty asks "how much would my estimate change if I had different data?" (Murphy, 2007, p.8). Frequentist statements about uncertainty are formalized in terms of sampling distributions which hold true in the limit of many observations. An interested reader can find discussions of the frequentists' reliance on asymptotics in Betancourt (2018).
48. For more background on the Bayesian-Frequentist controversy, Gelman (2008); Senn (2011); Vallverdú (2015); Wagenmakers et al. (2008) and references therein can serve as starting points.
49. Rather than focusing on a Bayesian-Frequentist dichotomy, this introduction means to primarily emphasize commonalities to all statistical inferences, such as the centrality of the likelihood function. A pragmatic view, as opposed to one that bases the choice on philosophical grounds, is, e.g., advocated by Kass (2011). Similarly, Gelman (2012) writes: "... it would be rash to let philosophical foundations be a justification for using Bayesian methods."
50. The term *implicit model* was coined by Diggle & Gratton (1984). See Mohamed & Lakshminarayanan (2017) for a review of the role of implicit models in machine learning.
51. This description renders the simulator an instance of a probabilistic program, in that it directly represents a data generating process (Goodman, 2013; van de Meent et al., 2018). Note that we do not assume any particular language; the simulator could even be physical.
52. A more in-depth discussion of these three reasons can be found in Papamakarios (2019).

53. See, for example, Cranmer et al. (2020).

54. Regarding these consequences of the curse of dimensionality for REJ-ABC, see Barber et al. (2015); Blum & François (2010); Prangle et al. (2018)

55. Only in the rare case where sufficient statistics of the data are known, the dimensionality can be reduced without incurring loss of information. The design of summary statistics is a large topic within the literature and new techniques continue to be developed, see, e.g., Prangle (2018).

56. Note that SL is classically performed on summary statistics of the data, i.e., $\mathbf{x_o}$ and $\mathbf{x}_i$ have been reduced to features.

57. In this case, the target distribution for MCMC is the synthetic likelihood multiplied by the prior.

58. At every step of the MCMC sampler, M new simulations are performed. In addition, burn-in time and thinning of chains may be required.

59. Historically, the approach by Papamakarios & Murray (2016) is related to regression adjustment approaches for ABC (Beaumont et al., 2002; Blum, 2018), in which a mapping from $\mathbf{x}$ to $\boldsymbol{\theta}$, e.g., Blum & François (2010) used a single-layer neural network to predict mean and variance of $\boldsymbol{\theta}$. However, instead of considering density estimation as a post-hoc correction step after an REJ-ABC algorithm, the algorithm by Papamakarios & Murray (2016) does not rely on rejection sampling and trade-offs associated with the choice of ϵ.

60. Papamakarios & Murray (2016) used mixture density network (MDN) architecture (Bishop, 1994). However, posterior estimation as described here can in principle be performed with any kind of conditional density estimator. Today, conditional normalizing flows are a powerful alternative to MDNs, see Papamakarios et al. (2021) for a review.

61. See Papamakarios & Murray (2016) and Lueckmann et al. (2021) for comparisons.

62. I originally named the algorithm we proposed in Lueckmann et al. (2017) sequential neural posterior estimation (SNPE). Following the publication of Greenberg et al. (2019), we started to instead use SNPE as an acronym for the class of neural network-based approaches that follow the sequential posterior estimation strategy outlined in Figure 2.5. When needed, we refer to the version of Papamakarios & Murray (2016) as SNPE-A, the version of Lueckmann et al. (2017) as SNPE-B, and the version of Greenberg et al. (2019) as SNPE-C (or APT, for Automatic Posterior Transformation), see also Papamakarios (2019). I introduced the acronym NPE for the single round version for Lueckmann et al. (2021).

63. Regarding comparisons of NPE versus SNPE, see for example Greenberg et al. (2019); Lueckmann et al. (2021, 2017); Papamakarios & Murray (2016).

64. Examples of practical use cases for amortized inference through NPE can for example be found in Dax et al. (2021); Gonçalves et al. (2020).

65. Note that ratio estimation algorithms for SBI were proposed prior to Hermans et al. (2020). Earlier variants include Cranmer et al. (2015); Dutta et al. (2016); Izbicki et al. (2014); Pham et al. (2014). Connections to recent methods are e.g. discussed in Durkan et al. (2020); Thomas et al. (2021).

66. Hermans et al. (2020) call the single round version of their algorithm amortized approximate likelihood ratio MCMC (AALR-MCMC). They also introduce a sequential ratio estimation algorithm. Durkan et al. (2020) shows a close relation between the algorithms of Hermans et al. (2020) and Greenberg et al. (2019), for which they slightly change the formulation of the loss function of Hermans et al. (2020) from binary to multi-class. In Lueckmann et al. (2021), we use the acronyms NRE

or SNRE for (sequential) neural ratio estimation to refer to the approach introduced by Hermans et al. (2020) with a multi-class loss (Durkan et al., 2020) in single round or sequential version.

67. Not all differences between the algorithms are captured well in terms of quantitative comparisons. Therefore, we also provide some problem-oriented practical advice in Lueckmann et al. (2021).

68. This statement is based on the following characterization of simulation:

> Simulation is a form of modeling whose purpose is usually comprehension, planning, prediction, and manipulation. It can be defined broadly as a behavioral or phenomenological approach to modeling; that is, a simulation is an active, behavioral analog of its referent. The essence of simulation is that it unfolds over time. It models sequences and (possibly) timing of events in the real world. Simulation is a *process* in which a model of *any* kind is used to imitate (some aspects of) the behavior of its referent. [...] Simulation is generally used to answer what-if questions. [...] As traditionally conceived, simulation works only in this “forward” direction: The user “winds it up” and lets it run to see what happens. (Rothenberg, 1989, p. 8)

All applications in this thesis are concerned with computer simulations, which provide flexible and effective means for simulation. Note, however, that simulations can also be physical analogs of their referent. SBI is general enough to be applied in such cases as well.

69. The importance of backward reasoning is, for example, discussed by Bunge (2019), who writes:

> ... the hardest problems are likely to be inverse, in that they go from effect to cause, from goal to means, or from conclusion to premises—as in diagnosing disease from symptoms, looking for a westward passage from Europe to Asia, and designing public policies to face social issues. In other words, forward thinking proceeds from premises to conclusions, whereas backward thinking goes the other way around, that is, in search for grounds[.] (Bunge, 2019, p. 483)

70. Statistician George Box advocates a view in which criticism is the step that should naturally follow all inferential procedures—as an integral part to any statistical workflow: We should carefully validate inferential outcomes and question whether our assumptions hold true. Ideally, the workflow forms an open loop in which we continue to refine the model class $\mathcal{M}$ or collect additional observations, as shown in the following diagram (after Blei, 2014; Box, 1976):

$$\begin{tikzcd}
& \text{Observations} \arrow[d] & \\
\text{Models} \arrow[r] & \text{Inference} \arrow[r] & \text{Criticism} \arrow[ll, bend left]
\end{tikzcd}$$

71. There are many different strategies for checking inferential outcomes, e.g., posterior predictive checks (Box, 1980). For a general overview of strategies and techniques for Bayesian model checking and refinement, see, for example, the workflow suggested by Gelman et al. (2020) and references therein. In the simulation-based inference setting, procedures relying on likelihood evaluations

cannot be used. Moreover, checks such as simulation-based calibration (SBC, Talts et al., 2018) can be computationally prohibitive, unless the inference algorithm allows for fast inference for many different observations.

72. In virtually all practical settings, the true data generating process p_o^* will not be contained in $\mathcal{M}$ since our models, as representations of the world and theoretical entities, will fail to capture the full complexity of natural phenomena. In other words, some mismatch between the models we identify through statistical inference and p_o^* is to be expected.

73. Model mismatch is a crucial caveat of inference in practice: If the model for which inference is performed does not accurately capture the empirical data of interest, wrong conclusions may be reached, potentially leading to adverse outcomes. This can have broad implications since the ability to constrain numerical simulators by data, as well as to capture and report uncertainty, is critical in scientific disciplines—some of which have a clear societal impact. Mitigating risks of algorithms that are used in decision-making poses profound challenges. Addressing these challenges will not only require the development of new methods but also ethical foresight and diverse perspectives (Mohamed et al., 2020; Tomašev et al., 2020).

REFERENCES

Abbott, L. F. (1999). Lapicque's introduction of the integrate-and-fire model neuron (1907). *Brain Research Bulletin*, 50(5-6), 303–304. https://doi.org/10.1016/s0361-9230(99)00161-6

Adams, J. (2021). Meet Raspberry Silicon: Raspberry Pi Pico now on sale at $4. https://www.raspberrypi.org/blog/raspberry-pi-silicon-pico-now-on-sale/

Andrieu, C., De Freitas, N., Doucet, A., & Jordan, M. I. (2003). An introduction to MCMC for machine learning. *Machine Learning*, 50(1), 5–43. https://doi.org/10.1023/A:1020281327116

Azevedo, F. A., et al. (2009). Equal numbers of neuronal and nonneuronal cells make the human brain an isometrically scaled-up primate brain. *Journal of Comparative Neurology*, 513(5), 532–541. https://doi.org/10.1002/cne.21974

Barber, S., Voss, J., & Webster, M. (2015). The rate of convergence for approximate Bayesian computation. *Electronic Journal of Statistics*, 9(1), 80–105. https://doi.org/10.1214/15-EJS988

Bayes, T. (1763). LII. An essay towards solving a problem in the doctrine of chances. By the late Rev. Mr. Bayes, F. R. S. communicated by Mr. Price, in a letter to John Canton, A. M. F. R. S. *Philosophical Transactions of the Royal Society of London*, 53, 370–418. https://doi.org/10.1098/rstl.1763.0053

Beaumont, M. A., Zhang, W., & Balding, D. J. (2002). Approximate Bayesian computation in population genetics. *Genetics*, 162(4), 2025–2035. https://doi.org/10.1093/genetics/162.4.2025

Betancourt, M. (2018). Calibrating model-based inferences and decisions. *arXiv*. https://arxiv.org/abs/1803.08393

Betancourt, M. (2019). Probabilistic modeling and statistical inference. https://betanalpha.github.io/assets/case_studies/modeling_and_inference.html

Bishop, C. M. (1994). Mixture density networks. http://publications.aston.ac.uk/id/eprint/373/1/NCRG_94_004.pdf

Blei, D. M. (2014). Build, compute, critique, repeat: Data analysis with latent variable models. *Annual Review of Statistics and Its Application*, 1, 203–232.

Blei, D. M., Kucukelbir, A., & McAuliffe, J. D. (2017). Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518), 859–877. https://doi.org/10.1080/01621459.2017.1285773

Blitzstein, J. K. & Hwang, J. (2019). *Introduction to probability*. CRC Press.

Blum, M. G. (2018). Regression approaches for ABC. In *Handbook of Approximate Bayesian Computation* (pp. 71–85). Chapman and Hall/CRC.

Blum, M. G. & François, O. (2010). Non-linear regression models for approximate Bayesian computation. *Statistics and Computing*, 20(1), 63–73. https://doi.org/10.1007/s11222-009-9116-0

Blundell, I., Brette, R., Cleland, T. A., Close, T. G., Coca, D., Davison, A. P., ..., & Eppler, J. M. (2018). Code generation in computational neuroscience: A review of tools and techniques. *Frontiers in Neuroinformatics*, 12, 68. https://doi.org/10.3389/fninf.2018.00068

Bower, J. M. & Beeman, D. (2012). *The book of GENESIS: Exploring realistic neural models with the GEneral NEural SImulation System*. Springer Science & Business Media.

Box, G. E. (1976). Science and statistics. *Journal of the American Statistical Association*, 71(356), 791–799. https://doi.org/10.1080/01621459.1976.10480949

Box, G. E. (1980). Sampling and Bayes' inference in scientific modelling and robustness. *Journal of the Royal Statistical Society: Series A (General)*, 143(4), 383–404.

Braitenberg, V. & Schüz, A. (1991). *Anatomy of the cortex: Statistics and geometry*. Springer-Verlag Berlin Heidelberg. https://doi.org/10.1007/978-3-662-02728-8

Branco, T. & Staras, K. (2009). The probability of neurotransmitter release: Variability and feedback control at single synapses. *Nature Reviews Neuroscience*, 10(5), 373–383. https://doi.org/10.1038/nrn2634

Brehmer, J. & Cranmer, K. (2020). Simulation-based inference methods for particle physics. *arXiv*. https://arxiv.org/abs/2010.06439

Brette, R. (2016). Action potentials. http://romainbrette.fr/WordPress3/wp-content/uploads/2016/06/theory-of-action-potentials-chapter-1.pdf

Brunel, N. & Van Rossum, M. C. (2007). Quantitative investigations of electrical nerve excitation treated as polarization. *Biological Cybernetics*, 97(5), 341–349. https://doi.org/10.1007/s00422-007-0189-6

Bughin, J., Seong, J., Manyika, J., Chui, M., & Joshi, R. (2018). Notes from the AI frontier: Modeling the impact of AI on the world economy. *McKinsey Global Institute*. https://www.mckinsey.com/featured-insights/artificial-intelligence/notes-from-the-ai-frontier-modeling-the-impact-of-ai-on-the-world-economy

Bunge, M. (2019). Inverse problems. *Foundations of Science*, 24(3), 483–525. https://doi.org/10.1007/s10699-018-09577-1

Catterall, W. A., Raman, I. M., Robinson, H. P. C., Sejnowski, T. J., & Paulsen, O. (2012). The Hodgkin-Huxley heritage: From channels to circuits. *Journal of Neuroscience*, 32(41), 14064–14073. https://doi.org/10.1523/JNEUROSCI.3403-12.2012

Cranmer, K., Brehmer, J., & Louppe, G. (2020). The frontier of simulation-based inference. *Proceedings of the National Academy of Sciences*, 117(48), 30055–30062. https://doi.org/10.1073/pnas.1912789117

Cranmer, K., Pavez, J., & Louppe, G. (2015). Approximating likelihood ratios with calibrated discriminative classifiers. *arXiv*. https://arxiv.org/abs/1506.02169

Craver, C. F. (2008). Physical law and mechanistic explanation in the Hodgkin and Huxley model of the action potential. *Philosophy of Science*, 75(5), 1022–1033. https://doi.org/10.1086/594543

Dax, M., Green, S. R., Gair, J., Macke, J. H., Buonanno, A., & Schölkopf, B. (2021). Real-time gravitational-wave science with neural posterior estimation. *arXiv*. https://arxiv.org/abs/2106.12594

De Schutter, E. (2001). Computational neuroscience: More math is needed to understand the human brain. In *Mathematics unlimited—2001 and beyond* (pp. 381–391). Springer. https://doi.org/10.1007/978-3-642-56478-9_17

de Witt, C. S., Gram-Hansen, B., Nardelli, N., Gambardella, A., Zinkov, R., Dokania, P., ..., & Baydin, A. G. (2020). Simulation-Based Inference for global health decisions. *arXiv*. Presented at ICML 2020 Workshop on Machine Learning for Global Health. https://arxiv.org/abs/2005.07062

Deistler, M., Macke, J. H., & Gonçalves, P. J. (2021). Disparate energy consumption despite similar network activity. *BioRxiv*. https://doi.org/10.1101/2021.07.30.454484

Delaunoy, A., Wehenkel, A., Hinderer, T., Nissanke, S., Weniger, C., Williamson, A. R., & Louppe, G. (2020). Lightning-fast gravitational wave parameter inference through neural amortization. *arXiv*. https://arxiv.org/abs/2010.12931

Diggle, P. J. & Gratton, R. J. (1984). Monte Carlo methods of inference for implicit statistical models. *Journal of the Royal Statistical Society: Series B (Methodological)*, 46(2), 193–212. https://doi.org/10.1111/j.2517-6161.1984.tb01290.x

DiNapoli, R. J., Crema, E. R., Lipo, C. P., Rieth, T. M., & Hunt, T. L. (2021). Approximate Bayesian computation of radiocarbon and paleoenvironmental record shows population resilience on Rapa Nui (Easter Island). *Nature Communications*, 12(1), 1–10. https://doi.org/10.1038/s41467-021-24252-z

Druckmann, S., Banitt, Y., Gidon, A. A., Schürmann, F., Markram, H., & Segev, I. (2007). A novel multiple objective optimization framework for constraining conductance-based neuron models by experimental data. *Frontiers in Neuroscience*, 1, 1. https://doi.org/10.3389/neuro.01.1.1.001.2007

Duan, W., Fan, Z., Zhang, P., Guo, G., & Qiu, X. (2015). Mathematical and computational approaches to epidemic modeling: A comprehensive review. *Frontiers of Computer Science*, 9(5), 806–826. https://doi.org/10.1007/s11704-014-3369-2

Dudai, Y. & Evers, K. (2014). To simulate or not to simulate: What are the questions? *Neuron*, 84(2), 254–261. https://doi.org/10.1016/j.neuron.2014.09.031

Durkan, C., Murray, I., & Papamakarios, G. (2020). On contrastive learning for likelihood-free inference. In *Proceedings of the 36th International Conference on Machine Learning*, volume 98 of *Proceedings of Machine Learning Research*. PMLR. http://proceedings.mlr.press/v119/durkan20a.html

Durkan, C., Papamakarios, G., & Murray, I. (2018). Sequential neural methods for likelihood-free inference. *arXiv*. https://arxiv.org/abs/1811.08723

Durrant, J. D. & McCammon, J. A. (2011). Molecular dynamics simulations and drug discovery. *BMC Biology*, 9(1), 1–9. https://doi.org/10.1186/1741-7007-9-71

Dutta, R., Corander, J., Kaski, S., & Gutmann, M. U. (2016). Likelihood-free inference by ratio estimation. *arXiv*. https://arxiv.org/abs/1611.10242

Edelman, G. M. & Gally, J. A. (2001). Degeneracy and complexity in biological systems. *Proceedings of the National Academy of Sciences*, 98(24), 13763–13768. https://doi.org/10.1073/pnas.231499798

Edwards, P. N. (2011). History of climate modeling. *Wiley Interdisciplinary Reviews: Climate Change*, 2(1), 128–139. https://doi.org/10.1002/wcc.95

Einevoll, G. T., et al. (2019). The scientific case for brain simulations. *Neuron*, 102(4), 735–744. https://doi.org/10.1016/j.neuron.2019.03.027

Eyal, G., Verhoog, M. B., Testa-Silva, G., Deitcher, Y., Benavides-Piccione, R., DeFelipe, J., ..., & Segev, I. (2018). Human cortical pyramidal neurons: From spines to spikes via models. *Frontiers in Cellular Neuroscience*, 12, 181. https://doi.org/10.3389/fncel.2018.00181

Gelman, A. (2008). Objections to Bayesian statistics. *Bayesian Analysis*, 3(3), 445–449. https://doi.org/10.1214/08-BA318

Gelman, A. (2012). Philosophy of Bayesian statistics: My reactions to Senn. https://statmodeling.stat.columbia.edu/2012/02/03/philosophy-of-bayesian-statistics-my-reactions-to-senn/

Gelman, A., Simpson, D., & Betancourt, M. (2017). The prior can often only be understood in the context of the likelihood. *Entropy*, 19(10), 555. https://doi.org/10.7916/D8V99MJX

Gelman, A., Vehtari, A., Simpson, D., Margossian, C. C., Carpenter, B., Yao, Y., ..., & Modrák, M. (2020). Bayesian workflow. *arXiv*. https://arxiv.org/abs/2011.01808

Gerstner, W., Kistler, W. M., Naud, R., & Paninski, L. (2014). *Neuronal dynamics: From single neurons to networks and models of cognition*. Cambridge University Press.

Gerstner, W., Sprekeler, H., & Deco, G. (2012). Theory and simulation in neuroscience. *Science*, 338(6103), 60–65. https://doi.org/10.1126/science.1227356

Golowasch, J., Goldman, M. S., Abbott, L., & Marder, E. (2002). Failure of averaging in the construction of a conductance-based neuron model. *Journal of Neurophysiology*, 87(2), 1129–1131. https://doi.org/10.1152/jn.00412.2001

Gomez-Marin, A., Paton, J. J., Kampff, A. R., Costa, R. M., & Mainen, Z. F. (2014). Big behavioral data: Psychology, ethology and the foundations of neuroscience. *Nature Neuroscience*, 17(11), 1455–1462. https://doi.org/10.1038/nn.3812

Gonçalves, P. J., Lueckmann, J.-M., Bassetto, G., Nonnenmacher, M., & Macke, J. H. (2017). Flexible Bayesian inference for mechanistic models of neural dynamics. In *Computational and Systems Neuroscience Meeting (COSYNE), Salt Lake City, USA.*

Gonçalves, P. J., Lueckmann, J.-M., Deistler, M., Nonnenmacher, M., Öcal, K., Bassetto, G., ..., & Macke, J. H. (2019). Training deep neural density estimators to identify mechanistic models of neural dynamics. *bioRxiv*. https://doi.org/10.1101/838383

Gonçalves, P. J., Lueckmann, J.-M., Deistler, M., Nonnenmacher, M., Öcal, K., Bassetto, G., ..., & Macke, J. H. (2020). Training deep neural density estimators to identify mechanistic models of neural dynamics. *Elife*, 9, e56261. https://doi.org/10.7554/eLife.56261

Goodman, N. D. (2013). The principles and practice of probabilistic programming. *ACM SIGPLAN Notices*, 48(1), 399–402. https://doi.org/10.1145/2480359.2429117

Gourieroux, C., Monfort, A., & Renault, E. (1993). Indirect inference. *Journal of Applied Econometrics*, 8(S1), S85–S118. https://doi.org/10.1002/jae.3950080507

Graham, M. M. & Storkey, A. J. (2017). Asymptotically exact inference in differentiable generative models. *Electronic Journal of Statistics*, 11(2), 5105–5164. https://doi.org/10.1214/17-EJS1340SI

Greenberg, D., Nonnenmacher, M., & Macke, J. (2019). Automatic posterior transformation for likelihood-free inference. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research* (pp. 2404–2414). PMLR. http://proceedings.mlr.press/v97/greenberg19a.html

Gutmann, M. U. & Corander, J. (2016). Bayesian optimization for likelihood-free inference of simulator-based statistical models. *The Journal of Machine Learning Research*, 17(1), 4256–4302. https://jmlr.csail.mit.edu/papers/v17/15-017.html

Hamood, A. W. & Marder, E. (2014). Animal-to-animal variability in neuromodulation and circuit function. In *Cold Spring Harbor Symposia on Quantitative Biology*, volume 79 (pp. 21–28). https://doi.org/10.1101/sqb.2014.79.024828

Hansen, N. (2006). The CMA evolution strategy: A comparing review. In *Towards a New Evolutionary Computation: Advances in the Estimation of Distribution Algorithms* (pp. 75–102). Springer-Verlag Berlin-Heidelberg. https://doi.org/10.1007/3-540-32494-1_4

Hay, E., Hill, S., Schürmann, F., Markram, H., & Segev, I. (2011). Models of neocortical layer 5b pyramidal cells capturing a wide range of dendritic and perisomatic active properties. *PLoS Computational Biology*, 7(7), e1002107. https://doi.org/10.1371/journal.pcbi.1002107

Hermans, J., Banik, N., Weniger, C., Bertone, G., & Louppe, G. (2021). Towards constraining warm dark matter with stellar streams through neural simulation-based inference. *Monthly Notices of the Royal Astronomical Society*, 507(2), 1999–2011. https://doi.org/10.1093/mnras/stab2181

Hermans, J., Begy, V., & Louppe, G. (2020). Likelihood-free MCMC with approximate likelihood ratios. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research* (pp. 4239–4248). PMLR. http://proceedings.mlr.press/v119/hermans20a

Hille, B., Armstrong, C. M., & MacKinnon, R. (1999). Ion channels: From idea to reality. *Nature Medicine*, 5(10), 1105–1109. https://doi.org/10.1038/13415

Hines, M. L. & Carnevale, N. T. (1997). The NEURON simulation environment. *Neural Computation*, 9(6), 1179–1209. https://doi.org/10.1162/neco.1997.9.6.1179

Hodgkin, A. L. (1976). Chance and design in electrophysiology: An informal account of certain experiments on nerve carried out between 1934 and 1952. *The Journal of Physiology*, 263(1), 1. https://doi.org/10.1113/jphysiol.1976.sp011620

Hodgkin, A. L. & Huxley, A. F. (1939). Action potentials recorded from inside a nerve fibre. *Nature*, 144(3651), 710–711. https://doi.org/10.1038/144710a0

Hodgkin, A. L. & Huxley, A. F. (1952a). Propagation of electrical signals along giant nerve fibres. *Proceedings of the Royal Society of London. Series B-Biological Sciences*, 140(899), 177–183. https://doi.org/10.1098/rspb.1952.0054

Hodgkin, A. L. & Huxley, A. F. (1952b). Currents carried by sodium and potassium ions through the membrane of the giant axon of Loligo. *The Journal of Physiology*, 116(4), 449–472. https://doi.org/10.1113/jphysiol.1952.sp004717

Hodgkin, A. L. & Huxley, A. F. (1952c). The components of membrane conductance in the giant axon of Loligo. *The Journal of Physiology*, 116(4), 473–496. https://doi.org/10.1113/jphysiol.1952.sp004718

Hodgkin, A. L. & Huxley, A. F. (1952d). The dual effect of membrane potential on sodium conductance in the giant axon of Loligo. *The Journal of Physiology*, 116(4), 497–506. https://doi.org/10.1113/jphysiol.1952.sp004719

Hodgkin, A. L. & Huxley, A. F. (1952e). A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of Physiology*, 117(4), 500–544. https://doi.org/10.1113/jphysiol.1952.sp004764

Hüllermeier, E. & Waegeman, W. (2021). Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *Machine Learning*, 110(3), 457–506. https://doi.org/10.1007/s10994-021-05946-3

Huppenkothen, D. & Bachetti, M. (2021). Accurate X-ray timing in the presence of systematic biases with simulation-based inference. *arXiv*. https://arxiv.org/abs/2104.03278

Huxley, A. F. (1972). The quantitative analysis of excitation and conduction in nerve. In *Nobel Lectures, Physiology or Medicine 1963–1970* (pp. 52–69). Elsevier Publishing Company. https://www.nobelprize.org/uploads/2018/06/huxley-lecture.pdf

Izbicki, R., Lee, A., & Schafer, C. (2014). High-dimensional density ratio estimation with extensions to approximate likelihood computation. In *Proceedings of the 17th International Conference on Artificial Intelligence and Statistics*, volume 33 of *Proceedings of Machine Learning Research* (pp. 420–429). PMLR. http://proceedings.mlr.press/v33/izbicki14.html

Jallais, M., Rodrigues, P. L., Gramfort, A., & Wassermann, D. (2021). Cytoarchitecture measurements in brain gray matter using likelihood-free inference. In *International Conference on Information Processing in Medical Imaging* (pp. 191–202). Springer. https://doi.org/10.1007/978-3-030-78191-0_15

Järvenpää, M., et al. (2019). A Bayesian model of acquisition and clearance of bacterial colonization incorporating within-host variation. *PLoS Computational Biology*, 15(4), e1006534. https://doi.org/10.1371/journal.pcbi.1006534

Jeffrey, N., Alsing, J., & Lanusse, F. (2021). Likelihood-free inference with neural compression of DES SV weak lensing map statistics. *Monthly Notices of the Royal Astronomical Society*, 501(1), 954–969. https://doi.org/10.1093/mnras/staa3594

Kandel, E. R., Koester, J. D., Mack, S. H., & Siegelbaum, S. (2021). *Principles of neural science* (6th ed.). McGraw Hill.

Kass, R. E. (2011). Statistical inference: The big picture. *Statistical Science*, 26(1), 1–9. https://doi.org/10.1214/10-STS337

Koperski, J. (2021). Models - Internet encyclopedia of philosophy. https://iep.utm.edu/models/

Kording, K. P., Blohm, G., Schrater, P., & Kay, K. (2020). Appreciating the variety of goals in computational neuroscience. *arXiv*. https://arxiv.org/abs/2002.03211

Lapicque, L. (1907). Recherches quantitatives sur l'excitation électrique des nerfs traitée comme une polarization. *Journal of Physiology and Patholody*, 9, 620–635. For a translated version, see https://dx.doi.org/10.1007/s00422-007-0189-6

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521(7553), 436–444. https://doi.org/10.1038/nature14539

Levy, A. (2014). What was Hodgkin and Huxley's achievement? *The British Journal for the Philosophy of Science*, 65(3), 469–492. https://doi.org/10.1093/bjps/axs043

Lindsay, G. (2021). *Models of the mind: How physics, engineering and mathematics have shaped our understanding of the brain*. Bloomsbury Publishing.

Lueckmann, J.-M., Bassetto, G., Karaletsos, T., & Macke, J. H. (2018). Likelihood-free inference with emulator networks. *arXiv*. https://arxiv.org/abs/1805.09294

Lueckmann, J.-M., Bassetto, G., Karaletsos, T., & Macke, J. H. (2019). Likelihood-free inference with emulator networks. In *Proceedings of The 1st Symposium on Advances in Approximate Bayesian Inference*, volume 96 of *Proceedings of Machine Learning Research* (pp. 32–53). PMLR. http://proceedings.mlr.press/v96/lueckmann19a.html

Lueckmann, J.-M., Boelts, J., Greenberg, D. S., Gonçalves, P. J., & Macke, J. H. (2021). Benchmarking simulation-based inference. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research* (pp. 343–351). PMLR. http://proceedings.mlr.press/v130/lueckmann21a.html

Lueckmann, J.-M., Gonçalves, P. J., Bassetto, G., Öcal, K., Nonnenmacher, M., & Macke, J. H. (2017). Flexible statistical inference for mechanistic models of neural dynamics. In *Advances in Neural Information Processing Systems 30* (pp. 1289–1299). Curran Associates, Inc. https://proceedings.neurips.cc/paper/2017/hash/addfa9b7e234254d26e9c7f2af1005cb-Abstract.html

Makridakis, S. (2017). The forthcoming Artificial Intelligence (AI) revolution: Its impact on society and firms. *Futures*, 90, 46–60. https://doi.org/10.1016/j.futures.2017.03.006

Marder, E., O'Leary, T., & Shruti, S. (2014). Neuromodulation of circuits with variable parameters: Single neurons and small circuits reveal principles of state-dependent and robust neuromodulation. *Annual Review of Neuroscience*, 37, 329–346. https://doi.org/10.1146/annurev-neuro-071013-013958

Marder, E. & Taylor, A. L. (2011). Multiple models to capture the variability in biological neurons and networks. *Nature Neuroscience*, 14(2), 133–138. https://doi.org/10.1038/nn.2735

Marin, J.-M., Pudlo, P., Robert, C. P., & Ryder, R. J. (2012). Approximate Bayesian computational methods. *Statistics and Computing*, 22(6), 1167–1180. https://doi.org/10.1007/s11222-011-9288-2

Marlier, N., Brüls, O., & Louppe, G. (2021). Simulation-based Bayesian inference for multi-fingered robotic grasping. *arXiv*. https://arxiv.org/abs/2109.14275

Mason, P. H. (2010). Degeneracy at multiple levels of complexity. *Biological Theory*, 5(3), 277–288. https://doi.org/10.1162/BIOT_a_00041

Mattingly, W. A., Chang, D.-j., Paris, R., Smith, N., Blevins, J., & Ouyang, M. (2012). Robot design using Unity for computer games and robotic simulations. In *17th International Conference on Computer Games* (pp. 56–59). IEEE. https://doi.org/10.1109/CGames.2012.6314552

McCallum, J. (2021). Cost of CPU performance through time 1944–2003. https://jcmit.net/cpu-performance.htm

Meent, J.-W. v. d., Paige, B., Yang, H., & Wood, F. (2018). An introduction to probabilistic programming. *arXiv*. https://arxiv.org/abs/1809.10756

Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., & Galstyan, A. (2021). A survey on bias and fairness in machine learning. *ACM Computing Surveys (CSUR)*, 54(6), 1–35. https://doi.org/10.1145/3457607

Meunier, C. & Segev, I. (2002). Playing the Devil's advocate: Is the Hodgkin–Huxley model useful? *Trends in Neurosciences*, 25(11), 558–563. https://doi.org/10.1016/S0166-2236(02)02278-6

Mikelson, J. & Khammash, M. (2020). Likelihood-free nested sampling for parameter inference of biochemical reaction networks. *PLOS Computational Biology*, 16(10), 1–24. https://doi.org/10.1371/journal.pcbi.1008264

Mitchell, T. M. (1997). *Machine learning*. McGraw Hill.

Mohamed, S. & Lakshminarayanan, B. (2017). Learning in implicit generative models. In *Fifth International Conference on Learning Representations*. OpenReview. https://openreview.net/forum?id=B16Jem9xe

Mohamed, S., Png, M.-T., & Isaac, W. (2020). Decolonial AI: Decolonial theory as sociotechnical foresight in artificial intelligence. *Philosophy & Technology*, 33(4), 659–684. https://doi.org/10.1007/s13347-020-00405-8

Muratore, F., Gruner, T., Wiese, F., Belousov, B., Gienger, M., & Peters, J. (2021). Neural posterior domain randomization. In *Fifth Conference on Robot Learning*. OpenReview. https://openreview.net/forum?id=59aUaAbVfMA

Murphy, K. P. (2007). Frequentist parameter estimation. https://www.cs.ubc.ca/~murphyk/Teaching/CS340-Fall07/reading/paramEst.pdf

Murre, J. M. & Sturdy, D. P. (1995). The connectivity of the brain: Multi-level quantitative analysis. *Biological Cybernetics*, 73(6), 529–545. https://doi.org/10.1007/BF00199545

Neher, E. & Sakmann, B. (1976). Single-channel currents recorded from membrane of denervated frog muscle fibres. *Nature*, 260(5554), 799–802. https://doi.org/10.1038/260799a0

Nobel Prize Outreach AB (2021). The Nobel Prize in physiology or medicine 1963. https://www.nobelprize.org/prizes/medicine/1963/summary/

Nourali, M. (2021). Comparison of likelihood-free inference approach and a formal Bayesian method in parameter uncertainty assessment: Case study with a single-event rainfall–runoff model. *Journal of Hydrologic Engineering*, 26(3), 05020049. https://doi.org/10.1061/(ASCE)HE.1943-5584.0002048

Oesterle, J., Behrens, C., Schröder, C., Hermann, T., Euler, T., Franke, K., ..., & Berens, P. (2020). Bayesian inference for biophysical neuron models enables stimulus optimization for retinal neuroprosthetics. *Elife*, 9, e54997. https://doi.org/10.7554/eLife.54997

Pacchiardi, L., Künzli, P., Schöngens, M., Chopard, B., & Dutta, R. (2021). Distance-learning for approximate Bayesian computation to model a volcanic eruption. *Sankhya B*, 83(1), 288–317. https://doi.org/10.1007/s13571-019-00208-8

Paninski, L. & Cunningham, J. P. (2018). Neural data science: Accelerating the experiment-analysis-theory cycle in large-scale neuroscience. *Current Opinion in Neurobiology*, 50, 232–241. https://doi.org/10.1016/j.conb.2018.04.007

Papamakarios, G. (2019). Neural density estimation and likelihood-free inference. *arXiv*. PhD thesis. https://arxiv.org/abs/1910.13233

Papamakarios, G. & Murray, I. (2016). Fast ϵ-free inference of simulation models with Bayesian conditional density estimation. In *Advances in Neural Information Processing Systems 29* (pp. 1028–1036). Curran Associates, Inc. https://proceedings.neurips.cc/paper/2016/hash/6aca97005c68f1206823815f66102863-Abstract.html

Papamakarios, G., Nalisnick, E., Rezende, D. J., Mohamed, S., & Lakshminarayanan, B. (2021). Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 22(57), 1–64. https://jmlr.csail.mit.edu/papers/v22/19-1028.html

Papamakarios, G., Sterratt, D., & Murray, I. (2019). Sequential neural likelihood: Fast likelihood-free inference with autoregressive flows. In *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 89 of *Proceedings of Machine Learning Research* (pp. 837–848). PMLR. http://proceedings.mlr.press/v89/papamakarios19a.html

Pham, K. C., Nott, D. J., & Chaudhuri, S. (2014). A note on approximating ABC-MCMC using flexible classifiers. *Stat*, 3(1), 218–227. https://doi.org/10.1002/sta4.56

Podlaski, W. F., Seeholzer, A., Groschner, L. N., Miesenböck, G., Ranjan, R., & Vogels, T. P. (2017). Mapping the function of neuronal ion channels in model and experiment. *Elife*, 6, e22152. https://doi.org/10.7554/eLife.22152

Pospischil, M., Toledo-Rodriguez, M., Monier, C., Piwkowska, Z., Bal, T., Frégnac, Y., Markram, H., & Destexhe, A. (2008). Minimal Hodgkin–Huxley type models for different classes of cortical and thalamic neurons. *Biological Cybernetics*, 99(4), 427–441. https://doi.org/10.1007/s00422-008-0263-8

Prangle, D. (2018). Summary statistics. In *Handbook of Approximate Bayesian Computation* (pp. 125–152). Chapman and Hall/CRC.

Prangle, D., Everitt, R. G., & Kypraios, T. (2018). A rare event approach to high-dimensional approximate Bayesian computation. *Statistics and Computing*, 28(4), 819–834. https://doi.org/10.1007/s11222-017-9764-4

Price, L. F., Drovandi, C. C., Lee, A., & Nott, D. J. (2018). Bayesian synthetic likelihood. *Journal of Computational and Graphical Statistics*, 27(1), 1–11. https://doi.org/10.1080/10618600.2017.1302882

Prinz, A. A., Bucher, D., & Marder, E. (2004). Similar network activity from disparate circuit parameters. *Nature Neuroscience*, 7(12), 1345–1352. https://doi.org/10.1038/nn1352

Pritchard, J. K., Seielstad, M. T., Perez-Lezaun, A., & Feldman, M. W. (1999). Population growth of human Y chromosomes: A study of Y chromosome microsatellites. *Molecular Biology and Evolution*, 16(12), 1791–1798. https://doi.org/10.1093/oxfordjournals.molbev.a026091

Rodrigues, P. L., Moreau, T., Louppe, G., & Gramfort, A. (2021). Leveraging global parameters for flow-based neural posterior estimation. *arXiv*. https://arxiv.org/abs/2102.06477

Rose, S. (2018). Neuroscience at scale: Synaptomics tries to make sense of the brain at a stupefying scale. https://theplosblog.plos.org/2018/09/neuroscience-at-scale-synaptomics-tries-to-make-sense-of-the-brain-at-a-stupefying-scale-by-samuel-rose/

Rothenberg, J. (1989). The nature of modeling. In *Artificial Intelligence, Simulation & Modeling* (pp. 75–92). John Wiley & Sons, Inc. Reprint available at https://www.rand.org/pubs/notes/N3027.html

Rubin, D. B. (1984). Bayesianly justifiable and relevant frequency calculations for the applied statistician. *The Annals of Statistics*, (pp. 1151–1172). https://doi.org/10.1214/aos/1176346785

Rusakov, D. A., Savtchenko, L. P., & Latham, P. E. (2020). Noisy synaptic conductance: Bug or a feature? *Trends in Neurosciences*, 43(6), 363–372. https://doi.org/10.1016/j.tins.2020.03.009

Samuel, A. L. (1959). Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, 3(3), 210–229. https://doi.org/10.1147/rd.33.0210

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks*, 61, 85–117. https://doi.org/10.1016/j.neunet.2014.09.003

Schröder, C., James, B., Lagnado, L., & Berens, P. (2019). Approximate Bayesian inference for a mechanistic model of vesicle release at a ribbon synapse. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc. https://papers.nips.cc/paper/2019/hash/0e57098d0318a954d1443e2974a38fac-Abstract.html

Schwiening, C. J. (2012). A brief historical perspective: Hodgkin and Huxley. *The Journal of Physiology*, 590(Pt 11), 2571. https://doi.org/10.1113/jphysiol.2012.230458

Senn, S. (2011). You may believe you are a Bayesian but you are probably wrong. *Rationality, Markets and Morals*, 2(42), 48–66. https://econpapers.repec.org/article/rmmjournl/v_3a2_3ay_3a2011_3ai_3a42.htm

Simola, U., Cisewski-Kehe, J., Gutmann, M. U., & Corander, J. (2021). Adaptive approximate Bayesian computation tolerance selection. *Bayesian Analysis*, 16(2), 397–423. https://doi.org/10.1214/20-BA1211

Sisson, S. A., Fan, Y., & Tanaka, M. M. (2007). Sequential monte carlo without likelihoods. *Proceedings of the National Academy of Sciences*, 104(6), 1760–1765. https://doi.org/10.1073/pnas.0607208104

Stan Developers, g. (2021). Prior Choice Recommendations. https://github.com/stan-dev/stan/wiki/Prior-Choice-Recommendations

Talts, S., Betancourt, M., Simpson, D., Vehtari, A., & Gelman, A. (2018). Validating Bayesian inference algorithms with simulation-based calibration. *arXiv*. https://arxiv.org/abs/1804.06788

Tavaré, S. (2018). On the history of ABC. In *Handbook of Approximate Bayesian Computation* (pp. 55–69). Chapman and Hall/CRC.

Tavaré, S., Balding, D. J., Griffiths, R. C., & Donnelly, P. (1997). Inferring coalescence times from DNA sequence data. *Genetics*, 145(2), 505–518. https://doi.org/10.1093/genetics/145.2.505

Taylor, A. L., Goaillard, J.-M., & Marder, E. (2009). How multiple conductances determine electrophysiological properties in a multicompartment model. *Journal of Neuroscience*, 29(17), 5573–5586. https://doi.org/10.1523/JNEUROSCI.4438-08.2009

Tejero-Cantero, A., Boelts, J., Deistler, M., Lueckmann, J.-M., Durkan, C., Gonçalves, P. J., Greenberg, D. S., & Macke, J. H. (2020). sbi: A toolkit for simulation-based inference. *Journal of Open Source Software*, 5(52), 2505. https://doi.org/10.21105/joss.02505

Thomas, O., Dutta, R., Corander, J., Kaski, S., & Gutmann, M. U. (2021). Likelihood-free inference by ratio estimation. *Bayesian Analysis*, (pp. 1 – 31). https://doi.org/10.1214/20-BA1238

Tomašev, N., Cornebise, J., Hutter, F., Mohamed, S., Picciariello, A., Connelly, B., ..., & Clopath, C. (2020). AI for social good: Unlocking the opportunity for positive impact. *Nature Communications*, 11(1), 1–6. https://doi.org/10.1038/s41467-020-15871-z

Toni, T., Welch, D., Strelkowa, N., Ipsen, A., & Stumpf, M. P. (2009). Approximate Bayesian computation scheme for parameter inference and model selection in dynamical systems. *Journal of the Royal Society Interface*, 6(31), 187–202. https://doi.org/10.1098/rsif.2008.0172

Urai, A. E., Doiron, B., Leifer, A. M., & Churchland, A. K. (2021). Large-scale neural recordings call for new insights to link brain and behavior. *arXiv*. https://arxiv.org/abs/2103.14662

Vallverdú, J. (2015). *Bayesians versus frequentists: A philosophical debate on statistical reasoning*. Springer. https://doi.org/10.1007/978-3-662-48638-2

van de Meent, J.-W., Paige, B., Yang, H., & Wood, F. (2018). An introduction to probabilistic programming. *arXiv*. https://arxiv.org/abs/1809.10756

Van Geit, W., Gevaert, M., Chindemi, G., Rössert, C., Courcol, J.-D., Muller, E. B., ..., & Markram, H. (2016). BluePyOpt: leveraging open source software and cloud infrastructure to optimise model parameters in neuroscience. *Frontiers in Neuroinformatics*, 10, 17. https://doi.org/10.3389/fninf.2016.00017

Wagenmakers, E.-J. (2007). A practical solution to the pervasive problems of p values. *Psychonomic Bulletin & Review*, 14(5), 779–804. https://doi.org/10.3758/bf03194105

Wagenmakers, E.-J., Lee, M., Lodewyckx, T., & Iverson, G. J. (2008). Bayesian versus frequentist inference. In *Bayesian evaluation of informative hypotheses* (pp. 181–207). Springer. https://doi.org/10.1007/978-0-387-09612-4_9

Wieschen, E. M., Voss, A., & Radev, S. (2020). Jumping to conclusion? A lévy flight model of decision making. *The Quantitative Methods for Psychology*, 16(2), 120–132. https://doi.org/10.20982/tqmp.16.2.p120

Winter, P. M., Eder, S., Weissenböck, J., Schwald, C., Doms, T., Vogt, T., Hochreiter, S., & Nessler, B. (2021). Trusted artificial intelligence: Towards certification of machine learning applications. *arXiv*. https://arxiv.org/abs/2103.16910

Wood, F., Warrington, A., Naderiparizi, S., Weilbach, C., Masrani, V., Harvey, W., ..., & Nasseri, A. (2020). Planning as inference in epidemiological models. *arXiv*. https://arxiv.org/abs/2003.13221

Wood, S. N. (2010). Statistical inference for noisy nonlinear ecological dynamic systems. *Nature*, 466(7310), 1102–1104. https://doi.org/10.1038/nature09319

APPENDIX

Flexible statistical inference for mechanistic models of neural dynamics

Jan-Matthis Lueckmann[*1], **Pedro J. Gonçalves**[*1], **Giacomo Bassetto**[1], **Kaan Öcal**[1,2], **Marcel Nonnenmacher**[1], **Jakob H. Macke**[†1]
[1] research center caesar, an associate of the Max Planck Society, Bonn, Germany
[2] Mathematical Institute, University of Bonn, Bonn, Germany
`{jan-matthis.lueckmann, pedro.goncalves, giacomo.bassetto, kaan.oecal, marcel.nonnenmacher, jakob.macke}@caesar.de`

Abstract

Mechanistic models of single-neuron dynamics have been extensively studied in computational neuroscience. However, identifying which models can quantitatively reproduce empirically measured data has been challenging. We propose to overcome this limitation by using likelihood-free inference approaches (also known as Approximate Bayesian Computation, ABC) to perform full Bayesian inference on single-neuron models. Our approach builds on recent advances in ABC by learning a neural network which maps features of the observed data to the posterior distribution over parameters. We learn a Bayesian mixture-density network approximating the posterior over multiple rounds of adaptively chosen simulations. Furthermore, we propose an efficient approach for handling missing features and parameter settings for which the simulator fails, as well as a strategy for automatically learning relevant features using recurrent neural networks. On synthetic data, our approach efficiently estimates posterior distributions and recovers ground-truth parameters. On in-vitro recordings of membrane voltages, we recover multivariate posteriors over biophysical parameters, which yield model-predicted voltage traces that accurately match empirical data. Our approach will enable neuroscientists to perform Bayesian inference on complex neuron models without having to design model-specific algorithms, closing the gap between mechanistic and statistical approaches to single-neuron modelling.

1 Introduction

Biophysical models of neuronal dynamics are of central importance for understanding the mechanisms by which neural circuits process information and control behaviour. However, identifying which models of neural dynamics can (or cannot) reproduce electrophysiological or imaging measurements of neural activity has been a major challenge [1]. In particular, many models of interest – such as multi-compartment biophysical models [2], networks of spiking neurons [3] or detailed simulations of brain activity [4] – have intractable or computationally expensive likelihoods, and statistical inference has only been possible in selected cases and using model-specific algorithms [5, 6, 7]. Many models are defined implicitly through *simulators*, i.e. a set of dynamical equations and possibly a description of sources of stochasticity [1]. In addition, it is often of interest to identify models which can reproduce particular *features* in the data, e.g. a firing rate or response latency, rather than the full temporal structure of a neural recording.

[*]Equal contribution
[†]Current primary affiliation: Centre for Cognitive Science, Technical University Darmstadt

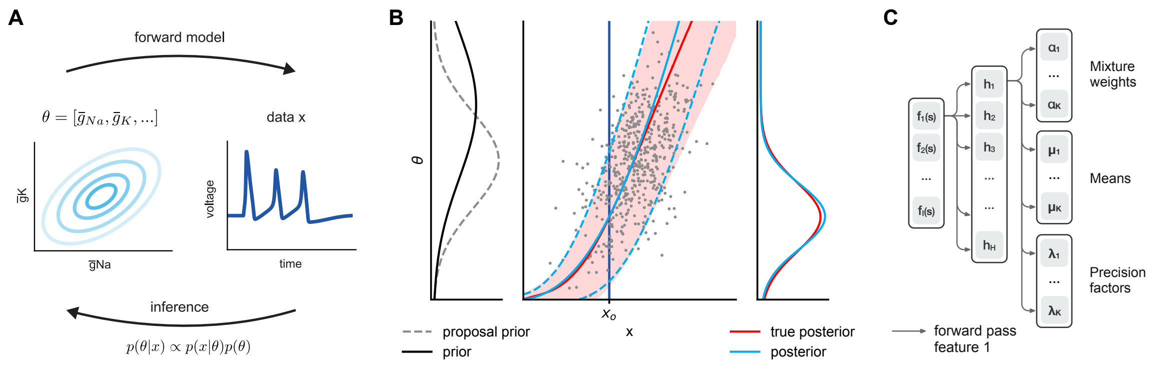


Figure 1: **Flexible likelihood-free inference for models of neural dynamics. A.** We want to flexibly and efficiently infer the posterior over model parameters given observed data, on a wide range of models of neural dynamics. **B.** Our method approximates the true posterior on $\boldsymbol{\theta}$ around the observed data $\mathbf{x}_o$ by performing density estimation on data simulated using a proposal prior. **C.** We train a Bayesian mixture-density network (MDN) for posterior density estimation.

In the absence of likelihoods, the standard approach in neuroscience has been to use heuristic parameter-fitting methods [2, 8, 9]: distance measures are defined on multiple features of interest, and brute-force search [10, 11] or evolutionary algorithms [2, 9, 12, 13] (neither of which scales to high-dimensional parameter spaces) are used to minimise the distances between observed and model-derived features. As it is difficult to trade off distances between different features, the state-of-the-art methods optimise multiple objectives and leave the final choice of a model to the user [2, 9]. As this approach is not based on statistical inference, it does not provide estimates of the full posterior distribution – thus, while this approach has been of great importance for identifying 'best fitting' parameters, it does not allow one to identify the full space of parameters that are consistent with data and prior knowledge, or to incrementally refine and reject models.

Bayesian inference for likelihood-free simulator models, also known as Approximate Bayesian Computation [14, 15, 16], provides an attractive framework for overcoming these limitations: like parameter-fitting approaches in neuroscience [2, 8, 9], it is based on comparing summary features between simulated and empirical data. However, unlike them, it provides a principled framework for full Bayesian inference and can be used to determine how to trade off goodness-of-fit across summary statistics. However, to the best of our knowledge, this potential has not been realised yet, and ABC approaches are not used for linking mechanistic models of neural dynamics with experimental data (for an exception, see [17]). Here, we propose to use ABC methods for statistical inference of mechanistic models of single neurons. We argue that ABC approaches based on conditional density estimation [18, 19] are particularly suited for neuroscience applications.

We present a novel method (Sequential Neural Posterior Estimation, SNPE) in which we sequentially train a mixture-density network across multiple rounds of adaptively chosen simulations[1]. Our approach is directly inspired by prior work [18, 19], but overcomes critical limitations: first, a flexible mixture-density network trained with an importance-weighted loss function enables us to use complex proposal distributions and approximate complex posteriors. Second, we represent a full posterior over network parameters of the density estimator (i.e. a "posterior on posterior-parameters") which allows us to take uncertainty into account when adjusting weights. This enables us to perform 'continual learning', i.e. to effectively utilise all simulations without explicitly having to store them. Third, we introduce an approach for efficiently dealing with simulations that return missing values, or which break altogether – a common situation in neuroscience and many other applications of simulator-based models – by learning a model that predicts which parameters are likely to lead to breaking simulations, and using this knowledge to modify the proposal distribution. We demonstrate the practical effectiveness and importance of these innovations on biophysical models of single neurons, on simulated and neurophysiological data. Finally, we show how recurrent neural networks can be used to directly learn relevant features from time-series data.

[1]Code available at https://github.com/mackelab/delfi

1.1 Related work using likelihood-free inference for simulator models

Given experimental data $\mathbf{x}_o$ (e.g. intracellular voltage measurements of a single neuron, or extracellular recordings from a neural population), a model $p(\mathbf{x}|\boldsymbol{\theta})$ parameterised by $\boldsymbol{\theta}$ (e.g. biophysical parameters, or connectivity strengths in a network simulation) and a prior distribution $p(\boldsymbol{\theta})$, our goal is to perform statistical inference, i.e. to find the posterior distribution $\hat{p}(\boldsymbol{\theta}|\mathbf{x} = \mathbf{x}_o)$. We assume that the model $p(\mathbf{x}|\boldsymbol{\theta})$ is only defined through a *simulator* [14, 15]: we can generate samples $\mathbf{x}_n \sim \mathbf{x}|\boldsymbol{\theta}$ from it, but not evaluate $p(\mathbf{x}|\boldsymbol{\theta})$ (or its gradients) explicitly. In neural modelling, many models are defined through specification of a dynamical system with external or intrinsic noise sources or even through a black-box simulator (e.g. using the NEURON software [20]).

In addition, and in line with parameter-fitting approaches in neuroscience and most ABC techniques [14, 15, 21], we are often interested in capturing summary statistics of the experimental data (e.g. firing rate, spike-latency, resting potential of a neuron). Therefore, we can think of $\mathbf{x}$ as resulting from applying a feature function f to the raw simulator output $\mathbf{s}$, $\mathbf{x} = f(\mathbf{s})$, with $\dim(\mathbf{x}) \ll \dim(\mathbf{s})$.

Classical ABC algorithms simulate from multiple parameters, and reject parameter sets which yield data that are not within a specified distance from the empirically observed features. In their basic form, proposals are drawn from the prior ('rejection-ABC' [22]). More efficient variants make use of a Markov-Chain Monte-Carlo [23, 24] or Sequential Monte-Carlo (SMC) samplers [25, 26]. Sampling-based ABC approaches require the design of a distance metric on summary features, as well as a rejection criterion (ε), and are exact only in the limit of small ε (i.e. many rejections) [27], implying strong trade-offs between accuracy and scalability. In SMC-ABC, importance sampling is used to sequentially sample from more accurate posteriors while ε is gradually decreased.

Synthetic-likelihood methods [28, 21, 29] approximate the likelihood $p(\mathbf{x}|\boldsymbol{\theta})$ using multivariate Gaussians fitted to repeated simulations given $\boldsymbol{\theta}$ (see [30, 31] for generalisations). While the Gaussianity assumption is often motivated by the central limit theorem, distributions over features can in practice be complex and highly non-Gaussian [32]. For example, neural simulations sometimes result in systematically missing features (e.g. spike latency is undefined if there are no spikes), or diverging firing rates.

Finally, methods originating from regression correction [33, 18, 19] simulate multiple data $\mathbf{x}_n$ from different $\boldsymbol{\theta}_n$ sampled from a proposal distribution $\tilde{p}(\boldsymbol{\theta})$, and construct a conditional density estimate $q(\boldsymbol{\theta}|\mathbf{x})$ by performing a regression from simulated data $\mathbf{x}_n$ to $\boldsymbol{\theta}_n$. Evaluating this density model at the observed data $\mathbf{x}_o$, $q(\boldsymbol{\theta}|\mathbf{x}_o)$ yields an estimate of the posterior distribution. These approaches do not require parametric assumptions on likelihoods or the choice of a distance function and a tolerance (ε) on features. Two approaches are used for correcting the mismatch between prior and proposal distributions: Blum and François [18] proposed the importance weights $p(\boldsymbol{\theta})/\tilde{p}(\boldsymbol{\theta})$, but restricted themselves to proposals which were truncated priors (i.e. all importance weights were 0 or 1), and did not sequentially optimise proposals over multiple rounds. Papamakarios and Murray [19] recently used stochastic variational inference to optimise the parameters of a mixture-density network, and a post-hoc division step to correct for the effect of the proposal distribution. While highly effective in some cases, this closed-form correction step can be numerically unstable and is restricted to Gaussian and uniform proposals, limiting both the robustness and flexibility of this approach. SNPE builds on these approaches, but overcomes their limitations by introducing four innovations: a highly flexible proposal distribution parameterised as a mixture-density network, a Bayesian approach for continual learning from multiple rounds of simulations, and a classifier for predicting which parameters will result in aborted simulations or missing features. Fourth, we show how this approach, when applied to time-series data of single-neuron activity, can automatically learn summary features from data.

2 Methods

2.1 Sequential Neural Posterior Estimation for likelihood-free inference

In SNPE, our goal is to learn the parameters $\boldsymbol{\phi}$ of a posterior model $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x} = f(\mathbf{s}))$ which, when evaluated at $\mathbf{x}_o$, approximates the true posterior $p(\boldsymbol{\theta}|\mathbf{x}_o) \approx q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x} = \mathbf{x}_o)$. Given a prior $p(\boldsymbol{\theta})$, a proposal prior $\tilde{p}(\boldsymbol{\theta})$, pairs of samples $(\boldsymbol{\theta}_n, \mathbf{x}_n)$ generated from the proposal prior and the simulator, and a calibration kernel K_τ, the posterior model can be trained by minimising the importance-weighted log-loss

$$\mathcal{L}(\boldsymbol{\phi}) = -\frac{1}{N} \sum_n \frac{p(\boldsymbol{\theta}_n)}{\tilde{p}(\boldsymbol{\theta}_n)} K_\tau(\mathbf{x}_n, \mathbf{x}_o) \log q_{\boldsymbol{\phi}}(\boldsymbol{\theta}_n|\mathbf{x}_n), \tag{1}$$

as is shown by extending the argument in [19] with importance-weights $p(\boldsymbol{\theta}_n)/\tilde{p}(\boldsymbol{\theta}_n)$ and a kernel K_τ in Appendix A.

Sampling from a proposal prior can be much more effective than sampling from the prior. By including the importance weights in the loss, the analytical correction step of [19] (i.e. division by the proposal prior) becomes unnecessary: SNPE directly estimates the posterior density rather than a conditional density that is reweighted post-hoc. The analytical step of [19] has the advantage of side-stepping the additional variance brought about by importance-weights, but has the disadvantages of (1) being restricted to Gaussian proposals, and (2) the division being unstable if the proposal prior has higher precision than the estimated conditional density.

The calibration kernel $K_\tau(\mathbf{x}, \mathbf{x}_o)$ can be used to calibrate the loss function by focusing it on simulated data points $\mathbf{x}$ which are close to $\mathbf{x}_o$ [18]. Calibration kernels $K_\tau(\mathbf{x}, \mathbf{x}_o)$ are to be chosen such that $K_\tau(\mathbf{x}_o, \mathbf{x}_o) = 1$ and that K_τ decreases with increasing distance $\|\mathbf{x} - \mathbf{x}_o\|$, given a bandwidth τ[2]. Here, we only used calibration kernels to exclude bad simulations by assigning them kernel value zero. An additional use of calibration kernels would be to limit the accuracy of the posterior density estimation to a region near $\mathbf{x}_o$. Choice of the bandwidth implies a bias-variance trade-off [18]. For the problems we consider here, we assumed our posterior model $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x})$ based on a multi-layer neural network to be sufficiently flexible, such that limiting bandwidth was not necessary.

We sequentially optimise the density estimator $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x}) = \sum_k \alpha_k \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ by training a mixture-density network (MDN) [19] with parameters $\boldsymbol{\phi}$ over multiple 'rounds' r with adaptively chosen proposal priors $\tilde{p}^{(r)}(\boldsymbol{\theta})$ (see Fig. 1). We initialise the proposal prior at the prior, $\tilde{p}^{(1)}(\boldsymbol{\theta}) = p(\boldsymbol{\theta})$, and subsequently take the posterior of the previous round as the next proposal prior (Appendix B). Our approach is not limited to Gaussian proposals, and in particular can utilise multi-modal and heavy-tailed proposal distributions.

2.2 Training the posterior model with stochastic variational inference

To make efficient use of simulation time, we want the posterior network $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x})$ to use *all* simulations, including ones from previous rounds. For computational and memory efficiency, it is desirable to avoid having to store all old samples, or having to train a new model at each round. To achieve this goal, we perform Bayesian inference on the weights $\mathbf{w}$ of the MDN across rounds. We approximate the distribution over weights as independent Gaussians [34, 35]. Note that the parameters $\boldsymbol{\phi}$ of this Bayesian MDN are are means and standard deviations per each weight, i.e., $\boldsymbol{\phi} = \{\boldsymbol{\phi}_m, \boldsymbol{\phi}_s\}$. As an extension to the approach of [19], rather than assuming a zero-centred prior over weights, we use the posterior over weights of the previous round, $\pi_{\boldsymbol{\phi}^{(r-1)}}(\mathbf{w})$, as a prior for the next round. Using stochastic variational inference, in each round, we optimise the modified loss

$$\begin{aligned}\mathcal{L}(\boldsymbol{\phi}^{(r)}) = &- \frac{1}{N}\sum_n \frac{p(\boldsymbol{\theta}_n)}{\tilde{p}^{(r)}(\boldsymbol{\theta}_n)} K_\tau(\mathbf{x}_n, \mathbf{x}_o) \big\langle \log q_{\mathbf{w}}(\boldsymbol{\theta}_n|\mathbf{x}_n) \big\rangle_{\pi_{\boldsymbol{\phi}^{(r)}}(\mathbf{w})} \\ &+ \frac{1}{N} D_{\mathrm{KL}}\left(\pi_{\boldsymbol{\phi}^{(r)}}(\mathbf{w}) || \pi_{\boldsymbol{\phi}^{(r-1)}}(\mathbf{w})\right).\end{aligned} \tag{2}$$

Here, the distributions $\pi(\mathbf{w})$ are approximated by multivariate normals with diagonal covariance. The continuity penalty ensures that MDN parameters that are already well constrained by previous rounds are less likely to be updated than parameters with large uncertainty (see Appendix C). In practice, gradients of the expectation over networks are approximated using the local reparameterisation trick [36].

2.3 Dealing with bad simulations and bad features, and learning features from time series

Bad simulations: Simulator-based models, and single-neuron models in particular, frequently generate nonsensical data (which we name 'bad simulations'), especially in early rounds in which the relevant region of parameter space has not yet been found. For example, models of neural dynamics can easily run into self-excitation loops with diverging firing rates [37] (Fig. 4A). We introduce a feature $b(\mathbf{s}) = 1$ to indicate that $\mathbf{s}$ and $\mathbf{x}$ correspond to a bad simulation. We set $K(\mathbf{x}_n, \mathbf{x}_o) = 0$

[2]While we did not investigate this here, an attractive idea would be to base the kernel of the distance between $\mathbf{x}_n$ and $\mathbf{x}_o$ on the divergence between the associated posteriors, e.g. $K_\tau(\mathbf{x}_n, \mathbf{x}_o) = \exp(-1/\tau D_{\mathrm{KL}}(q^{(r-1)}(\boldsymbol{\theta}|\mathbf{x}_n)||q^{(r-1)}(\boldsymbol{\theta}|\mathbf{x}_o)))$ – in this case, two data would be regarded as similar if the current estimation of the density network assigns similar posterior distributions to them, which is a natural measure of similarity in this context.

whenever $b(\mathbf{x}_n) = 1$ since the density estimator should not spend resources on approximating the posterior for bad data. With this choice of calibration kernel, bad simulations are ignored when updating the posterior model – however, this results in inefficient use of simulations.

We propose to learn a model $\hat{g} : \boldsymbol{\theta} \to [0, 1]$ to predict the probability that a simulation from $\boldsymbol{\theta}$ will break. While any probabilistic classifier could be used, we train a binary-output neural network with log-loss on $(\boldsymbol{\theta}_n, b(\mathbf{s}_n))$. For each proposed $\boldsymbol{\theta}$, we reject $\boldsymbol{\theta}$ with probability $\hat{g}(\boldsymbol{\theta})$, and do not carry out the expensive simulation[3]. The rejections could be incorporated into the importance weights (which would require estimating the corresponding partition function, or assuming it to be constant across rounds), but as these rejections do not depend on the data $\mathbf{x}_o$, we interpret them as modifying the prior: from an initially specified prior $p(\boldsymbol{\theta})$, we obtain a modified prior excluding those parameters which likely will lead to nonsensical simulations. Therefore, the predictive model $\hat{g}(\boldsymbol{\theta})$ does not only lead to more efficient inference (especially in strongly under-constrained scenarios), but is also useful in identifying an effective prior – the space of parameters deemed plausible a priori intersected with the space of parameters for which the simulator is well-behaved.

Bad features: It is frequently observed that individual features of interest for fitting single-neuron models cannot be evaluated: for example, the spike latency cannot be evaluated if a simulation does not generate spikes, but the fact that this feature is missing might provide valuable information (Fig. 4C). SNPE can be extended to handle 'bad features' by using a carefully designed posterior network. For each feature $f_i(\mathbf{s})$, we introduce a binary feature $m_i(\mathbf{s})$ which indicates whether f_i is missing. We parameterise the input layer of the posterior network with multiplicative terms of the form $h_i(\mathbf{s}) = f_i(\mathbf{s}) \cdot (1 - m_i(\mathbf{s})) + c_i \cdot m_i(\mathbf{s})$ where the term c_i is to be learned. This approach effectively learns an imputation value c_i for each missing feature. For a more expressive model, one could also include terms which learn interactions across different missing-feature indicators and/or features, but we did not explore this here.

Learning features: Finally, we point out that using a neural network for posterior estimation yields a straightforward way of *learning* relevant features from data [38, 39, 40]. Rather than feeding summary features $f(\mathbf{s})$ into the network, we directly feed time-series recordings of neural activity into the network. The first layer of the MDN becomes a recurrent layer instead of a fully-connected one. By minimising the variational objective (Eq.2), the network learns informative summary features about posterior densities.

3 Results

While SNPE is in principle applicable to any simulator-based model, we designed it for performing inference on models of neural dynamics. In our applications, we concentrate on single-neuron models. We demonstrate the ability of SNPE to recover ground-truth posteriors in Gaussian Mixtures and Generalised Linear Models (GLMs) [41], and apply SNPE to a Hodgkin-Huxley neuron model and an autapse model, which can have parameter regimes of unstable behaviour and missing features.

3.1 Statistical inference on simple models

Gaussian mixtures: We first demonstrate the effectiveness of SNPE for inferring the posterior of mixtures of two Gaussians, for which we can analytically compute true posteriors. We are interested in the numerical stability of the method ('robustness') and the 'flexibility' to approximate multi-modal posteriors. To illustrate the robustness of SNPE, we apply SNPE and the method proposed by [19] (which we refer to by Conditional Density Estimation for Likelihood-free Inference, CDE-LFI) to infer the common mean of a mixture of two Gaussians, given samples from the mixture distribution (Fig. 2A; details in Appendix D.1). Whereas SNPE works robustly across multiple algorithmic rounds, CDE-LFI can become unstable: its analytical correction requires a division by a Gaussian which becomes unstable if the precision of the Gaussian does not increase monotonically across rounds (see 2.1). Constraining the precision-matrix to be non-decreasing fixes the numerical issue, but leads to biased estimates of the posterior. Second, we apply both SNPE and CDE-LFI to infer the two means of a mixture of two Gaussians, given samples $\mathbf{x}$ from the mixture distribution (Fig. 2B; Appendix D.1). While SNPE can use bi-modal proposals, CDE-LFI cannot, implying reduced efficiency of proposals on strongly non-Gaussian or multi-modal problems.

[3]An alternative approach would be to first learn $p(\boldsymbol{\theta}|b(\mathbf{s}) = 0)$ by applying SNPE to a single feature, $f_1(\mathbf{s}) = b(\mathbf{s})$, and to subsequently run SNPE on the full feature-set, but using $p(\boldsymbol{\theta}|b(s) = 0)$ as prior – however, this would 'waste' simulations for learning $p(\boldsymbol{\theta}|b(\mathbf{s}) = 1)$.

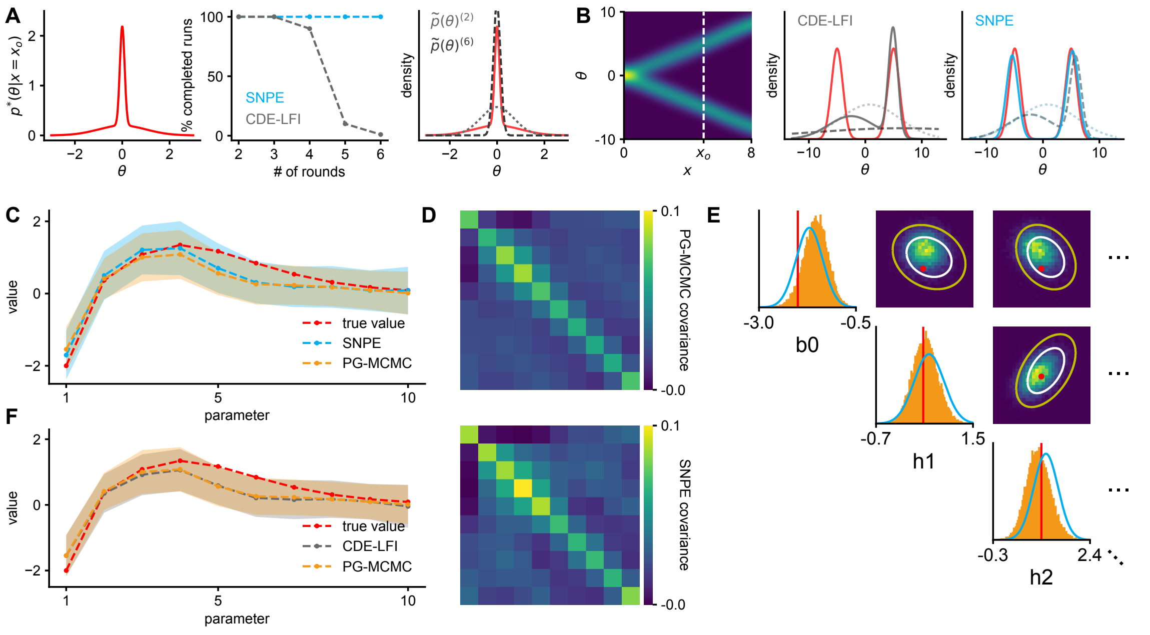


Figure 2: **Inference on simple statistical models. A.** Robustness of posterior inference on 1-D Gaussian Mixtures (GMs). Left: true posterior given observation at $\mathbf{x}_o = 0$. Middle: percentage of completed runs as a function of number of rounds; SNPE is robust. Right: Gaussian proposal priors tend to underestimate tails of posterior (red). **B.** Flexibility of posterior inference. Left: True posterior for 1-D bimodal GM and observation $\mathbf{x}_o$. Middle and right: First round proposal priors (dotted), second round proposal priors (dashed) and estimated posteriors (solid) for CDE-LFI and SNPE respectively (true posterior red). SNPE allows multi-modal proposals. **C, F.** Application to GLM. Posterior means and variances are recovered well by both CDE-LFI and SNPE. For reference, we approximate the posterior using likelihood-based PG-MCMC. **D.** Covariance matrices for SNPE and PG-MCMC. **E.** Partial view of the posterior for 3 out of 10 parameters (all 10 parameters in Appendix G). Ground-truth parameters in red. 2-D marginals for SNPE (lines) and PG-MCMC (histograms). White and yellow contour lines correspond to 68% and 95% of the mass, respectively.

