

Advancing Methods and Applicability of Simulation-Based Inference in Neuroscience

Dissertation

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

vorgelegt von
Jan Bölts
aus Fischerhude

Tübingen
2023

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

Tag der mündlichen Qualifikation: 10.05.2023

Dekan:

Prof. Dr. Thilo Stehle

1. Berichterstatter:

Prof. Dr. Jakob H. Macke

2. Berichterstatterin:

Prof. Dr. Anna Levina (Martius)

Abstract

The use of computer simulations as models of real-world phenomena plays an increasingly important role in science and engineering. Such models allow us to build hypotheses about the processes underlying a phenomenon and to test them, e.g., by simulating synthetic data from the model and comparing it to observed data. A key challenge in this approach is to find those model configurations that reproduce the observed data. Bayesian statistical inference provides a principled way to address this challenge, allowing us to infer multiple suitable model configurations and quantify uncertainty. However, classical Bayesian inference methods typically require access to the model's likelihood function and thus cannot be applied to many commonly used scientific simulators. With the increase in available computational resources and the advent of neural network-based machine learning methods, an alternative approach has recently emerged: simulation-based inference (SBI). SBI enables Bayesian parameter inference but only requires access to simulations from the model.

Several SBI methods have been developed and applied to individual inference problems in various fields, including computational neuroscience. Yet, many problems in these fields remain beyond the reach of current SBI methods. In addition, while there are many new SBI methods, there are no general guidelines for applying them to new inference problems, hindering their adoption by practitioners.

In this thesis, I want to address these challenges by (a) advancing SBI methods for two particular problems in computational neuroscience and (b) improving the general applicability of SBI methods through accessible guidelines and software tools. In my first project, I focus on the use of SBI in cognitive neuroscience by developing an SBI method designed explicitly for computational models used in decision-making research. By building on recent advances in probabilistic machine learning, this new method is substantially more efficient than previous methods, allowing researchers to perform SBI on a broader range of decision-making models. In a second project, I turn to computational connectomics and show how SBI can help to discover connectivity rules underlying the complex connectivity patterns between neurons in the sensory cortex of the rat. As a third contribution, I help establish a software package to facilitate access to current SBI methods, and I present an overview of the workflow required to apply SBI to new inference problems as part of this thesis.

Taken together, this thesis enriches the arsenal of SBI methods available for models of decision-making, demonstrates the potential of SBI for applications in computational connectomics, and bridges the gap between SBI method development and applicability, fostering scientific discovery in computational neuroscience and beyond.

Zusammenfassung

Der Einsatz von Computermodellen spielt in Wissenschaft und Technik eine immer größere Rolle. Solche Modelle erlauben es, Hypothesen über die einem Forschungsgegenstand zugrundeliegenden Prozesse aufzustellen und diese schrittweise zu verbessern, indem z.B. synthetische Daten aus dem Modell simuliert und mit beobachteten Daten verglichen werden. Allerdings haben Modelle in der Regel unbekannte Parameter. Eine zentrale Herausforderung besteht daher darin, Modellparameter zu finden, die in der Lage sind, die beobachteten Daten zu reproduzieren. Die statistische Methode der Bayes'schen Inferenz bietet eine ideale Lösung für diese Herausforderung: Sie ermöglicht es, viele verschiedene Modellparameter gleichzeitig zu testen, alle geeigneten zu identifizieren und dabei statistische Unsicherheiten zu berücksichtigen. Klassische Methoden der Bayes'schen Inferenz erfordern jedoch den Zugriff auf die sogenannte Likelihood-Funktion des Modells, was für viele gängige wissenschaftliche Modelle nicht möglich ist, da es sich oft um komplexe Computersimulationen handelt. Mit der Zunahme der Rechenressourcen und dem Aufkommen des maschinellen Lernens wurde ein alternativer Ansatz entwickelt, um dieses Problem zu lösen: Simulationsbasierte Inferenz (SBI). SBI verwendet vom Modell simulierte Daten, um Algorithmen des maschinellen Lernens zu trainieren und ermöglicht so Bayes'sche Parameter-Inferenz für komplexe simulationsbasierte wissenschaftliche Modelle.

In den letzten Jahren wurden viele SBI-Methoden entwickelt und auf Inferenzprobleme sowohl in den Neurowissenschaften als auch in vielen anderen Bereichen angewendet. Dennoch gibt es noch offene Herausforderungen: Zum einen bleiben viele Modelle aufgrund ihrer Komplexität außerhalb der Reichweite aktueller SBI-Methoden. Zum anderen mangelt es an zugänglichen Softwaretools und Anleitungen, um SBI-Methoden auf neue Inferenzprobleme anzuwenden.

In meiner Dissertation möchte ich diese Probleme angehen, indem ich einerseits die SBI-Methodik für konkrete Fragestellungen in den Neurowissenschaften verbessere und andererseits die allgemeine Anwendbarkeit von SBI-Methoden durch zugängliche Leitlinien und Softwaretools verbessere. In meinem ersten Projekt beschäftige ich mich mit der Anwendung von SBI in den kognitiven Neurowissenschaften und entwickle eine neue SBI-Methode, die speziell für Modelle der Entscheidungsfindung konzipiert ist. Da diese neue Methode auf den jüngsten Fortschritten im Bereich des maschinellen Lernens basiert, ist sie um ein Vielfaches effizienter als frühere Methoden und kann daher auf ein breiteres Spektrum von Modellen angewendet werden. In einem zweiten Projekt wende ich mich der Konnektomie zu, einem Bereich der Neurowissenschaften, der versucht, die Prinzipien hinter den komplexen Konnektivitätsmustern im Gehirn zu verstehen. Ich zeige, wie SBI dabei helfen kann, Modelle über neue Konnektivitätsregeln im sensorischen Kortex der Ratte zu testen und an die gemessenen Daten anzupassen. Als drittes Projekt präsentiere ich einen Leitfaden für die Anwendung von SBI auf neue Inferenzprobleme, und ich bin einer der Hauptentwickler eines neuen Softwarepakets, das den Zugang zu aktuellen SBI-Methoden erleichtert.

Zusammengenommen wird diese Arbeit den wissenschaftlichen Fortschritt in den Neurowissenschaften und darüber hinaus fördern, indem sie das Arsenal an SBI-Methoden bereichert, das Potential von SBI für die Konnektomie aufzeigt und die Lücke zwischen Entwicklung und Anwendbarkeit von SBI-Methoden im Allgemeinen überbrückt.

Acknowledgements

I would like to thank all those who have supported me throughout my PhD journey. This thesis would not have been possible without their guidance, encouragement, and help.

First and foremost, I would like to thank Prof. Dr. Jakob Macke for his supervision and mentorship over the past five years. I am especially grateful for his support and trust when I was working remotely after the lab moved to Tübingen. I would also like to thank Prof. Dr. Anna Levina for being my second supervisor in Tübingen.

I am incredibly grateful to all my colleagues and peers in the Macke lab for providing a supportive, collaborative, and fun working environment. Special thanks go to Dr. Jan-Matthis Lückmann and Dr. Pedro Gonçalves for supporting me from my start as an intern student in 2017 to the submission of this thesis; to Dr. Álvaro Tejero-Cantero, who taught me perseverance both in research and in ski mountaineering; to Auguste Schulz, Dr. Poornima Ramesh, Dr. Richard Gao, Janne Lappalainen, and Michael Deistler for being there when needed, supporting me with valuable advice and great discussions, and for hosting me so well during my visits to Tübingen—for being such good colleagues and friends.

Many thanks to the following people for their valuable feedback on the thesis: Auguste Schulz, Poornima Ramesh, Paul Pommer, Jan-Matthis Lückmann, Pedro Gonçalves, Michael Deistler, Cornelius Schröder, Sören Becker, and Clara Teusen.

I would also like to thank Franziska Weiler, Ilona Nar-Witte, and Alexandra Petelski for their administrative support in Tübingen and Munich.

Finally, I want to thank my dear friends, family, and partner with all my heart. Their unwavering love, support, and encouragement have carried me through this PhD journey: Bene, Sören, Nico, my chosen family Theresia and Herbert, my parents Meike and Uwe Bölts, and last but not most, Clara.

Contents

1	Introduction	6
2	General Background	9
2.1	Scientific discovery with simulation-based models	9
2.1.1	Simulation-based models	10
2.1.2	Statistical inference	12
2.2	Simulation-based inference	14
2.2.1	Approximate Bayesian computation	15
2.2.2	Neural simulation-based inference	16
3	Simulation-based inference in practice	21
3.1	A-priori checks	21
3.1.1	Model misspecification	21
3.1.2	Prior predictive checks	22
3.2	SBI hyperparameters	23
3.2.1	Choice of SBI method	24
3.2.2	Embedding networks	24
3.2.3	Density estimators	25
3.3	Training and inference	25
3.3.1	Neural network training	26
3.3.2	Inference	26
3.4	A-posteriori checks	28
3.4.1	Posterior predictive checks	28
3.4.2	Simulation-based calibration	28
3.4.3	Posterior analysis	31
4	Publications	34
4.2	Flexible and efficient simulation-based inference for models of decision-making	35
4.3	Simulation-based inference for efficient identification of generative models in computational connectomics	39
4.4	sbi: A toolkit for simulation-based inference	43
5	Conclusion	45
5.1	A simulation-based model is all you need: scientific discovery with SBI	45
5.2	All models are wrong: model misspecification in SBI	46
5.3	Mind the gap: applicability of SBI methods	46
	Appendices	58

Chapter 1

Introduction

Computer simulations play a crucial role in many areas of science and engineering and can have a profound impact on our daily life. The perhaps most drastic and evident influence of simulations on everyone's daily life may be their role during the onset of the covid-19 pandemic in early 2020: By the summer of 2020, we had all become accustomed to regularly checking the forecasts about new infection numbers to prepare for the next interventions. Behind the scenes, it was computer simulations of epidemiological models that enabled these forecasts and greatly influenced the decisions to shut down the economic, cultural, and personal life around the world (Spooner et al. 2021; Lewis 2022). Similarly, in the case of the global climate catastrophe, large-scale computer simulations are used to study the earth's climate system and to simulate the effects of global warming, forming the basis for political and individual actions (Peng et al. 2021; Glavovic et al. 2022). Another great example is the discovery of the Higgs boson: In addition to conducting extensive experiments of colliding particles at CERN¹, researchers built detailed computer simulations of the physical processes underlying particle collisions and used these simulations to confirm the existence of the Higgs boson (Massimi et al. 2015). These are just three among many examples of how computer simulations help tackle emerging global crises and facilitate scientific discovery. But what precisely are computer simulations, and what makes them so useful?

In the scientific context, a computer simulation can be defined as a computer program that simulates a real-world system (Winsberg 2022), e.g., a simulation of a pendulum. One run of this program would simulate the movement of the pendulum and would produce simulated data just like we would observe in an experiment, e.g., the period of the pendulum. Suppose we have measured the period of a real pendulum and want to understand the underlying physical processes. We could build a computer simulation of the pendulum that incorporates all our knowledge as well as hypotheses about the underlying physical processes and aim to reproduce the measured data in the simulation. If we succeed and the simulated data accurately matches the observed data, then we have some evidence in favor of our hypotheses about the pendulum. Unarguably, the simulation will never fully match the underlying real-world system. However, it can still be helpful to improve our mechanistic understanding of the system, motivate new experiments and make forecasts. By repeating this procedure, we can gradually improve the accuracy of the model and eventually arrive at new scientific discoveries (Hartmann 1996; Boge 2020).

Using such *simulation-based models* for scientific discovery comes with a central methodological challenge: The models usually have free parameters that represent our hypotheses or otherwise unknown properties of the underlying process. Therefore, we need reliable methods for identifying

¹European Organization for Nuclear Research

and interpreting these parameters. The overarching goal of my thesis is to address this challenge by improving the methods for identifying unknown parameters in simulation-based models.

A conceptually simple approach to finding suitable model parameters would be trying out many different parameter combinations and selecting those for which the simulated data matches the observed data most closely. However, simulation-based models can have many parameters of interest, and trying out all possible combinations is often not feasible. Furthermore, it is usually not enough to find one best-fitting parameter because there might be multiple parameter settings that reproduce the observed data equally well or because we are uncertain about the exact value of a parameter and rather want to specify a range of possible values.

Among the many techniques that have been developed for identifying model parameters, there is one technique called *Bayesian statistical inference*² that is particularly suited for these challenges: Instead of identifying one best-fitting parameter combination like in the example above, the Bayesian approach identifies an entire probability distribution over parameters that characterizes how likely each parameter is to reproduce the observed data. This probability distribution allows us to efficiently identify all suitable model parameters and study the relations between these alternative solutions, substantially improving the interpretability of the simulation-based approach. Additionally, it quantifies the uncertainty about the parameters, e.g., uncertainty due to the randomness of the underlying process or due to limited information in the observed data. Bayesian inference is thus a conceptually powerful tool for identifying model parameters. Unfortunately, it is based on a mathematical framework that has a specific technical requirement: It requires access to the so-called *likelihood function* of the model. The likelihood function is an analytical expression for the relationship between the observed data and the model parameters, and it is essential for identifying the distribution over suitable model parameters (see [General Background](#) for details). The problem is that most simulation-based models cannot be expressed in mathematical terms because they are defined as—potentially complex—computer simulations. Their likelihood function is thus only *implicitly* defined by the computer simulation and often cannot be accessed efficiently. As a consequence, standard Bayesian inference methods cannot be applied to most simulation-based models. However, several alternative approaches have been proposed that allow performing Bayesian inference by using only simulations from the model, i.e., without having to access the likelihood. These methods are collectively referred to as *simulation-based inference* and are the focus of this thesis.

In simulation-based inference (SBI, Cranmer et al. 2020) the idea is to generate many different simulations from the model based on different candidate parameter combinations and to use this large set of simulated data to approximate the desired probability distribution over suitable model parameters. While the original idea for this simulation-based approach dates back to Diggle et al. 1984, it gained momentum only much later when advances in computing resources made it possible to apply it to practically relevant simulation-based models (Beaumont et al. 2002; Sunnåker et al. 2013; Sisson et al. 2018)³. More recently, the advent of artificial neural networks and deep learning (LeCun et al. 2015) led to a new wave of SBI methods. These so-called *neural* SBI methods leverage the new abilities of artificial neural networks to approximate probability distributions (Germain et al. 2015; Mohamed et al. 2017): instead of accessing the likelihood function of the model, one simulates many different data sets from the model using different parameter combinations and uses them to train an artificial neural network to perform Bayesian inference, i.e., let the neural network predict the distribution over parameters likely to reproduce the observed data.

²After Thomas Bayes, an English statistician, philosopher, and pastor in the 18th century

³These *classical* SBI methods are often referred to as Approximate Bayesian Computation (ABC), see below for details.

Over the past few years, a considerable arsenal of SBI methods has been developed and successfully applied across many disciplines: from particle physics (Brehmer et al. 2020) to genetics (Beaumont 2010), evolutionary biology (Csilléry et al. 2010) and neuroscience (Gonçalves et al. 2020; Groschner et al. 2022; Hashemi et al. 2022), to robotics (Muratore et al. 2022), epidemiology (Witt et al. 2020; Arnst et al. 2022), economics (Dyer et al. 2022) and astrophysics (Dax et al. 2021; Mishra-Sharma et al. 2022). Nevertheless, SBI methods are facing several open challenges. There are still many simulation-based modeling problems where current SBI methods struggle, e.g., for complex models with many parameters. Thus, there is a need for new methods that push the limits of current SBI methods (Cranmer et al. 2020). At the same time, there are research questions in various fields that could be addressed with currently available SBI methods but have not yet been addressed. One reason for this gap in applicability is that the new developments and capabilities of SBI methods are relatively recent and have not yet been introduced in some research areas. In addition, there is a need for established software tools, standard guidelines, or tutorials on how to apply and evaluate SBI in practice to facilitate its application by practitioners.

In this thesis, I contribute three publications that address these challenges, focusing on applications in computational neuroscience. In the first publication, I address the need for new methods by developing an SBI method designed explicitly for inference problems encountered in decision-making research. When studying decision-making, one often uses experimental setups where subjects perform simple decision-making tasks with many repetitions and records their reaction times and choices. However, current SBI methods struggle with this particular experimental setup and the type of models used. By combining recent advances in machine learning techniques, we developed a new SBI method that is tailored to decision-making models and substantially more flexible and efficient than previous methods. Using this new method, researchers will be able to apply SBI to a broader range of computational models in decision-making research.

The second project demonstrates how SBI can help solve parameter inference problems in connectomics. In connectomics, researchers study the connectivity of the neurons that form the basis of our sensory and cognitive abilities. One particular goal in connectomics is to discover general principles underlying the connectivity patterns of neural networks, e.g., by finding so-called wiring rules. One way to test new wiring rules is by simulating them in a computational model and aiming to reproduce measured connectivity data—an ideal scenario to perform Bayesian inference using SBI. In our publication, we show how to apply SBI to inference problems in connectomics using the example of identifying the parameters of wiring rules simulated in the sensory cortex of the rat. Here, we put particular focus on demonstrating how to evaluate, analyze and interpret SBI to facilitate its application by practitioners across the field. We thereby set the stage for using SBI to infer wiring rules in the recently acquired dense reconstructions of human brain tissue (Shapson-Coe et al. 2021).

The third publication refers to a collaborative project where we address the general applicability of SBI methods. We develop a software package that gives access to the main machine-learning-based SBI algorithms currently available and includes detailed documentation and guidelines to facilitate the application of SBI by non-experts. As part of this thesis, I continue this endeavor by presenting a general workflow for applying SBI to new inference problems.

The remainder of the thesis is organized as follows. In the **General Background** chapter, I give a detailed introduction to using statistical inference for making scientific discoveries with simulation-based models. The **Simulation-based inference in practice** chapter contains the workflow applying SBI to new inference problems. In the **Publications** chapter, I give an overview and summaries of the three publications. The **Conclusion** chapter provides concluding remarks.

Chapter 2

General Background

This chapter provides the general background of the work presented in this thesis. First, I present the mathematical framework necessary for the use of simulation-based models in scientific discovery. In the second part of the chapter, I give an overview of the main simulation-based inference algorithms.

2.1 Scientific discovery with simulation-based models

The approach of using models to guide scientific discovery was outlined in chapter 1 as follows. We start with conducting experiments or making observations about a phenomenon in the world. We then use our knowledge and hypotheses derived from observations to build a model that mimics the processes underlying the observation and can generate synthetic data. Finally, we seek to identify the parameters of the model such that the synthetic data closely matches the observed data. If we succeed in building a model capable of reproducing the observed data, we draw conclusions and make predictions for future experiments. To perform these steps systematically and rigorously, we formulate them in the language of mathematics and statistics.

Given that the world and our experimental techniques are inherently stochastic, we assume that the observed data are stochastic as well, i.e., we describe it by a random variable X defined on a measurable space \mathcal{X} . We further assume that there is a latent unknown data-generating process, π_X , which describes the probability distribution of X and can be modeled mathematically as a probability measure (Betancourt 2015)

$$\pi_X : \mathcal{X} \rightarrow [0, 1]. \quad (2.1)$$

We aim to build a model π_m that approximates the unknown data-generating process π_X . However, the space P representing all data-generating processes that could potentially explain the observed data is enormous. Thus, it is impossible to explore P entirely, and we usually have to restrict our search to a set of candidate models $\mathcal{M} \subseteq P$. In formal terms, we aim to find a model $\pi_m \in \mathcal{M}$ in a set of candidate models \mathcal{M}

$$\pi_m : \mathcal{X} \rightarrow [0, 1], \quad (2.2)$$

that is as close as possible to $\pi_X \in P$ (see Fig. 2.1 for an illustration). As the set of candidate models \mathcal{M} is a subset of P , it may or may not contain the true data-generating process π_X (Fig. 2.1 left versus right). In practice, the latter is most likely. Thus, one could argue that all models we



Figure 2.1: When building models to approximate a data-generating process π_X , we need to select a set of candidate models from the vast space of possible processes P . While in theory, the selected set of candidate models \mathcal{M} may (left) or may not (right) contain the true π_X , in practice, the second case is much more likely. However, we hope to find a model that approximates π_X well enough to draw instructive conclusions. Figure adapted from Betancourt 2015.

build will be wrong (Box et al. 1986). However, some of them will still be useful. By carefully choosing M and by performing *statistical inference* to find a model π_m as close as possible to π_X , we might be able to learn something about X and make predictions to motivate new experiments.

In the remainder of this section, I will first give a more concrete overview of the type of models used in practice and then introduce *statistical inference* as the toolkit enabling scientific discoveries with these models.

2.1.1 Simulation-based models

The candidate models $\pi_m \in \mathcal{M}$ used to study the data-generating process can take various forms, from purely mathematical models for which one can use pen and paper to derive the model parameters that best explain the observed data to complex computer simulations for which no analytical expressions are available. However, they all have in common that one can simulate data from them and can thus jointly be defined as *simulation-based models*. Defined as general as possible, a simulation-based model is a computer program f capable of generating simulated data x given a set of parameters θ :

$$x = f(\theta). \quad (2.3)$$

Above, we assumed that the observed data x_o is a realization of a random variable X . Accordingly, $f(\theta)$ is usually stochastic, e.g., repeatedly simulating data x using the same parameters θ would result in different outcomes. If we further assume that the parameter θ is also a random variable, the simulation-based model defines a conditional probability distribution $p(x|\theta)$ and simulating data corresponds to sampling from the distribution:

$$x = f(\theta) \implies x \sim p(x|\theta). \quad (2.4)$$

In the following, I will consider the *drift-diffusion model* (DDM, Ratcliff 1978; Smith et al. 2004, Fig. 2.2) as a working example of a simulation-based model. The DDM is often used in cognitive neuroscience to model data recorded in decision-making experiments. In a classical decision-making experiment, subjects perform simple decision tasks based on sensory evidence, e.g., deciding whether a cloud of dots shown on a screen contains more rightward or leftward movements (Fig. 2.2a). The recorded data usually consists of the reaction times and choices of

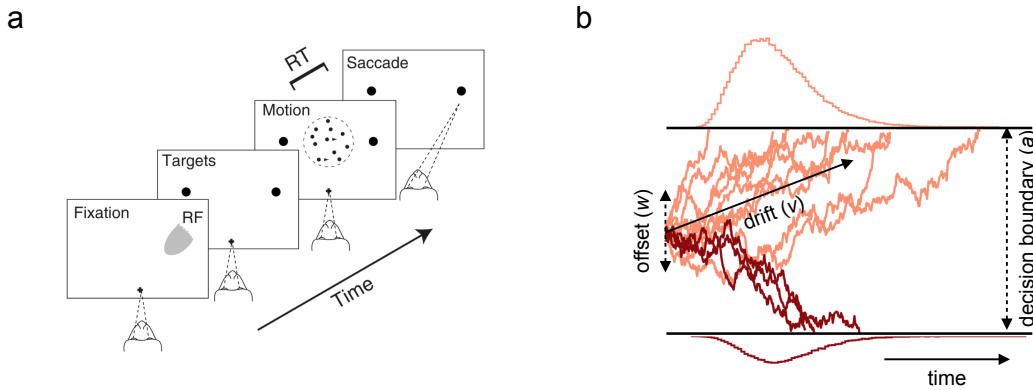


Figure 2.2: Studying perceptual decision-making using the drift-diffusion model. (a) A typical perceptual decision-making experiment: The subject fixates on a screen, observes a cloud of moving dots, and has to indicate the perceived direction of motion of the dots through a saccade, resulting in a recorded choice and reaction time (RT); adapted from Shadlen et al. 2013. (b) The recorded choices and reaction times can be modeled using a drift-diffusion model (DDM). The DDM simulates the internal decision variable as a stochastic process that integrates sensory evidence (drift) over time until one of two decision boundaries is crossed. The resulting reaction time distributions (colored histograms at the top and bottom) have been shown to match measured ones (Ratcliff 1978).

many repetitions of this task with varying difficulty. It is commonly assumed that we base our decisions during such a task on a continuous stream of sensory evidence (e.g., the dot movement), which we integrate over time until enough evidence has accumulated (e.g., the subject decides for one direction). Additionally, it is assumed that the integration of evidence is influenced by internal and external fluctuations (e.g., the subject’s visual attention or task difficulty). The DDM represents these two assumptions mathematically using two terms: a drift term representing the strength of evidence and a diffusion term representing the random fluctuations (Fig. 2.2b). The evolution of the decision variable over time t can be characterized by a stochastic differential equation

$$dX = v dt + dW, \quad X(0) = w, \quad (2.5)$$

where v is the drift, W is a Wiener noise process and w is the initial offset of the decision variable. Using this differential equation, one can simulate reaction times and choices by recording the time when the decision variable X crosses the upper or lower decision boundary given by $\pm a$. In the form presented here, the DDM has three parameters: the drift v , the boundary separation a , and the initial offset w . Simulating choices c and reaction times r given the parameters $\theta = [v, a, w]$ via equation 2.5 implicitly corresponds to sampling

$$[r, c] \sim p(x|\theta). \quad (2.6)$$

As such, the DDM is a relatively coarse model of decision-making in the brain, e.g., it does not explicitly model the underlying biological processes. Nevertheless, we can use it to analyze choices and reaction times recorded in decision-making experiments¹. With the broad definition of a simulation-based model at hand, we now turn to the toolkit of *statistical inference*, which enables us to infer model parameters given observed data.

¹Indeed, it was shown that the DDM can explain the choice and response time behavior in many tasks and species, which motivated further experiments and provided the ground for studying the neural mechanisms of decision making (Roitman et al. 2002; Gold et al. 2007; Shadlen et al. 2013; Latimer et al. 2015).

2.1.2 Statistical inference

Continuing with the DDM as the working example: Imagine we conduct a decision-making experiment, e.g., let a subject perform a series of decisions based on sensory evidence and record their reaction times and choices. We decide to model the decision-making process using the DDM. How can we tune the parameters of the DDM such that it reproduces the observed data so that we can draw conclusions about the underlying processes?

Two conceptually different statistical inference frameworks are available to solve this task: *frequentist inference* and *Bayesian inference*. The two approaches differ in their definition of *probability* itself. In the *frequentist inference* framework, probabilities are defined only through the frequency of a repeatable event (Casella et al. 2007), e.g., for events where uncertainty is due to randomness, as in the decision-making task. In contrast, the *Bayesian inference* framework defines probabilities more broadly as anything subject to uncertainty, be it uncertainty in the data due to randomness (*aleatoric uncertainty*) or uncertainty in the parameters due to lack of knowledge (*epistemic uncertainty*, Fishburn 1994).

To simplify the notation, from here on, we assume that the set of candidate models \mathcal{M} is given by a particular model p_θ with parameters θ where each parameter combination corresponds to a different candidate model. Data generated from the model is denoted as x and the observed data as x_o .

Frequentist inference

By definition of probability in the frequentist setting, we can treat only the data x as stochastic because only the data-generating process is a repeatable event. Thus, the model parameters are deterministic, and we must assume that there exists one specific set of parameters θ^* , which best explains the observed data x_o . How do we find this one model configuration?

A common approach to finding the best-fitting model parameters is *maximum likelihood estimation* (MLE), based on the following idea. The model $p_\theta(x)$ of the data-generating process defines, for any given parameter setting θ , a probability distribution over x . Therefore, we can assign the observed data x_o a probability under this distribution: $p_\theta(X = x_o)$. However, θ is not known. Thus, to find the best-fitting θ , we define the so-called *likelihood function*

$$\mathcal{L}(\theta; x) = p_\theta(X = x_o), \quad (2.7)$$

and maximize it with respect to θ such that we obtain a model configuration under which the observed data x_o is most likely. Importantly, the likelihood function $\mathcal{L}(\theta; x)$ is a function over θ and does not define a probability distribution over θ .

Performing MLE then amounts to maximizing the likelihood function with respect to parameters to obtain the best-fitting parameter θ^* given x_o :

$$\theta_{MLE}^* = \arg \max \mathcal{L}(\theta; x_o) \quad (2.8)$$

$$\frac{\partial \mathcal{L}}{\partial \theta} \stackrel{!}{=} 0. \quad (2.9)$$

The standard way to solve this optimization problem is to obtain the gradient of the likelihood function (or the logarithm thereof) and use calculus or numerical optimization techniques to find the maximum. For example, for the DDM, we would obtain the likelihood function given the reaction times and choices observed in the experiment and obtain θ_{MLE}^* by using numerical optimization techniques to maximize it with respect to θ .

Statistical inference using the MLE approach is a widely adopted technique. Yet, it has substantial limitations. The major limitation of the frequentist inference approach is that it is hard to quantify the uncertainty in the parameter estimate because one cannot assign the notion of probability to the parameters (Wagenmakers et al. 2008). However, uncertainty about the parameter estimate is inherent to any inference problem. First, there might be uncertainty due to the intrinsic randomness in the data-generating process. Second, there might be uncertainty due to the limited amount of information available in the observed data. Furthermore, several parameter settings in the model might reproduce the data equally well due to parameter degeneracies or parameter interaction and compensation mechanisms in the model. The *Bayesian inference framework* offers a more principled approach to these challenges.

Bayesian inference

Bayesian inference takes a probabilistic view of the data and the model parameters, i.e., in addition to treating the data as a random variable X , it assumes that the model parameters are also random variables. Following this assumption, the parametrized model p_θ defines both a *likelihood function* $L(\theta; x)$ and conditional probability distribution $p(x|\theta)$ often referred to as *likelihood*.

$$\mathcal{L}(\theta; x) = p_\theta(x|\theta). \quad (2.10)$$

When $c = x_o$ is fixed, equation 2.10 corresponds to the likelihood function, i.e., a function in θ . However, if θ is fixed, equation 2.10 is a conditional probability distribution in x conditioned on θ .