Generalised linear models: Generalised linear models (GLM) are commonly used to model neural responses to sensory stimuli. For these models, several techniques are available to estimate the posterior distribution over parameters, making them ideally suited to test SNPE in a single-neuron model. We evaluated the posterior distribution over the parameters of a GLM using a Pólya-Gamma sampler (PG-MCMC, [42, 43]) and compared it to the posterior distributions estimated by SNPE (Appendix D.2 for details). We found a good agreement of the posterior means and variances (Fig. 2C), covariances (Fig. 2D), as well as pairwise marginals (Fig. 2E). We note that, since GLMs have close-to-Gaussian posteriors, the CDE-LFI method works extremely well on this problem (Fig. 2F).

In summary, SNPE leads to accurate and robust estimation of the posterior in simple models. It works effectively even on multi-modal posteriors on which CDE-LFI exhibits worse performance. On a GLM-example with an (almost) Gaussian posterior, the CDE-LFI method works extremely well, but SNPE yields very similar posterior estimates (see Appendix F for additional comparison with SMC-ABC).

3.2 Statistical inference on Hodgkin-Huxley neuron models

Simulated data: The Hodgkin-Huxley equations [44] describe the dynamics of a neuron's membrane potential and ion channels given biophysical parameters (e.g. concentration of sodium and potassium channels) and an injected input current (Fig. 3A, see Appendix D.3). We applied SNPE to a Hodgkin-Huxley model with channel kinetics as in [45] and inferred the posterior over 12 biophysical parameters, given 20 voltage features of the simulated data. The true parameter values are close to the mode of the inferred posterior (Fig. 3B, D), and in a region of high posterior probability. Samples from the posterior lead to voltage traces that are similar to the original data, supporting the correctness of the approach (Fig. 3C).

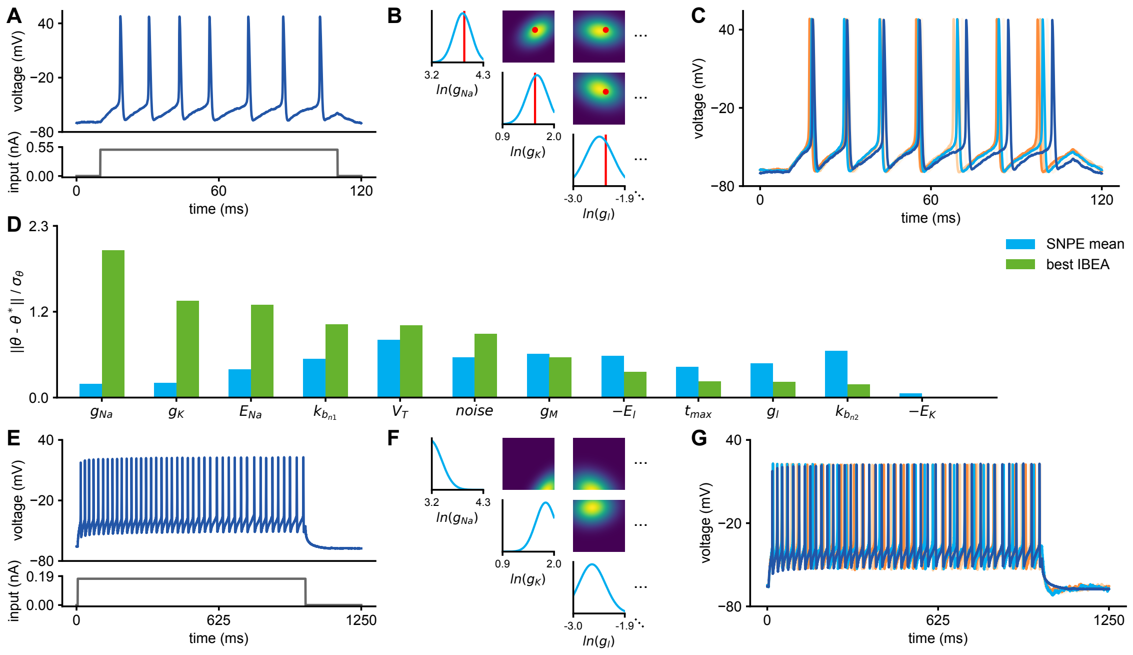


Figure 3: **Application to Hodgkin-Huxley model: A.** Simulation of Hodgkin-Huxley model with current injection. **B.** Posterior over 3 out of 12 parameters inferred with SNPE (12 parameters in Appendix G). True parameters have high posterior probabilities (red). **C.** Traces for the mode (cyan) of and samples (orange) from the inferred posterior match the original data (blue). **D.** Comparison between SNPE and a standard parameter-fitting procedure based on a genetic algorithm, IBEA: difference between the mode of SNPE or IBEA best parameter set, and the ground-truth parameters, normalised by the standard deviations obtained by SNPE. **E-G.** Application to real data from Allen Cell Type Database. Inference over 12 parameters for cell *464212183*. Results presented as in A-C.

Biophysical neuron models are typically fit to data with genetic algorithms applied to the distance between simulated and measured data-features [2, 8, 9, 46]. We compared the performance of SNPE with a commonly used genetic algorithm (Indicator Based Evolutionary Algorithm, IBEA, from the BluePyOpt package [9]), given the same number of model simulations (Fig. 3D). SNPE is comparable to IBEA in approximating the ground-truth parameters – note that defining an objective measure to compare the two approaches is difficult, as they both minimise different criteria. However, unlike IBEA, SNPE also returns a full posterior distribution, i.e. the space of all parameters consistent with the data, rather than just a 'best fit'.

In-vitro recordings: We also applied the approach to *in vitro* recordings from the mouse visual cortex (see Appendix D.4, Fig. 3E-G). The posterior mode over 12 parameters of a Hodgkin-Huxley model leads to a voltage trace which is similar to the data, and the posterior distribution shows the space of parameters for which the output of the model is preserved. These posteriors could be used to motivate further experiments for constraining parameters, or to study invariances in the model.

3.3 Dealing with bad simulations and features

Bad simulations: We demonstrate our approach (see Section 2.3) for dealing with 'bad simulations' (e.g. for which firing rates diverge) using a simple, two-parameter 'autapse' model for which the region of stability is known. During SNPE, we concurrently train a classifier to predict 'bad simulations' and update the prior accordingly. This approach does not only lead to a more efficient use of simulations, but also identifies the parameter space for which the simulator is well-defined, information that could be used for further model analysis (Fig. 4A, B).

Bad features: Many features of interest in neural models, e.g. the latency to first spike after the injection of a current input, are only well defined in the presence of other features, e.g. the presence of spikes (Fig. 4C). Given that large parts of the parameter space can lead to non-spiking behaviour, missing features occur frequently and cannot simply be ignored. We enriched our MDN with an extra layer which imputes values to the absent features, values which are optimised alongside the rest of the parameters of the network (Fig. 4D; Appendix E). Such imputation has marginal computational

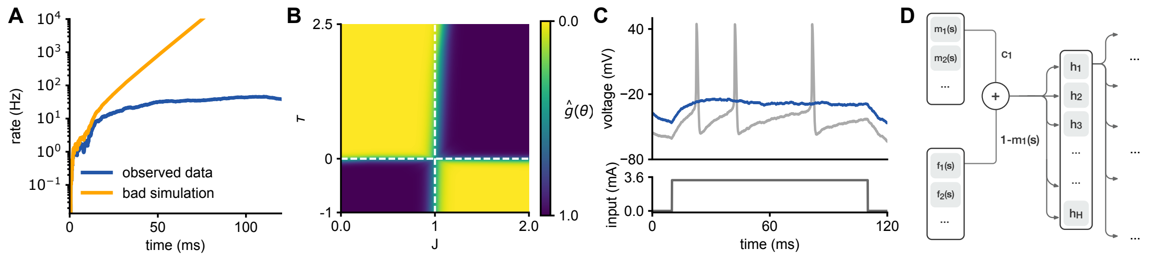


Figure 4: **Inference on neural dynamics has to deal with diverging simulations and missing features. A.** Firing rate of a model neuron connected to itself (autapse). If the strength of the self-connection (parameter J) is bigger than 1, the dynamics are unstable (orange line - bad simulation). **B.** Portion of parameter space leading to diverging simulations learned by the classifier (yellow: low probability of bad simulation, blue: high probability), and comparison with analytically computed boundaries (white, see Appendix D.5). **C.** Illustration of a model neuron in two parameter regimes, spiking (grey trace) and non-spiking (blue). When the neuron does not spike, features that depend on the presence of spiking, such as the latency to first spike, are not defined. **D.** Our MDN is augmented with a multiplicative layer which imputes values for missing features.

cost and grants us the convenience of not having to hand-tune imputation values, or to reject all simulations for which any individual feature might be missing.

Learning features with recurrent neural networks (RNNs): In neural modelling, it is often of interest to work with hand-designed features that are thought to be particularly important or informative for particular analysis questions [2]. For instance, the shape of the action potential is intimately related to the dynamics of sodium and potassium channels in the Hodgkin-Huxley model. However, the space of possible features is immense, and given the highly non-linear nature of many of the neural models in question, it can sometimes be of interest to simply perform statistical inference without having to hand-design features. Our approach provides a straightforward means of doing that: we augment the MDN with a RNN which runs along the recorded voltage trace (and stimulus, here a coloured-noise input) to learn appropriate features to constrain the model parameters. As illustrated in figure 5B, the first layer of the network, which previously received pre-computed summary statistics as inputs, is replaced by a recurrent layer that receives full voltage and current traces as inputs. In order to capture long-term dependencies in the sequence input, we use gated-recurrent units (GRUs) for the RNN [47]. Since we are using 25 GRU units and only keep the final output of the unrolled RNN (many-to-one), we introduce a bottleneck. The RNN thus transforms the voltage trace and stimulus into a set of 25 features, which allow SNPE to recover the posterior over the 12 parameters (Fig. 5C). As expected, the presence of spikes in the observed data leads to a tighter posterior for parameters associated to the main ion channels involved in spike generation, E_{Na}, E_{K}, g_{Na} and g_{K}.

4 Discussion

Quantitatively linking models of neural dynamics to data is a central problem in computational neuroscience. We showed that likelihood-free inference is at least as general and efficient as 'black-box' parameter fitting approaches in neuroscience, but provides full statistical inference, suggesting it to be the method of choice for inference on single-neuron models. We argued that ABC approaches based on density estimation are particularly useful for neuroscience, and introduced a novel algorithm (SNPE) for estimating posterior distributions. We can flexibly and robustly estimate posterior distributions, even when large regions of the parameter space correspond to unstable model behaviour, or when features of choice are missing. Furthermore, we have extended our approach with RNNs to automatically define features, thus increasing the potential for better capturing salient aspects of the data with highly non-linear models. SNPE is therefore equipped to estimate posterior distributions under common constraints in neural models.

Our approach directly builds on a recent approach for density estimation ABC (CDE-LFI, [19]). While we found CDE-LFI to work well on problems with unimodal, close-to-Gaussian posteriors and stable simulators, our approach extends the range of possible applications, and these extensions are critical for the application to neuron models. A key component of SNPE is the proposal prior, which guides the sampling on each round of the algorithm. Here, we used the posterior on the previous round as the proposal for the next one, as in CDE-LFI and in many Sequential-MC approaches. Our

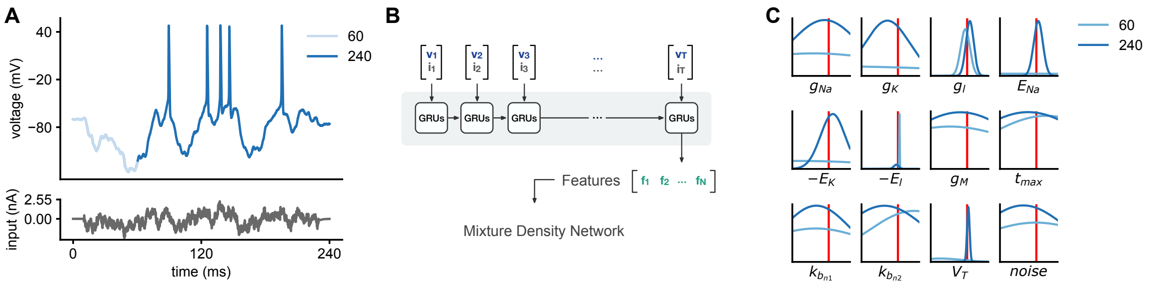


Figure 5: **We can learn informative features using a recurrent mixture-density network (R-MDN). A.** We consider a neuron driven by a colored-noise input current. **B.** Rather than engineering summary features to reduce the dimensionality of observations, we provide the complete voltage trace and input current as input to an R-MDN. The unrolled forward pass is illustrated, where a many-to-one recurrent network reduces the dimensionality of the inputs (T time steps long) to a feature vector of dimensionality N. **C.** Our goal is to infer the posterior density for two different observations: (1) the full 240ms trace shown in panel A; and (2) the initial 60ms of its duration, which do not show any spike. We show the obtained marginal posterior densities for the two observations, using a 25-dimensional feature vector learned by the RNN. In the presence of spikes, the posterior uncertainty gets tighter around the true parameters related to spiking.

method could be extended by alternative approaches to designing proposal priors [48, 49], e.g. by exploiting the fact that we also represent a posterior over MDN parameters: for example, one could design proposals that guide sampling towards regions of the parameter space where the uncertainty about the parameters of the posterior model is highest. We note that, while here we concentrated on models of single neurons, ABC methods and our approach will also be applicable to models of populations of neurons. Our approach will enable neuroscientists to perform Bayesian inference on complex neuron models without having to design model-specific algorithms, closing the gap between mechanistic and statistical models, and enabling theory-driven data-analysis [50].

Acknowledgements

We thank Maneesh Sahani, David Greenberg and Balaji Lakshminarayanan for useful comments on the manuscript. This work was supported by SFB 1089 (University of Bonn) and SFB 1233 (University of Tübingen) of the German Research Foundation (DFG) to JHM and by the caesar foundation.

References

[1] W Gerstner, W M Kistler, R Naud, and L Paninski. *Neuronal dynamics: From single neurons to networks and models of cognition*. Cambridge University Press, 2014.

[2] S Druckmann, Y Banitt, A Gidon, F Schürmann, H Markram, and I Segev. A novel multiple objective optimization framework for constraining conductance-based neuron models by experimental data. *Front Neurosci*, 1, 2007.

[3] C van Vreeswijk and H Sompolinsky. Chaos in neuronal networks with balanced excitatory and inhibitory activity. *Science*, 274(5293), 1996.

[4] H Markram et al. Reconstruction and Simulation of Neocortical Microcircuitry. *Cell*, 163(2), 2015.

[5] Q J M Huys and L Paninski. Smoothing of, and parameter estimation from, noisy biophysical recordings. *PLoS Comput Biol*, 5(5), 2009.

[6] L Meng, M A Kramer, and U T Eden. A sequential monte carlo approach to estimate biophysical neural models from spikes. *J Neural Eng*, 8(6), 2011.

[7] C D Meliza, M Kostuk, H Huang, A Nogaret, D Margoliash, and H D I Abarbanel. Estimating parameters and predicting membrane voltages with conductance-based neuron models. *Biol Cybern*, 108(4), 2014.

[8] C Rossant, D F M Goodman, B Fontaine, J Platkiewicz, A K Magnusson, and R Brette. Fitting neuron models to spike trains. *Front Neurosci*, 5:9, 2011.

[9] W Van Geit, M Gevaert, G Chindemi, C Rössert, J Courcol, E B Muller, F Schürmann, I Segev, and H Markram. Bluepyopt: Leveraging open source software and cloud infrastructure to optimise model parameters in neuroscience. *Front Neuroinform*, 10:17, 2016.

[10] A A Prinz, C P Billimoria, and E Marder. Alternative to hand-tuning conductance-based models: Construction and analysis of databases of model neurons. *J Neurophysiol*, 90(6), 2003.

[11] C Stringer, M Pachitariu, N A Steinmetz, M Okun, P Bartho, K D Harris, M Sahani, and N A Lesica. Inhibitory control of correlated intrinsic variability in cortical networks. *Elife*, 5, 2016.

[12] Kristofor D Carlson, Jayram Moorkanikara Nageswaran, Nikil Dutt, and Jeffrey L Krichmar. An efficient automated parameter tuning framework for spiking neural networks. *Front Neurosci*, 8:10, 2014. doi:10.3389/fnins.2014.00010.

[13] P Friedrich, M Vella, A I Gulyás, T F Freund, and S Káli. A flexible, interactive software tool for fitting the parameters of neuronal models. *Frontiers in neuroinformatics*, 8, 2014.

[14] P J Diggle and R J Gratton. Monte carlo methods of inference for implicit statistical models. *J R Stat Soc B Met*, 1984.

[15] F Hartig, J M Calabrese, B Reineking, T Wiegand, and A Huth. Statistical inference for stochastic simulation models–theory and application. *Ecol Lett*, 14(8), 2011.

[16] J Lintusaari, M U Gutmann, R Dutta, S Kaski, and J Corander. Fundamentals and recent developments in approximate bayesian computation. *Syst Biol*, 2016.

[17] Aidan C Daly, David J Gavaghan, Chris Holmes, and Jonathan Cooper. Hodgkin–huxley revisited: reparametrization and identifiability analysis of the classic action potential model with approximate bayesian methods. *Royal Society open science*, 2(12):150499, 2015.

[18] M G B Blum and O François. Non-linear regression models for approximate bayesian computation. *Stat Comput*, 20(1), 2010.

[19] G Papamakarios and I Murray. Fast epsilon-free inference of simulation models with bayesian conditional density estimation. In *Adv in Neur In*, 2017.

[20] N T Carnevale and M L Hines. *The NEURON Book*. Cambridge University Press, 2009.

[21] E Meeds, M Welling, et al. Gps-abc: Gaussian process surrogate approximate bayesian computation. *UAI*, 2014.

[22] J K Pritchard, M T Seielstad, A Perez-Lezaun, and M W Feldman. Population growth of human y chromosomes: a study of y chromosome microsatellites. *Mol Biol Evol*, 16(12), 1999.

[23] P Marjoram, J Molitor, V Plagnol, and S Tavare. Markov chain monte carlo without likelihoods. *Proc Natl Acad Sci U S A*, 100(26), 2003.

[24] E Meeds, R Leenders, and M Welling. Hamiltonian abc. *arXiv preprint arXiv:1503.01916*, 2015.

[25] M A Beaumont, J Cornuet, J Marin, and C P Robert. Adaptive approximate bayesian computation. *Biometrika*, 2009.

[26] F V Bonassi, M West, et al. Sequential monte carlo with adaptive weights for approximate bayesian computation. *Bayesian Anal*, 10(1), 2015.

[27] R Wilkinson. Accelerating abc methods using gaussian processes. In *AISTATS*, 2014.

[28] S N Wood. Statistical inference for noisy nonlinear ecological dynamic systems. *Nature*, 466(7310), 2010.

[29] V M H Ong, D J Nott, M Tran, S A Sisson, and C C Drovandi. Variational bayes with synthetic likelihood. *arXiv:1608.03069*, 2016.

[30] Y Fan, D J Nott, and S A Sisson. Approximate bayesian computation via regression density estimation. *Stat*, 2(1), 2013.

[31] B M Turner and P B Sederberg. A generalized, likelihood-free method for posterior estimation. *Psychonomic Bulletin & Review*, 21(2), 2014.

[32] L F Price, C C Drovandi, A Lee, and David J N. Bayesian synthetic likelihood. *J Comput Graph Stat*, (just-accepted), 2017.

[33] M Beaumont, W Zhang, and D J Balding. Approximate bayesian computation in population genetics. *Genetics*, 162(4), 2002.

[34] G E Hinton and D Van Camp. Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the sixth annual conference on Computational learning theory*, 1993.

[35] A Graves. Practical variational inference for neural networks. In *Adv Neur In*, 2011.

[36] D P Kingma, T Salimans, and M Welling. Neural adaptive sequential monte carlo. In *Variational Dropout and the Local Reparameterization Trick*, pages 2575–2583, 2015.

[37] F Gerhard, M Deger, and W Truccolo. On the stability and dynamics of stochastic spiking neuron models: Nonlinear hawkes process and point process glms. *PLoS Comput Biol*, 13(2), 2017.

[38] K Cho, B Van Merriënboer, C Gulcehre, D Bahdanau, F Bougares, H Schwenk, and Y Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[39] M G B Blum, M A Nunes, Ds Prangle, S A Sisson, et al. A comparative review of dimension reduction methods in approximate bayesian computation. *Statistical Science*, 28(2), 2013.

[40] B Jiang, T Wu, Cs Zheng, and W H Wong. Learning summary statistic for approximate bayesian computation via deep neural network. *arXiv preprint arXiv:1510.02175*, 2015.

[41] J W Pillow, J Shlens, L Paninski, A Sher, A M Litke, E J Chichilnisky, and E P Simoncelli. Spatio-temporal correlations and visual signalling in a complete neuronal population. *Nature*, 454(7207), 2008.

[42] N G Polson, J G Scott, and J Windle. Bayesian inference for logistic models using pólya–gamma latent variables. *J Am Stat Assoc*, 108(504), 2013.

[43] S Linderman, R P Adams, and J W Pillow. Bayesian latent structure discovery from multi-neuron recordings. In *Advances in Neural Information Processing Systems*, 2016.

[44] A L Hodgkin and A F Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *J Physiol*, 117(4), 1952.

[45] M Pospischil, M Toledo-Rodriguez, C Monier, Z Piwkowska, T Bal, Y Frégnac, H Markram, and A Destexhe. Minimal hodgkin-huxley type models for different classes of cortical and thalamic neurons. *Biol Cybern*, 99(4-5), 2008.

[46] E Hay, S Hill, F Schürmann, H Markram, and I Segev. Models of neocortical layer 5b pyramidal cells capturing a wide range of dendritic and perisomatic active properties. *PLoS Comput Biol*, 7(7), 2011.

[47] J Chung, C Gulcehre, K H Cho, and Y Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[48] Marko Järvenpää, Michael U Gutmann, Aki Vehtari, and Pekka Marttinen. Efficient acquisition rules for model-based approximate bayesian computation. *arXiv preprint arXiv:1704.00520*, 2017.

[49] S Gu, Z Ghahramani, and R E Turner. Neural adaptive sequential monte carlo. In *Advances in Neural Information Processing Systems*, pages 2629–2637, 2015.

[50] S W Linderman and S J Gershman. Using computational theory to constrain statistical models of neural data. *bioRxiv*, 2017.

[51] G De Nicolao, G Sparacino, and C Cobelli. Nonparametric input estimation in physiological systems: problems, methods, and case studies. *Automatica*, 33(5), 1997.

Appendix

A Convergence of the log-loss function

Let $K(\mathbf{x}, \mathbf{x}_o)$ be a kernel. We assume that $K \geq 0$ and that $K(\mathbf{x}_o, \mathbf{x}_o) > 0$. Starting from the marginal

$$p(\mathbf{x}) = \int p(\boldsymbol{\theta})p(\mathbf{x}|\boldsymbol{\theta})d\boldsymbol{\theta}$$

we define a weighted version as follows:

$$p_K(\mathbf{x}) := \frac{p(\mathbf{x})K(\mathbf{x}, \mathbf{x}_o)}{\int p(\mathbf{x}')K(\mathbf{x}', \mathbf{x}_o)d\mathbf{x}'} = \frac{1}{Z_K} p(\mathbf{x})K(\mathbf{x}, \mathbf{x}_o),$$

where we assume that the denominator is nonzero and finite. By the law of large numbers:

$$\begin{aligned}
-\frac{1}{N}\sum_n \frac{p(\boldsymbol{\theta}_n)}{\tilde{p}(\boldsymbol{\theta}_n)} K(\mathbf{x}, \mathbf{x}_o) \log q_{\boldsymbol{\phi}}(\boldsymbol{\theta}_n|\mathbf{x}_n) \xrightarrow{a.s.} & \langle -\frac{p(\boldsymbol{\theta})}{\tilde{p}(\boldsymbol{\theta})} K(\mathbf{x}, \mathbf{x}_o) \log q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x})\rangle_{\tilde{p}(\boldsymbol{\theta})p(\mathbf{x}|\boldsymbol{\theta})} \\
&= \langle -K(\mathbf{x}, \mathbf{x}_o) \log q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x})\rangle_{p(\boldsymbol{\theta})p(\mathbf{x}|\boldsymbol{\theta})} \\
&= \langle -K(\mathbf{x}, \mathbf{x}_o) \log q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x})\rangle_{p(\mathbf{x})p(\boldsymbol{\theta}|\mathbf{x})} \\
&= Z_K \langle -\log q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x})\rangle_{p_K(\mathbf{x})p(\boldsymbol{\theta}|\mathbf{x})}
\end{aligned}$$

This expression equals

$$Z_K D_{\mathrm{KL}}\left(p_K(\mathbf{x})p(\boldsymbol{\theta}|\mathbf{x})||p_K(\mathbf{x})q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x})\right) + \textit{const.},$$

where the constant does not depend on $\boldsymbol{\phi}$. Assuming that the family $q_{\boldsymbol{\phi}}$ is sufficiently flexible to model the posterior distribution $p(\boldsymbol{\theta}|\mathbf{x})$, the above quantity is minimised iff the two distributions agree, ie. iff

$$p(\boldsymbol{\theta}|\mathbf{x}) = q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x})$$

almost everywhere, where $p(\mathbf{x})K(\mathbf{x}, \mathbf{x}_o) \neq 0$.

B Details on algorithm and optimisation

Algorithm 1: Training SNPE

```
initialise Bayesian MDN with parameters $\boldsymbol{\phi} = \{\boldsymbol{\phi}_m, \boldsymbol{\phi}_s\}$ and K components
initialise proposal prior $\tilde{p}(\boldsymbol{\theta})^{(1)}$ with prior $p(\boldsymbol{\theta})$
initialise prior over network weights $\pi^{(0)}$ as $\mathcal{N}(\mathbf{w}|\mathbf{0}, \lambda^{-1}\mathbf{I})$
repeat
    for $n = 1 \dots N$ do
        sample $\boldsymbol{\theta}_n \sim \tilde{p}(\boldsymbol{\theta})^{(r)}$
        sample $\mathbf{x}_n \sim p(\mathbf{x}|\boldsymbol{\theta}_n)$
    optional: add components to neural network
    (re)train Bayesian MDN using Eq.2
    set $\hat{p}(\boldsymbol{\theta}|\mathbf{x} = \mathbf{x}_o)^{(r)} := q_{\mathbf{w}}(\boldsymbol{\theta}|\mathbf{x}_o)$ where $\mathbf{w} = \boldsymbol{\phi}_m$
    $\tilde{p}(\boldsymbol{\theta})^{(r+1)} \leftarrow \hat{p}(\boldsymbol{\theta}|\mathbf{x} = \mathbf{x}_o)^{(r)}$
until $\hat{p}(\boldsymbol{\theta}|\mathbf{x} = \mathbf{x}_o)$ has converged
```

The precision of the prior on $\mathbf{w}$ is fixed to $\lambda = 0.01$. We normalise importance weights per round. For optimisation, we use Adam with proposed default settings [1]. We rescale gradients such that their combined norm does not exceed a threshold of 0.1 [2].

[1] D Kingma and Ba J. *Adam: A Method for Stochastic Optimization.* In *ICLR*, 2015.

[2] I Sutskever, O Vinyals, and Q V Le. *Sequence to sequence learning with neural networks.* In *Adv in Neur In*, 2014.

C Continual learning through the D_{KL}-term

The D_{KL}-term in Eq. 2 implements continual learning. In Bayesian inference calculating the posterior given r rounds is equivalent to taking the posterior after $r-1$ rounds as the prior for round r. Translating this to variational inference formulation gives Eq. 2.

The D_{KL}-term is between two Gaussian distributions over weights $\mathbf{w}$:

$$D_{\mathrm{KL}}(\tilde{q}_\phi^{(r)}(\mathbf{w})||\tilde{q}_\phi^{(r-1)}(\mathbf{w})) = \frac{1}{2}\left[\log\frac{|\Sigma_1|}{|\Sigma_2|} + \mathrm{tr}(\Sigma_1^{-1}\Sigma_2) - d + (\mu_1-\mu_2)^T\Sigma_1^{-1}(\mu_1-\mu_2)\right],$$

where d is the dimension of the space (the number of weights) and the parameters of $\tilde{q}^{(r-1)}$ are Σ_1 and μ_1.

A central term in the D_{KL} above is a quadratic penalty on the change in posterior mean (of MDN weights) which is weighted by the posterior precision of round $r-1$. Thus, given that the posterior precision Σ_1^{-1} increases with r, the penalty on the change in means also increases with rounds.

D Details of simulated and neurophysiological data

D.1 Mixture-models

Models SNPE is applied to two distinct mixture-models. The first is a mixture of two Gaussians with a common mean:

$$p(x|\theta) = \alpha\mathcal{N}(x|\theta,\sigma_1^2) + (1-\alpha)\mathcal{N}(x|\theta,\sigma_2^2)$$

The second model is a mixture of two Gaussians, such that

$$p(x|\theta) = \alpha\mathcal{N}(x|\theta,\sigma_1^2) + (1-\alpha)\mathcal{N}(x|-\theta,\sigma_1^2).$$

Inference We set $\alpha = 0.5, \sigma_1 = 1, \sigma_2 = 0.1$. The prior is chosen as $p(\theta) \sim \mathcal{U}(-10, 10)$.

For the first model, we run 6 rounds of SNPE and CDE-LFI with 1000 samples per round. We initialise our SNPE with 2 components.

For the second model we draw 250 samples per round, use 3 rounds, and add a second component to SNPE after the second round.

D.2 Generalised linear model

Model We simulate the activity of a neuron depending on a single set of covariates. Neural activity is subdivided in bins and, within each bin i, spikes are generated according to a Bernoulli observation model:

$$y_i \sim \mathrm{Bern}(\eta(\mathbf{v}_i^\top\boldsymbol{\beta})),$$

where $\mathbf{v}_i^\top\boldsymbol{\beta}$ is the convolution of the white-noise input (represented by $\mathbf{v}_i$) and a linear filter with coefficients $\boldsymbol{\beta}$, and $\eta(\cdot) = \exp(\cdot)/(1+\exp(\cdot))$ is the canonical link function for a Bernoulli GLM.

Inference We apply SNPE to a GLM with a 10-dimensional parameter vector $\boldsymbol{\beta}$. As summary statistics, we use the cross-correlation between input and response, i.e., the sufficient statistics for the generative model.

A Gaussian prior with mean $\mathbf{0}$ and covariance $\boldsymbol{\Sigma}_{\boldsymbol{\beta}} = \sigma^2(\mathbf{F}^\top\mathbf{F})^{-1}$ is used, where $\mathbf{F}$ encourages smoothness, by penalizing the second-order differences in the vector of parameters [51]. SNPE is run for 5 rounds with 5000 GLM simulations each. We only enforce continuity in MDN weights after round 3, when the proposal distribution converged.

To perform PG-MCMC, the generative model is augmented with latent Pólya-Gamma distributed random variables $\boldsymbol{\omega}$, and samples from $\boldsymbol{\omega}$ and $\boldsymbol{\beta}$ are drawn according to the iterative scheme described by Polson and Scott [42]. The set of samples for $\boldsymbol{\beta}$ represents a draw from the posterior distribution of $\boldsymbol{\beta}$ given the data. PG-MCMC procedure uses the same prior as in the SNPE to estimate the posterior.

D.3 Single-compartment Hodgkin-Huxley neuron

Model The single-compartment Hodgkin-Huxley neuron uses channel kinetics as in [45]:

$$C_m\frac{dV}{dt} = g_{\mathrm{leak}}(E_{\mathrm{leak}}-V) + \bar{g}_{\mathrm{Na}}m^3h(E_{\mathrm{Na}}-V) + \bar{g}_{\mathrm{K}}n^4(E_{\mathrm{K}}-V) + \bar{g}_{\mathrm{M}}p(E_{\mathrm{K}}-V) + I_{\mathrm{inj}} + \sigma(t),$$

where C_m is membrane capacitance, V membrane potential, $\bar{g}_c$ density of channels of type c (m, h, n, p) of the channel gating kinetic variables, E_c reversal potential of c, and $\sigma(t)$ intrinsic neural noise. The right hand side is composed of a leak current, a Na-current, a K-current, a slow voltage-dependent K-current responsible for spike-frequency adaptation, and an injected current I_{inj}. Channel gating variables have dynamics fully characterised by the neuron membrane potential, once the kinetic parameters are known.

Inference We illustrate SNPE in a model with 12 parameters $(g_{\text{leak}}, \bar{g}_{\text{Na}}, \bar{g}_{\text{K}}, \bar{g}_{\text{M}}, E_{\text{leak}}, E_{\text{Na}}, E_{\text{K}}, V_T, \sigma, k_{\beta n1}, k_{\beta n2}, \tau_{\text{max}})$ (where $k_{\beta n1}$ and $k_{\beta n2}$ control the kinetics of K channel activation, and σ is the magnitude of the injected Gaussian noise), and 20 voltage features (number of spikes, resting potential, 10 lagged auto-correlations, and the first 8 voltage moments). SNPE is applied to the log absolute value of the parameters $(\log |g_{\text{leak}}|, \log |\bar{g}_{\text{Na}}|...)$.

The prior distribution over the parameters is uniform, and centred around the true parameter values:

$$\theta \sim \mathcal{U}\left(\frac{1}{2}\boldsymbol{\theta}^*, \frac{3}{2}\boldsymbol{\theta}^*\right)$$

SNPE is run for 5 rounds with 5000 Hodgkin-Huxley simulations each, and we fix the posterior to be a mixture of two Gaussians.

D.4 Inference in in-vitro recordings from Allen Cell-type database

We apply the approach to *in vitro* recordings from mouse visual cortex (Allen Cell Type Database), (illustration on cell *464212183* in Fig. 3E-G), and inferred the posterior over 12 parameters, as for the application to the simulated data from the Hodgkin-Huxley neuron. We choose the same prior as in the Hodgkin-Huxley simulated data. SNPE is run for 5 rounds with 5000 Hodgkin-Huxley simulations each, and we fix the posterior to be a mixture of two Gaussians.

D.5 Autapse

Model The autapse model corresponds to a neuron synapsing onto itself with connection strength J, time constant τ, injected current I_{inj} and external white noise source $\eta_t \sim \mathcal{N}(0, 1)$ with $\langle \eta_s, \eta_t \rangle = \delta_{t,s}$:

$$\tau \frac{dr}{dt} = -r + Jr + I_{\text{inj}} + \sigma \eta_t,$$

Using this formula it can be straightforwardly shown that the system has unstable dynamics if $J > 1$.

Inference We apply SNPE to infer two parameters of the autapse model (J, τ), where the feature of interest is the mean across time of the trace. The prior distribution over the parameters is uniform:

$$J \sim \mathcal{U}(0, 2)$$
$$\tau \sim \mathcal{U}(-1, 2.5),$$

where the true parameters are $(0.75, 1)$. We note that the prior allows for the time constant τ to take negative values: while negative time constants do not make physical sense, we note that these are mathematically equivalent to positive time constants where the autapse equation has flipped signs. We draw 1000 samples for each one of 5 rounds.

E Bad features

When sampling from the prior, many parameters sets can lead the Hodgkin-Huxley model to non-spiking behaviour, and therefore features that depend on the presence of spiking, such as latency to first spike, are not defined (Fig. E.1A). In the absence of spiking, the algorithm imputes values to the undefined features, values which are learned during the MDN training. In Fig. E.1B, the imputed value for the latency is close to the mean latency, although we have observed this not to be generally the case.

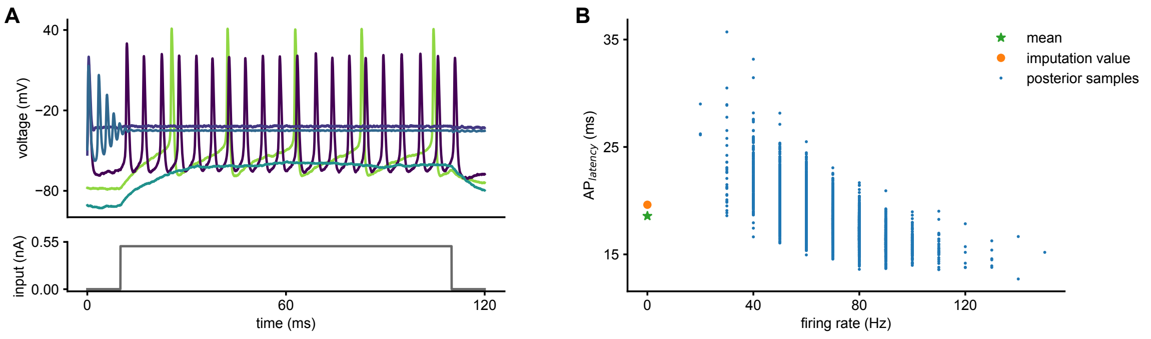


Figure E.1: **We can impute values to missing features. A.** Hodgkin-Huxley simulations with parameters sampled from the prior, with several parameters sets leading to non-spiking behaviour. **B.** Latency to first spike as a function of firing rate, for samples from the posterior distribution. In this case, the imputed value for the latency (in orange) is close to the mean latency (in green).

F Comparison with Sequential Monte-Carlo ABC

SNPE has been tested on problems with 10 or more parameters, whereas most ABC methods (such as SMC-ABC [25]) have addressed problems with fewer parameters, since sampling-based methods require large numbers of simulations. In a GLM with 10 parameters, we observe that our method consistently performs better than SMC-ABC, even when SMC is given orders of magnitude more simulations (Fig. F.1).

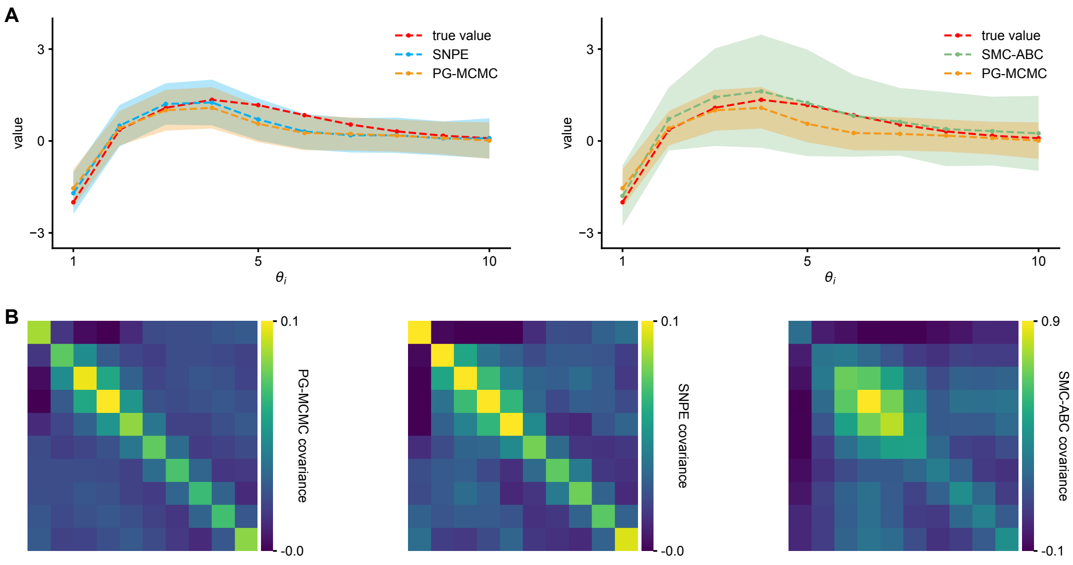


Figure F.1: **Comparison between SNPE and SMC-ABC in a GLM with 10 parameters.** The reference (PG-MCMC) posterior means and variances **A.** and covariance **B.** are recovered well by SNPE after 25000 simulations, whereas sequential Monte-Carlo ABC performs worse with over 4×10^6 simulations. For the application of the SMC-ABC algorithm, we used 1000 particles and a sequence of tolerances $\{\epsilon_i\}_{i=0}^n, \epsilon_i = 15 \times 0.9^i$.

G Supplementary Figures

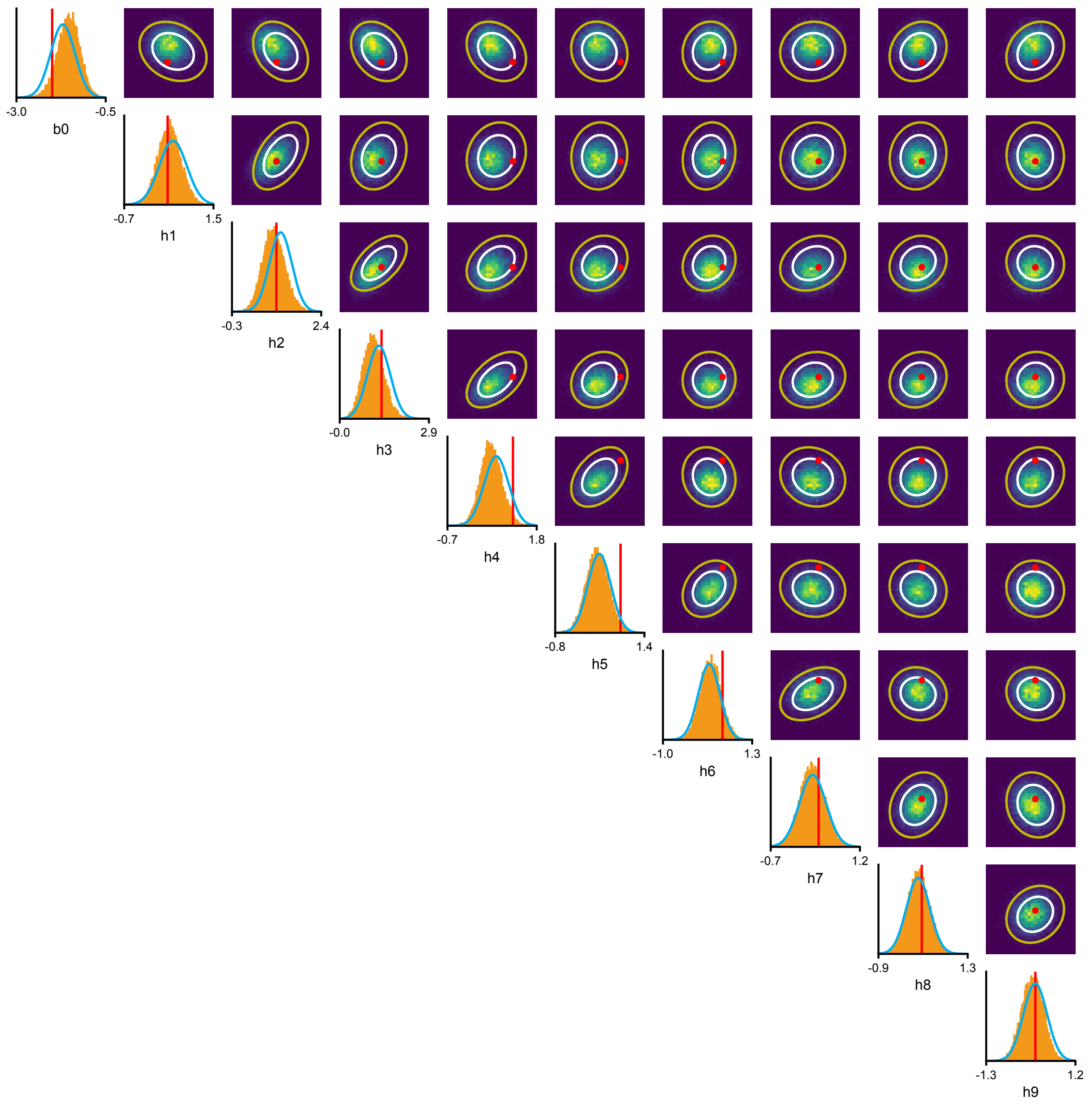


Figure G.1: Full posterior inferred for GLM by SNPE. In red, ground-truth parameter values. 2-D marginals for SNPE (lines) and PG-MCMC (histograms). White and yellow contour lines correspond respectively to 68% and 95% of the mass for SNPE.

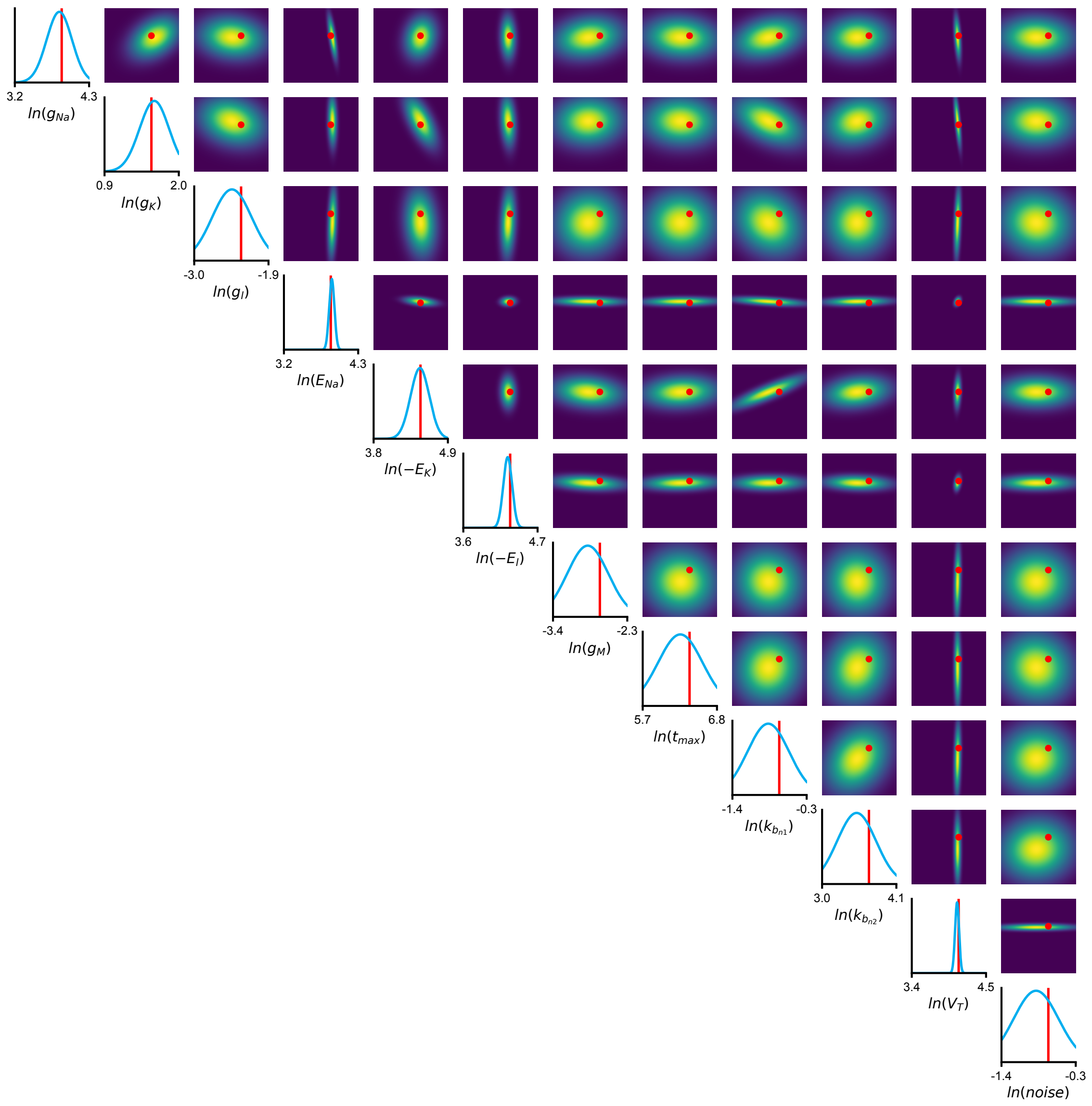


Figure G.2: Full posterior inferred for HH synthetic data

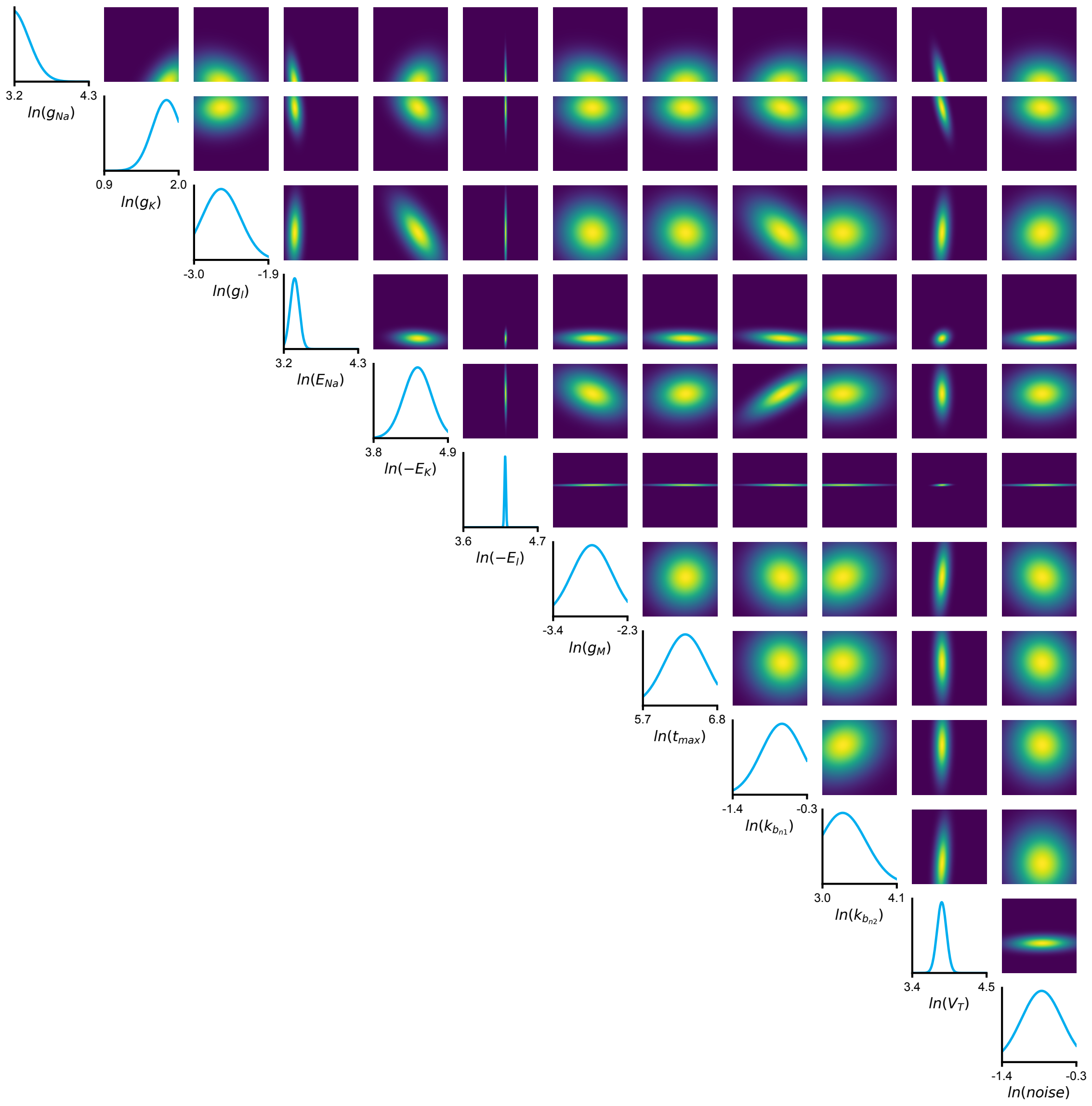


Figure G.3: Full posterior inferred for HH real data

eLife RESEARCH ARTICLE

Training deep neural density estimators to identify mechanistic models of neural dynamics

Pedro J Gonçalves[1,2†*], Jan-Matthis Lueckmann[1,2†*], Michael Deistler[1,3†*], Marcel Nonnenmacher[1,2,4], Kaan Öcal[2,5], Giacomo Bassetto[1,2], Chaitanya Chintaluri[6,7], William F Podlaski[6], Sara A Haddad[8], Tim P Vogels[6,7], David S Greenberg[1,4], Jakob H Macke[1,2,3,9*]

[1]Computational Neuroengineering, Department of Electrical and Computer Engineering, Technical University of Munich, Munich, Germany; [2]Max Planck Research Group Neural Systems Analysis, Center of Advanced European Studies and Research (caesar), Bonn, Germany; [3]Machine Learning in Science, Excellence Cluster Machine Learning, Tübingen University, Tübingen, Germany; [4]Model-Driven Machine Learning, Institute of Coastal Research, Helmholtz Centre Geesthacht, Geesthacht, Germany; [5]Mathematical Institute, University of Bonn, Bonn, Germany; [6]Centre for Neural Circuits and Behaviour, University of Oxford, Oxford, United Kingdom; [7]Institute of Science and Technology Austria, Klosterneuburg, Austria; [8]Max Planck Institute for Brain Research, Frankfurt, Germany; [9]Max Planck Institute for Intelligent Systems, Tübingen, Germany

***For correspondence:** pedro.goncalves@caesar.de (PJG); jan-matthis.lueckmann@tum.de (J-ML); michael.deistler@tum.de (MD); Jakob.Macke@gmail.com (JHM)

[†]These authors contributed equally to this work

Competing interests: The authors declare that no competing interests exist.

Funding: See page 27

Received: 21 February 2020
Accepted: 16 September 2020
Published: 17 September 2020

Reviewing editor: Timothy O'Leary, University of Cambridge, United Kingdom

Abstract Mechanistic modeling in neuroscience aims to explain observed phenomena in terms of underlying causes. However, determining which model parameters agree with complex and stochastic neural data presents a significant challenge. We address this challenge with a machine learning tool which uses deep neural density estimators—trained using model simulations—to carry out Bayesian inference and retrieve the full space of parameters compatible with raw data or selected data features. Our method is scalable in parameters and data features and can rapidly analyze new data after initial training. We demonstrate the power and flexibility of our approach on receptive fields, ion channels, and Hodgkin–Huxley models. We also characterize the space of circuit configurations giving rise to rhythmic activity in the crustacean stomatogastric ganglion, and use these results to derive hypotheses for underlying compensation mechanisms. Our approach will help close the gap between data-driven and theory-driven models of neural dynamics.

Introduction

New experimental technologies allow us to observe neurons, networks, brain regions, and entire systems at unprecedented scale and resolution, but using these data to understand how behavior arises from neural processes remains a challenge. To test our understanding of a phenomenon, we often take to rebuilding it in the form of a computational model that incorporates the mechanisms we believe to be at play, based on scientific knowledge, intuition, and hypotheses about the components of a system and the laws governing their relationships. The goal of such mechanistic models is to investigate whether a proposed mechanism can explain experimental data, uncover details that may have been missed, inspire new experiments, and eventually provide insights into the inner workings of an observed neural or behavioral phenomenon (***Herz et al., 2006***; ***Gerstner et al., 2012***; ***O'Leary et al., 2015***; ***Baker et al., 2018***). Examples

eLife digest Computational neuroscientists use mathematical models built on observational data to investigate what's happening in the brain. Models can simulate brain activity from the behavior of a single neuron right through to the patterns of collective activity in whole neural networks. Collecting the experimental data is the first step, then the challenge becomes deciding which computer models best represent the data and can explain the underlying causes of how the brain behaves.

Researchers usually find the right model for their data through trial and error. This involves tweaking a model's parameters until the model can reproduce the data of interest. But this process is laborious and not systematic. Moreover, with the ever-increasing complexity of both data and computer models in neuroscience, the old-school approach of building models is starting to show its limitations.

Now, Gonçalves, Lueckmann, Deistler et al. have designed an algorithm that makes it easier for researchers to fit mathematical models to experimental data. First, the algorithm trains an artificial neural network to predict which models are compatible with simulated data. After initial training, the method can rapidly be applied to either raw experimental data or selected data features. The algorithm then returns the models that generate the best match.

This newly developed machine learning tool was able to automatically identify models which can replicate the observed data from a diverse set of neuroscience problems. Importantly, further experiments showed that this new approach can be scaled up to complex mechanisms, such as how a neural network in crabs maintains its rhythm of activity. This tool could be applied to a wide range of computational investigations in neuroscience and other fields of biology, which may help bridge the gap between 'data-driven' and 'theory-driven' approaches.

for such a symbiotic relationship between model and experiments range from the now classical work of ***Hodgkin and Huxley, 1952***, to population models investigating rules of connectivity, plasticity and network dynamics (***van Vreeswijk and Sompolinsky, 1996***; ***Prinz et al., 2004***; ***Vogels et al., 2005***; ***Potjans and Diesmann, 2014***; ***Litwin-Kumar and Doiron, 2012***), network models of inter-area interactions (***Sporns, 2014***; ***Bassett et al., 2018***), and models of decision making (***Gold and Shadlen, 2007***; ***Wang, 2008***).

A crucial step in building a model is adjusting its free parameters to be consistent with experimental observations. This is essential both for investigating whether the model agrees with reality and for gaining insight into processes which cannot be measured experimentally. For some models in neuroscience, it is possible to identify the relevant parameter regimes from careful mathematical analysis of the model equations. But as the complexity of both neural data and neural models increases, it becomes very difficult to find well-fitting parameters by inspection, and *automated* identification of data-consistent parameters is required.

Furthermore, to understand how a model quantitatively explains data, it is necessary to find not only the *best*, but *all* parameter settings consistent with experimental observations. This is especially important when modeling neural data, where highly variable observations can lead to broad ranges of data-consistent parameters. Moreover, many models in biology are inherently robust to some perturbations of parameters, but highly sensitive to others (***Gutenkunst et al., 2007***; ***O'Leary et al., 2015***), for example because of processes such as homeostatic regulation. For these systems, identifying the full range of data-consistent parameters can reveal how multiple distinct parameter settings give rise to the same model behavior (***Foster et al., 1993***; ***Prinz et al., 2004***; ***Achard and De Schutter, 2006***; ***Alonso and Marder, 2019***). Yet, despite the clear benefits of mechanistic models in providing scientific insight, identifying their parameters given data remains a challenging open problem that demands new algorithmic strategies.

The gold standard for automated parameter identification is *statistical inference*, which uses the likelihood $p(\mathbf{x}|\theta)$ to quantify the match between parameters θ and data $\mathbf{x}$. Likelihoods can be efficiently computed for purely statistical models commonly used in neuroscience (***Truccolo et al., 2005***; ***Schneidman et al., 2006***; ***Pillow et al., 2008***; ***Yu et al., 2009***; ***Macke et al., 2011***; ***Cunningham and Yu, 2014***; ***Pandarinath et al., 2018***), but are computationally intractable for most mechanistic models. Mechanistic models are designed to reflect knowledge about biological mechanisms, and not necessarily to be amenable to efficient inference: many mechanistic models are defined implicitly through stochastic computer simulations (e.g. a simulation of a network of spiking neurons), and likelihood calculation would require the ability to integrate over all potential paths through the simulator code. Similarly, a common goal of mechanistic modeling is to capture selected summary features of the data (e.g. a certain firing rate, bursting behavior, etc. . .), *not* the full dataset in all its details. The same feature (such as a particular average firing rate) can be produced by infinitely many realizations of the simulated process (such as a time-series of membrane potential). This makes it impractical to compute likelihoods, as one would have to average over all possible realizations which produce the same output.

Since the toolkit of (likelihood-based) statistical inference is inaccessible for mechanistic models, parameters are typically tuned ad-hoc (often through laborious, and subjective, trial-and-error), or by computationally expensive parameter search: a large set of models is generated, and grid search (***Prinz et al., 2003***; ***Tomm et al., 2011***; ***Stringer et al., 2016***) or a genetic algorithm (***Druckmann et al., 2007***; ***Hay et al., 2011***; ***Rossant et al., 2011***; ***Van Geit et al., 2016***) is used to filter out simulations which do not match the data. However, these approaches require the user to define a heuristic rejection criterion on which simulations to keep (which can be challenging when observations have many dimensions or multiple units of measurement), and typically end up discarding most simulations. Furthermore, they lack the advantages of statistical inference, which provides principled approaches for handling variability, quantifying uncertainty, incorporating prior knowledge and integrating multiple data sources. Approximate Bayesian Computation (ABC) (***Beaumont et al., 2002***; ***Marjoram et al., 2003***; ***Sisson et al., 2007***) is a parameter-search technique which aims to perform statistical inference, but still requires definition of a rejection criterion and struggles in high-dimensional problems. Thus, computational neuroscientists face a dilemma: either create carefully designed, highly interpretable mechanistic models (but rely on ad-hoc parameter tuning), or resort to purely statistical models offering sophisticated parameter inference but limited mechanistic insight.

Here, we propose a new approach using machine learning to combine the advantages of mechanistic and statistical modeling. We present SNPE (Sequential Neural Posterior Estimation), a tool that makes it possible to perform Bayesian inference on mechanistic models in neuroscience without requiring access to likelihoods. SNPE identifies all mechanistic model parameters consistent with observed experimental data (or summary features). It builds on recent advances in simulation-based Bayesian inference (***Papamakarios and Murray, 2016***; ***Lueckmann et al., 2017***; ***Greenberg et al., 2019***; ***Cranmer et al., 2020***): given observed experimental data (or summary features) $\mathbf{x}_o$, and a mechanistic model with parameters θ, it expresses both prior knowledge and the range of data-compatible parameters through probability distributions. SNPE returns a posterior distribution $p(\theta|\mathbf{x}_o)$ which is high for parameters θ consistent with both the data $\mathbf{x}_o$ and prior knowledge, but approaches zero for θ inconsistent with either (***Figure 1***).

Similar to parameter search methods, SNPE uses simulations instead of likelihood calculations, but instead of filtering out simulations, it uses *all* simulations to train a multilayer artificial neural network to identify admissible parameters (***Figure 1***). By incorporating modern deep neural networks for conditional density estimation (***Rezende and Mohamed, 2015***; ***Papamakarios et al., 2017***), it can capture the full *distribution* of parameters consistent with the data, even when this distribution has multiple peaks or lies on curved manifolds. Critically, SNPE decouples the design of the model and design of the inference approach, giving the investigator maximal flexibility to design and modify mechanistic models. Our method makes minimal assumptions about the model or its implementation, and can for example also be applied to non-differentiable models, such as networks of spiking neurons. Its only requirement is that one can run model simulations for different parameters, and collect the resulting synthetic data or summary features of interest.

While the theoretical foundations of SNPE were originally developed and tested using simple inference problems on small models (***Papamakarios and Murray, 2016***; ***Lueckmann et al., 2017***; ***Greenberg et al., 2019***), here we show that SNPE can scale to complex mechanistic models in neuroscience, provide an accessible and powerful implementation, and develop validation and visualization techniques for exploring the derived posteriors. We illustrate SNPE using mechanistic models expressing key neuroscientific concepts: beginning with a simple neural encoding problem with a known solution, we progress to more complex data types, large datasets and many-parameter

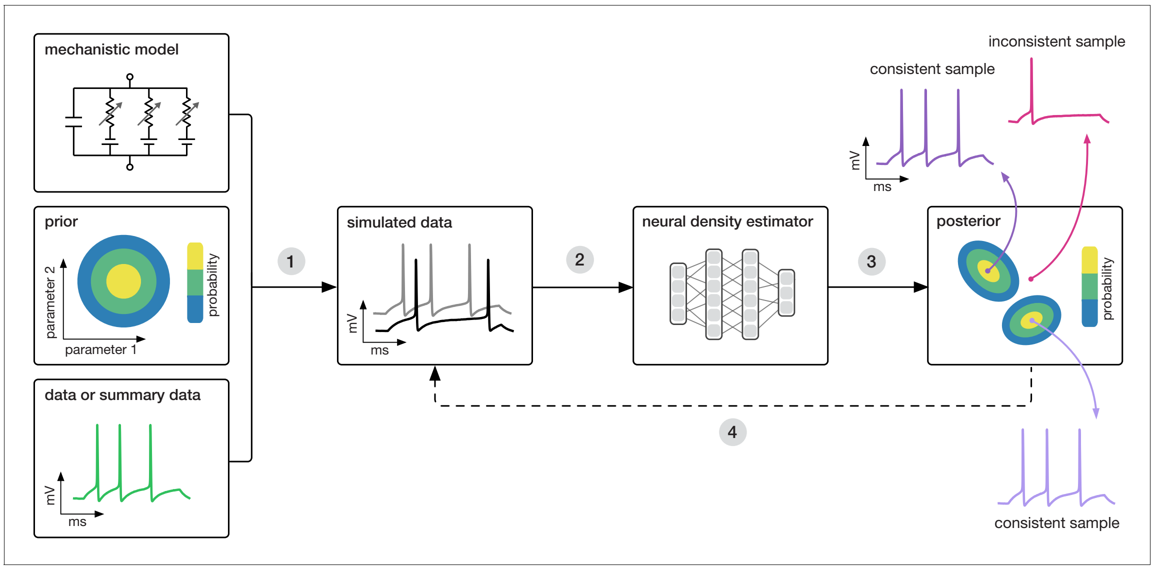


Figure 1. Goal: algorithmically identify mechanistic models which are consistent with data. Our algorithm (SNPE) takes three inputs: a candidate mechanistic model, prior knowledge or constraints on model parameters, and data (or summary statistics). SNPE proceeds by (1) sampling parameters from the prior and simulating synthetic datasets from these parameters, and (2) using a deep density estimation neural network to learn the (probabilistic) association between data (or data features) and underlying parameters, that is to learn statistical inference from simulated data. (3) This density estimation network is then applied to empirical data to derive the full space of parameters consistent with the data and the prior, that is, the posterior distribution. High posterior probability is assigned to parameters which are consistent with both the data and the prior, low probability to inconsistent parameters. (4) If needed, an initial estimate of the posterior can be used to adaptively guide further simulations to produce data-consistent results.

models inaccessible to previous methods. We estimate visual receptive fields using many data features, demonstrate rapid inference of ion channel properties from high-throughput voltage-clamp protocols, and show how Hodgkin–Huxley models are more tightly constrained by increasing numbers of data features. Finally, we showcase the power of SNPE by using it to identify the parameters of a network model which can explain an experimentally observed pyloric rhythm in the stomatogastric ganglion (***Prinz et al., 2004***)–in contrast to previous approaches, SNPE allows us to search over the full space of both single-neuron and synaptic parameters, allowing us to study the geometry of the parameter space, as well as to provide new hypotheses for which compensation mechanisms might be at play.

Results

Training neural networks to perform Bayesian inference without likelihood evaluations

SNPE performs Bayesian inference on mechanistic models using only model-simulations, without requiring likelihood evaluations. It requires three inputs: a model (i.e. computer code to simulate data from parameters), prior knowledge or constraints on parameters, and data (outputs from the model or the real system it describes, ***Figure 1***). SNPE runs simulations for a range of parameter values, and trains an artificial neural network to map any simulation result onto a range of possible parameters. Importantly, a network trained to maximize log-probability (of parameters given simulation results) will learn to approximate the posterior distribution as given by Bayes rule (***Papamakarios and Murray, 2016***) (see Materials and methods for details, ***Figure 1***). After training on *simulated* data with known model parameters, SNPE can perform Bayesian inference of unknown parameters for *empirical* data. This approach to Bayesian inference never requires evaluating likelihoods. SNPE's efficiency can be further improved by using the running estimate of the posterior distribution to guide further simulations toward data-compatible regions of the parameter space (**Papamakarios and Murray, 2016**; **Lueckmann et al., 2017**; **Greenberg et al., 2019**). Below, we apply SNPE to a range of stochastic models in neuroscience.

Estimating stimulus-selectivity in linear-nonlinear encoding models

We first illustrate SNPE on linear-nonlinear (LN) encoding models, a special case of generalized linear models (GLMs). These are simple, commonly used phenomenological models for which likelihood-based parameter estimation is feasible (**Brown et al., 1998**; **Paninski, 2004**; **Pillow, 2007**; **Gerwinn et al., 2010**; **Polson et al., 2013**; **Pillow and Scott, 2012**), and which can be used to validate the accuracy of our approach, before applying SNPE to more complex models for which the likelihood is unavailable. We will show that SNPE returns the correct posterior distribution over parameters, that it can cope with high-dimensional observation data, that it can recover multiple solutions to parameter inference problems, and that it is substantially more simulation efficient than conventional rejection-based ABC methods.

An LN model describes how a neuron's firing rate is modulated by a sensory stimulus through a linear filter θ, often referred to as the *receptive field* (**Pillow et al., 2005**; **Chichilnisky, 2001**). We first considered a model of a retinal ganglion cell (RGC) driven by full-field flicker (**Figure 2a**). A statistic that is often used to characterize such a neuron is the *spike-triggered average* (STA) (**Figure 2a**, right). We therefore used the STA, as well as the firing rate of the neuron, as input $\mathbf{x}_o$ to SNPE. (Note that, in the limit of infinite data, and for white noise stimuli, the STA will converge to the receptive field [**Paninski, 2004**]–for finite, and non-white data, the two will in general be different.) Starting with random receptive fields θ, we generated synthetic spike trains and calculated STAs from them (**Figure 2b**). We then trained a neural conditional density estimator to recover the receptive fields from the STAs and firing rates (**Figure 2c**). This allowed us to estimate the posterior distribution over receptive fields, that is to estimate which receptive fields are consistent with the data (and prior) (**Figure 2c**). For LN models, likelihood-based inference is possible, allowing us to validate the SNPE posterior by comparing it to a reference posterior obtained via Markov Chain Monte Carlo (MCMC) sampling (**Polson et al., 2013**; **Pillow and Scott, 2012**). We found that SNPE accurately estimates the posterior distribution (**Appendix 1—figure 1** and **Appendix 1—figure 2**), and substantially outperforms Sequential Monte Carlo (SMC) ABC methods (**Sisson et al., 2007**; **Beaumont et al., 2009**; **Figure 2d**). If SNPE works correctly, its posterior mean filter will match that of the reference posterior – however, it is not to be expected that either of them precisely matches the ground-truth filter (**Figure 2c** and **Appendix 1—figure 1**): In the presence of finite sampling and stochasticity, multiple different filters could have plausibly given rise to the observed data. A properly inferred posterior will reflect this uncertainty, and include the true filters as one of many plausible explanations of the data (but not necessarily as the 'mean' of all plausible explanations) (**Appendix 1—figure 2**). Increasing the number of Bernoulli samples in the observed data leads to progressively tighter posteriors, with posterior samples closer to the true filter (**Appendix 1—figure 3**). Furthermore, SNPE closely agrees with the MCMC reference solution in all these cases, further emphasizing the correctness of the posteriors inferred with SNPE.

As a more challenging problem, we inferred the receptive field of a neuron in primary visual cortex (V1) (**Niell and Stryker, 2008**; **Dyballa et al., 2018**). Using a model composed of a bias (related to the spontaneous firing rate) and a Gabor function with eight parameters (**Jones and Palmer, 1987**) describing the receptive field's location, shape and strength, we simulated responses to 5 min random noise movies of 41 $\times$ 41 pixels, such that the STA is high-dimensional, with a total of 1681 dimensions (**Figure 2e**). This problem admits multiple solutions (as e.g. rotating the receptive field by 180˚). As a result, the posterior distribution has multiple peaks ('modes'). Starting from a simulation result $\mathbf{x}_o$ with known parameters, we used SNPE to estimate the posterior distribution $p(\theta|\mathbf{x}_o)$. To deal with the high-dimensional data $\mathbf{x}_o$ in this problem, we used a convolutional neural network (CNN), as this architecture excels at learning relevant features from image data (**Krizhevsky et al., 2012**; **Simonyan and Zisserman, 2015**). To deal with the multiple peaks in the posterior, we fed the CNN's output into a mixture density network (MDN) (**Bishop, 1994**), which can learn to assign probability distributions with multiple peaks as a function of its inputs (details in Materials and methods). Using this strategy, SNPE was able to infer a posterior distribution that tightly enclosed the ground

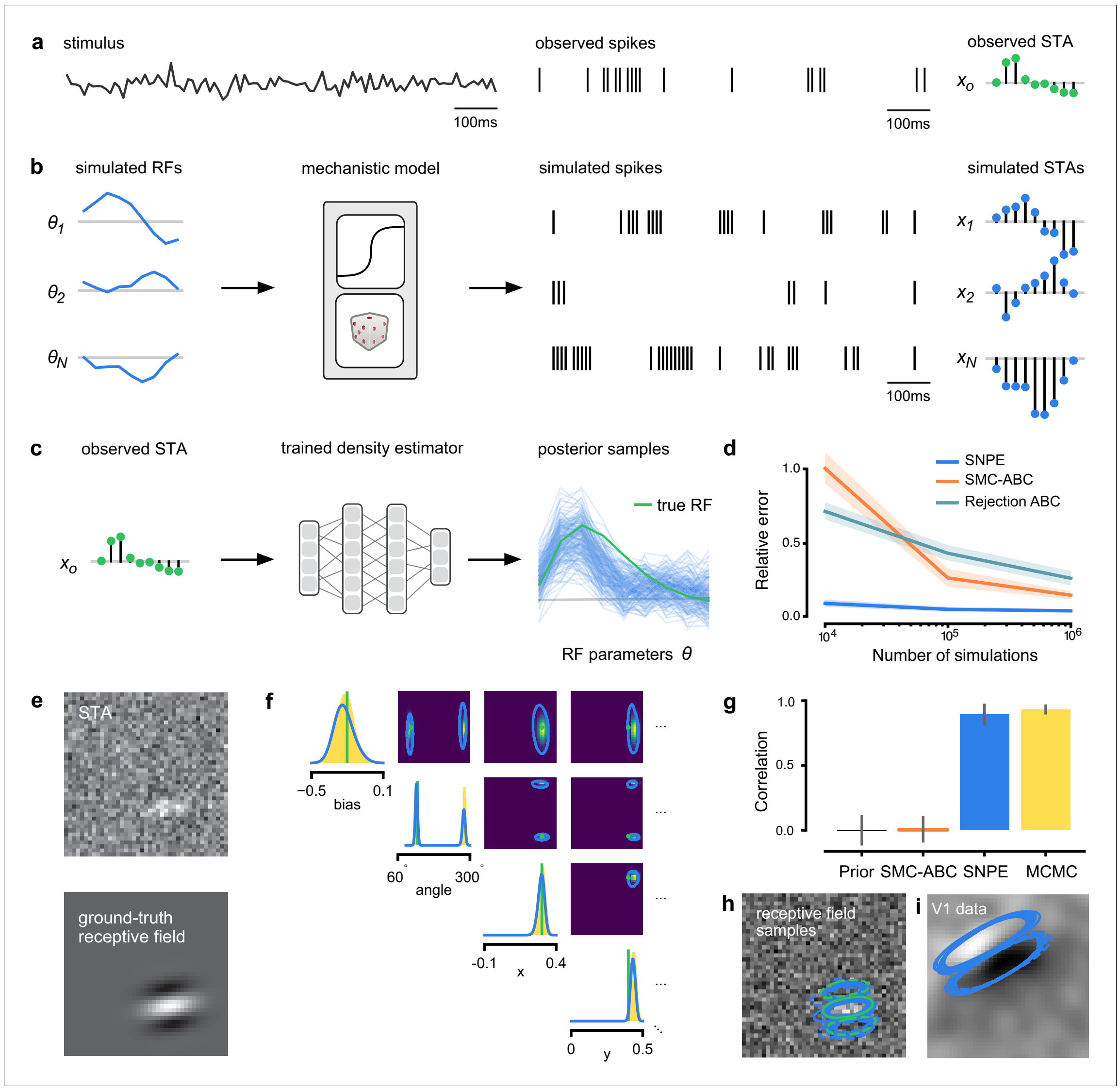


Figure 2. Estimating receptive fields in linear-nonlinear models of single neurons with statistical inference. (**a**) Schematic of a time-varying stimulus, associated observed spike train and resulting spike-triggered average (STA) (**b**) Sequential Neural Posterior Estimation (SNPE) proceeds by first randomly generating simulated receptive fields θ, and using the mechanistic model (here an LN model) to generate simulated spike trains and simulated STAs. (**c**) These simulated STAs and receptive fields are then used to train a deep neural density estimator to identify the distribution of receptive fields consistent with a given observed STA $\mathbf{x}_o$. (**d**) Relative error in posterior estimation of SNPE and alternative methods (mean and 95% CI; 0 corresponds to perfect estimation, one to prior-level, details in Materials and methods). (**e**) Example of spatial receptive field. We simulated responses and an STA of an LN-model with oriented receptive field. (**f**) We used SNPE to recover the distribution of receptive-field parameters. Univariate and pairwise marginals for four parameters of the spatial filter (MCMC, yellow histograms; SNPE, blue lines; ground truth, green; full posterior in ***Appendix 1—figure 4***). Non-identifiabilities of the Gabor parameterization lead to multimodal posteriors. (**g**) Average correlation (± SD) between ground-truth receptive field and receptive field samples from posteriors inferred with SMC-ABC, SNPE, and MCMC (which provides an upper

Figure 2 continued on next page

Figure 2 continued

bound given the inherent stochasticity of the data). (**h**) Posterior samples from SNPE posterior (SNPE, blue) compared to ground-truth receptive field (green; see panel (**e**)), overlaid on STA. (**i**) Posterior samples for V1 data; full posterior in ***Appendix 1—figure 6***.

truth simulation parameters which generated the original simulated data $\mathbf{x}_o$, and matched a reference MCMC posterior (***Figure 2f***, posterior over all parameters in ***Appendix 1—figure 4***). For this challenging estimation problem with high-dimensional summary features, an SMC-ABC algorithm with the same simulation-budget failed to identify the correct receptive fields (***Figure 2g***) and posterior distributions (***Appendix 1—figure 5***). We also applied this approach to electrophysiological data from a V1 cell (***Dyballa et al., 2018***), identifying a sine-shaped Gabor receptive field consistent with the original spike-triggered average (***Figure 2i***; posterior distribution in ***Appendix 1—figure 6***).

Functional diversity of ion channels: efficient high-throughput inference

We next show how SNPE can be efficiently applied to estimation problems in which we want to identify a large number of models for different observations in a database. We considered a flexible model of ion channels (***Destexhe and Huguenard, 2000***), which we here refer to as the *Omnimodel*. This model uses eight parameters to describe how the dynamics of currents through non-inactivating potassium channels depend on membrane voltage (***Figure 3a***). For various choices of its parameters θ, it can capture 350 specific models in publications describing this channel type, cataloged in the IonChannelGenealogy (ICG) database (***Podlaski et al., 2017***). We aimed to identify these ion channel parameters θ for each ICG model, based on 11 features of the model's response to a sequence of five noisy voltage clamp protocols, resulting in a total of 55 different characteristic features per model (***Figure 3b***, see Materials and methods for details).

Because this model's output is a typical format for functional characterization of ion channels both in simulations (***Podlaski et al., 2017***) and in high-throughput electrophysiological experiments (***Dunlop et al., 2008***; ***Suk et al., 2019***; ***Ranjan et al., 2019***), the ability to rapidly infer different parameters for many separate experiments is advantageous. Existing fitting approaches based on numerical optimization (***Destexhe and Huguenard, 2000***; ***Ranjan et al., 2019***) must repeat all computations anew for a new experiment or data point (***Figure 3c***). However, for SNPE the only heavy computational tasks are carrying out simulations to generate training data, and training the neural network. We therefore reasoned that by training a network once using a large number of simulations, we could subsequently carry out rapid 'amortized' parameter inference on new data using a single pass through the network (***Figure 3d***; ***Speiser et al., 2017***; ***Webb et al., 2018***). To test this idea, we used SNPE to train a neural network to infer the posterior from any data $\mathbf{x}$. To generate training data, we carried out 1 million Omnimodel simulations, with parameters randomly chosen across ranges large enough to capture the models in the ICG database (***Podlaski et al., 2017***). SNPE was run using a single round, that is, it learned to perform inference for all data from the prior (rather than a specific observed datum). Generating these simulations took around 1000 CPU-hours and training the network 150 CPU-hours, but afterwards a full posterior distribution could be inferred for new data in less than 10 ms.

As a first test, SNPE was run on simulation data, generated by a previously published model of a non-inactivating potassium channel (***McTavish et al., 2012***; ***Figure 3b***). Simulations of the Omnimodel using parameter sets sampled from the obtained posterior distribution (***Figure 3e***) closely resembled the input data on which the SNPE-based inference had been carried out, while simulations using 'outlier' parameter sets with low probability under the posterior generated current responses that were markedly different from the data $\mathbf{x}_o$ (***Figure 3f***). Taking advantage of SNPE's capability for rapid amortized inference, we further evaluated its performance on all 350 non-inactivating potassium channel models in ICG. In each case, we carried out a simulation to generate initial data from the original ICG model, used SNPE to calculate the posterior given the Omnimodel, and then generated a new simulation $\mathbf{x}$ using parameters sampled from the posterior (***Figure 3f***). This resulted in high correlation between the original ICG model response and the Omnimodel response, in every case (>0.98 for more than 90% of models, see ***Appendix 1—figure 7***). However, this approach was not able to capture all traces perfectly, as for example it failed to capture the shape of the onset of the bottom right model in ***Figure 3g***. Additional analysis of this example revealed

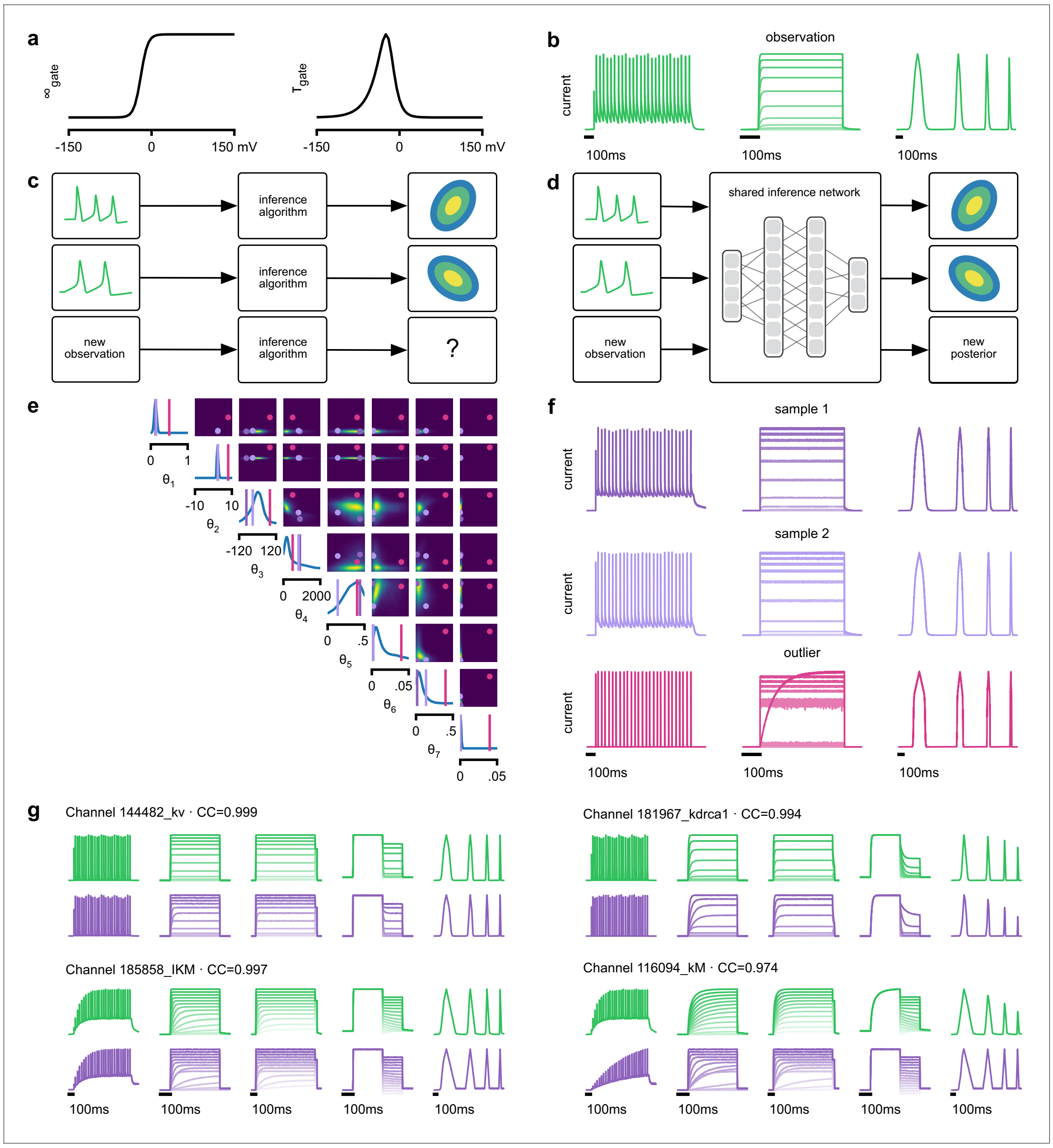


Figure 3. Inference on a database of ion-channel models. (a) We perform inference over the parameters of non-inactivating potassium channel models. Channel kinetics are described by steady-state activation curves, ∞_{gate}, and time-constant curves, τ_{gate}. (b) Observation generated from a channel model from ICG database: normalized current responses to three (out of five) voltage-clamp protocols (action potentials, activation, and ramping). Details in ***Podlaski et al., 2017***. (c) Classical approach to parameter identification: inference is optimized on each datum separately, requiring new computations for each new datum. (d) Amortized inference: an inference network is learned which can be applied to multiple data, enabling rapid inference on new data. (e) Posterior distribution over eight model parameters, θ_1 to θ_8. Ground truth parameters in green, high-probability parameters

Figure 3 continued on next page

Figure 3 continued

in purple, low-probability parameters in magenta. (**f**) Traces obtained by sampling from the posterior in (**e**). Purple: traces sampled from posterior, that is, with high posterior probability. Magenta: trace from parameters with low probability. (**g**) Observations (green) and traces generated by posterior samples (purple) for four models from the database.

that this example is not a failure of SNPE, but rather a limitation of the Omnimodel: in particular, directly fitting the steady-state activation and time-constant curves on this specific example yielded no further quantitative or qualitative improvement, suggesting that the limitation is in the model, not the fit. Thus, SNPE can be used to reveal limitations of candidate models and aid the development of more verisimilar mechanistic models.

Calculating the posterior for all 350 ICG models took only a few seconds, and was fully automated, that is, did not require user interactions. These results show how SNPE allows fast and accurate identification of biophysical model parameters on new data, and how SNPE can be deployed for applications requiring rapid automated inference, such as high-throughput screening-assays, closed-loop paradigms (e.g. for adaptive experimental manipulations or stimulus-selection [***Kleinegesse and Gutmann, 2019***]), or interactive software tools.

Hodgkin–Huxley model: stronger constraints from additional data features

The Hodgkin–Huxley (HH) model (***Hodgkin and Huxley, 1952***) of action potential generation through ion channel dynamics is a highly influential mechanistic model in neuroscience. A number of algorithms have been proposed for fitting HH models to electrophysiological data (***Prinz et al., 2003***; ***Huys et al., 2006***; ***Pospischil et al., 2008***; ***Rossant et al., 2011***; ***Meliza et al., 2014***; ***Van Geit et al., 2016***; ***Ben-Shalom et al., 2019***), but (with the exception of ***Daly et al., 2015***) these approaches do not attempt to estimate the full posterior. Given the central importance of the HH model in neuroscience, we sought to test how SNPE would cope with this challenging non-linear model.

As previous approaches for HH models concentrated on reproducing specified features (e.g. the number of spikes, [***Pospischil et al., 2008***]), we also sought to determine how various features provide different constraints. We considered the problem of inferring eight biophysical parameters in a HH single-compartment model, describing voltage-dependent sodium and potassium conductances and other intrinsic membrane properties, including neural noise, making the model stochastic by nature (***Figure 4a***, left). We simulated the neuron's voltage response to the injection of a square wave of depolarizing current, and defined the model output $\mathbf{x}$ used for inference as the number of evoked action potentials along with six additional features of the voltage response (***Figure 4a***, right, details in Materials and methods). We first applied SNPE to observed data $\mathbf{x}_o$ created by simulation from the model, calculating the posterior distribution using all seven features in the observed data (***Figure 4b***). The posterior contained the ground truth parameters in a high probability-region, as in previous applications, indicating the consistency of parameter identification. The variance of the posterior was narrower for some parameters than for others, indicating that the seven data features constrain some parameters strongly (such as the potassium conductance), but others only weakly (such as the adaptation time constant). Additional simulations with parameters sampled from the posterior closely resembled the observed data $\mathbf{x}_o$, in terms of both the raw membrane voltage over time and the seven data features (***Figure 4c***, purple and green). Parameters with low posterior probability (outliers) generated simulations that markedly differed from $\mathbf{x}_o$ (***Figure 4c***, magenta).

Genetic algorithms are commonly used to fit parameters of deterministic biophysical models (***Druckmann et al., 2007***; ***Hay et al., 2011***; ***Van Geit et al., 2016***; ***Gouwens et al., 2018***). While genetic algorithms can also return multiple data-compatible parameters, they do not perform inference (i.e. find the posterior distribution), and their outputs depend strongly on user-defined goodness-of-fit criteria. When comparing a state-of-the-art genetic algorithm (Indicator Based Evolutionary Algorithm, IBEA, [***Bleuler et al., 2003***; ***Zitzler and Künzli, 2004***; ***Van Geit et al., 2016***]) to SNPE, we found that the parameter-settings favored by IBEA produced simulations whose summary features were as similar to the observed data as those obtained by SNPE high-probability samples (***Appendix 1—figure 10***). However, high-scoring IBEA parameters were concentrated in

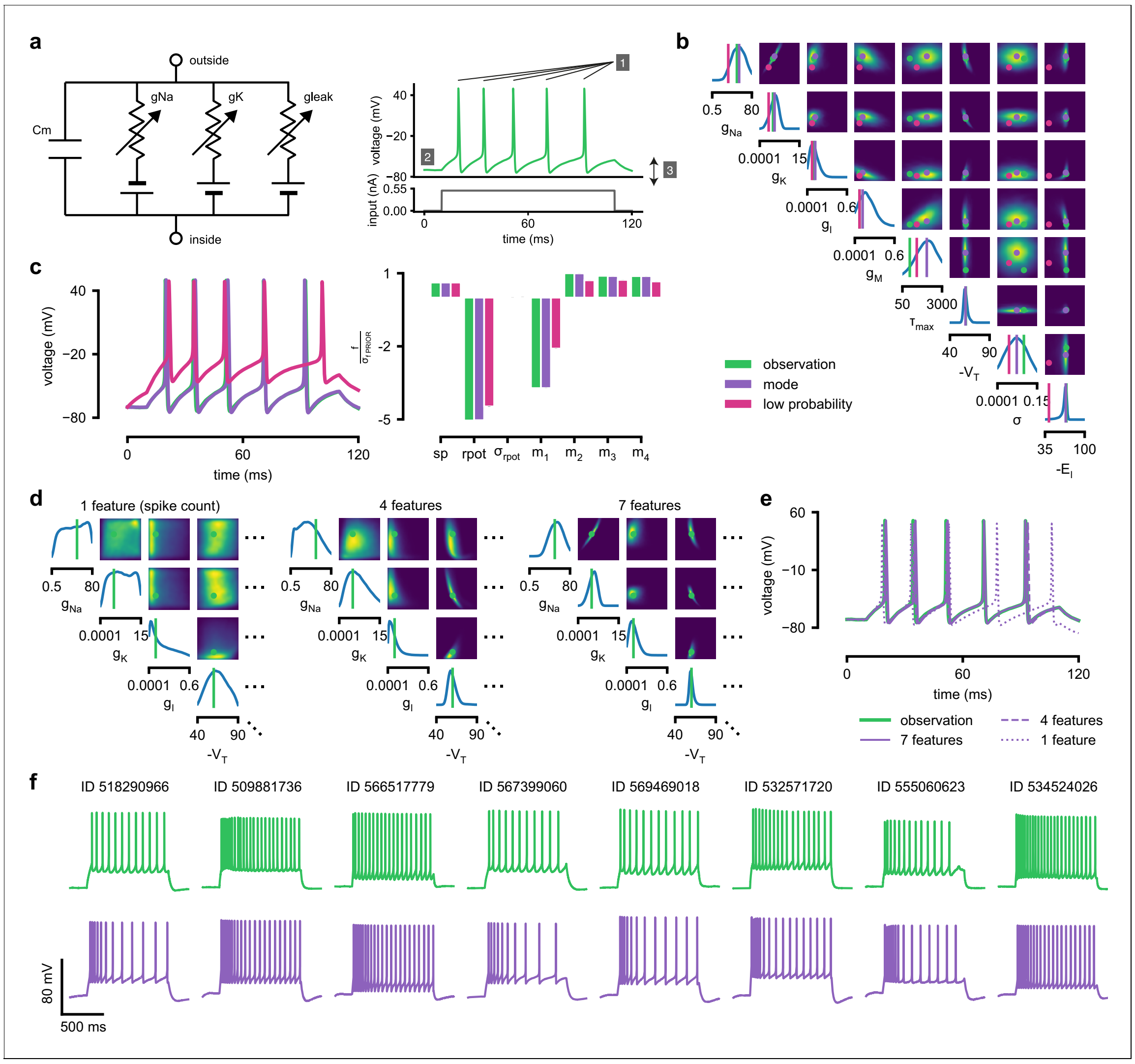


Figure 4. Inference for single compartment Hodgkin–Huxley model. (**a**) Circuit diagram describing the Hodgkin–Huxley model (left), and simulated voltage-trace given a current input (right). Three out of 7 voltage features are depicted: (1) number of spikes, (2) mean resting potential, and (3) standard deviation of the pre-stimulus resting potential. (**b**) Inferred posterior for 8 parameters given seven voltage features. Ground truth parameters in green, high-probability parameters in purple, low-probability parameters in magenta. (**c**) Traces (left) and associated features f (right) for the desired output (observation), the mode of the inferred posterior, and a sample with low posterior probability. The voltage features are: number of spikes sp, mean resting potential $rpot$, standard deviation of the resting potential σ_{rpot}, and the first four voltage moments, mean m_1, standard deviation m_2, skewness m_3 and kurtosis m_4. Each value plotted is the mean feature ± standard deviation across 100 simulations with the same parameter set. Each feature is normalized by $\sigma_{f\text{PRIOR}}$, the standard deviation of the respective feature of simulations sampled from the prior. (**d**) Partial view of the inferred posteriors (4 out of 8 parameters) given 1, 4 and 7 features (full posteriors over eight parameters in ***Appendix 1—figure 8***). (**e**) Traces for posterior modes given 1, 4 and 7 features. Increasing the number of features leads to posterior traces that are closer to the observed data. (**f**) Observations from Allen Cell Types Database (green) and corresponding mode samples (purple). Posteriors in ***Appendix 1—figure 9***.

small regions of the posterior, that is, IBEA did not identify the full space of data-compatible models.

To investigate how individual data features constrain parameters, we compared SNPE-estimated posteriors based (1) solely on the spike count, (2) on the spike count and three voltage-features, or (3) on all 7 features of $\mathbf{x}_o$. As more features were taken into account, the posterior became narrower and centered more closely on the ground truth parameters (**Figure 4d**, **Appendix 1—figure 8**). Posterior simulations matched the observed data only in those features that had been used for inference (e.g. applying SNPE to spike counts alone identified parameters that generated the correct number of spikes, but for which spike timing and subthreshold voltage time course were off, **Figure 4e**). For some parameters, such as the potassium conductance, providing more data features brought the peak of the posterior (the *posterior mode*) closer to the ground truth and also decreased uncertainty. For other parameters, such as V_T, a parameter adjusting the spike threshold (**Pospischil et al., 2008**), the peak of the posterior was already close to the correct value with spike counts alone, but adding additional features reduced uncertainty. While SNPE can be used to study the effect of additional data features in reducing parameter uncertainty, this would not be the case for methods that only return a single best-guess estimate of parameters. These results show that SNPE can reveal how information from multiple data features imposes collective constraints on channel and membrane properties in the HH model.

We also inferred HH parameters for eight in vitro recordings from the Allen Cell Types database using the same current-clamp stimulation protocol as in our model (**Allen Institute for Brain Science, 2016**; **Teeter et al., 2018**; **Figure 4f**, **Appendix 1—figure 9**). In each case, simulations based on the SNPE-inferred posterior closely resembled the original data (**Figure 4f**). We note that while inferred parameters differed across recordings, some parameters (the spike threshold, the density of sodium channels, the membrane reversal potential and the density of potassium channels) were consistently more strongly constrained than others (the intrinsic neural noise, the adaptation time constant, the density of slow voltage-dependent channels and the leak conductance) (**Appendix 1—figure 9**). Overall, these results suggest that the electrophysiological responses measured by this current-clamp protocol can be approximated by a single-compartment HH model, and that SNPE can identify the admissible parameters.

Crustacean stomatogastric ganglion: sensitivity to perturbations

We next aimed to demonstrate how the full posterior distribution obtained with SNPE can lead to novel scientific insights. To do so, we used the pyloric network of the stomatogastric ganglion (STG) of the crab *Cancer borealis*, a well-characterized neural circuit producing rhythmic activity. In this circuit, similar network activity can arise from vastly different sets of membrane and synaptic conductances (**Prinz et al., 2004**). We first investigated whether data-consistent sets of membrane and synaptic conductances are connected in parameter space, as has been demonstrated for single neurons (**Taylor et al., 2006**), and, second, which compensation mechanisms between parameters of this circuit allow the neural system to maintain its activity despite parameter variations. While this model has been studied extensively, answering these questions requires characterizing higher dimensional parameter spaces than those accessed previously. We demonstrate how SNPE can be used to identify the posterior distribution over both membrane and synaptic conductances of the STG (31 parameters total) and how the full posterior distribution can be used to study the above questions at the circuit level.

For some biological systems, multiple parameter sets give rise to the same system behavior (**Prinz et al., 2004**; **Marder and Goaillard, 2006**; **Gutierrez et al., 2013**; **Fisher et al., 2013**; **Marder et al., 2015**; **Alonso and Marder, 2019**). In particular, neural systems can be robust to specific perturbations of parameters (**O'Leary et al., 2014**; **Marder et al., 2015**; **O'Leary and Marder, 2016**), yet highly sensitive to others, properties referred to as *sloppiness* and *stiffness* (**Goldman et al., 2001**; **Gutenkunst et al., 2007**; **Machta et al., 2013**; **O'Leary et al., 2015**). We studied how perturbations affect model output using a model (**Prinz et al., 2004**) and data (**Haddad and Marder, 2018**) of the pyloric rhythm in the crustacean stomatogastric ganglion (STG). This model describes a triphasic motor pattern generated by a well-characterized circuit (**Figure 5a**). The circuit consists of two electrically coupled pacemaker neurons (anterior burster and pyloric dilator, AB/PD), modeled as a single neuron, as well as two types of follower neurons (lateral pyloric (LP) and pyloric (PY)), all connected through inhibitory synapses (details in Materials and methods). Eight membrane conductances are included for each modeled neuron, along with seven synaptic conductances, for a total of 31 parameters. This model has been used to demonstrate that virtually indistinguishable activity can arise from vastly different membrane and synaptic conductances in the STG (***Prinz et al., 2004***; ***Alonso and Marder, 2019***). Here, we build on these studies and extend the model to include intrinsic neural noise on each neuron (see Materials and methods).

We applied SNPE to an extracellular recording from the STG of the crab *Cancer borealis* (***Haddad and Marder, 2018***) which exhibited pyloric activity (***Figure 5b***), and inferred the posterior distribution over all 31 parameters based on 18 salient features of the voltage traces, including cycle period, phase delays, phase gaps, and burst durations (features in ***Figure 5B***, posterior in ***Figure 5c***, posterior over all parameters in ***Appendix 1—figure 11***, details in Materials and methods). Consistent with previous reports, the posterior distribution has high probability over extended value ranges for many membrane and synaptic conductances. To verify that parameter settings across these extended ranges are indeed capable of generating the experimentally observed network activity, we sampled two sets of membrane and synaptic conductances from the posterior distribution. These two samples have widely disparate parameters from each other (***Figure 5c***, purple dots, details in Materials and methods), but both exhibit activity highly similar to the experimental observation (***Figure 5d***, top left and top right).

We then investigated the geometry of the parameter space producing these rhythms (***Achard and De Schutter, 2006***; ***Alonso and Marder, 2019***). First, we wanted to identify directions of sloppiness, and we were interested in whether parameter settings producing pyloric rhythms form a single connected region, as has been shown for single neurons (***Taylor et al., 2006***), or whether they lie on separate 'islands'. Starting from the two parameter settings showing similar activity above, we examined whether they were connected by searching for a path

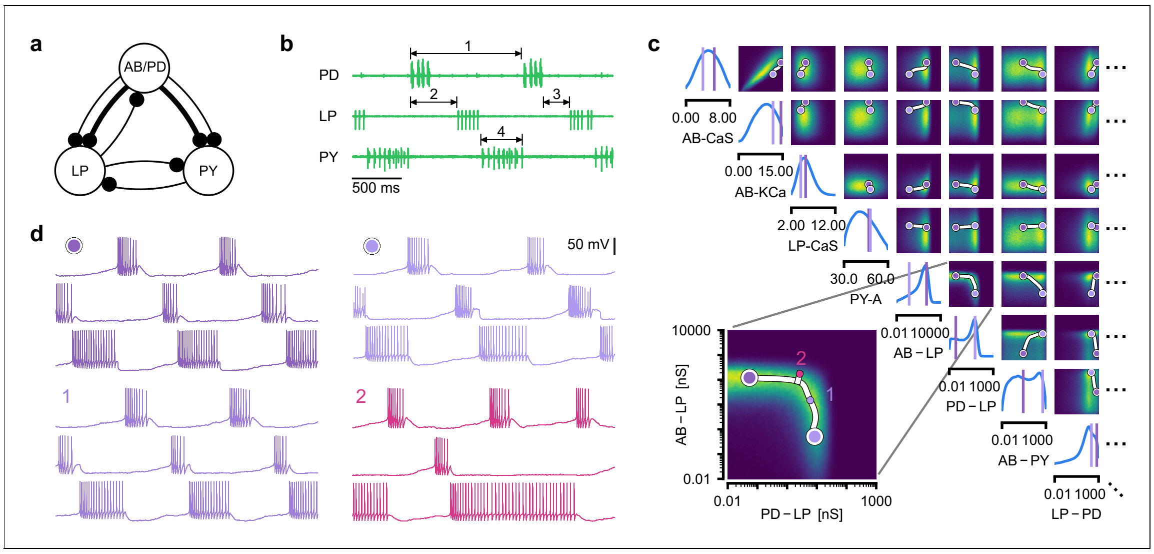


Figure 5. Identifying network models underlying an experimentally observed pyloric rhythm in the crustacean stomatogastric ganglion. (**a**) Simplified circuit diagram of the pyloric network from the stomatogastric ganglion. Thin connections are fast glutamatergic, thick connections are slow cholinergic. (**b**) Extracellular recordings from nerves of pyloric motor neurons of the crab *Cancer borealis* (***Haddad and Marder, 2018***). Numbers indicate some of the used summary features, namely cycle period (1), phase delays (2), phase gaps (3), and burst durations (4) (see Materials and methods for details). (**c**) Posterior over 24 membrane and seven synaptic conductances given the experimental observation shown in panel b (eight parameters shown, full posterior in ***Appendix 1—figure 11***). Two high-probability parameter sets in purple. Inset: magnified marginal posterior for the synaptic strengths AB to LP neuron vs. PD to LP neuron. (**d**) Identifying directions of sloppiness and stiffness. Two samples from the posterior both show similar network activity as the experimental observation (top left and top right), but have very different parameters (purple dots in panel c). Along the high-probability path between these samples, network activity is preserved (trace 1). When perturbing the parameters orthogonally off the path, network activity changes abruptly and becomes non-pyloric (trace 2).

through parameter space along which pyloric activity was maintained. To do this, we algorithmically identified a path lying only in regions of high posterior probability (***Figure 5c***, white, details in Materials and methods). Along the path, network output was tightly preserved, despite a substantial variation of the parameters (voltage trace 1 in ***Figure 5d***, ***Appendix 1—figure 12***). Second, we inspected directions of stiffness by perturbing parameters off the path. We applied perturbations that yield maximal drops in posterior probability (see Materials and methods for details), and found that the network quickly produced non-pyloric activity (voltage trace 2, ***Figure 5d***; ***Goldman et al., 2001***). Note that, while parameter set 2 seems to lie in regions of high probability when inspecting pairwise marginals, it in fact has low probability under the full posterior distribution (***Appendix 1—figure 13***). In identifying these paths and perturbations, we exploited the fact that SNPE provides a differentiable estimate of the posterior, as opposed to parameter search methods which provide only discrete samples.

Overall, these results show that the pyloric network can be robust to specific perturbations in parameter space, but sensitive to others, and that one can interpolate between disparate solutions while preserving network activity. This analysis demonstrates the flexibility of SNPE in capturing complex posterior distributions, and shows how the differentiable posterior can be used to study directions of sloppiness and stiffness.

Predicting compensation mechanisms from posterior distributions

Experimental and computational studies have shown that stable neural activity can be maintained despite variable circuit parameters (***Prinz et al., 2004***; ***Marder and Taylor, 2011***; ***O'Leary, 2018***). This behavior can emerge from two sources (***Marder and Taylor, 2011***): either, the variation of a certain parameter barely influences network activity at all, or alternatively, variations of several parameters influence network activity, but their effects compensate for one another. Here, we investigated these possibilities by using the posterior distribution over membrane and synaptic conductances of the STG.

We began by drawing samples from the posterior and inspecting their pairwise histograms (i.e. the pairwise marginals, ***Figure 6a***, posterior over all parameters in ***Appendix 1—figure 11***). Consistent with previously reported results (***Taylor et al., 2009***), many parameters seem only weakly constrained and only weakly correlated (***Figure 6b***). However, this observation does not imply that the parameters of the network do not have to be finely tuned: pairwise marginals are averages over many network configurations, where all other parameters may take on diverse values, which could disguise that each individual configuration is finely tuned. Indeed, when we sampled parameters independently from their posterior histograms, the resulting circuit configurations rarely produced pyloric activity, indicating that parameters have to be tuned relative to each other (***Appendix 1—figure 14***). This analysis also illustrates that the (common) approach of independently setting parameters can be problematic: although each parameter individually is in a realistic range, the network as a whole is not (***Golowasch et al., 2002***). Finally, it shows the importance of identifying the full posterior distribution, which is far more informative than just finding individual parameters and assigning error bars.

In order to investigate the need for tuning between pairs of parameters, we held all but two parameters constant at a given consistent circuit configuration (sampled from the posterior), and observed the network activity across different values of the remaining pair of parameters. We can do so by calculating the conditional posterior distribution (details in Materials and methods), and do not have to generate additional simulations (as would be required by parameter search methods). Doing so has a simple interpretation: when all but two parameters are fixed, what values of the remaining two parameters can then lead to the desired network activity? We found that the desired pattern of pyloric activity can emerge only from narrowly tuned and often highly correlated combinations of the remaining two parameters, showing how these parameters can compensate for one another (***Figure 6c***). When repeating this analysis across multiple network configurations, we found that these 'conditional correlations' are often preserved (***Figure 6c***, left and right). This demonstrates that pairs of parameters can compensate for each other in a similar way, independently of the values taken by other parameters. This observation about compensation could be interpreted as an instance of modularity, a widespread underlying principle of biological robustness (***Kitano, 2004***).

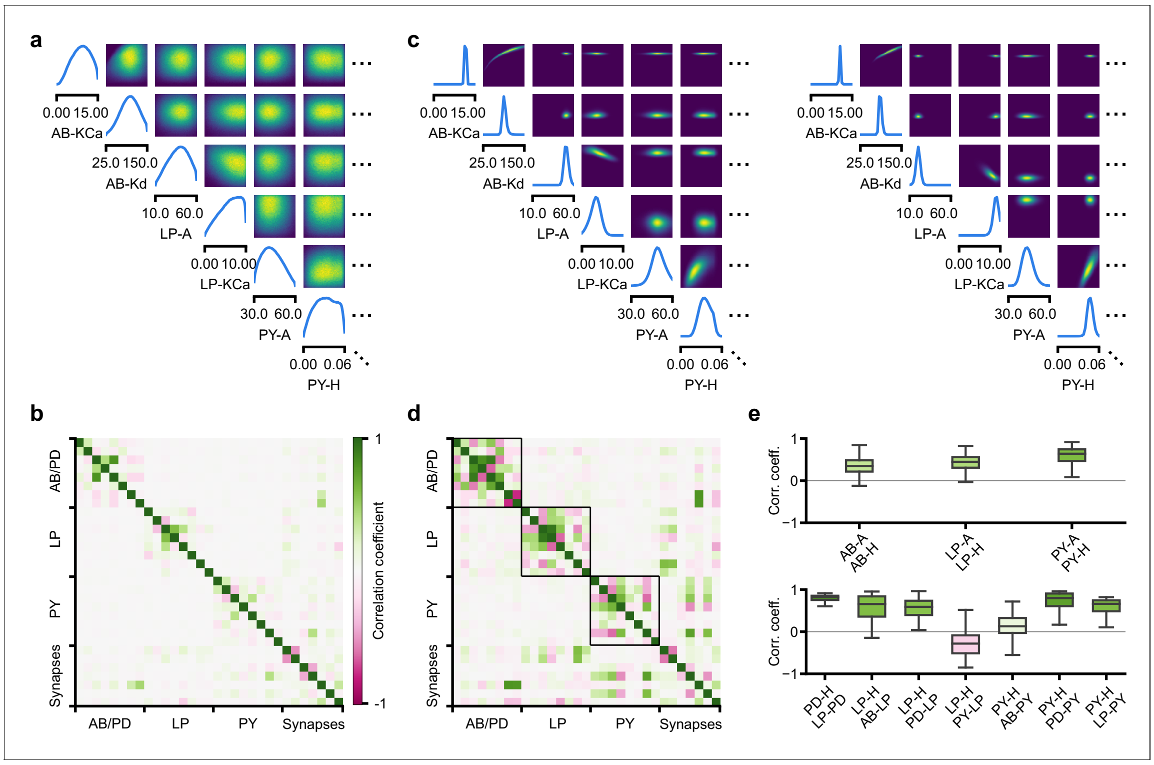


Figure 6. Predicting compensation mechanisms in the stomatogastric ganglion. (a) Inferred posterior. We show a subset of parameters which are weakly constrained (full posterior in ***Appendix 1—figure 11***). Pyloric activity can emerge from a wide range of maximal membrane conductances, as the 1D and 2D posterior marginals cover almost the entire extent of the prior. (b) Correlation matrix, based on the samples shown in panel (a). Almost all correlations are weak. Ordering of membrane and synaptic conductances as in ***Appendix 1—figure 11***. (c) Conditional distributions given a particular circuit configuration: for the plots on the diagonal, we keep all but one parameter fixed. For plots above the diagonal, we keep all but two parameters fixed. The remaining parameter(s) are narrowly tuned; tuning across parameters is often highly correlated. When conditioning on a different parameter setting (right plot), the conditional posteriors change, but correlations are often maintained. (d) Conditional correlation matrix, averaged over 500 conditional distributions like the ones shown in panel (c). Black squares highlight parameter-pairs within the same model neuron. (e) Consistency with experimental observations. Top: maximal conductance of the fast transient potassium current and the maximal conductance of the hyperpolarization current are positively correlated for all three neurons. This has also been experimentally observed in the PD and the LP neuron (***MacLean et al., 2005***). Bottom: the maximal conductance of the hyperpolarization current of the postsynaptic neuron can compensate the strength of the synaptic input, as experimentally observed in the PD and the LP neuron (***Grashow et al., 2010***; ***Marder, 2011***). The boxplots indicate the maximum, 75% quantile, median, 25% quantile, and minimum across 500 conditional correlations for different parameter pairs. Face color indicates mean correlation using the colorbar shown in panel (b).

We calculated conditional correlations for each parameter pair using 500 different circuit configurations sampled from the posterior (***Figure 6d***). Compared to correlations based on the pairwise marginals (***Figure 6b***), these conditional correlations were substantially stronger. They were particularly strong across membrane conductances of the same neuron, but primarily weak across different neurons (black boxes in ***Figure 6d***).

Finally, we tested whether the conditional correlations were in line with experimental observations. For the PD and the LP neuron, it has been reported that overexpression of the fast transient potassium current (I_A) leads to a compensating increase of the hyperpolarization current (I_H), suggesting a positive correlation between these two currents (***MacLean et al., 2003***; ***MacLean et al., 2005***). These results are qualitatively consistent with the positive conditional correlations between the maximal conductances of I_A and I_H for all three model neurons (***Figure 6e*** top). In addition, using the dynamic clamp, it has been shown that diverse combinations of the synaptic input strength and the maximal conductance of I_{H} lead to similar activity in the LP and the PD neuron (**Grashow et al., 2010**; **Marder, 2011**). Consistent with these findings, the non-zero conditional correlations reveal that there can indeed be compensation mechanisms between the synaptic strength and the maximal conductance of I_{H} of the postsynaptic neuron (**Figure 6e** bottom).

Overall, we showed how SNPE can be used to study parameter dependencies, and how the posterior distribution can be used to efficiently explore potential compensation mechanisms. We found that our method can predict compensation mechanisms which are qualitatively consistent with experimental studies. We emphasize that these findings would not have been possible with a direct grid-search over all parameters: defining a grid in a 31-dimensional parameter space would require more than $2^{31} > 2$ billion simulations, even if one were to use the coarsest-possible grid with only two values per dimension.

Discussion

How can we build models which give insights into the causal mechanisms underlying neural or behavioral dynamics? The cycle of building mechanistic models, generating predictions, comparing them to empirical data, and rejecting or refining models has been of crucial importance in the empirical sciences. However, a key challenge has been the difficulty of identifying mechanistic models which can quantitatively capture observed phenomena. We suggest that a generally applicable tool to constrain mechanistic models by data would expedite progress in neuroscience. While many considerations should go into designing a model that is appropriate for a given question and level of description (**Herz et al., 2006**; **Brette, 2015**; **Gerstner et al., 2012**; **O'Leary et al., 2015**), the question of whether and how one can perform statistical inference should not compromise model design. In our tool, SNPE, the process of model building and parameter inference are entirely decoupled. SNPE can be applied to *any* simulation-based model (requiring neither model nor summary features to be differentiable) and gives full flexibility on defining a prior. We illustrated the power of our approach on a diverse set of applications, highlighting the potential of SNPE to rapidly identify data-compatible mechanistic models, to investigate which data-features effectively constrain parameters, and to reveal shortcomings of candidate-models.

Finally, we used a model of the stomatogastric ganglion to show how SNPE can identify complex, high-dimensional parameter landscapes of neural systems. We analyzed the geometrical structure of the parameter landscape and confirmed that circuit configurations need to be finely tuned, even if individual parameters can take on a broad range of values. We showed that different configurations are connected in parameter space, and provided hypotheses for compensation mechanisms. These analyses were made possible by SNPE's ability to estimate full parameter posteriors, rather than just constraints on individual parameters, as is common in many statistical parameter-identification approaches.

Related work

SNPE builds on recent advances in machine learning and in particular in density-estimation approaches to likelihood-free inference (**Papamakarios and Murray, 2016**; **Le et al., 2017a**; **Lueckmann et al., 2017**; **Chan et al., 2018**; **Greenberg et al., 2019**, reviewed in **Cranmer et al., 2020**). We here scaled these approaches to canonical mechanistic models of neural dynamics and provided methods and software-tools for inference, visualization, and analysis of the resulting posteriors (e.g. the high-probability paths and conditional correlations presented here).

The idea of learning inference networks on simulated data can be traced back to *regression-adjustment* methods in ABC (**Beaumont et al., 2002**; **Blum and François, 2010**). **Papamakarios and Murray, 2016** first proposed to use expressive conditional density estimators in the form of deep neural networks (**Bishop, 1994**; **Papamakarios et al., 2017**), and to optimize them sequentially over multiple rounds with cost-functions derived from Bayesian inference principles. Compared to commonly used rejection-based ABC methods (**Rubin, 1984**; **Pritchard et al., 1999**), such as MCMC-ABC (**Marjoram et al., 2003**), SMC-ABC (**Sisson et al., 2007**; **Liepe et al., 2014**), Bayesian-Optimization ABC (**Gutmann and Corander, 2016**), or ensemble methods (**Britton et al., 2013**; **Lawson et al., 2018**), SNPE approaches do not require one to define a distance function in data space. In addition, by leveraging the ability of neural networks to learn informative features, they enable scaling to problems with high-dimensional observations, as are common in neuroscience and other fields in biology. We have illustrated this capability in the context of receptive field estimation, where a convolutional neural network extracts summary features from a 1681 dimensional spike-triggered average. Alternative likelihood-free approaches include *synthetic likelihood* methods (***Wood, 2010***; ***Costa et al., 2013***; ***Wilkinson, 2014***; ***Meeds and Welling, 2014***; ***Papamakarios et al., 2019a***; ***Lueckmann et al., 2019***; ***Durkan et al., 2018***), moment-based approximations of the posterior (***Barthelmé and Chopin, 2014***; ***Schröder et al., 2019***), inference compilation (***Le et al., 2017b***; ***Casado et al., 2017***), and density-ratio estimation (***Hermans et al., 2020***). For some mechanistic models in neuroscience (e.g. for integrate-and-fire neurons), likelihoods can be computed via stochastic numerical approximations (***Chen, 2003***; ***Huys and Paninski, 2009***; ***Meliza et al., 2014***) or model-specific analytical approaches (***Huys et al., 2006***; ***Hertäg et al., 2012***; ***Pozzorini et al., 2015***; ***Ladenbauer et al., 2018***; ***René et al., 2020***).

How big is the advance brought by SNPE relative to 'conventional' brute-force approaches that aim to exhaustively explore parameter space? A fundamental difference from grid search approaches that have been applied to neuroscientific models (***Prinz et al., 2003***; ***Caplan et al., 2014***; ***Stringer et al., 2016***) is that SNPE can perform Bayesian inference for *stochastic* models, whereas previous approaches identified parameters whose *deterministic* model-outputs were heuristically 'close' to empirical data. Depending on the goal of the analysis, either approach might be preferable. SNPE, and Bayesian inference more generally, is derived for stochastic models. SNPE can, in principle, also be applied to deterministic models, but a rigorous mathematical interpretation or empirical evaluation in this regime is beyond the scope of this study. SNPE also differs conceptually and quantitatively from rejection-ABC, in which random parameters are accepted or rejected based on a distance-criterion. SNPE uses *all* simulations during training instead of rejecting some, learns to identify data features informative about model parameters rather than relying on the user to choose the correct data features and distance metric, and performs considerably better than rejection-ABC, in particular for problems with high-dimensional observations (***Figure 2***). Another advantage over grid search and rejection-ABC is that SNPE can 'amortize' inference of parameter posteriors, so that one can quickly perform inference on new data, or explore compensation mechanisms, without having to carry out new simulations, or repeatedly search a simulation database. We should still note that SNPE can require the generation of large sets of simulations, which can be viewed as a brute-force step, emphasising that one of the main strengths of SNPE over conventional brute-force approaches relies on the processing of these simulations via deep neural density estimators.

Our approach is already finding its first applications in neuroscience–for example, ***Oesterle et al., 2020*** have used a variant of SNPE to constrain biophysical models of retinal neurons, with the goal of optimizing stimulation approaches for neuroprosthetics. Concurrently with our work, ***Bittner et al., 2019*** developed an alternative approach to parameter identification for mechanistic models and showed how it can be used to characterize neural population models which exhibit specific emergent computational properties. Both studies differ in their methodology and domain of applicability (see descriptions of underlying algorithms in our prior work [***Lueckmann et al., 2017***; ***Greenberg et al., 2019***] and theirs [***Loaiza-Ganem et al., 2017***]), as well in the focus of their neuroscientific contributions. Both approaches share the overall goal of using deep probabilistic inference tools to build more interpretable models of neural data. These complementary and concurrent advances will expedite the cycle of building, adjusting and selecting mechanistic models in neuroscience.

Finally, a complementary approach to mechanistic modeling is to pursue purely phenomenological models, which are designed to have favorable statistical and computational properties: these data-driven models can be efficiently fit to neural data (***Brown et al., 1998***; ***Truccolo et al., 2005***; ***Pillow, 2007***; ***Pillow et al., 2008***; ***Schneidman et al., 2006***; ***Macke et al., 2011***; ***Yu et al., 2009***; ***Pandarinath et al., 2018***; ***Cunningham and Yu, 2014***) or to implement desired computations (***Sussillo and Abbott, 2009***). Although tremendously useful for a quantitative characterization of neural dynamics, these models typically have a large number of parameters, which rarely correspond to physically measurable or mechanistically interpretable quantities, and thus it can be challenging to derive mechanistic insights or causal hypotheses from them (but see e.g. ***Mante et al., 2013***; ***Sussillo and Barak, 2013***; ***Maheswaranathan et al., 2019***).

Use of summary features

When fitting mechanistic models to data, it is common to target summary features to isolate specific behaviors, rather than the full data. For example, the spike shape is known to constrain sodium and potassium conductances (**Druckmann et al., 2007**; **Pospischil et al., 2008**; **Hay et al., 2011**). When modeling population dynamics, it is often desirable to achieve realistic firing rates, rate-correlations and response nonlinearities (**Rubin et al., 2015**; **Bittner et al., 2019**), or specified oscillations (**Prinz et al., 2004**). In models of decision making, one is often interested in reproducing psychometric functions or reaction-time distributions (**Ratcliff and McKoon, 2008**). Choice of summary features might also be guided by known limitations of either the model or the measurement approach, or necessitated by the fact that published data are only available in summarized form. Several methods have been proposed to automatically construct informative summary features (**Blum et al., 2013**; **Jiang et al., 2017**; **Izbicki et al., 2019**). SNPE can be applied to, and might benefit from the use of summary features, but it also makes use of the ability of neural networks to automatically learn informative features in high-dimensional data. Thus, SNPE can also be applied directly to raw data (e.g. using recurrent neural networks [**Lueckmann et al., 2017**]), or to high-dimensional summary features which are challenging for ABC approaches (**Figure 2**). In all cases, care is needed when interpreting models fit to summary features, as choice of features can influence the results (**Blum et al., 2013**; **Jiang et al., 2017**; **Izbicki et al., 2019**).

Applicability and limitations

A key advantage of SNPE is its general applicability: it can be applied whenever one has a simulator that allows to stochastically generate model outputs from specific parameters. Furthermore, it can be applied in a fully 'black-box manner', that is, does not require access to the internal workings of the simulator, its model equations, likelihoods or gradients. It does not impose any other limitations on the model or the summary features, and in particular does not require them to be differentiable. However, it also has limitations which we enumerate below.

First, current implementations of SNPE scale well to high-dimensional observations (~1000s of dimensions, also see **Greenberg et al., 2019**), but scaling SNPE to even higher-dimensional parameter spaces (above 30) is challenging (note that previous approaches were generally limited to less than 10 dimensions). Given that the difficulty of estimating full posteriors scales exponentially with dimensionality, this is an inherent challenge for all approaches that aim at full inference (in contrast to just identifying a single, or a few heuristically chosen parameter fits).

Second, while it is a long-term goal for these approaches to be made fully automatic, our current implementation still requires choices by the user: as described in Materials and methods, one needs to choose the type of the density estimation network, and specify settings related to network-optimization, and the number of simulations and inference rounds. These settings depend on the complexity of the relation between summary features and model parameters, and the number of simulations that can be afforded. In the documentation accompanying our code-package, we provide examples and guidance. For small-scale problems, we have found SNPE to be robust to these settings. However, for challenging, high-dimensional applications, SNPE might currently require substantial user interaction.

Third, the power of SNPE crucially rests on the ability of deep neural networks to perform density estimation. While deep nets have had ample empirical success, we still have an incomplete understanding of their limitations, in particular in cases where the mapping between data and parameters might not be smooth (e.g. near phase transitions).

Fourth, when applying SNPE (or any other model-identification approach), validation of the results is of crucial importance, both to assess the accuracy of the inference procedure, as well as to identify possible limitations of the mechanistic model itself. In the example applications, we used several procedures for assessing the quality of the inferred posteriors. One common ingredient of these approaches is to sample from the inferred model, and search for systematic differences between observed and simulated data, e.g. to perform *posterior predictive checks* (**Cook et al., 2006**; **Talts et al., 2018**; **Liepe et al., 2014**; **Lueckmann et al., 2017**; **Greenberg et al., 2019**; **Figure 2g**, **Figure 3f,g**, **Figure 4c**, and **Figure 5d**). These approaches allow one to detect 'failures' of SNPE, that is, cases in which samples from the posterior do not reproduce the data. However, when diagnosing any Bayesian inference approach, it is challenging to rigorously rule out the possibility that additional parameter-settings (e.g. in an isolated 'island') would also explain the data. Thus, it is good practice to use multiple initializations of SNPE, and/or a large number of simulations in the initial round. There are challenges and opportunities ahead in further scaling and automating simulation-based inference approaches. However, in its current form, SNPE will be a powerful tool for quantitatively evaluating mechanistic hypotheses on neural data, and for designing better models of neural dynamics.

Materials and methods

Code availability

Code implementing SNPE based on Theano, is available at http://www.mackelab.org/delfi/. An extended toolbox based on PyTorch is available at http://www.mackelab.org/sbi/ (***Tejero-Cantero et al., 2020***).

Simulation-based inference

To perform Bayesian parameter identification with SNPE, three types of input need to be specified:

1. A mechanistic model. The model only needs to be specified through a simulator, that is that one can generate a simulation result $\mathbf{x}$ for any parameters θ. We do not assume access to the likelihood $p(\mathbf{x}|\theta)$ or the equations or internals of the code defining the model, nor do we require the model to be differentiable. This is in contrast to many alternative approaches (including ***Bittner et al., 2019***), which require the model to be differentiable and to be implemented in a software code that is amenable to automatic differentiation packages. Finally, SNPE can both deal with inputs $\mathbf{x}$ which resemble 'raw' outputs of the model, or summary features calculated from data.
2. Observed data $\mathbf{x}_o$ of the same form as the results $\mathbf{x}$ produced by model simulations.
3. A prior distribution $p(\theta)$ describing the range of possible parameters. $p(\theta)$ could consist of upper and lower bounds for each parameter, or a more complex distribution incorporating mechanistic first principles or knowledge gained from previous inference procedures on other data. In our applications, we chose priors deemed reasonable or informed by previous studies (see Materials and methods), although setting such priors is an open problem in itself, and outside of the scope of this study.

For each problem, our goal was to estimate the posterior distribution $p(\theta|\mathbf{x}_o)$. To do this, we used SNPE (***Papamakarios and Murray, 2016***; ***Lueckmann et al., 2017***; ***Greenberg et al., 2019***). Setting up the inference procedure required three design choices:

1. A network architecture, including number of layers, units per layer, layer type (feedforward or convolutional), activation function and skip connections.
2. A parametric family of probability densities $q_\psi(\theta)$ to represent inferred posteriors, to be used as conditional density estimator. We used either a mixture of Gaussians (MoG) or a masked autoregressive flow (MAF) (***Papamakarios et al., 2017***). In the former case, the number of components *K* must be specified; in the latter the number of *MADES* (Masked Autoencoder for Distribution Estimation) n_{MADES}. Both choices are able to represent richly structured, and multimodal posterior distributions (more details on neural density estimation below).
3. A simulation budget, that is, number of rounds *R* and simulations per round N_r. The required number of simulations depends on both the dimensionality and complexity of the function between summary statistics and model parameters. While the number of parameters and summary-features can easily be determined, it can be hard to determine how 'complex' (or nonlinear) this mapping is. This makes it difficult to give general guidelines on how many simulations will be required. A practical approach is to choose a simulation-budget based on the computational cost of the simulation, inspect the results (e.g. with posterior predictive checks), and add more simulations when it seems necessary.

We emphasize that SNPE is highly modular, that is, that the the inputs (data, the prior over parameter, the mechanistic model), and algorithmic components (network architecture, probability density, optimization approach) can all be modified and chosen independently. This allows neuroscientists to work with models which are designed with mechanistic principles—and not convenience of inference—in mind. Furthermore, it allows SNPE to benefit from advances in more flexible density estimators, more powerful network architectures, or optimization strategies.

With the problem and inference settings specified, SNPE adjusts the network weights ϕ based on simulation results, so that $p(\theta|\mathbf{x}) \approx q_{F(\mathbf{x},\phi)}(\theta)$ for any $\mathbf{x}$. In the first round of SNPE, simulation parameters are drawn from the prior $p(\theta)$. If a single round of inference is not sufficient, SNPE can be run in multiple rounds, in which samples are drawn from the version of $q_{F(\mathbf{x}_o,\phi)}(\theta)$ at the beginning of the round. After the last round, $q_{F(\mathbf{x}_o,\phi)}$ is returned as the inferred posterior on parameters θ given observed data $\mathbf{x}_o$. If SNPE is only run for a single round, then the generated samples only depend on the prior, but not on $\mathbf{x}_o$: in this case, the inference network is applicable to any data (covered by the prior ranges), and can be used for rapid amortized inference.

SNPE learns the correct network weights ϕ by minimizing the objective function $\sum_j \mathcal{L}(\theta_j, \mathbf{x}_j)$ where the simulation with parameters θ_j produced result $\mathbf{x}_j$. For the first round of SNPE $\mathcal{L}(\theta_j, \mathbf{x}_j) = -\log q_{F(\mathbf{x}_j,\phi)}$, while in subsequent rounds a different loss function accounts for the fact that simulation parameters were not sampled from the prior. Different choices of the loss function for later rounds result in SNPE-A (***Papamakarios and Murray, 2016***), SNPE-B (***Lueckmann et al., 2017***) or SNPE-C algorithm (***Greenberg et al., 2019***). To optimize the networks, we used ADAM with default settings (***Kingma and Ba, 2014***).

The details of the algorithm are below:

Algorithm 1: SNPE

```
Input: simulator with (implicit) density $p(\mathbf{x}|\theta)$, observed data $\mathbf{x}_o$, prior $p(\theta)$, density family $q_\psi$,
  neural network $F(\mathbf{x}, \phi)$, number of rounds , simulation count for each round $N_r$
  randomly initialize $\phi$
  $\tilde{p}_1(\theta) := p(\theta)$
  $N := 0$
  for $r = 1$ to $R$ do
      for $i = 1 \ldots N_r$ do
          sample $\theta_{N+i} \sim \tilde{p}_r(\theta)$
          simulate $\mathbf{x}_{N+i} \sim p(\mathbf{x}|\theta_{N+i})$
      $N \leftarrow N + N_r$
      train $\phi \leftarrow_\phi \sum_{j=1}^{N} \mathcal{L}(\theta_j, \mathbf{x}_j)$
      $\tilde{p}_r(\theta) := q_{F(\mathbf{x}_o,\phi)}(\theta)$
  return $q_{F(\mathbf{x}_o,\phi)}(\theta)$
```

Bayesian inference without likelihood-evaluations with SNPE

In ***Papamakarios and Murray, 2016***, it was shown that the procedure described above (i.e. sample from the prior, train a flexible density estimator by minimizing the log-loss $\mathcal{L}(\theta_j, \mathbf{x}_j) = -\sum_j \log q_{F(\mathbf{x}_j,\phi)}(\theta_j)$) can be used to perform Bayesian inference without likelihood evaluations.

For the multi-round case, in which samples are no longer drawn from the prior, but adaptively generated from a (generally more focused) proposal distribution, the loss function needs to be modified. Different variants of SNPE differ in how exactly this is done:

- SNPE-A minimizes the same loss function as in the first round, but applies a post-hoc analytical correction (***Papamakarios and Murray, 2016***)
- SNPE-B minimizes an importance-weighted loss function, directly approximating the posterior and therefore not requiring a post-hoc correction (***Lueckmann et al., 2017***)
- SNPE-C avoids importance weights (which can have high variance), by either calculating normalization constants in closed-form or using a classifier-based loss (***Greenberg et al., 2019***)

Neural density estimation

As described above, SNPE approximates the posterior distribution with flexible neural density estimators: either a mixture density network (MDN) or a masked autoregressive flow (MAF). Below, we provide a few more details about these density estimators, how we chose their respective architectures, and when to choose one or the other.

The MDN outputs the parameters of a mixture of Gaussians (i.e. mixture weights, and for each component of the mixture, the mean vector and covariance entries). Thus, for an MDN composed of

K components, we chose an architecture with at least as many units per layer as $K(1+N_\theta+N_\theta(N_\theta+1)/2)-1$, where N_θ is the number of parameters to infer, to ensure enough flexibility to approximate well the parameters of the mixture of Gaussians. For example, when inferring the parameters of the Hodgkin-Huxley model given in vitro recordings from mouse cortex (Allen Cell Types Database, https://celltypes.brain-map.org/data), we infer the posterior over eight parameters with a mixture of two Gaussians, and the MDN needs at least 89 units per layer. Across applications, we found two layers to be sufficient to appropriately approximate the posterior distribution.

MAF is a specific type of normalizing flow, which is a highly flexible density estimator (***Rezende and Mohamed, 2015***; ***Papamakarios et al., 2017***; ***Papamakarios et al., 2019b***). Normalizing flows consist of a stack of bijections which transform a simple distribution (usually a multivariate Gaussian distribution) into the target distribution. Each bijection is parameterized by a specific type of neural network (for MAF: a Masked Autoencoder for Distribution Estimation, or MADE). In our experiments, five stacked bijections are enough to approximate even complex posterior distributions. Depending on the size of the parameter and data space, each neural network had between [50,50] and [100,100,100] hidden units.

When using SNPE in a single-round, we generally found superior performance for MAFs as compared to MDNs. When running inference across multiple rounds, training MAFs leads to additional challenges which might impede the quality of inference (***Greenberg et al., 2019***; ***Durkan et al., 2020***).

Linear-nonlinear encoding models

We used a Linear-Nonlinear (LN) encoding model (a special case of a generalized linear model, GLM, [***Brown et al., 1998***; ***Paninski, 2004***; ***Truccolo et al., 2005***; ***Pillow, 2007***; ***Pillow et al., 2008***; ***Gerwinn et al., 2010***]) to simulate the activity of a neuron in response to a univariate time-varying stimulus. Neural activity z_i was subdivided in $T=100$ bins and, within each bin i, spikes were generated according to a Bernoulli observation model,

$$z_i \sim \mathrm{Bern}(\eta(\mathbf{v}_i^\top \boldsymbol{f}+\beta)),$$

where $\mathbf{v}_i$ is a vector of white noise inputs between time bins $i-8$ and i, $\boldsymbol{f}$ a length-9 linear filter, β is the bias, and $\eta(\cdot)=\exp(\cdot)/(1+\exp(\cdot))$ is the canonical inverse link function for a Bernoulli GLM. As summary features, we used the total number of spikes N and the spike-triggered average $\frac{1}{N}\mathbf{Vz}$, where $\mathbf{V}=[v_1,v_2,\ldots,v_T]$ is the so-called design matrix of size $9\times T$. We note that the spike-triggered sum $\mathbf{Vz}$ constitutes sufficient statistics for this GLM, that is that selecting the STA and N together as summary features does not lead to loss of model relevant information over the full input-output dataset $\{\mathbf{V},\mathbf{z}\}$. We used a Gaussian prior with zero mean and covariance matrix $\boldsymbol{\Sigma}=\sigma^2(\mathbf{F}^\top\mathbf{F})^{-1}$, where $\mathbf{F}$ encourages smoothness by penalizing the second-order differences in the vector of parameters (***De Nicolao et al., 1997***).

For inference, we used a single round of 10,000 simulations, and the posterior was approximated with a Gaussian distribution ($\theta\in\mathbb{R}^{10},\mathbf{x}\in\mathbb{R}^{10}$). We used a feedforward neural network with two hidden layers of 50 units each. We used a Polya Gamma Markov Chain Monte Carlo sampling scheme (***Polson et al., 2013***) to estimate a reference posterior.

In ***Figure 2d***, we compare the performance of SNPE with two classical ABC algorithms, rejection ABC and Sequential Monte Carlo ABC as a function of the number of simulations. We report the relative error in Kullback-Leibler divergence, which is defined as:

$$\frac{D_{\mathrm{KL}}(p_{MCMC}(\theta|\mathbf{x})\,||\,\hat{p}(\theta|\mathbf{x}))}{D_{\mathrm{KL}}(p_{MCMC}(\theta|\mathbf{x})\,||\,p(\theta))}, \tag{1}$$

and which ranges between 0 (perfect recovery of the posterior) and 1 (estimated posterior no better than the prior). Here, $p_{MCMC}(\theta|\mathbf{x})$ is the ground-truth posterior estimated via Markov Chain Monte Carlo sampling, $\hat{p}(\theta|\mathbf{x})$ is the estimated posterior via SNPE, rejection ABC or Sequential Monte Carlo ABC, and $p(\theta)$ is the prior.

For the spatial receptive field model of a cell in primary visual cortex, we simulated the activity of a neuron depending on an image-valued stimulus. Neural activity was subdivided in bins of length $\Delta t=0.025s$ and within each bin i, spikes were generated according to a Poisson observation model,

$$z_i \sim \mathrm{Poiss}(\eta(\mathbf{v}_i^\top \boldsymbol{h} + \beta)),$$

where $\mathbf{v}_i$ is the vectorized white noise stimulus at time bin *i*, $\boldsymbol{h}$ a 41 × 41 linear filter, β is the bias, and $\eta(\cdot) = \exp(\cdot)$ is the canonical inverse link function for a Poisson GLM. The receptive field $\boldsymbol{h}$ is constrained to be a Gabor filter:

$$\begin{aligned}
h(g_x, g_y) &= g \exp\left(-\frac{x'^2 + r^2 y'^2}{2\sigma^2}\right) \cos(2\pi f x' - \phi)\\
x' &= (g_x - x)\cos\psi - (g_y - y)\sin\psi\\
y' &= (g_x - x)\sin\psi + (g_y - y)\cos\psi\\
\sigma &= \frac{\sqrt{2}\log 2}{2\pi f}\frac{2^w + 1}{2^w - 1},
\end{aligned}$$

where (g_x, g_y) is a regular grid of 41 × 41 positions spanning the 2D image-valued stimulus. The parameters of the Gabor are gain *g*, spatial frequency *f*, aspect-ratio *r*, width *w*, phase ϕ (between 0 and π), angle ψ (between 0 and 2π) and location x, y (assumed within the stimulated area, scaled to be between −1 and 1). Bounded parameters were transformed with a log-, or logit-transform, to yield unconstrained parameters. After applying SNPE, we back-transformed both the parameters and the estimated posteriors in closed form, as shown in ***Figure 2***. We did not transform the bias β.

We used a factorizing Gaussian prior for the vector of transformed Gabor parameters

$$[\,\log g,\ \log f,\ \log r,\ \log w,\ l_{0,\pi}(\phi),\ l_{0,2\pi}(\psi),\ l_{-1,1}(x),\ l_{-1,1}(y)\,],$$

where transforms $l_{0,\pi}(X) = \log(X/(2\pi - X))$, $l_{0,2\pi}(X) = \log(X/(\pi - X))$, $l_{-1,1}(X) = \log((X+1)/(1-X))$ ensured the assumed ranges for the Gabor parameters ϕ, ψ, x, y. Our Gaussian prior had zero mean and standard deviations $[0.5, 0.5, 0.5, 0.5, 1.9, 1.78, 1.78, 1.78]$. We note that a Gaussian prior on a logit-transformed random variable $\mathrm{logit} X$ with zero mean and standard deviation around 1.78 is close to a uniform prior over the original variable *X*. For the bias β, we used a Gaussian prior with mean −0.57 and variance 1.63, which approximately corresponds to an exponential prior $exp(\beta) \sim Exp(\lambda)$ with rate $\lambda = 1$ on the baseline firing rate $\exp(\beta)$ in absence of any stimulus.

The ground-truth parameters for the demonstration in ***Figure 2*** were chosen to give an asymptotic firing rate of 1 Hz for 5 min stimulation, resulting in 299 spikes, and a signal-to-noise ratio of −12dB.