According to the probabilistic view on the parameters, Bayesian inference allows us to incorporate prior knowledge about the parameters into the inference process by defining a *prior distribution* $p(\theta)$. The goal of Bayesian inference is then to combine the likelihood $p(x|\theta)$ and the prior $p(\theta)$ to infer the *posterior distribution* over the model parameters conditioned on the data $p(\theta|x)$. The posterior relates to the likelihood and the prior according to Bayes' rule (Bayes 1763):

$$p(\theta|x) = \frac{p(x|\theta)p(\theta)}{p(x)}. \quad (2.11)$$

Here, $p(x)$ refers to the so-called *evidence*, which defines the probability distribution of the data implied by the assumed model, $p(x) = \int p(x|\theta)p(\theta)d\theta$. The posterior distribution $p(\theta|x)$ resulting from Bayesian inference is a probability distribution over the parameters. The fact that we obtain a probability distribution rather than a point estimate θ_{MLE} as in the frequentist approach, makes it possible to quantify uncertainty about the inferred parameters.

When applying Bayesian inference to infer the parameters of the DDM given the observed reaction times and choices, we would first define a prior distribution over the three parameters $\theta = [v, a, w]^T$ taking into account our knowledge about the model, e.g., that the boundary parameter must be positive $a > 0$ and that the offset cannot be larger than the boundary, $w \in [0, a]$. Subsequently, we would aim to obtain the posterior distribution $p(\theta|x_o)$ following Bayes' rule. We could estimate the mode of this posterior to obtain the most likely parameter values for v , a , and w . However, importantly, we could also obtain an estimate of the uncertainty in the parameters by inspecting the posterior variances. For example, there would be epistemic uncertainty due to a lack of information in the observed data, which would decrease as we increase the number of observed reaction times and choices, and aleatoric uncertainty due to the randomness of the DDM itself, which we cannot change.

Solving Bayes' rule Overall, the Bayesian inference framework provides an intuitive and conceptually powerful way of performing statistical inference while accounting for uncertainty. However, its major drawback is that it can be challenging to solve Bayes' rule to obtain the posterior. Whether Bayes' rule is solvable depends on the choice of the model and the prior. There are three scenarios:

1. The model is a member of the exponential family (Koopman 1936), and the prior is the corresponding conjugate prior. Then the posterior is available analytically, e.g., when using a Bernoulli distribution as a model and the corresponding conjugate Beta distribution prior, the posterior will also follow a Beta distribution.
2. We use models for which we have access to the underlying likelihood function, e.g., it is possible to evaluate $\mathcal{L}(\theta; x)$ efficiently, as in the DDM example. In these cases, Bayes' rule is not solvable analytically (because the evidence $p(x)$ is intractable). However, several reliable methods exist for approximating the posterior with high accuracy. For example, Markov Chain Monte Carlo sampling (MCMC, see below, Metropolis et al. 1953; Hastings 1970; Hogg et al. 2018) allows us to obtain posterior samples, and variational inference approaches can obtain parametric approximations to the posterior (Blei et al. 2017).
3. All remaining cases, i.e., cases where we use entirely *simulation-based models* for which we do not have access to the underlying likelihood function. In these cases, standard Bayesian inference methods cannot be applied.

Various approaches have been developed to perform Bayesian inference for the third scenario of entirely simulation-based models. These developments started as early as 1984 (Rubin 1984) but were limited to relatively simple and low-dimensional models for a long time. Only over the last two decades, driven by the advances in computing resources, new sampling algorithms, and the advent of artificial neural networks, so-called *simulation-based inference* approaches enabled the application of Bayesian inference to a much broader range of scientific simulators.

2.2 Simulation-based inference

While simulation-based models do not give access to their underlying likelihood function, they do provide access to simulated data. The central idea of simulation-based inference (SBI) is to use access to simulated data to circumvent the evaluation of the likelihood required for standard Bayesian inference methods (Rubin 1984). Thus, the goal of SBI can be summarized as follows. Given observed data x_o , a simulation-based model $f(\theta)$ and prior $p(\theta)$, it aims to approximate the posterior $p(\theta|x_o)$ using only data x generated from the simulator with parameters sampled from the prior.

As simulation-based models are usually time-consuming to simulate, an important constraint of SBI is to consume as little simulated data as possible. Thus, the various SBI approaches differ mainly in how efficiently they use the simulated data to approximate the posterior. They can be separated into two classes. Classical SBI approaches based on rejection sampling, also known as *Approximate Bayesian Computation* (ABC, Sunnåker et al. 2013; Sisson et al. 2018), and more recent SBI approaches based on artificial neural networks, here referred to as *neural* simulation-based inference (Cranmer et al. 2020).

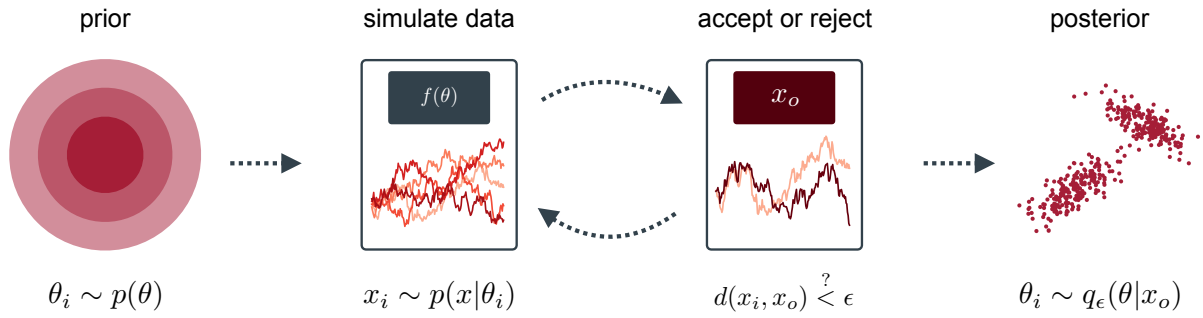


Figure 2.3: Approximate Bayesian Computation is based on the principle of rejection sampling: To approximate the posterior distribution, one aims to find those model parameters θ_i that correspond to simulated data x_i closely resembling the observed data x_o . To that end, one samples candidates θ_i from a prior distribution and generates corresponding simulated data x_i from the simulator (left). If x_i is close to x_o according to a given distance function d and criterion ϵ , one accepts θ_i ; otherwise, one rejects it (middle). The accepted parameters θ_i correspond to samples from the approximate posterior (right).

2.2.1 Approximate Bayesian computation

Approximate Bayesian Computation (ABC) refers to SBI approaches that rely on rejection sampling. The general principle of rejection sampling in ABC is to approximate the posterior by collecting model parameters that result in simulated data x resembling the observed data x_o . To that end, one defines a distance function between the simulated and observed data, $d(x, x_o)$, simulates data x_i from the model given a parameter θ_i sampled from the prior and accepts or rejects θ_i using a threshold parameter ϵ on the distance: $d(x_i, x_o) < \epsilon$ (Fig. 2.3).

The distribution of the accepted parameters is then given by

$$p(\theta | d(x, x_o) < \epsilon), \quad (2.12)$$

and will converge to the true posterior distribution in the limit of infinitely many simulated data and as $\epsilon \rightarrow 0$ (Tavaré et al. 1997). The idea of using this algorithm to perform Bayesian inference in simulation-based models dates to Diggle et al. 1984, who used it to approximate the likelihood function (not the posterior) of an intractable model. It took quite some time until this approach gained momentum, i.e., when Tavaré et al. 1997 and Pritchard et al. 1999 proposed to use it for posterior inference in population genetics. Beaumont et al. 2002 later introduced the term Approximate Bayesian Computation (ABC). The algorithmic steps for ABC with rejection sampling are summarized in Algorithm 1.

Classical rejection sampling-based ABC approaches generally suffer from the curse of dimensionality: As the dimensionality of the parameter spaces or the data increases, the number of model simulations required to obtain an accurate posterior estimate increases exponentially. Several sequential variants to the ABC scheme have been proposed to improve the sampling efficiency, inspired mainly by sequential sampling approaches from standard approximate Bayesian inference. The general idea of these variants is to improve the sampling efficiency by repeating it over multiple rounds and re-using the accepted parameters from the previous round as proposals for the current round (Marjoram et al. 2003; Sisson et al. 2007; Del Moral et al. 2012; Toni et al. 2009). Collectively, these sequential ABC approaches are often referred to as ABC-SMC (Toni 2010; Sisson et al. 2018). Another way to improve the accuracy of ABC approaches is to perform a posthoc regression adjustment to the mismatch between the simulated and observed data (Beaumont et al. 2009).

Algorithm 1: Approximate Bayesian computation via rejection sampling as in Beaumont et al. 2002

```

input simulator  $x \sim f(\theta)$ , prior  $p(\theta)$ , observed data  $x_o$ 
distance function  $d$ , rejection threshold  $\epsilon$ , simulation budget  $N$ 
for  $j = 1 : N$  do
  Sample  $\theta_i \sim p(\theta)$ 
  Simulate  $x_i \sim f(\theta_i)$ 
  if  $d(x_i, x_o) < \epsilon$  then
    | accept  $\theta_i$ 
  else
    | reject  $\theta_i$ 
  end
end
return Accepted samples  $\{\theta_i\} \sim \hat{p}(\theta | d(x, x_o) < \epsilon)$ 

```

This approach was later extended to non-linear regression using artificial neural networks by Blum et al. 2010, which can be seen as the root of modern neural network-based SBI approaches.

Although sequential sampling schemes and regression adjustment approaches improve the sampling efficiency, ABC methods often do not scale to high-dimensional problems (Cranmer et al. 2020; Lueckmann et al. 2021) and always require the ad-hoc choice of distance functions, rejection thresholds, and summary statistics. The advent of neural network-based SBI approaches enabled overcoming these limitations.

2.2.2 Neural simulation-based inference

The idea of neural-network-based SBI approaches is to replace the ABC rejection-sampling scheme with neural density estimation. In other words, instead of collecting approximate posterior samples by comparing simulated and observed data, one uses the simulated data as training data for artificial neural networks designed for estimating probability densities (Fig. 2.4). With the recent advances in flexible neural-network-based density estimators, this paradigm has enabled the application of SBI to substantially more challenging applications than before (Cranmer et al. 2020).

SBI based on neural density estimation was introduced by Papamakarios et al. 2016. However, predecessors of this idea can be found in Wood 2010 and Blum et al. 2010: In their “*synthetic likelihood*” approach, Wood 2010 proposed using data simulated from the model to estimate the mean and covariance of a multivariate Gaussian distribution to obtain an approximation of the unknown likelihood, i.e., to perform density estimation. In turn, it was possible to obtain posterior samples via MCMC sampling using the tractable Gaussian likelihood. Independently, Blum et al. 2010 proposed using neural networks to perform a non-linear regression to adjust the mismatch between simulated and observed data in the context of rejection sampling (see above). Crucially, they trained the neural networks on all the simulated data and thereby learned a regression model *conditional* on the data. Consequently, applying the regression model to unseen data was possible without repeating the training. While Blum et al. 2010 used this property to perform regression adjustment in the context of ABC, the idea was an essential ingredient for neural-network-based SBI approaches.

Neural SBI approaches can be separated into three groups, targeting either the posterior distribution directly (neural posterior estimation), the likelihood (neural likelihood estimation), or the

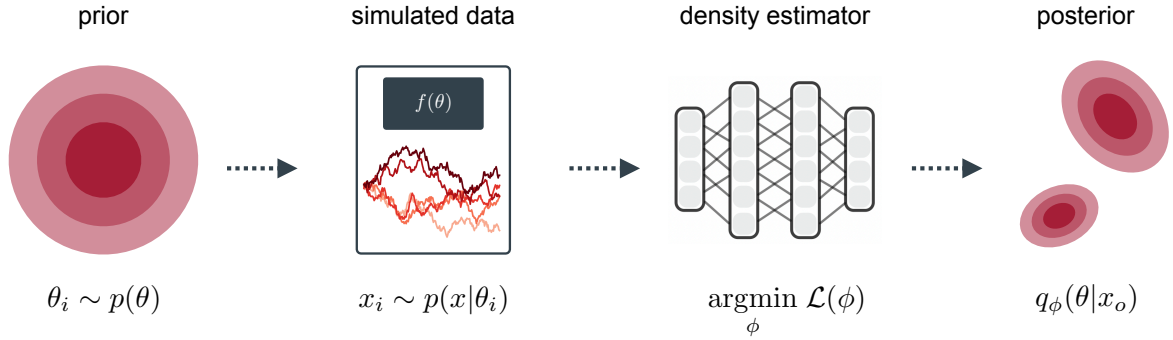


Figure 2.4: Neural simulation-based inference starts with generating simulated data x_i from a simulator $f(\theta)$ using parameters θ_i sampled from the prior. Rather than approximating the posterior with rejection sampling as in classical ABC approaches, it trains an artificial neural network q_{ϕ} to learn a *parametric* approximation of the posterior (or the likelihood, or the likelihood-ratio, see main text) by optimizing a corresponding loss function $\mathcal{L}(\phi)$.

likelihood-ratio (neural ratio estimation).

Neural posterior estimation (NPE)

NPE uses an artificial neural network to perform conditional density estimation to learn a parametric approximation of the posterior. The procedure starts by simulating data x_i from the model using parameters θ_i sampled from the prior distribution $p(\theta)$ to obtain a training data set of pairs $\{(\theta_i, x_i)\}_{i=1}^N$. Subsequently, the training data is used to train a neural network $F(x)$ that takes the data x as input and predicts the parameters ϕ of a density estimator q_{ϕ} . As a density estimator, Papamakarios et al. 2016 chose a mixture density network (MDN, Bishop 1994) given by a mixture of Gaussians. Thus, in their setting, ϕ contained the means, mixture weights, and covariances of the mixture. The density estimator q_{ϕ} then served as a parametric approximation of the unknown posterior distribution,

$$q_{\phi}(\theta|x) \approx p(\theta|x). \quad (2.13)$$

The loss function for optimizing the parameters of the neural network $F(x)$ is the negative log probability of the parameters θ_i under the current estimate $q_{\phi}(\theta|x_i)$:

$$\phi^* = \operatorname{argmin}_{\phi} - \frac{1}{N} \sum_{i=1}^N \log q_{\phi=F(x_i)}(\theta_i|x_i). \quad (2.14)$$

This loss function implicitly minimizes the Kullback-Leibler divergence between the true posterior and the approximation $D_{\text{KL}}(q_{\phi}(\theta|x), p(\theta|x))$, in the expectation of $\theta \sim p(\theta)$, $x \sim p(x|\theta)$ (Paige et al. 2016; Papamakarios et al. 2016; Le et al. 2017). In other words, if the density estimator q_{ϕ} is flexible enough and when taking the limit of infinite training data, NPE will approximate the posterior arbitrarily well. The algorithmic steps of NPE are summarized in Algorithm 2.

A noteworthy property of NPE is that, once the density estimator is trained, it can be applied to any new x without retraining, i.e., the posterior of a newly observed data x_o can be obtained through a single pass through the underlying neural network. This property is commonly referred to as amortization or *amortized inference*: the initially high cost of training the density estimator

Algorithm 2: Single round neural posterior estimation as in Papamakarios et al. 2016

```

input simulator  $x \sim f(\theta)$ , prior  $p(\theta)$ , observed data  $x_o$ , neural density estimator  $q_{\phi=F(x)}$ 
for  $j = 1 : N$  do
  | Sample  $\theta_i \sim p(\theta)$ 
  | Simulate  $x_i \sim f(\theta_i)$ 
end
 $\phi \leftarrow \arg \min 1/N \sum_i^N -\log q_{\phi=F(x_i)}(\theta_i|x_i)$ 
Set  $\hat{p}(\theta|x_o) = q_{\phi=F(x_o)}(\theta|x_o)$ 
return Samples from  $\hat{p}(\theta|x_o)$  and density estimator  $q_{\phi=F(x)}(\theta|x)$ 

```

with a large number of simulated data *amortizes* when performing inference for many different x_o (Gershman et al. 2014; Paige et al. 2016; Papamakarios et al. 2016).

Sequential neural posterior estimation While amortization is a valuable property in some scenarios, there are other scenarios where it is not. For example, when we are interested in only one specific observation x_o , the initial cost of training the density estimator to be accurate across an extensive range of different x might be too high. For this setting, Papamakarios et al. 2016 proposed a Sequential version of NPE (SNPE), which trains the density estimator q_ϕ over multiple rounds. In the first round, it performs NPE with training data simulated with parameters sampled from the prior. Then, in the following rounds, the parameters for simulating new training data are not sampled from the prior but from a proposal distribution focusing on parameters likely to have generated x_o . A good candidate for the proposal distribution is the current estimate of the posterior $q_\phi(\theta|x_o)$. Following this approach, one can focus the training of the density estimator on those regions in the data and parameter spaces relevant for x_o , which can substantially reduce the number of required training simulations.

SNPE tends to be more simulation-efficient than NPE (Lueckmann et al. 2021). However, it comes with an additional algorithmic cost. When we train the density estimators with training data simulated from a proposal distribution $\tilde{p}(\theta)$ different from the prior $p(\theta)$, the resulting posterior will differ from the desired posterior, e.g., it will be the so-called *proposal posterior* $\tilde{p}(\theta|x)$ —the posterior with respect to the proposal prior:

$$\tilde{p}(\theta|x) = p(\theta|x) \frac{\tilde{p}(\theta)p(x)}{p(\theta)\tilde{p}(x)}, \quad (2.15)$$

where $\tilde{p}(x) = \int_\theta \tilde{p}(\theta)p(x|\theta)$. Therefore, Papamakarios et al. 2016 introduced an analytical posthoc correction to obtain the desired posterior corresponding to the actual prior. However, the correction step worked only for specific proposal distributions, e.g., Gaussian or uniform proposals, and it was numerically unstable in some cases. Over the past years, several variants of SNPE have been proposed (Lueckmann et al. 2017; Greenberg et al. 2019; Deistler et al. 2022a) that successively improved its numerical stability and enabled the use of more flexible density estimator like normalizing flows (Rezende et al. 2015; Papamakarios et al. 2021).

Neural likelihood estimation

Neural likelihood estimation (NLE) follows the same principle as NPE, except that it estimates the intractable likelihood $p(x|\theta)$ defined by the simulator and not the posterior $p(\theta|x)$. As outlined

above, the idea of learning such a *synthetic likelihood* was first introduced by Wood 2010, who used model simulations to estimate a Gaussian likelihood. Papamakarios et al. 2019 moved this idea into the realm of highly flexible and conditional density estimators based on neural networks. In NLE, one uses a training data set comprised of pairs $\{(\theta_i, x_i)\}_{i=1}^N$ to train a neural network F to learn the parameters ϕ of a density estimator $q_\phi(x|\theta) \approx p(x|\theta)$. The neural network F is conditional on the parameters, i.e., it takes as an input θ and learns a density in x . Being a density estimator, $q_\phi(x|\theta)$ defines a tractable probability density that can be evaluated and sampled efficiently. Thus, q_ϕ gives access to the intractable likelihood of the simulator and enables obtaining posterior samples by combining the approximate likelihood and the prior and applying MCMC:

$$p(\theta|x_o) \propto q_\phi(\theta|x_o)p(\theta). \quad (2.16)$$

The loss function for optimizing q_ϕ is the negative log probability of q_ϕ given the current training data point, $\mathcal{L}(\phi) = -\log q_{\phi=F(\theta_i)}(x_i|\theta_i)$, which implicitly minimizes $D_{\text{KL}}(q_\phi(x|\theta), p(x|\theta))$.

Like NPE, NLE can be extended to the sequential setting to focus the inference on a particular x_o (Papamakarios et al. 2019). Unlike NPE, the Sequential version of NLE (SNLE) does not require a correction step for using proposal distributions during training. The reason is that the proposal distribution affects only the learning of the likelihood and does not interfere with the inference step in equation 2.16. Thus, it does not affect the posterior estimate. Papamakarios et al. 2019 proposed to sample the parameters for the next round from the current posterior estimate (as in NPE). Lueckmann et al. 2019 extended the SNLE framework to more flexible proposals based on local and global acquisition functions commonly used in the active learning literature.

Neural ratio estimation

A third class of neural-network-based SBI methods is based on the insight that one can recast the approximation of a ratio of two densities as a classification task between samples from the densities (Cranmer et al. 2016). Building on this, neural ratio estimation (NRE) uses neural network classifiers to estimate density ratios to perform SBI (Izbicki et al. 2014; Cranmer et al. 2016; Thomas et al. 2022; Durkan et al. 2020). For illustration, I here focus on a recent neural-network-based approach by Hermans et al. 2020. They trained a neural network classifier to learn the likelihood-to-evidence ratio to be then able to perform inference with MCMC. Their approach builds on the fact that during MCMC sampling, the calculation of the acceptance probability of a newly proposed parameter θ_{new} through the posterior ratio can be expressed in terms of the likelihood-to-evidence ratio $r(x|\theta) = \frac{p(x|\theta)}{p(x)}$:

$$\frac{p(\theta_{new}|x)}{p(\theta_{old}|x)} = \frac{p(\theta_{new})p(x|\theta_{new})/p(x)}{p(\theta_{old})p(x|\theta_{old})/p(x)} = \frac{p(\theta_{new})r(x|\theta_{new})}{p(\theta_{old})r(x|\theta_{old})}. \quad (2.17)$$

Hermans et al. 2020 showed that training a classifier $d(\theta, x)$ to distinguish between dependent parameter-data pairs $(\theta, x) \sim p(x|\theta)p(\theta)$ and independent pairs $(\theta, x) \sim p(x)p(\theta)$ converges to an optimal classifier that recovers the likelihood-to-evidence ratio:

$$d^*(\theta, x) = \frac{p(\theta, x)}{p(\theta, x) + p(x)p(\theta)} \quad (2.18)$$

$$\frac{d^*(\theta, x)}{1 - d^*(\theta, x)} = \frac{p(\theta, x)}{p(x)p(\theta)} = \frac{p(x|\theta)}{p(x)} = r(x|\theta). \quad (2.19)$$

Thus, it is possible to use simulated training data $\{(\theta_i, x_i)\}_{i=1}^N$ for training a classifier $d(\theta, x)$ to approximate $\hat{r}(x|\theta) \approx r(x|\theta)$. Subsequently, one can evaluate the trained classifier on the observed data to obtain $\hat{r}(x_o|\theta)$ and use it as a replacement for the likelihood-to-evidence ratio of the intractable model. Like NLE, NRE can also be extended to a Sequential scheme (SNRE, Hermans et al. 2020; Durkan et al. 2020) by repeating the training and MCMC sampling over multiple rounds. As in SNLE, SNRE does not require a correction step.

Amortization in neural SBI methods Similar to NPE, the non-sequential versions NLE and NRE perform *conditional* density (ratio) estimation: Once trained on simulations sampled from the prior, applying them to a new data point corresponds to a single pass through the neural network. However, unlike NPE, NLE and NRE additionally require MCMC sampling to obtain posterior samples, which can be challenging and computationally costly in high-dimensional parameter spaces. Thus, inference with NLE or NRE is *not fully amortized*, only the training of neural networks is. Recently, Glöckler et al. 2022 showed how the MCMC sampling step in NLE and NRE can be replaced by variational inference, substantially reducing the computational costs.

2.3 Summary

In this chapter, I outlined how we can use simulation-based models for scientific discovery and emphasized the benefits of the Bayesian inference framework for identifying the model parameters given observed data. I then gave an overview of SBI methods that enable approximate Bayesian inference for simulation-based models where standard methods do not apply. Given the approximate nature of these methods and their variety, the question remains how to choose the suitable method in practice and evaluate it in scenarios where we do not have access to a reference solution. In the next chapter, I aim to answer these questions.

Chapter 3

Simulation-based inference in practice

In the previous chapter, I presented three neural SBI approaches that have emerged over the past years. Simulation-based inference is a vivid field of research in which improvements, variants, and new applications of all three approaches appear regularly. These new SBI methods are usually presented by evaluating them on a tractable toy example with access to reference posteriors, followed by an application to an actual SBI problem of the authors' choice. From a practitioner's view, this poses the problem of deciding which algorithm to use for which application and how to tailor it to their specific needs.

There have been efforts to address this problem, e.g., by establishing a standardized set of benchmark tasks with reference posteriors, providing a common ground for studying and comparing available and new SBI algorithms systematically (Lueckmann et al. 2021). However, a general guideline for the choices and evaluation steps involved in applying SBI to real-world problems, e.g., a practitioner's guide to SBI similar to Gelman et al. 2020's guide developed for standard approximate Bayesian inference methods, has yet to be established. In this chapter, I present a first step towards such a guide by giving an overview of the choices in preparing, executing, and analyzing *neural* SBI approaches in practice. I use the drift-diffusion model (DDM, Fig. 2.2) as a running example.

The workflow for using SBI to statistically infer the parameters of a simulation-based model given observed data can be divided into three steps (Fig. 3.1):

1. *A-priori* checks performed before running inference,
2. choosing a suitable SBI method and applying it, and
3. *a-posteriori* checks performed after inference.

3.1 A-priori checks

A-priori checks are performed to make sure that the model and the corresponding prior can capture the observed data, e.g., that the model is not misspecified and the prior is chosen appropriately.

3.1.1 Model misspecification

As outlined in section 2.1, we generally assume that our model will only partially reproduce the aspects of the underlying data-generating process, i.e., it will always be misspecified with respect

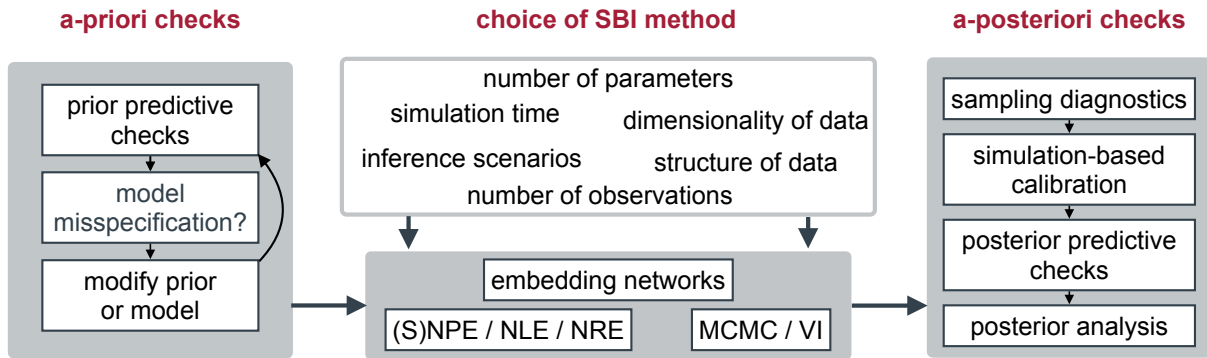


Figure 3.1: The simulation-based inference workflow can be divided into three stages. 1) Before applying SBI, we need to ensure the model is not misspecified, e.g., by performing prior predictive checks and refining the prior or the model (left). 2) When choosing an SBI method, we need to take into account the properties of the data, the model, and the inference problem (center). 3) After applying SBI, we need to validate the posterior using predictive checks and calibration checks before we can analyze it (right).

to the true data-generating process. Nevertheless, we can still build valuable models by focusing on specific features of the observed data we aim to reproduce. However, the model should at least be able to reproduce these features of interest. Suppose this was not the case, and we would still perform SBI. Then we would effectively train the SBI algorithm with training data different from the observed data to which it is applied at inference time. The accuracy of the inference will then depend on how well the SBI algorithm generalizes to out-of-distribution data. However, it is known that especially large neural networks commonly used in neural SBI methods can be highly inaccurate when applied to unseen data (Nalisnick et al. 2022). Indeed, it was shown for neural SBI approaches (Cannon et al. 2022) and also for ABC approaches (Frazier et al. 2019) that in the misspecified scenario, the approximate posterior samples can be highly inaccurate.

Recently, several approaches have been proposed that automatically detect and correct for model misspecification in SBI (Frazier et al. 2019; Frazier et al. 2020; Schmitt et al. 2022; Ward et al. 2022; Kelly et al. 2023). However, these methods are in their early development and often involve additional algorithmic steps. A computationally cheap and easy-to-interpret alternative is given by prior predictive checks.

3.1.2 Prior predictive checks

The reason for model misspecification can either lie in the model or the choice of the prior distribution. So-called *prior predictive checks* are a common way to check whether the prior is well-chosen. A prior can be considered well-chosen if the data resulting from simulating data with parameters sampled from the prior, i.e., *the prior predictive distribution*, is realistic with respect to the problem at hand. If it is not, the prior should be adapted. The prior predictive distribution should also contain the observed data. If it does not, this is a sign of model misspecification; however, this misspecification can be due to the model itself.