As summary features, we used the total number of spikes *N* and the spike-triggered average $\frac{1}{N}\mathbf{V}\mathbf{z}$, where $\mathbf{V} = [v_1, v_2, \ldots, v_T]$ is the stimulation video of length $T = 300/\Delta t = 12000$. As for the GLM with a temporal filter, the spike-triggered sum $\mathbf{V}\mathbf{z}$ constitutes sufficient statistics for this GLM.

For inference, we applied SNPE-A with in total two rounds: an initial round serves to first roughly identify the relevant region of parameter space. Here we used a Gaussian distribution to approximate the posterior from 100,000 simulations. A second round then used a mixture of eight Gaussian components to estimate the exact shape of the posterior from another 100,000 simulations ($\theta \in R^9, \mathbf{x} \in R^{1682}$). We used a convolutional network with five convolutional layers with 16 to 32 convolutional filters followed by two fully connected layers with 50 units each. The total number of spikes *N* within a simulated experiment was passed as an additional input directly to the fully-connected layers of the network. Similar to the previous GLM, this model has a tractable likelihood, so we use MCMC to obtain a reference posterior.

We applied this approach to extracelullar recordings from primary visual cortex of alert mice obtained using silicon microelectrodes in response to colored-noise visual stimulation. Experimental methods are described in ***Dyballa et al., 2018***.

Comparison with Sequential Monte Carlo (SMC) ABC

In order to illustrate the competitive performance of SNPE, we obtained a posterior estimate with a classical ABC method, Sequential Monte Carlo (SMC) ABC (***Sisson et al., 2007***; ***Beaumont et al., 2009***). Likelihood-free inference methods from the ABC family require a distance function $d(\mathbf{x}_o, \mathbf{x})$ between observed data $\mathbf{x}_o$ and possible simulation outputs $\mathbf{x}$ to characterize dissimilarity between simulations and data. A common choice is the (scaled) Euclidean distance $d(\mathbf{x}_o, \mathbf{x}) = ||\mathbf{x} - \mathbf{x}_o||_2$. The Euclidean distance here was computed over 1681 summary features given by the spike-triggered average (one per pixel) and a single summary feature given by the 'spike count'. To ensure that the distance measure was sensitive to differences in both STA and spike count, we scaled the summary feature 'spike count' to account for about 20% of the average total distance (other values did not yield better results). The other 80% were computed from the remaining 1681 summary features given by spike-triggered averages.

To showcase how this situation is challenging for ABC approaches, we generated 10,000 input-output pairs $(\theta_i, \mathbf{x}_i) \sim p(\mathbf{x}|\theta)p(\theta)$ with the prior and simulator used above, and illustrate the 10 STAs and spike counts with closest $d(\mathbf{x}_o, \mathbf{x}_i)$ in ***Appendix 1—figure 5a***. Spike counts were comparable to the observed data (299 spikes), but STAs were noise-dominated and the 10 'closest' underlying receptive fields (orange contours) showed substantial variability in location and shape of the receptive field. If even the 'closest' samples do not show any visible receptive field, then there is little hope that even an appropriately chosen acceptance threshold will yield a good approximation to the posterior. These findings were also reflected in the results from SMC-ABC with a total simulation budget of 10^6 simulations (***Appendix 1—figure 5b***). The estimated posterior marginals for 'bias' and 'gain' parameters show that the parameters related to the firing rate were constrained by the data $\mathbf{x}_o$, but marginals of parameters related to shape and location of the receptive field did not differ from the prior, highlighting that SMC-ABC was not able to identify the posterior distribution. The low correlations between the ground-truth receptive field and receptive fields sampled from SMC-ABC posterior further highlight the failure of SMC-ABC to infer the ground-truth posterior (***Appendix 1—figure 5c***). Further comparisons of neural-density estimation approaches with ABC-methods can be found in the studies describing the underlying machine-learning methodologies (***Papamakarios and Murray, 2016***; ***Lueckmann et al., 2019***; ***Greenberg et al., 2019***).

Ion channel models

We simulated non-inactivating potassium channel currents subject to voltage-clamp protocols as:

$$I_K = \bar{g}_{\mathrm{K}} m(V - E_{\mathrm{K}}),$$

where *V* is the membrane potential, $\bar{g}_{\mathrm{K}}$ is the density of potassium channels, E_{K} is the reversal potential of potassium, and *m* is the gating variable for potassium channel activation. *m* is modeled according to the first-order kinetic equation

$$\frac{dm}{dt} = \frac{m_\infty(V) - m}{\tau_m(V)},$$

where $m_\infty(V)$ is the steady-state activation, and $\tau_m(V)$ the respective time constant. We used a general formulation of $m_\infty(V)$ and $\tau_m(V)$ (***Destexhe and Huguenard, 2000***), where the steady-state activation curve has two parameters (slope and offset) and the time constant curve has six parameters, amounting to a total of 8 parameters (θ_1 to θ_8):

$$\begin{aligned} m_\infty(V) &= \frac{1}{1 + e^{-\theta_1 V + \theta_2}} \\ \tau_m(V) &= \frac{\theta_4}{e^{-[\theta_5(V-\theta_3) + \theta_6(V-\theta_3)^2]} + e^{[\theta_7(V-\theta_3) + \theta_8(V-\theta_3)^2]}}. \end{aligned}$$

Since this model can be used to describe the dynamics of a wide variety of channel models, we refer to it as *Omnimodel*.

We modeled responses of the Omnimodel to a set of five noisy voltage-clamp protocols (***Podlaski et al., 2017***): as described in ***Podlaski et al., 2017***, the original voltage-clamp protocols correspond to standard protocols of activation, inactivation, deactivation, ramp and action potential, to which we added Gaussian noise with zero mean and standard deviation 0.5 mV. Current responses were reduced to 55 summary features (11 per protocol). Summary features were coefficients to basis functions derived via Principal Components Analysis (PCA) (10 per protocol) plus a linear offset (one per protocol) found via least-squares fitting. PCA basis functions were found by simulating responses of the non-inactivating potassium channel models to the five voltage-clamp protocols and reducing responses to each protocol to 10 dimensions (explaining 99.9% of the variance).

To amortize inference on the model, we specified a wide uniform prior over the parameters: $\theta_1 \in \mathcal{U}(0,1), \theta_2 \in \mathcal{U}(-10., 10.)$, $\theta_3 \in \mathcal{U}(-120., 120.), \theta_4 \in \mathcal{U}(0., 2000)$, $\theta_5 \in \mathcal{U}(0., 0.5), \theta_6 \in \mathcal{U}(0, 0.05)$, $\theta_7 \in \mathcal{U}(0., 0.5), \theta_8 \in \mathcal{U}(0, 0.05)$.

For inference, we trained a shared inference network in a single round of 10^6 simulations generated by sampling from the prior ($\theta \in R^8, \mathbf{x} \in R^{55}$). The density estimator was a masked autoregressive flow (MAF) (***Papamakarios et al., 2017***) with five MADES with [250,250] hidden units each.

We evaluated performance on 350 non-inactivating potassium ion channels selected from IonChannelGenealogy (ICG) by calculating the correlation coefficient between traces generated by the original model and traces from the Omnimodel using the posterior mode (***Appendix 1—figure 7***).

Single-compartment Hodgkin–Huxley neurons

We simulated a single-compartment Hodgkin–Huxley type neuron with channel kinetics as in ***Pospischil et al., 2008***,

$$\begin{aligned} C_m \frac{dV}{dt} &= g_l(E_l - V) + \bar{g}_{\mathrm{Na}} m^3 h (E_{\mathrm{Na}} - V) + \bar{g}_{\mathrm{K}} n^4 (E_{\mathrm{K}} - V) + \bar{g}_{\mathrm{M}} p (E_{\mathrm{K}} - V) + I_{\mathrm{inj}} + \sigma \eta(t) \\ \frac{dq}{dt} &= \frac{q_\infty(V) - q}{\tau_q(V)}, q \in \{m, h, n, p\}, \end{aligned}$$

where V is the membrane potential, C_m is the membrane capacitance, g_l is the leak conductance, E_l is the membrane reversal potential, $\bar{g}_c$ is the density of channels of type c (Na$^+$, K$^+$, M), E_c is the reversal potential of c, (m, h, n, p) are the respective channel gating kinetic variables, and $\sigma\eta(t)$ is the intrinsic neural Gaussian noise. The right hand side of the voltage dynamics is composed of a leak current, a voltage-dependent Na$^+$ current, a delayed-rectifier K$^+$ current, a slow voltage-dependent K$^+$ current responsible for spike-frequency adaptation, and an injected current I_{inj}. Channel gating variables q have dynamics fully characterized by the neuron membrane potential V, given the respective steady-state $q_\infty(V)$ and time constant $\tau_q(V)$ (details in ***Pospischil et al., 2008***). Two additional parameters are implicit in the functions $q_\infty(V)$ and $\tau_q(V)$: V_T adjusts the spike threshold through m_∞, h_∞, n_∞, τ_m, τ_h and τ_n; $\tau_{\max}$ scales the time constant of adaptation through $\tau_p(V)$ (details in ***Pospischil et al., 2008***). We set $E_{\mathrm{Na}} = 53$ mV and $E_{\mathrm{K}} = -107$ mV, similar to the values used for simulations in Allen Cell Types Database (http://help.brain-map.org/download/attachments/8323525/BiophysModelPeri.pdf).

We applied SNPE to infer the posterior over eight parameters ($\bar{g}_{\mathrm{Na}}$, $\bar{g}_{\mathrm{K}}$, g_l, $\bar{g}_{\mathrm{M}}$, $\tau_{\max}$, V_T, σ, E_l), given seven voltage features (number of spikes, mean resting potential, standard deviation of the resting potential, and the first four voltage moments, mean, standard deviation, skewness and kurtosis).

The prior distribution over the parameters was uniform,

$$\theta \sim \mathcal{U}(p_{\mathrm{low}}, p_{\mathrm{high}}),$$

where $p_{\mathrm{low}} = [0.5, 10^{-4}, 10^{-4}, 10^{-4}, 50, 40, 10^{-4}, 35]$ and $p_{\mathrm{high}} = [80, 15, 0.6, 0.6, 3000, 90, 0.15, 100]$. These ranges are similar to the ones obtained in ***Pospischil et al., 2008***, when fitting the above model to a set of electrophysiological recordings.

For inference in simulated data, we used a single round of 100,000 simulations ($\theta \in R^8, \mathbf{x} \in R^7$). The density estimator was a masked autoregressive flow (MAF) (***Papamakarios et al., 2017***) with five MADES with [50,50] hidden units each.

For the inference on in vitro recordings from mouse cortex (Allen Cell Types Database, https://celltypes.brain-map.org/data), we selected eight recordings corresponding to spiny neurons with at least 10 spikes during the current-clamp stimulation. The respective cell identities and sweeps are: (518290966,57), (509881736,39), (566517779,46), (567399060,38), (569469018,44), (532571720,42), (555060623,34), (534524026,29). For each recording, SNPE-B was run for two rounds with 125,000 Hodgkin–Huxley simulations each, and the posterior was approximated by a mixture of two Gaussians. In this case, the density estimator was composed of two fully connected layers of 100 units each.

Comparison with genetic algorithm

We compared SNPE posterior with a state-of-the-art genetic algorithm (Indicator Based Evolutionary Algorithm IBEA, [**Bleuler et al., 2003**; **Zitzler and Künzli, 2004**] from the BluePyOpt package [**Van Geit et al., 2016**]), in the context of the Hodgkin-Huxley model with 8 parameters and seven features (**Appendix 1—figure 10**). For each Hodgkin-Huxley model simulation *i* and summary feature *j*, we used the following objective score:

$$\epsilon_{ij} = \left| \frac{x_{ij} - x_{oj}}{\sigma_j} \right|, \; j = 1, \ldots, 7,$$

where x_{ij} is the value of summary feature *j* for simulation *i*, x_{oj} is the observed summary feature *j*, and σ_j is the standard deviation of the summary feature *j* computed across 1000 previously simulated datasets. IBEA outputs the hall-of-fame, which corresponds to the 10 parameter sets with the lowest sum of objectives $\sum_j^7 \epsilon_{ij}$. We ran IBEA with 100 generations and an offspring size of 1000 individuals, corresponding to a total of 100,000 simulations.

Circuit model of the crustacean stomatogastric ganglion

We used extracellular nerve recordings made from the stomatogastric motor neurons that principally comprise the triphasic pyloric rhythm in the crab *Cancer borealis* (**Haddad and Marder, 2018**). The preparations were decentralized, that is, the axons of the descending modulatory inputs were severed. The data was recorded at a temperature of 11°C. See **Haddad and Marder, 2018** for full experimental details.

We simulated the circuit model of the crustacean stomatogastric ganglion by adapting a model described in **Prinz et al., 2004**. The model is composed of three single-compartment neurons, AB/PD, LP, and PD, where the electrically coupled AB and PD neurons are modeled as a single neuron. Each of the model neurons contains eight currents, a Na$^+$ current I_{Na}, a fast and a slow transient Ca^{2+} current I_{CaT} and I_{CaS}, a transient K$^+$ current I_{A}, a Ca^{2+}-dependent K$^+$ current I_{KCa}, a delayed rectifier K$^+$ current I_{Kd}, a hyperpolarization-activated inward current I_{H}, and a leak current I_{leak}. In addition, the model contains seven synapses. As in **Prinz et al., 2004**, these synapses were simulated using a standard model of synaptic dynamics (**Abbott and Marder, 1998**). The synaptic input current into the neurons is given by $I_{\mathrm{s}} = g_{\mathrm{s}} s (V_{\mathrm{post}} - E_{\mathrm{s}})$, where g_{s} is the maximal synapse conductance, V_{post} the membrane potential of the postsynaptic neuron, and E_{s} the reversal potential of the synapse. The evolution of the activation variable s is given by

$$\frac{ds}{dt} = \frac{\bar{s}(V_{\mathrm{pre}}) - s}{\tau_s}$$

with

$$\bar{s}(V_{\mathrm{pre}}) = \frac{1}{1 + \exp((V_{\mathrm{th}} - V_{\mathrm{pre}})/\delta)} \quad \text{and} \quad \tau_s = \frac{1 - \bar{s}(V_{\mathrm{pre}})}{k_-}.$$

Here, V_{pre} is the membrane potential of the presynaptic neuron, V_{th} is the half-activation voltage of the synapse, δ sets the slope of the activation curve, and k_- is the rate constant for transmitter-receptor dissociation rate.

As in **Prinz et al., 2004**, two types of synapses were modeled since AB, LP, and PY are glutamatergic neurons whereas PD is cholinergic. We set $E_s = -70$ mV and $k_- = 1/40$ ms for all glutamatergic synapses and $E_s = -80$ mV and $k_- = 1/100$ ms for all cholinergic synapses. For both synapse types, we set $V_{\mathrm{th}} = -35$ mV and $\delta = 5$ mV.

For each set of membrane and synaptic conductances, we numerically simulated the rhythm for 10 s with a step size of 0.025 ms. At each time step, each neuron received Gaussian noise with mean zero and standard deviation 0.001 mV.ms$^{-0.5}$.

We applied SNPE to infer the posterior over 24 membrane parameters and 7 synaptic parameters, that is, 31 parameters in total. The seven synaptic parameters were the maximal conductances g_{s} of all synapses in the circuit, each of which was varied uniformly in logarithmic domain from $0.01\mathrm{nS}$ to $1000\mathrm{nS}$, with the exception of the synapse from AB to LP, which was varied uniformly in logarithmic domain from $0.01\mathrm{nS}$ to $10000\mathrm{nS}$. The membrane parameters were the maximal membrane conductances for each of the neurons. The membrane conductances were varied over an extended range of previously reported values (**Prinz et al., 2004**), which led us to the uniform prior bounds $p_{\mathrm{low}} = [0, 0, 0, 0, 0, 25, 0, 0]\mathrm{mScm}^{-2}$ and $p_{\mathrm{high}} = [500, 7.5, 8, 60, 15, 150, 0.2, 0.01]\mathrm{mScm}^{-2}$ for the maximal membrane conductances of the AB neuron, $p_{\mathrm{low}} = [0, 0, 2, 10, 0, 0, 0, 0.01]\mathrm{mScm}^{-2}$ and $p_{\mathrm{high}} = [200, 2.5, 12, 60, 10, 125, 0.06, 0.04]\mathrm{mScm}^{-2}$ for the maximal membrane conductances of the LP neuron, and $p_{\mathrm{low}} = [0, 0, 0, 30, 0, 50, 0, 0]\mathrm{mScm}^{-2}$ and $p_{\mathrm{high}} = [600, 12.5, 4, 60, 5, 150, 0.06, 0.04]\mathrm{mScm}^{-2}$ for the maximal membrane conductances of the PY neuron. The order of the membrane currents was: [Na, CaT, CaS, A, KCa, Kd, H, leak].

We used the 15 summary features proposed by **Prinz et al., 2004**, and extended them by three additional features. The features proposed by **Prinz et al., 2004** are 15 salient features of the pyloric rhythm, namely: cycle period T (s), AB/PD burst duration $d^{\mathrm{b}}_{\mathrm{AB}}$ (s), LP burst duration $d^{\mathrm{b}}_{\mathrm{LP}}$ (s), PY burst duration $d^{\mathrm{b}}_{\mathrm{PY}}$ (s), gap AB/PD end to LP start $\Delta t^{\mathrm{es}}_{\mathrm{AB-LP}}$ (s), gap LP end to PY start $\Delta t^{\mathrm{es}}_{\mathrm{LP-PY}}$ (s), delay AB/PD start to LP start $\Delta t^{\mathrm{ss}}_{\mathrm{AB-LP}}$ (s), delay LP start to PY start $\Delta t^{\mathrm{ss}}_{\mathrm{LP-PY}}$ (s), AB/PD duty cycle d_{AB}, LP duty cycle d_{LP}, PY duty cycle d_{PY}, phase gap AB/PD end to LP start $\Delta\phi_{\mathrm{AB-LP}}$, phase gap LP end to PY start $\Delta\phi_{\mathrm{LP-PY}}$, LP start phase ϕ_{LP}, and PY start phase ϕ_{PY}. Note that several of these values are only defined if each neuron produces rhythmic bursting behavior. In addition, for each of the three neurons, we used one feature that describes the maximal duration of its voltage being above -30 mV. We did this as we observed plateaus at around -10 mV during the onset of bursts, and wanted to distinguish such traces from others. If the maximal duration was below 5 ms, we set this feature to 5 ms. To extract the summary features from the observed experimental data, we first found spikes by searching for local maxima above a hand-picked voltage threshold, and then extracted the 15 above described features. We set the additional 3 features to 5 ms.

We used SNPE to infer the posterior distribution over the 18 summary features from experimental data. For inference, we used a single round with 18.5 million samples, out of which 174,000 samples contain bursts in all neurons. We therefore used these 174,000 samples with well defined summary features for training the inference network ($\theta \in R^{31}, \mathbf{x} \in R^{18}$). The density estimator was a masked autoregressive flow (MAF) (**Papamakarios et al., 2017**) with five MADES with [100,100,100] hidden units each. The synaptic conductances were transformed into logarithmic space before training and for the entire analysis.

Previous approaches for fitting the STG circuit (**Prinz et al., 2004**) first fit individual neuron features and reduce the number of possible neuron models (**Prinz et al., 2003**), and then fit the whole circuit model. While powerful, this approach both requires the availability of single-neuron data, and cannot give access to potential compensation mechanisms between single-neuron and synaptic parameters. Unlike **Prinz et al., 2004**, we apply SNPE to directly identify the full 31 dimensional parameter space without requiring experimental measurements of each individual neuron in the circuit. Despite the high-dimensional parameter space, SNPE can identify the posterior distribution using 18 million samples, whereas a direct application of a full-grid method would require $4.65 \cdot 10^{21}$ samples to fill the 31 dimensional parameter space on a grid with five values per dimension.

Finding paths in the posterior

In order to find directions of robust network output, we searched for a path of high posterior probability. First, as in **Prinz et al., 2004**, we aimed to find two similar model outputs with disparate parameters. To do so, we sampled from the posterior and searched for two parameter sets whose summary features were within 0.1 standard deviations of all 174,000 samples from the observed experimental data, but that had strongly disparate parameters from each other. In the following, we denote the obtained parameter sets by θ_s and θ_g.

Second, in order to identify whether network output can be maintained along a continuous path between these two samples, we searched for a connection in parameter space lying in regions of high posterior probability. To do so, we considered the connection between the samples as a path and minimized the following path integral:

$$\mathcal{L}(\gamma) = \int_0^1 -\log(p_{\theta|\mathbf{x}}(\gamma(s)|\mathbf{x_o}))\|\dot{\gamma}(s)\|ds. \tag{2}$$

To minimize this term, we parameterized the path $\gamma(s)$ using sinusoidal basis-functions with coefficients $\alpha_{n,k}$:

$$\gamma(s) = \begin{bmatrix} \sum_{k=1}^{K} \alpha_{1,k} \cdot \sin(\pi ks) \\ \vdots \\ \sum_{k=1}^{K} \alpha_{N,k} \cdot \sin(\pi ks) \end{bmatrix} + \begin{bmatrix} \sum_{k=K+1}^{2K} \alpha_{1,k} \cdot \sin^2(\pi ks) \\ \vdots \\ \sum_{k=K+1}^{2K} \alpha_{N,k} \cdot \sin^2(\pi ks) \end{bmatrix} + (1-s) \cdot \theta_s + s\theta_g$$

These basis functions are defined such that, for any coefficients $\alpha_{n,k}$, the start and end points of the path are exactly the two parameter sets defined above:

$$\gamma(0) = \theta_s \qquad \gamma(1) = \theta_g$$

With this formulation, we have framed the problem of finding the path as an unconstrained optimization problem over the parameters $\alpha_{n,k}$. We can therefore minimize the path integral $\mathcal{L}$ using gradient descent over $\alpha_{n,k}$. For numerical simulations, we approximated the integral in ***Equation 2*** as a sum over 80 points along the path and use two basis functions for each of the 31 dimensions, that is, $K = 2$.

In order to demonstrate the sensitivity of the pyloric network, we aimed to find a path along which the circuit output quickly breaks down. For this, we picked a starting point along the high-probability path and then minimized the posterior probability. In addition, we enforced that the orthogonal path lies within an orthogonal disk to the high-probability path, leading to the following constrained optimization problem:

$$\min_{\theta} \log(p(\theta|\mathbf{x})) \qquad \text{s.t.} \quad n^T \Delta\theta = 0$$

where n is the tangent vector along the path of high probability. This optimization problem can be solved using the gradient projection method (***Rosen, 1960***):

$$\Delta\theta = -\frac{P(\nabla \log(p(\theta|\mathbf{x})))}{\sqrt{(\nabla \log(p(\theta|\mathbf{x})))^T P(\nabla \log(p(\theta|\mathbf{x})))}}$$

with projection matrix $P = \mathbb{1} - \frac{1}{n^T n} nn^T$ and $\mathbb{1}$ indicating the identity matrix. Each gradient update is a step along the orthogonal path. We let the optimization run until the distance along the path is 1/27 of the distance along the high-probability path.

Identifying conditional correlations

In order to investigate compensation mechanisms in the STG, we compared marginal and conditional correlations. For the marginal correlation matrix in ***Figure 6b***, we calculated the Pearson correlation coefficient based on 1.26 million samples from the posterior distribution $p(\theta|\mathbf{x})$. To find the two-dimensional conditional distribution for any pair of parameters, we fixed all other parameters to values taken from an arbitrary posterior sample, and varied the remaining two on an evenly spaced grid with 50 points along each dimension, covering the entire prior space. We evaluated the posterior distribution at every value on this grid. We then calculated the conditional correlation as the Pearson correlation coefficient over this distribution. For the 1-dimensional conditional distribution, we varied only one parameter and kept all others fixed. Lastly, in ***Figure 6d***, we sampled 500 parameter sets from the posterior, computed the respective conditional posteriors and conditional correlation matrices, and took the average over the conditional correlation matrices.

Acknowledgements

We thank Mahmood S Hoseini and Michael Stryker for sharing their data for ***Figure 2***, and Philipp Berens, Sean Bittner, Jan Boelts, John Cunningham, Richard Gao, Scott Linderman, Eve Marder, Iain Murray, George Papamakarios, Astrid Prinz, Auguste Schulz and Srinivas Turaga for discussions and/or comments on the manuscript. This work was supported by the German Research Foundation (DFG) through SFB 1233 'Robust Vision', (276693517), SFB 1089 'Synaptic Microcircuits', SPP 2041 'Computational Connectomics' and Germany's Excellence Strategy – EXC-Number 2064/1 – Project number 390727645 and the German Federal Ministry of Education and Research (BMBF, project

'ADIMEM', FKZ 01IS18052 A-D) to JHM, a Sir Henry Dale Fellowship by the Wellcome Trust and the Royal Society (WT100000; WFP and TPV), a Wellcome Trust Senior Research Fellowship (214316/Z/18/Z; TPV), a ERC Consolidator Grant (SYNAPSEEK; WPF and CC), and a UK Research and Innovation, Biotechnology and Biological Sciences Research Council (CC, UKRI-BBSRC BB/N019512/1). We gratefully acknowledge the Leibniz Supercomputing Centre for funding this project by providing computing time on its Linux-Cluster.

Additional information

Funding

Funder	Grant reference number	Author
Deutsche Forschungsgemeinschaft	SFB 1233	Jan-Matthis Lueckmann Marcel Nonnenmacher Giacomo Bassetto Jakob H Macke
Deutsche Forschungsgemeinschaft	SFB 1089	Pedro J Gonçalves Jakob H Macke
Deutsche Forschungsgemeinschaft	SPP 2041	Jakob H Macke
Bundesministerium für Bildung und Forschung	01IS18052 A-D	Michael Deistler Marcel Nonnenmacher Jakob H Macke
H2020 European Research Council	SYNAPSEEK	Chaitanya Chintaluri William F Podlaski Tim P Vogels
Wellcome Trust Senior Research Fellowship	214316/Z/18/Z	Tim P Vogels
UK Research and Innovation	UKRI-BBSRC BB/N019512/1	Chaitanya Chintaluri
Deutsche Forschungsgemeinschaft	Germany's Excellence Strategy - EXC-Number 2064/1 610 - Project number 390727645	Jakob H Macke
Wellcome Trust	Sir Henry Dale Fellowship WT100000	William F Podlaski Tim P Vogels
Royal Society	Sir Henry Dale Fellowship WT100000	William F Podlaski Tim P Vogels

The funders had no role in study design, data collection and interpretation, or the decision to submit the work for publication.

Author contributions

Pedro J Gonçalves, Conceptualization, Data curation, Software, Formal analysis, Supervision, Validation, Investigation, Visualization, Methodology, Writing - original draft, Project administration, Writing - review and editing; Jan-Matthis Lueckmann, Michael Deistler, Conceptualization, Data curation, Software, Formal analysis, Validation, Investigation, Visualization, Methodology, Writing - original draft, Writing - review and editing; Marcel Nonnenmacher, Data curation, Software, Formal analysis, Validation, Investigation, Visualization, Methodology, Writing - review and editing; Kaan Öcal, Software, Investigation, Visualization, Methodology, Writing - review and editing; Giacomo Bassetto, Software, Visualization, Methodology, Writing - review and editing; Chaitanya Chintaluri, William F Podlaski, Sara A Haddad, Resources, Data curation, Writing - review and editing; Tim P Vogels, Conceptualization, Resources, Supervision, Funding acquisition, Project administration, Writing - review and editing; David S Greenberg, Data curation, Software, Formal analysis, Supervision, Validation, Investigation, Visualization, Methodology, Writing - original draft, Writing - review and editing; Jakob H Macke, Conceptualization, Supervision, Funding acquisition, Validation, Visualization, Methodology, Writing - original draft, Project administration, Writing - review and editing

Author ORCIDs
Pedro J Gonçalves https://orcid.org/0000-0002-6987-4836
Jan-Matthis Lueckmann https://orcid.org/0000-0003-4320-4663
Michael Deistler https://orcid.org/0000-0002-3573-0404
Kaan Öcal http://orcid.org/0000-0002-8528-6858
Chaitanya Chintaluri http://orcid.org/0000-0003-4252-1608
William F Podlaski http://orcid.org/0000-0001-6619-7502
Sara A Haddad https://orcid.org/0000-0003-0807-0823
Jakob H Macke https://orcid.org/0000-0001-5154-8912

Decision letter and Author response
Decision letter https://doi.org/10.7554/eLife.56261.sa1
Author response https://doi.org/10.7554/eLife.56261.sa2

Additional files

Supplementary files

- Transparent reporting form

Data availability

The contributions of the work are primarily the development and application of computational models, no new data has been obtained or is being published. All code and associated data are available at https://github.com/mackelab/delfi/ (archived at https://archive.softwareheritage.org/swh:1:rev:62a99a879145bdc675917fc33eed69293b964048;origin=https://github.com/mackelab/delfi/;visit=swh:1:snp:95a69f58c195feb5660760cd4ab833de366f5441/) and https://github.com/mackelab/IdentifyMechanisticModels_2020 (archived at https://archive.softwareheritage.org/swh:1:rev:b93c90ec6156ae5f8afee6aaac7317373e9caf5e;origin=https://github.com/mackelab/IdentifyMechanisticModels_2020;visit=swh:1:snp:20bfde9fbbb134657b75bc29ab4905a2ef3d5b17/).

The following datasets were generated:

References

Abbott L, Marder E. 1998. *Modeling Small Networks.* MIT Press.
Achard P, De Schutter E. 2006. Complex parameter landscape for a complex neuron model. *PLOS Computational Biology* **2**:e94. DOI: https://doi.org/10.1371/journal.pcbi.0020094, PMID: 16848639
Allen Institute for Brain Science. 2016. Allen cell types database. http://celltypes.brain-map.org/ [Accessed June 8, 2018].
Alonso LM, Marder E. 2019. Visualization of currents in neural models with similar behavior and different conductance densities. *eLife* **8**:e42722. DOI: https://doi.org/10.7554/eLife.42722, PMID: 30702427
Baker RE, Peña JM, Jayamohan J, Jérusalem A. 2018. Mechanistic models versus machine learning, a fight worth fighting for the biological community? *Biology Letters* **14**:20170660. DOI: https://doi.org/10.1098/rsbl.2017.0660, PMID: 29769297
Barthelmé S, Chopin N. 2014. Expectation propagation for Likelihood-Free inference. *Journal of the American Statistical Association* **109**:315–333. DOI: https://doi.org/10.1080/01621459.2013.864178
Bassett DS, Zurn P, Gold JI. 2018. On the nature and use of models in network neuroscience. *Nature Reviews Neuroscience* **19**:566–578. DOI: https://doi.org/10.1038/s41583-018-0038-8, PMID: 30002509
Beaumont MA, Zhang W, Balding DJ. 2002. Approximate bayesian computation in population genetics. *Genetics* **162**:2025–2035. PMID: 12524368
Beaumont MA, Cornuet J-M, Marin J-M, Robert CP. 2009. Adaptive approximate bayesian computation. *Biometrika* **96**:983–990. DOI: https://doi.org/10.1093/biomet/asp052
Ben-Shalom R, Balewski J, Siththaranjan A, Baratham V, Kyoung H, Kim KG, Bender KJ, Bouchard KE. 2019. Inferring neuronal ionic conductances from membrane potentials using cnns. *bioRxiv*. DOI: https://doi.org/10.1101/727974
Bishop CM. 1994. *Mixture Density Networks, Technical Report*: Aston University.
Bittner SR, Palmigiano A, Piet AT, Duan CA, Brody CD, Miller KD, Cunningham JP. 2019. Interrogating theoretical models of neural computation with deep inference. *bioRxiv*. DOI: https://doi.org/10.1101/837567
Bleuler S, Laumanns M, Thiele L, Zitzler E. 2003. Pisa'a platform and programming language independent interface for search algorithms. International Conference on Evolutionary Multi-Criterion Optimization 494–508.
Blum MGB, Nunes MA, Prangle D, Sisson SA. 2013. A comparative review of dimension reduction methods in approximate bayesian computation. *Statistical Science* **28**:189–208. DOI: https://doi.org/10.1214/12-STS406

Blum MGB, François O. 2010. Non-linear regression models for approximate bayesian computation. *Statistics and Computing* **20**:63–73. DOI: https://doi.org/10.1007/s11222-009-9116-0

Brette R. 2015. What is the most realistic single-compartment model of spike initiation? *PLOS Computational Biology* **11**:e1004114. DOI: https://doi.org/10.1371/journal.pcbi.1004114, PMID: 25856629

Britton OJ, Bueno-Orovio A, Van Ammel K, Lu HR, Towart R, Gallacher DJ, Rodriguez B. 2013. Experimentally calibrated population of models predicts and explains intersubject variability in cardiac cellular electrophysiology. *PNAS* **110**:E2098–E2105. DOI: https://doi.org/10.1073/pnas.1304382110, PMID: 23690584

Brown EN, Frank LM, Tang D, Quirk MC, Wilson MA. 1998. A statistical paradigm for neural spike train decoding applied to position prediction from ensemble firing patterns of rat hippocampal place cells. *The Journal of Neuroscience* **18**:7411–7425. DOI: https://doi.org/10.1523/JNEUROSCI.18-18-07411.1998, PMID: 9736661

Caplan JS, Williams AH, Marder E. 2014. Many parameter sets in a multicompartment model oscillator are robust to temperature perturbations. *The Journal of Neuroscience* **34**:4963–4975. DOI: https://doi.org/10.1523/JNEUROSCI.0280-14.2014, PMID: 24695714

Casado ML, Baydin AG, Rubio DM, Le TA, Wood F, Heinrich L, Louppe G, Cranmer K, Ng K, Bhimji W. 2017. Improvements to inference compilation for probabilistic programming in large-scale scientific simulators. NeurIPS Workshop on Deep Learning for Physical Sciences.

Chan J, Perrone V, Spence J, Jenkins P, Mathieson S, Song Y. 2018. A likelihood-free inference framework for population genetic data using exchangeable neural networks. Advances in Neural Information Processing Systems 8594–8605.

Chen Z. 2003. Bayesian filtering: from Kalman filters to particle filters, and beyond. *Statistics* **182**:1–69. DOI: https://doi.org/10.1080/02331880309257

Chichilnisky EJ. 2001. A simple white noise analysis of neuronal light responses. *Network: Computation in Neural Systems* **12**:199–213. DOI: https://doi.org/10.1080/713663221

Cook SR, Gelman A, Rubin DB. 2006. Validation of software for bayesian models using posterior quantiles. *Journal of Computational and Graphical Statistics* **15**:675–692. DOI: https://doi.org/10.1198/106186006X136976

Costa RP, Sjöström PJ, van Rossum MC. 2013. Probabilistic inference of short-term synaptic plasticity in neocortical microcircuits. *Frontiers in Computational Neuroscience* **7**:75. DOI: https://doi.org/10.3389/fncom.2013.00075, PMID: 23761760

Cranmer K, Brehmer J, Louppe G. 2020. The frontier of simulation-based inference. *PNAS* **46**:201912789. DOI: https://doi.org/10.1073/pnas.1912789117

Cunningham JP, Yu BM. 2014. Dimensionality reduction for large-scale neural recordings. *Nature Neuroscience* **17**:1500–1509. DOI: https://doi.org/10.1038/nn.3776, PMID: 25151264

Daly AC, Gavaghan DJ, Holmes C, Cooper J. 2015. Hodgkin-Huxley revisited: reparametrization and identifiability analysis of the classic action potential model with approximate bayesian methods. *Royal Society Open Science* **2**:150499. DOI: https://doi.org/10.1098/rsos.150499, PMID: 27019736

De Nicolao G, Sparacino G, Cobelli C. 1997. Nonparametric input estimation in physiological systems: problems, methods, and case studies. *Automatica* **33**:851–870. DOI: https://doi.org/10.1016/S0005-1098(96)00254-3

Destexhe A, Huguenard JR. 2000. Nonlinear thermodynamic models of voltage-dependent currents. *Journal of Computational Neuroscience* **9**:259–270. DOI: https://doi.org/10.1023/a:1026535704537, PMID: 11139042

Druckmann S, Banitt Y, Gidon A, Schürmann F, Markram H, Segev I. 2007. A novel multiple objective optimization framework for constraining conductance-based neuron models by experimental data. *Frontiers in Neuroscience* **1**:7–18. DOI: https://doi.org/10.3389/neuro.01.1.1.001.2007, PMID: 18982116

Dunlop J, Bowlby M, Peri R, Vasilyev D, Arias R. 2008. High-throughput electrophysiology: an emerging paradigm for ion-channel screening and physiology. *Nature Reviews Drug Discovery* **7**:358–368. DOI: https://doi.org/10.1038/nrd2552, PMID: 18356919

Durkan C, Papamakarios G, Murray I. 2018. Sequential neural methods for likelihood-free inference. NeurIPS Bayesian Deep Learning Workshop.

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

Dyballa L, Hoseini MS, Dadarlat MC, Zucker SW, Stryker MP. 2018. Flow stimuli reveal ecologically appropriate responses in mouse visual cortex. *PNAS* **115**:11304–11309. DOI: https://doi.org/10.1073/pnas.1811265115, PMID: 30327345

Fisher D, Olasagasti I, Tank DW, Aksay ER, Goldman MS. 2013. A modeling framework for deriving the structural and functional architecture of a short-term memory microcircuit. *Neuron* **79**:987–1000. DOI: https://doi.org/10.1016/j.neuron.2013.06.041, PMID: 24012010

Foster WR, Ungar LH, Schwaber JS. 1993. Significance of conductances in Hodgkin-Huxley models. *Journal of Neurophysiology* **70**:2502–2518. DOI: https://doi.org/10.1152/jn.1993.70.6.2502, PMID: 7509859

Gerstner W, Sprekeler H, Deco G. 2012. Theory and simulation in neuroscience. *Science* **338**:60–65. DOI: https://doi.org/10.1126/science.1227356, PMID: 23042882

Gerwinn S, Macke JH, Bethge M. 2010. Bayesian inference for generalized linear models for spiking neurons. *Frontiers in Computational Neuroscience* **4**:12. DOI: https://doi.org/10.3389/fncom.2010.00012, PMID: 20577627

Gold JI, Shadlen MN. 2007. The neural basis of decision making. *Annual Review of Neuroscience* **30**:535–574. DOI: https://doi.org/10.1146/annurev.neuro.29.051605.113038, PMID: 17600525

Goldman MS, Golowasch J, Marder E, Abbott LF. 2001. Global structure, robustness, and modulation of neuronal models. *The Journal of Neuroscience* **21**:5229–5238. DOI: https://doi.org/10.1523/JNEUROSCI.21-14-05229.2001, PMID: 11438598

Golowasch J, Goldman MS, Abbott LF, Marder E. 2002. Failure of averaging in the construction of a conductance-based neuron model. *Journal of Neurophysiology* **87**:1129–1131. DOI: https://doi.org/10.1152/jn.00412.2001, PMID: 11826077

Gouwens NW, Berg J, Feng D, Sorensen SA, Zeng H, Hawrylycz MJ, Koch C, Arkhipov A. 2018. Systematic generation of biophysically detailed models for diverse cortical neuron types. *Nature Communications* **9**:710. DOI: https://doi.org/10.1038/s41467-017-02718-3, PMID: 29459718

Grashow R, Brookings T, Marder E. 2010. Compensation for variable intrinsic neuronal excitability by circuit-synaptic interactions. *Journal of Neuroscience* **30**:9145–9156. DOI: https://doi.org/10.1523/JNEUROSCI.0980-10.2010, PMID: 20610748

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

Gutenkunst RN, Waterfall JJ, Casey FP, Brown KS, Myers CR, Sethna JP. 2007. Universally sloppy parameter sensitivities in systems biology models. *PLOS Computational Biology* **3**:e189. DOI: https://doi.org/10.1371/journal.pcbi.0030189

Gutierrez GJ, O'Leary T, Marder E. 2013. Multiple mechanisms switch an electrically coupled, synaptically inhibited neuron between competing rhythmic oscillators. *Neuron* **77**:845–858. DOI: https://doi.org/10.1016/j.neuron.2013.01.016, PMID: 23473315

Gutmann MU, Corander J. 2016. Bayesian optimization for likelihood-free inference of simulator-based statistical models. *The Journal of Machine Learning Research* **17**:4256–4302.

Haddad SA, Marder E. 2018. Circuit robustness to temperature perturbation is altered by neuromodulators. *Neuron* **100**:609–623. DOI: https://doi.org/10.1016/j.neuron.2018.08.035, PMID: 30244886

Hay E, Hill S, Schürmann F, Markram H, Segev I. 2011. Models of neocortical layer 5b pyramidal cells capturing a wide range of dendritic and perisomatic active properties. *PLOS Computational Biology* **7**:e1002107. DOI: https://doi.org/10.1371/journal.pcbi.1002107, PMID: 21829333

Hermans J, Begy V, Louppe G. 2020. Likelihood-free mcmc with approximate likelihood ratios. International Conference on Machine Learning.

Hertäg L, Hass J, Golovko T, Durstewitz D. 2012. An approximation to the adaptive exponential Integrate-and-Fire neuron model allows fast and predictive fitting to physiological data. *Frontiers in Computational Neuroscience* **6**:62. DOI: https://doi.org/10.3389/fncom.2012.00062, PMID: 22973220

Herz AV, Gollisch T, Machens CK, Jaeger D. 2006. Modeling single-neuron dynamics and computations: a balance of detail and abstraction. *Science* **314**:80–85. DOI: https://doi.org/10.1126/science.1127240, PMID: 17023649

Hodgkin AL, Huxley AF. 1952. A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of Physiology* **117**:500–544. DOI: https://doi.org/10.1113/jphysiol.1952.sp004764, PMID: 12991237

Huys QJ, Ahrens MB, Paninski L. 2006. Efficient estimation of detailed single-neuron models. *Journal of Neurophysiology* **96**:872–890. DOI: https://doi.org/10.1152/jn.00079.2006, PMID: 16624998

Huys QJ, Paninski L. 2009. Smoothing of, and parameter estimation from, noisy biophysical recordings. *PLOS Computational Biology* **5**:e1000379. DOI: https://doi.org/10.1371/journal.pcbi.1000379, PMID: 19424506

Izbicki R, Lee AB, Pospisil T. 2019. ABC–CDE: Toward Approximate Bayesian Computation With Complex High-Dimensional Data and Limited Simulations. *Journal of Computational and Graphical Statistics* **28**:481–492. DOI: https://doi.org/10.1080/10618600.2018.1546594

Jiang B, Wu T-y, Zheng C, Wong WH. 2017. Learning summary statistic for approximate bayesian computation via deep neural network. *Statistica Sinica* **27**:1595–1618. DOI: https://doi.org/10.5705/ss.202015.0340

Jones JP, Palmer LA. 1987. An evaluation of the two-dimensional gabor filter model of simple receptive fields in cat striate cortex. *Journal of Neurophysiology* **58**:1233–1258. DOI: https://doi.org/10.1152/jn.1987.58.6.1233, PMID: 3437332

Kingma DP, Ba J. 2014. Adam: a method for stochastic optimization. International Conference on Learning Representations.

Kitano H. 2004. Biological robustness. *Nature Reviews Genetics* **5**:826–837. DOI: https://doi.org/10.1038/nrg1471, PMID: 15520792

Kleinegesse S, Gutmann MU. 2019. Efficient bayesian experimental design for implicit models. The 22nd International Conference on Artificial Intelligence and Statistics 476–485.

Krizhevsky A, Sutskever I, Hinton GE. 2012. Imagenet classification with deep convolutional neural networks. Advances in Neural Information Processing Systems 1097–1105.

Ladenbauer J, McKenzie S, English DF, Hagens O, Ostojic S. 2018. Inferring and validating mechanistic models of neural microcircuits based on spike-train data. *bioRxiv*. DOI: https://doi.org/10.1101/261016

Lawson BAJ, Drovandi CC, Cusimano N, Burrage P, Rodriguez B, Burrage K. 2018. Unlocking data sets by calibrating populations of models to data density: a study in atrial electrophysiology. *Science Advances* **4**:e1701676. DOI: https://doi.org/10.1126/sciadv.1701676, PMID: 29349296

Le TA, Baydin AG, Zinkov R, Wood F. 2017a. Using synthetic data to train neural networks is model-based reasoning, in 2017. International Joint Conference on Neural Networks (IJCNN) IEEE 3514–3521. DOI: https://doi.org/10.1109/IJCNN.2017.7966298

Le TA, Baydin AG, Wood F. 2017b. Inference compilation and universal probabilistic programming. Artificial Intelligence and Statistics 1338–1348.
Liepe J, Kirk P, Filippi S, Toni T, Barnes CP, Stumpf MP. 2014. A framework for parameter estimation and model selection from experimental data in systems biology using approximate bayesian computation. *Nature Protocols* **9**:439–456. DOI: https://doi.org/10.1038/nprot.2014.025, PMID: 24457334
Litwin-Kumar A, Doiron B. 2012. Slow dynamics and high variability in balanced cortical networks with clustered connections. *Nature Neuroscience* **15**:1498–1505. DOI: https://doi.org/10.1038/nn.3220, PMID: 23001062
Loaiza-Ganem G, Gao Y, Cunningham JP. 2017. Maximum entropy flow networks. 5th International Conference on Learning Representations, ICLR.
Lueckmann J-M, Goncalves PJ, Bassetto G, Öcal K, Nonnenmacher M, Macke JH. 2017. Flexible statistical inference for mechanistic models of neural dynamics. Advances in Neural Information Processing Systems 1289–1299.
Lueckmann J-M, Bassetto G, Karaletsos T, Macke JH. 2019. Likelihood-free inference with emulator networks. Proceedings of the 1st Symposium on Advances in Approximate Bayesian Inference, Volume 96 of Proceedings of Machine Learning Research 32–53.
Machta BB, Chachra R, Transtrum MK, Sethna JP. 2013. Parameter space compression underlies emergent theories and predictive models. *Science* **342**:604–607. DOI: https://doi.org/10.1126/science.1238723, PMID: 24179222
Macke JH, Buesing L, Cunningham JP, Yu BM, Shenoy KV, Sahani M. 2011. Empirical models of spiking in neural populations. Advances in Neural Information Processing Systems 1350–1358.
MacLean JN, Zhang Y, Johnson BR, Harris-Warrick RM. 2003. Activity-independent homeostasis in rhythmically active neurons. *Neuron* **37**:109–120. DOI: https://doi.org/10.1016/S0896-6273(02)01104-2, PMID: 12526777
MacLean JN, Zhang Y, Goeritz ML, Casey R, Oliva R, Guckenheimer J, Harris-Warrick RM. 2005. Activity-independent coregulation of IA and ih in rhythmically active neurons. *Journal of Neurophysiology* **94**:3601–3617. DOI: https://doi.org/10.1152/jn.00281.2005, PMID: 16049145
Maheswaranathan N, Williams A, Golub MD, Ganguli S, Sussillo D. 2019. Reverse engineering recurrent networks for sentiment classification reveals line attractor dynamics. Advances in Neural Information Processing Systems 15696 15705.
Mante V, Sussillo D, Shenoy KV, Newsome WT. 2013. Context-dependent computation by recurrent dynamics in prefrontal cortex. *Nature* **503**:78–84. DOI: https://doi.org/10.1038/nature12742, PMID: 24201281
Marder E. 2011. Variability, compensation, and modulation in neurons and circuits. *PNAS* **108 Suppl 3**:15542–15548. DOI: https://doi.org/10.1073/pnas.1010674108, PMID: 21383190
Marder E, Goeritz ML, Otopalik AG. 2015. Robust circuit rhythms in small circuits arise from variable circuit components and mechanisms. *Current Opinion in Neurobiology* **31**:156–163. DOI: https://doi.org/10.1016/j.conb.2014.10.012, PMID: 25460072
Marder E, Goaillard JM. 2006. Variability, compensation and homeostasis in neuron and network function. *Nature Reviews Neuroscience* **7**:563–574. DOI: https://doi.org/10.1038/nrn1949, PMID: 16791145
Marder E, Taylor AL. 2011. Multiple models to capture the variability in biological neurons and networks. *Nature Neuroscience* **14**:133–138. DOI: https://doi.org/10.1038/nn.2735, PMID: 21270780
Marjoram P, Molitor J, Plagnol V, Tavare S. 2003. Markov chain monte carlo without likelihoods. *PNAS* **100**: 15324–15328. DOI: https://doi.org/10.1073/pnas.0306899100, PMID: 14663152
McTavish TS, Migliore M, Shepherd GM, Hines ML. 2012. Mitral cell spike synchrony modulated by dendrodendritic synapse location. *Frontiers in Computational Neuroscience* **6**:3. DOI: https://doi.org/10.3389/fncom.2012.00003, PMID: 22319487
Meeds E, Welling M. 2014. Gps-abc: gaussian process surrogate approximate bayesian computation. Conference on Uncertainty in Artificial Intelligence.
Meliza CD, Kostuk M, Huang H, Nogaret A, Margoliash D, Abarbanel HD. 2014. Estimating parameters and predicting membrane voltages with conductance-based neuron models. *Biological Cybernetics* **108**:495–516. DOI: https://doi.org/10.1007/s00422-014-0615-5, PMID: 24962080
Niell CM, Stryker MP. 2008. Highly selective receptive fields in mouse visual cortex. *Journal of Neuroscience* **28**: 7520–7536. DOI: https://doi.org/10.1523/JNEUROSCI.0623-08.2008, PMID: 18650330
O'Leary T, Williams AH, Franci A, Marder E. 2014. Cell types, network homeostasis, and pathological compensation from a biologically plausible ion channel expression model. *Neuron* **82**:809–821. DOI: https://doi.org/10.1016/j.neuron.2014.04.002, PMID: 24853940
O'Leary T, Sutton AC, Marder E. 2015. Computational models in the age of large datasets. *Current Opinion in Neurobiology* **32**:87–94. DOI: https://doi.org/10.1016/j.conb.2015.01.006, PMID: 25637959
O'Leary T, Marder E. 2016. Temperature-Robust neural function from Activity-Dependent ion channel regulation. *Current Biology* **26**:2935–2941. DOI: https://doi.org/10.1016/j.cub.2016.08.061, PMID: 27746024
Oesterle J, Behrens C, Schroeder C, Herrmann T, Euler T, Franke K, Smith RG, Zeck G, Berens P. 2020. Bayesian inference for biophysical neuron models enables stimulus optimization for retinal neuroprosthetics. *bioRxiv*. DOI: https://doi.org/10.1101/2020.01.08.898759
O'Leary T. 2018. Homeostasis, failure of homeostasis and degenerate ion channel regulation. *Current Opinion in Physiology* **2**:129–138. DOI: https://doi.org/10.1016/j.cophys.2018.01.006
Pandarinath C, O'Shea DJ, Collins J, Jozefowicz R, Stavisky SD, Kao JC, Trautmann EM, Kaufman MT, Ryu SI, Hochberg LR, Henderson JM, Shenoy KV, Abbott LF, Sussillo D. 2018. Inferring single-trial neural population dynamics using sequential auto-encoders. *Nature Methods* **15**:805–815. DOI: https://doi.org/10.1038/s41592-018-0109-9, PMID: 30224673

Paninski L. 2004. Maximum likelihood estimation of cascade point-process neural encoding models. *Network: Computation in Neural Systems* **15**:243–262. DOI: https://doi.org/10.1088/0954-898X_15_4_002, PMID: 15600233

Papamakarios G, Pavlakou T, Murray I. 2017. Masked autoregressive flow for density estimation. Advances in Neural Information Processing Systems 2338–2347.

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

Papamakarios G, Nalisnick E, Rezende DJ, Mohamed S, Lakshminarayanan B. 2019b. Normalizing flows for probabilistic modeling and inference. *arXiv*. https://arxiv.org/abs/1912.02762.

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

Pillow JW, Paninski L, Uzzell VJ, Simoncelli EP, Chichilnisky EJ. 2005. Prediction and decoding of retinal ganglion cell responses with a probabilistic spiking model. *Journal of Neuroscience* **25**:11003–11013. DOI: https://doi.org/10.1523/JNEUROSCI.3305-05.2005, PMID: 16306413

Pillow J. 2007. Likelihood-based approaches to modeling the neural code. Bayesian Brain: Probabilistic Approaches to Neural Coding 53–70. DOI: https://doi.org/10.7551/mitpress/9780262042383.003.0003

Pillow JW, Shlens J, Paninski L, Sher A, Litke AM, Chichilnisky EJ, Simoncelli EP. 2008. Spatio-temporal correlations and visual signalling in a complete neuronal population. *Nature* **454**:995–999. DOI: https://doi.org/10.1038/nature07140, PMID: 18650810

Pillow JW, Scott J. 2012. Fully bayesian inference for neural models with negative-binomial spiking. Advances in Neural Information Processing Systems 1898–1906.

Podlaski WF, Seeholzer A, Groschner LN, Miesenböck G, Ranjan R, Vogels TP. 2017. Mapping the function of neuronal ion channels in model and experiment. *eLife* **6**:e22152. DOI: https://doi.org/10.7554/eLife.22152, PMID: 28267430

Polson NG, Scott JG, Windle J. 2013. Bayesian inference for logistic models using Pólya–Gamma Latent Variables. *Journal of the American Statistical Association* **108**:1339–1349. DOI: https://doi.org/10.1080/01621459.2013.829001

Pospischil M, Toledo-Rodriguez M, Monier C, Piwkowska Z, Bal T, Frégnac Y, Markram H, Destexhe A. 2008. Minimal Hodgkin-Huxley type models for different classes of cortical and thalamic neurons. *Biological Cybernetics* **99**:427–441. DOI: https://doi.org/10.1007/s00422-008-0263-8, PMID: 19011929

Potjans TC, Diesmann M. 2014. The cell-type specific cortical microcircuit: relating structure and activity in a full-scale spiking network model. *Cerebral Cortex* **24**:785–806. DOI: https://doi.org/10.1093/cercor/bhs358, PMID: 23203991

Pozzorini C, Mensi S, Hagens O, Naud R, Koch C, Gerstner W. 2015. Automated High-Throughput characterization of single neurons by means of simplified spiking models. *PLOS Computational Biology* **11**: e1004275. DOI: https://doi.org/10.1371/journal.pcbi.1004275, PMID: 26083597

Prinz AA, Billimoria CP, Marder E. 2003. Alternative to hand-tuning conductance-based models: construction and analysis of databases of model neurons. *Journal of Neurophysiology* **90**:3998–4015. DOI: https://doi.org/10.1152/jn.00641.2003, PMID: 12944532

Prinz AA, Bucher D, Marder E. 2004. Similar network activity from disparate circuit parameters. *Nature Neuroscience* **7**:1345–1352. DOI: https://doi.org/10.1038/nn1352, PMID: 15558066

Pritchard JK, Seielstad MT, Perez-Lezaun A, Feldman MW. 1999. Population growth of human Y chromosomes: a study of Y chromosome microsatellites. *Molecular Biology and Evolution* **16**:1791–1798. DOI: https://doi.org/10.1093/oxfordjournals.molbev.a026091, PMID: 10605120

Ranjan R, Logette E, Marani M, Herzog M, Tâche V, Scantamburlo E, Buchillier V, Markram H. 2019. A kinetic map of the homomeric Voltage-Gated potassium channel (Kv) Family. *Frontiers in Cellular Neuroscience* **13**: 358. DOI: https://doi.org/10.3389/fncel.2019.00358, PMID: 31481875

Ratcliff R, McKoon G. 2008. The diffusion decision model: theory and data for two-choice decision tasks. *Neural Computation* **20**:873–922. DOI: https://doi.org/10.1162/neco.2008.12-06-420, PMID: 18085991

René A, Longtin A, Macke JH. 2020. Inference of a mesoscopic population model from population spike trains. *Neural Computation* **32**:1448–1498. DOI: https://doi.org/10.1162/neco_a_01292

Rezende DJ, Mohamed S. 2015. Variational inference with normalizing flows. Proceedings of the 32nd International Conference on International Conference on Machine Learning 1530–1538.

Rosen JB. 1960. The gradient projection method for nonlinear programming. Part I. Linear constraints. *Journal of the Society for Industrial and Applied Mathematics* **8**:181–217. DOI: https://doi.org/10.1137/0108011

Rossant C, Goodman DF, Fontaine B, Platkiewicz J, Magnusson AK, Brette R. 2011. Fitting neuron models to spike trains. *Frontiers in Neuroscience* **5**:9. DOI: https://doi.org/10.3389/fnins.2011.00009, PMID: 21415925

Rubin DB. 1984. Bayesianly justifiable and relevant frequency calculations for the applied statistician. *The Annals of Statistics* **12**:1151–1172. DOI: https://doi.org/10.1214/aos/1176346785

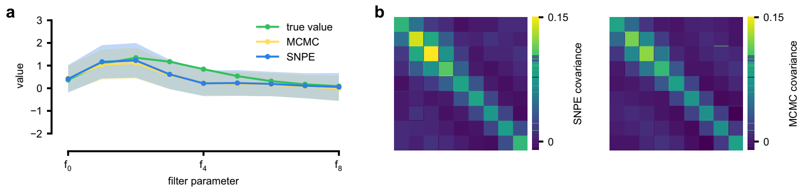
Rubin DB, Van Hooser SD, Miller KD. 2015. The stabilized supralinear network: a unifying circuit motif underlying multi-input integration in sensory cortex. *Neuron* **85**:402–417. DOI: https://doi.org/10.1016/j.neuron.2014.12.026, PMID: 25611511

Schneidman E, Berry MJ, Segev R, Bialek W. 2006. Weak pairwise correlations imply strongly correlated network states in a neural population. *Nature* **440**:1007–1012. DOI: https://doi.org/10.1038/nature04701, PMID: 16625187

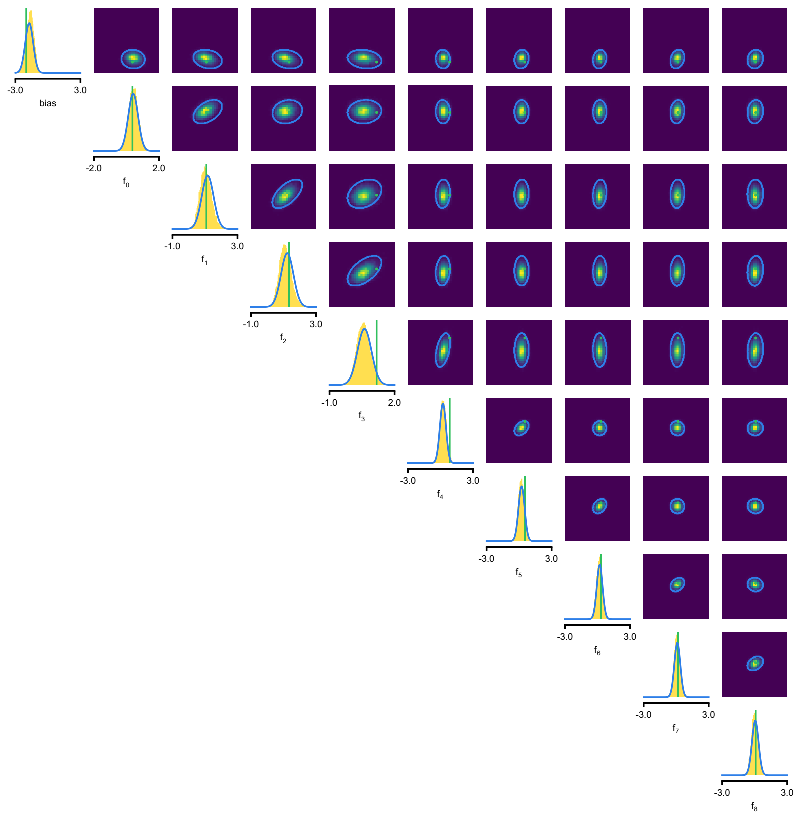
Schröder C, Lagnado L, James B, Berens P. 2019. Approximate bayesian inference for a mechanistic model of vesicle release at a ribbon synapse. *bioRxiv*. DOI: https://doi.org/10.1101/669218

Simonyan K, Zisserman A. 2015. Very deep convolutional networks for large-scale image recognition. International Conference on Learning Representations.
Sisson SA, Fan Y, Tanaka MM. 2007. Sequential monte carlo without likelihoods. *PNAS* **104**:1760–1765. DOI: https://doi.org/10.1073/pnas.0607208104, PMID: 17264216
Speiser A, Yan J, Archer EW, Buesing L, Turaga SC, Macke JH. 2017. Fast amortized inference of neural activity from calcium imaging data with variational autoencoders. Advances in Neural Information Processing Systems 4024–4034.
Sporns O. 2014. Contributions and challenges for network models in cognitive neuroscience. *Nature Neuroscience* **17**:652–660. DOI: https://doi.org/10.1038/nn.3690, PMID: 24686784
Stringer C, Pachitariu M, Steinmetz NA, Okun M, Bartho P, Harris KD, Sahani M, Lesica NA. 2016. Inhibitory control of correlated intrinsic variability in cortical networks. *eLife* **5**:e19695. DOI: https://doi.org/10.7554/eLife.19695, PMID: 27926356
Suk HJ, Boyden ES, van Welie I. 2019. Advances in the automation of whole-cell patch clamp technology. *Journal of Neuroscience Methods* **326**:108357. DOI: https://doi.org/10.1016/j.jneumeth.2019.108357, PMID: 31336060
Sussillo D, Abbott LF. 2009. Generating coherent patterns of activity from chaotic neural networks. *Neuron* **63**:544–557. DOI: https://doi.org/10.1016/j.neuron.2009.07.018, PMID: 19709635
Sussillo D, Barak O. 2013. Opening the black box: low-dimensional dynamics in high-dimensional recurrent neural networks. *Neural Computation* **25**:626–649. DOI: https://doi.org/10.1162/NECO_a_00409, PMID: 23272922
Talts S, Betancourt M, Simpson D, Vehtari A, Gelman A. 2018. Validating bayesian inference algorithms with simulation-based calibration. *arXiv*. https://arxiv.org/abs/1804.06788.
Taylor AL, Hickey TJ, Prinz AA, Marder E. 2006. Structure and visualization of high-dimensional conductance spaces. *Journal of Neurophysiology* **96**:891–905. DOI: https://doi.org/10.1152/jn.00367.2006, PMID: 16687617
Taylor AL, Goaillard JM, Marder E. 2009. How multiple conductances determine electrophysiological properties in a multicompartment model. *Journal of Neuroscience* **29**:5573–5586. DOI: https://doi.org/10.1523/JNEUROSCI.4438-08.2009, PMID: 19403824
Teeter C, Iyer R, Menon V, Gouwens N, Feng D, Berg J, Szafer A, Cain N, Zeng H, Hawrylycz M, Koch C, Mihalas S. 2018. Generalized leaky integrate-and-fire models classify multiple neuron types. *Nature Communications* **9**:709. DOI: https://doi.org/10.1038/s41467-017-02717-4, PMID: 29459723
Tejero-Cantero A, Boelts J, Deistler M, Lueckmann J-M, Durkan C, Gonçalves PJ, Greenberg DS, Macke JH. 2020. Sbi: a toolkit for simulation-based inference. *Journal of Open Source Software* **5**:2505. DOI: https://doi.org/10.21105/joss.02505
Tomm C, Avermann M, Vogels T, Gerstner W, Petersen C. 2011. The influence of structure on the response properties of biologically plausible neural network models. *BMC Neuroscience* **12**:P30. DOI: https://doi.org/10.1186/1471-2202-12-S1-P30
Truccolo W, Eden UT, Fellows MR, Donoghue JP, Brown EN. 2005. A point process framework for relating neural spiking activity to spiking history, neural ensemble, and extrinsic covariate effects. *Journal of Neurophysiology* **93**:1074–1089. DOI: https://doi.org/10.1152/jn.00697.2004, PMID: 15356183
Van Geit W, Gevaert M, Chindemi G, Rössert C, Courcol JD, Muller EB, Schürmann F, Segev I, Markram H. 2016. BluePyOpt: leveraging open source software and cloud infrastructure to optimise model parameters in neuroscience. *Frontiers in Neuroinformatics* **10**:17. DOI: https://doi.org/10.3389/fninf.2016.00017, PMID: 27375471
van Vreeswijk C, Sompolinsky H. 1996. Chaos in neuronal networks with balanced excitatory and inhibitory activity. *Science* **274**:1724–1726. DOI: https://doi.org/10.1126/science.274.5293.1724, PMID: 8939866
Vogels TP, Rajan K, Abbott LF. 2005. Neural network dynamics. *Annual Review of Neuroscience* **28**:357–376. DOI: https://doi.org/10.1146/annurev.neuro.28.061604.135637, PMID: 16022600
Wang XJ. 2008. Decision making in recurrent neuronal circuits. *Neuron* **60**:215–234. DOI: https://doi.org/10.1016/j.neuron.2008.09.034, PMID: 18957215
Webb S, Golinski A, Zinkov R, Narayanaswamy S, Rainforth T, Teh YW, Wood F. 2018. Faithful inversion of generative models for effective amortized inference. Advances in Neural Information Processing Systems 3070–3080.
Wilkinson R. 2014. Accelerating abc methods using gaussian processes. AISTATS.
Wood SN. 2010. Statistical inference for noisy nonlinear ecological dynamic systems. *Nature* **466**:1102–1104. DOI: https://doi.org/10.1038/nature09319, PMID: 20703226
Yu BM, Cunningham JP, Santhanam G, Ryu SI, Shenoy KV, Sahani M. 2009. Gaussian-process factor analysis for low-dimensional single-trial analysis of neural population activity. *Journal of Neurophysiology* **102**:614–635. DOI: https://doi.org/10.1152/jn.90941.2008, PMID: 19357332
Zitzler E, Künzli S. 2004. Indicator-based selection in multiobjective search. International Conference on Parallel Problem Solving From Nature 832–842.

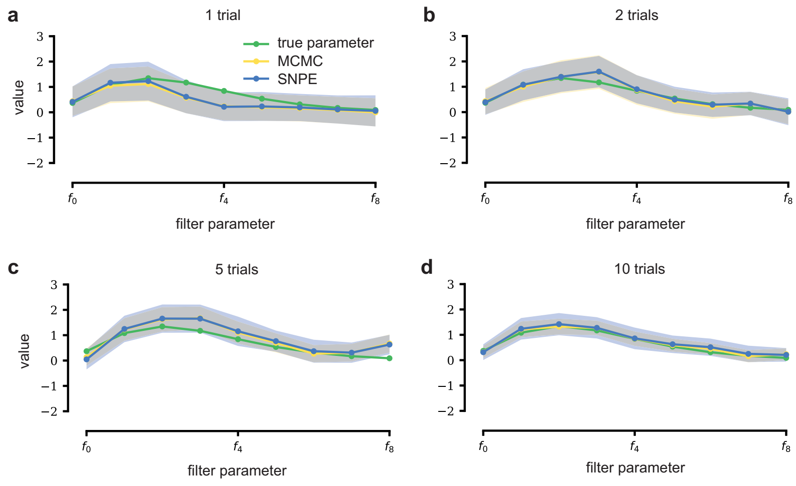
Appendix 1



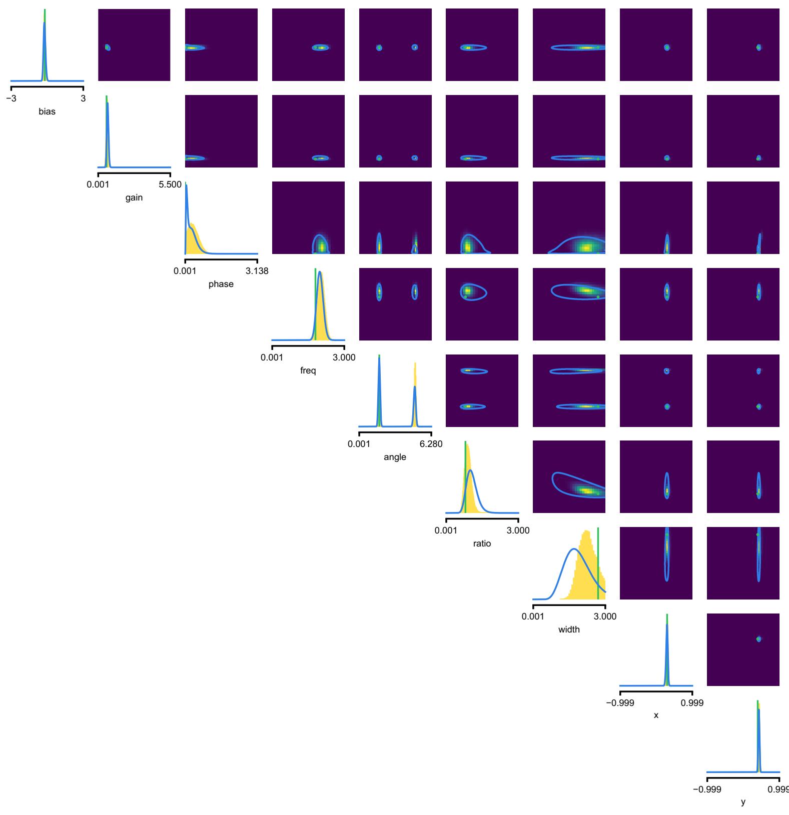
Appendix 1—figure 1. Comparison between SNPE-estimated posterior and reference posterior (obtained via MCMC) on LN model. (**a**) Posterior mean ± one standard deviation of temporal filter (receptive field) from SNPE posterior (SNPE, blue) and reference posterior (MCMC, yellow). (**b**) Full covariance matrices from SNPE posterior (left) and reference (MCMC, right).