To perform prior predictive checks, we define a prior distribution for each parameter in the model we want to infer, taking into account the domain knowledge we have about the inference problem. For example, in the DDM, we would define a prior for the three parameters v , a , and w , choosing uniform priors in the ranges that result in realistic distributions of reaction times and

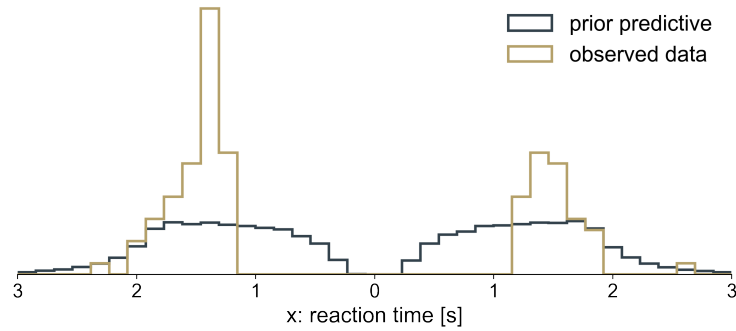


Figure 3.2: Prior predictive check for the drift-diffusion model. Reaction times and choices simulated from the DDM with parameters sampled from the prior (black), compared to the observed data (gold; up and down choices shown to the left and the right, respectively).

choices (Shadlen et al. 2013):

$$v \sim \mathcal{U}(-2, 2) \quad (3.1)$$

$$a \sim \mathcal{U}(0.5, 2) \quad (3.2)$$

$$w \sim \mathcal{U}(0.2, 0.8). \quad (3.3)$$

Subsequently, we generate a large set of simulated data by running the DDM with parameters sampled from the prior, $(\theta, x) \sim p_{DDM}(x|\theta)p(\theta)$, and compare the simulated $\{x_i\}_{i=1}^N$ with the observed data x_o . If the dimensionality of the data allows, the comparison could be visual, e.g., by plotting each component of the observed data on top of histograms of the simulated data. Alternatively, one would need to define summary statistics to reduce the dimensionality, use a distance function to compare high-dimensional data, or use automated approaches as proposed by Schmitt et al. 2022; Ward et al. 2022.

In the DDM, there are only two data dimensions (reaction times and choices); thus, a visual prior predictive check is possible. For example, let the observed data x_o be given by 100 reaction times and choices recorded from one subject in a two-alternative perceptual decision-making task (see Fig. 2.2 and Ratcliff et al. 2008). To perform the prior predictive check, we would sample 100,000 parameters from the prior, simulate them in the DDM, and plot corresponding simulated data on top of the observed data with the two possible choices encoded as the sign of the reaction time (Fig. 3.2). If the distribution of simulated data covers the observed data well (Fig. 3.2), we would consider the DDM well-specified for the observed reaction times and choices and continue with SBI. Note that the distribution of simulated data does not have to resemble the observed data, as we have not performed inference yet.

3.2 SBI hyperparameters

The next step in the SBI workflow is to select the SBI hyperparameters, e.g., the type of SBI method, the neural network architecture, or the type of density estimator. The answer depends on multiple factors, e.g., the dimensionality of the data and parameter spaces, the run time of the simulator, or whether we have one or multiple observed data points (see Fig. 3.1, center).

3.2.1 Choice of SBI method

The three different SBI approaches introduced in section 2.2 solve different learning problems: NPE takes data x as input and estimates a density over parameters θ ; NLE takes parameters θ as input and estimates a density over the data x ; NRE performs classification taking both θ and x as input. Thus, the difficulty of the corresponding optimization problems depends on the dimensionality of x and θ .

Dimensionality of data and parameters If the x is high-dimensional compared to the number of parameters θ , then applying NPE can be a better choice than applying NLE because, for NLE, one would have to learn a high-dimensional density in x . Concurrently, in scenarios where θ is high-dimensional compared to x , performing density (ratio) estimation will likely be easier for NLE or NRE. Yet, in these cases, one still would have to apply MCMC sampling in θ space to obtain posterior samples, which can be challenging for high-dimensional θ . Thus, while the dimensionality of x can guide the choice between NPE and NLE or NRE, high-dimensional parameter spaces can be problematic for either. As a heuristic, current SBI methods require on the order of 10,000 training data points per parameter dimension, e.g., at least 100,000 simulations for a model with ten parameters. However, these numbers strongly depend on the problem (see Gonçalves et al. 2020; Lueckmann et al. 2021; Deistler et al. 2022b; Ramesh et al. 2022, for examples).

Inference scenarios Another factor determining the choice of the SBI method is whether one carries out inference once or repeatedly for multiple observations. Sequential SBI methods can be substantially more simulation efficient by focusing the inference on a particular observation x_o . Thus, they are a good choice in scenarios with a limited simulation budget and a single observed data point x_o of interest. In contrast, in a scenario where one repeatedly performs inference for different observations x_o , fully amortized NPE or partially amortized methods like NLE or NRE are more suitable. In scenarios with multiple observations that can be assumed to be identically and independently distributed (iid), it can be beneficial to use NLE or NRE as they can leverage the fact that iid trials and hierarchical inference settings often can be rewritten in terms of single-trials likelihoods (Hermans et al. 2020; Fengler et al. 2021; Boelts et al. 2022, see [Publications](#) for details).

3.2.2 Embedding networks

Neural SBI methods have the advantage over classical ABC methods in that they can augment the neural density (ratio) estimators with additional neural network layers to automatically learn low-dimensional embeddings from high-dimensional inputs. The embedding networks are trained end-to-end with the neural density estimator and can be applied to x or θ depending on the input to the density estimator. For example, NPE can learn embeddings for x , NLE can learn embeddings for θ , and NRE can learn embeddings for both. The architecture of the embedding network should be selected according to the input type. For example, recurrent neural networks (RNN) can serve as embedding nets to encode high-dimensional time series (Lueckmann et al. 2017; Greenberg et al. 2019), convolutional neural networks (CNN) can serve as embeddings for spatially structured data like images (Greenberg et al. 2019; Ramesh et al. 2022), and equivariant embedding nets (Zaheer et al. 2017; Dax et al. 2022) can help to exploit equivariances in the data, e.g., for independent and identically distributed data (Chan et al. 2018; Radev et al. 2022).

3.2.3 Density estimators

Applying NPE or NLE requires choosing a neural density estimator. Driven by advances in probabilistic machine learning, the repertoire of neural density estimators available for SBI has evolved over the years. While Papamakarios et al. 2016 relied on *mixture density networks* (MDN, Bishop 1994) when introducing NPE, they later were able to use *normalizing flows* (Papamakarios et al. 2017) when introducing NLE (Papamakarios et al. 2019).

MDNs are neural networks designed to predict the parameters of a parametric family of probability distributions from data. For example, they predict the means, mixture coefficients, and covariance matrices of a mixture of Gaussians (or a mixture of other tractable distributions) from input data x to approximate the posterior over θ . While a mixture of Gaussians with enough mixture components can, in principle, approximate any probability distribution, the number of components used in MDNs is limited by the resulting number of neural network weights and units in the output layers. Thus, they are capable of fitting multi-modal Gaussian-like distributions but tend to struggle for more complex distributions (see e.g., Greenberg et al. 2019).

Normalizing flows provide substantially more flexibility than MDNs. They consist of a series of invertible and differentiable transformations T that map from a tractable base distribution $p_u(\mathbf{u})$ to a target distribution $p_x(\mathbf{x})$ (Papamakarios et al. 2021):

$$\mathbf{x} = T(\mathbf{u}) \text{ where } \mathbf{u} \sim p_u(\mathbf{u}) \quad (3.4)$$

$$p_x(\mathbf{x}) = p_u(\mathbf{u}) |\det J_T(\mathbf{u})|^{-1} \text{ where } \mathbf{u} = T^{-1}(\mathbf{x}), \quad (3.5)$$

where $J_T(\mathbf{u})$ is the Jacobian matrix.

The transformations T are parametrized by invertible neural networks, which are designed such that the corresponding Jacobians can be calculated efficiently. Consequently, using standard neural network training procedures to optimize the transformations is possible, resulting in a highly flexible density estimator. See Papamakarios et al. 2021 for an extensive review of normalizing flows.

Normalizing flows are currently the default choice in many SBI applications. However, while flows tend to have substantially more capacity than MDNs, they also tend to be slower during evaluation and sampling, resulting in slower neural network training and MCMC sampling. Therefore, there are scenarios where MDNs are the better choice (see e.g., Beck et al. 2022). Other density estimators used in SBI methods include Gaussian processes (Meeds et al. 2014; Wilkinson 2014) or score-based diffusion models (Song et al. 2019; Geffner et al. 2022; Sharrock et al. 2022).

3.3 Training and inference

All neural SBI methods are based on training artificial neural networks and therefore require choices about corresponding hyperparameters and convergence metrics. The likelihood-based SBI methods (NLE and NRE) also rely on approximate methods like MCMC or variational inference (VI) to obtain posterior samples, which come with additional hyperparameter choices and calibration metrics.

3.3.1 Neural network training

Although the different SBI approaches solve different learning problems, e.g., unsupervised conditional density estimation for NPE and NLE versus supervised classification for NRE, the training procedures for the underlying neural networks are very similar. The neural network weights are optimized by iterative updating according to the gradient of the loss function, which is usually calculated via backpropagation. Thus, training neural SBI algorithms involves the standard neural network training hyperparameter choices, e.g., choosing a training batch size, learning rate, number of epochs, and stopping criteria, for which good heuristics and default choices are available in open-source software packages like PyTorch (Paszke et al. 2019) or the SBI toolkit (Tejero-Cantero* et al. 2020) and in benchmarking studies, e.g., Lueckmann et al. 2021.

3.3.2 Inference

When using NPE, one can perform inference given the observed data directly after training as it returns a parametric approximation to the posterior that can be sampled and evaluated with a single pass through the underlying neural network. In contrast, NLE and NRE return an estimate of the likelihood (ratio) and, therefore, additionally require sampling algorithms to obtain posterior samples. The classical choice for sampling in SBI is Markov Chain Monte Carlo methods (MCMC, Metropolis et al. 1953; Hastings 1970). More recently, variational inference (VI, Blei et al. 2017) has been proposed as a sampling method for SBI as well (Wiqvist et al. 2021; Glöckler et al. 2022).

MCMC MCMC is a method to obtain samples from a distribution for which one does not have the normalization constant, e.g., for obtaining posterior samples when we only have the likelihood and the prior. The general idea of MCMC is to start with (randomly) initialized parameters, e.g., sampled from the prior, and to iteratively construct a chain of parameters that will eventually converge to the target distribution. The next parameter θ' in the chain is obtained by perturbing the current parameter θ_t using a specific perturbation kernel $q(\theta'|\theta_t)$ (e.g., a Gaussian) and accepting it according to an acceptance probability ρ calculated from the prior, likelihood (ratio), and the kernel,

$$\rho = \min \left(1, \frac{p(\theta') p(x|\theta') q(\theta_t|\theta')}{p(\theta_t) p(x|\theta_t) q(\theta'|\theta_t)} \right). \quad (3.6)$$

The chain constructed using this equation will converge to the target distribution Metropolis et al. 1953.

Several variants of the metropolis MCMC algorithm have emerged that substantially improve its efficiency and accuracy. The most prominent ones are slice sampling (Radford M. Neal 2003) and Hamiltonian Monte Carlo (HMC, Radford M Neal et al. 2011; Homan et al. 2014; Betancourt 2017). Thus, in SBI practice, it is common to use slice sampling, HMC, or a combination of the two like the no-u-turn sampler (NUTS, Homan et al. 2014). Additionally, one has to select MCMC hyperparameters and evaluate several performance metrics, e.g., whether to subsample the chain to avoid autocorrelation, how many MCMC chains to run in parallel, how to initialize the chains, and how to detect convergence—see Hogg et al. 2018 and Vehtari et al. 2021 for a general overview of using MCMC in practice, and Lueckmann et al. 2021 for SBI-specific recipes.

Variational inference Variational inference (VI) takes a different approach than MCMC. Instead of generating posterior samples, it aims to learn a parametric approximation to the posterior. To that end, one defines a parametric family of distributions and then tries to find the member in the family closest to the unknown posterior distribution. The formal goal of VI is to minimize a divergence measure between the posterior $p(\theta|x)$ and the parametric approximation $q_\phi(\theta|x)$, e.g., the Kullback-Leibler divergence (KL). Minimizing this divergence is intractable for most practical applications. The crux of VI is to reformulate the intractable problem of minimizing the divergence into a tractable optimization problem. In particular, it was shown that minimizing the KL is equivalent to maximizing the so-called *evidence lower bound* (ELBO), which is solvable using standard numerical optimization methods (Blei et al. 2017).

Recently, VI was proposed as an alternative sampling algorithm for NLE and NRE (Wiqvist et al. 2021; Glöckler et al. 2022). The approach by Glöckler et al. 2022 called SNVI (sequential neural variational inference) combines the advantages and avoids the disadvantages of NPE and NLE / NRE, respectively. Like NPE, it results in a parametric approximation to the posterior that can be sampled and evaluated efficiently and does not require expensive MCMC sampling. At the same time, like NLE and NRE, SNVI does not require corrections when performing sequential inference with active learning schemes. Thus, while MCMC is the default choice in most likelihood-based SBI applications, the recent VI approaches provide a promising alternative for high-dimensional SBI problems.

3.3.3 DDM example

Given the properties of the different SBI approaches outlined above, which one would we choose for the DDM? In the form presented here, the DDM has a three-dimensional parameter space and two-dimensional data space and is fast to simulate. Therefore, it represents a relatively simple SBI problem. However, one essential feature is its frequent use in experimental setups with many iid observations and corresponding hierarchical inference scenarios (Shiffrin et al. 2008; Wiecki et al. 2013). Thus, applying single-round NLE or NRE trained on single-trial simulations would be beneficial. Subsequently, one could perform inference via MCMC (or VI) given observations with varying numbers of trials without having to retrain the underlying neural networks. Alternatively, one could use NPE with permutation invariant embeddings networks. While this would substantially speed up inference as no MCMC sampling would be required, it would likely slow down the simulation and training phases, as NPE requires multiple simulated trials for each training data point.

3.4 A-posteriori checks

After obtaining an approximation to the posterior over model parameters—be it in the form of posterior samples or as parametric approximation—there are several checks available for testing the accuracy of the approximation, even if the underlying posterior is unknown. Below, I first present posterior predictive checks, which provide an intuitive and direct evaluation of the predictive performance of the posterior. I then outline more indirect checks probing the internal statistical consistency of the SBI algorithm. Lastly, I show how the posterior can be analyzed after successful evaluation to reveal parameter interactions in the model.

3.4.1 Posterior predictive checks

Posterior predictive checking is analogous to the prior predictive check (see section 3.1). It tests how the posterior predictive distribution,

$$p(\tilde{x}|x_o) = \int_{\theta} p(\tilde{x}|\theta)p(\theta|x_o)d\theta, \quad (3.7)$$

i.e., the distribution obtained by simulating data $\tilde{x} \sim p(x|\theta_i)$ with parameters sampled from the inferred posterior, $\theta_i \sim p(\theta|x_o)$, compares to the observed data x_o . Several sophisticated approaches have been developed for performing this test in the context of classical approximate inference, e.g., leave-one-out cross-validation or specific information criteria (Gelman et al. 2020). However, most of these tests do not apply to the SBI setting as they require direct access to the model’s likelihood. Thus, in SBI, a common approach is directly comparing the predicted and observed data. For example, we sample 1,000 parameters θ from the posterior and run the simulator to obtain 1,000 predictions \tilde{x} . Subsequently, we plot the simulated data next to or on top of the observed data and compare them visually. Ideally, the predicted data would cluster around the observed data with a variance matching the noise expected in the simulator (see 3.4b). If this is not the case, e.g., if parts of observed data lie clearly outside the distribution of predicted data, this is a sign of model misspecification or a problem in the inference method.

3.4.2 Simulation-based calibration

Simulation-based calibration (SBC, Cook et al. 2006; Talts et al. 2020) provides a way to systematically evaluate the internal consistency of the inference procedure without requiring access to the true posterior. In essence, it checks whether the uncertainties of the inferred posteriors are well-calibrated, i.e., that they are neither too narrow (overconfident) nor too wide (underconfident). To perform SBC, we use a given SBI method to repeat the inference procedure N times with pseudo-observed data generated from the simulator using parameters sampled from the prior:

$$\theta^* \sim p(\theta) \quad (3.8)$$

$$x^* \sim p(x|\theta^*). \quad (3.9)$$

Following this procedure, we obtain N posteriors $p_i(\theta|x^*)$, one for each pair in $\{(\theta^*, x^*)\}_{i=1}^N$. Subsequently, we obtain a set of L posterior samples from each posterior and calculate the rank r_i of

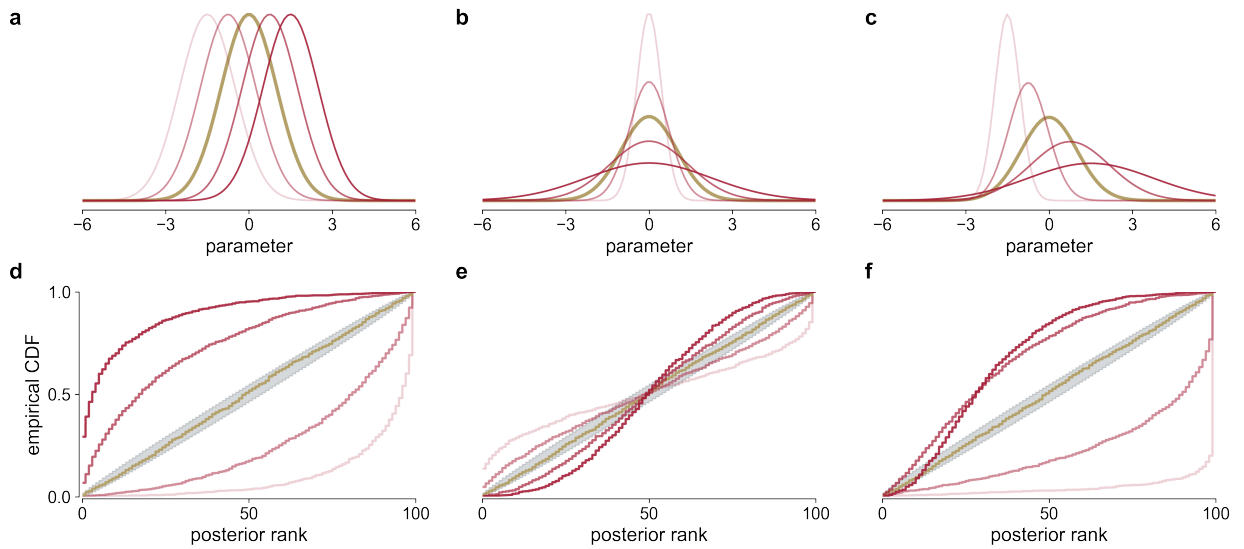


Figure 3.3: Visual interpretation of simulation-based calibration. A systematic bias in an SBI method may be due to a shift in the posterior mean (a), an over- or underestimated posterior variance (b), or a mixture of both ((c), ground-truth posterior in gold, biased posteriors in shades of red). By performing SBC, these different biases become visible in the shapes of the corresponding empirical CDFs of SBC ranks: Symmetric offsets from the diagonal indicate a mean shift (d), S-shapes around the diagonal indicate over- or underdispersion (e) or a mixture of both (f).

the underlying parameter θ_i^* among the posterior samples:

$$\{\theta_1, \dots, \theta_L\} \sim p_i(\theta|x^*) \quad (3.10)$$

$$r_i = \sum_{j=1}^L \mathbb{I}[f(\theta_j) < f(\theta_i^*)] \in [0, L], \quad (3.11)$$

where \mathbb{I} is the indicator function and $f : \Theta \rightarrow \mathbb{R}$ can be any one-dimensional random variable (Talts et al. 2020). The ranking is well-defined only in the one-dimensional case. Thus, in practice, one usually performs the ranking separately for each dimension of the posterior and sets $f(\theta) = \theta$.

The central insight in the SBC procedure is that *if* the posterior uncertainties are well-calibrated, *then* the calculated ranks $\{r_1, \dots, r_N\}$ follow a uniform distribution across integers $[0, L]$ (Talts et al. 2020). Equivalently, if the ranks deviate from a uniform distribution, then the posterior uncertainties are not well-calibrated, indicating that the posteriors are systematically biased. Thus, the uniformity check of SBC provides a necessary condition for the validity of the inference procedure.

Interpreting SBC results One way to test for the uniformity of the ranks is null-hypothesis significance testing, e.g., using the Kolmogorov-Smirnov test of uniformity (Kolmogorov 1933; Smirnov 1948). In practice, however, it is often more instructive to visualize the histograms or the empirical cumulative density function (CDF) of the ranks and to check for uniformity by visual comparison (see Fig. 3.3). Furthermore, visualizing the distributions of the ranks can give insights into the type of bias present in the posterior, e.g., whether the posteriors are systematically too broad or too narrow. For example, an accumulation of the ranks at smaller values indicates that the posteriors are systematically biased towards larger parameter values. Conversely, a tendency

towards larger values indicates a bias towards inferring too small parameter values. Fig. 3.3 illustrates this for a Gaussian toy example. Posteriors could be systematically biased towards smaller or values (Fig. 3.3a), systematically over- or underestimate the variance (Fig. 3.3b), or both (Fig. 3.3c). When visualizing the SBC ranks as empirical CDF lines, these three scenarios correspond to characteristic shapes: A systematically shifted posterior corresponds to CDF lines lying above or below the diagonal (Fig. 3.3d), over- or under-dispersed posteriors show S-shapes around the diagonal (Fig. 3.3e), and a mixture of both biases shows corresponding mixtures of both CDF effects (Fig. 3.3f).

SBC in practice In practice, performing SBC requires repeating the inference N times, where N should be on the order of hundreds for SBC to give reliable results (Talts et al. 2020). For NPE, this is computationally cheap because, in the fully amortized setting of NPE, one can obtain posteriors for each x^* instantly. Running SBC for NLE or NRE is computationally more demanding as it requires rerunning MCMC or VI to obtain posterior samples for each x^* (retraining the underlying neural networks is not required). SBC is especially time-consuming for all sequential SBI variants because they require rerunning training and simulating from round two onwards for each new x^* .

SBC can be gamed: when setting the posterior equal to the prior, all SBC checks would pass. However, this can be detected when running complementary posterior predictive checks. Another caveat of SBC is that, in its basic form, it applies only to the one-dimensional marginals of the posterior, i.e., it has to be applied for each posterior dimension separately. Several alternative approaches have been proposed to address some of SBC’s limitations. These include expected posterior coverage tests (Dalmaso et al. 2020; Hermans et al. 2021; Miller et al. 2021; Deistler et al. 2022a), conditional coverage tests (Masserano et al. 2022), or a reframed SBC version that is conditional on a specific observation (Modrák et al. 2022).

Probability of θ^* Another posterior evaluation metric that can be calculated without access to the true posterior is the average negative log probability of the true parameters (NLTP) θ^* under the inferred posterior $p(\theta|x^*)$:

$$\text{NLTP} = \mathbb{E}_{\theta^* \sim p(\theta)} \mathbb{E}_{x^* \sim p(x|\theta^*)} [q(\theta|x^*)]. \quad (3.12)$$

NLTP provides a metric for comparing the performance of different SBI approaches and has been used in the SBI literature extensively (Papamakarios et al. 2016; Durkan et al. 2020; Greenberg et al. 2019; Papamakarios et al. 2019; Hermans et al. 2020). Alternatively, it could be a criterion for comparing different neural network architectures within one SBI approach. Importantly, it is only a valid performance measure when calculated over many (θ^*, x^*) generated from the prior (Talts et al. 2020; Lueckmann et al. 2021). In the limit of an infinite number of pairs (θ^*, x^*) , it converges to an expression with two terms: the Kullback-Leibler divergence between the approximate and the true posterior averaged over all x^* and the average entropy of the true posterior (Lueckmann et al. 2021):

$$\mathbb{E}_{x^* \sim p(x)} D_{\text{KL}}(p(\theta|x^*) || q(\theta|x^*)) + \mathbb{E}_{x^* \sim p(x)} \mathbb{H}(p(\theta|x^*)) \quad (3.13)$$

Only the latter depends on the SBI algorithm and provides a measure for the accuracy of the approximation.

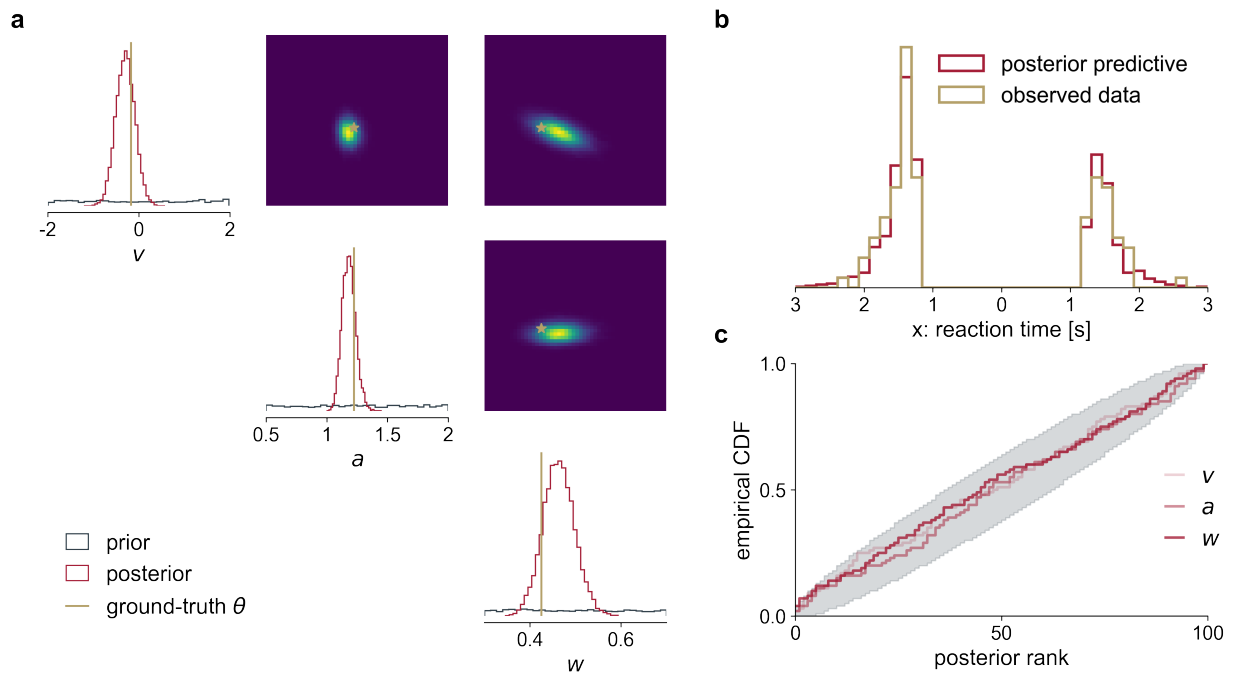


Figure 3.4: Posterior analysis for the drift-diffusion model. (a) Visualization of the three-dimensional posterior over DDM parameters with one-dimensional marginals on the diagonal and two-dimensional marginals on the off-diagonal, ground-truth parameters in shown gold. (b) Reaction times and choices simulated from the ground-truth parameters (observed data, gold) and the posterior distribution (posterior predictive, red). (c) Simulation-based calibration results for each parameter of the DDM, shown as empirical cumulative density functions of the calculated ranks (see main text). The grey area represents random fluctuations expected under the uniform distribution.

3.4.3 Posterior analysis

After choosing an SBI method, training the corresponding neural networks, obtaining posteriors samples, and performing posterior checks to ensure the validity of the inference, we can turn to the initial goal of the SBI endeavor: running Bayesian parameter inference and interpreting the results. To that end, we apply the trained and validated SBI method to the observed data x_o and obtain an approximation to the posterior (or samples from it) over the model parameters conditioned on x_o .

One essential benefit of the Bayesian inference framework is that the posterior identifies *all* parameter combinations likely to reproduce the data, as well as their co-relation. The plausible values for each parameter are characterized by the corresponding one-dimensional marginal distribution, where the variance represents the uncertainty concerning the parameter. This uncertainty could be due to noise in the data-generating process or because several parameter values can explain the data equally well due to compensation mechanisms or degeneracies in the model (Gutenkunst et al. 2007; Gonçalves et al. 2020). Additionally, the covariance structure of the full posterior distribution characterizes the co-relations between each pair of parameters. Analyzing this structure can help reveal degeneracies and compensation mechanisms between model parameters (Golowasch et al. 2002; Gonçalves et al. 2020; Deistler et al. 2022b).

Visualization One way to visualize potential high-dimensional posteriors in two dimensions is the so-called *pair plot*. The pair plot consists of a matrix of subpanels showing each parameter’s one-dimensional marginal on the diagonal and the two-dimensional marginals for every possible pair of parameters on the upper- or lower off-diagonal (see Fig. 3.4a). Thus, on the diagonal, one can read off the inferred ranges of possible parameter values for each dimension, while the two-dimensional densities on the off-diagonal already give us a glimpse of the covariance structure of the posterior that may reveal parameter interactions in the model.

Correlations The pair plot can be inspected visually for potential interactions between the parameters by searching for elliptically shaped regions in the two-dimensional marginals. For example, in the posterior of the DDM example presented in Fig. 3.4b, the shape of the two-dimensional marginal between the drift parameter v and the offset parameter w (upper right panel) indicates a negative correlation, while the other two pairs appear to show almost no correlations. The interpretation of this shape would be that as we increase the drift, the posterior density remains high if we simultaneously decrease the offset in the initial condition. In other words, one way to obtain the same data (choices and reaction times) while increasing the drift (more evidence towards the upper boundary) would be to induce an initial bias towards the lower boundary by decreasing w . A common way to quantify the suspected correlations is to calculate the corresponding Pearson correlation coefficients from posterior samples.

Conditional correlations In some situations, the inferred posteriors are relatively broad for many model parameters, suggesting that the observed data only weakly constrain the parameters. However, this can be misleading because the univariate and pairwise marginals represent averages over all possible values of the remaining posterior dimensions. Consequently, if the posterior is high-dimensional, parameter interactions and compensation mechanisms might not be visible in a low-dimensional visualization and the corresponding correlation analysis. To still detect potential dependencies between the parameters, one can instead calculate the *conditional correlations* in the posterior (see Gonçalves et al. 2020; Deistler et al. 2022b; Boelts et al. 2023, for examples).

Sensitivity analysis The above steps enable us to identify regions of data-consistent parameters and directions of interactions between the parameters. It is possible to investigate these regions further. One approach proposed in Gonçalves et al. 2020 is identifying paths in the parameter space along which parameters are data-consistent. For example, given two distinct parameter combinations closely reproducing the observed data, one can use optimization techniques to find a path of high posterior density that connects to two parameters in posterior space. Along this path through the posterior space, all parameter combinations are data-consistent, whereas small perturbations away from the path can lead to inconsistent data (see, e.g., Gonçalves et al. 2020, Fig. 5).

Another way of analyzing these posterior subspaces is conducting a *sensitivity analysis* in posterior space, as proposed in Deistler et al. 2022b. Here, the idea is to identify the directions in the posterior space in which the parameters are most sensitive, in the sense that changing them would make them data-inconsistent. For example, given an elliptical shape in the two-dimensional marginal of v and w in the DDM example and the corresponding negative correlation coefficient (Fig. 3.4a, upper right subpanel), we would expect that the parameters are less sensitive in the elongated direction of the ellipse and more sensitive in the direction orthogonal to that. Formally, these directions can be identified by constructing the outer product of the gradients of the posterior

density with respect to the parameters

$$M = \mathbb{E}_{\theta \sim p(\theta|x)}[\nabla p(\theta|x)\nabla p(\theta|x)^\top], \quad (3.14)$$

and obtaining the eigenvectors and eigenvalues of the resulting matrix M (Deistler et al. 2022b; Constantine 2015). The eigenvectors identify the directions and the eigenvalues the degree of sensitivity. This sensitivity concerns all aspects of the observed data because the posterior is conditioned on x_o . The sensitivity analysis can be further generalized to study the parameters' sensitivity to specific properties of the data, e.g., specific summary statistics of interest $s(x)$ (see Constantine 2015; Deistler et al. 2022b, for details).

3.5 Summary

This chapter provided an overview of the workflow for using SBI in practice. We have seen that the actual execution of the SBI algorithm is only one step among many other important evaluation steps and choices to be performed before and after the inference: Before applying SBI, prior predictive and model misspecification checks are required to ensure that SBI will give reliable results. When selecting a specific SBI algorithm and a suitable neural network architecture, the structure and dimensionality of the data and the parameters, as well as many other factors, must be considered. After obtaining the approximation to the posterior, several checks are required, e.g., checks to ensure that the posterior variances are well-calibrated or that the posterior accurately reproduces the observed data before, finally, the posterior can be analyzed and interpreted.

This workflow considers the currently available procedures for applying SBI to new inference problems. With ongoing research on new SBI methods and validation techniques, the workflow will need to be adapted to improve the applicability and reliability of SBI further.

Chapter 4

Publications

In this chapter, I present the publications forming the basis of this thesis. I start with a short overview of all publications. In the remainder of the chapter, I provide a summary and a note on my contributions for each publication. The full papers are attached in the appendix.

4.1 Overview

1. *Flexible and efficient simulation-based inference for models of decision-making.*
Jan Boelts, Jan-Matthis Lueckmann, Richard Gao, Jakob H. Macke (2022)
eLife (Boelts et al. 2022).
2. *Simulation-based inference for efficient identification of generative models in computational connectomics.*
Jan Boelts, Philipp Harth, Richard Gao, Daniel Udvary, Felipe Yanez, Daniel Baum, Hans-Christian Hege, Marcel Oberlaender, Jakob H. Macke (2023)
bioRxiv (Boelts et al. 2023).
3. *sbi: A toolkit for simulation-based inference.*
Álvaro Tejero-Cantero*, **Jan Boelts***, Michael Deistler*, Jan-Matthis Lueckmann*, Conor Durkan*, Pedro J. Gonçalves, David S. Greenberg, and Jakob H. Macke (2020)¹
Journal of Open Source Software (Tejero-Cantero* et al. 2020).

All three publications contribute to my overall aim of advancing methods and the applicability of simulation-based inference (SBI). The first paper contributes a new SBI method that enables efficient Bayesian parameter inference in computational models of decision-making. As parameter inference is a central task in cognitive neuroscience, an extensive toolkit of parameter-tuning methods for models of decision-making already exists. However, current methods are often limited to non-Bayesian approaches that optimize for single best-fitting parameters or Bayesian approaches that work only for simplified analytically tractable models. While SBI would overcome these limitations, current SBI methods often struggle with the data types and experimental setups commonly encountered in decision research. Therefore, we develop an SBI method tailored to decision-making models. By building on recent advances in neural-network-based density estimation (see [General Background](#)), our method is several orders of magnitude more simulation-efficient than previous approaches, enabling its application to a broader range of inference problems.

¹*equal contribution

The second paper demonstrates how SBI can be applied to inference problems in *computational connectomics*. In connectomics, large amounts of data about the physical and functional connectivity of neurons are acquired to study the general principles underlying brain connectivity. In addition to these experimental efforts, researchers build computational models to efficiently develop and test hypotheses about the data. For example, one way to test whether a hypothesized wiring rule can explain the measured data is to generate corresponding simulated connectivity data in a computational model and compare it to measured data. As with most computational modeling approaches, this poses the challenge of identifying the free parameters of the wiring rule such that the simulated data matches the measured data. We show how to apply SBI to computational models used in connectomics to address this challenge. With this, we set the stage for applying SBI to challenging inference problems in connectomics to enable the efficient testing of hypotheses derived from connectivity measurements.

The third contribution addresses the applicability of SBI. The recent advances in neural-network-based density estimation fostered the development of many new neural SBI methods (see [Neural simulation-based inference](#) for details). However, from a practitioner's view, these methods are often challenging to apply because of a lack of software tools and application guidelines. To address this gap between method development and applicability, we develop an SBI software package that provides access to the main neural SBI approaches in a well-documented and user-friendly way. We thereby hope to facilitate the application of SBI for researchers and practitioners.

4.2 Flexible and efficient simulation-based inference for models of decision-making

4.2.1 Summary

Motivation

In cognitive neuroscience, computational models are essential for analyzing experimental data and testing hypotheses about the neural mechanisms underlying cognition. For example, a common approach for studying perceptual decision-making processes is to record behavioral and neural data of subjects performing a simple decision-making task. This experimental data is often analyzed using so-called *drift-diffusion models* (DDM, see [General Background](#), Fig. 2.2) to relate experimentally recorded behavioral data (reaction times and choices), and neural data (Ratcliff 1978; Bogacz et al. 2006; Gold et al. 2007). To identify the free parameters of the employed computational models like the DDM, optimization methods that identify single best-fitting model parameters Tavares et al. 2017; Shinn et al. 2020 as well as Bayesian statistical inference methods are commonly used (Wiecki et al. 2013; Kangasräsiö et al. 2019). While Bayesian inference methods tend to provide a more principled approach to parameter inference in cognitive models (see [General Background](#), Shiffrin et al. 2008; Schad et al. 2021), they are usually limited to a specific type of models for which the likelihood function is accessible, e.g., they only work for a subset of DDMs. SBI thus provides a suitable alternative as it enables Bayesian inference using only simulations from the model. However, the currently available SBI methods can be challenging to apply to the data and experimental setups typically used in decision-making research, e.g., they struggle to deal with the mixed data types of *discrete* choices and *continuous* reaction times. Our work addresses these challenges by providing an SBI method tailored to decision-making models.

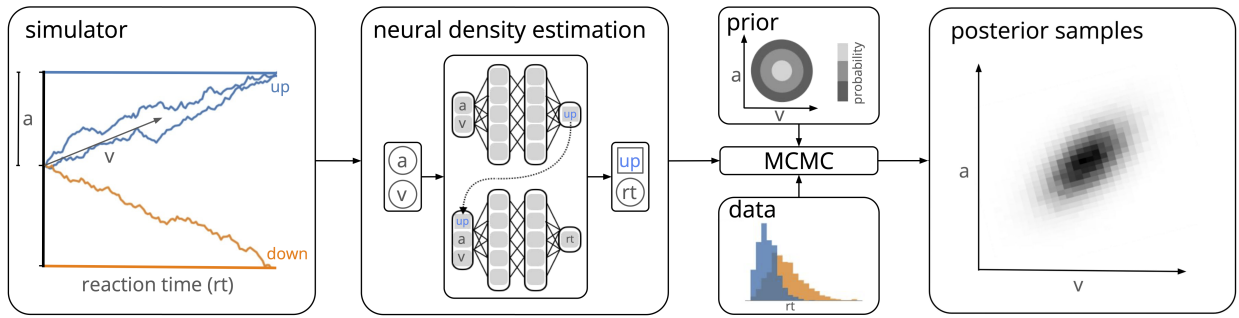


Figure 4.1: Mixed Neural Likelihood Estimation (MNLE) for models of decision-making. Decision-making models commonly deal with data of mixed types, e.g., continuous reaction times and discrete choices (left). To enable SBI for this scenario, we propose training separate neural density estimators on the discrete and the continuous data, respectively (middle). After training, the estimators can be combined into one likelihood estimate to enable Bayesian inference using standard MCMC sampling (right). Figure adapted from Boelts et al. 2022.

Methods

The data recorded in decision-making experiments usually consist of continuous reaction times and discrete choices. Additionally, it is common to record many repetitions with the same settings for each experimental condition, i.e., the data consists of a set of N independently and identically distributed (iid) trials. We can use the DDM to analyze this data. The DDM simulates reaction times and choices, $x = [rt, c]$, using a stochastic differential equation based on a drift parameter v , an initial offset w , and decision boundary parameter a :

$$dX = v dt + dW, X(0) = w, \quad (4.1)$$

(see Boelts et al. 2022 for details). Given experimentally observed data $x_o = \{rt_i, c_i\}_{i=1:N}$, we can use SBI to infer parameters $\theta = [a, w, a]$ that reproduce x_o .

When selecting the type of SBI algorithm to apply, the properties of the above setup have to be taken into account. For example, applying neural posterior estimation (NPE) to this type of data can be challenging because NPE approximates the posterior directly from the data, e.g., the neural network underlying NPE takes the *iid* data as input. Consequently, one would need to account for the permutation invariant structure of a set of iid trials (Chan et al. 2018; Radev et al. 2022), which can be challenging for large and varying numbers of observed trials. In contrast, neural likelihood estimation (NLE) could be more suitable as it takes the parameters as input and performs density estimation on the data to approximate the likelihood. Thus, one could adapt NLE to the iid-trial setting of decision-making models by training it on single-trial data points to approximate single-trial likelihoods. Subsequently, one could perform inference given observed data with large and varying numbers of iid trials by leveraging the fact that iid-likelihoods factorize and combining the single-trial likelihood estimates. However, to run NLE, we would need to perform density estimation on the mixed (continuous and discrete) data of decision-making models, which is challenging for current neural network-based density estimators as they are typically designed for continuous or discrete data but not for both.

Our central contribution in this paper is to enable neural density estimation for mixed data types, i.e., to perform *Mixed Neural Likelihood Estimation* (MNLE, Fig. 4.1). To do so, we leveraged

the fact that the probability density over the mixed data factorizes:

$$p(x|\theta) = p(rt, c|\theta) \quad (4.2)$$

$$= p(c|\theta) \cdot p(rt|c, \theta). \quad (4.3)$$

Consequently, it is possible to use separate density estimators: one for the discrete data to estimate $p(c|\theta)$, and one for the continuous data $p(rt|c, \theta)$ (Fig. 4.1, middle left). To estimate the discrete distribution $p(c|\theta)$, we trained a feed-forward neural network $F(\cdot)$ to predict the parameter ρ of a categorical distribution q_c from the parameters θ : $q_c(c|\theta)$. To estimate the continuous distribution $p(rt|c, \theta)$, we used a neural spline flow q_{rt} parametrized by a neural network $G(\cdot)$ (Papamakarios et al. 2021; Durkan et al. 2019) conditioned on both c and θ : $q_{rt}(rt|c, \theta)$. We conditioned q_{rt} on the parameters *and* the choices to capture potential dependencies between choices and reaction times. The training was performed separately for F and G but using the same training data simulated from the DDM with parameters sampled from the prior. After training, we obtained the likelihood estimator by combining the two density estimators:

$$\hat{p}(x|\theta) = q_c(c|\theta) \cdot q_{rt}(rt|c, \theta). \quad (4.4)$$

The likelihood estimator enabled us to obtain posterior samples given the observed data x_o using MCMC or variational inference (Fig. 4.1, middle right) as in any other NLE approach (see **General Background** for details). Importantly, we trained the density estimators on single trials simulated from the DDM. At inference time, this enabled us to combine single-trial likelihood estimates into the multi-trial estimates needed for the observed data $x_o = \{rt_i, c_i\}_{i=1:N}$ and to repeat inference for different x_o or N without having to retrain F and G .

Results

We demonstrated the utility of MNLE by performing SBI on two variants of the DDM: one with access to reference likelihoods and posterior samples to show that MNLE works accurately and one without access to a reference solution to test MNLE in a more realistic setting. Additionally, we compared MNLE against a recently proposed SBI method called likelihood approximation networks (LAN, Fengler et al. 2021) that was designed for decision-making models too. We compared MNLE and LANs regarding likelihood accuracy (mean squared error to the reference solution) and simulation efficiency on the DDM with a reference solution. Both methods could accurately predict the DDM likelihoods and obtain accurate posterior samples via MCMC. However, we found that MNLE was six orders of magnitude more simulation efficient than LANs, i.e., where LANs required 10^{11} training simulations to achieve good likelihood accuracy, MNLE needed only 10^5 simulations. For the second DDM variant (no reference solution available), we found that MNLE obtained posteriors with well-calibrated variances and accurate predictive performance.

The substantial difference in simulation efficiency between MNLE and LAN is due to a crucial difference in the way they perform density estimation: LANs are trained in a supervised fashion, i.e., by performing regression from data and parameters of the DDM (features) onto the corresponding value of the likelihood function (target). This approach works very well; however, it requires likelihood targets for training, which have to be estimated from simulated data, e.g., using kernel density estimation (KDE). Thus, to train LANs, Fengler et al. 2021 needed to perform KDE with thousands of simulations for each of their millions of training data points. In contrast, MNLE performs *conditional* density estimation: instead of predicting a likelihood target from parameters

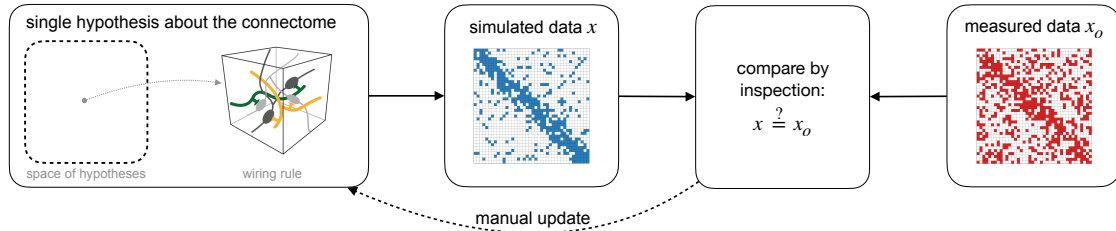
and data, the neural networks are trained to maximize the probability density of the data conditional on the parameters and are supervised implicitly by the likelihood of the current estimate, i.e., without requiring additional simulated data for obtaining training targets. Consequently, MNLE is more simulation efficient than LANs by at least the number of simulated data LANs need to construct likelihood targets. In summary, by introducing conditional neural likelihood estimation for mixed data, we enabled flexible and efficient SBI for decision-making models, facilitating more efficient model design in decision-making research.

4.2.2 Contributions

Contributions are listed according to the **CRediT** system. This paper is co-authored by me, Jan-Matthis Lueckmann, Richard Gao, and Jakob Macke. As the leading author of the paper, I conceived and conducted the project with feedback from all co-authors. I wrote all code, ran all the experiments, performed all analyses, and prepared the initial draft of the manuscript. I conducted the formal analysis, preparation of figures, and editing of the manuscript with help from all co-authors. Jakob Macke provided funding resources and the main supervision throughout the project.

4.3 Simulation-based inference for efficient identification of generative models in computational connectomics

a conventional generative modeling



b automated model identification with SBI

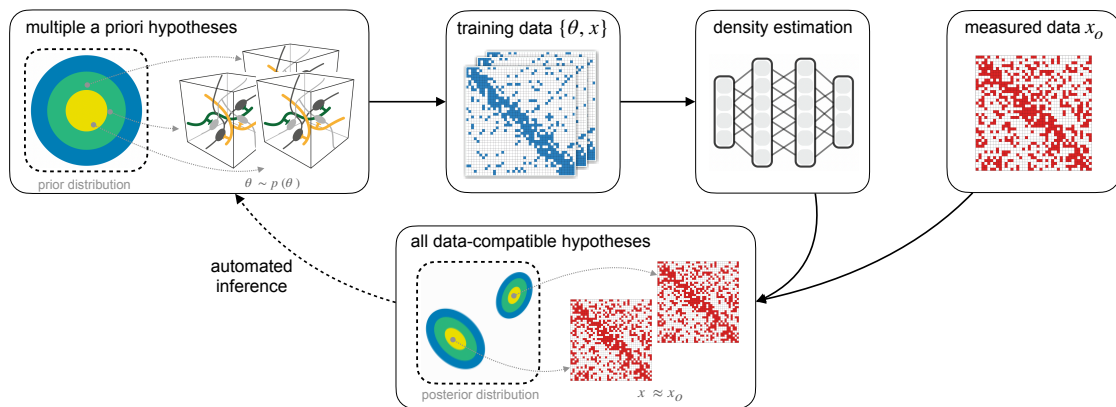


Figure 4.2: Enhancing generative modeling in computational connectomics using simulation-based inference. (a) Conventional generative modeling approaches in connectomics often test individual hypotheses, e.g., wiring rules, by comparing simulated and measured data. (b) Here, we propose to replace this approach with simulation-based Bayesian inference (SBI): Given a hypothesized wiring rule, we define a prior distribution over the parameters of the rule representing multiple *a-priori* variants of the rule (left). We then simulate connectivity data from many rule variants sampled from the prior and perform neural density estimation to estimate the posterior distribution over rule parameters conditioned on the measured data. The posterior characterizes *all* wiring rule parameters that reproduce the measured data. Figure adapted from Boelts et al. 2023.

4.3.1 Summary

Motivation

In recent years, advances in experimental and computational techniques have made it possible to obtain connectivity measurements and digital reconstructions of brain tissue in unprecedented detail. Consequently, there is a need for efficient computational tools to analyze this wealth of data and understand the underlying general connectivity principles. One computational approach to develop and test hypotheses about connectivity principles is building computational models that generate synthetic connectivity data according to a specific hypothesis. Subsequently, one can compare the synthetic data to the measurements to evaluate and refine the hypothesis.

For example, Udvary et al. 2022 built a digital model of a part of the rat sensory cortex, the so-called barrel cortex. Their model is based on reconstructions of neuron morphologies of different cell types and reconstructions of the barrel cortex geometry, cytoarchitecture, and cellular organization. These structural features were combined into a 3D model to obtain a realistic estimate of the structural composition of a large part of the barrel cortex. Consequently, they could use this model to simulate the effect of hypothesized wiring rules in the barrel cortex, e.g., by applying a wiring rule to the structural features of the model and comparing the resulting simulated connectivity patterns to connectivity measurements. This approach worked well for one specific wiring rule they constructed, a wiring rule that predicts connectivity using only morphological features of neurons: The rule predicted connectivity patterns that were in line with several connectivity measurements from the barrel cortex. However, several questions arise when we want to extend the rule to study additional wiring mechanisms, e.g., by introducing parameters. How can we adapt the parameters such that they can explain the measurements? Are there other rules that can explain the data equally well?

Previous approaches to generative modeling in connectomics often addressed these questions by optimizing for single best-fitting model configurations, e.g., by comparing simulated with empirical data and iteratively refining the model (Fig. 4.2a). This approach can be challenging if models have many parameters or if the data is high-dimensional, which is often the case given the recent advances in data acquisition methods. However, if we could apply SBI to the type of models used in connectomics, this would allow us to test many different parameter combinations simultaneously and to identify all those that match the data without having to compare data points explicitly (Fig. 4.2b). This project investigated whether and how existing SBI methods can help to constrain generative models in connectomics with observed data.

Methods

To demonstrate how SBI can be applied in computational connectomics, we used the generative model of the rat barrel cortex by Udvary et al. 2022 as an example. In particular, we extended the wiring rule they introduced with parameters and then showed how SBI can infer the parameters given connectivity data measured in the barrel cortex (see Boelts et al. 2023, Fig. 2).

The dense structural overlap (DSO) wiring rule The wiring rule proposed by Udvary et al. 2022 predicts the probability that two neurons i and j form a synapse in a small subvolume k of the model from their so-called *dense structural overlap* (DSO). The DSO is defined as the product of the relevant structural features of the pre-synaptic neuron i (*pre*) with the postsynaptic features of neuron j (*post*), divided by the sum of the postsynaptic features of all neurons in the direct neighborhood k :

$$DSO_{ijk} = \frac{pre_i \cdot post_j}{postAll_k}. \quad (4.5)$$

Udvary et al. 2022 simulated the effect of the DSO rule in the barrel cortex model by applying it to every neuron-pair-subvolume combination and generating synapses stochastically according to the predicted connection probabilities. However, this parameter-free DSO rule is limited to this specific combination of features for predicting connectivity, e.g., it assumes that the pre- and postsynaptic features contribute equally to the predicted connection probability. Different rule variants would allow the pre- and postsynaptic features to contribute with different weights to the synapse probability or have different contributions for different cell types in the barrel cortex.

To test extensions of the DSO rule efficiently using SBI, we introduced three parameters, one for each of the structural features:

$$DSO_{ijk}(\boldsymbol{\theta}) = \frac{pre_i^{\theta_{pre}} \cdot post_j^{\theta_{post}}}{postAll_k^{\theta_{postAll}}}. \quad (4.6)$$

In this new definition, the DSO rule can be seen as a simulation-based model (see [General Background](#)): Applying the rule with different parameter combinations to the structural features of the model corresponds to simulating different wiring rule configurations, such that each simulation results in a different simulated connectome.

To perform SBI, we needed to define a prior distribution over the three parameters and select measured data with which we want to constrain the parameters. We used a Gaussian prior distribution $p(\boldsymbol{\theta}) = \mathcal{N}([1, 1, 1]^\top, 0.5 \mathbb{I})$. As measured data x_o , we selected connection probabilities that have been measured in the rat barrel cortex (Bruno et al. 2006; Constantinople et al. 2013). These measurements are available as estimated connection probabilities for seven neuron populations projecting from the thalamus to the barrel cortex. However, the DSO wiring rule simulator generates an entire connectome of the rat barrel cortex. Therefore, to simulate these seven connection probabilities in the wiring rule simulator, we first generated the entire connectome according to the parametrized DSO rule and then estimated the connection probabilities by indexing and averaging over the corresponding seven sets of connections in the connectome.

To run SBI, we generated 1,000,000 simulated connection probabilities and performed sequential neural posterior estimation (SNPE) with neural spline flows (Papamakarios et al. 2016; Greenberg et al. 2019; Durkan et al. 2019) to obtain the desired approximation of the posterior $p(\boldsymbol{\theta}|x_o)$.

Validating SBI To validate our approach, we followed the procedure outlined in [Simulation-based inference in practice](#): First, we performed prior predictive checks to check for misspecification of the wiring rule simulator. Second, we derived a simplified version of the wiring rule simulator for which it was possible to calculate a reference solution using MCMC sampling, enabling us to validate the approximate SBI posterior. Third, we performed simulation-based calibration (SBC, Talts et al. 2020) to evaluate the calibration of the posterior variances. Lastly, we performed posterior predictive checks by simulating connection probabilities with parameters sampled from the approximate SBI posterior and comparing them to x_o .

Results

We found that SBI showed reliable inference performance on the wiring rule simulator: When applied to the simplified wiring rule simulator variant, it accurately recovered the reference solution. The SBC analysis with simulated data showed that posteriors inferred with SNPE on the full wiring rule simulator have well-calibrated variances. Additionally, the posterior predictive checks were accurate.

The posterior $p(\boldsymbol{\theta}|x_o)$ we obtained given the actual measurements x_o identified many parameter combinations able to reproduce x_o , including the one corresponding to the parameter-free DSO rule defined by Udvary et al. 2022 ($\boldsymbol{\theta} = [1, 1, 1]$, see equation 4.6). This result illustrated how SBI can identify many different data-compatible wiring rule parameters. Moreover, it enabled us to demonstrate another benefit of SBI—the access to the full posterior distribution: We found that the one-dimensional posterior marginals were similar to the prior marginals. However, the shape of

the two-dimensional marginals, i.e., the covariance structure, was substantially different from the uncorrelated prior distribution (see Boelts et al. 2023, Fig. 3a). Using access to the full posterior distribution, we found substantial correlations between all parameters. The correlations indicated that, despite the wide ranges of data-compatible parameters, the inferred parameters were highly interdependent. Once one parameter was fixed, the other parameters were highly constrained. This result further indicated that the DSO wiring with three parameters was overparametrized. Indeed, we were able to show that a version with only two parameters inferred with SBI could explain the measurements similarly well (see appendix of Boelts et al. 2023). Additionally, we showed how to leverage the predictive properties and the access to the complete simulated connectome of the rat barrel cortex provided by the wiring rule simulator to make experimentally testable predictions constrained by x_o .

Collectively, this paper demonstrated the potential of the SBI approach for computational connectomics: SBI allows researchers to systematically and efficiently identify parameters of computational models given empirical connectivity data, which will facilitate the evaluation of hypotheses in connectomics. Our approach sets the stage for using generative modeling to study connectivity principles in dense reconstructions of brain tissue recently made publicly available. Compared to the relatively sparse barrel cortex measurements used in our example, these dense reconstructions will allow us to constrain more complex models, opening up exciting possibilities to study connectivity principles in the mouse (Turner et al. 2022) and human cortex (Shapson-Coe et al. 2021).

4.3.2 Contributions

This publication was a collaboration with the group of Marcel Oberlaender (caesar, Bonn) and the group of Hans-Christian Hege and Daniel Baum (Zuse Institute Berlin). It was co-authored by me, Philipp Harth, Richard Gao, Daniel Udvary, Felipe Yanez, Daniel Baum, Hans-Christian Hege, Marcel Oberlaender, and Jakob Macke. The initial idea of applying SBI to inference problems in computational connectomics was developed by Jakob Macke, Marcel Oberlaender, and Hans-Christian Hege. I took the lead in realizing this idea and refined the conceptualization and formal analysis of the project together with Jakob Macke. Specifically, I developed the software for the inference procedure and validation and conducted the experiments with help from Philipp Harth and Felipe Yanez. I developed the wiring rule simulator with Philipp Harth; Daniel Udvary provided the rat barrel cortex model. I created the visualizations with help from Philipp Harth and wrote the initial draft of the manuscript. For editing the manuscript, I received help from Richard Gao and Jakob Macke and comments from all other authors. Jakob Macke provided the main supervision throughout the project.

4.4 sbi: A toolkit for simulation-based inference

Motivation

This publication refers to a software repository published in the Journal of Open Source Software (JOSS, Tejero-Cantero* et al. 2020). The motivation for this software project was to provide a central open-source resource for developing and applying neural-network-based SBI algorithms. While it is common to provide access to research code when publishing new SBI algorithms, these code bases are rarely designed in an easy-to-use way. Usually, they exist as standalone packages based on different machine-learning frameworks. Our goal was to change this for SBI, i.e., to unite the established neural-network-based SBI approaches in one package called `sbi`. We aimed to design the package with a clear and modular structure and to provide detailed documentation and tutorials such that the package would serve not only experienced SBI researchers but also scientists from different domains.

Summary

The publication of the `sbi` package in 2020 was only the starting point of our efforts to facilitate access to SBI. Since then, we have actively maintained the package, fixed issues, improved the API and documentation, and released new versions regularly. In 2021, we organized a public online workshop on “[SBI for scientific discovery](#)”, providing a general introduction to SBI, and on how to apply it using the `sbi` package. In 2022, we held a one-week hackathon with external participants from Germany and France, in which we fixed issues and implemented new features in `sbi`.

In its initial version, the package implemented the three main SBI approaches (S)NPE (Papamakarios et al. 2016; Lueckmann et al. 2017; Greenberg et al. 2019), (S)NLE (Papamakarios et al. 2019), and (S)NRE (Hermans et al. 2020; Durkan et al. 2020). Since then, the package has received over 790 issues and pull requests on [GitHub](#) and contributions from 35 developers. We implemented additional features like posterior checks (Cook et al. 2006; Talts et al. 2020; Miller et al. 2021; Deistler et al. 2022a), posterior analysis tools (Deistler et al. 2022b), and five new SBI algorithms (Glöckler et al. 2022; Boelts et al. 2022; Deistler et al. 2022a; Miller et al. 2022; Delaunoy et al. 2022). Indicated by the increasing number of GitHub issues, pull requests by external users and contributors, and the growing number of citations, the `sbi` package can be seen as one of the primary resources for both SBI research and applications.