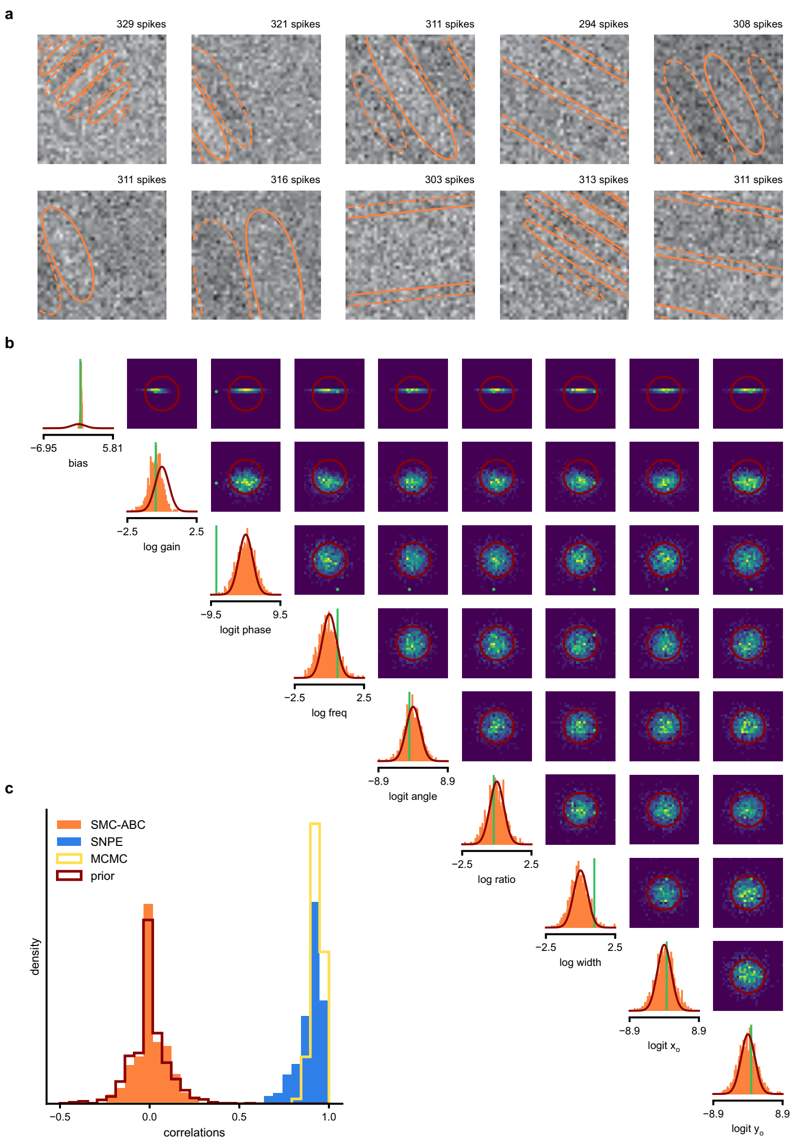
Appendix 1—figure 2. Full posterior for LN model. In green, ground-truth parameters. Marginals (blue lines) and 2D marginals for SNPE (contour lines correspond to 95% of the mass) and MCMC (yellow histograms).



Appendix 1—figure 3. LN model with additional data. With additional data, posterior samples cluster more tightly around the true filter. From left to right and top to bottom, SNPE (blue) and MCMC (yellow, for reference) are applied to observations with more independent Bernoulli trials, leading to progressively tighter posteriors and posterior samples closer to the true filter (which is the same across panels). Mean ± one standard deviation is shown. Note that SNPE closely agrees with the MCMC reference solution in all cases (**a-d**).

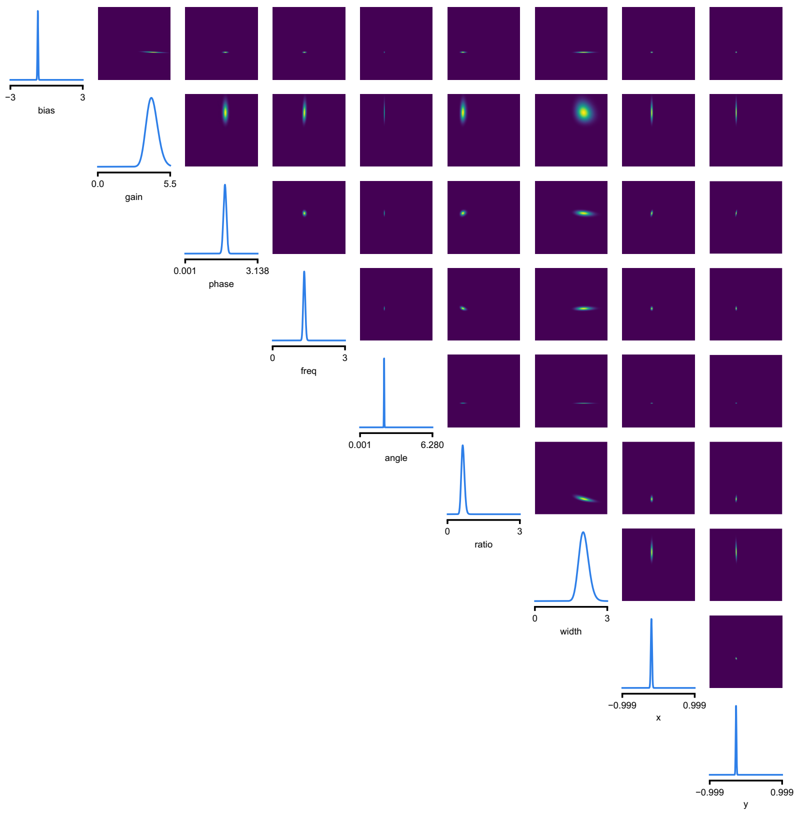


Appendix 1—figure 4. Full posterior for Gabor GLM receptive field model. SNPE posterior estimate (blue lines) compared to reference posterior (MCMC, histograms). Ground-truth parameters used to simulate the data in green. We depict the distributions over the original receptive field parameters, whereas we estimate the posterior as a Gaussian mixture over transformed parameters, see Materials and methods for details. We find that a (back-transformed) Gaussian mixture with four components approximates the posterior well in this case.

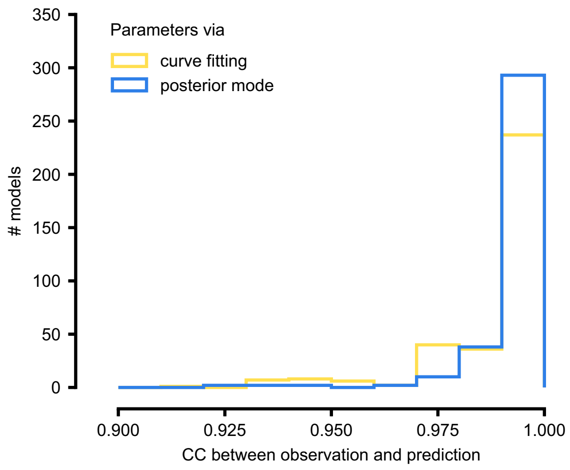


Appendix 1—figure 5. SMC-ABC posterior estimate for Gabor GLM receptive field model. (**a**) Spike-triggered averages (STAs) and spike counts with closest distance to the observed data out of 10000 simulations with parameters sampled from the prior. Spike counts are comparable to the observed data (299 spikes), but receptive fields (contours) are not well constrained. (**b**) Results for SMC-ABC with a total of 10^6 simulations. Histograms of 1000 particles (orange) returned in the final iteration of SMC-ABC, compared to prior (red contour lines) and ground-truth parameters (green). Distributions over (log-/logit-)transformed parameters, axis limits scaled to mean ± 3 standard deviations of the prior. (**c**) Correlations between ground-truth receptive field and receptive fields sampled from SMC-ABC posterior (orange), SNPE posterior (blue), reference MCMC posterior

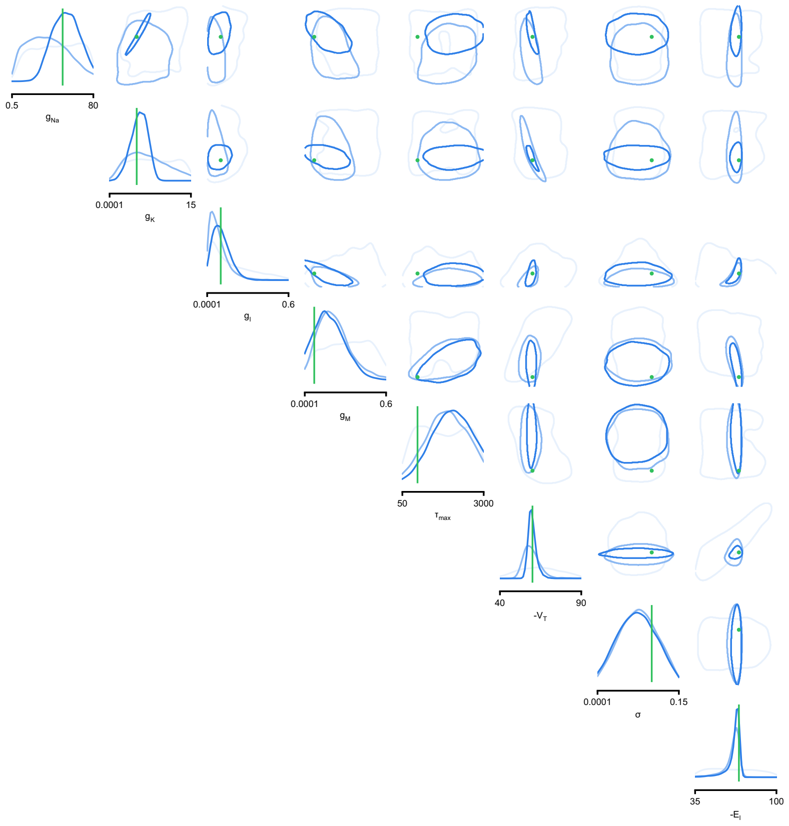
(yellow) and prior (red). The SNPE-estimated receptive fields are almost as good as those of the reference posterior, the SMC-ABC estimated ones no better than the prior.



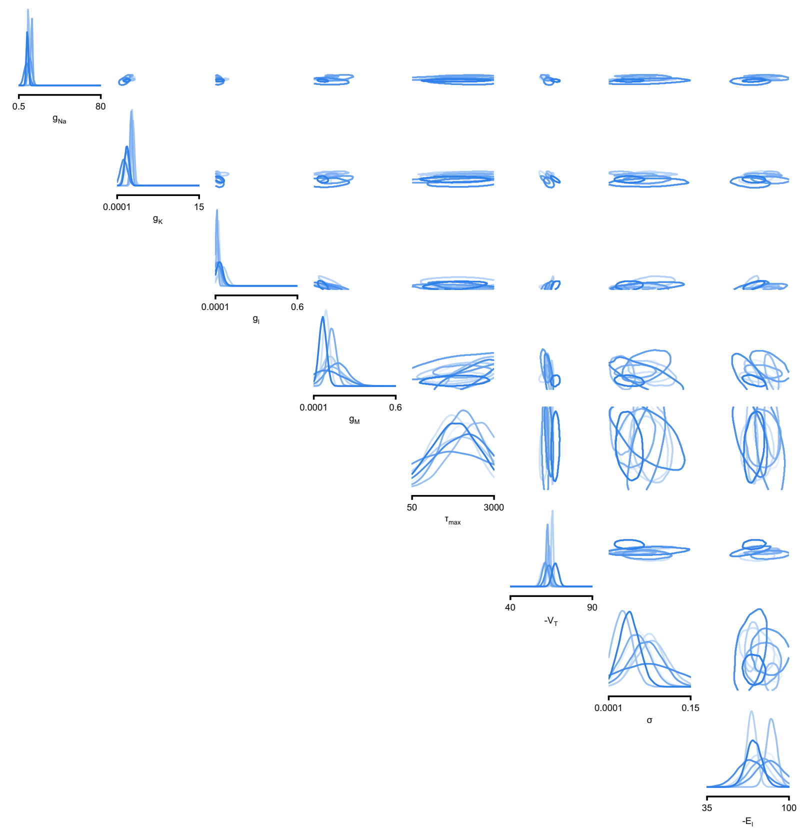
Appendix 1—figure 6. Full posterior for Gabor LN receptive field model on V1 recordings. We depict the distributions over the receptive field parameters, derived from the Gaussian mixture over transformed-parameters (see Materials and methods for details).



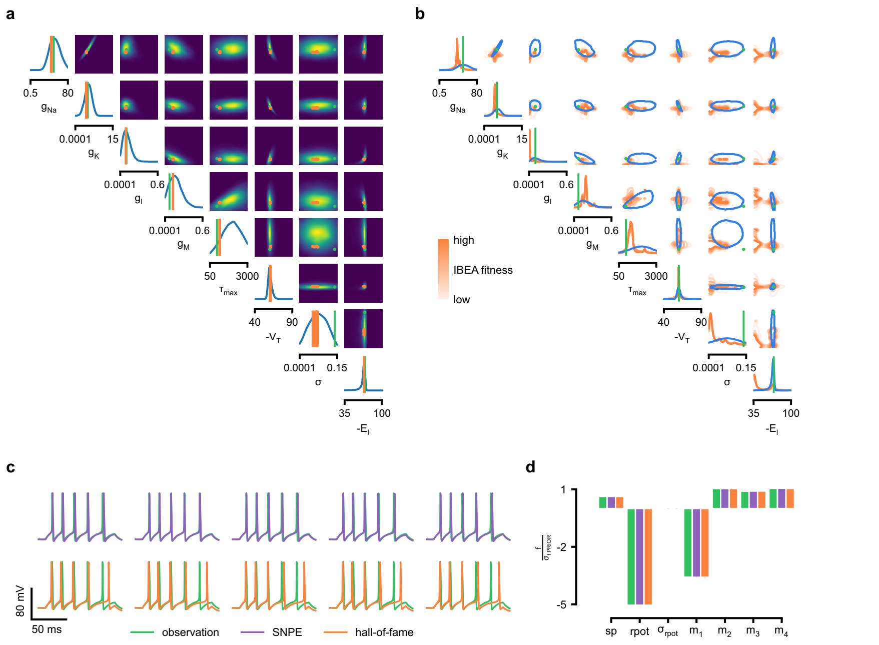
Appendix 1—figure 7. Summary results on 350 ICG channel models, and comparison with direct fits. We generate predictions either with the posterior mode (blue) or with parameters obtained by directly fitting steady-state activation and time-constant curves (yellow). We calculate the correlation coefficient (CC) between observation and prediction. The distribution of CCs is similar for both approaches.



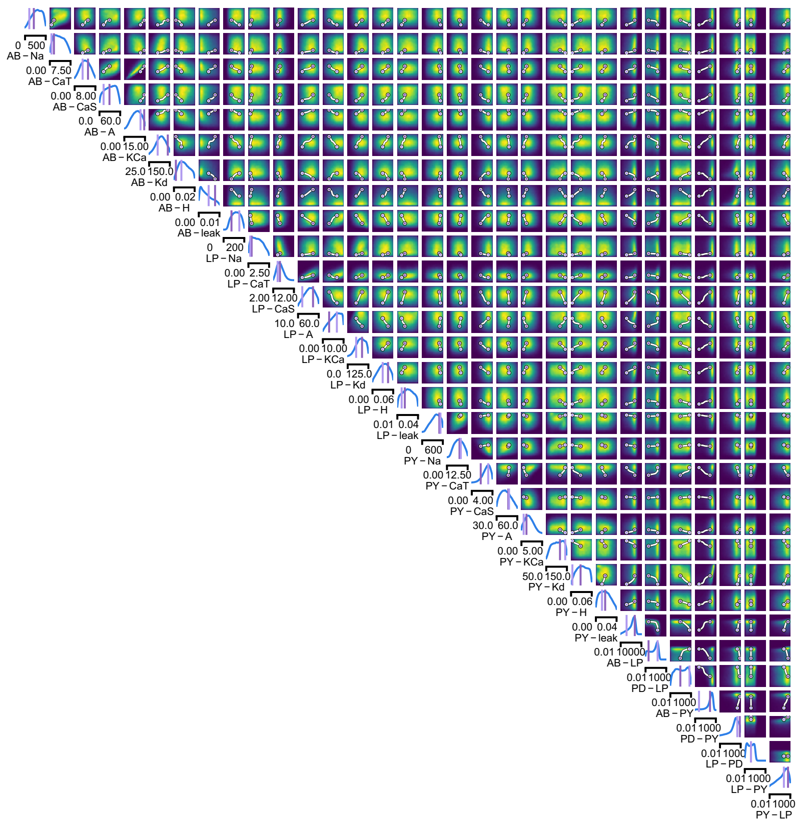
Appendix 1—figure 8. Full posteriors for Hodgkin-Huxley model for 1, 4, and 7 features. Images show the pairwise marginals for 7 features. Each contour line corresponds to 68% density mass for a different inferred posterior. Light blue corresponds to 1 feature and dark blue to 7 features. Ground truth parameters in green.



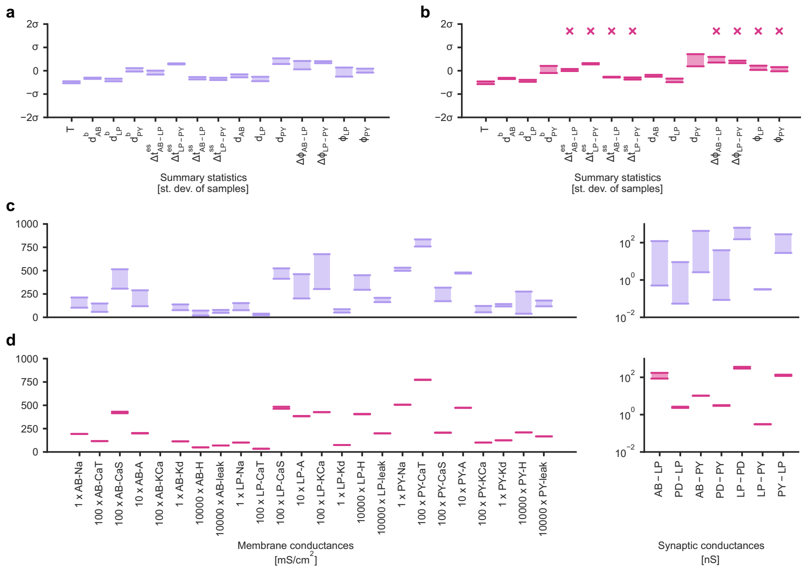
Appendix 1—figure 9. Full posteriors for Hodgkin-Huxley model on 8 different recordings from Allen Cell Type Database. Images show the pairwise marginals for 7 features. Each contour line corresponds to 68% density mass for a different inferred posterior.



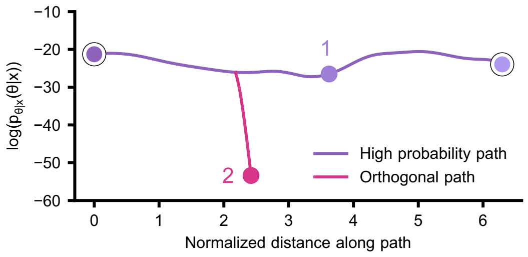
Appendix 1—figure 10. Comparison between SNPE posterior and IBEA samples for Hodgkin-Huxley model with 8 parameters and 7 features. (**a**) Full SNPE posterior distribution. Ground truth parameters in green and IBEA 10 parameters with highest fitness ('hall-of-fame') in orange. (**b**) Blue contour line corresponds to 68% density mass for SNPE posterior. Light orange corresponds to IBEA sampled parameters with lowest IBEA fitness and dark orange to IBEA sampled parameters with highest IBEA fitness. This plot shows that, in general, SNPE and IBEA can return very different answers– this is not surprising, as both algorithms have different objectives, but this highlights that genetic algorithms do not in general perform statistical inference. (**c**) Traces for samples with high probability under SNPE posterior (purple), and for samples with high fitness under IBEA objective (hall-of-fame; orange traces). (**d**) Features for the desired output (observation), the mode of the inferred posterior (purple) and the best sample under IBEA objective (orange). Each voltage feature is normalized by $\sigma_{f\mathrm{PRIOR}}$, the standard deviation of the respective feature of simulations sampled from the prior.



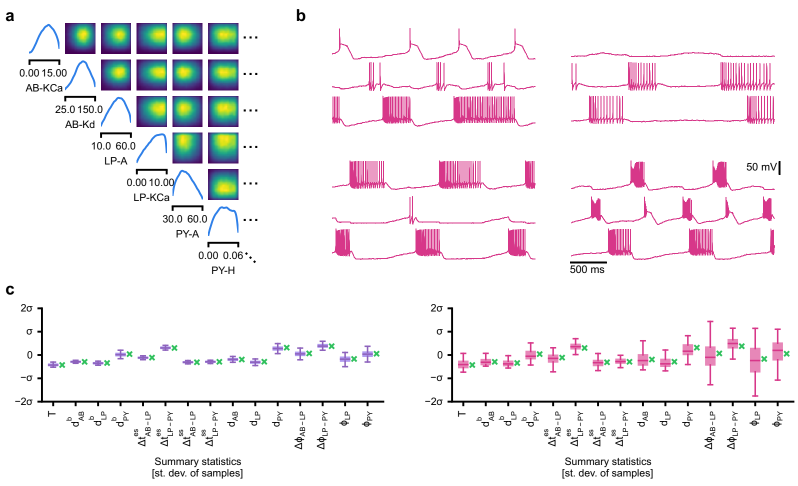
Appendix 1—figure 11. Full posterior for the stomatogastric ganglion over 24 membrane and 7 synaptic conductances. The first 24 dimensions depict membrane conductances (top left), the last 7 depict synaptic conductances (bottom right). All synaptic conductances are logarithmically spaced. Between two samples from the posterior with high posterior probability (purple dots), there is a path of high posterior probability (white).



Appendix 1—figure 12. Identifying directions of sloppiness and stiffness in the pyloric network of the crustacean stomatogastric ganglion. (**a**) Minimal and maximal values of all summary statistics along the path lying in regions of high posterior probability, sampled at 20 evenly spaced points. Summary statistics change only little. The summary statistics are z-scored with the mean and standard deviation of 170,000 bursting samples in the created dataset. (**b**) Summary statistics sampled at 20 evenly spaced points along the orthogonal path. The summary statistics show stronger changes than in panel a and, in particular, often could not be defined because neurons bursted irregularly, as indicated by an 'x' above barplots. (**c**) Minimal and maximal values of the circuit parameters along the path lying in regions of high posterior probability. Both membrane conductances (left) and synaptic conductances (right) vary over large ranges. Axes as in panel (**d**). (**d**) Circuit parameters along the orthogonal path. The difference between the minimal and maximal value is much smaller than in panel (**c**).



Appendix 1—figure 13. Posterior probability along high probability and orthogonal path. Along the path that was optimized to lie in regions of high posterior probability (purple), the posterior probability remains relatively constant. Along the orthogonal path (pink), optimized to quickly reduce posterior probability, the probability quickly drops. The start and end points as well as the points labeled 1 and 2 correspond to the points shown in ***Figure 5c***.



Appendix 1—figure 14. Evaluating circuit configurations in which parameters have been sampled independently. (**a**) Factorized posterior, that is, posterior obtained by sampling each parameter independently from the associated marginals. Many of the pairwise marginals look similar to the full posterior shown in ***Appendix 1—figure 11***, as the posterior correlations are low. (**b**) Samples from the factorized posterior– only a minority of these samples produce pyloric activity, highlighting the significance of the posterior correlations between parameters. (**c**) Left: summary features for 500 samples from the posterior. Boxplot for samples where all summary features are well-defined (80% of all samples). Right: summary features for 500 samples from the factorized posterior. Only 23% of these samples have well-defined summary features. The summary features from the factorized posterior have higher variation than the posterior ones. Summary features are z-scored using the mean and standard deviation of all samples in our training dataset obtained from prior samples. The boxplots indicate the maximum, 75% quantile, median, 25% quantile, and minimum. The green 'x' indicates the value of the experimental data (the observation, shown in ***Figure 5b***).

Likelihood-free inference with emulator networks

Jan-Matthis Lueckmann[1,2] JAN-MATTHIS.LUECKMANN@CAESAR.DE
Giacomo Bassetto[1,2] GIACOMO.BASSETTO@CAESAR.DE
Theofanis Karaletsos[3] THEOFANIS@UBER.COM
Jakob H. Macke[1,2,4] MACKE@TUM.DE

Abstract

Approximate Bayesian Computation (ABC) provides methods for Bayesian inference in simulation-based models which do not permit tractable likelihoods. We present a new ABC method which uses probabilistic neural *emulator* networks to learn synthetic likelihoods on simulated data – both 'local' emulators which approximate the likelihood for specific observed data, as well as 'global' ones which are applicable to a range of data. Simulations are chosen adaptively using an acquisition function which takes into account uncertainty about either the posterior distribution of interest, or the parameters of the emulator. Our approach does not rely on user-defined rejection thresholds or distance functions. We illustrate inference with emulator networks on synthetic examples and on a biophysical neuron model, and show that emulators allow accurate and efficient inference even on problems which are challenging for conventional ABC approaches.

1. Introduction

Many areas of science and engineering make extensive use of complex, stochastic, numerical simulations to describe the structure and dynamics of the processes being investigated (Karabatsos and Leisen, 2017). A key challenge in simulation-based science is linking simulation models to empirical data: Bayesian inference provides a general and powerful framework for identifying the set of parameters which are consistent both with empirical data and prior knowledge. One of the key quantities required for statistical inference, the likelihood of observed data given parameters, $\mathcal{L}(\boldsymbol{\theta}) = p(\mathbf{x}_o|\boldsymbol{\theta})$, is typically intractable for simulation-based models, rendering conventional statistical approaches inapplicable.

Approximate Bayesian Computation (ABC) aims to close this gap (Beaumont et al., 2002), but classical algorithms (Pritchard et al., 1999; Marjoram et al., 2003) scale poorly to high-dimensional non-Gaussian data, and require *ad-hoc* choices (i.e., rejection thresholds, distance functions and summary statistics) which can significantly affect both computational efficiency and accuracy. In *synthetic likelihood* approaches to ABC (Wood, 2010; Ong et al., 2016; Price et al., 2018), one instead uses density estimation to approximate the likelihood $p(s(\mathbf{x}_o)|\boldsymbol{\theta})$ on summary statistics $s(\cdot)$ of simulated data. A recent proposal by Järvenpää et al. (2017), Gutmann and Corander (2016) uses a Gaussian process ($\mathcal{GP}$) to approximate the

1. Computational Neuroengineering, Department of Electrical and Computer Engineering, Technical University of Munich, Germany
2. Neural Systems Analysis, Research Center caesar, an associate of the Max Planck Society, Bonn, Germany
3. Uber AI Labs, Uber Technologies, Inc., San Francisco, CA
4. Part of this work was done while J.H.M was at the Centre for Cognitive Science, Technische Universität Darmstadt, Germany

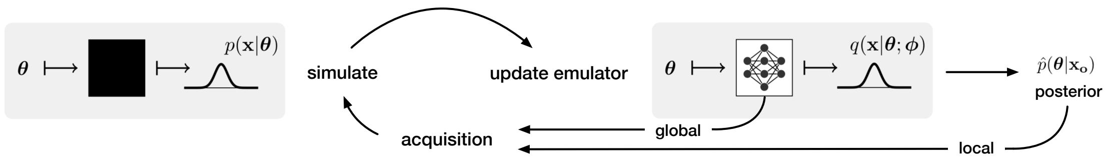


Figure 1: **Likelihood-free inference with emulator networks.** Our goal is to perform approximate Bayesian inference on simulator-models, i.e. models from which we can generate samples, but for which we can not evaluate likelihoods. We learn a tractable probabilistic emulator $q(\mathbf{x}|\boldsymbol{\theta};\phi)$ approximating the simulator $p(\mathbf{x}|\boldsymbol{\theta})$. The emulator then serves as a synthetic likelihood to obtain an approximate posterior. To train the emulator using a low number of simulations, we use active learning to select informative samples: The acquisition rule is either based on the current posterior estimate (if observed data $\mathbf{x}_o$ is given, 'local' learning), or on our uncertainty about the weights of the emulator network ('global' learning).

distribution of the discrepancy $d(s(\mathbf{x}), s(\mathbf{x}_o))$ as a function of $\boldsymbol{\theta}$, and Bayesian Optimization to propose new parameters. While this approach can be very effective even with a small number of simulations, it still requires summary statistics, choice of a distance function $d(\cdot,\cdot)$, and relies on assuming a homoscedastic $\mathcal{GP}$.

The goal of this paper is to scale synthetic-likelihood methods to multivariate and (potentially) non-Gaussian, heteroscedastic data. We use neural-network based conditional density estimators (which we call 'emulator networks', inspired by classical work on emulation methods; Kennedy and O'Hagan, 2002), to develop likelihood-free inference algorithms which are efficient, flexible, and scale to high-dimensional observations. Our approach does not require the user to specify rejection thresholds or distance functions, or to restrict oneself to a small number of summary statistics.

2. Likelihood-free inference with emulator networks

Our goal is to obtain an approximation to the true posterior $p(\boldsymbol{\theta}|\mathbf{x}_o)$ of a black-box simulator model, i.e. models from which we can generate samples $\mathbf{x} \sim p(\mathbf{x}|\boldsymbol{\theta})$, but for which we cannot evaluate likelihoods $\mathcal{L}(\boldsymbol{\theta})$. To solve this task, we learn a synthetic likelihood function $\hat{\mathcal{L}}(\boldsymbol{\theta})$ by training a conditional density estimator on simulated data. We actively propose parameters for simulations, since simulations are often the dominant cost in ABC: Therefore, we want to keep the number of calls to the simulator as low as possible (Fig. 1).

Core to our approach is an emulator $q(\mathbf{x}|\boldsymbol{\theta};\boldsymbol{\phi})$, a conditional density estimator with parameters $\boldsymbol{\phi}$ that approximates the simulator $p(\mathbf{x}|\boldsymbol{\theta})$. Having collected an initial simulated dataset $\mathcal{D}$, e.g. by repeatedly drawing from the prior $p(\boldsymbol{\theta})$ and simulating data, the emulator is trained. We actively select new locations $\boldsymbol{\theta}^*$ for which to simulate new data points $\mathcal{D}^* = \{(\boldsymbol{\theta}^*, \mathbf{x}^*)\}$ to keep the number of calls to the (potentially computationally expensive) simulator low. $\mathcal{D}^*$ is appended to the dataset, the emulator is updated, and the active learning loop repeats. The emulator defines a synthetic likelihood function $\hat{\mathcal{L}}(\boldsymbol{\theta}) = q(\mathbf{x} = \mathbf{x}_o|\boldsymbol{\theta};\boldsymbol{\phi})$

that we use to find an approximate posterior, which is proportional to $\tilde{p}(\boldsymbol{\theta}|\mathbf{x}_o) := \hat{\mathcal{L}}(\boldsymbol{\theta})p(\boldsymbol{\theta})$. This approach is summarized in Appendix A in form of an algorithm.

Thus, our approach requires (1) an emulator, i.e., a flexible conditional density estimator, (2) an approach for learning the emulator on simulated data and expressing our uncertainty about its parameters, (3) an acquisition rule for proposing new sampling locations, and (4) an inference procedure for obtaining the posterior distribution from the synthetic likelihood and the prior. We will describe these steps in the following.

2.1. Choice of emulator

We use neural network based emulators $q(\mathbf{x}|\boldsymbol{\theta};\boldsymbol{\phi})$: parameters $\boldsymbol{\theta}$ are given as inputs to the network, and the network is trained to approximate $p(\mathbf{x}|\boldsymbol{\theta})$. In contrast to traditional synthetic likelihood approaches (Wood, 2010), we are not restricted to using a (multivariate) normal distribution to approximate the conditional density $p(\mathbf{x}|\boldsymbol{\theta})$. The output form of the emulator is chosen according to our knowledge regarding the conditional density of the simulator. In our second example application, we e.g. model $\mathbf{x}|\boldsymbol{\theta}$ as a binomial distribution over 8-bit integer pixel values, and in the third example we model a categorical distribution. If the noise model of the simulation process is unknown, flexible conditional density estimators such as conditional autoregressive models (Oord et al., 2016; Papamakarios et al., 2018) can be readily used in our approach.

2.2. Inference on the parameters of the emulator

We use probabilistic neural networks, i.e. we represent uncertainty about the parameters $\boldsymbol{\phi}$ of the emulator $q(\mathbf{x}|\boldsymbol{\theta};\boldsymbol{\phi})$. We then use these uncertainties to guide the acquisition of training data for the emulator using active learning (as discussed in the next section).

In the Bayesian framework, uncertainty is represented through the posterior distribution. Multiple approaches for estimating the posterior distributions over neural network parameters have been proposed, including MCMC methods to draw samples from the full posterior (Welling and Teh, 2011; Chen et al., 2014) and variational methods, e.g. using factorising posteriors (Blundell et al., 2015) or normalizing flows (Louizos and Welling, 2017). Finally, deep ensemble approaches (Lakshminarayanan et al., 2017) represent predictive distributions through ensembles of networks. They have the advantage of not requiring the choice of a functional form of the approximation, and are simple to set up.

Our approach can be applied with any method that represents uncertainty over network parameters. In our experiments, we use deep ensembles to represent uncertainty about $\boldsymbol{\phi}$, as we found them to combine simplicity with good empirical performance. Instead of training a single emulator network and inferring its posterior distribution, we train an ensemble of M networks with parameters $\{\boldsymbol{\phi}_m\}_{m=1}^{M}$. From here on, we treat $\boldsymbol{\phi}_m$ as if they were samples from $p(\boldsymbol{\phi}|\mathcal{D})$, the posterior over network parameters given data. (In practice, these samples will describe local maxima of the posterior.) The posterior-predictive distribution is approximated by $\mathbb{E}_{\boldsymbol{\phi}|\mathcal{D}}\big[q(\mathbf{x}|\boldsymbol{\theta},\boldsymbol{\phi})\big] \approx \frac{1}{M}\sum_{m=1}^{M} q(\mathbf{x}|\boldsymbol{\theta};\boldsymbol{\phi}_m)$.

Networks are trained supervised with data $\mathcal{D} = \big\{(\boldsymbol{\theta}_n, \mathbf{x}_n)\big\}_{n=1}^{N}$. During training, the parameters of the networks are optimized subject to the loss $-\sum_{m=1}^{M}\sum_{n=1}^{N}\log q(\mathbf{x}_n|\boldsymbol{\theta}_n;\boldsymbol{\phi}_m)$ w.r.t. $\boldsymbol{\phi}$ (a proper scoring rule as discussed in Lakshminarayanan et al., 2017). Networks in the ensemble are initialized differently, and data points are randomly shuffled during training.

2.3. Acquisition rules

We use active learning to selectively acquire new samples. We distinguish between two scenarios: In the first, we have particular observed data $\mathbf{x}_o$ available, and train a *local emulator* which approximates the likelihood near $\mathbf{x}_o$. This approach requires learning a new emulator for each new observed data $\mathbf{x}_o$.

We also consider a second scenario, in which we learn a *global emulator* – which approximates $p(\mathbf{x}|\boldsymbol{\theta})$ globally. Learning a global emulator is more challenging and may potentially require more flexible density estimators. However, once the emulator is learned, we can readily approximate the likelihood for *any* $\mathbf{x}_o$, therefore amortizing the cost of learning the emulator.

The two scenarios call for different acquisition functions for proposing new samples, which we will discuss next.

2.3.1. Acquisitions for local emulator learning

With given $\mathbf{x}_o$, we want to learn a local emulator that allows us to derive a good approximation to the (unnormalized) posterior $\tilde{p}(\boldsymbol{\theta}|\mathbf{x}_o) \propto \mathbb{E}_{\phi|\mathcal{D}}\big[q(\mathbf{x}=\mathbf{x}_o|\boldsymbol{\theta};\boldsymbol{\phi})\big]p(\boldsymbol{\theta})$.

As we are interested in increasing our certainty about the posterior, we target its variance, $\mathbb{V}_{\boldsymbol{\phi}|\mathcal{D}}[\tilde{p}(\boldsymbol{\theta}|\mathbf{x}_o,\boldsymbol{\phi})]$, where $\mathbb{V}_{\boldsymbol{\phi}|\mathcal{D}}$ denotes that we take the variance with respect to the posterior over network weights given data $\mathcal{D}$. Thus, we use an acquisition rule which targets the region of maximum variance in the predicted (unnormalized) posterior,

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \mathbb{V}_{\boldsymbol{\phi}|\mathcal{D}}[\tilde{p}(\boldsymbol{\theta}|\mathbf{x}_o,\boldsymbol{\phi})] = \arg\max_{\boldsymbol{\theta}} \log p(\boldsymbol{\theta}) + \log\sqrt{\mathbb{V}_{\boldsymbol{\phi}|\mathcal{D}}[\hat{\mathcal{L}}(\boldsymbol{\theta})]}. \tag{1}$$

We approximate $\mathbb{V}_{\boldsymbol{\phi}|\mathcal{D}}$ with the sample variance across $\boldsymbol{\phi}_m$ drawn from the posterior over networks. We refer to this rule as the *MaxVar* rule (Järvenpää et al., 2017). We optimize this acquisition rule by using gradient descent, making use of automatic differentiation to take gradients with respect to $\boldsymbol{\theta}$ through the synthetic likelihood specified by the emulator.

2.3.2. Acquisitions for global emulator learning

A global emulator may be used to do inference once $\mathbf{x}_o$ becomes available. Here, the goal for active learning is to bring the emulator $q(\mathbf{x}|\boldsymbol{\theta};\boldsymbol{\phi})$ close to the simulator $p(\mathbf{x}|\boldsymbol{\theta})$ for all $\boldsymbol{\theta}$s using as few runs of the simulator as possible. We use a rule based on information theory from the active learning literature (Houlsby et al., 2011; Gal et al., 2017; Depeweg et al., 2017). We refer to the rule

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \mathbb{I}[\mathbf{x},\boldsymbol{\phi}|\boldsymbol{\theta},\mathcal{D}] = \arg\max_{\boldsymbol{\theta}} \mathbb{H}[\mathbf{x}|\boldsymbol{\theta},\mathcal{D}] - \mathbb{E}_{\boldsymbol{\phi}|\mathcal{D}}\big[\mathbb{H}[\mathbf{x}|\boldsymbol{\theta},\boldsymbol{\phi}]\big] \tag{2}$$

as the maximum mutual information rule (*MaxMI*). See Appendix B for details.

2.4. Deriving the posterior distribution from the emulator

Once we have learned the emulator, we use Hamiltonian Monte Carlo (HMC, Neal, 2010) to draw samples from our approximate posterior, using the emulator-based synthetic likelihood. We generate samples of $\boldsymbol{\theta}$ drawn from the distribution $\tilde{p}(\boldsymbol{\theta}|\mathbf{x}_o) = \mathbb{E}_{\boldsymbol{\phi}|\mathcal{D}}\big[q(\mathbf{x}_o|\boldsymbol{\theta})\big]p(\boldsymbol{\theta})$. In practice, we sample $\boldsymbol{\theta}$ from each ensemble member individually and use the union of all

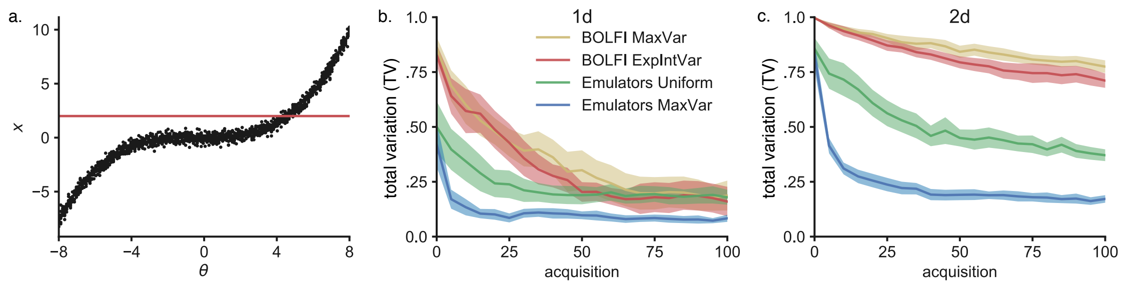


Figure 2: **Inference on simulator with Gaussian noise**. **a.** Data is generated from $\mathbf{x} \sim \mathcal{N}(\mathbf{x}|f(\boldsymbol{\theta}), \boldsymbol{\Sigma})$ with cubic non-linearity. We illustrate posterior inference $p(\boldsymbol{\theta}|\mathbf{x}_o)$ given $\mathbf{x}_o = 2$ (red line at $\mathbf{x}_o$). **b.** In 1-D, emulator-based inference with *MaxVar* acquisitions leads to faster convergence to true posterior than uniform sampling, or BOLFI. Total variation (TV) is measured between true and approximate posteriors. 100 acquisitions starting from $N_{\text{initial}} = 10$ initial points. Lines are means and SEMs from 20 runs. **c.** Same problem, but $\mathbf{x}$ and $\boldsymbol{\theta} \in \mathbb{R}^2$, non-linearity applied point-wise, starting from $N_{\text{initial}} = 25$ points.

samples as a draw from the approximate posterior. We could also obtain the posterior through variational inference, but here prefer to retain full flexibility in the shape of the inferred posterior.

3. Results

We demonstrate likelihood-free inference with emulator networks on three examples: i) we show that emulators are competitive with state-of-the-art on an example with Gaussian observations; ii) we demonstrate the ability of emulators to work with high-dimensional observations while learning to amortize the simulator; iii) we show an application from neuroscience, and infer the posterior over parameters of a biophysical neuron model.

i) Low-dimensional example: Simulator with Gaussian observations

We first demonstrate emulator networks on a non-linear model between parameters and data, corrupted by additive Gaussian observation noise: data is generated according to $\mathbf{x}_i \sim \mathcal{N}(\cdot|f(\boldsymbol{\theta}), \boldsymbol{\Sigma})$, $i = 1 \dots n$, where $f(\boldsymbol{\theta})$ is cubic in $\boldsymbol{\theta}$, $\boldsymbol{\Sigma}$ is fixed, and $\boldsymbol{\theta}$ is distributed uniformly (see Appendix D for complete specification). The goal is to approximate the posterior $p(\boldsymbol{\theta}|\bar{\mathbf{x}}_o)$ from a small number of draws from the generative model (Fig. 2a). We parameterize $q(\mathbf{x}|\boldsymbol{\theta}; \boldsymbol{\phi})$ using a Gaussian distribution whose mean and precision are the output of a neural network with one hidden layer consisting of 10 tanh units.

We will compare our method to BOLFI (Bayesian Optimization for Likelihood-free Inference, Gutmann and Corander, 2016), an ABC method which – given a user-specified discrepancy measure – learns a $\mathcal{GP}$ that models the distribution of discrepancies between summary statistics of $\mathbf{x}$ and $\mathbf{x}_o$. Järvenpää et al. (2017) proposed multiple acquisition rules for BOLFI. The most principled (but also most costly) rule minimizes the expected integrated variance (*ExpIntVar*) of the approximate posterior after acquiring new data. BOLFI is a state-of-the-art method for simulation-efficient likelihood-free inference, and substantially

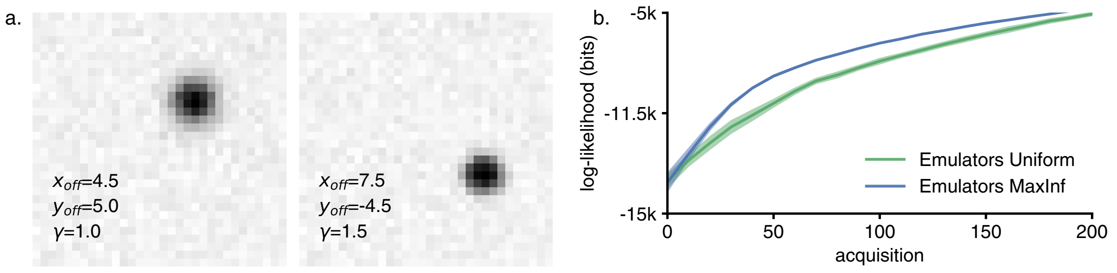


Figure 3: **Inferring location and contrast of a blob**. **a.** Two sample images from the generative model. Parameters are the spatial position and the contrast of the blob. **b.** Acquiring samples using the *MaxMI* rule yield to faster emulator learning than samples acquired uniformly in the parameter space. Performances are reported as log-likelihood of held-out test data. 200 acquisitions starting from $N_{\text{initial}} = 50$ initial points. Lines are means and SEMs from 20 runs.

more efficient than classical rejection-based methods such as rejection-ABC (Pritchard et al., 1999), MCMC-ABC (Marjoram et al., 2003), SMC-ABC (Sisson et al., 2007).

We use the total variation (TV) between true and approximate posterior (evaluated using numerical integration) to quantify performance as a function of the number of acquisitions. The emulator is trained on an initial dataset and updated after each new acquisition. We find that emulators with *MaxVar* sampling work better than uniform sampling (Fig. 2b). Both BOLFI rules (*ExpIntVar* and *MaxVar*) exhibit very similar performance, but require higher number of simulations than emulators to reach low TV values. On a 2-dimensional version of the problem, the qualitative ordering is the same, but the differences between methods are greater (Fig. 2c). We did additional runs of BOLFI *MaxVar* to confirm that it eventually converges towards the correct posterior. However, convergence is slow and the quality of the inferred posterior depends strongly on the choice of the threshold parameter used in BOLFI (see Appendix G).

ii) High-dimensional observations: Inferring the location and contrast of a blob

We show that our method can be applied to estimation problems with high-dimensional observations without having to resort to using summary statistics. We model the rendering of a blob on a 2D image, and learn a global emulator for the forward model.

The forward model takes as inputs three parameters (x_{off}, y_{off} and γ) – which encode horizontal and vertical displacement, and contrast of the blob – and returns per-pixels activation probabilities p_{ij}. The value of each pixel v_{ij} is then generated according to a binomial distribution with total count 255 (8-bits gray-scale image) and probability p_{ij}, resulting in a 32×32 pixel image (Fig. 3a). In this application, we use a multi-layer neural network whose output is, for each pixel, the mean parameter of the binomial distribution (see Appendix E for further details).

Using the *MaxMI* rule to acquire new test points in parameters space results in faster learning of the emulator, compared to uniform random acquisitions. Eventually, both rules converge towards the log-likelihood of the held-out test set, indicating successful global

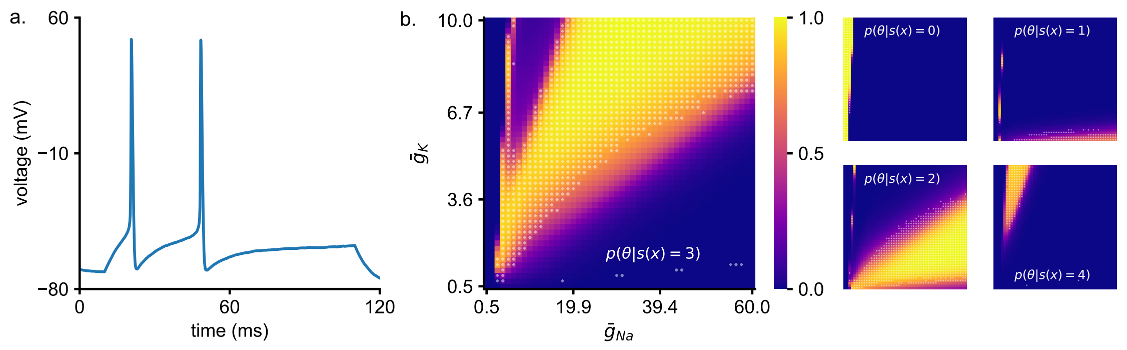


Figure 4: **Hodgkin-Huxley model. a.** Example trace from differential equations describing the model. **b.** Posterior inferred for number of spikes as a function of two biophysical parameters. Panels show posteriors for a given number of spikes. The largest panel shows the posterior given three spikes. As a posterior predictive check, we overlay white transparent markers on top of the posteriors where a simulation produced the given number of spikes (and no marker otherwise).

emulation of the forward model (Fig. 3b). We show posteriors distributions and samples in Appendix H. Since alternative approaches for likelihood-free inference (e.g. BOLFI) do not allow one to globally approximate a simulator, no performance benchmark against these methods was performed.

iii) Scientific application: Hodgkin-Huxley model

As an example of a scientific application, we use the Hodgkin-Huxley model (Hodgkin and Huxley, 1952) which describes the evolution of membrane potential in neurons (Fig. 4a). Fitting single- and multi-compartment Hodgkin-Huxley models to neurophysiological data is a central problem in neuroscience, and typically addressed using non-Bayesian approaches based on evolutionary optimization (Druckmann et al., 2007; Van Geit et al., 2016). In contrast to the previous examples, we do not model the raw data $\mathbf{x}$, but summary features derived from them. While this is often done out of necessity, calculating the posterior relative to summary statistics can be of scientific interest (Cornebise and Girolami, 2012). This is indeed the case when fitting biophysical models in neuroscience, which is typically performed with carefully chosen summary statistics representing properties of interest.

Here, we chose to model the number of action potentials (or spikes) in response to a step-current input, and we are interested in the set of parameters that are consistent with the observed number of action potentials. The conditional density of the emulator networks becomes a categorical distribution with 6 classes, modelling the probabilities of exactly 0, 1, . . . 4 spikes, and 5 or more spikes (which never occurred under the parameter ranges we explored). Model parameters $\boldsymbol{\theta}$ are the ion-channel conductances $\bar{g}_{\mathrm{Na}}$ and $\bar{g}_{\mathrm{K}}$, controlling the shape and frequency of the spikes (further details in Appendix F).

We trained emulator networks using *MaxMI* to infer the posterior probabilities over $\boldsymbol{\theta}$ generating a given number of observed spikes – the acquisition surface is shown in Appendix I. Resulting posterior distributions are shown in Fig. 4b, along with a posterior predictive check showing that the mapping between parameters and summary features was learned correctly.

4. Discussion

We presented an approach for performing statistical inference on simulation-based models which do not permit tractable likelihood. We learn an 'emulator network', i.e. a probabilistic model that is consistent with the simulation, and for which likelihoods are tractable. The likelihoods of the emulator can then be plugged into any Bayesian inference approach (as in synthetic likelihood approaches Wood, 2010; Ong et al., 2016, 2018) to calculate the posterior. Active learning can be used to adaptively suggest new samples to reduce the number of calls to the simulator. We discussed two acquisition functions for learning 'local' and 'global' emulators, we showed that our approach scales to high-dimensional observation spaces, does not require user-defined distance functions or acceptance thresholds, and is not limited to Gaussian observations – all of which are challenging for conventional ABC approaches.

Our approach uses density estimation to approximate the likelihood. A complementary use of density-estimation in ABC is to directly target the posterior distribution (Papamakarios and Murray, 2017; Lueckmann et al., 2018; Le et al., 2017; Izbicki et al., 2018). This approach can be very useful – however, one advantage of likelihood-based approaches is that they allow one to apply the same synthetic likelihood to multiple priors (without having to retrain), or to pool information from multiple observations (by multiplying the corresponding synthetic likelihoods). More technically, posterior density estimation gives less flexibility in proposing samples – in order to yield the correct posterior, samples have to be drawn from the prior, or approaches such as importance-weighting (Lueckmann et al., 2018) or other post-hoc corrections (Papamakarios and Murray, 2017) have to be applied. We discuss additional related work published concurrently with this manuscript in Appendix C.

There are multiple ways in which our approach can be improved further: First, one could use alternative, and more expressive neural-network based density estimators, e.g. ones based on normalizing flows (Papamakarios et al., 2018). Second, one could use Bayesian posterior estimation (rather than ensembles) to capture parameter uncertainty, and/or use variational inference (rather than HMC) to derive an estimate of the posterior from the synthetic likelihood provided by the emulator. Third, we presented two acquisition functions (one for local and one for global estimation) – it is likely that the approach can be made more simulation-efficient by using different, and more sophisticated acquisition functions. In particular, our *MaxVar* rule targets the parameters with maximal uncertainty, but does not try to predict whether that uncertainty will be effectively reduced. However, evaluating acquisition functions like *ExpIntVar* can be computationally expensive – it will be useful to develop approaches which are sensitive to the relative cost of simulations and proposals, and adaptively adjust the acquisition function used.

Numerical simulations make it possible to model complex phenomena from first principles, and are indispensable tools in many fields in engineering and science. The advent of powerful approaches for statistical inference in simulation-based models (Brehmer et al., 2018) is opening up exciting opportunities for closing the gap between mechanistic, simulation-based and statistical approaches to modelling complex systems. Our Bayesian methodology based on emulators provides a fast, effective surrogate model for the intractable likelihood implied by the simulator, and the active-learning based rules lead to bounded-rational decisions about which simulations to run. In combination, they form a rigorous and resource-efficient basis for data analysis with simulators in the loop.

Acknowledgements

We thank Marcel Nonnenmacher and Pedro J. Gonçalves for discussions, and help with simulation of the Hodgkin-Huxley model. We thank David Greenberg and all members of the Neural Systems Analysis group for comments on the manuscript.

This work was supported by BMBF (FKZ 01IS18052 A-D, Project ADIMEM) and DFG (SFB 1089, SPP 2041, and SFB 1233, Project 'Robust Vision', 276693517) grants and by the caesar foundation.

References

M Beaumont, W Zhang, and D J Balding. Approximate bayesian computation in population genetics. *Genetics*, 162(4), 2002.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning*, 2015.

Johann Brehmer, Kyle Cranmer, Gilles Louppe, and Juan Pavez. Constraining Effective Field Theories with Machine Learning. *Physical Review Letters*, 121(11), 2018.

T Chen, E B Fox, and C Guestrin. Stochastic gradient hamiltonian monte carlo. In *Proceedings of the 31st International Conference on International Conference on Machine Learning*, 2014.

J Cornebise and M Girolami. Inference on summary statistics: a mean or an end? *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 74(3):419–474, 2012.

S Depeweg, J M Hernández-Lobato, F Doshi-Velez, and S Udluft. Decomposition of uncertainty for active learning and reliable reinforcement learning in stochastic systems. *arXiv:1710.07283*, 2017.

S Druckmann, Y Banitt, A Gidon, F Schürmann, H Markram, and I Segev. A novel multiple objective optimization framework for constraining conductance-based neuron models by experimental data. *Frontiers in Neuroscience*, 1, 2007.

Y Gal, R Islam, and Z Ghahramani. Deep bayesian active learning with image data. *arXiv:1703.02910*, 2017.

M U Gutmann and J Corander. Bayesian Optimization for Likelihood-Free Inference of Simulator-Based Statistical Models. *Journal of Machine Learning Research*, 17(125), 2016.

A L Hodgkin and A F Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of Physiology*, 117(4), 1952.

N Houlsby, F Huszar, Z Ghahramani, and M Lengyel. Bayesian active learning for classification and preference learning. *arXiv:1112.5745*, 2011.

R Izbicki, A B Lee, and T Pospisil. Abc-cde: Towards approximate bayesian computation with complex high-dimensional data and limited simulations. *arXiv:1805.05480*, 2018.

M Järvenpää, M U Gutmann, A Pleska, Vehtari, A, and P Marttinen. Efficient acquisition rules for model-based approximate Bayesian computation. *arXiv:1704.00520v2*, 2017.

G Karabatsos and F Leisen. An Approximate Likelihood Perspective on ABC Methods. *arXiv:1708.05341*, 2017.

M C Kennedy and A O'Hagan. Bayesian calibration of computer models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(3), 2002.

B Lakshminarayanan, A Pritzel, and C Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems 30*, 2017.

T A Le, A G Baydin, R Zinkov, and F D Wood. Using synthetic data to train neural networks is model-based reasoning. *arXiv:1703.00868*, 2017.

C Louizos and M Welling. Multiplicative normalizing flows for variational bayesian neural networks. In *Proceedings of the 34th International Conference on International Conference on Machine Learning*, 2017.

JM Lueckmann, P J Goncalves, G Bassetto, K Öcal, M Nonnenmacher, and J H Macke. Flexible statistical inference for mechanistic models of neural dynamics. In *Advances in Neural Information Processing Systems 30*, 2018.

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

R M Neal. MCMC using Hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*, 54:113–162, 2010.

V M H Ong, D J Nott, M Tran, S A Sisson, and C C Drovandi. Variational bayes with synthetic likelihood. *arXiv:1608.03069*, 2016.

V MH Ong, D J Nott, and M S Smith. Gaussian variational approximation with a factor covariance structure. *Journal of Computational and Graphical Statistics*, 27(3), 2018.

A van den Oord, N Kalchbrenner, and K Kavukcuoglu. Pixel recurrent neural networks. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning*, 2016.

G Papamakarios and I Murray. Fast epsilon-free inference of simulation models with bayesian conditional density estimation. In *Advances in Neural Information Processing Systems*, 2017.

G Papamakarios, T Pavlakou, and I Murray. Masked autoregressive flow for density estimation. In *Advances in Neural Information Processing Systems*, 2018.

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

J K Pritchard, M T Seielstad, A Perez-Lezaun, and M W Feldman. Population growth of human y chromosomes: a study of y chromosome microsatellites. *Mol Biol Evol*, 16(12), 1999.

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

W Van Geit, M Gevaert, G Chindemi, C Rössert, JD Courcol, E B Muller, F Schürmann, I Segev, and H Markram. Bluepyopt: Leveraging open source software and cloud infrastructure to optimise model parameters in neuroscience. *Frontiers in Neuroinformatics*, 10, 2016.

M Welling and Y W Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, 2011.

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

Appendix A.

Algorithm 1: ABC via active learning to learn a synthetic likelihood

```
Input  : $p(\boldsymbol{\theta})$, $p(\mathbf{x}|\boldsymbol{\theta})$, $\mathbf{x}_o$          // prior, stochastic simulator, observed data
Output : $\hat{p}(\boldsymbol{\theta}|\mathbf{x}_o)$                                   // approximate posterior
1 $\mathcal{D} \leftarrow \mathcal{D}^{N_{\text{initial}}} = \{(\boldsymbol{\theta}_n, \mathbf{x}_n)\}_{n=1}^{N_{\text{initial}}} \sim p(\mathbf{x}, \boldsymbol{\theta})$      // $\boldsymbol{\theta}_n \sim p(\boldsymbol{\theta})$, $\mathbf{x}_n \sim p(\mathbf{x}|\boldsymbol{\theta}_n)$
2 do
3     Train emulator $q(\mathbf{x}|\boldsymbol{\theta}; \boldsymbol{\phi})$ on $\mathcal{D}$
4     Find $\boldsymbol{\theta}^*$ as the maximum of an acquisition function
5     Acquire new data point $\mathcal{D}^* = \{(\boldsymbol{\theta}^*, \mathbf{x}^*)\}$ by simulating for $\boldsymbol{\theta}^*$     // $\mathbf{x}^*|\boldsymbol{\theta}^* \sim p(\mathbf{x}|\boldsymbol{\theta}^*)$
6     $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}^*$
7 while not converged
8 Find $\hat{p}(\boldsymbol{\theta}|\mathbf{x}_o)$ using the synthetic likelihood $\hat{\mathcal{L}}(\boldsymbol{\theta}) = q(\mathbf{x}_o|\boldsymbol{\theta}; \boldsymbol{\phi})$
```

Appendix B. Acquisition rule for global emulator learning

For global emulator learning, we use a rule based on information theory from the active learning literature that maximizes information gain (Houlsby et al., 2011; Gal et al., 2017; Depeweg et al., 2017). We refer to the rule

$$\begin{aligned}\boldsymbol{\theta}^* &= \arg\max_{\boldsymbol{\theta}} \mathbb{I}[\mathbf{x}, \boldsymbol{\phi}|\boldsymbol{\theta}, \mathcal{D}] \\ &= \arg\max_{\boldsymbol{\theta}} \underbrace{\mathbb{H}[\mathbf{x}|\boldsymbol{\theta}, \mathcal{D}]}_{\text{entropy}} - \underbrace{\mathbb{E}_{\boldsymbol{\phi}|\mathcal{D}}\big[\mathbb{H}[\mathbf{x}|\boldsymbol{\theta}, \boldsymbol{\phi}]\big]}_{\text{expected conditional entropy}}\end{aligned} \tag{3}$$

as the maximum mutual information rule (*MaxMI*).

The first term is the entropy of the data under the posterior-predictive distribution implied by the emulator:

$$\mathbb{H}[\mathbf{x}|\boldsymbol{\theta}, \mathcal{D}] = -\int \hat{p}(\mathbf{x}|\boldsymbol{\theta}, \mathcal{D}) \ln \hat{p}(\mathbf{x}|\boldsymbol{\theta}, \mathcal{D}) \mathrm{d}\mathbf{x}, \tag{4}$$

where $\hat{p}(\mathbf{x}|\boldsymbol{\theta}, \mathcal{D})$ is obtained by marginalizing out the emulator's parameters w.r.t. $p(\boldsymbol{\phi}|\mathcal{D})$:

$$\hat{p}(\mathbf{x}|\boldsymbol{\theta}, \mathcal{D}) = \int q(\mathbf{x}|\boldsymbol{\theta}, \boldsymbol{\phi}) p(\boldsymbol{\phi}|\mathcal{D}) \mathrm{d}\boldsymbol{\phi}. \tag{5}$$

The expected conditional entropy, $\mathbb{E}_{\boldsymbol{\phi}|\mathcal{D}}\big[\mathbb{H}[\mathbf{x}|\boldsymbol{\theta}, \boldsymbol{\phi}]\big]$, is the average entropy of the output $\mathbf{x}$ for a particular choice of inputs $\boldsymbol{\theta}$ and emulator parameters $\boldsymbol{\phi}$, under the posterior distribution of emulator parameters $p(\boldsymbol{\phi}|\mathcal{D})$. Again, we treat ensemble members $\boldsymbol{\phi}_m$ as if they were draws from $p(\boldsymbol{\phi}|\mathcal{D})$. Houlsby et al. refer to this rule as Bayesian Active Learning by Disagreement (BALD): we query parameters $\boldsymbol{\theta}$ where the posterior predictive is very uncertain about the output (entropy is high), but the emulator, conditioned on the value of its parameters $\boldsymbol{\phi}$, is on average quite certain about the model output (conditional entropy low on average).

For many distributions closed-form expressions of $\mathbb{H}\big[\mathbf{x}|\boldsymbol{\theta}, \boldsymbol{\phi}\big]$ are available, but this is in general not true for the entropy of the marginal predictive distribution $\hat{p}(\mathbf{x}|\boldsymbol{\theta}, \mathcal{D})$. To overcome this problem, we derived an upper-bound approximation to the entropy term based on the law of total variance: if we characterize the marginal distribution only in terms of its (co)variance $\Sigma_{\mathcal{D}}(\boldsymbol{\theta})$, then $\mathbb{H}[\mathbf{x}|\boldsymbol{\theta}, \mathcal{D}] \;\leq\; \frac{1}{2} \ln \big[(2\pi e)^N |(\Sigma_{\mathcal{D}}(\boldsymbol{\theta}))|\big]$. Using the law of total (co)variance, we get

$$\Sigma_{\mathcal{D}}(\boldsymbol{\theta}|\mathcal{D}) = \mathrm{Cov}[\mathbf{x}|\boldsymbol{\theta}] = \mathbb{E}_{\boldsymbol{\phi}|\mathcal{D}}\big[\mathrm{Cov}[\mathbf{x}|\boldsymbol{\theta}, \boldsymbol{\phi}]\big] + \mathrm{Cov}_{\boldsymbol{\phi}|\mathcal{D}}\big[\mathbb{E}[\mathbf{x}|\boldsymbol{\theta}, \boldsymbol{\phi}]\big], \tag{6}$$

where all expectations can be approximated by samples drawn from $p(\boldsymbol{\phi}|\mathcal{D})$.

Note that the density of the forward model, $p(\mathbf{x}|\boldsymbol{\theta})$, does not appear in this rule. By using the upper-bound, we can use gradient-based optimization to find $\boldsymbol{\theta}^*$. Alternatively, entropies could be approximated using sample, which, however, would be slower.

References

S Depeweg, J M Hernández-Lobato, F Doshi-Velez, and S Udluft. Decomposition of uncertainty for active learning and reliable reinforcement learning in stochastic systems. *arXiv:1710.07283*, 2017.

Y Gal, R Islam, and Z Ghahramani. Deep bayesian active learning with image data. *arXiv:1703.02910*, 2017.

N Houlsby, F Huszar, Z Ghahramani, and M Lengyel. Bayesian active learning for classification and preference learning. *arXiv:1112.5745*, 2011.

Appendix C. Additional related work

Papamakarios et al. (2018), concurrently and independently to our approach (Lueckmann et al., 2018, an earlier preprint version of this work), proposed learning synthetic likelihoods using neural density estimators for likelihood-free inference: They use Masked Autoregressive Flows as synthetic likelihoods and report state-of-the-art performance compared to methods that directly target the posterior. Like our approach, the density estimator is trained on sequentially chosen simulations. Rather than using acquisition functions that take into account uncertainty to guide sampling, they draw samples from the current estimate of the posterior. Their approach corresponds to an alternative way of learning a local emulator.

The recent workshop paper of Durkan et al. (2018) compares Papamakarios et al. (2018) and our approach on three toy problems learning local emulators. On these toy-problems, both methods are similarly efficient (and more efficient than methods directly targeting the posterior), however, the wallclock time of our method is substantially higher, because of the additional cost of evaluating the acquisition function. Whether this additional cost is warranted on a given problem will depend both on any additional gain brought about by the active selection of samples, as well as the cost of the simulator. For expensive simulation costs, additional computational budget should be spent to carefully decide for which parameters to simulate.

References

C Durkan, G Papamakarios, and I Murray. Sequential neural methods for likelihood-free inference. *arXiv:1811.08723*, 2018.

JM Lueckmann, G Bassetto, T Karaletsos, and J H Macke. Likelihood-free inference with emulator networks. *arXiv:1805.09294v1*, 2018.

G Papamakarios, D C Sterratt, and I Murray. Sequential Neural Likelihood: Fast Likelihood-free Inference with Autoregressive Flows. *arXiv:1805.07226*, 2018.

Appendix D. Gaussian simulator example

D.1. Model

Data is generated independently according to $\mathbf{x}_i \sim \mathcal{N}(\cdot|f(\boldsymbol{\theta}), \boldsymbol{\Sigma})$, $i = 1 \ldots n$, where $n = 10$, $f(\boldsymbol{\theta}) = (1.5\ \boldsymbol{\theta} + 0.5)^3/200$, $\boldsymbol{\Sigma}_{ii} = 0.1$, $\boldsymbol{\Sigma}_{ij} = 0$ for $i \neq j$, $\bar{\mathbf{x}}_o = \frac{1}{n}\sum_i^n \bar{\mathbf{x}}_o^{(i)} = \mathbf{2}$, and $\boldsymbol{\theta}$ is distributed uniformly in $[-8, 8]^p$ where p is the dimensionality of the problem.

This problem is inspired by the Gaussian example studied in Järvenpää et al. (2017), where f was chosen as $f(\boldsymbol{\theta}) = \boldsymbol{\theta}$. We introduce a nonlinearity in f, since our method with uniform acquisitions would otherwise trivially generalize across the space – we observed that a neural network with the right amount of ReLu units can learn the linear mapping perfectly, independently of where the training samples are acquired.

D.2. Evaluation

We evaluate our method and BOLFI (Järvenpää et al., 2017) on this problem in 1D and 2D. In 1D, algorithms start with $N_{\text{initial}} = 10$ initial samples, in 2D with $N_{\text{initial}} = 25$, and make 100 acquisitions after each of which we evaluate how well the ground truth posterior is recovered.

As performance metric, we calculate total variation (TV) between $\hat{p}(\boldsymbol{\theta}|\mathbf{x}_o)$ and $p(\boldsymbol{\theta}|\mathbf{x}_o)$, defined as

$$\frac{1}{2}\int \Big|\hat{p}(\boldsymbol{\theta}|\mathbf{x}_o) - p(\boldsymbol{\theta}|\mathbf{x}_o)\Big| \mathrm{d}\boldsymbol{\theta}.$$

D.3. Network architecture and training

Emulator networks model a normal distribution as output, so that the outputs of the network parametrise mean and covariance (Cholesky factor of the covariance matrix). Neural networks have one hidden layer consisting of 10 tanh units. We train an ensemble of $M = 50$ networks using Adam (Kingma and Ba, 2014) with default parameters ($\beta_1 = 0.9, \beta_2 = 0.999$) for SGD, and a learning rate of 0.01.

D.4. BOLFI

BOLFI requires choice of a distance function: We use the the Mahalanobis distance

$$\Delta_{\boldsymbol{\theta}} = \left((\bar{\mathbf{x}} - \bar{\mathbf{x}}_o)^T \boldsymbol{\Sigma}^{-1} (\bar{\mathbf{x}} - \bar{\mathbf{x}}_o)\right)^{1/2},$$

in line with the distance function used for the Gaussian example studied in Järvenpää et al. (2017). We use the implementation provided by the authors (Lintusaari et al., 2017).

References

M Järvenpää, M U Gutmann, A Pleska, Vehtari, A, and P Marttinen. Efficient acquisition rules for model-based approximate Bayesian computation. *arXiv:1704.00520v2*, 2017.

D P Kingma and J Ba. Adam: a method for stochastic optimization. *arXiv:1412.6980*, 2014.

J Lintusaari, H Vuollekoski, A Kangasrääsiö, K Skytén, M Järvenpää, M Gutmann, A Vehtari, J Corander, and S Kaski. Elfi: Engine for likelihood free inference. *arXiv:1708.00707*, 2017.

Appendix E. Image example

E.1. Model

Images are generated according to:

$$\begin{aligned}
I_{xy} &\sim \mathrm{Bin}(\cdot|255, p_{xy}) \\
p_{xy} &= 0.9 - 0.8 \exp^{-0.5\left(r_{xy}/\sigma^2\right)^{\gamma}} \\
r_{xy} &= (x - x_{\text{off}})^2 + (y - y_{\text{off}})^2,
\end{aligned}$$

where x and y are coordinates in the image, and $\mathrm{Bin}(\cdot|n, p)$ is the binomial distribution.

Model parameters are x_{off} and y_{off}, which respectively determine the horizontal and the vertical offset of the blob, γ, defining its contrast, and σ^2, determining the width.

For our experiments, we use images of size 32×32 pixels. We choose uniform priors in the range $[-16, 16]$ for x_{off} and y_{off}, and a uniform prior in the range $[0.25, 5]$ for γ. We fix σ to 2.

E.2. Evaluation

We evaluate different acquisition methods by keeping track of the log-likelihood of a test set consisting of 5000 parameters-image pairs over the course of acquisitions (starting from an initial sample of size $N_{\text{initial}} = 50$).

E.3. Network architecture and training

Emulator networks model a binomial distribution as output. Neural networks have two hidden layers (200 units each) with ReLu activation functions. We train an ensemble of $M = 25$ networks using Adam (Kingma and Ba, 2014) with default parameters ($\beta_1 = 0.9, \beta_2 = 0.999$) for SGD with a learning rate of 0.001.

References

D P Kingma and J Ba. Adam: a method for stochastic optimization. *arXiv:1412.6980*, 2014.

Appendix F. Hodgkin-Huxley example

F.1. Model

The dynamic equations describing the evolution of the membrane potential and of the gating variables of the neuron are taken from Pospischil et al. (2008):

$$\begin{aligned} C_m\dot{V} &= -(I_{\text{leak}} + I_{\text{Na}} + I_{\text{K}} + I_{\text{M}} + I_{\text{ext}}) \\ &= g_{\text{leak}}(E_{\text{leak}} - V) + \bar{g}_{\text{Na}}m^3h(E_{\text{Na}} - V) + \\ &+ \bar{g}_{\text{K}}n^4(E_{\text{K}} - V) + \bar{g}_{\text{M}}p(E_{\text{K}} - V) + I_{\text{in}}(t), \end{aligned}$$

where C_m is membrane capacitance, V the membrane potential, I_c are ionic currents ($c = \{\text{Na}, \text{K}, \text{M}\}$) and $I_{\text{in}}(t)$ is an externally applied current which we can imagine as the sum of a static bias I_{bias} and a time-varying zero-mean noise signal $\varepsilon(t)$. I_{Na} and I_{K} shape the up- and down-stroke phases of the action potential (spike), I_{M} is responsible for spike-frequency adaptation, and I_{leak} is a leak current describing the passive properties of the cell membrane. Each current is in turn expressed as the product of a maximum conductance ($\bar{g}_c$) and the voltage difference between the membrane potential and the reversal potential for that current(E_c), possibly modulated by zero or more 'gating' variables (m, h, n, p).

Each $x \in \{m, h, n, p\}$ evolves according to first order kinetics in the form:

$$\dot{x} = \frac{1}{\tau_x(V)}\big(x_\infty(V) - x\big)$$

We provide a step current as input.

In our example application, free model parameters are $\bar{g}_{\text{Na}}$ and $\bar{g}_{\text{K}}$. We model uniform priors over these parameters: $\bar{g}_{\text{Na}}$ is between 0.5 and 60 and $\bar{g}_{\text{K}}$ is between 0.5 and 10.

F.2. Evaluation

We evaluate the posterior obtained through the emulator after $t = 250$ acquisitions, starting from an initial sample size $N_{\text{initial}} = 30$. As posterior predictive check, we span a grid over the parameter space and compare simulator outputs to the posterior.

F.3. Network architecture and training

Emulator networks model a categorical distribution with $K = 6$ classes as output. Neural networks have two hidden layer (200 units each) with a ReLu activation functions. We train an ensemble of $M = 25$ networks using Adam (Kingma and Ba, 2014) with default parameters ($\beta_1 = 0.9, \beta_2 = 0.999$) for SGD with a learning rate of 0.001.

References

D P Kingma and J Ba. Adam: a method for stochastic optimization. *arXiv:1412.6980*, 2014.

M Pospischil, M Toledo-Rodriguez, C Monier, Z Piwkowska, T Bal, Y Frégnac, H Markram, and A Destexhe. Minimal hodgkin-huxley type models for different classes of cortical and thalamic neurons. *Biological Cybernetics*, 99(4-5), 2008.

Appendix G. BOLFI convergence

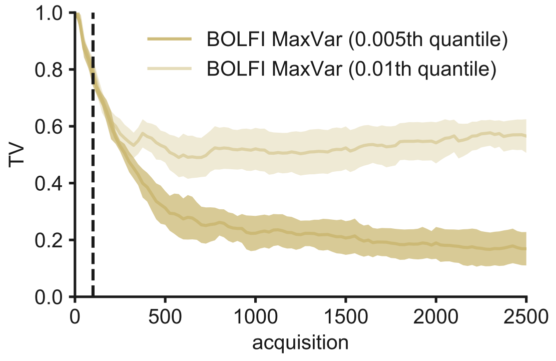


Figure 5: **Convergence of BOLFI *MaxVar*.** In the manuscript, we show performance up to 100 acquisitions (indicated by the dotted line). With additional acquisitions, BOLFI converges. The quality of the inferred posterior strongly depends on the value of the threshold hyperparameter used in BOLFI.

Appendix H. Posteriors and samples for image example

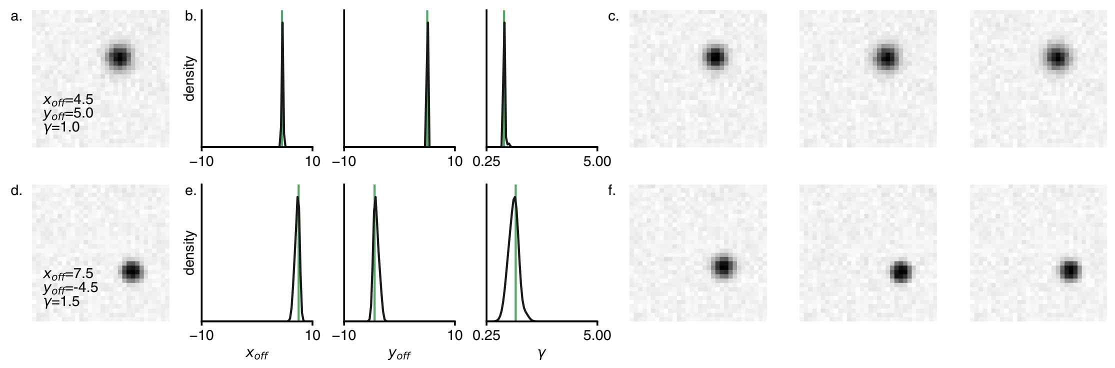


Figure 6: **Posteriors and samples for image example. a.** Observed image. **b.** Inferred posteriors. **c.** Posterior samples. **d.** Another observation, with posteriors in **e.** and samples in **f.**

Appendix I. *MaxMI* acquisition for Hodgkin-Huxley model

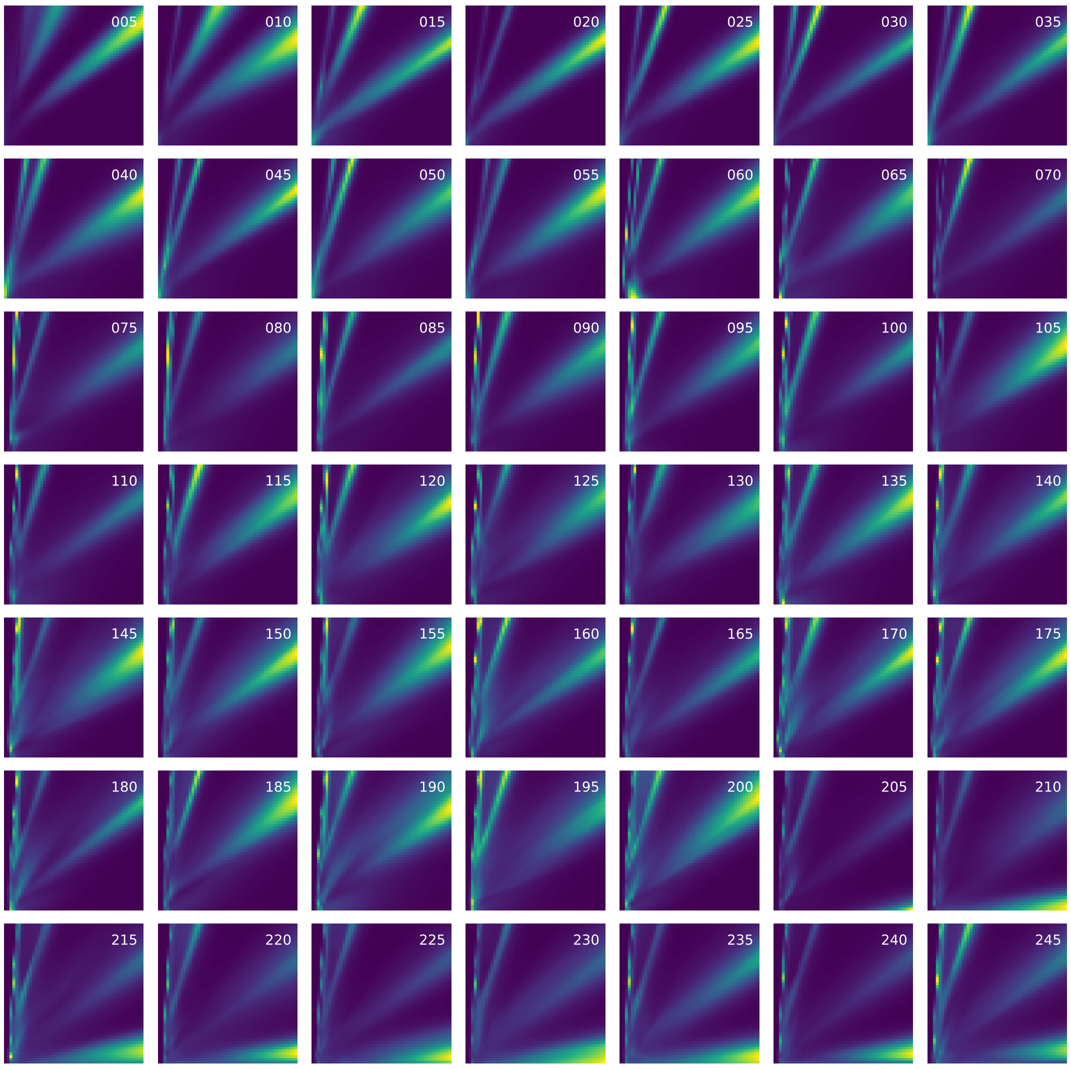


Figure 7: **Acquistion surface for *MaxMI* rule on Hodgkin-Huxley example.** Individual panels show the acquisition surface over $\boldsymbol{\theta}$ as additional samples have been acquired. The acquisition rule proposes datapoints at the decision boundaries of the posterior.

Benchmarking Simulation-Based Inference

Jan-Matthis Lueckmann^{1,2} **Jan Boelts**^{2} **David S. Greenberg**^{2,3}
Pedro J. Gonçalves^{4} **Jakob H. Macke**^{1,2,5}

^{1}University of Tübingen ^{2}Technical University of Munich ^{3}Helmholtz Centre Geesthacht
^{4}Research Center caesar ^{5}Max Planck Institute for Intelligent Systems, Tübingen

Abstract

Recent advances in probabilistic modelling have led to a large number of simulation-based inference algorithms which do not require numerical evaluation of likelihoods. However, a public benchmark with appropriate performance metrics for such 'likelihood-free' algorithms has been lacking. This has made it difficult to compare algorithms and identify their strengths and weaknesses. We set out to fill this gap: We provide a benchmark with inference tasks and suitable performance metrics, with an initial selection of algorithms including recent approaches employing neural networks and classical Approximate Bayesian Computation methods. We found that the choice of performance metric is critical, that even state-of-the-art algorithms have substantial room for improvement, and that sequential estimation improves sample efficiency. Neural network-based approaches generally exhibit better performance, but there is no uniformly best algorithm. We provide practical advice and highlight the potential of the benchmark to diagnose problems and improve algorithms. The results can be explored interactively on a companion website. All code is open source, making it possible to contribute further benchmark tasks and inference algorithms.

1 Introduction

Many domains of science, engineering, and economics make extensive use of models implemented as stochastic numerical simulators (Gourieroux et al., 1993; Ratmann et al., 2007; Alsing et al., 2018; Brehmer et al., 2018; Karabatsos and Leisen, 2018; Gonçalves et al., 2020). A key challenge when studying and validating such simulation-based models is the statistical identification of parameters which are consistent with observed data. In many cases, calculation of the likelihood is intractable or impractical, rendering conventional approaches inapplicable. The goal of simulation-based inference (SBI), also known as 'likelihood-free inference', is to perform Bayesian inference without requiring numerical evaluation of the likelihood function (Sisson et al., 2018; Cranmer et al., 2020). In SBI, it is generally not required that the simulator is differentiable, nor that one has access to its internal random variables.

In recent years, several new SBI algorithms have been developed (e.g., Gutmann and Corander, 2016; Papamakarios and Murray, 2016; Lueckmann et al., 2017; Chan et al., 2018; Greenberg et al., 2019; Papamakarios et al., 2019b; Prangle, 2019; Brehmer et al., 2020; Hermans et al., 2020; Järvenpää et al., 2020; Picchini et al., 2020; Rodrigues et al., 2020; Thomas et al., 2020), energized, in part, by advances in probabilistic machine learning (Rezende and Mohamed, 2015; Papamakarios et al., 2017, 2019a). Despite—or possibly *because*—of these rapid and exciting developments, it is currently difficult to assess how different approaches relate to each other theoretically and empirically: First, different studies often use different tasks and metrics for comparison, and comprehensive comparisons on multiple tasks and simulation budgets are rare. Second, some commonly employed metrics might not be appropriate or might be biased through the choice of hyperparameters. Third, the absence of a benchmark has made it necessary to reimplement tasks and algorithms for each new study. This practice is wasteful, and makes it hard to rapidly evaluate the potential of new algorithms. Overall, it is difficult to discern the most promising approaches and decide on which algorithm to use when. These problems are exacerbated by the interdisciplinary nature of research on SBI, which has

Proceedings of the 24th International Conference on Artificial Intelligence and Statistics (AISTATS) 2021, San Diego, California, USA. PMLR: Volume 130. Copyright 2021 by the author(s).

led to independent development and co-existence of closely-related algorithms in different disciplines.

There are many exciting challenges and opportunities ahead, such as the scaling of these algorithms to high-dimensional data, active selection of simulations, and gray-box settings, as outlined in Cranmer et al. (2020). To tackle such challenges, researchers will need an extensible framework to compare existing algorithms and test novel ideas. Carefully curated, a benchmark framework will make it easier for researchers to enter SBI research, and will fuel the development of new algorithms through community involvement, exchange of expertise and collaboration. Furthermore, benchmarking results could help practitioners to decide which algorithm to use on a given problem of interest, and thereby contribute to the dissemination of SBI.

The catalyzing effect of benchmarks has been evident, e.g., in computer vision (Russakovsky et al., 2015), speech recognition (Hirsch and Pearce, 2000; Wang et al., 2018), reinforcement learning (Bellemare et al., 2013; Duan et al., 2016), Bayesian deep learning (Filos et al., 2019; Wenzel et al., 2020), and many other fields drawing on machine learning. Open benchmarks can be an important component of transparent and reproducible computational research. Surprisingly, a benchmark framework for SBI has been lacking, possibly due to the challenging endeavor of designing benchmarking tasks and defining suitable performance metrics.

Here, we begin to address this challenge, and provide a benchmark framework for SBI to allow rapid and transparent comparisons of current and future SBI algorithms: First, we selected a set of initial algorithms representing distinct approaches to SBI (Fig. 1; Cranmer et al., 2020). Second, we analyzed multiple performance metrics which have been used in the SBI literature. Third, we implemented ten tasks including tasks popular in the field. The shortcomings of commonly used metrics led us to focus on tasks for which a likelihood *can* be evaluated, which allowed us to calculate reference ('ground-truth') posteriors. These reference posteriors are made available to allow rapid evaluation of SBI algorithms. Code for the framework is available at `github.com/sbi-benchmark/sbibm` and we maintain an interactive version of all results at `sbi-benchmark.github.io`.

The full potential of the benchmark will be realized when it is populated with additional community-contributed algorithms and tasks. However, our initial version already provides useful insights: 1) the choice of performance metric is critical; 2) the performance of the algorithms on some tasks leaves substantial room for improvement; 3) sequential estimation generally improves sample efficiency; 4) for small and moderate simulation budgets, neural-network based approaches outperform classical ABC algorithms, confirming recent progress in the field; and 5) there is no algorithm to rule them all. The performance ranking of algorithms is task-dependent, pointing to a need for better guidance or automated procedures for choosing which algorithm to use when. We highlight examples of how the benchmark can be used to diagnose shortcomings of algorithms and facilitate improvements. We end with a discussion of the limitations of the benchmark.

2 Benchmark

The benchmark consists of a set of algorithms, performance metrics and tasks. Given a prior $p(\boldsymbol{\theta})$ over parameters $\boldsymbol{\theta}$, a simulator to sample $\mathbf{x} \sim p(\mathbf{x}|\boldsymbol{\theta})$ and an observation $\mathbf{x}_o$, an algorithm returns an approximate posterior $q(\boldsymbol{\theta}|\mathbf{x}_o)$, or samples from it, $\boldsymbol{\theta} \sim q$. The approximate solution is tested, according to a performance metric, against a reference posterior $p(\boldsymbol{\theta}|\mathbf{x}_o)$.

2.1 Algorithms

Following the classification introduced in the review by Cranmer et al. (2020), we selected algorithms addressing SBI in four distinct ways, as schematically depicted in Fig. 1. An important difference between algorithms is how new simulations are acquired: Sequential algorithms adaptively choose informative simulations to increase sample efficiency. While crucial for expensive simulators, it can require non-trivial algorithmic steps and hyperparameter choices. To evaluate whether the potential is realized empirically and justifies the algorithmic burden, we included sequential and non-sequential counterparts for algorithms of each category.

Keeping our initial selection focused allowed us to carefully consider implementation details and hyperparameters: We extensively explored performance and sensitivity to different choices in more than 10k runs, all results and details of which can be found in Appendix H. Our selection is briefly described below, full algorithm details are in Appendix A.

REJ-ABC and SMC-ABC. Approximate Bayesian Computation (ABC, Sisson et al., 2018) is centered around the idea of Monte Carlo rejection sampling (Tavaré et al., 1997; Pritchard et al., 1999). Parameters $\boldsymbol{\theta}$ are sampled from a proposal distribution, simulation outcomes $\mathbf{x}$ are compared with observed data $\mathbf{x}_o$, and are accepted or rejected depending on a (user-specified) distance function and rejection criterion. While rejection ABC (REJ-ABC) uses the prior as a proposal distribution, the efficiency can be improved by using sequentially refined proposal distributions (SMC-ABC, Beaumont et al., 2002; Marjoram and Tavaré, 2006;

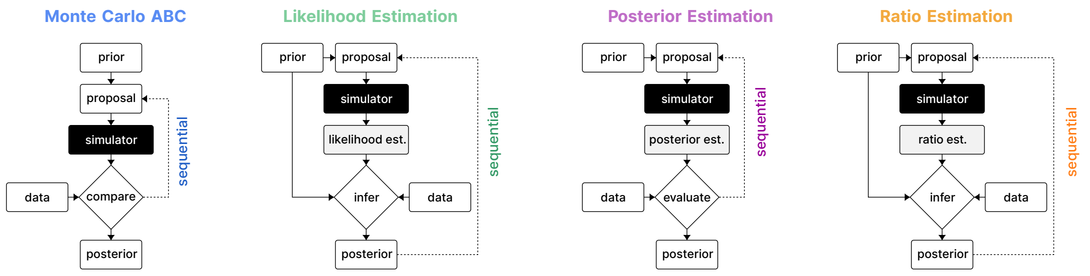


Figure 1: **Overview of algorithms.** We compare algorithms belonging to four distinct approaches to SBI: Classical ABC approaches as well as model-based approaches approximating likelihoods, posteriors, or density ratios. We contrast algorithms that use the prior distribution to propose parameters against ones that sequentially adapt the proposal. Classification and schemes following Cranmer et al. (2020).

Sisson et al., 2007; Toni et al., 2009; Beaumont et al., 2009). We implemented REJ-ABC with quantile-based rejection and used the scheme of Beaumont et al. (2009) for SMC-ABC. We extensively varied hyperparameters and compared the implementation of an ABC-toolbox (Klinger et al., 2018) against our own (Appendix H). We investigated linear regression adjustment (Blum and François, 2010) and the summary statistics approach by Prangle et al. (2014) (Suppl. Fig. 1).

NLE and SNLE. Likelihood estimation (or 'synthetic likelihood') algorithms learn an approximation to the intractable likelihood, for an overview see Drovandi et al. (2018). While early incarnations focused on Gaussian approximations (SL; Wood, 2010), recent versions utilize deep neural networks (Papamakarios et al., 2019b; Lueckmann et al., 2019) to approximate a density over $\mathbf{x}$, followed by MCMC to obtain posterior samples. Since we primarily focused on these latter versions, we refer to them as neural likelihood estimation (NLE) algorithms, and denote the sequential variant with proposals as SNLE. In particular, we used the scheme proposed by Papamakarios et al. (2019b) which uses masked autoregressive flows (MAFs, Papamakarios et al., 2017) for density estimation. We improved MCMC sampling for (S)NLE and compared MAFs against Neural Spline Flows (NSFs; Durkan et al., 2019), see Appendix H.

NPE and SNPE. Instead of approximating the likelihood, these approaches directly target the posterior. Their origins date back to regression adjustment approaches (Blum and François, 2010). Modern variants (Papamakarios and Murray, 2016; Lueckmann et al., 2017; Greenberg et al., 2019) use neural networks for density estimation (approximating a density over $\boldsymbol{\theta}$). Here, we used the recent algorithmic approach proposed by Greenberg et al. (2019) for sequential acquisitions. We report performance using NSFs for density estimation, which outperformed MAFs (Appendix H).

NRE and SNRE. Ratio Estimation approaches to SBI use classifiers to approximate density ratios (Izbicki et al., 2014; Pham et al., 2014; Cranmer et al., 2015; Dutta et al., 2016; Durkan et al., 2020; Thomas et al., 2020). Here, we used the recent approach proposed by Hermans et al. (2020) as implemented in Durkan et al. (2020): A neural network-based classifier approximates probability ratios and MCMC is used to obtain samples from the posterior. SNRE denotes the sequential variant of neural ratio estimation (NRE). In Appendix H we compare different classifier architectures for (S)NRE.

In addition, we benchmarked Random Forest ABC (RF-ABC; Raynal et al., 2019), a recent ABC variant, and Synthetic Likelihood (SL; Wood, 2010), mentioned above. However, RF-ABC only targets individual parameters (i.e. assumes posteriors to factorize), and SL requires new simulations for every MCMC step, thus requiring orders of magnitude more simulations than other algorithms. Therefore, we report results for these algorithms separately, in Suppl. Fig. 2 and Suppl. Fig. 3, respectively.

Algorithms can be grouped with respect to how their output is represented: 1) some return samples from the posterior, $\boldsymbol{\theta} \sim q(\boldsymbol{\theta}|\mathbf{x}_o)$ (REJ-ABC, SMC-ABC); 2) others return samples and allow evaluation of unnormalized posteriors $\tilde{q}(\boldsymbol{\theta}|\mathbf{x}_o)$ ((S)NLE, (S)NRE); and 3) for some, the posterior density $q(\boldsymbol{\theta}|\mathbf{x}_o)$ can be evaluated and sampled directly, without MCMC ((S)NPE). As discussed below, these properties constrain the metrics that can be used for comparison.

2.2 Performance metrics

Choice of a suitable performance metric is central to any benchmark. As the goal of SBI algorithms is to perform full inference, the 'gold standard' would be to quantify the similarity between the true posterior and the inferred one with a suitable distance (or divergence) measure on probability distributions. This would require both access to the ground-truth posterior, and a reliable means of estimating similarity between (potentially) richly structured distributions. Several performance metrics have been used in past research, depending on the constraints imposed by knowledge about ground-truth and the inference algorithm (see Table 1). In real-world applications, typically only the observation $\mathbf{x}_o$ is known. However, in a benchmarking setting, it is reasonable to assume that one has at least access to the ground-truth parameters $\boldsymbol{\theta}_o$. There are two commonly used metrics which only require $\boldsymbol{\theta}_o$ and $\mathbf{x}_o$, but suffer severe drawbacks for our purposes:

Probability $\boldsymbol{\theta}_o$. The negative log probability of true parameters averaged over different $(\boldsymbol{\theta}_o, \mathbf{x}_o)$, $-\mathbb{E}[\log q(\boldsymbol{\theta}_o|\mathbf{x}_o)]$, has been used extensively in the literature (Papamakarios and Murray, 2016; Durkan et al., 2018; Greenberg et al., 2019; Papamakarios et al., 2019b; Durkan et al., 2020; Hermans et al., 2020). Its appeal lies in the fact that one does not need access to the ground-truth posterior. However, using it only for a small set of $(\boldsymbol{\theta}_o, \mathbf{x}_o)$ is highly problematic: It is only a valid performance measure if averaged over a large set of observations sampled from the prior (Talts et al., 2018, detailed discussion including connection to simulation-based calibration in Appendix M). For reliable results, one would require inference for hundreds of $\mathbf{x}_o$ which is only feasible if inference is rapid (amortized) and the density q can be evaluated directly (among the algorithms used here this applies only to NPE).

Posterior-Predictive Checks (PPCs). As the name implies, PPCs should be considered a mere check rather than a metric, although the *median distance* between predictive samples and $\mathbf{x}_o$ has been reported in the SBI literature (Papamakarios et al., 2019b; Greenberg et al., 2019; Durkan et al., 2020). A failure mode of such a metric is that an algorithm obtaining a good MAP point estimate, could perfectly pass this check even if the estimated posterior is poor. Empirically, we found median-distances (MEDDIST) to be in disagreement with other metrics (see Results).

The shortcomings of these commonly-used metrics led us to focus on tasks for which it is possible to get samples from ground-truth posterior $\boldsymbol{\theta} \sim p$, thus allowing us to use metrics based on two-sample tests:

Maximum Mean Discrepancy (MMD). MMD (Gretton et al., 2012; Sutherland et al., 2017) is a kernel-based 2-sample test. Recent papers (Papamakarios et al., 2019b; Greenberg et al., 2019; Hermans et al., 2020) reported MMD using translation-invariant Gaussian kernels with length scales determined by the median heuristic (Ramdas et al., 2015). We empirically found that MMD can be sensitive to hyperparameter choices, in particular on posteriors with multiple modes and length scales (see Results and Liu et al., 2020).

Classifier 2-Sample Tests (C2ST). C2STs (Friedman, 2004; Lopez-Paz and Oquab, 2017) train a classifier to discriminate samples from the true and inferred posteriors, which makes them simple to apply and easy to interpret. Therefore, we prefer to report and compare algorithms in terms of accuracy in classification-based tests. In the context of SBI, C2ST has e.g. been used in Gutmann et al. (2018); Dalmasso et al. (2020).

Other metrics that could be used include:

Kernelized Stein Discrepancy (KSD). KSD (Liu et al., 2016; Chwialkowski et al., 2016) is a 1-sample test, which require access to $\nabla_{\boldsymbol{\theta}}\, \tilde{p}(\boldsymbol{\theta}|\mathbf{x}_o)$ rather than samples from p ($\tilde{p}$ is the unnormalized posterior). Like MMD, current estimators use translation-invariant kernels.

f-Divergences. Divergences such as Total Variation (TV) divergence and KL divergences can only be computed when the densities of true and approximate posteriors can be evaluated (Table 1). Thus, we did not use f-divergences for the benchmark.

Full discussion and details of metrics in Appendix M.

2.3 Tasks

The preceding considerations guided our selection of inference tasks: We focused on tasks for which reference posterior samples $\boldsymbol{\theta} \sim p$ can be obtained, to allow calculation of 2-sample tests. We focused on eight purely statistical problems and two problems relevant in applied domains, with diverse dimensionalities of parameters and data (details in Appendix T):

Gaussian Linear/Gaussian Linear Uniform. We included two versions of simple, linear, 10-d Gaussian models, in which the parameter $\boldsymbol{\theta}$ is the mean, and the covariance is fixed. The first version has a Gaussian (conjugate) prior, the second one a uniform prior. These tasks allow us to test how algorithms deal with trivial scaling of dimensionality, as well as truncated support.

SLCP/SLCP Distractors. A challenging inference task designed to have a simple likelihood and a complex posterior (Papamakarios et al., 2019b; Greenberg et al., 2019): The prior is uniform over five parameters $\boldsymbol{\theta}$ and the data are a set of four two-dimensional points sampled from a Gaussian likelihood whose mean and variance are nonlinear functions of $\boldsymbol{\theta}$. This induces a complex posterior with four symmetrical modes and vertical cut-offs. We included a second version with 92 additional, non-informative outputs (distractors) to test the ability to detect informative features.

Bernoulli GLM/Bernoulli GLM Raw. 10-parameter Generalized Linear Model (GLM) with

Table 1: **Applicability of metrics given knowledge about ground truth and algorithm.** Whether a metric can be used depends on *both* what is known about the ground-truth of an inference task and what an algorithm returns: Information about ground truth can vary between just having observed data $\mathbf{x}_o$ (typical setting in practice), knowing the generating parameter $\boldsymbol{\theta}_o$, having posterior samples, gradients, or being able to evaluate the true posterior p. Tilde denotes unnormalized distributions. Access to information is cumulative.

	Ground truth $\rightarrow$				
$\downarrow$ **Algorithm**	$\mathbf{x}_o$	$\boldsymbol{\theta}_o$	$\boldsymbol{\theta} \sim p$	$\nabla \tilde{p}(\boldsymbol{\theta}\|\mathbf{x}_o)$	$p(\boldsymbol{\theta}\|\mathbf{x}_o)$
$\boldsymbol{\theta} \sim q$	1	1	1, 3	1, 3, 4	1, 3, 4
$\tilde{q}(\boldsymbol{\theta}\|\mathbf{x}_o)$	1	1	1, 3	1, 3, 4	1, 3, 4
$q(\boldsymbol{\theta}\|\mathbf{x}_o)$	1	1, 2	1, 2, 3	1, 2, 3, 4	1, 2, 3, 4, 5

1 = PPCs, 2 = Probability $\boldsymbol{\theta}_0$, 3 = 2-sample tests, 4 = 1-sample tests, 5 = f-divergences.

Bernoulli observations. Inference was either performed on sufficient statistics (10-d) or raw data (100-d).

Gaussian Mixture. This inference task, introduced by Sisson et al. (2007), has become common in the ABC literature (Beaumont et al., 2009; Toni et al., 2009; Simola et al., 2020). It consists of a mixture of two two-dimensional Gaussian distributions, one with much broader covariance than the other.

Two Moons. A two-dimensional task with a posterior that exhibits both global (bimodality) and local (crescent shape) structure (Greenberg et al., 2019) to illustrate how algorithms deal with multimodality.

SIR. Dynamical systems represent paradigmatic use cases for SBI. SIR is an influential epidemiological model describing the dynamics of the number of individuals in three possible states: susceptible S, infectious I, and recovered or deceased, R. We infer the contact rate β and the mean recovery rate γ, given observed infection counts I at 10 evenly-spaced time points.

Lotka-Volterra. An influential model in ecology describing the dynamics of two interacting species, widely used in SBI studies. We infer four parameters $\boldsymbol{\theta}$ related to species interaction, given the number of individuals in both populations at 10 evenly-spaced points in time.

2.4 Experimental Setup

For each task, we sampled 10 sets of true parameters from the prior and generated corresponding observations $(\boldsymbol{\theta}_o, \mathbf{x}_o)_{1:10}$. For each observation, we generated 10k samples from the reference posterior. Some reference posteriors required a customised (likelihood-based) approach (Appendix B).

In SBI, it is typically assumed that total computation cost is dominated by simulation time. We therefore report performance at different simulation budgets. For each observation, each algorithm was run with a simulation budget ranging from 1k to 100k simulations.

For each run, we calculated metrics described above. To estimate C2ST accuracy, we trained a multilayer perceptron to tell apart approximate and reference posterior samples and performed five-fold cross-validation. We used two hidden layers, each with 10 times as many ReLu units as the dimensionality of the data. We also measured and report runtimes (Appendix R).

2.5 Software

Code. All code is released publicly at `github.com/sbi-benchmark/sbibm`. Our framework includes tasks, reference posteriors, metrics, plotting, and infrastructure tooling and is designed to be 1) easily extensible, 2) used with external toolboxes implementing algorithms. All tasks are implemented as probabilistic programs in `Pyro` (Bingham et al., 2019), so that likelihoods and gradients for reference posteriors can be extracted automatically. To make this possible for tasks that use ODEs, we developed a new interface between `DifferentialEquations.jl` (Rackauckas and Nie, 2017; Bezanson et al., 2017) and `PyTorch` (Paszke et al., 2019). In addition, specifying simulators in a probabilistic programming language has the advantage that 'gray-box' algorithms (Brehmer et al., 2020; Cranmer et al., 2020) can be added in the future. We here evaluated algorithms implemented in `pyABC` (Klinger et al., 2018), `pyabcranger` (Collin et al., 2020), and `sbi` (Tejero-Cantero et al., 2020). See Appendix B for details and existing SBI toolboxes.

Reproducibility. Instructions to reproduce experiments on cloud-based infrastructure are in Appendix B.

Website. Along with the code, we provide a web interface which allows interactive exploration of all the results (`sbi-benchmark.github.io`; Appendix W).

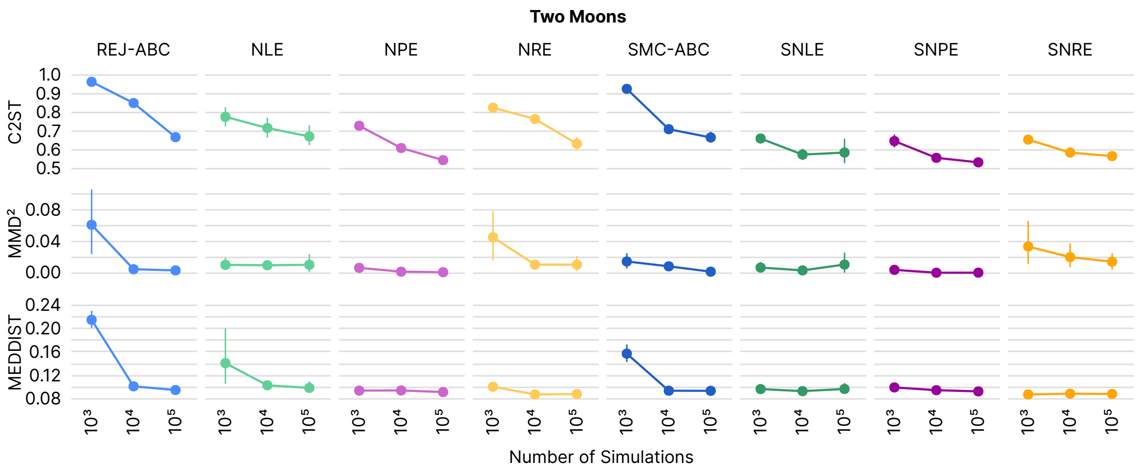


Figure 2: **Performance on Two Moons according to various metrics.** Best possible performance would be 0.5 for C2ST, 0 for MMD^2 and MEDDIST. Results for 10 observations each, means and 95% confidence intervals.

3 Results

We first consider empirical results on a single task, Two Moons, according to different metrics, which illustrate the following important insight:

#1: Choice of performance metric is key. While C2ST results on Two Moons show that performance increases with higher simulation budgets and that sequential algorithms outperform non-sequential ones for low to medium budgets, these results were not reflected in MMD and MEDDIST (Fig. 2): In our analyses, we found MMD to be sensitive to hyperparameter choices, in particular on tasks with complex posterior structure. When using the commonly employed median heuristic to set the kernel length scale on a task with multi-modal posteriors (like Two Moons), MMD had difficulty discerning markedly different posteriors. This can be 'fixed' by using hyperparameters adapted to the task (Suppl. Fig. 4). As discussed above, the median distance (though commonly used) can be 'gamed' by a good point estimate even if the estimated posterior is poor and is thus not a suitable performance metric. Computation of KSD showed numerical problems on Two Moons, due to the gradient calculation.

We assessed relationships between metrics empirically via the correlations across tasks (Suppl. Fig. 5). As discussed above, the log-probability of ground-truth parameters can be problematic when averaged over too few observations (e.g., 10, as is common in the literature): indeed, this metric had a correlation of only 0.3 with C2ST on Two Moons and 0.6 on the SLCP task. Based on these considerations, we used C2ST for reporting performance (Fig. 3; results for MMD, KSD and median distance on the website).

Based on the comparison of the performance across all tasks, we highlight the following main points:

#2: These are not solved problems. C2ST uses an interpretable scale (1 to 0.5), which makes it possible to conclude that, for several tasks, no algorithm could solve them with the specified budget (e.g., SLCP, Lotka-Volterra). This highlights that our problems—though conceptually simple—are challenging, and there is room for development of more powerful algorithms.

#3: Sequential estimation improves sample efficiency. Our results show that sequential algorithms outperform non-sequential ones (Fig. 3). The difference was small on simple tasks (i.e. linear Gaussian cases), yet pronounced on most others. However, we also found these methods to exhibit diminishing returns as the simulation budget grows, which points to an opportunity for future improvements.

#4: Density or ratio estimation-based algorithms generally outperform classical techniques. REJ-ABC and SMC-ABC were generally outperformed by more recent techniques which use neural networks for density- or ratio-estimation, and which can therefore efficiently interpolate between different simulations (Fig. 3). Without such model-based interpolation, even a simple 10-d Gaussian task can be challenging. However, classical rejection-based methods have a computational footprint that is orders of magnitudes smaller, as no network training is involved (Appendix R). Thus, on low-dimensional problems and for cheap simulators, these methods can still be competitive. See Suppl. Fig. 1 for results with additional ABC variants (Blum and François, 2010; Prangle et al., 2014) and Suppl. Fig. 2 for results on RF-ABC.

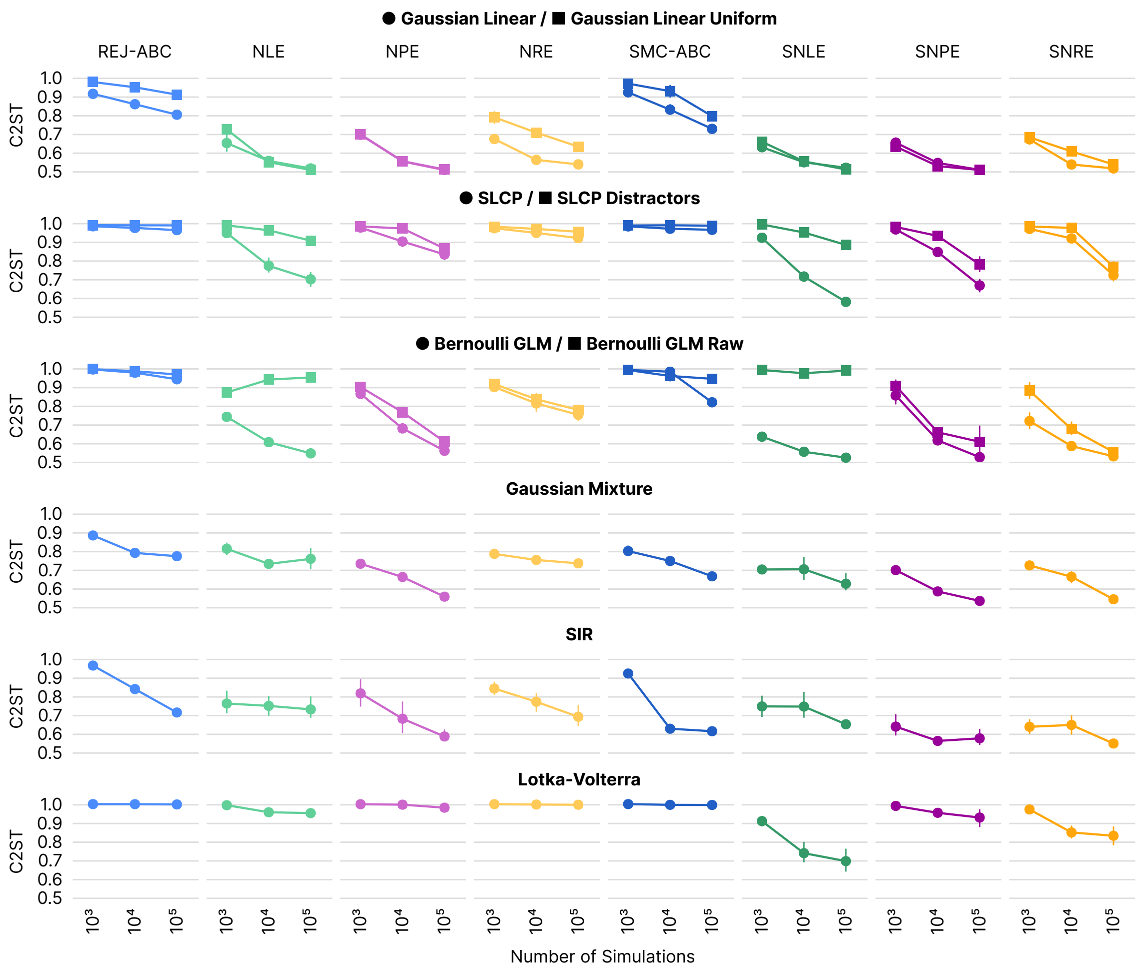


Figure 3: **Performance on other benchmark tasks.** Classification accuracy (C2ST) of REJ-ABC, SMC-ABC, NLE, SNLE, NPE, SNPE, NRE, SNRE for 10 observations each, means and 95% confidence intervals.

#5: No one algorithm to rule them all. Although sequential density or ratio estimation-based algorithms performed better than their non-sequential counterparts, there was no clear-cut answer as to which sequential method (SNLE, SNRE, and SNPE) should be preferred. To some degree, this is to be expected: these algorithms have distinct strengths that can play out differently depending on the problem structure (see discussions e.g., in Greenberg et al., 2019; Durkan et al., 2018, 2020). However, this has not shown systematically before. We formulate some practical guidelines for choosing appropriate algorithms in Box 1.

#6: The benchmark can be used to diagnose implementation issues and improve algorithms. For example, (S)NLE and (S)NRE rely on MCMC sampling to compute posteriors, and this sampling step can limit the performance. Access to a reference posterior can help identify and improve such issues: We found that single chains initialized by sampling from the prior with axis-aligned slice sampling (as introduced in Papamakarios et al., 2019b) frequently got stuck in single modes. Based on this observation, we changed the MCMC strategy (details in Appendix A), which, though simple, yielded significant performance and speed improvements on the benchmark tasks. Similarly, (S)NLE and (S)NRE improved by transforming parameters to be unbounded: Without transformations, runs on some tasks can get stuck during MCMC sampling (e.g., Lotka-Volterra). While this is common advice for MCMC (Hogg and Foreman-Mackey, 2018), it has been lacking in code and papers of SBI approaches.

We used the benchmark to systematically compare hyperparameters: For example, as density estimators

Do we need the Bayesian posterior, or is a point estimate sufficient?

Our focus was on SBI algorithms that target the Bayesian posterior. If one only aims for a single estimate, optimization methods (e.g. Rios and Sahinidis, 2013; Shahriari et al., 2015) might be more efficient.

Is the simulator really 'black-box'?

The SBI algorithms presented in the benchmark can be applied to any 'black-box' simulator. However, if the likelihood is available, methods exploiting it (e.g. MCMC, variational inference) will generally be more efficient. Similarly, if one has access to the internal random numbers, probabilistic programming approaches (Le et al., 2017; Baydin et al., 2019; Wood et al., 2020) might be preferable. If additional quantities that characterize the latent process are available, i.e., the simulator is 'gray-box', they can be used to augment training data and improve inference (Brehmer et al., 2020; Cranmer et al., 2020).

What domain knowledge do we have about the problem?

For any practical application of SBI, it is worth thinking carefully about domain knowledge. First, knowledge about plausible parameters should inform the choice of the prior. Second, domain knowledge can help design appropriate distance functions or summary statistics required for classical ABC algorithms. When using model-based approaches, domain knowledge can potentially be built into the SBI algorithm itself, for example, by incorporating neural network layers with appropriate inductive biases or invariances.

Do we have, or can we learn summary statistics?

Summary statistics are especially important when facing problems with high-dimensional data: It is important to point out that the posterior given summary statistics $p(\boldsymbol{\theta}|s(\mathbf{x}_o))$ is only equivalent to $p(\boldsymbol{\theta}|\mathbf{x}_o)$ if the summary statistics are sufficient. The problem at hand can guide the manual design of summary statistics that are regarded particularly important or informative. Alternatively, many automatic approaches exist (e.g., Prangle et al., 2014; Charnock et al., 2018; Dinev and Gutmann, 2018) and this is an active area of research (e.g., Chen et al. 2021 recently proposed an approach to learn approximately sufficient statistics for SMC-ABC and (S)NLE). (S)NPE and (S)NRE can directly reduce high-dimensional data as part of their network architectures.

Do we have low-dimensional data and parameters, and a cheap simulator?

If both the parameters and the data (or suitable summary-statistics thereof) are low-dimensional, and a very large number of simulations can be generated, model-free algorithms such as classical ABC can be competitive. These have the benefit of adding little computational overhead. Conversely, for limited simulation budgets and/or higher dimensionalities, approaches that train a model of the likelihood, posterior, or likelihood ratio will generally be preferable.

Are simulations expensive? Can we simulate online?

For time-intensive and complex simulators, it can be beneficial to use *sequential* methods to increase sample efficiency: We found that sequential schemes generally outperformed non-sequential ones. While we focused on simple strategies which use the previous estimate of the posterior to propose new parameters, more sophisticated schemes (e.g., Gutmann and Corander, 2016; Lueckmann et al., 2019; Järvenpää et al., 2019) may increase sample efficiency if only few simulations can be obtained. For some applications, inference is performed on a fixed dataset, and one cannot resort to sequential algorithms.

Do we want to carry out inference once, or repeatedly?

To perform SBI *separately* for different data points (i.e. compute $p(\boldsymbol{\theta}|\mathbf{x}_1), p(\boldsymbol{\theta}|\mathbf{x}_2), \ldots$), methods that allow 'amortization' (NPE) are likely preferable. While NLE and NRE allow amortisation of the neural network, MCMC sampling is required, which takes additional time. Conversely, if we want to run SBI conditioned on many i.i.d. data (e.g. $p(\boldsymbol{\theta}|\mathbf{x}_1, \mathbf{x}_2, \ldots)$) methods based on likelihood or ratio estimation (NLE, NRE), or NPE with exchangeable neural networks (Chan et al., 2018) would be appropriate.

Box 1: **Practitioners' advice for applying SBI algorithms.** Based on our current results and understanding, we provide advice to practitioners seeking to apply SBI. There is no one-fits-all solution—which algorithm to use in practice will depend on the problem at hand. For additional advice, see Cranmer et al. (2020).

for (S)NLE and (S)NPE, we used NSFs (Durkan et al., 2020) which were developed after these algorithms were published. This revealed that higher capacity density estimators were beneficial for posterior but not likelihood estimation (detailed analysis in Appendix H).

These examples show how the benchmark makes it possible to diagnose problems and improve algorithms.

4 Limitations

Our benchmark, in its current form, has several limitations. First, the algorithms considered here do not cover the entire spectrum of SBI algorithms: We did not include sequential algorithms using active learning or Bayesian Optimization (Gutmann and Corander, 2016; Järvenpää et al., 2019; Lueckmann et al., 2019; Aushev et al., 2020), or 'gray-box' algorithms, which use additional information about or from the simulator (e.g., Baydin et al., 2019; Brehmer et al., 2020). We focused on approaches using neural networks for density estimation and did not compare to alternatives using Gaussian Processes (e.g., Meeds and Welling, 2014; Wilkinson, 2014). There are many other algorithms which the benchmark is currently lacking (e.g., Nott et al., 2014; Ong et al., 2018; Clarté et al., 2020; Prangle, 2019; Priddle et al., 2019; Picchini et al., 2020; Radev et al., 2020; Rodrigues et al., 2020). Keeping our initial selection small allowed us to carefully investigate hyperparameter choices. We focused on sequential algorithms with less sophisticated acquisition schemes and the black-box scenario, since we think these are important baselines for future comparisons.

Second, the tasks we considered do not cover the variety of possible challenges. Notably, while we have tasks with high dimensional data with and without structure, we have not included tasks with high-dimensional spatial structure, e.g., images. Such tasks would require algorithms that automatically learn summary statistics while exploring the structure of the data (e.g., Dinev and Gutmann, 2018; Greenberg et al., 2019; Hermans et al., 2020; Chen et al., 2021), an active research area.

Third, while we extensively investigated tuning choices and compared implementations, the results might nevertheless reflect our own areas of expertise.

Fourth, in line with common practice in SBI, results presented in the paper focused on performance as a function of the number of simulation calls. It is important to remember that differences in computation time can be substantial (see Appendix R): For example, (S)ABC was much faster than approaches requiring network training. Overall, sequential neural algorithms exhibited longest runtimes.

Fifth, for reasons described above, we focused on problems for which reference posteriors can be computed. This raises the question of how insights on these problems will generalize to 'real-world' simulators. Notably, even these simple problems already identify clear differences between, and limitations of, different SBI approaches. Since it is not possible to rigorously compare the performance of different algorithms directly on 'real-world' simulators due to the lack of appropriate metrics, we see the benchmark as a necessary stepping stone towards the development of (potentially automated) selection strategies for practical problems.

Sixth, in practice, the choice of algorithm can depend on aspects that are difficult to quantify: It will depend on the available information about a problem, the inference goal, and the speed of the simulator, among other considerations. We included some practical considerations and recommendations in Box 1.

Finally, benchmarking is an important tool, but not an end in itself—for example, conceptually new ideas might initially not yield competitive results but only reveal their true value later. Conversely, 'overfitting' on benchmarks can lead to the illusion of progress, and result in an undue focus on small implementation details which might not generalize beyond it. It would certainly be possible to cheat on this benchmark: In particular, as the simulators are available, one could use samples (or even likelihoods) to excessively tune hyperparameters *for each task*. This would hardly transfer to practice where such tuning is usually impossible (lack of metrics and expensive simulators). Therefore, we carefully compared choices and selected hyperparameters performing best *across tasks* (Appendix H).

5 Discussion

Quantitatively evaluating, comparing and improving algorithms through benchmarking is at the core of progress in machine learning. We here provided an initial benchmark for simulation-based inference. If used sensibly, it will be an important tool for clarifying and expediting progress in SBI. We hope that the current results on multiple widely-used algorithms already provide insights into the state of the field, assist researchers with algorithm development, and that our recommendations for practitioners will help them in selecting appropriate algorithms.

We believe that the full potential of the benchmark will be revealed as more researchers participate and contribute. To facilitate this process, and allow users to quickly explore and compare algorithms, we are providing precomputed reference posteriors, a website (`sbi-benchmark.github.io`), and open-source code (`github.com/sbi-benchmark/sbibm`).

Acknowledgements

We thank Álvaro Tejero-Cantero, Auguste Schulz, Conor Durkan, François Lanusse, Leandra White, Marcel Nonnenmacher, Michael Deistler, Pedro Rodrigues, Poornima Ramesh, Sören Becker and Theofanis Karaletsos for discussions and comments on the manuscript. In addition, J.-M.L. would like to thank the organisers and participants of the Likelihood-Free Inference Workshop hosted by the Simons Foundation for discussions, in particular, Danley Hsu, François Lanusse, George Papamakarios, Henri Pesonen, Joeri Hermans, Johann Brehmer, Kyle Cranmer, Owen Thomas and Umberto Simola. We also acknowledge and thank the Python (Van Rossum and Drake Jr, 1995) and Julia (Bezanson et al., 2017) communities for developing the tools enabling this work, including `Altair` (VanderPlas et al., 2018), `DifferentialEquations.jl` (Rackauckas and Nie, 2017), `Hydra` (Yadan, 2019), `kernel-gof` (Jitkrittum et al., 2017), `igms` (Sutherland, 2017), `NumPy` (Harris et al., 2020), `pandas` (pandas development team, 2020), `pyABC` (Klinger et al., 2018), `pyabcranger` (Collin et al., 2020), `Pyro` (Bingham et al., 2019), `PyTorch` (Paszke et al., 2019), `sbi` (Tejero-Cantero et al., 2020), `Scikit-learn` (Pedregosa et al., 2011), `torch-two-sample` (Djolonga, 2017), and `vega-lite` (Satyanarayan et al., 2017).

This work was supported by the German Research Foundation (DFG; SFB 1233 PN 276693517, SFB 1089, SPP 2041, Germany's Excellence Strategy – EXC number 2064/1 PN 390727645) and the German Federal Ministry of Education and Research (BMBF; project 'ADIMEM', FKZ 01IS18052 A-D).

References

Alsing, J., B. Wandelt, and S. Feeney
2018. Massive optimal data compression and density estimation for scalable, likelihood-free inference in cosmology. *Monthly Notices of the Royal Astronomical Society*, 477(3):2874–2885.

Aushev, A., H. Pesonen, M. Heinonen, J. Corander, and S. Kaski
2020. Likelihood-free inference with deep gaussian processes. *Deep Learning and Inverse Problems Workshop at Neural Information Processing Systems.*

Baydin, A. G., L. Shao, W. Bhimji, L. Heinrich, L. Meadows, J. Liu, A. Munk, S. Naderiparizi, B. Gram-Hansen, G. Louppe, et al.
2019. Etalumis: bringing probabilistic programming to scientific simulators at scale. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, Pp. 1–24.

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

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

Bellemare, M. G., Y. Naddaf, J. Veness, and M. Bowling
2013. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279.

Bezanson, J., A. Edelman, S. Karpinski, and V. B. Shah
2017. Julia: A fresh approach to numerical computing. *SIAM review*, 59(1):65–98.

Bingham, E., J. P. Chen, M. Jankowiak, F. Obermeyer, N. Pradhan, T. Karaletsos, R. Singh, P. Szerlip, P. Horsfall, and N. D. Goodman
2019. Pyro: Deep universal probabilistic programming. *Journal of Machine Learning Research*, 20(1):973–978.

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

Brehmer, J., K. Cranmer, G. Louppe, and J. Pavez
2018. Constraining effective field theories with machine learning. *Physical Review Letters*, 121(11):111801.

Brehmer, J., G. Louppe, J. Pavez, and K. Cranmer
2020. Mining gold from implicit models to improve likelihood-free inference. *Proceedings of the National Academy of Sciences*, 117(10):5242–5249.

Chan, J., V. Perrone, J. Spence, P. Jenkins, S. Mathieson, and Y. Song
2018. A likelihood-free inference framework for population genetic data using exchangeable neural networks. In *Advances in Neural Information Processing Systems 31*, Pp. 8594–8605. Curran Associates, Inc.

Charnock, T., G. Lavaux, and B. D. Wandelt
2018. Automatic physical inference with information maximizing neural networks. *Physical Review D*, 97(8):083004.

Chen, Y., D. Zhang, M. Gutmann, A. Courville, and Z. Zhu
2021. Neural approximate sufficient statistics for implicit models. In *Proceedings of the 9th International Conference on Learning Representations, ICLR.*

Chwialkowski, K., H. Strathmann, and A. Gretton
2016. A kernel test of goodness of fit. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, Pp. 2606–2615. PMLR.

Clarté, G., C. P. Robert, R. J. Ryder, and J. Stoehr
2020. Component-wise approximate bayesian computation via gibbs-like steps. *Biometrika*.

Collin, F.-D., A. Estoup, J.-M. Marin, and L. Raynal
2020. Bringing abc inference to the machine learning realm: Abcranger, an optimized random forests library for abc. In *JOBIM 2020*, volume 2020.

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

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

Dalmasso, N., A. B. Lee, R. Izbicki, T. Pospisil, and C.-A. Lin
2020. Validation of approximate likelihood and emulator models for computationally intensive simulations. In *Proceedings of The 23rd International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Dinev, T. and M. U. Gutmann
2018. Dynamic likelihood-free inference via ratio estimation (dire). *arXiv preprint arXiv:1810.09899*.

Djolonga, J.
2017. torch-two-sample: A pytorch library for differentiable two-sample tests. Github.

Drovandi, C. C., C. Grazian, K. Mengersen, and C. Robert
2018. Approximating the likelihood in approximate bayesian computation. In *Handbook of Approximate Bayesian Computation*, S. Sisson, Y. Fan, and M. Beaumont, eds., chapter 12. CRC Press, Taylor & Francis Group.

Duan, Y., X. Chen, R. Houthooft, J. Schulman, and P. Abbeel
2016. Benchmarking deep reinforcement learning for continuous control. In *Proceedings of the 33th International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, Pp. 1329–1338. PMLR.

Durkan, C., A. Bekasov, I. Murray, and G. Papamakarios
2019. Neural spline flows. In *Advances in Neural Information Processing Systems*, Pp. 7509–7520. Curran Associates, Inc.

Durkan, C., I. Murray, and G. Papamakarios
2020. On contrastive learning for likelihood-free inference. In *Proceedings of the 36th International Conference on Machine Learning*, volume 98 of *Proceedings of Machine Learning Research*. PMLR.

Durkan, C., G. Papamakarios, and I. Murray
2018. Sequential neural methods for likelihood-free inference. *Bayesian Deep Learning Workshop at Neural Information Processing Systems*.

Dutta, R., J. Corander, S. Kaski, and M. U. Gutmann
2016. Likelihood-free inference by ratio estimation. *arXiv preprint arXiv:1611.10242*.

Filos, A., S. Farquhar, A. N. Gomez, T. G. Rudner, Z. Kenton, L. Smith, M. Alizadeh, A. de Kroon, and Y. Gal
2019. A systematic comparison of bayesian deep learning robustness in diabetic retinopathy tasks. *Bayesian Deep Learning Workshop at Neural Information Processing Systems*.

Friedman, J.
2004. On multivariate goodness-of-fit and two-sample testing. In *Conference on Statistical Problems in Particle Physics, Astrophysics and Cosmology*.

Gonçalves, P. J., J.-M. Lueckmann, M. Deistler, M. Nonnenmacher, K. Öcal, G. Bassetto, C. Chintaluri, W. F. Podlaski, S. A. Haddad, T. P. Vogels, D. S. Greenberg, and J. H. Macke
2020. Training deep neural density estimators to identify mechanistic models of neural dynamics. *eLife*.

Gourieroux, C., A. Monfort, and E. Renault
1993. Indirect inference. *Journal of Applied Econometrics*, 8(S1):S85–S118.

Greenberg, D., M. Nonnenmacher, and J. Macke
2019. Automatic posterior transformation for likelihood-free inference. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, Pp. 2404–2414. PMLR.

Gretton, A., K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola
2012. A kernel two-sample test. *The Journal of Machine Learning Research*, 13(Mar):723–773.

Gutmann, M. U. and J. Corander
2016. Bayesian optimization for likelihood-free inference of simulator-based statistical models. *The Journal of Machine Learning Research*, 17(1):4256–4302.

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

Harris, C. R., K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, et al.
2020. Array programming with numpy. *Nature*, 585(7825):357–362.

Hermans, J., V. Begy, and G. Louppe
2020. Likelihood-free mcmc with approximate likelihood ratios. In *Proceedings of the 37th International*

Conference on Machine Learning, volume 98 of *Proceedings of Machine Learning Research*. PMLR.

Hirsch, H.-G. and D. Pearce
2000. The aurora experimental framework for the performance evaluation of speech recognition systems under noisy conditions. In *ASR2000-Automatic Speech Recognition: Challenges for the new Millenium ISCA Tutorial and Research Workshop (ITRW)*.

Hogg, D. W. and D. Foreman-Mackey
2018. Data analysis recipes: Using markov chain monte carlo. *The Astrophysical Journal Supplement Series*, 236(1):11.

Izbicki, R., A. Lee, and C. Schafer
2014. High-dimensional density ratio estimation with extensions to approximate likelihood computation. In *Artificial Intelligence and Statistics*, Pp. 420–429.

Järvenpää, M., M. U. Gutmann, A. Pleska, A. Vehtari, P. Marttinen, et al.
2019. Efficient acquisition rules for model-based approximate bayesian computation. *Bayesian Analysis*, 14(2):595–622.

Järvenpää, M., M. U. Gutmann, A. Vehtari, P. Marttinen, et al.
2020. Parallel gaussian process surrogate bayesian inference with noisy likelihood evaluations. *Bayesian Analysis*.

Jitkrittum, W., W. Xu, Z. Szabó, K. Fukumizu, and A. Gretton
2017. A linear-time kernel goodness-of-fit test. In *Advances in Neural Information Processing Systems*, Pp. 262–271.

Karabatsos, G. and F. Leisen
2018. An approximate likelihood perspective on abc methods. *Statistics Surveys*, 12:66–104.

Klinger, E., D. Rickert, and J. Hasenauer
2018. pyabc: distributed, likelihood-free inference. *Bioinformatics*, 34(20):3591–3593.

Le, T. A., A. G. Baydin, and F. Wood
2017. Inference compilation and universal probabilistic programming. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 54. JMLR.

Liu, F., W. Xu, J. Lu, G. Zhang, A. Gretton, and D. J. Sutherland
2020. Learning deep kernels for non-parametric two-sample tests. In *Proceedings of the 37th International Conference on Machine Learning*, volume 98 of *Proceedings of Machine Learning Research*. PMLR.

Liu, Q., J. Lee, and M. Jordan
2016. A kernelized stein discrepancy for goodness-of-fit tests. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, Pp. 276–284. PMLR.

Lopez-Paz, D. and M. Oquab
2017. Revisiting classifier two-sample tests. In *5th International Conference on Learning Representations, ICLR*.

Lueckmann, J.-M., G. Bassetto, T. Karaletsos, and J. H. Macke
2019. Likelihood-free inference with emulator networks. In *Proceedings of The 1st Symposium on Advances in Approximate Bayesian Inference*, volume 96 of *Proceedings of Machine Learning Research*, Pp. 32–53. PMLR.

Lueckmann, J.-M., P. J. Goncalves, G. Bassetto, K. Öcal, M. Nonnenmacher, and J. H. Macke
2017. Flexible statistical inference for mechanistic models of neural dynamics. In *Advances in Neural Information Processing Systems 30*, Pp. 1289–1299. Curran Associates, Inc.

Marjoram, P. and S. Tavaré
2006. Modern computational approaches for analysing molecular genetic variation data. *Nature Reviews Genetics*, 7(10):759–770.

Meeds, E. and M. Welling
2014. Gps-abc: Gaussian process surrogate approximate bayesian computation. In *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence*, UAI'14, P. 593–602, Arlington, Virginia, USA. AUAI Press.

Nott, D. J., Y. Fan, L. Marshall, and S. Sisson
2014. Approximate bayesian computation and bayes' linear analysis: toward high-dimensional abc. *Journal of Computational and Graphical Statistics*, 23(1):65–86.

Ong, V. M.-H., D. J. Nott, M.-N. Tran, S. A. Sisson, and C. C. Drovandi
2018. Likelihood-free inference in high dimensions with synthetic likelihood. *Computational Statistics & Data Analysis*, 128:271 – 291.

pandas development team, T.
2020. pandas-dev/pandas: Pandas.

Papamakarios, G. and I. Murray
2016. Fast ϵ-free inference of simulation models with bayesian conditional density estimation. In *Advances in Neural Information Processing Systems 29*, Pp. 1028–1036. Curran Associates, Inc.

Papamakarios, G., E. Nalisnick, D. J. Rezende, S. Mohamed, and B. Lakshminarayanan
2019a. Normalizing flows for probabilistic modeling and inference. *arXiv preprint arXiv:1912.02762*.

Papamakarios, G., T. Pavlakou, and I. Murray
2017. Masked autoregressive flow for density estimation. In *Advances in Neural Information Processing Systems 30*, Pp. 2338–2347. Curran Associates, Inc.

Papamakarios, G., D. Sterratt, and I. Murray
2019b. Sequential neural likelihood: Fast likelihood-free inference with autoregressive flows. In *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 89 of *Proceedings of Machine Learning Research*, Pp. 837–848. PMLR.

Paszke, A., S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala
2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, Pp. 8024–8035. Curran Associates, Inc.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay
2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Pham, K. C., D. J. Nott, and S. Chaudhuri
2014. A note on approximating abc-mcmc using flexible classifiers. *Stat*, 3(1):218–227.

Picchini, U., U. Simola, and J. Corander
2020. Adaptive mcmc for synthetic likelihoods and correlated synthetic likelihoods. *arXiv preprint arXiv:2004.04558*.

Prangle, D.
2019. Distilling importance sampling. *arXiv preprint arXiv:1910.03632*.

Prangle, D., P. Fearnhead, M. P. Cox, P. J. Biggs, and N. P. French
2014. Semi-automatic selection of summary statistics for abc model choice. *Statistical applications in genetics and molecular biology*, 13(1):67–82.

Priddle, J. W., S. A. Sisson, and C. Drovandi
2019. Efficient bayesian synthetic likelihood with whitening transformations. *arXiv preprint arXiv:1909.04857*.

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

Rackauckas, C. and Q. Nie
2017. Differentialequations.jl – a performant and feature-rich ecosystem for solving differential equations in julia. *The Journal of Open Research Software*, 5(1).

Radev, S. T., U. K. Mertens, A. Voss, L. Ardizzone, and U. Köthe
2020. Bayesflow: Learning complex stochastic models with invertible neural networks. *IEEE Transactions on Neural Networks and Learning Systems*.

Ramdas, A., S. J. Reddi, B. Poczos, A. Singh, and L. Wasserman
2015. On the decreasing power of kernel and distance based nonparametric hypothesis tests in high dimensions. *AAAI Conference on Artificial Intelligence*.

Ratmann, O., O. Jørgensen, T. Hinkley, M. Stumpf, S. Richardson, and C. Wiuf
2007. Using likelihood-free inference to compare evolutionary dynamics of the protein networks of h. pylori and p. falciparum. *PLoS Computational Biology*, 3(11).

Raynal, L., J.-M. Marin, P. Pudlo, M. Ribatet, C. P. Robert, and A. Estoup
2019. Abc random forests for bayesian parameter inference. *Bioinformatics*, 35(10):1720–1728.

Rezende, D. and S. Mohamed
2015. Variational inference with normalizing flows. In *Proceedings of The 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, Pp. 1530–1538. PMLR.

Rios, L. M. and N. V. Sahinidis
2013. Derivative-free optimization: a review of algorithms and comparison of software implementations. *Journal of Global Optimization*, 56(3):1247–1293.

Rodrigues, G., D. J. Nott, and S. Sisson
2020. Likelihood-free approximate gibbs sampling. *Statistics and Computing*, Pp. 1–17.

Russakovsky, O., J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al.
2015. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252.

Satyanarayan, A., D. Moritz, K. Wongsuphasawat, and J. Heer
2017. Vega-lite: A grammar of interactive graphics. *IEEE Trans. Visualization & Comp. Graphics (Proc. InfoVis)*.

Shahriari, B., K. Swersky, Z. Wang, R. P. Adams, and N. De Freitas
2015. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175.

Simola, U., J. Cisewski-Kehe, M. U. Gutmann, J. Corander, et al.
2020. Adaptive approximate bayesian computation tolerance selection. *Bayesian Analysis*.

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

Sisson, S. A., F. Y., and B. M. A.
2018. Overview of abc. In *Handbook of Approximate Bayesian Computation*, chapter 1. CRC Press, Taylor & Francis Group.

Sutherland, D. J.
2017. igms: Implicit generative models. Github.

Sutherland, D. J., H.-Y. Tung, H. Strathmann, S. De, A. Ramdas, A. Smola, and A. Gretton
2017. Generative models and model criticism via optimized maximum mean discrepancy. *5th International Conference on Learning Representations, ICLR*.

Talts, S., M. Betancourt, D. Simpson, A. Vehtari, and A. Gelman
2018. Validating bayesian inference algorithms with simulation-based calibration. *arXiv preprint arXiv:1804.06788*.

Tavaré, S., D. J. Balding, R. C. Griffiths, and P. Donnelly
1997. Inferring coalescence times from dna sequence data. *Genetics*, 145(2).

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

Thomas, O., R. Dutta, J. Corander, S. Kaski, and M. U. Gutmann
2020. Likelihood-free inference by ratio estimation. *Bayesian Analysis*.

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

Van Rossum, G. and F. L. Drake Jr
1995. *Python tutorial*. Centrum voor Wiskunde en Informatica Amsterdam, The Netherlands.

VanderPlas, J., B. E. Granger, J. Heer, D. Moritz, K. Wongsuphasawat, A. Satyanarayan, E. Lees, I. Timofeev, B. Welsh, and S. Sievert
2018. Altair: Interactive statistical visualizations for python. *The Journal of Open Source Software*, 3(32).

Wang, A., A. Singh, J. Michael, F. Hill, O. Levy, and S. Bowman
2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, Pp. 353–355. Association for Computational Linguistics.

Wenzel, F., K. Roth, B. S. Veeling, J. Świątkowski, L. Tran, S. Mandt, J. Snoek, T. Salimans, R. Jenatton, and S. Nowozin
2020. How good is the bayes posterior in deep neural networks really? In *Proceedings of the 37th International Conference on Machine Learning*, volume 98 of *Proceedings of Machine Learning Research*. PMLR.

Wilkinson, R. D.
2014. Accelerating abc methods using gaussian processes. In *Proceedings of the 17th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 33 of *Proceedings of Machine Learning Research*, Pp. 1015–1023. PMLR.

Wood, F., A. Warrington, S. Naderiparizi, C. Weilbach, V. Masrani, W. Harvey, A. Scibior, B. Beronov, and A. Nasseri
2020. Planning as inference in epidemiological models. *arXiv preprint arXiv:2003.13221*.

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

Yadan, O.
2019. Hydra - a framework for elegantly configuring complex applications. Github.

Appendices

A Algorithms

A.1 Rejection Approximate Bayesian Computation (REJ-ABC)

Algorithm 1: Rejection ABC

```
while in simulation budget do
    Sample $\boldsymbol{\theta}'$ from $p(\boldsymbol{\theta})$
    Simulate data $\mathbf{x}'$ from $p(\mathbf{x}|\boldsymbol{\theta}')$
    if $d(\mathbf{x}', \mathbf{x}_o) \leq \epsilon$ then
        Accept $\boldsymbol{\theta}'$
    else
        Reject $\boldsymbol{\theta}'$
    end
end
return Accepted samples $\{\boldsymbol{\theta}'\}$ from $\hat{p}(\boldsymbol{\theta}|d(\mathbf{x}, \mathbf{x}_o) \leq \epsilon)$
```

Classical Approximate Bayesian Computation (ABC) is based on Monte Carlo rejection sampling (Tavaré et al., 1997; Pritchard et al., 1999): In rejection ABC, the evaluation of the likelihood is replaced by a comparison between observed data $\mathbf{x}_o$ and simulated data $\mathbf{x}$, based on a distance measure $d(\mathbf{x}, \mathbf{x}_o)$. Samples $\boldsymbol{\theta}$ from the approximate posterior are obtained by collecting simulation parameters that result in simulated data that is close to the observed data.

More formally, given observed data $\mathbf{x}_o$, a prior $p(\boldsymbol{\theta})$ over parameters of simulation-based model $p(\mathbf{x}|\boldsymbol{\theta})$, a distance measure $d(\mathbf{x}, \mathbf{x}_o)$ and an acceptance threshold ϵ, rejection ABC obtains parameter samples $\boldsymbol{\theta}$ from the approximate posterior as outlined in Algorithm 1.

In theory, rejection ABC obtains samples from the true posterior $p(\boldsymbol{\theta}|\mathbf{x}_o)$ in the limit $\epsilon \to 0$ and $N \to \infty$, where N is the simulation budget. In practice, its accuracy depends on the trade-off between simulation budget and the rejection criterion ϵ. Rejection ABC suffers from the curse of dimensionality, i.e., with linear increase in the dimensionality of $\mathbf{x}$, an exponential increase in simulation budget is required to maintain accurate results.