Contributions

The initial version of the package published in JOSS was a joint effort of all authors, with equal first-author contributions by Álvaro Tejero-Cantero, me, Michael Deistler, Jan-Matthis Lueckmann, and Conor Durkan. Conor Durkan contributed his research code of the publication Durkan et al. 2020 and the “[nflows](#)” package, which both provided the basis of the `sbi` package. Álvaro Tejero-Cantero, Michael Deistler, Jan-Matthis Lueckmann, and I contributed equally to the initial implementation of the package. Since then, I have shared the leading role in maintaining and developing the package with Michael Deistler.

Most package features were developed collaboratively through GitHub pull requests (PR). I contributed to many of these features and reviewed all PRs. There are several features that I implemented myself, e.g., all approximate Bayesian computation (ABC)-related algorithms; the extensive checks and automated processing steps applied to the prior distribution, the simulator,

and observed data passed by the user; multi-processing features for running simulations and MCMC sampling; implementation of the Mixed Neural Likelihood Estimation (MNLE) algorithm.

Chapter 5

Conclusion

This thesis aims to advance the methods and applicability of simulation-based inference (SBI) in computational neuroscience. I made three contributions to achieve this: I proposed a new SBI method tailored to computational models used in decision-making research, demonstrated the utility of SBI for inferring wiring rules in computational connectomics, and contributed to an SBI software package to facilitate access for practitioners. To conclude, I want to touch on three themes that recur throughout the thesis.

5.1 A simulation-based model is all you need: scientific discovery with SBI

In a time of multiple global crises, simulation-based models provide an essential tool to foster scientific discoveries and technological solutions, e.g., by enabling more accurate predictions of climate scenarios (Peng et al. 2021; Glavovic et al. 2022). Given the increasing amount of data we can acquire and the resulting complexity of the models we build, it is crucial to the success of the modeling approach to advance the methods for identifying and interpreting model parameters (Cranmer et al. 2020; Lavin et al. 2022). While Bayesian inference provides a robust framework for scientific discovery, e.g., because it enables us to express uncertainties, it is often limited by the fact that it requires access to the inner workings of the simulator. SBI overcomes this limitation by requiring access only to simulations of the model. Thus, whereas previously, we often had to limit our modeling efforts to certain classes of tractable models, we can now say that *a simulation-based model is all you need*.

Throughout this thesis, we have seen the potential of SBI to facilitate scientific discovery. The first publication showed that by adapting SBI methods to the experimental setups standard in perceptual decision-making research, we can extend the application of Bayesian parameter inference beyond the commonly used canonical cognitive models, e.g., allowing researchers to replace the classical drift-diffusion model with custom-tailored models of decision-making. In the second publication, we have seen how the introduction of SBI to computational connectomics can open up new avenues for efficient and flexible testing of hypotheses about brain connectivity. While these are important contributions, there are, of course, many ways in which the utility of SBI could be enhanced, for example, by further improving the capabilities of SBI methods or by tailoring and applying current methods to specific problems of high practical relevance.

5.2 All models are wrong: model misspecification in SBI

When building models, we often make simplifying assumptions and leave aside potentially important aspects of the phenomenon of interest. As a consequence, *all models are wrong* by construction (Box et al. 1986; Betancourt 2015), i.e., there will always be some misspecification between the model and the actual data-generating process. Standard approximate Bayesian inference methods, e.g., Markov Chain Monte Carlo methods, can explicitly test for and deal with model misspecification because they benefit from access to the likelihood function of the model (Hogg et al. 2018; Gelman et al. 2020). However, for SBI methods, model misspecification can be more problematic because they rely on simulated data from the model. For example, when using artificial neural networks to approximate the posterior distribution, these approximations will be accurate only for the training data distribution. They can thus be highly inaccurate when probed with an x_o different from the training data (Quinonero-Candela et al. 2008; Szegedy et al. 2014).

In the section on **Simulation-based inference in practice**, I outlined currently available techniques to detect and cope with model misspecification in SBI (see, e.g., Frazier et al. 2019; Cannon et al. 2022). One of these techniques helped us refine the wiring rule simulator for the rat sensory cortex used in the second publication. We applied prior predictive checks to detect that our simulator could not reproduce the empirical data because it did not account for the observation noise due to the small sample size used in the experiments. We could resolve this mismatch by correcting the model, e.g., by incorporating the experimental subsampling process into the simulator (see appendix Boelts et al. 2023 for details).

Addressing the problem of model misspecification will be essential for the success and broader adoption of SBI methods. Over the past few years, the topic has received increasing attention. However, the resulting new methods for detecting (Schmitt et al. 2022; Frazier et al. 2019) and dealing with model misspecification (Frazier et al. 2020; Ward et al. 2022) in SBI have had only limited success and require additional research. Recently, the framework of generalized Bayesian inference has been proposed for dealing with model misspecification in SBI (Pacchiardi et al. 2022; Jewson et al. 2018; Knoblauch et al. 2019; Matsubara et al. 2022), suggesting a promising direction for future research.

5.3 Mind the gap: applicability of SBI methods

Simulation-based inference is an active field of research with many open challenges and new methods being proposed regularly. While tackling open technical challenges and proposing new methods is essential, it is also important to address the applicability of SBI methods, i.e., to close the gap between method development and application. On the one hand, this can be achieved by advocating the utility of SBI in domains where it has yet to be employed. For example, there are specific subfields of the computational sciences in which SBI has become a well-known technique for constraining models with observed data, e.g., in computational biology (Beaumont 2010; Csilléry et al. 2010), particle physics (Brehmer et al. 2020), astrophysics (Dax et al. 2021; Legin et al. 2022), or computational neuroscience (Gonçalves et al. 2020). Yet, in other fields, it is barely known. On the other hand, it is essential to establish generally accessible software tools and general guidelines and heuristics that enable researchers with little SBI experience to benefit from new SBI developments.

These challenges have—in parts—been addressed in this thesis: In the first publication, we improved the applicability of SBI by making it substantially more efficient. In the second publication,

we demonstrated the use of SBI in computational connectomics, providing clear guidelines on validating and interpreting SBI results. Lastly, the software package presented in the third publication established a central and well-accessible resource for applying SBI algorithms. Nevertheless, there are open challenges to the applicability of SBI (see, e.g., Hermans et al. 2021). One important next step will be to give an overview of these challenges and provide general guidelines for how to address them, e.g., by providing a practitioners' guide to SBI in the vein of Gelman et al. 2020.

Bibliography

- Arnst, M., G. Louppe, R. Van Hulle, L. Gillet, F. Bureau, and V. Denoël (2022). “A hybrid stochastic model and its Bayesian identification for infectious disease screening in a university campus with application to massive COVID-19 screening at the University of Liège”. In: *Mathematical Biosciences* 347, p. 108805. DOI: [10.1016/j.mbs.2022.108805](https://doi.org/10.1016/j.mbs.2022.108805) (cit. on p. 8).
- Bayes, Thomas (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”. In: *Philosophical Transactions of the Royal Society of London* 53, pp. 370–418. DOI: [10.1098/rstl.1763.0053](https://doi.org/10.1098/rstl.1763.0053) (cit. on p. 13).
- Beaumont, Mark (2010). “Approximate Bayesian Computation in Evolution and Ecology”. In: *Annual Review of Ecology, Evolution, and Systematics* 41, pp. 379–406. DOI: [10.1146/annurev-ecolsys-102209-144621](https://doi.org/10.1146/annurev-ecolsys-102209-144621) (cit. on pp. 8, 46).
- Beaumont, Mark, Jean-Marie Cornuet, Jean-Michel Marin, and Christian P. Robert (2009). “Adaptive approximate Bayesian computation”. In: *Biometrika* 96.4, pp. 983–990. DOI: [10.1093/biomet/asp052](https://doi.org/10.1093/biomet/asp052) (cit. on p. 15).
- Beaumont, Mark, Wenyang Zhang, and David J Balding (2002). “Approximate Bayesian Computation in Population Genetics”. In: *Genetics* 162.4, pp. 2025–2035. DOI: [10.1093/genetics/162.4.2025](https://doi.org/10.1093/genetics/162.4.2025) (cit. on pp. 7, 15, 16).
- Beck, Jonas, Michael Deistler, Yves Bernaerts, Jakob H. Macke, and Philipp Berens (2022). “Efficient identification of informative features in simulation-based inference”. In: *Advances in Neural Information Processing Systems*. DOI: [10.48550/arXiv.2210.11915](https://doi.org/10.48550/arXiv.2210.11915) (cit. on p. 25).
- Betancourt, Michael (2015). *A Unified Treatment of Predictive Model Comparison*. DOI: [10.48550/arXiv.1506.02273](https://doi.org/10.48550/arXiv.1506.02273) (cit. on pp. 9, 10, 46).
- Betancourt, Michael (2017). *A Conceptual Introduction to Hamiltonian Monte Carlo*. DOI: [10.48550/arxiv.1701.02434](https://doi.org/10.48550/arxiv.1701.02434) (cit. on p. 26).
- Bishop, Christopher M. (1994). *Mixture density networks*. Monograph (cit. on pp. 17, 25).
- Blei, David M., Alp Kucukelbir, and Jon D. McAuliffe (2017). “Variational Inference: A Review for Statisticians”. In: *Journal of the American Statistical Association* 112.518, pp. 859–877. DOI: [10.1080/01621459.2017.1285773](https://doi.org/10.1080/01621459.2017.1285773) (cit. on pp. 14, 26, 27).
- Blum, Michael G. B. and Olivier François (2010). “Non-linear regression models for Approximate Bayesian Computation”. In: *Statistics and Computing* 20.1, pp. 63–73. DOI: [10.1007/s11222-009-9116-0](https://doi.org/10.1007/s11222-009-9116-0) (cit. on p. 16).
- Boelts, Jan, Philipp Harth, Richard Gao, Daniel Udvary, Felipe Yanez, Daniel Baum, Hans-Christian Hege, Marcel Oberlaender, and Jakob H. Macke (2023). “Simulation-based Bayesian inference for efficient model identification in computational connectomics”. In: *bioRxiv*. DOI: [10.1101/2023.01.31.526269](https://doi.org/10.1101/2023.01.31.526269) (cit. on pp. 32, 34, 39, 40, 42, 46).

- Boelts, Jan, Jan-Matthis Lueckmann, Richard Gao, and Jakob H Macke (2022). “Flexible and efficient simulation-based inference for models of decision-making”. In: *eLife* 11, e77220. DOI: [10.7554/eLife.77220](https://doi.org/10.7554/eLife.77220) (cit. on pp. 24, 34, 36, 43).
- Bogacz, R., E. Brown, J. Moehlis, P. Holmes, and J. Cohen (2006). “The physics of optimal decision making: a formal analysis of models of performance in two-alternative forced-choice tasks.” English. In: *Psychological review* 113.4. DOI: [10.1037/0033-295x.113.4.700](https://doi.org/10.1037/0033-295x.113.4.700) (cit. on p. 35).
- Boge, Florian J. (2020). “How to infer explanations from computer simulations”. In: *Studies in History and Philosophy of Science Part A* 82, pp. 25–33. DOI: [10.1016/j.shpsa.2019.12.003](https://doi.org/10.1016/j.shpsa.2019.12.003) (cit. on p. 6).
- Box, George E P and Norman R Draper (1986). *Empirical model-building and response surface*. USA (cit. on pp. 10, 46).
- Brehmer, Johann and Kyle Cranmer (2020). *Simulation-based inference methods for particle physics*. DOI: [10.48550/arXiv.2010.06439](https://doi.org/10.48550/arXiv.2010.06439) (cit. on pp. 8, 46).
- Bruno, Randy M and Bert Sakmann (2006). “Cortex is driven by weak but synchronously active thalamocortical synapses”. In: *Science* 312.5780, pp. 1622–1627. DOI: [10.1126/science.1124593](https://doi.org/10.1126/science.1124593) (cit. on p. 41).
- Cannon, Patrick, Daniel Ward, and Sebastian M. Schmon (2022). *Investigating the Impact of Model Misspecification in Neural Simulation-based Inference*. DOI: [10.48550/arXiv.2209.01845](https://doi.org/10.48550/arXiv.2209.01845) (cit. on pp. 22, 46).
- Casella, George and Roger L Berger (2007). *Statistical inference* (cit. on p. 12).
- Chan, Jeffrey, Valerio Perrone, Jeffrey Spence, Paul Jenkins, Sara Mathieson, and Yun Song (2018). “A Likelihood-Free Inference Framework for Population Genetic Data using Exchangeable Neural Networks”. In: *Advances in Neural Information Processing Systems*. Vol. 31 (cit. on pp. 24, 36).
- Constantine, Paul (2015). *Active Subspaces: Emerging Ideas for Dimension Reduction in Parameter Studies* (cit. on p. 33).
- Constantinople, Christine M and Randy M Bruno (2013). “Deep cortical layers are activated directly by thalamus”. In: *Science* 340.6140, pp. 1591–1594. DOI: [10.1126/science.1236425](https://doi.org/10.1126/science.1236425) (cit. on p. 41).
- Cook, Samantha R, Andrew Gelman, and Donald B Rubin (2006). “Validation of Software for Bayesian Models Using Posterior Quantiles”. In: *Journal of Computational and Graphical Statistics* 15.3, pp. 675–692. DOI: [10.1198/106186006X136976](https://doi.org/10.1198/106186006X136976) (cit. on pp. 28, 43).
- Cranmer, Kyle, Johann Brehmer, and Gilles Louppe (2020). “The frontier of simulation-based inference”. In: *Proceedings of the National Academy of Sciences* 117.48, pp. 30055–30062. DOI: [10.1073/pnas.1912789117](https://doi.org/10.1073/pnas.1912789117) (cit. on pp. 7, 8, 14, 16, 45).
- Cranmer, Kyle, Juan Pavez, and Gilles Louppe (2016). *Approximating Likelihood Ratios with Calibrated Discriminative Classifiers*. DOI: [10.48550/arXiv.1506.02169](https://doi.org/10.48550/arXiv.1506.02169) (cit. on p. 19).
- Csilléry, Katalin, Michael G. B. Blum, Oscar E. Gaggiotti, and Olivier François (2010). “Approximate Bayesian Computation (ABC) in practice”. In: *Trends in Ecology & Evolution* 25.7, pp. 410–418. DOI: [10.1016/j.tree.2010.04.001](https://doi.org/10.1016/j.tree.2010.04.001) (cit. on pp. 8, 46).
- Dalmasso, N., T. Pospisil, A. B. Lee, R. Izbicki, P. E. Freeman, and A. I. Malz (2020). “Conditional density estimation tools in python and R with applications to photometric redshifts and likelihood-free cosmological inference”. In: *Astronomy and Computing* 30, p. 100362. DOI: [10.1016/j.ascom.2019.100362](https://doi.org/10.1016/j.ascom.2019.100362) (cit. on p. 30).
- Dax, M., S. R. Green, J. Gair, M. Deistler, B. Schölkopf, and J. H. Macke (2022). “Group equivariant neural posterior estimation”. In: *10th International Conference on Learning Representations (ICLR)* (cit. on p. 24).

- Dax, M., Stephen R. Green, Jonathan Gair, Jakob H. Macke, Alessandra Buonanno, and Bernhard Schölkopf (2021). “Real-Time Gravitational Wave Science with Neural Posterior Estimation”. In: *Physical Review Letters* 127.24, p. 241103. DOI: [10.1103/PhysRevLett.127.241103](https://doi.org/10.1103/PhysRevLett.127.241103) (cit. on pp. 8, 46).
- Deistler, Michael, Pedro J. Goncalves, and Jakob H. Macke (2022a). “Truncated proposals for scalable and hassle-free simulation-based inference”. In: *Advances in Neural Information Processing Systems*. DOI: [10.48550/arXiv.2210.04815](https://doi.org/10.48550/arXiv.2210.04815) (cit. on pp. 18, 30, 43).
- Deistler, Michael, Jakob H. Macke, and Pedro J. Gonçalves (2022b). “Energy-efficient network activity from disparate circuit parameters”. In: *Proceedings of the National Academy of Sciences* 119.44, e2207632119. DOI: [10.1073/pnas.2207632119](https://doi.org/10.1073/pnas.2207632119) (cit. on pp. 24, 31–33, 43).
- Del Moral, Pierre, Arnaud Doucet, and Ajay Jasra (2012). “An adaptive sequential Monte Carlo method for approximate Bayesian computation”. In: *Statistics and Computing* 22.5, pp. 1009–1020. DOI: [10.1007/s11222-011-9271-y](https://doi.org/10.1007/s11222-011-9271-y) (cit. on p. 15).
- Delaunoy, Arnaud, Joeri Hermans, François Rozet, Antoine Wehenkel, and Gilles Louppe (2022). “Towards Reliable Simulation-Based Inference with Balanced Neural Ratio Estimation”. In: *Advances in Neural Information Processing Systems*. DOI: [10.48550/arxiv.2208.13624](https://doi.org/10.48550/arxiv.2208.13624) (cit. on p. 43).
- Diggle, Peter J. and Richard J. Gratton (1984). “Monte Carlo Methods of Inference for Implicit Statistical Models”. In: *Journal of the Royal Statistical Society: Series B (Methodological)* 46.2, pp. 193–212. DOI: [10.1111/j.2517-6161.1984.tb01290.x](https://doi.org/10.1111/j.2517-6161.1984.tb01290.x) (cit. on pp. 7, 15).
- Durkan, Conor, Artur Bekasov, Iain Murray, and George Papamakarios (2019). “Neural spline flows”. In: *Advances in Neural Information Processing Systems*, pp. 7509–7520. DOI: [10.48550/arxiv.1906.04032](https://doi.org/10.48550/arxiv.1906.04032) (cit. on pp. 37, 41).
- Durkan, Conor, Iain Murray, and George Papamakarios (2020). “On Contrastive Learning for Likelihood-free Inference”. In: *Proceedings of the 37th International Conference on Machine Learning*, pp. 2771–2781. DOI: [10.48550/arxiv.2002.03712](https://doi.org/10.48550/arxiv.2002.03712) (cit. on pp. 19, 20, 30, 43).
- Dyer, Joel, Patrick Cannon, J. Doyne Farmer, and Sebastian Schmon (2022). *Black-box Bayesian inference for economic agent-based models*. DOI: [10.48550/arXiv.2202.00625](https://doi.org/10.48550/arXiv.2202.00625) (cit. on p. 8).
- Fengler, Alexander, Lakshmi N Govindarajan, Tony Chen, and Michael J Frank (2021). “Likelihood approximation networks (LANs) for fast inference of simulation models in cognitive neuroscience”. In: *eLife* 10, e65074. DOI: [10.7554/eLife.65074](https://doi.org/10.7554/eLife.65074) (cit. on pp. 24, 37).
- Fishburn, Peter (1994). “A variational model of preference under uncertainty”. In: *Journal of Risk and Uncertainty* 8.2, pp. 127–152. DOI: [10.1007/BF01065369](https://doi.org/10.1007/BF01065369) (cit. on p. 12).
- Frazier, David T. and Christopher Drovandi (2020). *Robust Approximate Bayesian Inference with Synthetic Likelihood* (cit. on pp. 22, 46).
- Frazier, David T., Christian P. Robert, and Judith Rousseau (2019). *Model Misspecification in ABC: Consequences and Diagnostics*. DOI: [10.1111/369--7412/20/82421](https://doi.org/10.1111/369--7412/20/82421) (cit. on pp. 22, 46).
- Geffner, Tomas, George Papamakarios, and Andriy Mnih (2022). *Score Modeling for Simulation-based Inference*. DOI: [10.48550/arxiv.2209.14249](https://doi.org/10.48550/arxiv.2209.14249) (cit. on p. 25).
- Gelman, Andrew, Aki Vehtari, Daniel Simpson, Charles C. Margossian, Bob Carpenter, Yuling Yao, Lauren Kennedy, Jonah Gabry, Paul-Christian Bürkner, and Martin Modrák (2020). *Bayesian Workflow*. DOI: [10.48550/arXiv.2011.01808](https://doi.org/10.48550/arXiv.2011.01808) (cit. on pp. 21, 28, 46, 47).
- Germain, Mathieu, Karol Gregor, Iain Murray, and Hugo Larochelle (2015). “MADE: Masked Autoencoder for Distribution Estimation”. In: *Proceedings of the 32nd International Conference on Machine Learning*, pp. 881–889. DOI: [10.48550/arxiv.1502.03509](https://doi.org/10.48550/arxiv.1502.03509) (cit. on p. 7).
- Gershman, Samuel and Noah Goodman (2014). “Amortized Inference in Probabilistic Reasoning”. In: *Proceedings of the Annual Meeting of the Cognitive Science Society* 36.36 (cit. on p. 18).

- Glavovic, Bruce C., Timothy F. Smith, and Iain White (2022). “The tragedy of climate change science”. In: *Climate and Development* 14.9, pp. 829–833. DOI: [10.1080/17565529.2021.2008855](https://doi.org/10.1080/17565529.2021.2008855) (cit. on pp. 6, 45).
- Glöckler, Manuel, Michael Deistler, and Jakob H. Macke (2022). “Variational methods for simulation-based inference”. In: *International Conference on Learning Representations*. DOI: [10.48550/arxiv.2203.04176](https://doi.org/10.48550/arxiv.2203.04176) (cit. on pp. 20, 26, 27, 43).
- Gold, Joshua I. and Michael N. Shadlen (2007). “The neural basis of decision making”. In: *Annual Review of Neuroscience* 30, pp. 535–574. DOI: [10.1146/annurev.neuro.29.051605.113038](https://doi.org/10.1146/annurev.neuro.29.051605.113038) (cit. on pp. 11, 35).
- Golowasch, Jorge, Mark S. Goldman, L. F. Abbott, and Eve Marder (2002). “Failure of Averaging in the Construction of a Conductance-Based Neuron Model”. In: *Journal of Neurophysiology* 87.2, pp. 1129–1131. DOI: [10.1152/jn.00412.2001](https://doi.org/10.1152/jn.00412.2001) (cit. on p. 31).
- Gonçalves, Pedro J, Jan-Matthis Lueckmann, 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 (2020). “Training deep neural density estimators to identify mechanistic models of neural dynamics”. In: *eLife* 9. Ed. by John R Huguenard, Timothy O’Leary, and Mark S Goldman, e56261. DOI: [10.7554/eLife.56261](https://doi.org/10.7554/eLife.56261) (cit. on pp. 8, 24, 31, 32, 46).
- Greenberg, David, Marcel Nonnenmacher, and Jakob Macke (2019). “Automatic Posterior Transformation for Likelihood-Free Inference”. In: *Proceedings of the 36th International Conference on Machine Learning*, pp. 2404–2414. DOI: [10.48550/arxiv.1905.07488](https://doi.org/10.48550/arxiv.1905.07488) (cit. on pp. 18, 24, 25, 30, 41, 43).
- Groschner, Lukas N., Jonatan G. Malis, Birte Zuidinga, and Alexander Borst (2022). “A biophysical account of multiplication by a single neuron”. In: *Nature* 603.7899, pp. 119–123. DOI: [10.1038/s41586-022-04428-3](https://doi.org/10.1038/s41586-022-04428-3) (cit. on p. 8).
- Gutenkunst, Ryan N., Joshua J. Waterfall, Fergal P. Casey, Kevin S. Brown, Christopher R. Myers, and James P. Sethna (2007). “Universally Sloppy Parameter Sensitivities in Systems Biology Models”. In: *PLOS Computational Biology* 3.10, e189. DOI: [10.1371/journal.pcbi.0030189](https://doi.org/10.1371/journal.pcbi.0030189) (cit. on p. 31).
- Hartmann, Stephan (1996). “The World as a Process”. In: *Modelling and Simulation in the Social Sciences from the Philosophy of Science Point of View*. Ed. by Rainer Hegselmann, Ulrich Mueller, and Klaus G. Troitzsch. Theory and Decision Library. Dordrecht, pp. 77–100. DOI: [10.1007/978-94-015-8686-3_5](https://doi.org/10.1007/978-94-015-8686-3_5) (cit. on p. 6).
- Hashemi, Meysam, Anirudh N. Vattikonda, Jayant Jha, Viktor Sip, Marmaduke M. Woodman, Fabrice Bartolomei, and Viktor K. Jirsa (2022). *Simulation-Based Inference for Whole-Brain Network Modeling of Epilepsy using Deep Neural Density Estimators*. DOI: [10.1101/2022.06.02.22275860](https://doi.org/10.1101/2022.06.02.22275860) (cit. on p. 8).
- Hastings, W. K. (1970). “Monte Carlo sampling methods using Markov chains and their applications”. In: *Biometrika* 57.1, pp. 97–109. DOI: [10.1093/biomet/57.1.97](https://doi.org/10.1093/biomet/57.1.97) (cit. on pp. 14, 26).
- Hermans, Joeri, Volodimir Begy, and Gilles Louppe (2020). “Likelihood-free MCMC with Amortized Approximate Ratio Estimators”. In: *Proceedings of the 37th International Conference on Machine Learning*, pp. 4239–4248 (cit. on pp. 19, 20, 24, 30, 43).
- Hermans, Joeri, Arnaud Delaunoy, François Rozet, Antoine Wehenkel, and Gilles Louppe (2021). *Averting A Crisis In Simulation-Based Inference*. DOI: [10.48550/arXiv.2110.06581](https://doi.org/10.48550/arXiv.2110.06581) (cit. on pp. 30, 47).

- Hogg, David W. and Daniel Foreman-Mackey (2018). “Data Analysis Recipes: Using Markov Chain Monte Carlo*”. In: *The Astrophysical Journal Supplement Series* 236.1, p. 11. DOI: [10.3847/1538-4365/aab76e](https://doi.org/10.3847/1538-4365/aab76e) (cit. on pp. 14, 26, 46).
- Homan, Matthew D. and Andrew Gelman (2014). “The No-U-turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo”. In: *The Journal of Machine Learning Research* 15.1, pp. 1593–1623 (cit. on p. 26).
- Izbicki, Rafael, Ann Lee, and Chad Schafer (2014). “High-Dimensional Density Ratio Estimation with Extensions to Approximate Likelihood Computation”. In: *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics*, pp. 420–429. DOI: [10.48550/arxiv.1404.7063](https://doi.org/10.48550/arxiv.1404.7063) (cit. on p. 19).
- Jewson, Jack, Jim Q. Smith, and Chris Holmes (2018). “Principles of Bayesian Inference Using General Divergence Criteria”. In: *Entropy* 20.6, p. 442. DOI: [10.3390/e20060442](https://doi.org/10.3390/e20060442) (cit. on p. 46).
- Kangasrääsio, Antti, Jussi PP Jokinen, Antti Oulasvirta, Andrew Howes, and Samuel Kaski (2019). “Parameter inference for computational cognitive models with Approximate Bayesian Computation”. In: *Cognitive science* 43.6, e12738. DOI: [10.1111/cogs.12738](https://doi.org/10.1111/cogs.12738) (cit. on p. 35).
- Kelly, Ryan P., David J. Nott, David T. Frazier, David J. Warne, and Chris Drovandi (2023). *Misspecification-robust Sequential Neural Likelihood*. DOI: [10.48550/arXiv.2301.13368](https://doi.org/10.48550/arXiv.2301.13368) (cit. on p. 22).
- Knoblauch, Jeremias, Jack Jewson, and Theodoros Damoulas (2019). *Generalized Variational Inference: Three arguments for deriving new Posteriors*. DOI: [10.48550/arXiv.1904.02063](https://doi.org/10.48550/arXiv.1904.02063) (cit. on p. 46).
- Kolmogorov, Andrey (1933). “Sulla determinazione empirica di una legge di distribuzione”. In: *Inst. Ital. Attuari, Giorn.* 4, pp. 83–91 (cit. on p. 29).
- Koopman, B. O. (1936). “On distributions admitting a sufficient statistic”. en. In: *Transactions of the American Mathematical Society* 39.3, pp. 399–409. DOI: [10.1090/S0002-9947-1936-1501854-3](https://doi.org/10.1090/S0002-9947-1936-1501854-3) (cit. on p. 14).
- Latimer, Kenneth W., Jacob L. Yates, Miriam L. R. Meister, Alexander C. Huk, and Jonathan W. Pillow (2015). “Single-trial spike trains in parietal cortex reveal discrete steps during decision-making”. In: *Science* 349.6244, pp. 184–187. DOI: [10.1126/science.aaa4056](https://doi.org/10.1126/science.aaa4056) (cit. on p. 11).
- Lavin, Alexander, David Krakauer, Hector Zenil, Justin Gottschlich, Tim Mattson, Johann Brehmer, Anima Anandkumar, Sanjay Choudry, Kamil Rocki, Atılım Güneş Baydin, Carina Prunkl, Brooks Paige, Olexandr Isayev, Erik Peterson, Peter L. McMahon, Jakob Macke, Kyle Cranmer, Jiaxin Zhang, Haruko Wainwright, Adi Hanuka, Manuela Veloso, Samuel Assefa, Stephan Zheng, and Avi Pfeffer (2022). *Simulation Intelligence: Towards a New Generation of Scientific Methods*. DOI: [10.48550/arXiv.2112.03235](https://doi.org/10.48550/arXiv.2112.03235) (cit. on p. 45).
- Le, Tuan Anh, Atılım Güneş Baydin, Robert Zinkov, and Frank Wood (2017). “Using synthetic data to train neural networks is model-based reasoning”. In: *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 3514–3521. DOI: [10.1109/IJCNN.2017.7966298](https://doi.org/10.1109/IJCNN.2017.7966298) (cit. on p. 17).
- LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton (2015). “Deep learning”. In: *Nature* 521.7553, pp. 436–444. DOI: [10.1038/nature14539](https://doi.org/10.1038/nature14539) (cit. on p. 7).
- Legin, Ronan, Yashar Hezaveh, Laurence Perreault Levasseur, and Benjamin Wandelt (2022). *Simulation-Based Inference of Strong Gravitational Lensing Parameters*. DOI: [10.48550/arXiv.2112.05278](https://doi.org/10.48550/arXiv.2112.05278) (cit. on p. 46).
- Lewis, Dyani (2022). “What scientists have learnt from COVID lockdowns”. In: *Nature* 609.7926, pp. 236–239. DOI: [10.1038/d41586-022-02823-4](https://doi.org/10.1038/d41586-022-02823-4) (cit. on p. 6).
- Lueckmann, Jan-Matthis, Giacomo Bassetto, Theofanis Karaletsos, and Jakob H. Macke (2019). “Likelihood-free inference with emulator networks”. In: *Proceedings of The 1st Symposium on*

- Advances in Approximate Bayesian Inference*, pp. 32–53. DOI: [10.48550/arxiv.1805.09294](https://doi.org/10.48550/arxiv.1805.09294) (cit. on p. 19).
- Lueckmann, Jan-Matthis, Jan Boelts, David Greenberg, Pedro Goncalves, and Jakob Macke (2021). “Benchmarking Simulation-Based Inference”. In: *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, pp. 343–351. DOI: [10.48550/arXiv.2101.04653](https://doi.org/10.48550/arXiv.2101.04653) (cit. on pp. 16, 18, 21, 24, 26, 30).
- Lueckmann, Jan-Matthis, Pedro J Goncalves, Giacomo Bassetto, Kaan Öcal, Marcel Nonnenmacher, and Jakob H Macke (2017). “Flexible statistical inference for mechanistic models of neural dynamics”. In: *Advances in Neural Information Processing Systems*. Vol. 30. DOI: [10.48550/arxiv.1711.01861](https://doi.org/10.48550/arxiv.1711.01861) (cit. on pp. 18, 24, 43).
- Marjoram, Paul, John Molitor, Vincent Plagnol, and Simon Tavaré (2003). “Markov chain Monte Carlo without likelihoods”. In: *Proceedings of the National Academy of Sciences* 100.26, pp. 15324–15328. DOI: [10.1073/pnas.0306899100](https://doi.org/10.1073/pnas.0306899100) (cit. on p. 15).
- Masserano, Luca, Tommaso Dorigo, Rafael Izbicki, Mikael Kuusela, and Ann B. Lee (2022). *Simulation-Based Inference with WALDO: Perfectly Calibrated Confidence Regions Using Any Prediction or Posterior Estimation Algorithm*. Tech. rep. arXiv:2205.15680. arXiv. DOI: [10.48550/arxiv.2205.15680](https://doi.org/10.48550/arxiv.2205.15680) (cit. on p. 30).
- Massimi, Michela and Wahid Bhimji (2015). “Computer simulations and experiments: The case of the Higgs boson”. In: *Studies in History and Philosophy of Science Part B: Studies in History and Philosophy of Modern Physics* 51, pp. 71–81. DOI: [10.1016/j.shpsb.2015.06.003](https://doi.org/10.1016/j.shpsb.2015.06.003) (cit. on p. 6).
- Matsubara, Takuo, Jeremias Knoblauch, François-Xavier Briol, and Chris J. Oates (2022). *Robust Generalised Bayesian Inference for Intractable Likelihoods*. DOI: [10.48550/arXiv.2104.07359](https://doi.org/10.48550/arXiv.2104.07359) (cit. on p. 46).
- Meeds, Edward and Max Welling (2014). “GPS-ABC: Gaussian process surrogate approximate Bayesian computation”. In: *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence*. UAI’14. Arlington, Virginia, USA, pp. 593–602. DOI: [10.48550/arxiv.1401.2838](https://doi.org/10.48550/arxiv.1401.2838) (cit. on p. 25).
- Metropolis, Nicholas, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller (1953). “Equation of State Calculations by Fast Computing Machines”. In: *The Journal of Chemical Physics* 21.6, pp. 1087–1092. DOI: [10.1063/1.1699114](https://doi.org/10.1063/1.1699114) (cit. on pp. 14, 26).
- Miller, Benjamin K, Alex Cole, Patrick Forré, Gilles Louppe, and Christoph Weniger (2021). “Truncated Marginal Neural Ratio Estimation”. In: *Advances in Neural Information Processing Systems*. Vol. 34, pp. 129–143 (cit. on pp. 30, 43).
- Miller, Benjamin K, Christoph Weniger, and Patrick Forré (2022). “Contrastive Neural Ratio Estimation”. In: *Advances in Neural Information Processing Systems*. DOI: [10.48550/arxiv.2210.06170](https://doi.org/10.48550/arxiv.2210.06170) (cit. on p. 43).
- Mishra-Sharma, Siddharth and Kyle Cranmer (2022). “Neural simulation-based inference approach for characterizing the Galactic Center gamma-ray excess”. In: *Physical Review D* 105.6, p. 063017. DOI: [10.1103/PhysRevD.105.063017](https://doi.org/10.1103/PhysRevD.105.063017) (cit. on p. 8).
- Modrák, Martin, Angie H. Moon, Shinyoung Kim, Paul Bürkner, Niko Huurre, Kateřina Faltejsková, Andrew Gelman, and Aki Vehtari (2022). *Simulation-Based Calibration Checking for Bayesian Computation: The Choice of Test Quantities Shapes Sensitivity*. DOI: [10.48550/arXiv.2211.02383](https://doi.org/10.48550/arXiv.2211.02383) (cit. on p. 30).
- Mohamed, Shakir and Balaji Lakshminarayanan (2017). *Learning in Implicit Generative Models*. DOI: [10.48550/arXiv.1610.03483](https://doi.org/10.48550/arXiv.1610.03483) (cit. on p. 7).

- Muratore, Fabio, Fabio Ramos, Greg Turk, Wenhao Yu, Michael Gienger, and Jan Peters (2022). “Robot Learning From Randomized Simulations: A Review”. In: *Frontiers in Robotics and AI* 9, p. 799893. DOI: [10.3389/frobt.2022.799893](https://doi.org/10.3389/frobt.2022.799893) (cit. on p. 8).
- Nalisnick, Eric, Akihiro Matsukawa, Yee Whye Teh, Dilan Gorur, and Balaji Lakshminarayanan (2022). “Do Deep Generative Models Know What They Don’t Know?”. In: DOI: [10.48550/arxiv.1810.09136](https://doi.org/10.48550/arxiv.1810.09136) (cit. on p. 22).
- Neal, Radford M et al. (2011). “MCMC using Hamiltonian dynamics”. In: *Handbook of markov chain monte carlo* 2.11, p. 2 (cit. on p. 26).
- Neal, Radford M. (2003). “Slice sampling”. In: *The Annals of Statistics* 31.3, pp. 705–767. DOI: [10.1214/aos/1056562461](https://doi.org/10.1214/aos/1056562461) (cit. on p. 26).
- Pacchiardi, Lorenzo and Ritabrata Dutta (2022). *Generalized Bayesian Likelihood-Free Inference Using Scoring Rules Estimators*. DOI: [10.48550/arXiv.2104.03889](https://doi.org/10.48550/arXiv.2104.03889) (cit. on p. 46).
- Paige, Brooks and Frank Wood (2016). “Inference Networks for Sequential Monte Carlo in Graphical Models”. In: *Proceedings of The 33rd International Conference on Machine Learning*, pp. 3040–3049 (cit. on pp. 17, 18).
- Papamakarios, George and Iain Murray (2016). “Fast epsilon-free Inference of Simulation Models with Bayesian Conditional Density Estimation”. In: *Advances in Neural Information Processing Systems*. Vol. 29. DOI: [10.48550/arxiv.1605.06376](https://doi.org/10.48550/arxiv.1605.06376) (cit. on pp. 16–18, 25, 30, 41, 43).
- Papamakarios, George, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan (2021). “Normalizing Flows for Probabilistic Modeling and Inference”. In: *Journal of Machine Learning Research* 22.57, pp. 1–64 (cit. on pp. 18, 25, 37).
- Papamakarios, George, Theo Pavlakou, and Iain Murray (2017). “Masked Autoregressive Flow for Density Estimation”. In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Vol. 30. DOI: [10.48550/arxiv.1705.07057](https://doi.org/10.48550/arxiv.1705.07057) (cit. on p. 25).
- Papamakarios, George, David Sterratt, and Iain Murray (2019). “Sequential Neural Likelihood: Fast Likelihood-free Inference with Autoregressive Flows”. In: *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, pp. 837–848 (cit. on pp. 19, 25, 30, 43).
- Paszke, Adam, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala (2019). “PyTorch: An Imperative Style, High-Performance Deep Learning Library”. In: *Advances in Neural Information Processing Systems* 32, pp. 8024–8035. DOI: [10.48550/arxiv.1912.01703](https://doi.org/10.48550/arxiv.1912.01703) (cit. on p. 26).
- Peng, Wei, Gokul Iyer, Valentina Bosetti, Vaibhav Chaturvedi, James Edmonds, Allen Fawcett, Stephane Hallegatte, David G. Victor, Detlef van Vuuren, and John Weyant (2021). “Climate policy models need to get real about people — here’s how”. In: *Nature* 594.7862, pp. 174–176. DOI: [10.1038/d41586-021-01500-2](https://doi.org/10.1038/d41586-021-01500-2) (cit. on pp. 6, 45).
- 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.” In: *Molecular Biology and Evolution* 16.12, pp. 1791–1798. DOI: [10.1093/oxfordjournals.molbev.a026091](https://doi.org/10.1093/oxfordjournals.molbev.a026091) (cit. on p. 15).
- Quinonero-Candela, Joaquin, Masashi Sugiyama, Anton Schwaighofer, and Neil D Lawrence (2008). *Dataset shift in machine learning* (cit. on p. 46).
- Radev, Stefan T., Ulf K. Mertens, Andreas Voss, Lynton Ardizzone, and Ullrich Köthe (2022). “BayesFlow: Learning Complex Stochastic Models With Invertible Neural Networks”. In: *IEEE*

- Transactions on Neural Networks and Learning Systems* 33.4, pp. 1452–1466. DOI: [10.1109/TNNLS.2020.3042395](https://doi.org/10.1109/TNNLS.2020.3042395) (cit. on pp. 24, 36).
- Ramesh, Poornima, Jan-Matthis Lueckmann, Jan Boelts, Álvaro Tejero-Cantero, David S. Greenberg, Pedro J. Goncalves, and Jakob H. Macke (2022). “GATSBI: Generative Adversarial Training for Simulation-Based Inference”. In: *International Conference on Learning Representations*. DOI: [10.48550/arxiv.2203.06481](https://doi.org/10.48550/arxiv.2203.06481) (cit. on p. 24).
- Ratcliff, Roger (1978). “A Theory of Memory Retrieval”. In: *Psychological Review* 85.2, pp. 59–108. DOI: [10.1037/0033-295x.85.2.59](https://doi.org/10.1037/0033-295x.85.2.59) (cit. on pp. 10, 11, 35).
- Ratcliff, Roger and Gail McKoon (2008). “The Diffusion Decision Model: Theory and Data for Two-Choice Decision Tasks”. In: *Neural computation* 20.4, pp. 873–922. DOI: [10.1162/neco.2008.12-06-420](https://doi.org/10.1162/neco.2008.12-06-420) (cit. on p. 23).
- Rezende, Danilo and Shakir Mohamed (2015). “Variational Inference with Normalizing Flows”. In: *Proceedings of the 32nd International Conference on Machine Learning*, pp. 1530–1538 (cit. on p. 18).
- Roitman, Jamie D. and Michael N. Shadlen (2002). “Response of Neurons in the Lateral Intraparietal Area during a Combined Visual Discrimination Reaction Time Task”. In: *Journal of Neuroscience* 22.21, pp. 9475–9489. DOI: [10.1523/JNEUROSCI.22-21-09475.2002](https://doi.org/10.1523/JNEUROSCI.22-21-09475.2002) (cit. on p. 11).
- Rubin, Donald B. (1984). “Bayesianly Justifiable and Relevant Frequency Calculations for the Applied Statistician”. In: *The Annals of Statistics* 12.4, pp. 1151–1172. DOI: [10.1214/aos/1176346785](https://doi.org/10.1214/aos/1176346785) (cit. on p. 14).
- Schad, Daniel J, Michael Betancourt, and Shravan Vasishth (2021). “Toward a principled Bayesian workflow in cognitive science.” In: *Psychological methods* 26.1, p. 103 (cit. on p. 35).
- Schmitt, Marvin, Paul-Christian Bürkner, Ullrich Köthe, and Stefan T. Radev (2022). *Detecting Model Misspecification in Amortized Bayesian Inference with Neural Networks*. DOI: [10.48550/arXiv.2112.08866](https://doi.org/10.48550/arXiv.2112.08866) (cit. on pp. 22, 23, 46).
- Shadlen, Michael N. and Roozbeh Kiani (2013). “Decision making as a window on cognition”. eng. In: *Neuron* 80.3, pp. 791–806. DOI: [10.1016/j.neuron.2013.10.047](https://doi.org/10.1016/j.neuron.2013.10.047) (cit. on pp. 11, 23).
- Shapson-Coe, Alexander, Michał Januszewski, Daniel R. Berger, Art Pope, Yuelong Wu, Tim Blakely, Richard L. Schalek, Peter Li, Shuohong Wang, Jeremy Maitin-Shepard, Neha Karlupia, Sven Dorkenwald, Evelina Sjostedt, Laramie Leavitt, Dongil Lee, Luke Bailey, Angerica Fitzmaurice, Rohin Kar, Benjamin Field, Hank Wu, Julian Wagner-Carena, David Aley, Joanna Lau, Zudi Lin, Donglai Wei, Hanspeter Pfister, Adi Peleg, Viren Jain, and Jeff W. Lichtman (2021). “A connectomic study of a petascale fragment of human cerebral cortex”. In: *bioRxiv*. DOI: [10.1101/2021.05.29.446289](https://doi.org/10.1101/2021.05.29.446289) (cit. on pp. 8, 42).
- Sharrock, Louis, Jack Simons, Song Liu, and Mark Beaumont (2022). *Sequential Neural Score Estimation: Likelihood-Free Inference with Conditional Score Based Diffusion Models*. DOI: [10.48550/arXiv.2210.04872](https://doi.org/10.48550/arXiv.2210.04872) (cit. on p. 25).
- Shiffrin, Richard M., Michael D. Lee, Woojae Kim, and Eric-Jan Wagenmakers (2008). “A Survey of Model Evaluation Approaches With a Tutorial on Hierarchical Bayesian Methods”. In: *Cognitive Science* 32.8, pp. 1248–1284. DOI: [10.1080/03640210802414826](https://doi.org/10.1080/03640210802414826) (cit. on pp. 27, 35).
- Shinn, Maxwell, Norman H Lam, and John D Murray (2020). “A flexible framework for simulating and fitting generalized drift-diffusion models”. en. In: *eLife* 9, e56938. DOI: [10.7554/eLife.56938](https://doi.org/10.7554/eLife.56938) (cit. on p. 35).
- Sisson, Scott, Yanan Fan, and Mark M. Tanaka (2007). “Sequential Monte Carlo without likelihoods”. In: *Proceedings of the National Academy of Sciences* 104.6, pp. 1760–1765. DOI: [10.1073/pnas.0607208104](https://doi.org/10.1073/pnas.0607208104) (cit. on p. 15).

- Sisson, Scott, Yanan. Fan, and Mark Beaumont (2018). “Overview of ABC”. In: *Handbook of Approximate Bayesian Computation* (cit. on pp. 7, 14, 15).
- Smirnov, N. (1948). “Table for Estimating the Goodness of Fit of Empirical Distributions”. In: *The Annals of Mathematical Statistics* 19.2, pp. 279–281. DOI: [10.1214/aoms/1177730256](https://doi.org/10.1214/aoms/1177730256) (cit. on p. 29).
- Smith, Philip L. and Roger Ratcliff (2004). “Psychology and neurobiology of simple decisions”. English. In: *Trends in Neurosciences* 27.3, pp. 161–168. DOI: [10.1016/j.tins.2004.01.006](https://doi.org/10.1016/j.tins.2004.01.006) (cit. on p. 10).
- Song, Yang and Stefano Ermon (2019). “Generative Modeling by Estimating Gradients of the Data Distribution”. In: *Advances in Neural Information Processing Systems*. Vol. 32 (cit. on p. 25).
- Spooner, Fiona, Jesse F. Abrams, Karyn Morrissey, Gavin Shaddick, Michael Batty, Richard Milton, Adam Dennett, Nik Lomax, Nick Malleson, Natalie Nelissen, Alex Coleman, Jamil Nur, Ying Jin, Rory Greig, Charlie Shenton, and Mark Birkin (2021). “A dynamic microsimulation model for epidemics”. In: *Social Science & Medicine* 291, p. 114461. DOI: [10.1016/j.socscimed.2021.114461](https://doi.org/10.1016/j.socscimed.2021.114461) (cit. on p. 6).
- Sunnåker, Mikael, Alberto Giovanni Busetto, Elina Numminen, Jukka Corander, Matthieu Foll, and Christophe Dessimoz (2013). “Approximate Bayesian Computation”. In: *PLOS Computational Biology* 9.1, e1002803. DOI: [10.1371/journal.pcbi.1002803](https://doi.org/10.1371/journal.pcbi.1002803) (cit. on pp. 7, 14).
- Szegedy, Christian, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus (2014). *Intriguing properties of neural networks*. DOI: [10.48550/arXiv.1312.6199](https://doi.org/10.48550/arXiv.1312.6199) (cit. on p. 46).
- Talts, Sean, Michael Betancourt, Daniel Simpson, Aki Vehtari, and Andrew Gelman (2020). *Validating Bayesian Inference Algorithms with Simulation-Based Calibration*. DOI: [10.48550/arXiv.1804.06788](https://doi.org/10.48550/arXiv.1804.06788) (cit. on pp. 28–30, 41, 43).
- Tavaré, Simon, David J Balding, R C Griffiths, and Peter Donnelly (1997). “Inferring Coalescence Times From DNA Sequence Data”. In: *Genetics* 145.2, pp. 505–518. DOI: [10.1093/genetics/145.2.505](https://doi.org/10.1093/genetics/145.2.505) (cit. on p. 15).
- Tavares, Gabriela, Pietro Perona, and Antonio Rangel (2017). “The Attentional Drift Diffusion Model of Simple Perceptual Decision-Making”. In: *Frontiers in Neuroscience* 11. DOI: [10.3389/fnins.2017.00468](https://doi.org/10.3389/fnins.2017.00468) (cit. on p. 35).
- Tejero-Cantero*, Alvaro, Jan Boelts*, Michael Deistler*, Jan-Matthis Lueckmann*, Conor Durkan*, Pedro J. Gonçalves, David S. Greenberg, and Jakob H. Macke (2020). “sbi: A toolkit for simulation-based inference”. en. In: *Journal of Open Source Software* 5.52, p. 2505. DOI: [10.21105/joss.02505](https://doi.org/10.21105/joss.02505) (cit. on pp. 26, 34, 43).
- Thomas, Owen, Ritabrata Dutta, Jukka Corander, Samuel Kaski, and Michael U. Gutmann (2022). “Likelihood-Free Inference by Ratio Estimation”. In: *Bayesian Analysis* 17.1, pp. 1–31. DOI: [10.1214/20-BA1238](https://doi.org/10.1214/20-BA1238) (cit. on p. 19).
- Toni, Tina (2010). “Approximate Bayesian computation for parameter inference and model selection in systems biology”. eng. Ph.D. Imperial College London (cit. on p. 15).
- Toni, Tina, David Welch, Natalja Strelkowa, Andreas Ipsen, and Michael P.H Stumpf (2009). “Approximate Bayesian computation scheme for parameter inference and model selection in dynamical systems”. In: *Journal of The Royal Society Interface* 6.31, pp. 187–202. DOI: [10.1098/rsif.2008.0172](https://doi.org/10.1098/rsif.2008.0172) (cit. on p. 15).
- Turner, Nicholas L, Thomas Macrina, J Alexander Bae, Runzhe Yang, Alyssa M Wilson, Casey Schneider-Mizell, Kisuk Lee, Ran Lu, Jingpeng Wu, Agnes L Bodor, Adam A Bleckert, Derrick Brittain, Emmanouil Froudarakis, Sven Dorkenwald, Forrest Collman, Nico Kemnitz, Dodam Ih, William M Silversmith, Jonathan Zung, Aleksandar Zlateski, Ignacio Tartavull, Szi-chieh Yu,

- Sergiy Popovych, Shang Mu, William Wong, Chris S Jordan, Manuel Castro, JoAnn Buchanan, Daniel J Bumbarger, Marc Takeno, Russel Torres, Gayathri Mahalingam, Leila Elabbady, Yang Li, Erick Cobos, Pengcheng Zhou, Shelby Suckow, Lynne Becker, Liam Paninski, Franck Polleux, Jacob Reimer, Andreas S Tolia, R Clay Reid, Nuno Maçarico da Costa, and H Sebastian Seung (2022). “Reconstruction of neocortex: Organelles, compartments, cells, circuits, and activity”. In: *Cell* 185 (6), 1082–1100.e24. DOI: [10.1016/j.cell.2022.01.023](https://doi.org/10.1016/j.cell.2022.01.023) (cit. on p. 42).
- Udvary, Daniel, Philipp Harth, Jakob H. Macke, Hans-Christian Hege, Christiaan P. J. de Kock, Bert Sakmann, and Marcel Oberlaender (2022). “The impact of neuron morphology on cortical network architecture”. eng. In: *Cell Reports* 39.2, p. 110677. DOI: [10.1016/j.celrep.2022.110677](https://doi.org/10.1016/j.celrep.2022.110677) (cit. on pp. 40, 41).
- Vehtari, Aki, Andrew Gelman, Daniel Simpson, Bob Carpenter, and Paul-Christian Bürkner (2021). “Rank-Normalization, Folding, and Localization: An Improved \hat{R} for Assessing Convergence of MCMC (with Discussion)”. In: *Bayesian Analysis* 16.2. DOI: [10.1214/20-BA1221](https://doi.org/10.1214/20-BA1221) (cit. on p. 26).
- Wagenmakers, Eric-Jan, Michael Lee, Tom Lodewyckx, and Geoffrey J. Iverson (2008). “Bayesian Versus Frequentist Inference”. In: *Bayesian Evaluation of Informative Hypotheses*. Ed. by Herbert Hoijtink, Irene Klugkist, and Paul A. Boelen, pp. 181–207. DOI: [10.1007/978-0-387-09612-4_9](https://doi.org/10.1007/978-0-387-09612-4_9) (cit. on p. 13).
- Ward, Daniel, Patrick Cannon, Mark Beaumont, Matteo Fasiolo, and Sebastian M. Schmon (2022). *Robust Neural Posterior Estimation and Statistical Model Criticism*. DOI: [10.48550/arXiv.2210.06564](https://doi.org/10.48550/arXiv.2210.06564) (cit. on pp. 22, 23, 46).
- Wiecki, Thomas, Imri Sofer, and Michael Frank (2013). “HDDM: Hierarchical Bayesian estimation of the Drift-Diffusion Model in Python”. In: *Frontiers in Neuroinformatics* 7. DOI: [10.3389/fninf.2013.00014](https://doi.org/10.3389/fninf.2013.00014) (cit. on pp. 27, 35).
- Wilkinson, Richard (2014). “Accelerating ABC methods using Gaussian processes”. In: *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics*, pp. 1015–1023. DOI: [10.48550/arxiv.1401.1436](https://doi.org/10.48550/arxiv.1401.1436) (cit. on p. 25).
- Winsberg, Eric (2022). “Computer Simulations in Science”. In: *The Stanford Encyclopedia of Philosophy*. Ed. by Edward N. Zalta and Uri Nodelman. Winter 2022 (cit. on p. 6).
- Wiqvist, Samuel, Jes Frellesen, and Umberto Picchini (2021). *Sequential Neural Posterior and Likelihood Approximation*. DOI: [10.48550/arXiv.2102.06522](https://doi.org/10.48550/arXiv.2102.06522) (cit. on pp. 26, 27).
- Witt, Christian Schroeder de, Bradley Gram-Hansen, Nantas Nardelli, Andrew Gambardella, Rob Zinkov, Puneet Dokania, N. Siddharth, Ana Belen Espinosa-Gonzalez, Ara Darzi, Philip Torr, and Atılım Güneş Baydin (2020). *Simulation-Based Inference for Global Health Decisions*. DOI: [10.48550/arXiv.2005.07062](https://doi.org/10.48550/arXiv.2005.07062) (cit. on p. 8).
- Wood, Simon N. (2010). “Statistical inference for noisy nonlinear ecological dynamic systems”. In: *Nature* 466.7310, pp. 1102–1104. DOI: [10.1038/nature09319](https://doi.org/10.1038/nature09319) (cit. on pp. 16, 19).
- Zaheer, Manzil, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander J Smola (2017). “Deep Sets”. In: *Advances in Neural Information Processing Systems*. Vol. 30. DOI: [10.48550/arxiv.1703.06114](https://doi.org/10.48550/arxiv.1703.06114) (cit. on p. 24).

Appendices

Flexible and efficient simulation-based inference for models of decision-making

Jan Boelts^{1,2*}, Jan-Matthis Lueckmann¹, Richard Gao¹, Jakob H Macke^{1,3}

¹Machine Learning in Science, Excellence Cluster Machine Learning, University of Tübingen, Tübingen, Germany; ²Technical University of Munich, Munich, Germany; ³Max Planck Institute for Intelligent Systems Tübingen, Tübingen, Germany

Abstract Inferring parameters of computational models that capture experimental data is a central task in cognitive neuroscience. Bayesian statistical inference methods usually require the ability to evaluate the likelihood of the model—however, for many models of interest in cognitive neuroscience, the associated likelihoods cannot be computed efficiently. Simulation-based inference (SBI) offers a solution to this problem by only requiring access to simulations produced by the model. Previously, Fengler et al. introduced likelihood approximation networks (LANs, Fengler et al., 2021) which make it possible to apply SBI to models of decision-making but require billions of simulations for training. Here, we provide a new SBI method that is substantially more simulation efficient. Our approach, mixed neural likelihood estimation (MNLE), trains neural density estimators on model simulations to emulate the simulator and is designed to capture both the continuous (e.g., reaction times) and discrete (choices) data of decision-making models. The likelihoods of the emulator can then be used to perform Bayesian parameter inference on experimental data using standard approximate inference methods like Markov Chain Monte Carlo sampling. We demonstrate MNLE on two variants of the drift-diffusion model and show that it is substantially more efficient than LANs: MNLE achieves similar likelihood accuracy with six orders of magnitude fewer training simulations and is significantly more accurate than LANs when both are trained with the same budget. Our approach enables researchers to perform SBI on custom-tailored models of decision-making, leading to fast iteration of model design for scientific discovery.

*For correspondence:
jan.boelts@uni-tuebingen.de

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

Funding: See page 15

Preprinted: 23 December 2021

Received: 25 January 2022

Accepted: 26 July 2022

Published: 27 July 2022

Reviewing Editor: Valentin Wyart, École normale supérieure, PSL University, INSERM, France

© Copyright Boelts et al. This article is distributed under the terms of the [Creative Commons Attribution License](#), which permits unrestricted use and redistribution provided that the original author and source are credited.

Editor's evaluation

This paper provides a new approach, Mixed Neural Likelihood Estimator (MNLE) to build likelihood emulators for simulation-based models where the likelihood is unavailable. The authors show that the MNLE approach is equally accurate but orders of magnitude more efficient than a recent proposal, likelihood approximation networks (LAN), on two variants of the drift-diffusion model (a widely used model in cognitive neuroscience). This work provides a practical approach for fitting more complex models of behavior and neural activity for which likelihoods are unavailable.

Introduction

Computational modeling is an essential part of the scientific process in cognitive neuroscience: Models are developed from prior knowledge and hypotheses, and compared to experimentally observed phenomena (*Churchland and Sejnowski, 1988; McClelland, 2009*). Computational models usually have free parameters which need to be tuned to find those models that capture experimental data. This is often approached by searching for single best-fitting parameters using grid search or optimization methods. While this point-wise approach has been used successfully (*Lee et al., 2016; Patil et al., 2016*) it can be scientifically more informative to perform Bayesian inference over the

model parameters: Bayesian inference takes into account prior knowledge, reveals *all* the parameters consistent with observed data, and thus can be used for quantifying uncertainty, hypothesis testing, and model selection (Lee, 2008; Shiffrin et al., 2008; Lee and Wagenmakers, 2014; Schad et al., 2021). Yet, as the complexity of models used in cognitive neuroscience increases, Bayesian inference becomes challenging for two reasons. First, for many commonly used models, computational evaluation of likelihoods is challenging because often no analytical form is available. Numerical approximations of the likelihood are typically computationally expensive, rendering standard approximate inference methods like Markov Chain Monte Carlo (MCMC) inapplicable. Second, models and experimental paradigms in cognitive neuroscience often induce scenarios in which inference is repeated for varying numbers of experimental trials and changing hierarchical dependencies between model parameters (Lee, 2011). As such, fitting computational models with arbitrary hierarchical structures to experimental data often still requires idiosyncratic and complex inference algorithms.