For the benchmark, we did not use a fixed ϵ-threshold, but quantile-based rejection. Depending on the simulation budget (1k, 10k, 100k), we used a quantile of (0.1, 0.01, or, 0.001), so that REJ-ABC returned 100 samples with smallest distance to $\mathbf{x}_o$ in each of these cases (see Appendix H for different hyperparameter choices). In order to compute metrics on 10k samples, we sampled from a KDE fitted on the accepted parameters (details about KDE resampling in Appendix H). REJ-ABC requires the choice of the distance measure $d(\mathbf{x}, \mathbf{x}_o)$: here we used the l_2-norm.

A.2 Sequential Monte Carlo Approximate Bayesian Computation (SMC-ABC)

Algorithm 2: Population Monte Carlo ABC (ABC-PMC) as in Beaumont et al. (2009)

```
Set schedule $\boldsymbol{\epsilon}$ (including initial $\epsilon_0$), population indicator $t = 0$, and population size $N$
Initialize weights $W_0 = 1/N$ uniformly
Sample initial population $\{\boldsymbol{\theta}_0^{(i)}\}$ using rejection sampling with $\epsilon_0$
while in simulation budget do
    Increase population indicator $t = t + 1$
    Set particle indicator $i = 0$
    while $i < N$ do
        Sample $\boldsymbol{\theta}'$ from previous population $\{\boldsymbol{\theta}_{t-1}^{(i)}\}$ with weights $\{W_{t-1}^{(i)}\}$;
        Perturb $\boldsymbol{\theta}'$: $\boldsymbol{\theta}'' \sim K_t(\boldsymbol{\theta}|\boldsymbol{\theta}')$
        Simulate data $x''$ from $p(\mathbf{x}|\boldsymbol{\theta}'')$
        if $d(\mathbf{x}'', \mathbf{x}_o) \leq \epsilon_t$ then
            Set $\boldsymbol{\theta}_t^{(i)} = \boldsymbol{\theta}''$ and $W_t^i = \frac{p(\boldsymbol{\theta}_t^{(i)})}{\sum_{j=1}^N W_{t-1}^j K_t(\boldsymbol{\theta}_t^{(i)}|\boldsymbol{\theta}_{t-1}^j)}$
            Increase particle indicator $i = i + 1$
        else
            reject $\boldsymbol{\theta}''$
        end
    end
    Normalize weights so that $\sum_i W_t^{(i)} = 1$
end
return Weighted samples $\{\boldsymbol{\theta}_t^{(i)}\}$ from $\hat{p}(\boldsymbol{\theta}|d(\mathbf{x}, \mathbf{x}_o) \leq \epsilon)$
```

Sequential Monte Carlo Approximate Bayesian Computation (SMC-ABC) algorithms (Beaumont et al., 2002; Marjoram and Tavaré, 2006; Sisson et al., 2007; Toni et al., 2009) are an extension of the classical rejection ABC approach, inspired by importance sampling and sequential Monte Carlo sampling. Central to SMC-ABC is the idea to approach the final set of samples from the approximate posterior by constructing a series of intermediate sets of samples slowly approaching the final set through perturbations.

Several variants have been developed (e.g., Sisson et al., 2007; Beaumont et al., 2009; Toni et al., 2009; Simola et al., 2020). Here, we used the scheme ABC-PMC scheme of Beaumont et al. (2009) and refer to it as SMC-ABC in the manuscript. More formally, the description of the ABC-PMC algorithm is as follows: Given observed data $\mathbf{x}_o$, a prior $p(\boldsymbol{\theta})$ over parameters of a simulation-based model $p(\mathbf{x}|\boldsymbol{\theta})$, a distance measure $d(\mathbf{x}, \mathbf{x}_o)$, a schedule of acceptance thresholds ϵ_i, and a kernel $K(\boldsymbol{\theta}|\boldsymbol{\theta}')$ to perturb intermediate samples, weighted samples of the approximate posterior are obtained as described in Algorithm 2.

SMC-ABC can improve the sampling efficiency compared to REJ-ABC and avoids severe inefficiencies due to a mismatch between initial sampling and the target distribution. However, it comes with more hyperparameters that can require careful tuning to the problem at hand, e.g., the choice of distance measure, kernel, and ϵ-schedule. Like, REJ-ABC, SMC-ABC suffers from the curse of dimensionality.

For the benchmark, we considered the popular toolbox `pyABC` (Klinger et al., 2018). Additionally, to fully understand the details of the SMC-ABC approach, we also implemented our own version. In the main paper we report results obtained with our implementation because it yielded slightly better results. A careful comparison of the two approaches, and the optimization of hyperparameters like ϵ-schedule, population size and perturbation kernel variance across different tasks are shown in Appendix H. After optimization, the crucial parameters of SMC-ABC were set to: l_2-norm as distance metric, quantile-based epsilon decay with 0.2 quantile, population size 100 for simulation budgets 1k and 10k, population size 1000 for simulation budget 100k, Gaussian perturbation kernel with empirical covariance from previous population scaled by 0.5. We obtained 10k samples required for calculation of metrics as follows: If a population is not complete within the simulation budget we completed it with accepted particles from the last population and recalculated all weights. We then fitted a KDE on all those particles and sampled 10k samples from the KDE.

A.3 Neural Likelihood Estimation (NLE)

Algorithm 3: Single round Neural Likelihood as in Papamakarios et al. (2019b)

```
Set $\mathcal{D} = \{\}$
**for** $n = 1 : N$ **do**
    Sample $\boldsymbol{\theta}_n \sim p(\boldsymbol{\theta})$
    Simulate $\mathbf{x}_n \sim p(\mathbf{x}|\boldsymbol{\theta}_n)$
    Add $(\boldsymbol{\theta}_n, \mathbf{x}_n)$ to $\mathcal{D}$
**end**
Train $q_{\boldsymbol{\psi}}(\mathbf{x}|\boldsymbol{\theta})$ on $\mathcal{D}$
**return** Samples from $\hat{p}(\boldsymbol{\theta}|\mathbf{x}_o) \propto q_{\boldsymbol{\psi}}(\mathbf{x}_o|\boldsymbol{\theta})p(\boldsymbol{\theta})$ via MCMC; $q_{\boldsymbol{\psi}}(\mathbf{x}|\boldsymbol{\theta})$
```

Likelihood estimation approaches to SBI use density estimation to approximate the likelihood $p(\mathbf{x}_o|\boldsymbol{\theta})$. After learning a surrogate $q_{\boldsymbol{\psi}}$ ($\boldsymbol{\psi}$ denoting the parameters of the estimator) for the likelihood function, one can for example use Markov Chain Monte Carlo (MCMC) based sampling algorithms to obtain samples from the approximate posterior $\hat{p}(\boldsymbol{\theta}|\mathbf{x}_o)$. This idea dates back to using Gaussian approximations of the likelihood (Wood, 2010; Drovandi et al., 2018), and more recently, was extended to density estimation with neural networks (Papamakarios et al., 2019b; Lueckmann et al., 2019).

We refer to the single-round version of the (sequential) neural likelihood approach by Papamakarios et al. (2019b) as NLE, and outline it in Algorithm 3: Given a set of samples $\{\boldsymbol{\theta}_n, \mathbf{x}_n\}_{1:N}$ obtained by sampling $\boldsymbol{\theta}_n \sim p(\boldsymbol{\theta})$ from the prior and simulating $\mathbf{x}_n \sim p(\mathbf{x}|\boldsymbol{\theta}_n)$, we train a conditional neural density estimator $q_{\boldsymbol{\psi}}(\mathbf{x}|\boldsymbol{\theta})$ modelling the conditional of data given parameters on the set $\{\boldsymbol{\theta}_n, \mathbf{x}_n\}_{1:N}$. Training proceeds by maximizing the log likelihood $\sum_n \log q_{\boldsymbol{\psi}}(\mathbf{x}|\boldsymbol{\theta})$. Given enough simulations, a sufficiently flexible conditional neural density estimator approximates the likelihood in the support of the prior $p(\boldsymbol{\theta})$ (Papamakarios et al., 2019b). Once $q_{\boldsymbol{\psi}}$ is trained, samples from the approximate posterior $\hat{p}(\boldsymbol{\theta}|\mathbf{x}_o)$ are obtained using MCMC sampling based on the approximate likelihood $\hat{p}(\mathbf{x}_o|\boldsymbol{\theta})$ and the prior $p(\boldsymbol{\theta})$.

For MCMC sampling, Papamakarios et al. (2019b) suggest to use Slice Sampling (Neal, 2003) with a single chain. However, we observed that the accuracy of the obtained posterior samples can be substantially improved by changing the Slice Sampling scheme as follows: 1) Instead of a single chain, we used 100 parallel MCMC chains; 2) for initialization of the chains, we sampled 10k candidate parameters from the prior, evaluated them under the unnormalized approximate posterior, and used these values as weights to resample initial locations; 3) we transformed parameters to be unbounded as suggested e.g. in Bingham et al. (2019); Carpenter et al. (2017); Hogg and Foreman-Mackey (2018). In addition, we reimplemented the slice sampler to allow vectorized evaluations of the likelihood, which yielded significant computational speed-ups.

For the benchmark, we used as density estimator a Masked Autoregressive Flow (MAF, Papamakarios et al., 2017) with five flow transforms, each with two blocks and 50 hidden units, tanh non-linearity and batch normalization after each layer. For the MCMC step, we used the scheme as outlined above with 250 warm-up steps and ten-fold thinning, to obtain 10k samples from the approximate posterior (1k samples from each chain). In Appendix H we show results for all tasks obtained with a Neural Spline Flow (NSF, Durkan et al., 2019) for density estimation, using five flow transforms, two residual blocks of 50 hidden units each, ReLU non-linearity, and 10 bins.

A.4 Sequential Neural Likelihood Estimation (SNLE)

Algorithm 4: Sequential Neural Likelihood as in Papamakarios et al. (2019b)

```
Set $\hat{p}_0(\boldsymbol{\theta}|\mathbf{x}_o) = p(\boldsymbol{\theta})$ and $\mathcal{D} = \{\}$
for $r = 1 : R$ do
    for $n = 1 : N$ do
        Sample $\boldsymbol{\theta}_n \sim \hat{p}_{r-1}(\boldsymbol{\theta}|\mathbf{x}_o)$ with MCMC
        Simulate $\mathbf{x}_n \sim p(\mathbf{x}|\boldsymbol{\theta}_n)$
        Add $(\boldsymbol{\theta}_n, \mathbf{x}_n)$ to $\mathcal{D}$
    end
    (Re-)train $q_{\boldsymbol{\psi}}(\mathbf{x}|\boldsymbol{\theta})$ on $\mathcal{D}$
    Set $\hat{p}_r(\boldsymbol{\theta}|\mathbf{x}_o) \propto q_{\boldsymbol{\psi}}(\mathbf{x}_o|\boldsymbol{\theta})p(\boldsymbol{\theta})$
end
return Samples from $\hat{p}(\boldsymbol{\theta}|\mathbf{x}_o) \propto q_{\boldsymbol{\psi}}(\mathbf{x}_o|\boldsymbol{\theta})p(\boldsymbol{\theta})$ via MCMC; $q_{\boldsymbol{\psi}}(\mathbf{x}|\boldsymbol{\theta})$
```

Sequential Neural Likelihood estimation (SNLE or SNL, Papamakarios et al., 2019b) extends the neural likelihood estimation approach described in the previous section to be sequential.

The idea behind sequential SBI algorithms is based on the following intuition: If for a particular inference problem, there is only a single $\mathbf{x}_o$ one is interested in, then simulating data using parameters from the entire prior space might be inefficient, leading to a training set $\mathcal{D}$ that contains training data $(\boldsymbol{\theta}, \mathbf{x})$ which carries little information about the posterior $p(\boldsymbol{\theta}|\mathbf{x}_o)$. Instead, to increase sample efficiency, one may draw training data points from a proposal distribution $\tilde{p}(\boldsymbol{\theta})$, ideally obtaining $\boldsymbol{\theta}$ for which $\mathbf{x}$ is close to $\mathbf{x}_o$. One candidate that has been commonly used in the literature for such a proposal is the approximate posterior distribution itself.

SNLE is a multi-round version of NLE, where in each round new training samples are drawn from a proposal $\tilde{p}(\boldsymbol{\theta})$. The proposal is chosen to be the posterior estimate at $\mathbf{x}_o$ from the previous round $\hat{p}(\boldsymbol{\theta}|\mathbf{x}_o)$ and its samples are obtained using MCMC. The proposal controls where $q_{\boldsymbol{\psi}}(\mathbf{x}|\boldsymbol{\theta})$ is learned most accurately. Thus, by iterating over multiple rounds, a good approximation to the posterior can be learned more efficiently than by sampling all training data from the prior. SNLE is summarized in Algorithm 4.

For the benchmark, we used as density estimator a Masked Autoregressive Flow (Papamakarios et al., 2017), and MCMC to obtain posterior samples after every round, both with the same settings as described for NLE. The simulation budget was equally split across 10 rounds. In Appendix H, we show results for all tasks obtained with a Neural Spline Flow (NSF, Durkan et al., 2019) for density estimation, using five flow transforms, two residual blocks of 50 hidden units each, ReLU non-linearity, and 10 bins.

A.5 Neural Posterior Estimation (NPE)

Algorithm 5: Single round Neural Posterior Estimation as in Papamakarios and Murray (2016)

```
for $j = 1 : N$ do
    Sample $\boldsymbol{\theta}_j \sim p(\boldsymbol{\theta})$
    Simulate $\mathbf{x}_j \sim p(\mathbf{x}|\boldsymbol{\theta}_j)$
end
$\boldsymbol{\phi} \leftarrow \arg\min \sum_j^N -\log q_{F(\mathbf{x}_j,\boldsymbol{\phi})}(\boldsymbol{\theta}_j)$
Set $\hat{p}(\boldsymbol{\theta}|\mathbf{x}_o) = q_{F(\mathbf{x}_o,\boldsymbol{\phi})}(\boldsymbol{\theta})$
return Samples from $\hat{p}(\boldsymbol{\theta}|\mathbf{x}_o)$; $q_{F(\mathbf{x},\boldsymbol{\phi})}(\boldsymbol{\theta})$
```

NPE uses conditional density estimation to directly estimate the posterior. This idea dates back to regression adjustment approaches (Blum and François, 2010) and was extended to density estimators using neural networks (Papamakarios and Murray, 2016) more recently.

As outlined in Algorithm 5, the approach is as follows: Given a prior over parameters $p(\boldsymbol{\theta})$ and a simulator, a set of training data points $(\boldsymbol{\theta}, \mathbf{x})$ is generated. This training data is used to learn the parameters $\boldsymbol{\psi}$ of a conditional density estimator $q_{\boldsymbol{\psi}}(\boldsymbol{\theta}|x)$ using a neural network $F(\mathbf{x}, \boldsymbol{\phi})$, i.e., $\boldsymbol{\psi} = F(\mathbf{x}, \boldsymbol{\phi})$. The loss function is given by the negative log probability $-\log q_{\boldsymbol{\psi}}(\boldsymbol{\theta}|x)$. If the density estimator q is flexible enough and training data is infinite, this loss function leads to perfect recovery of the ground-truth posterior (Papamakarios and Murray, 2016).

For the benchmark, we used the approach by Papamakarios and Murray (2016) with a Neural Spline Flow (NSF, Durkan et al., 2019) as density estimator, using five flow transforms, two residual blocks of 50 hidden units each, ReLU non-linearity, and 10 bins. We sampled 10k samples from the approximate posterior $q_{F(\mathbf{x}_o,\boldsymbol{\phi})}(\boldsymbol{\theta})$. In Appendix H, we compare NSFs to Masked Autoregressive Flows (MAFs, Papamakarios et al., 2017), as used in Greenberg et al. (2019); Durkan et al. (2020), with five flow transforms, each with two blocks and 50 hidden units, tanh non-linearity and batch normalization after each layer.

A.6 Sequential Neural Posterior Estimation (SNPE)

Algorithm 6: Sequential Neural Posterior Estimation with atomic proposals (Greenberg et al., 2019)

```
Set $\tilde{p}_1(\boldsymbol{\theta}) = p(\boldsymbol{\theta})$
$c \leftarrow 0$
for $r = 1 : R$ do
    for $j = 1 : N$ do
        $c \leftarrow c + 1$
        Sample $\boldsymbol{\theta}_c \sim \tilde{p}_r(\boldsymbol{\theta})$
        Simulate $\mathbf{x}_c \sim p(\mathbf{x}|\boldsymbol{\theta}_c)$
    end
    $V_r(\Theta) := \begin{cases} \binom{c}{M}^{-1} & \text{if } \Theta = \{\boldsymbol{\theta}_{b_1}, \boldsymbol{\theta}_{b_1}, \dots, \boldsymbol{\theta}_{b_M}\} \text{ and } 1 \leq b_1 < b_2 < \dots < b_M \leq c \\ 0 & \text{otherwise} \end{cases}$
    $\boldsymbol{\phi} \leftarrow \arg\min_{\boldsymbol{\phi}} \mathbb{E}_{\Theta \sim V_r(\Theta)} \left[ \sum_{\boldsymbol{\theta}_j \in \Theta} -\log \tilde{q}_{\mathbf{x}_j, \boldsymbol{\phi}}(\boldsymbol{\theta}_j) \right]$
    Set $\tilde{p}_{r+1}(\boldsymbol{\theta}) := q_{F(\mathbf{x}_o, \boldsymbol{\phi})}(\boldsymbol{\theta})$
end
return Samples from $\hat{p}_R(\boldsymbol{\theta}|\mathbf{x}_o)$; $q_{F(x,\boldsymbol{\phi})}(\boldsymbol{\theta})$
```

Sequential Neural Posterior Estimation SNPE is the sequential analog of NPE, and meant to increase sample efficiency (see also subsection A.4). When the posterior is targeted directly, using a proposal distribution $\tilde{p}(\boldsymbol{\theta})$ different from the prior requires a correction step—without it, the posterior under the proposal distribution would be inferred (Papamakarios and Murray, 2016). This so-called proposal posterior is denoted by $\tilde{p}(\boldsymbol{\theta}|\mathbf{x})$:

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

where $\tilde{p}(\mathbf{x}) = \int_{\boldsymbol{\theta}} \tilde{p}(\boldsymbol{\theta}) p(\mathbf{x}|\boldsymbol{\theta})$. Note that for $\tilde{p}(\boldsymbol{\theta}) = p(\boldsymbol{\theta})$, it directly follows that $\tilde{p}(\boldsymbol{\theta}|\mathbf{x}) = p(\boldsymbol{\theta}|\mathbf{x})$.

There have been three different approaches to this correction step so far, leading to three versions of SNPE (Papamakarios and Murray, 2016; Lueckmann et al., 2017; Greenberg et al., 2019). All three algorithms have in common that they train a neural network $F(\mathbf{x}, \boldsymbol{\phi})$ to learn the parameters of a family of densities $q_{\boldsymbol{\psi}}$ to estimate the posterior. They differ in what is targeted by $q_{\boldsymbol{\psi}}$ and which loss is used for F.

SNPE-A (Papamakarios and Murray, 2016) trains F to target the proposal posterior $\tilde{p}(\boldsymbol{\theta}|\mathbf{x})$ by minimizing the log likelihood loss $-\sum_n \log q_{\boldsymbol{\psi}}(\boldsymbol{\theta}_n|\mathbf{x}_n)$, and then post-hoc solves for $p(\boldsymbol{\theta}|\mathbf{x})$. The analytical post-hoc step places restrictions on $q_{\boldsymbol{\psi}}$, the proposal, and prior. Papamakarios and Murray (2016) used Gaussian mixture density networks, single Gaussians proposals, and Gaussian or uniform priors. SNPE-B (Lueckmann et al., 2017) trains F with the importance weighted loss $-\sum_n \frac{p(\boldsymbol{\theta}_n}{\tilde{p}(\boldsymbol{\theta}_n)} \log q_{\boldsymbol{\psi}}(\boldsymbol{\theta}_n|\mathbf{x}_n)$ to directly recover $p(\boldsymbol{\theta}|\mathbf{x})$ without the need for post-hoc correction, removing restrictions with respect to $q_{\boldsymbol{\psi}}$, the proposal, and prior. However, the importance weights can have high variance during training, leading to inaccurate inference for some tasks (Greenberg et al., 2019). SNPE-C (APT) (Greenberg et al., 2019) alleviates this issue by reparameterizing the problem such that it can infer the posterior by maximizing an estimated proposal posterior. It trains F to approximate $p(\boldsymbol{\theta}|\mathbf{x})$ with $q_{F(\mathbf{x},\boldsymbol{\phi})}(\boldsymbol{\theta})$, using a loss defined on the approximate proposal posterior $\tilde{q}_{\mathbf{x},\boldsymbol{\phi}}(\boldsymbol{\theta})$. Greenberg et al. (2019) introduce 'atomic' proposals to allow for arbitrary choices of the density estimator, e.g., flows (Papamakarios et al., 2019a): The loss on $\tilde{q}_{\mathbf{x},\boldsymbol{\phi}}(\boldsymbol{\theta})$ is calculated as the expectation over proposal sets Θ sampled from a so-called 'hyperproposal' $V(\Theta)$ as outlined in Algorithm 6 (see Greenberg et al., 2019, for full details).

For the benchmark, we used the approach by Greenberg et al. (2019) with 'atomic' proposals and referred to it as SNPE. As density estimator, we used a Neural Spline Flow (Durkan et al., 2019) with the same settings as for NPE. For the 'atomic' proposals, we used $M = 10$ atoms (larger M was too demanding in terms of memory). The simulation budget was equally split across 10 rounds and for the final round, we obtained 10k samples from the approximate posterior $\hat{p}_R(\boldsymbol{\theta}|\mathbf{x}_o)$. In Appendix H, we compare NSFs to Masked Autoregressive Flows (MAFs, Papamakarios et al., 2017), as used in Greenberg et al. (2019); Durkan et al. (2020), with five flow transforms, each with two blocks and 50 hidden units, tanh non-linearity and batch normalization after each layer.

A.7 Neural Ratio Estimation (NRE)

Algorithm 7: Single round Neural Ratio Estimation as in Hermans et al. (2020)

```
Set optimization criterion $l$ (e.g., BCE)
for $j = 1 : N$ do
    Sample $\boldsymbol{\theta}_j \sim p(\boldsymbol{\theta})$
    Sample $\boldsymbol{\theta}'_j \sim p(\boldsymbol{\theta})$
    Simulate $\mathbf{x}_j \sim p(\mathbf{x}|\boldsymbol{\theta}_j)$
end
$\boldsymbol{\phi} \leftarrow \arg\min l(d_{\boldsymbol{\phi}}(\mathbf{x}_n, \boldsymbol{\theta}_n), 1) + l(d_{\boldsymbol{\phi}}(\mathbf{x}_n, \boldsymbol{\theta}'_n), 0)$
Parameterize $d_{\boldsymbol{\phi}}(\mathbf{x}, \boldsymbol{\theta})$
return Samples from $\hat{p}(\boldsymbol{\theta}|\mathbf{x}_o)$ via MCMC; $d_{\boldsymbol{\phi}}(\mathbf{x}, \boldsymbol{\theta})$
```

Neural ratio estimation (NRE) uses neural-network based classifiers to approximate the posterior $p(\boldsymbol{\theta}|\mathbf{x}_o)$. While neural-network based approaches described in the previous sections use *density estimation* to either estimate the likelihood ((S)NLE) or the posterior ((S)NPE), NRE algorithms ((S)NRE) use *classification* to estimate a ratio of likelihoods. The ratio can then be used for posterior evaluation or MCMC-based sampling.

Likelihood ratio estimation can be used for SBI because it allows to perform MCMC without evaluating the intractable likelihood. In MCMC, the transition probability from a current parameter $\boldsymbol{\theta}_t$ to a proposed parameter $\boldsymbol{\theta}'$ depends on the posterior ratio and in turn on the likelihood ratio between the two parameters:

$$\frac{p(\boldsymbol{\theta}'|\mathbf{x})}{p(\boldsymbol{\theta}_t|\mathbf{x})} = \frac{p(\boldsymbol{\theta}')p(\mathbf{x}|\boldsymbol{\theta}')/p(\mathbf{x})}{p(\boldsymbol{\theta}_t)p(\mathbf{x}|\boldsymbol{\theta}_t)/p(\mathbf{x})} = \frac{p(\boldsymbol{\theta}')p(\mathbf{x}|\boldsymbol{\theta}')}{p(\boldsymbol{\theta}_t)p(\mathbf{x}|\boldsymbol{\theta}_t)}.$$

Therefore, given a ratio estimator $r(\mathbf{x}|\boldsymbol{\theta}', \boldsymbol{\theta}_t) = \frac{p(\mathbf{x}|\boldsymbol{\theta}')}{p(\mathbf{x}|\boldsymbol{\theta}_t)}$ learned from simulations, one can perform MCMC to obtain samples from the posterior, even if evaluating $p(\mathbf{x}|\boldsymbol{\theta})$ is intractable.

Hermans et al. (2020) proposed the following approach for MCMC with classifiers to approximate density ratios: A classifier is trained to distinguish samples from an arbitrary $(\boldsymbol{\theta}, \mathbf{x}) \sim p(\mathbf{x}|\boldsymbol{\theta})p(\boldsymbol{\theta})$ and samples from the marginal model $(\boldsymbol{\theta}, \mathbf{x}) \sim p(\boldsymbol{\theta})p(\mathbf{x})$. This results in a likelihood-to-evidence estimator that needs to be trained only once to be evaluated for any $\boldsymbol{\theta}$. The training of the classifier $d_{\boldsymbol{\phi}}(\mathbf{x}, \boldsymbol{\theta})$ proceeds by minimizing the binary cross-entropy loss (BCE), as outlined in Algorithm 7. Once the classifier $d_{\boldsymbol{\phi}}(\mathbf{x}, \boldsymbol{\theta})$ is parameterized, it can be used to perform MCMC to obtain samples from the posterior. The authors name their approach *Amortized Approximate Likelihood Ratio MCMC* (AALR-MCMC): It is amortized because once the likelihood ratio estimator is trained, it is possible to run MCMC for any $\mathbf{x} \sim p(\mathbf{x})$.

Earlier ratio estimation algorithms for SBI (e.g., Izbicki et al., 2014; Pham et al., 2014; Cranmer et al., 2015; Dutta et al., 2016) and their connections to recent methods are discussed in Thomas et al. (2020), as well as in Durkan et al. (2020). AALR-MCMC is closely related to LFIRE (Dutta et al., 2016) but trains an amortized classifier rather than a separate one per posterior evaluation. Durkan et al. (2020) showed that the loss of AALR-MCMC is closely related to the atomic SNPE-C/APT approach of Greenberg et al. (2019) (SNPE) and that both can be combined in a unified framework. Durkan et al. (2020) changed the formulation of the loss function for training the classifier from binary to multi-class.

For the benchmark, we used neural ratio estimation (NRE) as formulated by Durkan et al. (2020) and implemented in the `sbi` toolbox (Tejero-Cantero et al., 2020). As a classifier, we used a residual network architecture (ResNet) with two hidden layers of 50 units and ReLU non-linearity, trained with Adam (Kingma and Ba, 2015). Following the notation of Durkan et al. (2020), we used $K = 10$ as the size of the contrasting set. For the MCMC step, we followed the same procedure as described for NLE, i.e., using Slice Sampling with 100 chains, to obtain 10k samples from each approximate posterior. In Appendix H, we show results for all tasks obtained with a multi-layer perceptron (MLP) architecture with two hidden layers of 50 ReLu units, and batch normalization.

A.8 Sequential Neural Ratio Estimation (SNRE)

Algorithm 8: Sequential Neural Ratio Estimation as in Hermans et al. (2020)

```
Set optimization criterion $l$ (e.g., BCE)
Set $\tilde{p}(\boldsymbol{\theta}) = p(\boldsymbol{\theta})$
for $r = 1 : R$ do
    for $j = 1 : N$ do
        Sample $\boldsymbol{\theta}_j \sim \tilde{p}(\boldsymbol{\theta})$ (via $d_{\boldsymbol{\phi}}$ and MCMC)
        Sample $\boldsymbol{\theta}'_j \sim \tilde{p}(\boldsymbol{\theta})$ (via $d_{\boldsymbol{\phi}}$ and MCMC)
        Simulate $\mathbf{x}_j \sim p(\mathbf{x}|\boldsymbol{\theta}_j)$
    end
    $\boldsymbol{\phi} \leftarrow \arg\min l(I'n, \boldsymbol{\theta}_n), 1) + l(d_{\boldsymbol{\phi}}(\mathbf{x}_n, \boldsymbol{\theta}'_n), 0);$
    Parameterize $d_{\boldsymbol{\phi}}(\mathbf{x}, \boldsymbol{\theta})$
end
return Samples from $\hat{p}(\boldsymbol{\theta}|\mathbf{x}_o)$ via MCMC; $d_{\boldsymbol{\phi}}(\mathbf{x}, \boldsymbol{\theta})$
```

Sequential Neural Ratio Estimation (SNRE) is the sequential version of NRE, and meant to increase sample efficiency, at the cost of needing to train new classifiers for different $\mathbf{x}_o$.

A sequential version of neural ratio estimation was proposed by Hermans et al. (2020). As with other sequential algorithms, the idea is to replace the prior by a proposal distribution $\tilde{p}(\boldsymbol{\theta})$ that is focused on $\mathbf{x}_o$ in the sense that the sampled parameters $\boldsymbol{\theta}$ result in simulated data $\mathbf{x}$ that are informative about $\mathbf{x}_o$. The proposal for the next round is the posterior estimate from the previous round. The ratio estimator then becomes $\tilde{r}(\mathbf{x}, \boldsymbol{\theta})$ and is refined over rounds by training the underlying classifier with positive examples $(\mathbf{x}, \boldsymbol{\theta}) \sim p(\mathbf{x}|\boldsymbol{\theta})\tilde{p}(\boldsymbol{\theta})$ and negative examples $(\mathbf{x}, \boldsymbol{\theta}) \sim p(\mathbf{x})\tilde{p}(\boldsymbol{\theta})$. Exact posterior evaluation is not possible anymore, but samples can be obtained as before via MCMC. These steps are outlined in Algorithm 8.

For the benchmark, we used SNRE as formulated by Durkan et al. (2020) and implemented in the `sbi` toolbox (Tejero-Cantero et al., 2020). The classifier had the same architecture as described for NRE. For the MCMC step, we followed the same procedure as described for NLE. The simulation budget was equally split across 10 rounds. In Appendix H, we show results for all tasks obtained with a multi-layer perceptron (MLP) architecture with two hidden layers of 50 ReLu units, and batch normalization.

A.9 Random Forest Approximate Bayesian Computation (RF-ABC)

Algorithm 9: Random Forest ABC (RF-ABC) as in Raynal et al. (2019)

```
Set $\mathcal{D} = \{\}$ Set simulation budget $N$
Set number of trees $B$
Set minimum node size $N_{min}$
for $n = 1 : N$ do
    Sample $\boldsymbol{\theta}_n \sim p(\boldsymbol{\theta})$
    Simulate $\mathbf{x}_n \sim p(\mathbf{x}|\boldsymbol{\theta}_n)$
    Add $(\boldsymbol{\theta}_n, \mathbf{x}_n)$ to $\mathcal{D}$
end
Run random forest regression of $\mathbf{x}$ on $\boldsymbol{\theta}$ using $\mathcal{D}$, $B$ and $N_{min}$
return $N$ samples $\{\boldsymbol{\theta}^{(i)}\}$ and associated weights $\{w^{(i)}\}$ for drawing approximate posterior samples
```

Random forest Approximate Bayesian Computation (RF-ABC, Pudlo et al., 2016; Raynal et al., 2019) is a more recently developed ABC algorithm based on a regression approach. Similar to previous regression approaches to ABC (Beaumont et al., 2002; Blum and François, 2010), RF-ABC aims at improving classical ABC inference (REJ-ABC, SMC-ABC) in the setting of high-dimensional data.

The idea of the RF-ABC algorithm is to use random forests (RF, Breiman, 2001) to run a non-parametric regression of a set of potential summary statistics of the data on the corresponding parameters. That is, the RF regression is trained on data simulated from the model, such that the covariates are the summary statistics and the response variable is a parameter. For a detailed description of the algorithm, we refer to Raynal et al. (2019).

The only hyperparameters for the RF-ABC algorithm are the number of trees and the minimum node size for the RF regression. Following Raynal et al. (2019), we chose the default of 500 trees and a minimum of 5 nodes. The output of the algorithm is a RF weight for each of the simulated parameters. This set of weights can be used to calculate posterior quantiles or to obtain an approximate posterior density as described in Raynal et al. (2019). We obtained 10k posterior samples for the benchmark by using the random forest weights to sample from the simulated parameters. We used the implementation in the `abcranger` toolbox Collin et al. (2020).

One important property of RF-ABC is that it can only be applied in the unidimensional setting, i.e., for 1-D dimensional parameter spaces, or for multidimensional parameters spaces with the assumption that the posterior factorizes over parameters (thus ignoring potential posterior correlations). This assumptions holds only for a few tasks in our benchmark (Gaussian Linear, Gaussian Linear Uniform, Gaussian Mixture). Due to this inherent limitation, we report RF-ABC in the supplement (see Suppl. Fig. 2).

A.10 Synthetic Likelihood (SL)

Algorithm 10: Synthetic Likelihood algorithm as in Wood (2010)

```
Set number of simulations per step $M$
Set number of MCMC steps $T$
for $t = 1 : T$ do
    Get new candidate $\boldsymbol{\theta}_t$ from MCMC scheme
    Set $\mathcal{D}_t = \{\}$
    for $m = 1 : M$ do
        Simulate $\mathbf{x}_m \sim p(\mathbf{x}|\boldsymbol{\theta}_t)$
        Add $(\boldsymbol{\theta}_t, \mathbf{x}_m)$ to $\mathcal{D}_t$
    end
    Use $\mathcal{D}_t$ to estimate mean and covariance of a Gaussian approximation of the likelihood $\hat{L}(\mathbf{x}_o|\theta_t)$
    Perform the next MCMC step using $\hat{L}(\mathbf{x}_o|\theta_t)$
end
return $N$ samples $\{\boldsymbol{\theta}^{(i)}\}$ from MCMC chain
```

The Synthetic Likelihood (SL) approach circumvents the evaluation of the intractable likelihood by estimating a *synthetic* one from simulated data or summary statistics. This approach was introduced by Wood (2010). Its main motivation is that the classical ABC approach of comparing simulated and observed data with a distance metric can be problematic if parts of the differences are entirely noise-driven. Wood (2010) instead approximated the distribution of the summary statistics (the likelihood) of a nonlinear ecological dynamic system as a Gaussian distribution, thereby capturing the underlying noise as well. The approximation of the likelihood can then be used to obtain posterior sampling via Markov Chain Monte Carlo (MCMC) (Wood, 2010).

The SL approach can be seen as the predecessor of the (S)NLE approaches: They replaced the Gaussian approximation of the likelihood with a much more flexible one that uses neural networks and normalizing flows (see A.3). Moreover, there are modern approaches from the classical ABC field that further developed SL using a Gaussian approximation (e.g., Drovandi et al., 2018; Priddle et al., 2019).

For the benchmark, we implemented our own version of the algorithm proposed by Wood (2010). We used Slice Sampling MCMC (Neal, 2003) and estimated the Gaussian likelihood from 100 samples at each sampling step. To ensure a positive definite covariance matrix, we added a small value ϵ to the diagonal of the estimated covariance matrix for some of the tasks. In particular, we used $\epsilon = 0.01$ for SIR and Bernoulli GLM Raw tasks, and we tried without success $\epsilon = [0, 0.01, 0.1, 1.0]$ for Lotka-Volterra and SLCP with distractors. For all remaining tasks, we set $\epsilon = 0$. For Slice Sampling, we used a single chain initialized with sequential importance sampling (SIR) as described for NLE, 1k warm-up steps and no thinning, in order to keep the number of required simulations tractable. This resulted in an overall simulation budget on the order of 10^8 to 10^9 simulations per run in order to generate 10k posterior samples, as new simulations are required for every MCMC step.

The high simulation budget makes it problematic to directly compare SL and other other algorithms in the benchmark. Therefore, we report SL in the supplement (see Suppl. Fig. 3).

B Benchmark

B.1 Reference posteriors

We generated 10k reference posterior samples for each observation. For the Gaussian Linear task, reference samples were obtained by using the analytic solution for the true posterior. Similarly, for Gaussian Linear Uniform and Gaussian Mixture, the analytic solution was used, combined with an additional rejection step, in order to account for the bounded support of the posterior due to the use of a uniform prior. For the Two Moons task, we devised a custom scheme based on the model equations, which samples both modes and rejects samples outside the prior bounds.

For SLCP, SIR, and Lotka-Volterra, we devised a likelihood-based procedure to ensure obtaining a valid set of reference posterior samples: First, we either used Sampling/Importance Resampling (Rubin, 1988) (for SLCP, SIR) or Slice Sampling MCMC (Neal, 2003) (for Lotka-Volterra) to obtain a set of 10k proposal samples from the unnormalized posterior $f(\boldsymbol{\theta}) = \tilde{p}(\boldsymbol{\theta}|\mathbf{x}_o) = p(\mathbf{x}_o|\boldsymbol{\theta})p(\boldsymbol{\theta})$. We used these proposal samples to train a density estimator, for which we used a neural spline flow (NSF) (Durkan et al., 2019). Next, we created a mixture composed of the NSF and the prior with weights 0.9 and 0.1, respectively, as a proposal distribution $g(\boldsymbol{\theta})$ for rejection sampling (Martino et al., 2018). Rejection sampling relies on finding a constant M such that $f(\boldsymbol{\theta}) \leq Mg(\boldsymbol{\theta})$ for all values of $\boldsymbol{\theta}$: To find this constant, we initialized $M = 1$, sampled $\boldsymbol{\theta} \sim g(\boldsymbol{\theta})$, and updated $M = 1.2 f(\boldsymbol{\theta})/g(\boldsymbol{\theta})$ if $f(\boldsymbol{\theta})/g(\boldsymbol{\theta}) > M$. This loop stopped only after at least 100k samples without updating M were reached. We then used M, f, and g to generate 10k reference posterior samples. We found that the NSF-based proposal distribution resulted in high acceptance rates. We used this custom scheme rather than relying on MCMC directly, since we found that standard MCMC approaches (Slice Sampling, HMC, and NUTS) all struggled with multi-modal posteriors and wanted to avoid bias in the reference samples, e.g. due to correlations in MCMC chains.

As a sanity check, we ran this scheme twice on all tasks and observation and found that the resulting reference posterior samples were indistinguishable in terms of C2ST.

B.2 Code

We provide `sbibm`, a benchmarking framework that implements all tasks, reference posteriors, different metrics and tooling to run and analyse benchmark results at scale. The framework is available at:

`github.com/sbi-benchmark/sbibm`

We make benchmarking new algorithms maximally easy by providing an open, modular framework for *integration* with SBI toolboxes. We here evaluated algorithms implemented in `pyABC` (Klinger et al., 2018), `pyabcranger` (Collin et al., 2020), and `sbi` (Tejero-Cantero et al., 2020). We emphasize that the goal of `sbibm` is orthogonal to any toolbox: It could easily be used with other toolboxes, or even be used to compare results for the same algorithm implemented by different ones. There are currently several SBI toolboxes available or under active development. `elfi` (Lintusaari et al., 2018) is a general purpose toolbox, including ABC algorithms as well as BOLFI (Gutmann and Corander, 2016). There are many toolboxes for ABC algorithms, e.g., `abcpy` (Dutta et al., 2017), `astroABC` (Jennings and Madigan, 2017), `CosmoABC` (Ishida et al., 2015), see also Kousathanas et al. (2018) for an overview. `carl` (Louppe et al., 2016) implements the algorithm by Cranmer et al. (2015). `hypothesis` (Hermans, 2019), and `pydelfi` (Alsing, 2019) are SBI toolboxes under development.

B.3 Reproducibility

To ensure reproducibility of our results, we publicly released all code including instructions on how to run the benchmark on cloud-based infrastructure.

F Figures

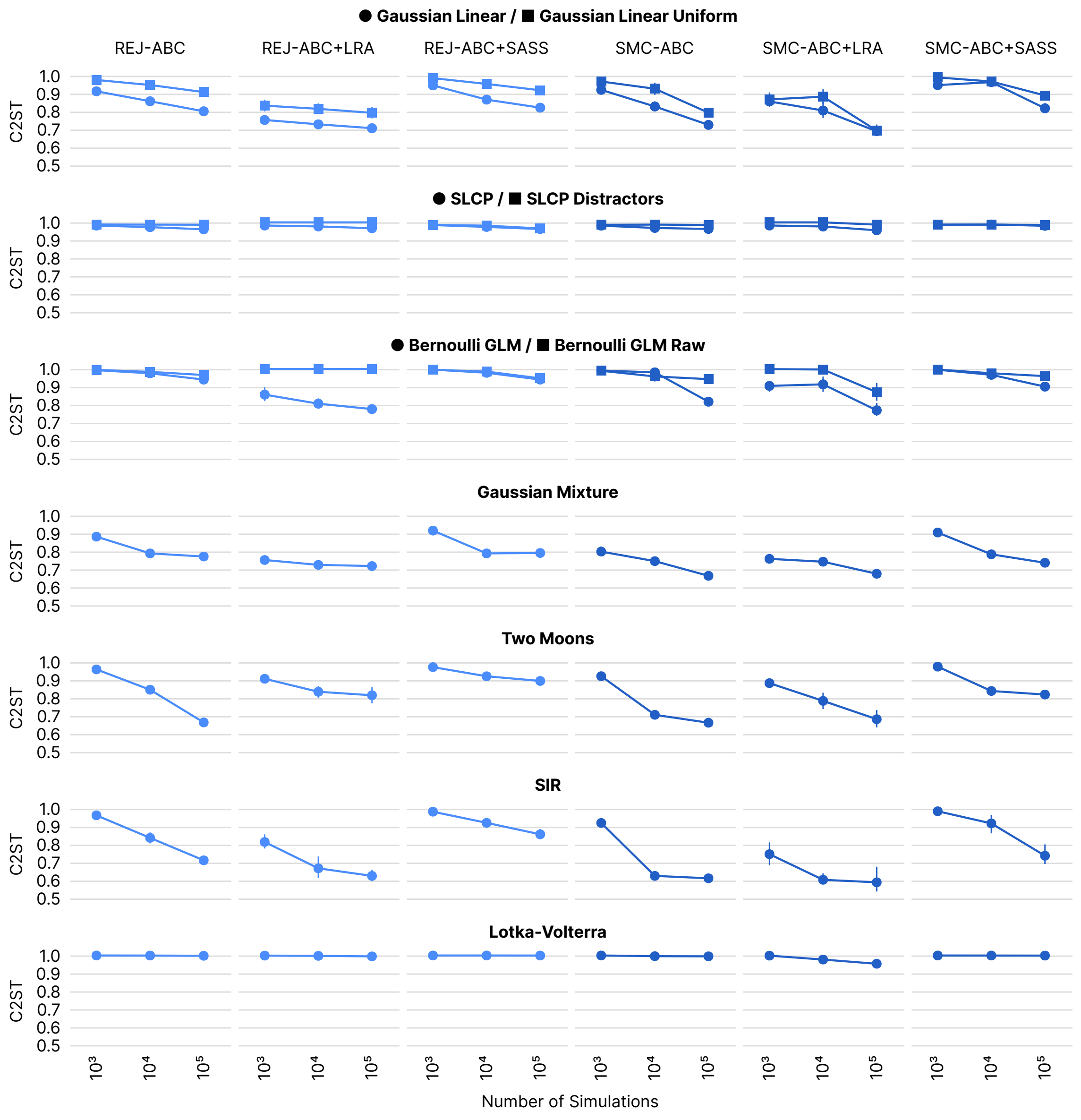


Figure 1: **Additional ABC results with linear regression adjustment (LRA) and semi-automatic summary-statistics (SASS).** We ran ABC with post-hoc LRA (Beaumont et al., 2002; Blum, 2018). On some tasks, this led to an improvement relative to versions without post-hoc adjustment. On Two Moons (bimodal posterior), linear adjustment decreased performance. We implemented our own SASS (Prangle et al., 2014b) with a third order polynomial feature expansion, and observed similar performance as with the implementation in `abcpy` toolbox (Dutta et al., 2017).

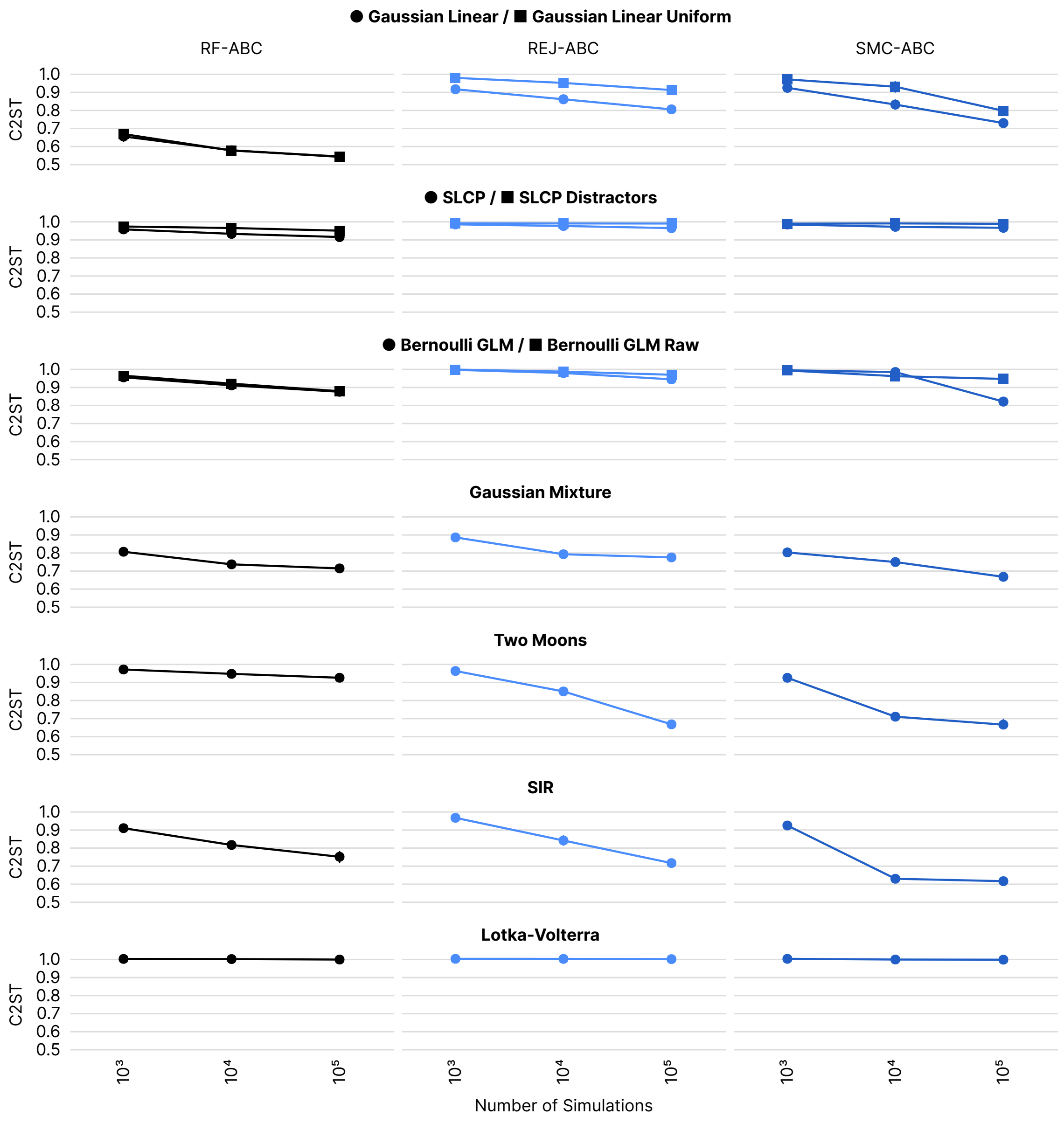


Figure 2: **RF-ABC results.** Results for RF-ABC (as described in A.9) compared to REJ-ABC and SMC-ABC on all benchmark tasks, using C2ST. Note that RF-ABC predicts each parameter individually, i.e. effectively assumes the posterior to be factorized– this is only appropriate for the Gaussian Linear, Gaussian Linear Uniform, and Gaussian Mixture tasks. On other tasks, the posterior deviates markedly from being factorized, and therefore it is to be expected that RF-ABC performance is limited, even when using many samples. Each data point corresponds to the mean and 95% confidence interval across 10 observations.

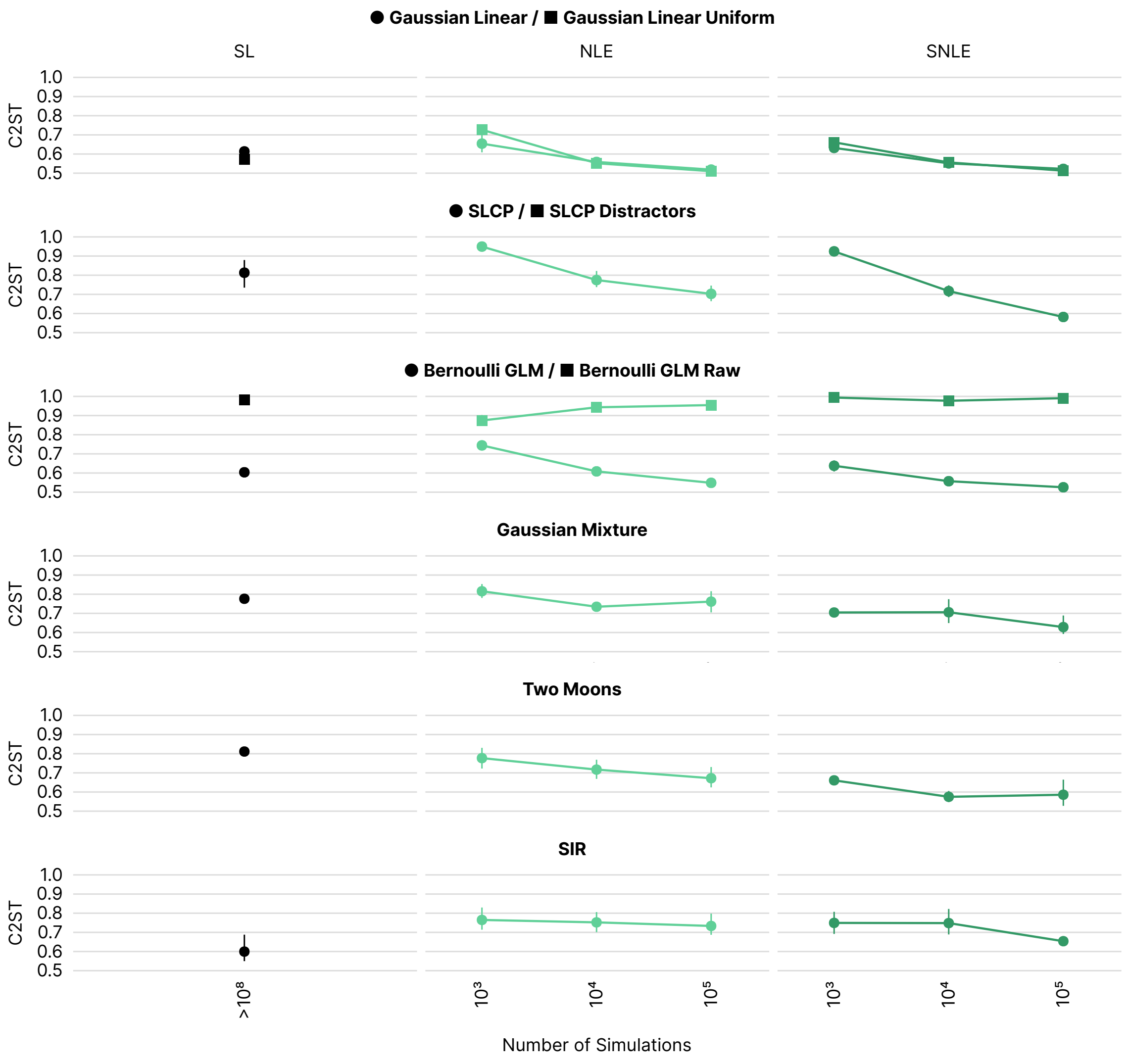


Figure 3: **SL results.** Results for SL compared to NLE and SNLE on benchmark tasks in terms of C2ST. Note that SL performs simulations at every MCMC step to approximate a Gaussian likelihood (see A.10 for details), and therefore it does not produce sensible results with the simulation budgets of other algorithms (between 1k and 100k), . In our experiments, SL required on the order of 10^8 to 10^9 simulations. For the SLCP Distractors and Lotka-Volterra stable estimation of covariances was not possible, which is why these tasks were omitted (details in A.10). We do not report SL results in the main paper, given the huge difference in simulation budget. Each data point corresponds to the mean and 95% confidence interval across 10 observations.

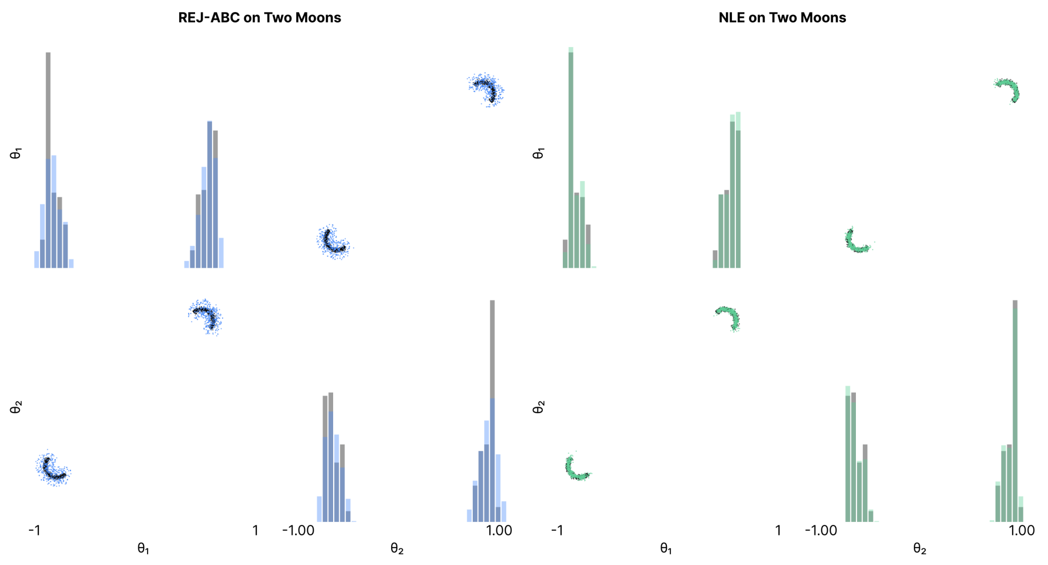


Figure 4: **MMD on Two Moons.** When using MMD with the median heuristic (as commonly done, including in SBI papers), MMD is slightly lower for the posterior obtained by REJ-ABC (left, blue samples), than for SNLE samples (right, green samples): 0.00729 (REJ-ABC) versus 0.00772 (NLE). This is at odds with the visual impression of the quality of the fit (reference samples in gray) as well as C2ST results: A classifier performed near chance level (.502) for SNLE samples while being able to tell apart REJ-ABC samples from the reference with accuracy 0.794. When manually choosing a length scale on the median distance of a *single crescent* (i.e., 0.09 instead of 1.78), MMD results were in agreement with C2ST results: 0.00738 (REJ-ABC) versus 0.00035 (SNLE), i.e., they also suggested a better fit for SNLE. In the main paper, we prefer to report C2ST because we found it less sensitive to hyperparameters: reliance on the commonly used median heuristic can be problematic on tasks with complex posterior structure, e.g., multi-modality in Two Moons, as demonstrated here. We refer the interested reader to Liu et al. (2020) for further illustrative examples of where MMD with Gaussian kernels can have limited power. We also want to point out that new kernel-based two sample tests are being actively developed which might make them easier to use on such problems in the future.

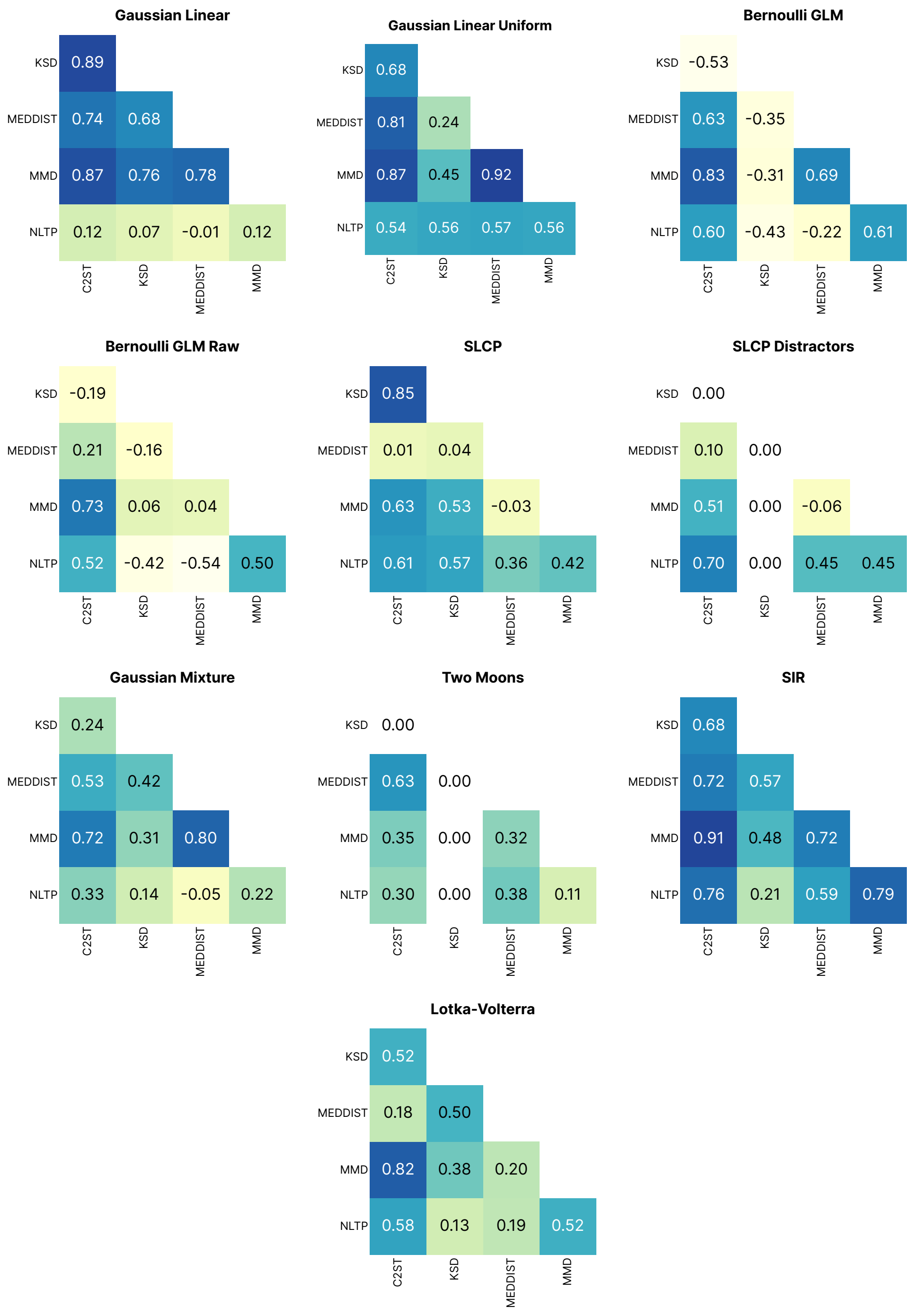


Figure 5: **Correlations between metrics for all tasks.** NLTP is the negative log probability of true parameters. Note that calculation of KSD was numerically unstable when calculating gradients for SLCP Distractors and Two Moons, resulting in correlation of zero for these tasks.

H Hyperparameter Choices

In this section, we address two central questions for any benchmark: (1) how hyperparameters are chosen and (2) how sensitive results are to the respective choices.

Rather than tuning hyperparameters on a per-task basis, we changed hyperparameters on multiple or all tasks at once and selected configurations that worked best across tasks. We wanted to avoid overfitting on individual benchmark tasks and were instead interested in settings that can generalize across multiple tasks. In practice, tuning an algorithm on a given task would typically be impossible, due to the lack of suitable metrics that can be computed without reference posteriors as well as high computational demands that SBI tasks often have.

To find good general settings, we performed more than 10 000 individual runs. We explored hyperparameter choices that have not been previously reported, and revealed substantial improvements. The benchmark offers the possibility to systematically compare different choices and design better and more robust SBI algorithms.

H.1 REJ-ABC

Classical ABC algorithms have crucial hyperparameters, most importantly, the distance metric and acceptance tolerance ϵ. We used our own implementation of REJ-ABC as it is straightforward to implement (see A.1). The distance metric was fixed to be the l_2-norm for all tasks and we varied different acceptance tolerances ϵ across tasks on which REJ-ABC performed sufficiently well. Our implementation of REJ-ABC is quantile based, i.e,. we select a quantile of the samples with the smallest distance to the observed data, which implicitly defines an ϵ. The 10k samples needed for the comparison to the reference posterior samples are then resampled from the selected samples. In order to check whether this resampling significantly impaired performance, we alternatively fit a KDE in order to obtain 10k samples.

Below, we show results for different schedules of quantiles for each simulation budget, e.g., a schedule of 0.1, 0.01, 0.001 corresponds to the 10, 1 and 0.1 percent quantile, or the top 100 samples for each simulation budget. Across tasks and budgets the 0.1, 0.01, 0.001 quantile schedule performed best (Fig. 6). Performance showed improvement by the KDE fit, especially on the Gaussian tasks. We therefore report the version using the top 100 samples and KDE in the main paper.

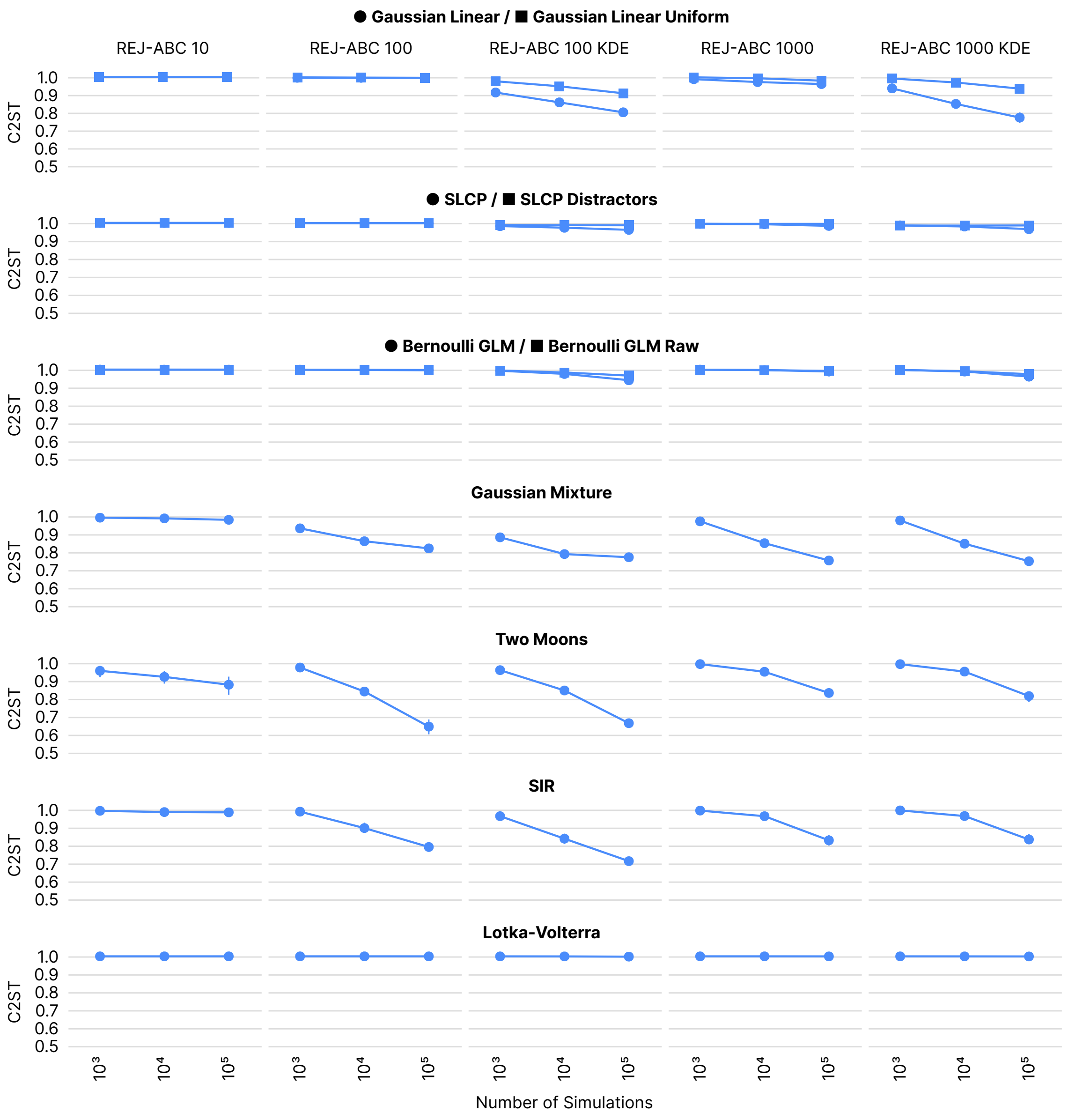


Figure 6: **Hyperparameter selection for REJ-ABC.** C2ST performance of different percentile schedules across simulation budgets (columns) for all tasks (rows). Top label for each plot column: number of samples retained, and optional KDE. Across tasks and budgets, the schedule of 0.1, 0.01, 0.001 percentiles, which corresponds to the top 100 samples closest to the observation, performed best. Each data point corresponds to the mean and 95% confidence interval across 10 observations.

H.2 SMC-ABC

SMC-ABC has several hyperparameters including the population size, the perturbation kernel, the epsilon schedule and the distance metric. In order to ensure that we report the best possible SMC-ABC results for a fair comparison, we sweeped over three hyperparameters that are especially critical: the population size, the quantile used to select the epsilon from the distances of the particles of the previous population, and the scaling factor of the covariance of the Gaussian perturbation kernel. The remaining hyperparameters were fixed to values common in the literature: Gaussian perturbation kernel and $l2$-norm distance metric.

Additionally, we compared our implementation against one from the popular pyABC toolbox (Klinger et al., 2018) to which we refer as versions A and B respectively. We sweeped over these hyperparameters and optionally added a post-hoc KDE fit for drawing the samples needed for two-sample based performance metrics.

Overall, the parameter setting with a population size of 100, a kernel covariance scale of 0.5, and an epsilon quantile 0.2 performed best. Although the results of the two different implementations were qualitatively very similar (compare Fig. 7 and Fig. 8, respectively), version A was slightly better on the Gaussian tasks. Although we tried to match the implementations and the exact settings, there are small differences between the two, which might explain the difference in the results: Implementation B constructs the Gaussian perturbation kernel using kernel density estimation on the weighted samples of the previous population, whereas A constructs it using the mean and covariance estimated from samples from the previous population. The latter could be advantageous in case of a Gaussian-like (high-dimensional) posterior (Gaussian Mixture and Gaussian linear task) and disadvantageous in a non-Gaussian-like posteriors (e.g., Two Moons). We decided to report results for SMC-ABC in the main paper using implementation A (ours) with population size 100 for simulation budgets 1k and 10k, and population size 1000 for simulation budget 100k, a kernel covariance scale of 0.5, and epsilon quantile 0.2. This choice of kernel covariance scale is different from recommendations in the literature (Sisson et al., 2007; Beaumont et al., 2009). We only found very small performance differences for different scales and note that our choice is in line with the recommendation of the `pyABC` toolbox (pyABC API Documentation, 2020), i.e., using a scale between 0 and 1. Performance showed improvement by the KDE fit, especially on the Gaussian tasks. We therefore report the version with KDE in the main paper.

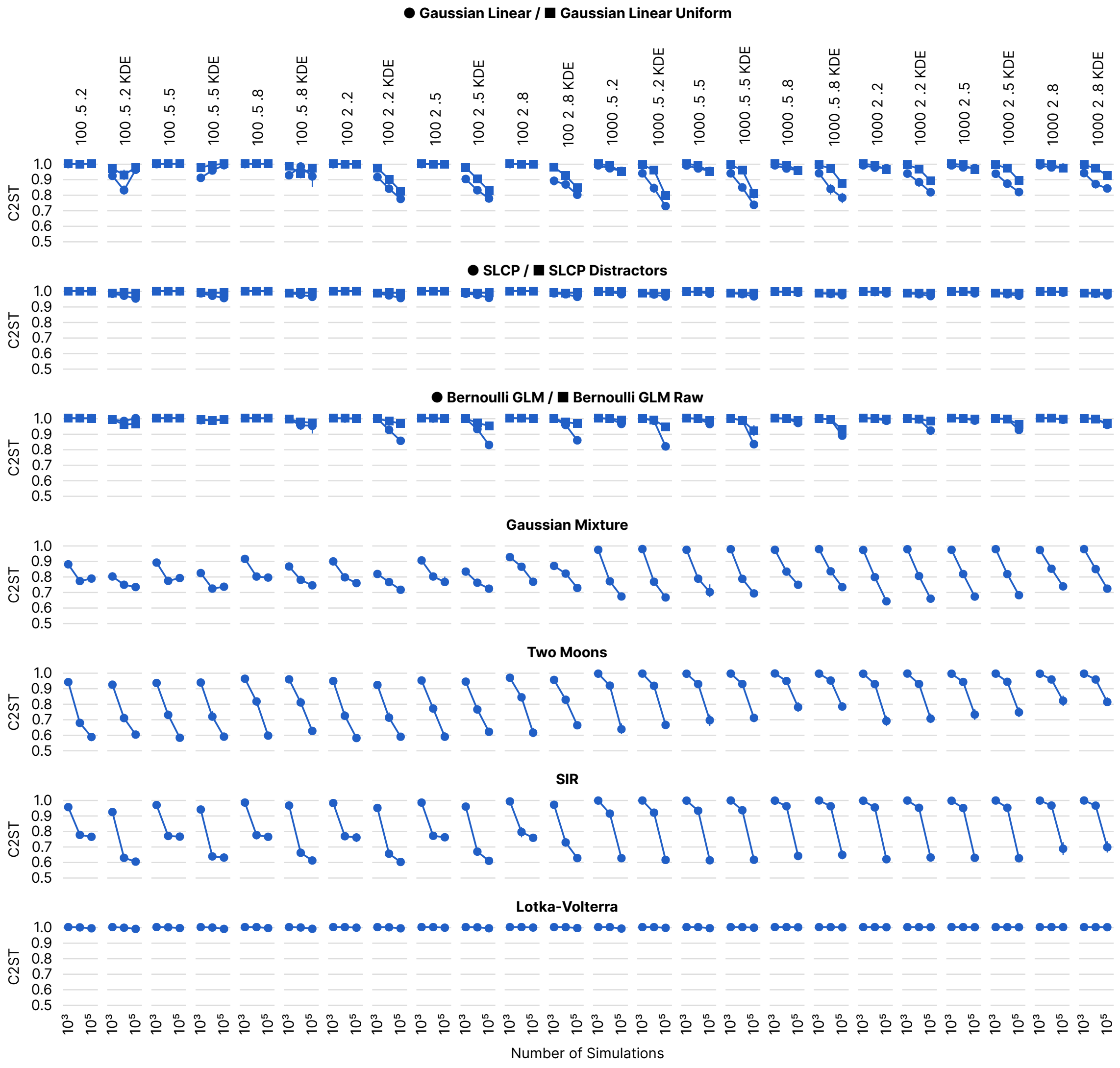


Figure 7: **Hyperparameter selection for SMC-ABC with our implementation**. Top label for each plot column: population size, kernel covariance scale, epsilon quantile/epsilon-decay parameter, and optional KDE. Each data point corresponds to the mean and 95% confidence interval across 10 observations.