Approximate Bayesian computation (ABC, Sisson, 2018) offers a solution to the first challenge by enabling Bayesian inference based on comparing simulated with experimental data, without the need to evaluate an explicit likelihood function. Accordingly, various ABC methods have been applied to and developed for models in cognitive neuroscience and related fields (Turner and Van Zandt, 2012; Turner and Van Zandt, 2018; Palestro et al., 2009; Kangasrääsiö et al., 2019). However, ABC methods are limited regarding the second challenge because they become inefficient as the number of model parameters increases (Lueckmann et al., 2021) and require generating new simulations whenever the observed data or parameter dependencies change.

More recent approaches from the field simulation-based inference (SBI, Cranmer et al., 2020) have the potential to overcome these limitations by using machine learning algorithms such as neural networks. Recently, Fengler et al., 2021 presented an SBI algorithm for a specific problem in cognitive neuroscience—*inference for drift-diffusion models (DDMs)*. They introduced a new approach, called likelihood approximation networks (LANs), which uses neural networks to predict log-likelihoods from data and parameters. The predicted likelihoods can subsequently be used to generate posterior samples using MCMC methods. LANs are trained in a three-step procedure. First, a set of N parameters is generated and for each of the N parameters the model is simulated M times. Second, for each of the N parameters, empirical likelihood targets are estimated from the M model simulations using kernel density estimation (KDE) or empirical histograms. Third, a training dataset consisting of parameters, data points, and empirical likelihood targets is constructed by augmenting the initial set of N parameters by a factor of 1000: for each parameter, 1000 data points and empirical likelihood targets are generated from the learned KDE. Finally, supervised learning is used to train a neural network to predict log-likelihoods, by minimizing a loss function (the Huber loss) between the network-predicted log-likelihoods and the (log of) the empirically estimated likelihoods. Overall, LANs require a large number of model simulations such that the histogram probability of each possible observed data and for each possible combination of input parameters, can be accurately estimated— $N \cdot M$ model simulations, for example, 1.5 (150 billion) for the examples used in Fengler et al., 2021. The extremely high number of model simulations will make it infeasible for most users to run this training themselves, so that there would need to be a repository from which users can download pretrained LANs. This restricts the application of LANs to a small set of canonical models like DDMs, and prohibits customization and iteration of models by users. In addition, the high simulation requirement limits this approach to models whose parameters and observations are sufficiently low dimensional for histograms to be sampled densely.

To overcome these limitations, we propose an alternative approach called mixed neural likelihood estimation (MNLE). MNLE builds on recent advances in probabilistic machine learning, and in particular on the framework of neural likelihood estimation (Papamakarios et al., 2019b; Lueckmann et al., 2019) but is designed to specifically capture the mixed data types arising in models of decision-making, for example, discrete choices and continuous reaction times. Neural likelihood estimation has its origin in classical synthetic likelihood (SL) approaches (Wood, 2010; Price et al., 2018). Classical SL approaches assume the likelihood of the simulation-based model to be Gaussian (so that its moments can be estimated from model simulations) and then use MCMC methods for inference. This approach and various extensions of it have been widely used (Price et al., 2018; Ong et al., 2009; An et al., 2019; Priddle et al., 2022)—but inherently they need multiple model simulations for each parameter in the MCMC chain to estimate the associated likelihood.

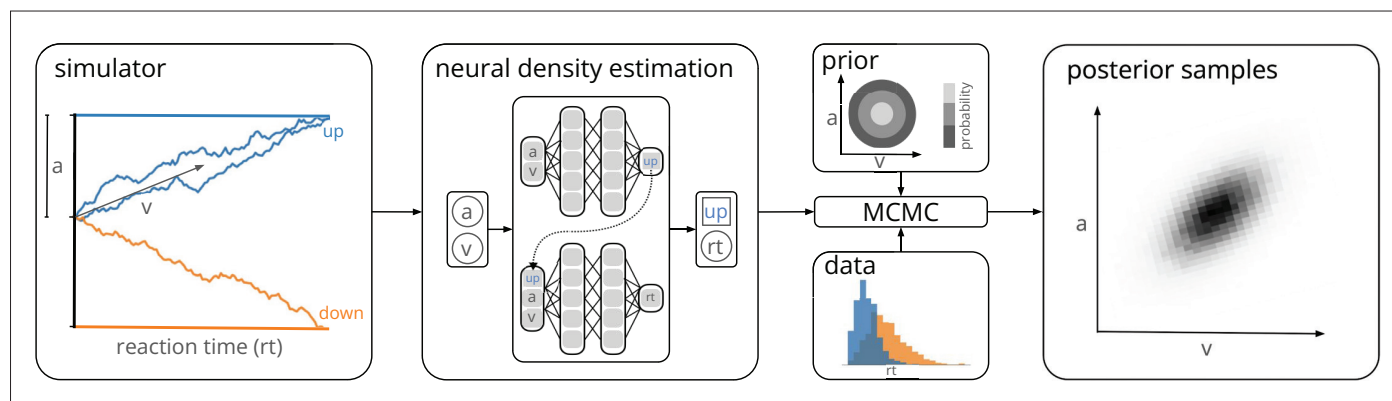


Figure 1. Training a neural density estimator on simulated data to perform parameter inference. Our goal is to perform Bayesian inference on models of decision-making for which likelihoods cannot be evaluated (here a drift-diffusion model for illustration, left). We train a neural density estimation network on synthetic data generated by the model, to provide access to (estimated) likelihoods. Our neural density estimators are designed to account for the mixed data types of decision-making models (e.g., discrete valued choices and continuous valued reaction times, middle). The estimated likelihoods can then be used for inference with standard Markov Chain Monte Carlo (MCMC) methods, that is, to obtain samples from the posterior over the parameters of the simulator given experimental data (right). Once trained, our method can be applied to flexible inference scenarios like varying number of trials or hierarchical inference without having to retrain the density estimator.

Neural likelihood approaches instead perform *conditional* density estimation, that is, they train a neural network to predict the parameters of the approximate likelihood conditioned on the model parameters (e.g., [Papamakarios et al., 2019b](#); [Lueckmann et al., 2019](#)). By using a conditional density estimator, it is possible to exploit continuity across different model parameters, rather than having to learn a separate density for each individual parameter as in classical SL. Recent advances in conditional density estimation (such as normalizing flows, [Papamakarios et al., 2019a](#)) further allow lifting the parametric assumptions of classical SL methods and learning flexible conditional density estimators which are able to model a wide range of densities, as well as highly nonlinear dependencies on the conditioning variable. In addition, neural likelihood estimators yield estimates of the probability density which are guaranteed to be non-negative and normalized, and which can be both sampled and evaluated, acting as a probabilistic emulator of the simulator ([Lueckmann et al., 2019](#)).

Our approach, MNLE, uses neural likelihood estimation to learn an emulator of the simulator. The training phase is a simple two-step procedure: first, a training dataset of N parameters θ is sampled from a proposal distribution and corresponding model simulations \mathbf{x} are generated. Second, the N parameter–data pairs (θ, \mathbf{x}) are directly used to train a conditional neural likelihood estimator to estimate $p(\mathbf{x}|\theta)$. Like for LANs, the proposal distribution for the training data can be any distribution over θ , and should be chosen to cover all parameter values one expects to encounter in empirical data. Thus, the prior distribution used for Bayesian inference constitutes a useful choice, but in principle any distribution that contains the support of the prior can be used. To account for mixed data types, we learn the likelihood estimator as a mixed model composed of one neural density estimator for categorical data and one for continuous data, conditioned on the categorical data. This separation allows us to choose the appropriate neural density estimator for each data type, for example, a Bernoulli model for the categorical data and a normalizing flow ([Papamakarios et al., 2019a](#)) for the continuous data. The resulting joint density estimator gives access to the likelihood, which enables inference via MCMC methods. See [Figure 1](#) for an illustration of our approach, and [Methods and materials](#) for details.

Both LANs and MNLEs allow for flexible inference scenarios common in cognitive neuroscience, for example, varying number of trials with same underlying experimental conditions or hierarchical inference, and need to be trained only once. However, there is a key difference between the two approaches. LANs use feed-forward neural networks to perform regression from model parameters to empirical likelihood targets obtained from KDE. MNLE instead learns the likelihood directly by performing conditional density estimation on the simulated data without requiring likelihood targets. This makes MNLE by design more simulation efficient than LANs—we demonstrate empirically that it can learn likelihood estimators which are as good or better than those reported in the LAN paper,

but using a factor of 1,000,000 fewer simulations (Fengler et al., 2021). When using the same simulation budget for both approaches, MNLE substantially outperforms LAN across several performance metrics. Moreover, MNLE results in a density estimator that is guaranteed to correspond to a valid probability distribution and can also act as an emulator that can generate synthetic data without running the simulator. The simulation efficiency of MNLEs allows users to explore and iterate on their own models without generating a massive training dataset, rather than restricting their investigations to canonical models for which pretrained networks have been provided by a central resource. To facilitate this process, we implemented our method as an extension to an open-source toolbox for SBI methods (Tejero-Cantero et al., 2020), and provide detailed documentation and tutorials.

Results

Evaluating the performance of MNLE on the DDM

We first demonstrate the efficiency and performance of MNLEs on a classical model of decision-making, the DDM (Ratcliff and McKoon, 2008). The DDM is an influential phenomenological model of a two-alternative perceptual decision-making task. It simulates the evolution of an internal decision variable that integrates sensory evidence until one of two decision boundaries is reached and a choice is made (Figure 1, left). The decision variable is modeled with a stochastic differential equation which, in the 'simple' DDM version (as used in Fengler et al., 2021), has four parameters: the drift rate v , boundary separation a , the starting point w of the decision variable, and the non-decision time τ . Given these four parameters $\theta = (v, a, w, \tau)$, a single simulation of the DDM returns data \mathbf{x} containing a choice $c \in \{0, 1\}$ and the corresponding reaction time in seconds $rt \in (\tau, \infty)$.

MNLE learns accurate likelihoods with a fraction of the simulation budget

The simple version of the DDM is the ideal candidate for comparing the performance of different inference methods because the likelihood of an observation given the parameters, $L(\mathbf{x}|\theta)$, can be calculated analytically (Navarro and Fuss, 2009, in contrast to more complicated versions of the DDM, e.g., Ratcliff and Rouder, 1998; Usher and McClelland, 2001; Reynolds and Rhodes, 2009). This enabled us to evaluate MNLE's performance with respect to the analytical likelihoods and the corresponding inferred posteriors of the DDM, and to compare against that of LANs on a range of simulation budgets. For MNLE, we used a budget of 10^5 simulations (henceforth referred to as MNLE⁵), for LANs we used budgets of 10^5 and 10^8 simulations (LAN⁵ and LAN⁸, respectively, trained by us) and the pretrained version based on 10^{11} simulations (LAN¹¹) provided by Fengler et al., 2021.

First, we evaluated the quality of likelihood approximations of MNLE⁵, and compared it to that of LAN^{5,8,11}. Both MNLEs and LANs were in principle able to accurately approximate the likelihoods for both decisions and a wide range of reaction times (see Figure 2a for an example, and Details of the numerical comparison). However, LANs require a much larger simulation budget than MNLE to achieve accurate likelihood approximations, that is, LANs trained with 10^5 or 10^8 simulations show visible deviations, both in the linear and in log-domain (Figure 2a, lines for LAN⁵ and LAN⁸).

To quantify the quality of likelihood approximation, we calculated the Huber loss and the mean-squared error (MSE) between the true and approximated likelihoods (Figure 2b, c), as well as between the log-likelihoods (Figure 2d, e). The metrics were calculated as averages over (log-) likelihoods of a fixed observation given 1000 parameters sampled from the prior, repeated for 100 observations simulated from the DDM. For metrics calculated on the untransformed likelihoods (Figure 2b, c), we found that MNLE⁵ was more accurate than LAN^{5,8,11} on all simulation budgets, showing smaller Huber loss than LAN^{5,8,11} in 99, 81, and 66 out of 100 comparisons, and smaller MSE than LAN^{5,8,11} on 98, 81, and 66 out of 100 comparisons, respectively. Similarly, for the MSE calculated on the log-likelihoods (Figure 2e), MNLE⁵ achieved smaller MSE than LAN^{5,8,11} on 100, 100, and 75 out of 100 comparisons, respectively. For the Huber loss calculated on the log-likelihoods (Figure 2d), we found that MNLE⁵ was more accurate than LAN⁵ and LAN⁸, but slightly less accurate than LAN¹¹, showing smaller Huber loss than LAN^{5,8} in all 100 comparisons, and larger Huber loss than LAN¹¹ in 62 out of 100 comparisons. All the above pairwise comparisons were significant under the binomial test ($p < 0.01$), but note that these are simulated data and therefore the p value can be arbitrarily inflated by increasing the number of comparisons. We also

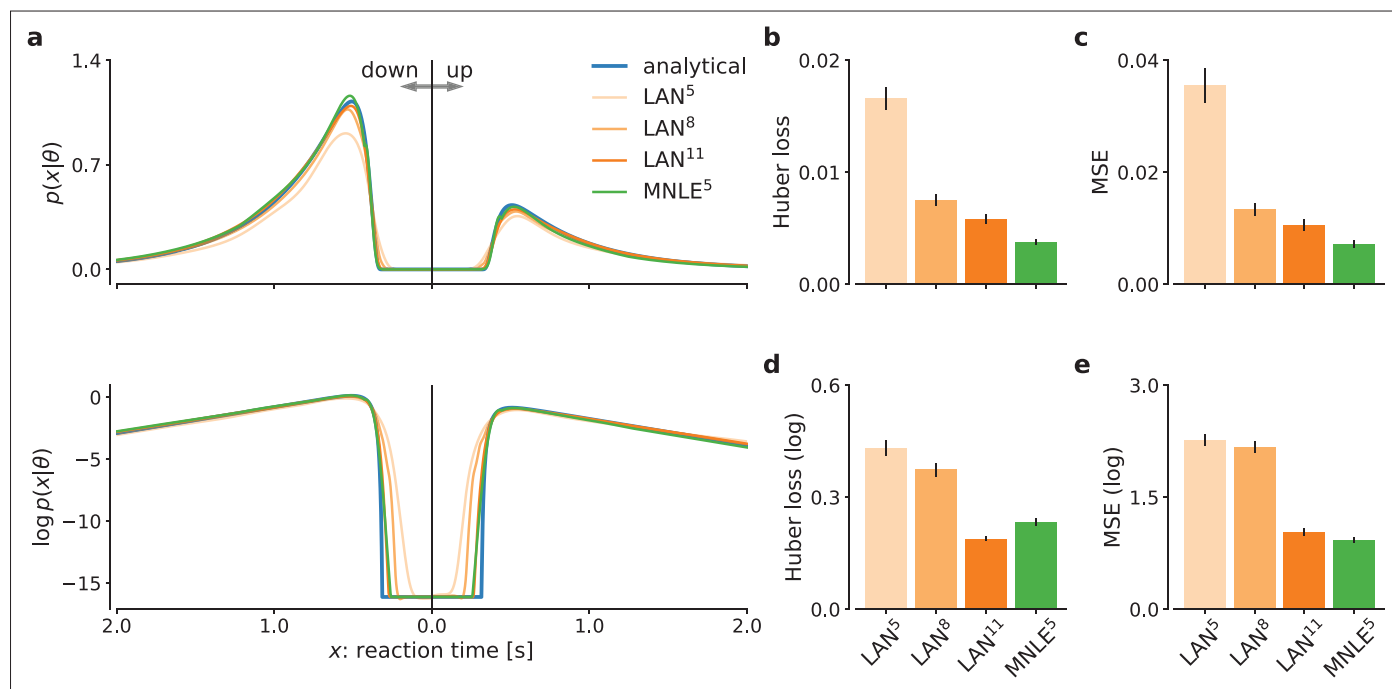


Figure 2. Mixed neural likelihood estimation (MNLE) estimates accurate likelihoods for the drift-diffusion model (DDM). The classical DDM simulates reaction times and choices of a two-alternative decision task and has an analytical likelihood which can be used for comparing the likelihood approximations of MNLE and likelihood approximation network (LAN). We compared MNLE trained with a budget of 10^5 simulations (green, MNLE⁵) to LAN trained with budgets of 10^5 , 10^8 , and 10^{11} simulations (shades of orange, LAN^{5,8,11}, respectively). (a) Example likelihood for a fixed parameter θ over a range of reaction times (reaction times for down- and up-choices shown toward the left and right, respectively). Shown on a linear scale (top panel) and a logarithmic scale (bottom panel). (b) Huber loss between analytical and estimated likelihoods calculated for a fixed simulated data point over 1000 test parameters sampled from the prior, averaged over 100 data points (lower is better). Bar plot error bars show standard error of the mean. (c) Same as in (b), but using mean-squared error (MSE) over likelihoods (lower is better). (d) Huber loss on the log-likelihoods (LAN's training loss). (e) MSE on the log-likelihoods.

The online version of this article includes the following figure supplement(s) for figure 2:

Figure supplement 1. Comparison of simulated drift-diffusion model (DDM) data and synthetic data sampled from the mixed neural likelihood estimation (MNLE) emulator.

note that the Huber loss on the log-likelihoods is the loss which is directly optimized by LANs, and thus this comparison should in theory favor LANs over alternative approaches. Furthermore, the MNLE⁵ results shown here represent averages over 10 random neural network initializations (five of which achieved smaller Huber loss than LAN¹¹), whereas the LAN¹¹ results are based on a single pretrained network. Finally, we also investigated MNLE's property to act as an emulator of the simulator and found the synthetic reaction times and choices generated from the MNLE emulator to match corresponding data simulated from the DDM accurately (see **Figure 2—figure supplement 1** and Appendix 1).

When using the learned likelihood estimators for inference with MCMC methods, their evaluation speed can also be important because MCMC often requires thousands of likelihood evaluations. We found that evaluating MNLE for a batch of 100 trials and 10 model parameters (as used during MCMC) took $4.14 \pm$ (mean over 100 repetitions \pm standard error of the mean), compared to $1.02 \pm$ for LANs, that is, MNLE incurred a larger computational foot-print at evaluation time. Note that these timings are based on an improved implementation of LANs compared to the one originally presented in **Fengler et al., 2021**, and evaluation times can depend on the implementation, compute infrastructure and parameter settings (see Details of the numerical comparison and Discussion). In summary, we found that MNLE trained with 10^5 simulations performed substantially better than LANs trained with 10^5 or 10^8 simulations, and similarly well or better than LANs trained with 10^{11} simulations, on all likelihood approximation accuracy metrics.

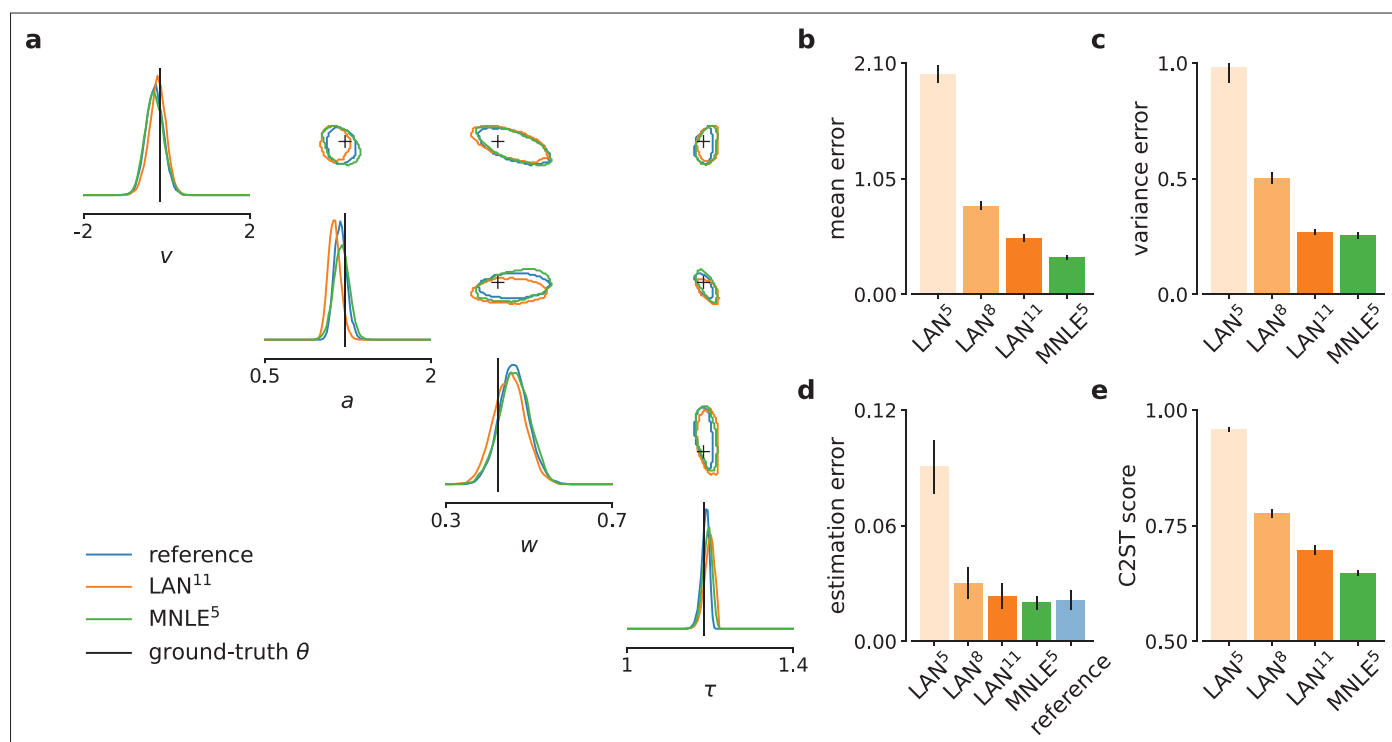


Figure 3. Mixed neural likelihood estimation (MNLE) infers accurate posteriors for the drift-diffusion model. Posteriors were obtained given 100-trial independent and identically distributed (i.i.d.) observations with Markov Chain Monte Carlo (MCMC) using analytical (i.e., reference) likelihoods, or those approximated using LAN^{5,8,11} trained with simulation budgets $10^{\{5,8,11\}}$, respectively, and MNLE⁵ trained with a budget of 10^5 simulations. (a) Posteriors given an example observation generated from the prior and the simulator, shown as 95% contour lines in a corner plot, that is, one-dimensional marginal (diagonal) and all pairwise two-dimensional marginals (upper triangle). (b) Difference in posterior sample mean of approximate (LAN^{5,8,11}, MNLE⁵) and reference posteriors (normalized by reference posterior standard deviation, lower is better). (c) Same as in (b) but for posterior sample variance (normalized by reference posterior variance, lower is better). (d) Parameter estimation error measured as mean-squared error (MSE) between posterior sample mean and the true underlying parameters (smallest possible error is given by reference posterior performance shown in blue). (e) Classification 2-sample test (C2ST) score between approximate (LAN^{5,8,11}, MNLE⁵) and reference posterior samples (0.5 is best). All bar plots show metrics calculated from 100 repetitions with different observations; error bars show standard error of the mean.

The online version of this article includes the following figure supplement(s) for figure 3:

Figure supplement 1. Drift-diffusion model (DDM) inference accuracy metrics for individual model parameters.

Figure supplement 2. Drift-diffusion model (DDM) example posteriors and parameter recovery for likelihood approximation networks (LANs) trained with smaller simulation budgets.

Figure supplement 3. Drift-diffusion model (DDM) inference accuracy metrics for different numbers of observed trials.

MNLE enables accurate flexible posterior inference with MCMC

In the previous section, we showed that MNLE requires substantially fewer training simulations than LANs to approximate the likelihood accurately. To investigate whether these likelihood estimates were accurate enough to support accurate parameter inference, we evaluated the quality of the resulting posteriors, using a framework for benchmarking SBI algorithms (Lueckmann et al., 2021). We used the analytical likelihoods of the simple DDM to obtain reference posteriors for 100 different observations, via MCMC sampling. Each observation consisted of 100 independent and identically distributed (i.i.d.) trials simulated with parameters sampled from the prior (see Figure 3a for an example, details in Materials and Methods). We then performed inference using MCMC based on the approximate likelihoods obtained with MNLE (10^5 budget, MNLE⁵) and the ones obtained with LAN for each of the three simulation budgets (LAN^{5,8,11}, respectively).

Overall, we found that the likelihood approximation performances presented above were reflected in the inference performances: MNLE⁵ performed substantially better than LAN⁵ and LAN⁸, and equally well or better than LAN¹¹ (Figure 3b–d). In particular, MNLE⁵ approximated the posterior

mean more accurately than LAN^(5,8,11) (Figure 3b), being more accurate than LAN^(5,8,11) in 100, 90, and 67 out of 100 comparisons, respectively. In terms of posterior variance, MNLE⁵ performed better than LAN^(5,8) and on par with LAN¹¹ (Figure 3c), being more accurate than LAN^(5,8,11) in 100, 93 ($p < 0.01$, binomial test), and 58 ($p = 0.13$) out of 100 pairwise comparisons, respectively.

Additionally, we measured the parameter estimation accuracy as the MSE between the posterior mean and the ground-truth parameters underlying the observed data. We found that MNLE⁵ estimation error was indistinguishable from that of the reference posterior, and that LAN performance was similar only for the substantially larger simulation budget of LAN¹¹ (Figure 3c), with MNLE being closer to reference performance than LAN^(5,8,11) in 100, 91, and 66 out of 100 comparisons, respectively (all $p < 0.01$). Note that all three metrics were reported as averages over the four parameter dimensions of the DDM to keep the visualizations compact, and that this average did not affect the results qualitatively. We report metrics for each dimension in Figure 3—figure supplement 1, as well as additional inference accuracy results for smaller LAN simulation budgets (Figure 3—figure supplement 2) and for different numbers of observed trials (Figure 3—figure supplement 3).

Finally, we used the classifier 2-sample test (C2ST, Lopez-Paz and Oquab, 2017; Lueckmann et al., 2021) to quantify the similarity between the estimated and reference posterior distributions. The C2ST is defined to be the error rate of a classification algorithm which aims to classify whether samples belong to the true or the estimated posterior. Thus, it ranges from 0.5 (no difference between the distributions, the classifier is at chance level) to 1.0 (the classifier can perfectly distinguish the two

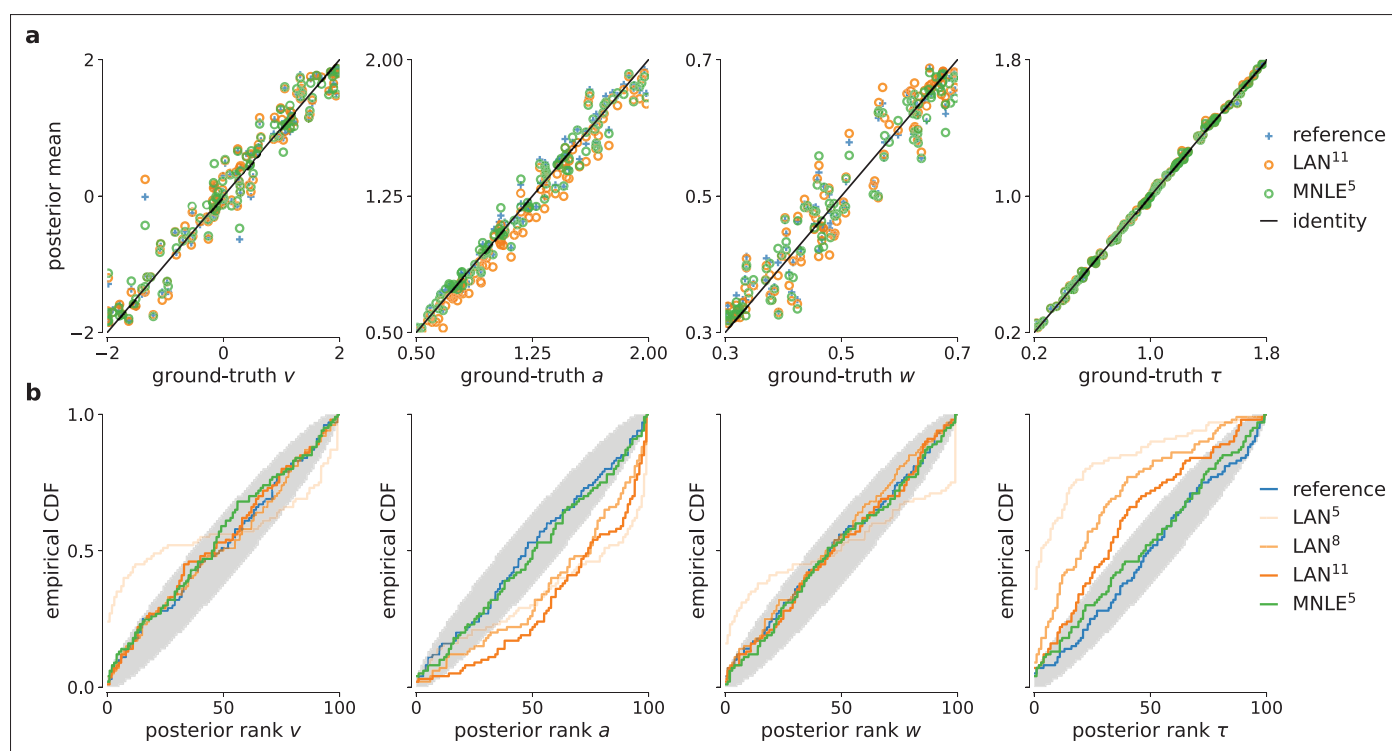


Figure 4. Parameter recovery and posterior uncertainty calibration for the drift-diffusion model (DDM). (a) Underlying ground-truth DDM parameters plotted against the sample mean of posterior samples inferred with the analytical likelihoods (reference, blue crosses), likelihood approximation network (LAN; orange circles), and mixed neural likelihood estimation (MNLE; green circles), for 100 different observations. Markers close to diagonal indicate good recovery of ground-truth parameters; circles on top of blue reference crosses indicate accurate posterior means. (b) Simulation-based calibration results showing empirical cumulative density functions (CDF) of the ground-truth parameters ranked under the inferred posteriors calculated from 100 different observations. A well-calibrated posterior must have uniformly distributed ranks, as indicated by the area shaded gray. Shown for reference posteriors (blue), LAN posteriors obtained with increasing simulation budgets (shades of orange, LAN^(5,8,11)), and MNLE posterior (green, MNLE⁵), and for each parameter separately (v , a , w , and τ).