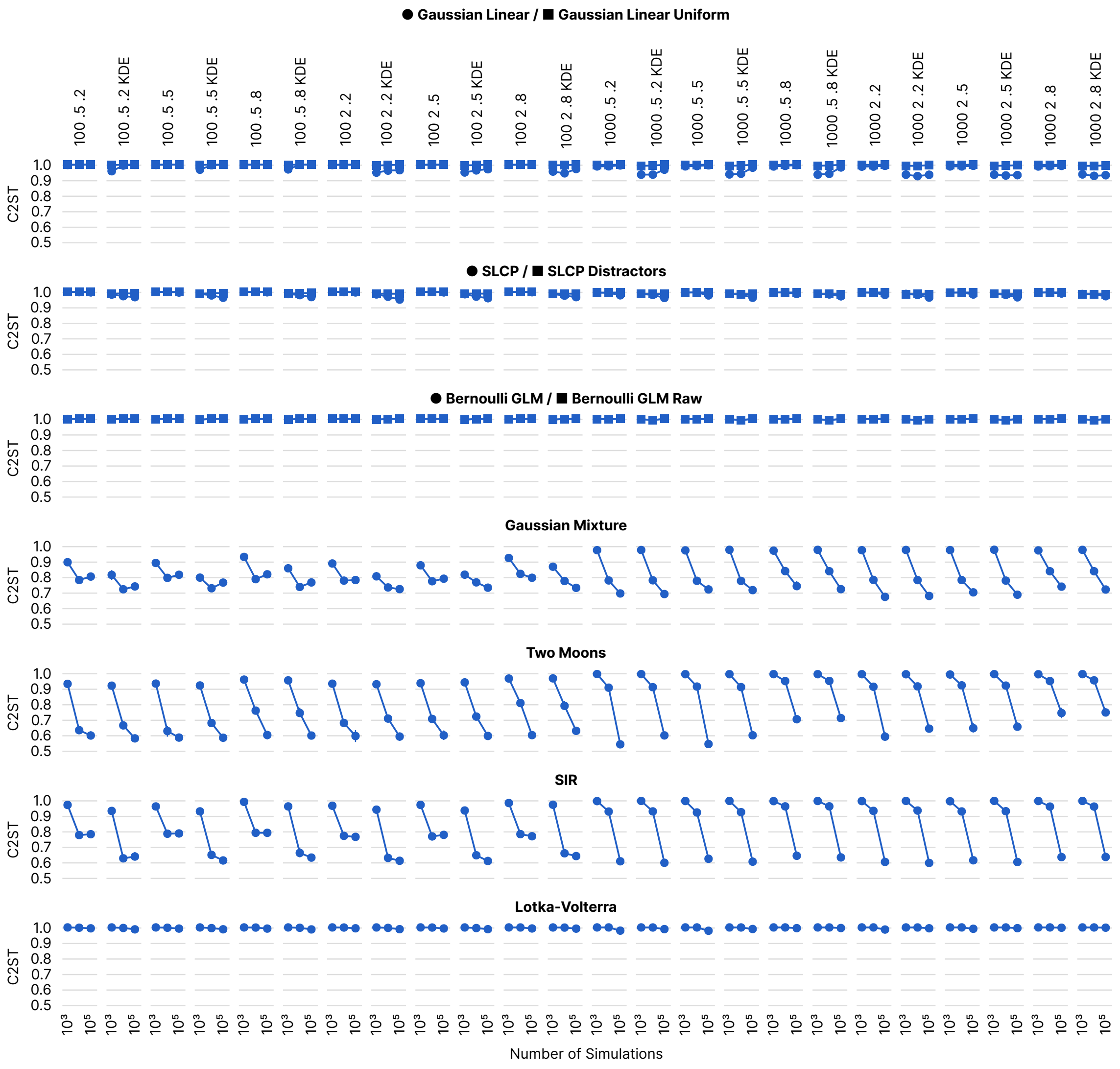


Figure 8: **Hyperparameter selection for SMC-ABC. with pyABC implementation**. Top label for each plot column: population size, kernel covariance scale, epsilon quantile/epsilon-decay parameter, and optional KDE. Each data point corresponds to the mean and 95% confidence interval across 10 observations.

H.3 MCMC for (S)NLE and (S)NRE

(S)NLE and (S)NRE both rely on MCMC sampling, which has several hyperparameters. In line with Papamakarios et al. (2019b) and Durkan et al. (2020), we used Slice Sampling (Neal, 2003). However, we modified the MCMC schemes used in these papers and obtained significant improvements in performance and speed.

Number of chains and initialization. While Papamakarios et al. (2019b); Durkan et al. (2020) used a single chain with axis-aligned updates, we found that on tasks with multi-modal posteriors, it can be essential to run multiple MCMC chains in order to sample all modes. Performance on Two Moons, for example, was poor with a single chain, since usually only one of the crescent shapes was sampled. Rather than initialising chains by drawing initial locations from the prior, we found the resampling scheme as described in A.3 to work better for initialisation, and used 100 chains instead of a single one.

Transformation of variables. When implementing MCMC, it is common advice to transform problems to have unbounded support (Hogg and Foreman-Mackey, 2018), although this has not been discussed in SBI papers or implemented in accompanying code. We found that without this transformation, MCMC sampling could get stuck in endless loops, e.g., on the Lotka-Volterra task. Apart from the transformation to unbounded space, we found z-scoring of parameters and data to be crucial for some tasks.

Vectorization of MCMC sampling. We reimplemented Slice Sampling so that all chains could perform likelihood evaluations in parallel. Evaluating likelihoods, e.g., in the case of (S)NLE, requires passes through a flow-based density estimator, which is significantly faster when batched. This allowed us to sample all chains in parallel rather than sequentially and yielded huge speed-ups: For example, SNLE on Gaussian Linear took more than 36 hours on average for 100k simulations without vectorization, and less than 2 hours with vectorization.

H.4 Density estimator for (S)NLE

Approaches based on neural networks (NN) tend to have many hyperparameters, including the concrete type of NN architecture and hyperparameters for training. We strove to keep our choices close to Durkan et al. (2020), which are the defaults in the toolbox we used (`sbi`, Tejero-Cantero et al., 2020).

While Papamakarios et al. (2019b); Durkan et al. (2020) used Masked Autoregressive Flows (MAFs, Papamakarios et al., 2017) for density estimation, we explored how results change when using Neural Spline Flows (NSFs, Durkan et al., 2019) for density estimation. These results are shown in Fig. 9.

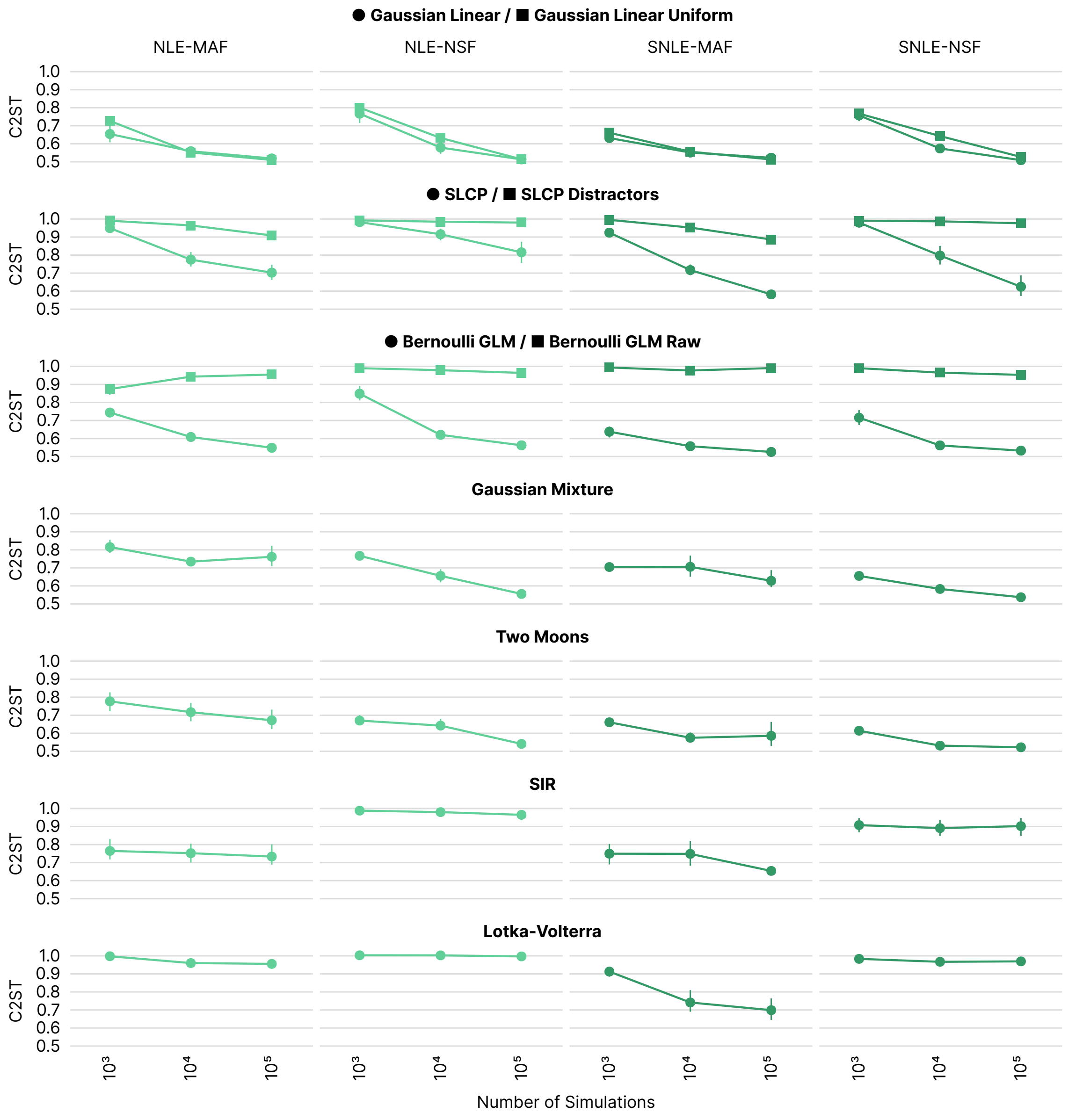


Figure 9: **Density estimator selection for (S)NLE.** Performance of (S)NLE in terms of C2ST across tasks using MAFs or NSFs for density estimation. Considering all tasks, NSFs generally performed worse, e.g., using NSFs significantly reduced performance on SIR and Lotka-Volterra, indicating that the added flexibility of NSFs was not needed for (S)NLE. We thus reported performance using MAFs in the main paper. Each data point corresponds to the mean and 95% confidence interval across 10 observations.

H.5 Density estimator for (S)NPE

We performed the analogous experiments for (S)NPE as for (S)NLE: Here, we found NSFs to increase performance relative to MAFs (Fig. 10). When directly estimating the posterior distribution, especially on tasks with complex multi-modal structure like Two Moons or SLCP, the additional flexibility offered by NSFs improved performance. With NSFs, artifacts from density transformation that were visible e.g. in Two Moons posteriors, vanished. To our knowledge, results on (S)NPE with NSFs have not been previously published.

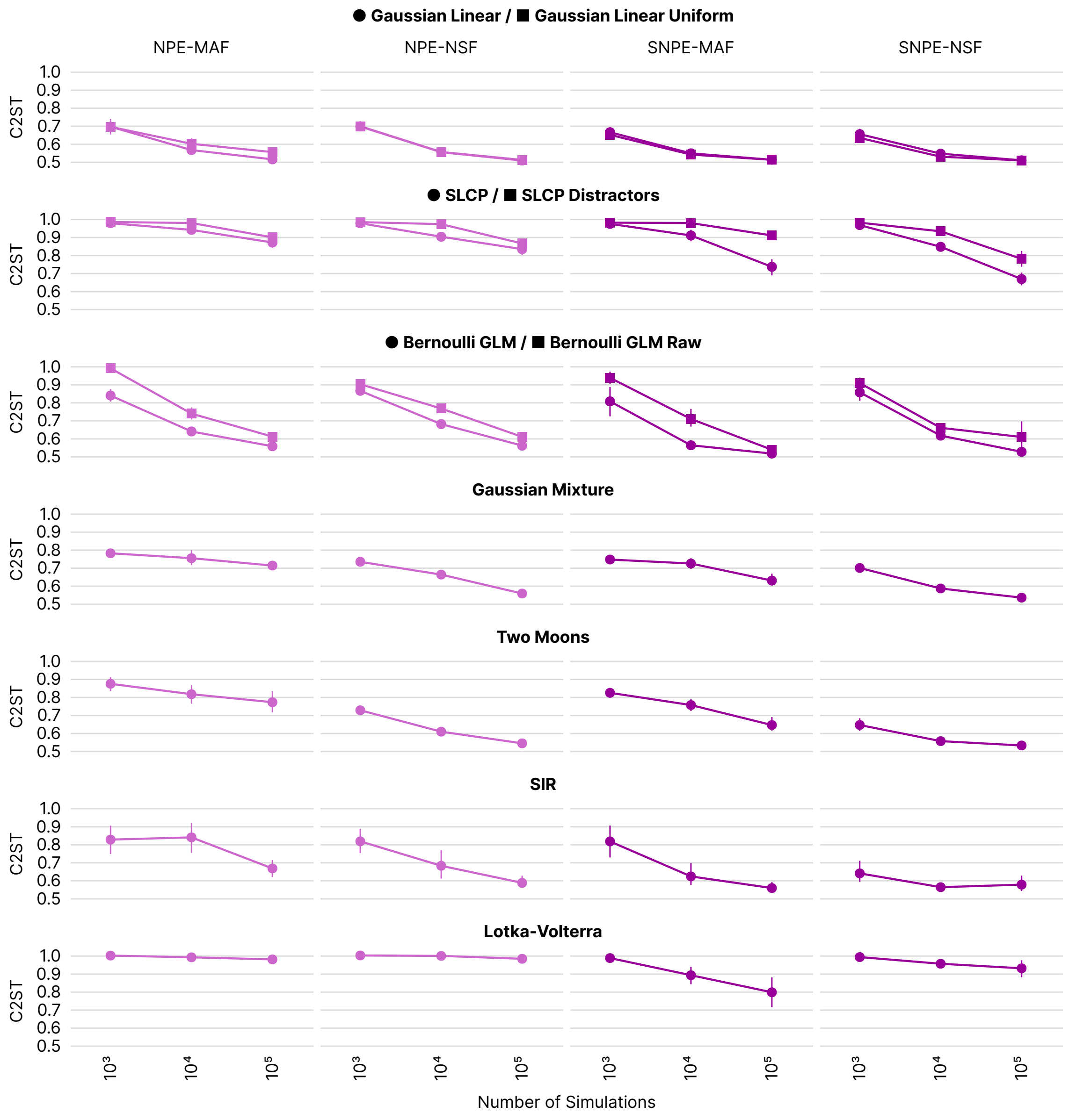


Figure 10: **Density estimator selection for (S)NPE.** Performance of (S)NPE in terms of C2ST across tasks using MAFs or NSFs for density estimation. Considering all tasks, NSFs generally performed better, especially on Gaussian Mixture, Two Moons, and SIR. We thus reported performance using NSFs in the main paper. Each data point corresponds to the mean and 95% confidence interval across 10 observations.

H.6 Classifier choice for (S)NRE

For (S)NRE, we compared two different choices of classifier architectures: an MLP and a ResNet architecture, as described in A.7. While results were similar for most tasks (Fig. 11), we decided to use the ResNet architecture in the main paper due to the better performance on Two Moons and SIR for low to medium simulation budgets.

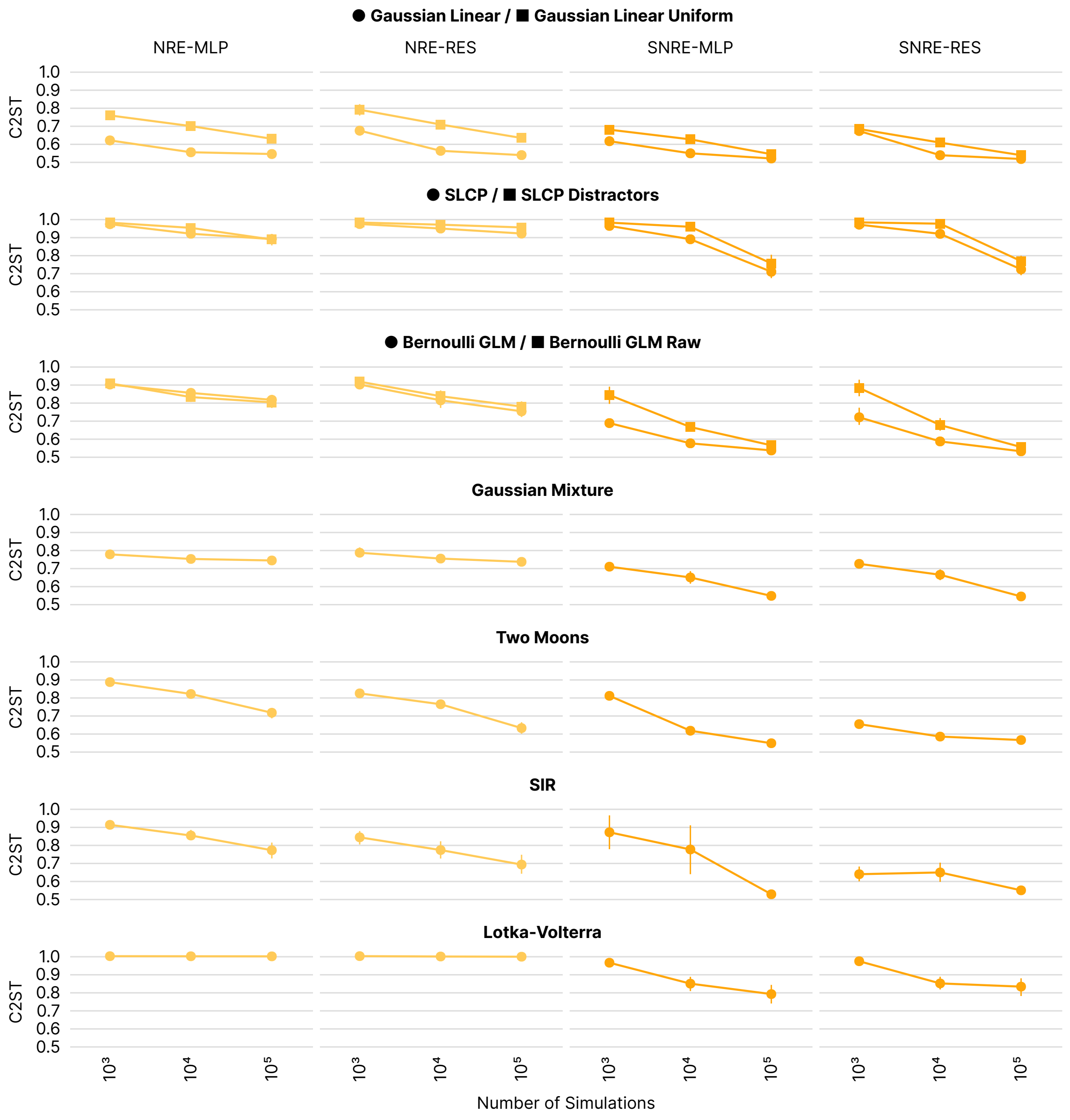


Figure 11: **Classifier architecture for (S)NRE.** Performance of (S)NRE in terms of C2ST across tasks using MLPs or ResNets for classification. Considering all tasks, ResNets generally performed better, especially on Two Moons and SIR. We thus reported performance using ResNets in the main paper. Each data point corresponds to the mean and 95% confidence interval across 10 observations.

M Metrics

M.1 Negative log probability of $\boldsymbol{\theta}_o$ (NLTP)

In simulation-based inference, the average negative log likelihood of true parameters $-\mathbb{E}[\log q(\boldsymbol{\theta}_o|\mathbf{x}_o)]$ (NLTP) is commonly reported as a performance metric in the literature (Papamakarios and Murray, 2016; Durkan et al., 2018; Papamakarios et al., 2019b; Greenberg et al., 2019; Hermans et al., 2020; Durkan et al., 2020). An attractive property of this metric is that the access to the ground-truth posterior is not required.

It is important to point out, however, that calculating this metric on a single or small number of pairs $(\boldsymbol{\theta}_o, \mathbf{x}_o)$ is problematic. To illustrate the issue, consider the following example (as discussed in Talts et al. (2018)): Consider $\theta \sim \mathcal{N}(0, 1^2), x|\theta \sim \mathcal{N}(\theta, 1^2)$, and a single pair $(\boldsymbol{\theta}_o, \mathbf{x}_o)$ with $\theta_o = 0$ and an implausible (but possible) $x_o = 2.1$. In this case, the true posterior is $\mathcal{N}(\theta|1.05, 0.5^2)$ under which the θ_o has low probability since it is more than two standard deviations away from the posterior mean. If an algorithm fitted a wrong posterior, e.g., by overestimating the standard deviation as 1 instead of 0.5, the probability of θ_o under the estimated posterior would be higher than under the true posterior.

Therefore, a large number of pairs $(\boldsymbol{\theta}_o, \mathbf{x}_o)$ should be used. Indeed, in the limit of infinite number of pairs $(\boldsymbol{\theta}_o, \mathbf{x}_o)$, the metric converges to a D_{KL}:

$$\begin{aligned}
&\mathbb{E}_{\boldsymbol{\theta}_o \sim p(\boldsymbol{\theta})} \mathbb{E}_{\mathbf{x}_o \sim p(\mathbf{x}|\boldsymbol{\theta}_o)} \big[-\log q(\boldsymbol{\theta}_o|\mathbf{x}_o) \big] \\
&= \mathbb{E}_{\mathbf{x}_o \sim p(\mathbf{x}), \boldsymbol{\theta}_o \sim p(\boldsymbol{\theta}|\mathbf{x}_o)} \big[-\log q(\boldsymbol{\theta}_o|\mathbf{x}_o) \big] \\
&= \mathbb{E}_{\mathbf{x}_o \sim p(\mathbf{x}), \boldsymbol{\theta}_o \sim p(\boldsymbol{\theta}|\mathbf{x}_o)} \big[-\log q(\boldsymbol{\theta}_o|\mathbf{x}_o) + \log p(\boldsymbol{\theta}_o|\mathbf{x}_o) \big] - \mathbb{E}_{\mathbf{x}_o \sim p(\mathbf{x}), \boldsymbol{\theta}_o \sim p(\boldsymbol{\theta}|\mathbf{x}_o)} \big[\log p(\boldsymbol{\theta}_o|\mathbf{x}_o) \big] \\
&= \mathbb{E}_{\mathbf{x}_o \sim p(\mathbf{x})} D_{\mathrm{KL}}(p(\boldsymbol{\theta}|\mathbf{x}_o) || q(\boldsymbol{\theta}|\mathbf{x}_o)) + \mathbb{E}_{\mathbf{x}_o \sim p(\mathbf{x})} \mathbb{H}(p(\boldsymbol{\theta}|\mathbf{x}_o))
\end{aligned}$$

The first term in the final equation is the average D_{KL} between true and approximate posteriors over all observations $\mathbf{x}_o$ that can be generated when sampling parameters $\boldsymbol{\theta}_o$ from the prior. The second term, the entropy term, would be the same for all algorithms compared.

In the context of this benchmark, we decided against using the probability of $\boldsymbol{\theta}_o$ as a metric: For all algorithms that are not amortized (all but one), evaluating posteriors at different $\mathbf{x}_o$ would require rerunning inference. As the computational requirements for running the benchmark at 10 observations per task are already high, running tasks for hundreds of observations would become prohibitively expensive.

M.2 Simulation-based calibration (SBC)

In simulation-based calibration (SBC), samples $\boldsymbol{\theta}'$ are drawn from the data-averaged posterior, i.e., the posterior obtained by running inference for many observations. When the posterior approximation is exact, $\boldsymbol{\theta}'$ is distributed according to the prior (Talts et al., 2018).

Let us briefly illustrate this: In SBC, we draw $\boldsymbol{\theta} \sim p(\boldsymbol{\theta}), \mathbf{x} \sim p(\mathbf{x}|\boldsymbol{\theta}), \boldsymbol{\theta}' \sim q(\boldsymbol{\theta}'|\mathbf{x})$, which implies a joint distribution $\pi(\boldsymbol{\theta}, \mathbf{x}, \boldsymbol{\theta}') = p(\boldsymbol{\theta})p(\mathbf{x}|\boldsymbol{\theta})q(\boldsymbol{\theta}'|\mathbf{x})$. The marginal $\pi(\boldsymbol{\theta}')$ is then:

$$\pi(\boldsymbol{\theta}') = \int \int p(\boldsymbol{\theta})p(\mathbf{x}|\boldsymbol{\theta})q(\boldsymbol{\theta}'|\mathbf{x}) \, \mathrm{d}\mathbf{x} \, \mathrm{d}\boldsymbol{\theta} = \int \int p(\boldsymbol{\theta}, \mathbf{x})q(\boldsymbol{\theta}'|\mathbf{x}) \, \mathrm{d}\mathbf{x} \, \mathrm{d}\boldsymbol{\theta} = \int p(\mathbf{x}) \; q(\boldsymbol{\theta}'|\mathbf{x}) \, \mathrm{d}\mathbf{x}.$$

If the approximate posterior is the true posterior, the marginal on $\boldsymbol{\theta}'$ is equal to the prior: If $q(\boldsymbol{\theta}'|\mathbf{x}) = p(\boldsymbol{\theta}'|\mathbf{x})$, then $\pi(\boldsymbol{\theta}') = \int p(\mathbf{x}, \boldsymbol{\theta}') \, \mathrm{d}\mathbf{x} = p(\boldsymbol{\theta}')$, i.e., one can set up a consistency test that is based on the distribution of $\boldsymbol{\theta}'$ samples. Talts et al. (2018) do this by using frequentist tests per dimension.

Note that SBC as described above is merely a consistency check. For example, if the approximate posterior were the prior, a calibration test as described above would not be able to detect this. This is a realistic failure mode in simulation-based inference. It could happen with rejection ABC in the limit $\epsilon \to \infty$, or when learned summary statistics have no information about $\boldsymbol{\theta}$. One way around this is issue is proposed in Prangle et al. (2014a), who propose to restrict observations to a subset of all possible $\mathcal{X}$.

SBC is similar to the average negative log likelihood of true parameters described above, in that inference needs to be carried out for many observations generated by sampling from the prior. Running inference for hundreds of observations would become prohibitively expensive in terms of compute for most algorithms, which is why we do not rely on SBC in the benchmark.

M.3 Median distance (MEDDIST)

Posterior predictive checks (PPCs) use the posterior predictive distribution to predict new data, $\mathbf{x}' \sim p(\mathbf{x}'|\mathbf{x}_o) = \int p(\mathbf{x}'|\boldsymbol{\theta})q(\boldsymbol{\theta}|\mathbf{x}_o)\,\mathrm{d}\boldsymbol{\theta}$. The observed data $\mathbf{x}_o$ should look plausible under the posterior predictive distribution (Gelman et al. (2004), chapter 6). A particular PPC, used for example in Papamakarios et al. (2019b); Greenberg et al. (2019); Durkan et al. (2020), is to assess the median L2 distance between N' posterior predictive samples $\mathbf{x}'$ and $\mathbf{x}_o$. The median is used since the mean would be more sensitive to outliers.

In the benchmark, we refer to this metric as median distance (MEDDIST) and drew $N' = 10000$ samples from each posterior predictive distribution to compute it. In contrast with other metrics considered here, the median distance is computed in the space of data $\mathbf{x}$ and requires additional simulations (which could be expensive, depending on the simulator). The median distance should be considered a mere check rather than a metric and it does not necessarily test the structure of the estimated posterior.

M.4 Maximum Mean Discrepancy (MMD)

Maximum Mean Discrepancy (MMD) is an Integral Probability Metric (IPM). Linear and quadratic time estimates for using MMD as a two-sample test were derived in Gretton et al. (2012). MMD has been commonly used in the SBI literature with Gaussian kernels (Papamakarios et al., 2019b; Greenberg et al., 2019; Hermans et al., 2020), setting a single length-scale hyperparameter by using a median heuristic (Ramdas et al., 2015). We follow the same procedure, i.e., use Gaussian kernels with length-scale determined by the median heuristic on reference samples. MMDs are calculated using 10k samples from reference and approximate posteriors.

If simple kernels are used to compare distributions with complex, multimodal structure, distinct distributions can be mapped to nearby mean embeddings, resulting in low test power. On SLCP and Two Moons, for example, we found a translation-invariant kernel to be limiting, since it cannot adapt to the local structure (see Suppl. Fig. 4). This is reflected in the low correlation of MMD and C2ST (Suppl. Fig. 5). We emphasize that these issues are strictly related to simple kernels with hyperparameters commonly used in the literature. Posteriors of the Two Moons task have a structure similar to the blobs example of Liu et al. (2020), who argue for using learned kernels to overcome the aforementioned problem.

M.5 Classifier-based tests (C2ST)

In classifier-based testing, a classifier is trained to distinguish samples of the true posterior $p(\boldsymbol{\theta}|\mathbf{x}_o)$ from samples of the estimated posterior $q(\boldsymbol{\theta}|\mathbf{x}_o)$. If the samples are indistinguishable, the classification performance should be at chance level, 0.5. Practical use and properties of classifier-based 2-sample testing (C2ST) are discussed in Lopez-Paz and Oquab (2017) (see Gutmann et al., 2018; Dalmasso et al., 2020, for examples in the context of SBI).

To compute C2ST, we trained a two-layer neural network with 10 times as many ReLU units as the dimensionality of parameters, and optimize with Adam (Kingma and Ba, 2015). Classifiers were trained on 10k z-scored samples from reference and approximate posterior each. Classification accuracy was reported using 5-fold cross-validation.

M.6 Kernelized Stein Discrepancy (KSD)

Kernelized Stein Discrepancy (KSD) is a 1-sample goodness-of-fit test proposed independently by Chwialkowski et al. (2016) and Liu et al. (2016). KSD tests samples from algorithms against the gradient of unnormalized true posterior density, $\nabla_{\boldsymbol{\theta}}\,\tilde{p}(\boldsymbol{\theta}|\mathbf{x}_o)$. We used KSD with Gaussian kernels, setting the length-scale through the median heuristic, and 10k samples from each algorithm.

R Runtimes

In applications of SBI, simulations are commonly assumed to be the dominant cost. In order to make the benchmark feasible at this scale, we focused on simple simulators and optimized runtimes, e.g. we developed a new package bridging `DifferentialEquations.jl` (Rackauckas and Nie, 2017; Bezanson et al., 2017) and `PyTorch` (Paszke et al., 2019) so that generating simulations for all implemented tasks is extremely fast. This differs from many cases in practice, where the runtime costs for an algorithm are often negligible compared to the cost of simulations. Having said that, algorithms show significant differences in runtime costs, which we measured and report here.

We recorded runtimes for all algorithms on all tasks. In principle, runtimes could be reduced by employing multi-CPU architectures, however, we decided for the single CPU setup to accurately compare runtimes across all algorithms and tasks. We did not employ GPUs for training neural-networks (NN). This is because the type of NNs used in the algorithms currently in the benchmark do not benefit much from GPU versus CPU training (e.g., no CNN architecture, rather shallow and narrow networks). In fact, running SNPE on SLCP using a GeForce GTX 1080 showed slightly longer runtimes than on CPU, due to the added overhead resulting from copying data back and forth to the device. Therefore, it was more economical and comparable to run the benchmark on CPUs.

All neural network-based algorithms were run on single 3.6 GHz CPU cores of AWS C5-instances. ABC algorithms were run on single CPU cores of an internal cluster with 2.4 GHz CPUs. We observed a difference in runtimes of less than 100ms when running ABC algorithms on the same hardware as used for neural network-based algorithms.

Figure 12 shows the recorded runtimes in minutes. We observed short runtimes for REJ-ABC and SMC-ABC, as these do not require NN training or MCMC. The sequential versions of all three NN-based algorithms yielded longer runtimes than the non-sequential versions because these involve 10 rounds of NN training. Among the sequential algorithms, SNPE showed the longest runtimes. Runtimes with MAFs instead of NSFs tend to be faster, e.g. the difference between MAFs and NSFs using SNPE on SLCP at 100k simulations was about 50 minutes on average. We also emphasize that the speed of (S)NLE reported here was only obtained after vectorizing MCMC sampling. Without vectorization, runtime on the Gaussian Linear for SNLE was more than 36 hours instead of less than 2 hours (see Appendix H).

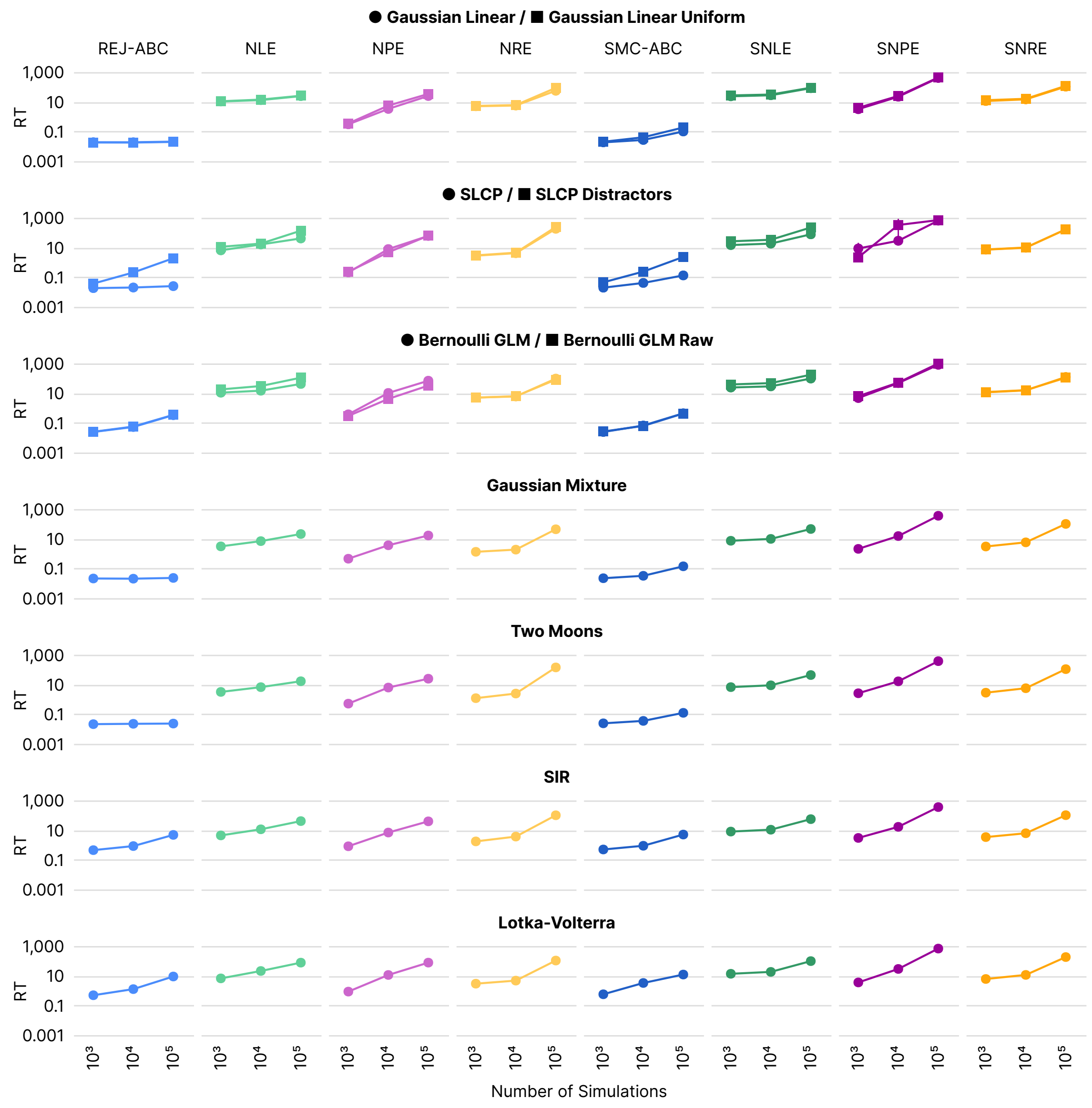


Figure 12: **Runtime on benchmark tasks.** Runtime of REJ-ABC, SMC-ABC, NLE, SNLE, NPE, SNPE, NRE, SNRE in minutes, for 10 observations each, means and 95% confidence intervals. Each run was allocated a single CPU core, see Appendix R for details.

T Tasks

T.1 Gaussian Linear

Inference of the mean of a 10-d Gaussian model, in which the covariance is fixed. The (conjugate) prior is Gaussian:

Prior	$\mathcal{N}(\mathbf{0}, 0.1 \odot \mathbf{I})$
Simulator	$\mathbf{x}\vert\boldsymbol{\theta} \sim \mathcal{N}(\mathbf{x}\vert\mathbf{m}_{\boldsymbol{\theta}} = \boldsymbol{\theta}, \mathbf{S} = 0.1 \odot \mathbf{I})$
Dimensionality	$\boldsymbol{\theta} \in \mathbb{R}^{10}, \mathbf{x} \in \mathbb{R}^{10}$

T.2 Gaussian Linear Uniform

Inference of the mean of a 10-d Gaussian model, in which the covariance is fixed. The prior is uniform:

Prior	$\mathcal{U}(-\mathbf{1}, \mathbf{1})$
Simulator	$\mathbf{x}\vert\boldsymbol{\theta} \sim \mathcal{N}(\mathbf{x}\vert\mathbf{m}_{\boldsymbol{\theta}} = \boldsymbol{\theta}, \mathbf{S} = 0.1 \odot \mathbf{I})$
Dimensionality	$\boldsymbol{\theta} \in \mathbb{R}^{10}, \mathbf{x} \in \mathbb{R}^{10}$

T.3 SLCP

A challenging inference task designed to have a simple likelihood and a complex posterior. The prior is uniform over five parameters $\boldsymbol{\theta}$ and the data are a set of four two-dimensional points sampled from a Gaussian likelihood whose mean and variance are nonlinear functions of $\boldsymbol{\theta}$:

Prior	$\mathcal{U}(-\mathbf{3}, \mathbf{3})$
Simulator	$\mathbf{x}\vert\boldsymbol{\theta} = (\mathbf{x}_1, \ldots, \mathbf{x}_4)$, $\mathbf{x}_i \sim \mathcal{N}(\mathbf{m}_{\boldsymbol{\theta}}, \mathbf{S}_{\boldsymbol{\theta}})$, where $\mathbf{m}_{\boldsymbol{\theta}} = \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix}$, $\mathbf{S}_{\boldsymbol{\theta}} = \begin{bmatrix} s_1^2 & \rho s_1 s_2 \\ \rho s_1 s_2 & s_2^2 \end{bmatrix}$, $s_1 = \theta_3^2, s_2 = \theta_4^2, \rho = \tanh \theta_5$
Dimensionality	$\boldsymbol{\theta} \in \mathbb{R}^{5}, \mathbf{x} \in \mathbb{R}^{8}$
References	Papamakarios et al. (2019b); Greenberg et al. (2019); Hermans et al. (2020) Durkan et al. (2020)

T.4 SLCP with Distractors

This task is similar to T.3, with the difference that we add uninformative dimensions (distractors) to the observation:

Prior	$\mathcal{U}(-\mathbf{3}, \mathbf{3})$
Simulator	$\mathbf{x}\vert\boldsymbol{\theta} = (\mathbf{x}_1, \ldots, \mathbf{x}_{100})$, $\mathbf{x} = p(\mathbf{y})$, where p re-orders the dimensions of $\mathbf{y}$ with a fixed random permutation, $\mathbf{y}_{[1:8]} \sim \mathcal{N}(\mathbf{m}_{\boldsymbol{\theta}}, \mathbf{S}_{\boldsymbol{\theta}})$, $\mathbf{y}_{[9:100]} \sim \frac{1}{20} \sum_{i=1}^{20} t_2(\boldsymbol{\mu}^i, \boldsymbol{\Sigma}^i)$ where $\mathbf{m}_{\boldsymbol{\theta}} = \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix}$, $\mathbf{S}_{\boldsymbol{\theta}} = \begin{bmatrix} s_1^2 & \rho s_1 s_2 \\ \rho s_1 s_2 & s_2^2 \end{bmatrix}$, $s_1 = \theta_3^2, s_2 = \theta_4^2, \rho = \tanh \theta_5$, $\boldsymbol{\mu}^i \sim \mathcal{N}(0, 15^2\mathbf{I})$, $\boldsymbol{\Sigma}^i_{j,k} \sim \mathcal{N}(0, 9)$, for $j > k$, $\boldsymbol{\Sigma}^i_{j,j} = 3e^a$, where $a \sim \mathcal{N}(0, 1)$, $\boldsymbol{\Sigma}^i_{j,k} = 0$ otherwi
Dimensionality	$\boldsymbol{\theta} \in \mathbb{R}^{5}, \mathbf{x} \in \mathbb{R}^{100}$
References	Greenberg et al. (2019)

T.5 Bernoulli GLM

Inference of a 10-parameter Generalized linear model (GLM) with Bernoulli observations, and Gaussian prior with covariance matrix which encourages smoothness by penalizing the second-order differences in the vector of parameters (De Nicolao et al., 1997). The observations are the sufficient statistics for this GLM:

Prior	$\beta \sim \mathcal{N}(0, 2)$, $\mathbf{f} \sim \mathcal{N}(\mathbf{0}, (\mathbf{F}^\top \mathbf{F})^{-1})$,
	$\mathbf{F}_{i,i-2} = 1$, $\mathbf{F}_{i,i-1} = -2$, $\mathbf{F}_{i,i} = 1 + \sqrt{\frac{i-1}{9}}$, $\mathbf{F}_{i,j} = 0$ otherwise, $1 \leq i, j \leq 9$
Simulator	$\mathbf{x} \vert \boldsymbol{\theta} = (\mathbf{x}_1, \ldots, \mathbf{x}_{10})$, $\mathbf{x}_1 = \sum_i^T z_i$, $\mathbf{x}_{2:10} = \frac{1}{\mathbf{x}_1} \mathbf{V}\mathbf{z}$,
	$z_i \sim \text{Bern}(\eta(\mathbf{v}_i^\top \mathbf{f} + \beta))$, $\eta(\cdot) = \exp(\cdot)/(1 + \exp(\cdot))$,
	frozen input between time bins $i - 8$ and i: $\mathbf{v}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, $\mathbf{V} = [v_1, v_2, \ldots, v_T]$
Dimensionality	$\boldsymbol{\theta} \in \mathbb{R}^{10}, \mathbf{x} \in \mathbb{R}^{10}$
Fixed parameters	Duration of task $T = 100$.
References	Lueckmann et al. (2017); Gonçalves et al. (2020)

T.6 Bernoulli GLM Raw

This task is similar to T.5, the sole difference being that the observations are not the sufficient statistics for the Bernoulli GLM process but the raw observations:

Prior	$\beta \sim \mathcal{N}(0, 2)$, $\mathbf{f} \sim \mathcal{N}(\mathbf{0}, (\mathbf{F}^\top \mathbf{F})^{-1})$,
	$\mathbf{F}_{i,i-2} = 1$, $\mathbf{F}_{i,i-1} = -2$, $\mathbf{F}_{i,i} = 1 + \sqrt{\frac{i-1}{9}}$, $\mathbf{F}_{i,j} = 0$ otherwise $1 \leq i, j \leq 9$
Simulator	$\mathbf{x} \vert \boldsymbol{\theta} = (\mathbf{x}_1, \ldots, \mathbf{x}_{100})$, $x_i \sim \text{Bern}(\eta(\mathbf{v}_i^\top \mathbf{f} + \beta))$, $\eta(\cdot) = \exp(\cdot)/(1 + \exp(\cdot))$
	frozen input between time bins $i - 8$ and i: $\mathbf{v}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$,
Dimensionality	$\boldsymbol{\theta} \in \mathbb{R}^{10}, \mathbf{x} \in \mathbb{R}^{100}$
Fixed parameters	Duration of task $T = 100$.

T.7 Gaussian Mixture

This task is common in the ABC literature. It consists of inferring the common mean of a mixture of two two-dimensional Gaussian distributions, one with much broader covariance than the other:

Prior	$\mathcal{U}(-\mathbf{10}, \mathbf{10})$
Simulator	$\mathbf{x} \vert \boldsymbol{\theta} \sim 0.5\, \mathcal{N}(\mathbf{x} \vert \mathbf{m}_{\boldsymbol{\theta}} = \boldsymbol{\theta}, \mathbf{S} = \mathbf{I}) + 0.5\, \mathcal{N}(\mathbf{x} \vert \mathbf{m}_{\boldsymbol{\theta}} = \boldsymbol{\theta}, \mathbf{S} = 0.01 \odot \mathbf{I})$
Dimensionality	$\boldsymbol{\theta} \in \mathbb{R}^2, \mathbf{x} \in \mathbb{R}^2$
References	Sisson et al. (2007); Beaumont et al. (2009); Toni et al. (2009); Simola et al. (2020)

T.8 Two Moons

A two-dimensional task with a posterior that exhibits both global (bimodality) and local (crescent shape) structure to illustrate how algorithms deal with multimodality:

Prior	$\mathcal{U}(-\mathbf{1}, \mathbf{1})$
Simulator	$\boldsymbol{x}\vert\boldsymbol{\theta} = \begin{bmatrix} r\cos(\alpha) + 0.25 \\ r\sin(\alpha) \end{bmatrix} + \begin{bmatrix} -\lvert\theta_1 + \theta_2\rvert/\sqrt{2} \\ (-\theta_1 + \theta_2)/\sqrt{2} \end{bmatrix}$, where $\alpha \sim \mathcal{U}(-\pi/2, \pi/2)$, $r \sim \mathcal{N}(0.1, 0.01^2)$
Dimensionality	$\boldsymbol{\theta} \in \mathbb{R}^2, \mathbf{x} \in \mathbb{R}^2$
References	Greenberg et al. (2019)

T.9 SIR

The SIR model is an epidemiological model describing the dynamics of the number of individuals in three possible states: susceptible S, infectious I, and recovered or deceased R.

The SIR task consists in inferring the contact rate β and the mean recovery rate γ, given a sampled number of individuals in the infectious group I in 10 evenly-spaced points in time:

Prior	$\beta \sim \text{LogNormal}(\log(0.4), 0.5)$ $\gamma \sim \text{LogNormal}(\log(1/8), 0.2)$
Simulator	$\mathbf{x}\vert\boldsymbol{\theta} = (x_1, \ldots, x_{10})$, $x_i \sim \mathcal{B}(1000, \frac{I}{N})$, where I is simulated from $\frac{dS}{dt} = -\beta\frac{SI}{N}$ $\frac{dI}{dt} = \beta\frac{SI}{N} - \gamma I$ $\frac{dR}{dt} = \gamma I$
Dimensionality	$\boldsymbol{\theta} \in \mathbb{R}^2, \mathbf{x} \in \mathbb{R}^{10}$
Fixed parameters	Population size $N = 1000000$ and duration of task $T = 160$. Initial conditions: $(S(0), I(0), R(0)) = (N-1, 1, 0)$
References	Kermack and McKendrick (1927)

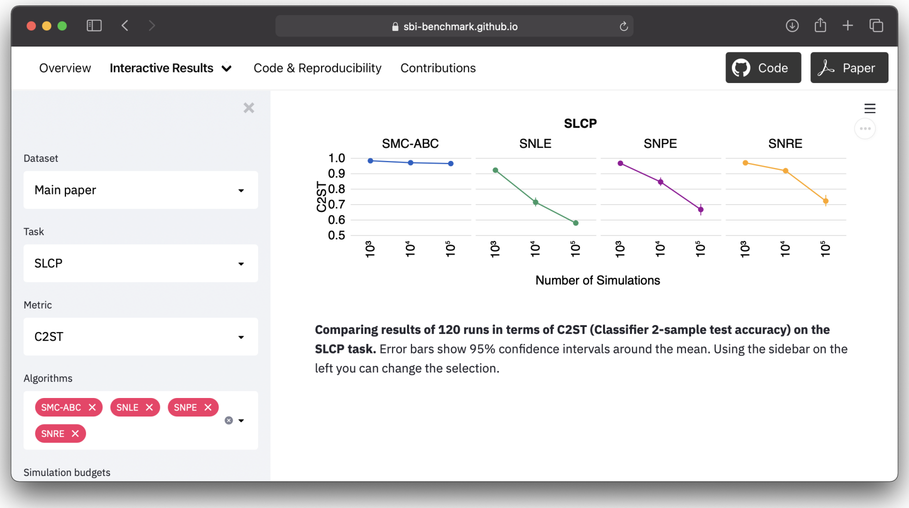
T.10 Lotka-Volterra

This is an influential model in ecology describing the dynamics of two interacting species, most commonly prey and predator interactions. Our task consists in the inference of four parameters $\boldsymbol{\theta}$ related to species interaction, given 20 summary statistics consisting of the number of individuals in both populations in 10 evenly-spaced points in time:

Prior	$\alpha \sim \text{LogNormal}(-0.125, 0.5)$, $\beta \sim \text{LogNormal}(-3, 0.5)$, $\gamma \sim \text{LogNormal}(-0.125, 0.5)$, $\delta \sim \text{LogNormal}(-3, 0.5)$
Simulator	$\mathbf{x}\vert\boldsymbol{\theta} = (\mathbf{x}_1, \ldots, \mathbf{x}_{10})$, $\mathbf{x}_{1,i} \sim \text{LogNormal}(\log(X), 0.1)$, $\mathbf{x}_{2,i} \sim \text{LogNormal}(\log(Y), 0.1)$, X and Y are simulated from $\frac{dX}{dt} = \alpha X - \beta XY$ $\frac{dY}{dt} = -\gamma Y + \delta XY$
Dimensionality	$\boldsymbol{\theta} \in \mathbb{R}^4, \mathbf{x} \in \mathbb{R}^{20}$
Fixed parameters	Duration of task $T = 20$. Initial conditions: $(X(0), Y(0)) = (30, 1)$
References	Lotka (1920)

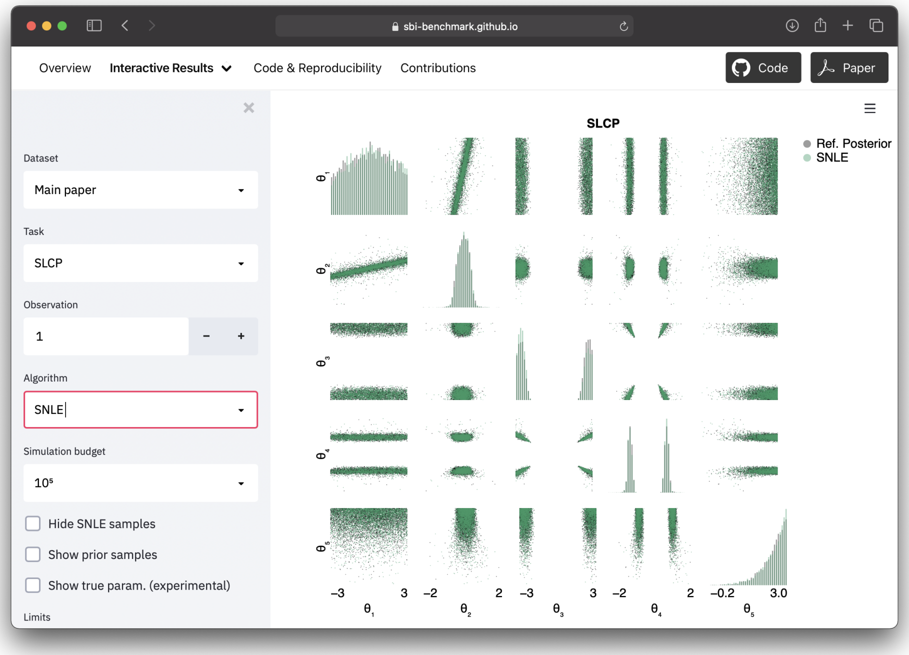
W Website

The companion website (sbi-benchmark.github.io) allows interactive comparisons in terms of all metrics. It also allows inspection of posterior samples of all runs, which we found insightful when choosing hyperparameters and diagnosing implementation issues. Two screenshots are provided in Fig. 13.



Dataset: Main paper
Task: SLCP
Metric: C2ST
Algorithms: SMC-ABC, SNLE, SNPE, SNRE

Comparing results of 120 runs in terms of C2ST (Classifier 2-sample test accuracy) on the SLCP task. Error bars show 95% confidence intervals around the mean. Using the sidebar on the left you can change the selection.



Dataset: Main paper
Task: SLCP
Observation: 1
Algorithm: SNLE
Simulation budget: 10^5
☐ Hide SNLE samples
☐ Show prior samples
☐ Show true param. (experimental)

Figure 13: **Screenshots from the companion website**. Top: Classification accuracy (C2ST) for a subset of sequential algorithms on the SLCP task. Bottom: SNLE posterior on SLCP for $\mathbf{x}_o^{(1)}$ at 100k simulations.

References

Alsing, J.
2019. pydelfi: Density estimation likelihood-free inference. `https://github.com/justinalsing/pydelfi`.

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

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

Bezanson, J., A. Edelman, S. Karpinski, and V. B. Shah
2017. Julia: A fresh approach to numerical computing. *SIAM review*, 59(1):65–98.

Bingham, E., J. P. Chen, M. Jankowiak, F. Obermeyer, N. Pradhan, T. Karaletsos, R. Singh, P. Szerlip, P. Horsfall, and N. D. Goodman
2019. Pyro: Deep universal probabilistic programming. *Journal of Machine Learning Research*, 20(1):973–978.

Blum, M. G.
2018. Regression approaches for abc. In *Handbook of Approximate Bayesian Computation*, chapter 3. CRC Press, Taylor & Francis Group.

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

Breiman, L.
2001. Random forests. *Machine Learning*, 45(1):5–32.

Carpenter, B., A. Gelman, M. Hoffman, D. Lee, B. Goodrich, M. Betancourt, M. Brubaker, J. Guo, P. Li, and A. Riddell
2017. Stan: A probabilistic programming language. *Journal of Statistical Software, Articles*, 76(1):1–32.

Chwialkowski, K., H. Strathmann, and A. Gretton
2016. A kernel test of goodness of fit. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, Pp. 2606–2615. PMLR.

Collin, F.-D., A. Estoup, J.-M. Marin, and L. Raynal
2020. Bringing abc inference to the machine learning realm: Abcranger, an optimized random forests library for abc. In *JOBIM 2020*, volume 2020.

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

Dalmasso, N., A. B. Lee, R. Izbicki, T. Pospisil, and C.-A. Lin
2020. Validation of approximate likelihood and emulator models for computationally intensive simulations. In *Proceedings of The 23rd International Conference on Artificial Intelligence and Statistics (AISTATS).*

De Nicolao, G., G. Sparacino, and C. Cobelli
1997. Nonparametric input estimation in physiological systems: problems, methods, and case studies. *Automatica*, 33(5).

Drovandi, C. C., C. Grazian, K. Mengersen, and C. Robert
2018. Approximating the likelihood in approximate bayesian computation. In *Handbook of Approximate Bayesian Computation*, S. Sisson, Y. Fan, and M. Beaumont, eds., chapter 12. CRC Press, Taylor & Francis Group.

Durkan, C., A. Bekasov, I. Murray, and G. Papamakarios
2019. Neural spline flows. In *Advances in Neural Information Processing Systems*, Pp. 7509–7520. Curran Associates, Inc.

Durkan, C., I. Murray, and G. Papamakarios
2020. On contrastive learning for likelihood-free inference. In *Proceedings of the 36th International Conference on Machine Learning*, volume 98 of *Proceedings of Machine Learning Research*. PMLR.

Durkan, C., G. Papamakarios, and I. Murray
2018. Sequential neural methods for likelihood-free inference. *Bayesian Deep Learning Workshop at Neural Information Processing Systems.*

Dutta, R., J. Corander, S. Kaski, and M. U. Gutmann
2016. Likelihood-free inference by ratio estimation. *arXiv preprint arXiv:1611.10242.*

Dutta, R., M. Schoengens, J.-P. Onnela, and A. Mira
2017. Abcpy: A user-friendly, extensible, and parallel library for approximate bayesian computation. In *Proceedings of the Platform for Advanced Scientific Computing Conference*, PASC '17.

Gelman, A., J. B. Carlin, H. S. Stern, and D. B. Rubin
2004. *Bayesian Data Analysis*, 2nd ed. edition. Chapman and Hall/CRC.

Gonçalves, P. J., J.-M. Lueckmann, M. Deistler, M. Nonnenmacher, K. Öcal, G. Bassetto, C. Chintaluri, W. F. Podlaski, S. A. Haddad, T. P. Vogels, D. S. Greenberg, and J. H. Macke
2020. Training deep neural density estimators to identify mechanistic models of neural dynamics. *eLife*.

Greenberg, D., M. Nonnenmacher, and J. Macke
2019. Automatic posterior transformation for likelihood-free inference. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, Pp. 2404–2414. PMLR.

Gretton, A., K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola
2012. A kernel two-sample test. *The Journal of Machine Learning Research*, 13(Mar):723–773.

Gutmann, M. U. and J. Corander
2016. Bayesian optimization for likelihood-free inference of simulator-based statistical models. *The Journal of Machine Learning Research*, 17(1):4256–4302.

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

Hermans, J.
2019. Hypothesis. `https://github.com/montefiore-ai/hypothesis`.

Hermans, J., V. Begy, and G. Louppe
2020. Likelihood-free mcmc with approximate likelihood ratios. In *Proceedings of the 37th International Conference on Machine Learning*, volume 98 of *Proceedings of Machine Learning Research*. PMLR.

Hogg, D. W. and D. Foreman-Mackey
2018. Data analysis recipes: Using markov chain monte carlo. *The Astrophysical Journal Supplement Series*, 236(1):11.

Ishida, E., S. Vitenti, M. Penna-Lima, J. Cisewski, R. de Souza, A. Trindade, E. Cameron, V. Busti, C. collaboration, et al.
2015. Cosmoabc: likelihood-free inference via population monte carlo approximate bayesian computation. *Astronomy and Computing*, 13:1–11.

Izbicki, R., A. Lee, and C. Schafer
2014. High-dimensional density ratio estimation with extensions to approximate likelihood computation. In *Artificial Intelligence and Statistics*, Pp. 420–429.

Jennings, E. and M. Madigan
2017. astroabc: an approximate bayesian computation sequential monte carlo sampler for cosmological parameter estimation. *Astronomy and computing*, 19:16–22.

Kermack, W. O. and A. G. McKendrick
1927. A contribution to the mathematical theory of epidemics. *Proceedings of the royal society of London. Series A, Containing papers of a mathematical and physical character*, 115(772):700–721.

Kingma, D. P. and J. Ba
2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations, ICLR.*

Klinger, E., D. Rickert, and J. Hasenauer
2018. pyabc: distributed, likelihood-free inference. *Bioinformatics*, 34(20):3591–3593.

Kousathanas, A., P. Duchen, and D. Wegmann
2018. A guide to general-purpose abc software. In *Handbook of Approximate Bayesian Computation*, chapter 13. CRC Press, Taylor & Francis Group.

Lintusaari, J., H. Vuollekoski, A. Kangasrääsiö, K. Skytén, M. Järvenpää, P. Marttinen, M. U. Gutmann, A. Vehtari, J. Corander, and S. Kaski
2018. Elfi: engine for likelihood-free inference. *The Journal of Machine Learning Research*, 19(1):643–649.

Liu, F., W. Xu, J. Lu, G. Zhang, A. Gretton, and D. J. Sutherland
2020. Learning deep kernels for non-parametric two-sample tests. In *Proceedings of the 37th International Conference on Machine Learning*, volume 98 of *Proceedings of Machine Learning Research*. PMLR.

Liu, Q., J. Lee, and M. Jordan
2016. A kernelized stein discrepancy for goodness-of-fit tests. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, Pp. 276–284. PMLR.

Lopez-Paz, D. and M. Oquab
2017. Revisiting classifier two-sample tests. In *5th International Conference on Learning Representations, ICLR*.

Lotka, A. J.
1920. Analytical note on certain rhythmic relations in organic systems. *Proceedings of the National Academy of Sciences*, 6(7):410–415.

Louppe, G., K. Cranmer, and J. Pavez
2016. carl: a likelihood-free inference toolbox. *Journal of Open Source Software*, 1(1):11.

Lueckmann, J.-M., G. Bassetto, T. Karaletsos, and J. H. Macke
2019. Likelihood-free inference with emulator networks. In *Proceedings of The 1st Symposium on Advances in Approximate Bayesian Inference*, volume 96 of *Proceedings of Machine Learning Research*, Pp. 32–53. PMLR.

Lueckmann, J.-M., P. J. Goncalves, G. Bassetto, K. Öcal, M. Nonnenmacher, and J. H. Macke
2017. Flexible statistical inference for mechanistic models of neural dynamics. In *Advances in Neural Information Processing Systems 30*, Pp. 1289–1299. Curran Associates, Inc.

Marjoram, P. and S. Tavaré
2006. Modern computational approaches for analysing molecular genetic variation data. *Nature Reviews Genetics*, 7(10):759–770.

Martino, L., D. Luengo, and J. Míguez
2018. Accept–reject methods. In *Independent Random Sampling Methods*, Pp. 65–113. Springer.

Neal, R. M.
2003. Slice sampling. *Annals of Statistics*, Pp. 705–741.

Papamakarios, G. and I. Murray
2016. Fast ϵ-free inference of simulation models with bayesian conditional density estimation. In *Advances in Neural Information Processing Systems 29*, Pp. 1028–1036. Curran Associates, Inc.

Papamakarios, G., E. Nalisnick, D. J. Rezende, S. Mohamed, and B. Lakshminarayanan
2019a. Normalizing flows for probabilistic modeling and inference. *arXiv preprint arXiv:1912.02762*.

Papamakarios, G., T. Pavlakou, and I. Murray
2017. Masked autoregressive flow for density estimation. In *Advances in Neural Information Processing Systems 30*, Pp. 2338–2347. Curran Associates, Inc.

Papamakarios, G., D. Sterratt, and I. Murray
2019b. Sequential neural likelihood: Fast likelihood-free inference with autoregressive flows. In *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 89 of *Proceedings of Machine Learning Research*, Pp. 837–848. PMLR.

Paszke, A., S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala
2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, Pp. 8024–8035. Curran Associates, Inc.

Pham, K. C., D. J. Nott, and S. Chaudhuri
2014. A note on approximating abc-mcmc using flexible classifiers. *Stat*, 3(1):218–227.

Prangle, D., M. G. B. Blum, G. Popovic, and S. A. Sisson
2014a. Diagnostic tools of approximate bayesian computation using the coverage property. *Australian & New Zealand Journal of Statistics*, 56(4):309–329.

Prangle, D., P. Fearnhead, M. P. Cox, P. J. Biggs, and N. P. French
2014b. Semi-automatic selection of summary statistics for abc model choice. *Statistical applications in genetics and molecular biology*, 13(1):67–82.

Priddle, J. W., S. A. Sisson, and C. Drovandi
2019. Efficient bayesian synthetic likelihood with whitening transformations. *arXiv preprint arXiv:1909.04857*.

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

Pudlo, P., J.-M. Marin, A. Estoup, J.-M. Cornuet, M. Gautier, and C. P. Robert
2016. Reliable abc model choice via random forests. *Bioinformatics*, 32(6):859–866.

pyABC API Documentation
2020. Multivariate normal transition. `https://pyabc.readthedocs.io/en/latest/api_transition.html#pyabc.transition.MultivariateNormalTransition`. Accessed: 2020-10-20.

Rackauckas, C. and Q. Nie
2017. Differentialequations.jl – a performant and feature-rich ecosystem for solving differential equations in julia. *The Journal of Open Research Software*, 5(1).

Ramdas, A., S. J. Reddi, B. Poczos, A. Singh, and L. Wasserman
2015. On the decreasing power of kernel and distance based nonparametric hypothesis tests in high dimensions. *AAAI Conference on Artificial Intelligence*.

Raynal, L., J.-M. Marin, P. Pudlo, M. Ribatet, C. P. Robert, and A. Estoup
2019. Abc random forests for bayesian parameter inference. *Bioinformatics*, 35(10):1720–1728.

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

Simola, U., J. Cisewski-Kehe, M. U. Gutmann, J. Corander, et al.
2020. Adaptive approximate bayesian computation tolerance selection. *Bayesian Analysis*.

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

Talts, S., M. Betancourt, D. Simpson, A. Vehtari, and A. Gelman
2018. Validating bayesian inference algorithms with simulation-based calibration. *arXiv preprint arXiv:1804.06788*.

Tavaré, S., D. J. Balding, R. C. Griffiths, and P. Donnelly
1997. Inferring coalescence times from dna sequence data. *Genetics*, 145(2).

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

Thomas, O., R. Dutta, J. Corander, S. Kaski, and M. U. Gutmann
2020. Likelihood-free inference by ratio estimation. *Bayesian Analysis*.

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

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

AUTHOR CONTRIBUTIONS

PUBLICATION I: FLEXIBLE STATISTICAL INFERENCE FOR MECHANISTIC MODELS OF NEURAL DYNAMICS

The paper is co-authored by me, Pedro J. Gonçalves, Giacomo Bassetto, Kaan Öcal, Marcel Nonnenmacher, and Jakob H. Macke. I am equally contributing leading author with Pedro J. Gonçalves. The project was jointly conceptualized by me, Pedro J. Gonçalves, and Jakob H. Macke. I proposed end-to-end learning of summary statistics with an RNN, training neural network classifiers to avoid unstable simulation regimes, the fork model, and the algorithm's name (SNPE). I implemented the algorithm and wrote the initial version of the associated software package (`delfi`). All experiments were conducted by me, except those in section 2.2, and I prepared all figures except figure 1A, 1B, and 3. Pedro J. Gonçalves implemented biophysical models, conducted experiments in section 2.2, and prepared figure 1A and 3. Giacomo Bassetto implemented the GLM. Kaan Öcal proposed the converge proof and helped prepare figure 1B. Jakob H. Macke proposed the importance-weighted loss function. I wrote the initial draft of the manuscript with Pedro J. Gonçalves and Jakob H. Macke, which all authors completed and revised.

PUBLICATION II: TRAINING DEEP NEURAL DENSITY ESTIMATORS TO IDENTIFY MECHANISTIC MODELS OF NEURAL DYNAMICS

The paper is co-authored by Pedro J. Gonçalves, me, Michael Deistler, Marcel Nonnenmacher, Kaan Öcal, Giacomo Bassetto, Chaitanya Chintaluri, William F. Podlaski, Sara A. Haddad, Tim P. Vogels, David S. Greenberg, and Jakob H. Macke. I am equally contributing leading author with Pedro J. Gonçalves and Michael Deistler. The project was jointly conceptualized by me, Pedro J. Gonçalves, Michael Deistler, and Jakob H. Macke. I conducted the sub-project on ion channel models in external collaboration with Chaitanya Chintaluri, William F. Podlaski, and Tim P. Vogels, who provided resources and feedback. I wrote the code for amortized inference on ion channel models, conducted the experiments, and prepared associated figures and sections of the manuscripts. I coordinated code and figures across all sub-projects for consistency, and prepared figure 1, showing goal and approach of the project. Pedro J. Gonçalves implemented biophysical models and conducted the sub-project on single-compartment Hodgkin-Huxley models. Michael Deistler conducted the sub-project on the pyloric network for which Sara A. Haddad provided experimental data. Giacomo Bassetto conducted the sub-project on linear-nonlinear (LN) encoding models and Marcel Nonnenmacher the sub-project on receptive field models. I wrote the initial draft of the manuscript with Pedro J. Gonçalves, Michael Deistler, David S. Greenberg, and Jakob H. Macke, which all authors revised.

PUBLICATION III: LIKELIHOOD-FREE INFERENCE WITH EMULATOR NETWORKS

The paper is co-authored by me, Giacomo Bassetto, Theofanis Karaletsos, and Jakob H. Macke, who jointly conceived the project. As the leading author of the paper, I developed the method and associated experiments with feedback from the co-authors. I wrote all code, conducted all experiments, performed all analyses, and prepared all figures. Giacomo Bassetto derived the approximations for the MaxMI acquisition rule. I wrote the initial draft of the manuscript, which all authors completed and revised.

PUBLICATION IV: BENCHMARKING SIMULATION-BASED INFERENCE

The paper is co-authored by me, Jan Boelts, David S. Greenberg, Pedro J. Gonçalves, and Jakob H. Macke. As the leading author of the paper, I conceived and developed the benchmark with feedback from the co-authors. I wrote all code except for the ABC algorithms, which were implemented by Jan Boelts. I conducted all experiments, performed all analyses, and prepared all figures and the accompanying interactive website. I wrote the initial draft of the manuscript, which all authors completed and revised.