The online version of this article includes the following figure supplement(s) for figure 4:

Figure supplement 1. Drift-diffusion model (DDM) parameter recovery for different numbers of observed trials.

Figure supplement 2. Drift-diffusion model (DDM) simulation-based calibration (SBC) results for different numbers of observed trials.

distributions). We note that the C2ST is a highly sensitive measure of discrepancy between two multivariate distributions—for example if the two distributions differ in any dimension, the C2ST will be close to 1 even if all other dimensions match perfectly. We found that neither of the two approaches was able to achieve perfect approximations, but that MNLE⁵ achieved lower C2ST scores compared to LAN^(5,8,11) on all simulation budgets (**Figure 3e**): mean C2ST score LAN^(5,8,11), 0.96, 0.78, 0.70 vs. MNLE⁵, 0.65, with MNLE⁵ showing a better score than LAN^(5,8,11) on 100, 91, and 68 out of 100 pairwise comparisons, respectively (all $p < 0.01$). In summary, MNLE achieves more accurate recovery of posterior means than LANs, similar or better recovery of posterior variances, and overall more accurate posteriors (as quantified by C2ST).

MNLE posteriors have uncertainties which are well calibrated

For practical applications of inference, it is often desirable to know how well an inference procedure can recover the ground-truth parameters, and whether the uncertainty estimates are well calibrated, (**Cook et al., 2006**), that is, that the *uncertainty* estimates of the posterior are balanced, and neither over-confident nor under-confident. For the DDM, we found that the posteriors inferred with MNLE and LANs (when using LAN¹¹) recovered the ground-truth parameters accurately (in terms of posterior means, **Figure 3d** and **Figure 4a**)—in fact, parameter recovery was similarly accurate to using the ‘true’ analytical likelihoods, indicating that much of the residual error is due to stochasticity of the observations, and not the inaccuracy of the likelihood approximations.

To assess posterior calibration, we used simulation-based calibration (SBC, **Talts et al., 2018**). The basic idea of SBC is the following: If one repeats the inference with simulations from many different prior samples, then, with a well-calibrated inference method, the combined samples from all the inferred posteriors should be distributed according to the prior. One way to test this is to calculate the rank of each ground-truth parameter (samples from the prior) under its corresponding posterior, and to check whether all the ranks follow a uniform distribution. SBC results indicated that MNLE posteriors were as well calibrated as the reference posteriors, that is, the empirical cumulative density functions of the ranks were close to that of a uniform distribution (**Figure 4b**)—thus, on this example, MNLE inferences would likely be of similar quality compared to using the analytical likelihoods. When trained with the large simulation budget of 10^{11} simulations, LANs, too appeared to recover most of the ground-truth parameters well. However, SBC detected a systematic underestimation of the parameter α and overestimation of the parameter τ , and this bias increased for the smaller simulation budgets of LAN⁵ and LAN⁸ (**Figure 4b**, see the deviation below and above the desired uniform distribution of ranks, respectively).

The results so far (i.e., **Figures 3 and 4**) indicate that both LAN¹¹ and MNLE⁵ lead to similar parameter recovery, but only MNLE⁵ leads to posteriors which were well calibrated for all parameters. These results were obtained using a scenario with 100 i.i.d. trials. When increasing the number of trials (e.g., to 1000 trials), posteriors become very concentrated around the ground-truth value. In that case, while the posteriors overall identified the ground-truth parameter value very well (**Figure 4—figure supplement 1c**), even small deviations in the posteriors can have large effects on the posterior metrics (**Figure 3—figure supplement 3**). This effect was also detected by SBC, showing systematic biases for some parameters (**Figure 4—figure supplement 2**). For MNLE, we found that these biases were smaller, and furthermore that it was possible to mitigate this effect by inferring the posterior using ensembles, for example, by combining samples inferred with five MNLEs trained with identical settings but different random initialization (see Appendix 1 for details). These results show the utility of using SBC as a tool to test posterior coverage, especially when studying models for which reference posteriors are not available, as we demonstrate in the next section.

MNLE infers well-calibrated, predictive posteriors for a DDM with collapsing bounds

MNLE was designed to be applicable to models for which evaluation of the likelihood is not practical so that standard inference tools cannot be used. To demonstrate this, we applied MNLE to a variant of the DDM for which analytical likelihoods are not available (note, however, that numerical approximation of likelihoods for this model would be possible, see e.g., **Shinn et al., 2020**, Materials and methods for details). This DDM variant simulates a decision variable like the simple DDM used above, but with linearly collapsing instead of constant decision boundaries (see e.g., **Hawkins et al.,**

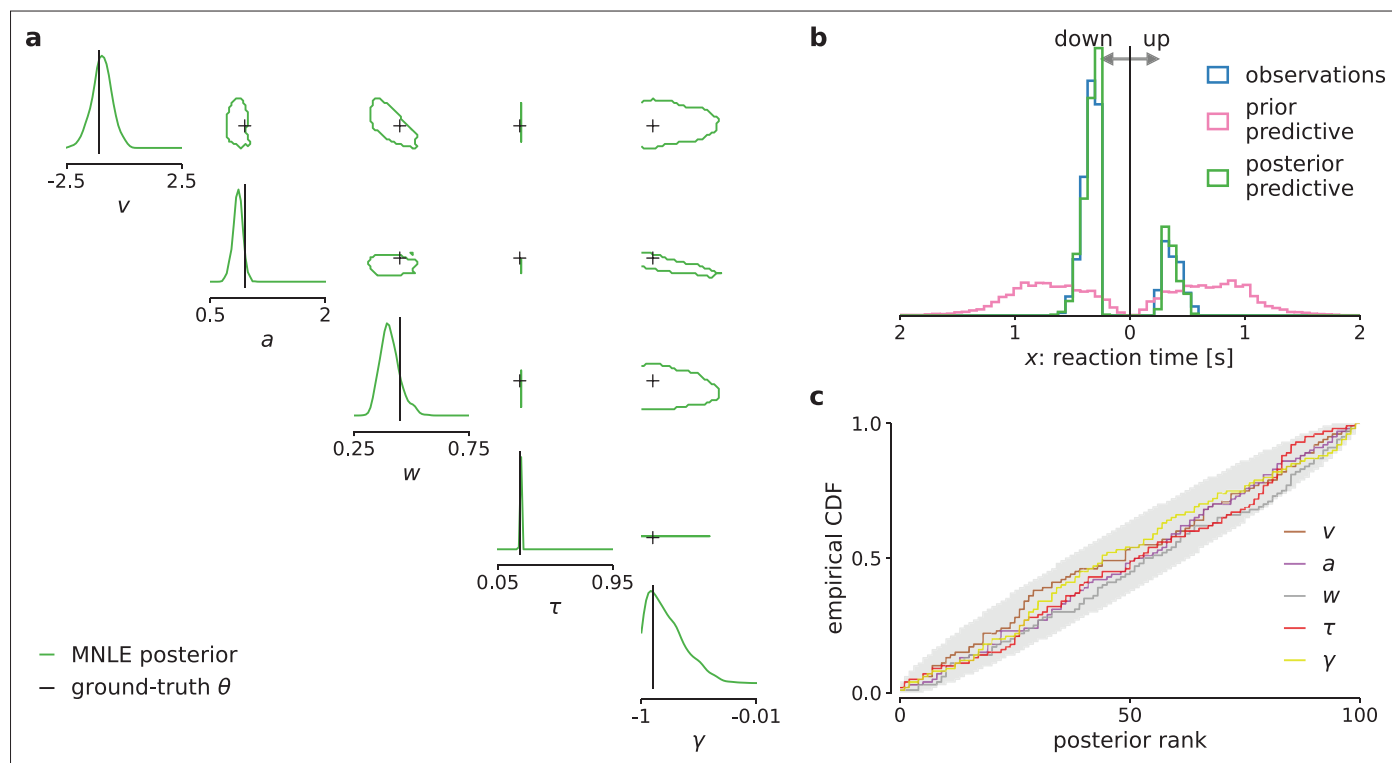


Figure 5. Mixed neural likelihood estimation (MNLE) infers accurate posteriors for the drift-diffusion model (DDM) with collapsing bounds. Posterior samples were obtained given 100-trial observations simulated from the DDM with linearly collapsing bounds, using MNLE and Markov Chain Monte Carlo (MCMC). **(a)** Approximate posteriors shown as 95% contour lines in a corner plot of one- (diagonal) and two-dimensional (upper triangle) marginals, for a representative 100-trial observation simulated from the DDM. **(b)** Reaction times and choices simulated from the ground-truth parameters (blue) compared to those simulated given parameters sampled from the prior (prior predictive distribution, purple) and from the MNLE posterior shown in **(a)** (posterior predictive distribution, green). **(c)** Simulation-based calibration results showing empirical cumulative density functions (CDF) of the ground-truth parameters ranked under the inferred posteriors, calculated from 100 different observations. A well-calibrated posterior must have uniformly distributed ranks, as indicated by the area shaded gray. Shown for each parameter separately (v , a , w , τ , and γ).

2015; Palestro et al., 2018). The collapsing bounds are incorporated with an additional parameter γ indicating the slope of the decision boundary, such that $\theta = (a, v, w, \tau, \gamma)$ (see Details of the numerical comparison).

We tested inference with MNLE on the DDM with linearly collapsing bound using observations comprised of 100 i.i.d. trials simulated with parameters sampled from the prior. Using the same MNLE training and MCMC settings as above, we found that posterior density concentrated around the underlying ground-truth parameters (see **Figure 5a**), suggesting that MNLE learned the underlying likelihood accurately. To assess inference quality systematically without needing reference posteriors, we performed posterior predictive checks by running simulations with the inferred posteriors samples and comparing them to observed (simulated) data, and checked posterior calibration properties using SBC. We found that the inferred posteriors have good predictive performance, that is, data simulated from the inferred posterior samples accurately matched the observed data (**Figure 5b**), and that their uncertainties are well calibrated as quantified by the SBC results (**Figure 5c**). Overall, this indicated that MNLE accurately inferred the posterior of this intractable variant of the DDM.

Discussion

Statistical inference for computational models in cognitive neuroscience can be challenging because models often do not have tractable likelihood functions. The recently proposed LAN method (**Fengler et al., 2021**) performs SBI for a subset of such models (DDMs) by training neural networks with model simulations to approximate the intractable likelihood. However, LANs require large amounts of

training data, restricting its usage to canonical models. We proposed an alternative approach called MNLE, a synthetic neural likelihood method which is tailored to the data types encountered in many models of decision-making.

Our comparison on a tractable example problem used in *Fengler et al., 2021* showed that MNLE performed on par with LANs using six orders of magnitude fewer model simulations for training. While *Fengler et al., 2021* discuss that LANs were not optimized for simulation efficiency and that it might be possible to reduce the required model simulations, we emphasize that the difference in simulation efficiency is due to an inherent property of LANs. For each parameter in the training data, LANs require empirical likelihood targets that have to be estimated by building histograms or kernel density estimates from thousands of simulations. MNLE, instead, performs conditional density estimation without the need of likelihood targets and can work with only one simulation per parameter. Because of these conceptual differences, we expect the substantial performance advantage of MNLE to be robust to the specifics of the implementation.

After the networks are trained, the time needed for each evaluation determines the speed of inference. In that respect, both LANs and MNLEs are conceptually similar in that they require a single forward-pass through a neural network for each evaluation, and we found MNLE and the original implementation of LANs to require comparable computation times. However, evaluation time will depend, for example, on the exact network architecture, software framework, and computing infrastructure used. Code for a new PyTorch implementation of LANs has recently been released and improved upon the evaluation speed original implementation we compared to. While this new implementation made LAN significantly faster for a single forward-pass, we observed that the resulting time difference with the MCMC settings used here was only on the order of minutes, whereas the difference in simulation time for LAN¹¹ vs. MNLE⁵ was on the order of days. The exact timings will always be implementation specific and whether or not these differences are important will depend on the application at hand. In a situation where iteration over model design is required (i.e., custom DDMs), an increase in training efficiency on the order of days would be advantageous.

There exist a number of approaches with corresponding software packages for estimating parameters of cognitive neuroscience models, and DDMs in particular. However, these approaches either only estimate single best-fitting parameters (*Voss and Voss, 2007; Wagenmakers et al., 2007; Chandrasekaran and Hawkins, 2019; Heathcote et al., 2019; Shinn et al., 2020*) or, if they perform fully Bayesian inference, can only produce Gaussian approximations to posteriors (*Feltgen and Daunizeau, 2021*), or are restricted to variants of the DDM for which the likelihood can be evaluated (*Wiecki et al., 2013*, HDDM [Hierarchical DDM] toolbox). A recent extension of the HDDM toolbox includes LANs, thereby combining HDDM's flexibility with LAN's ability to perform inference without access to the likelihood function (but this remains restricted to variants of the DDM for which LAN can be pretrained). In contrast, MNLE can be applied to any simulation-based model with intractable likelihoods and mixed data type outputs. Here, we focused on the direct comparison to LANs based on variants of the DDM. We note that these models have a rather low-dimensional observation structure (as common in many cognitive neuroscience models), and that our examples did not include additional parameter structure, for example, stimulus encoding parameters, which would increase the dimensionality of the learning problem. However, other variants of neural density estimation have been applied successfully to a variety of problems with higher dimensionality (see e.g., *Gonçalves et al., 2020; Lueckmann et al., 2021; Glöckler et al., 2021; Dax et al., 2022*). Therefore, we expect MNLE to be applicable to other simulation-based problems with higher-dimensional observation structure and parameter spaces, and to scale more favorably than LANs. Like for any neural network-based approach, applying MNLE to different inference problems may require selecting different architecture and training hyperparameters settings, which may induce additional computational training costs. To help with this process, we adopted default settings which have been shown to work well on a large range of SBI benchmarking problems (*Lueckmann et al., 2021*), and we integrated MNLE into the established *sbi* python package that provides well-documented implementations for training- and inference performance of SBI algorithms.

Several extensions to classical SL approaches have addressed the problem of a bias in the likelihood approximation due to the strong parametric assumptions, that is, Gaussianity, the use of summary statistics, or finite-sample biases (*Price et al., 2018; Ong et al., 2009; van Opheusden et al., 2020*). MNLE builds on flexible neural likelihood estimators, for example, normalizing flows, and does not

require summary statistics for a low-dimensional simulator like the DDM, so would be less susceptible to these first two biases. It could be subject to biases resulting from the estimation of the log-likelihoods from a finite number of simulations. In our numerical experiments, and for the simulation budgets we used, we did not observe biased inference results. We speculate that the ability of neural density estimators to pool data across multiple parameter settings (rather than using only data from a specific parameter set, like in classical SL methods) mitigates finite-sample effects.

MNLE is an SBI method which uses neural density estimators to estimate likelihoods. Alternatives to neural likelihood estimation include neural posterior estimation (NPE, [Papamakarios and Murray, 2016](#); [Lueckmann et al., 2017](#); [Greenberg et al., 2019](#), which uses conditional density estimation to learn the posterior directly) and neural ratio estimation (NRE, [Hermans et al., 2020](#); [Durkan et al., 2020](#), which uses classification to approximate the likelihood-to-evidence ratio to then perform MCMC). These approaches could in principle be applied here as well, however, they are not as well suited for the flexible inference scenarios common in decision-making models as MNLE. NPE directly targets the posterior and therefore, by design, typically requires retraining if the structure of the problem changes (e.g., if the prior or the hierarchical structure of the model changes). There are variants of NPE that use embedding nets which can amortize over changing number of trials, avoiding retraining ([Radev et al., 2022](#), [von Krause et al., 2022](#)). NRE learns the likelihood-to-evidence ratio using ratio estimation (and not density estimation) and would not provide an emulator of the simulator.

Regarding future research directions, MNLE has the potential to become more simulation efficient by using weight sharing between the discrete and the continuous neural density estimators (rather than to use separate neural networks, as we did here). Moreover, for high-dimensional inference problems in which slice sampling-based MCMC might struggle, MNLE could be used in conjunction with gradient-based MCMC methods like Hamiltonian Monte Carlo (HMC, [Brooks et al., 2011](#); [Hoffman and Gelman, 2014](#)), or variational inference as recently proposed by [Wiqvist et al., 2021](#) and [Glöckler et al., 2021](#). With MNLE's full integration into the `sbi` package, both gradient-based MCMC methods from Pyro ([Bingham et al., 2019](#)), and variational inference for SBI (SNVI, [Glöckler et al., 2021](#)) are available as inference methods for MNLE (a comparison of HMC and SNVI to slice sampling MCMC on one example observation resulted in indistinguishable posterior samples). Finally, using its emulator property (see Appendix 1), MNLE could be applied in an active learning setting for highly expensive simulators in which new simulations are chosen adaptively according to a acquisition function in a Bayesian optimization framework ([Gutmann and Corander, 2016](#); [Lueckmann et al., 2019](#); [Järvenpää et al., 2019](#)).

In summary, MNLE enables flexible and efficient inference of parameters of models in cognitive neuroscience with intractable likelihoods. The training efficiency and flexibility of the neural density estimators used overcome the limitations of LANs ([Fengler et al., 2021](#)). Critically, these features enable researchers to develop customized models of decision-making and not just apply existing models to new data. We implemented our approach as an extension to a public `sbi` python package with detailed documentation and examples to make it accessible for practitioners.

Materials and methods

Mixed neural likelihood estimation

MNLE extends the framework of neural likelihood estimation ([Papamakarios et al., 2019a](#); [Lueckmann et al., 2019](#)) to be applicable to simulation-based models with mixed data types. It learns a parametric model $q_{\psi}(\mathbf{x}|\theta)$ of the intractable likelihood $p(\mathbf{x}|\theta)$ defined implicitly by the simulation-based model. The parameters ψ are learned with training data $\{\theta_n, \mathbf{x}_n\}_{1:N}$ comprised of model parameters θ_n and their corresponding data simulated from the model $\mathbf{x}_n|\theta_n \sim p(\mathbf{x}|\theta_n)$. The parameters are sampled from a proposal distribution over parameters $\theta_n \sim p(\theta)$. The proposal distribution could be any distribution, but it determines the parameter regions for which the density estimator will be good in estimating likelihoods. Thus, the prior, or a distribution that contains the support of the prior (or even all priors which one expects to use in the future) constitutes a useful choice. After training, the emulator can be used to generate synthetic data $\mathbf{x}|\theta \sim q_{\psi}(\mathbf{x}|\theta)$ given parameters, and to evaluate the SL $q_{\psi}(\mathbf{x}|\theta)$ given experimentally observed data. Finally, the SL can be used to obtain posterior samples via

$$p(\theta|\mathbf{x}) \propto q_{\psi}(\mathbf{x}|\theta)p(\theta), \quad (1)$$

through approximate inference with MCMC. Importantly, the training is amortized, that is, the emulator can be applied to new experimental data without retraining (running MCMC is still required).

We tailored MNLE to simulation-based models that return mixed data, for example, in form of reaction times rt and (usually categorical) choices c as for the DDM. Conditional density estimation with normalizing flows has been proposed for continuous random variables (Papamakarios et al., 2019a), or discrete random variables (Tran et al., 2019), but not for mixed data. Our solution for estimating the likelihood of mixed data is to simply factorize the likelihood into continuous and discrete variables,

$$p(rt, c|\theta) = p(rt|\theta, c) p(c|\theta), \quad (2)$$

and use two separate neural likelihood estimators: A discrete one q_{ψ_c} to estimate $p(c|\theta)$ and a continuous one $q_{\psi_{rt}}$ to estimate $p(rt|\theta, c)$. We defined q_{ψ_c} to be a Bernoulli model and use a neural network to learn the Bernoulli probability ρ given parameters θ . For $q_{\psi_{rt}}$ we used a conditional neural spline flow (NSF, Durkan et al., 2019) to learn the density of rt given a parameter θ and choice c . The two estimators are trained separately using the same training data (see Neural network architecture, training and hyperparameters for details). After training, the full neural likelihood can be constructed by multiplying the likelihood estimates q_{ψ_c} and $q_{\psi_{rt}}$ back together:

$$q_{\psi_c, \psi_{rt}}(rt, c|\theta) = q_{\psi_c}(c|\theta) q_{\psi_{rt}}(rt|c, \theta). \quad (3)$$

Note that, as the second estimator $q_{\psi_{rt}}(rt|c, \theta)$ is conditioned on the choice c , our likelihood model can account for statistical dependencies between choices and reaction times. The neural likelihood can then be used to approximate the intractable likelihood defined by the simulator, for example, for inference with MCMC. Additionally, it can be used to sample synthetic data given model parameters, without running the simulator (see The emulator property of MNLE).

Relation to LAN

Neural likelihood estimation can be much more simulation efficient than previous approaches because it exploits continuity across the parameters by making the density estimation conditional. Fengler et al., 2021, too, aim to exploit continuity across the parameter space by training a neural network to predict the value of the likelihood function from parameters θ and data \mathbf{x} . However, the difference to neural likelihood estimation is that they do not use the neural network for density estimation directly, but instead do classical neural network-based regression on likelihood targets. Crucially, the likelihood targets first have to be obtained for each parameter in the training dataset. To do so, one has to perform density estimation using KDE (as proposed by Turner et al., 2015) or empirical histograms for every parameter separately. Once trained, LANs do indeed exploit the continuity across the parameter space (they can predict log-likelihoods given unseen data and parameters), however, they do so at a very high simulation cost: For a training dataset of N parameters, they perform N times KDE based on M simulations each¹¹ For models with categorical output, that is, all decision-making models, KDE is performed separately for each choice., resulting is an overall simulation budget of $N \cdot M$ ($N = 1.5$ million and $M = 100,000$ for 'pointwise' LAN approach).

Details of the numerical comparison

The comparison between MNLE and LAN is based on the DDM. The DDM simulates a decision variable X as a stochastic differential equation with parameters $\theta = (v, a, w, \tau)$:

$$dX_{t+\tau} = vdt + dW, \quad X_{\tau} = w, \quad (4)$$

where W a Wiener noise process. The priors over the parameters are defined to be uniform: $v \sim \mathcal{U}(-2, 2)$ is the drift, $a \sim \mathcal{U}(0.5, 2)$ the boundary separation, $w \sim \mathcal{U}(0.3, 0.7)$ the initial offset, and $\tau \sim \mathcal{U}(0.2, 1.8)$ the nondesideration time. A single simulation from the model returns a choice $c \in \{0, 1\}$ and the corresponding reaction time in seconds $rt \in (\tau, \infty)$.

For this version of the DDM the likelihood of an observation (c, rt) given parameters θ , $L(c, rt|\theta)$, can be calculated analytically (Navarro and Fuss, 2009). To simulate the DDM and calculate analytical likelihoods we used the approach and the implementation proposed by Drugowitsch, 2016. We

numerically confirmed that the simulations and the analytical likelihoods match those obtained from the research code provided by [Fengler et al., 2021](#).

To run LANs, we downloaded the neural network weights of the pretrained models from the repository mentioned in [Fengler et al., 2021](#). The budget of training simulations used for the LANs was 1.5×10^{11} (1.5 million training data points, each obtained from KDE based on 10^5 simulations). We only considered the approach based on training a multilayer perceptron on single-trial likelihoods ('pointwise approach', [Fengler et al., 2021](#)). At a later stage of our study we performed additional experiments to evaluate the performance of LANs trained at smaller simulation budgets, for example, for 10^5 and 10^8 simulations. For this analysis, we used an updated implementation of LANs based on PyTorch that was provided by the authors. We used the training routines and default settings provided with that implementation. To train LANs at the smaller budgets we used the following splits of budgets into number of parameter settings drawn from the prior, and number of simulations per parameter setting used for fitting the KDE: for the 10^5 budget we used 10^2 and 10^3 , respectively (we ran experiments splitting the other way around, but results were slightly better for this split); for the 10^8 budget we used an equal split of 10^4 each (all code publicly available, see Code availability).

To run MNLE, we extended the implementation of neural likelihood estimation in the sbi toolbox ([Tejero-Cantero et al., 2020](#)). All comparisons were performed on a single AMD Ryzen Threadripper 1920X 12-Core processor with 2.2 GHz and the code is publicly available (see Code availability).

For the DDM variant with linearly collapsing decision boundaries, the boundaries were parameterized by the initial boundary separation, a , and one additional parameter, γ , indicating the slope with which the boundary approaches zero. This resulted in a five-dimensional parameter space for which we used the same prior as above, plus the an additional uniform prior for the slope $\gamma \sim \mathcal{U}(-1.0, 0)$. To simulate this DDM variant, we again used the Julia package by [Drugowitsch, 2016](#), but we note that for this variant no analytical likelihoods are available. While it would be possible to approximate the likelihoods numerically using the Fokker–Planck equations (see e.g., [Shinn et al., 2020](#)), this would usually involve a trade-off between computation time and accuracy as well as design of bespoke solutions for individual models, and was not pursued here.

Flexible Bayesian inference with MCMC

Once the MNLE is trained, it can be used for MCMC to obtain posterior samples $\theta \sim p(\theta|\mathbf{x})$ given experimentally observed data \mathbf{x} . To sample from posteriors via MCMC, we transformed the parameters to unconstrained space, used slice sampling ([Neal, 2009](#)), and initialized ten parallel chains using sequential importance sampling ([Papamakarios et al., 2019a](#)), all as implemented in the sbi toolbox. We ran MCMC with identical settings for MNLE and LAN.

Importantly, performing MNLE and then using MCMC to obtain posterior samples allows for flexible inference scenarios because the form of \mathbf{x} is not fixed. For example, when the model produces trial-based data that satisfy the i.i.d. assumption, for example, a set of reaction times and choices $\mathbf{X} = \{rt, c\}_{i=1}^N$ in a DDM, then MNLE allows to perform inference given varying numbers of trials, without retraining. This is achieved by training MNLE based on single-trial likelihoods once and then combining multiple trials into the joint likelihood only when running MCMC:

$$p(\theta|\mathbf{X}) \propto \prod_{i=1}^N q(rt_i, c_i|\theta) p(\theta). \quad (5)$$

Similarly, one can use the neural likelihood to perform hierarchical inference with MCMC, all without the need for retraining (see [Hermans et al., 2020](#); [Fengler et al., 2021](#), for examples).

Stimulus- and intertrial dependencies

Simulation-based models in cognitive neuroscience often depend not only on a set of parameters θ , but additionally on (a set of) stimulus variables s which are typically given as part of the experimental conditions. MNLE can be readily adapted to this scenario by generating simulated data with multiple stimulus variables, and including them as additional inputs to the network during inference. Similarly, MNLE could be adapted to scenarios in which the i.i.d. assumption across trials as used above (see Flexible Bayesian inference with MCMC) does not hold. Again, this would be achieved by adapting the model simulator accordingly. For example, when inferring parameters θ of a DDM for

which the outcome of the current trial i additionally depends on current stimulus variables s_i , as well as on previous stimuli s_j and responses r_j , then one would implement the DDM simulator as a function $f(\theta; s_{i-T}, \dots, s_i; r_{i-T}, \dots, r_{i-1})$ (where T is a history parameter) and then learn the underlying likelihood by emulating f with MNLE.

Neural network architecture, training, and hyperparameters

Architecture

For the architecture of the Bernoulli model we chose a feed-forward neural network that takes parameters θ as input and predicts the Bernoulli probability ρ of the corresponding choices. For the normalizing flow we used the NSF architecture (Durkan et al., 2019). NSFs use a standard normal base distribution and transform it using several modules of monotonic rational-quadratic splines whose parameters are learned by invertible neural networks. Using an unbounded base distribution for modeling data with bounded support, for example, reaction time data $rt \in (0, \infty)$, can be challenging. To account for this, we tested two approaches: We either transformed the reaction time data to logarithmic space to obtain an unbounded support ($\log rt \in (-\infty, \infty)$), or we used a log-normal base distribution with rectified (instead of linear) tails for the splines (see Durkan et al., 2019, for details and Architecture and training hyperparameters for the architecture settings used).

Training

The neural network parameters ψ_c and ψ_{rt} were trained using the maximum likelihood loss and the Adam optimizer (Kingma and Ba, 2015). As proposal distribution for the training dataset we used the prior over DDM parameters. Given a training dataset of parameters, choices, and reaction times $\{\theta_i, (c_i, rt_i)\}_{i=1}^N$ with $\theta_i \sim p(\theta)$; $(c_i, rt_i) \sim \text{DDM}(\theta_i)$, we minimized the negative log-probability of the model. In particular, using the same training data, we trained the Bernoulli choice model by minimizing

$$-\frac{1}{N} \sum_{i=1}^N \log q_{\psi_c}(c_i | \theta_i), \quad (6)$$

and the NSF by minimizing

$$-\frac{1}{N} \sum_{i=1}^N \log q_{\psi_{rt}}(rt | c_i, \theta_i). \quad (7)$$

Training was performed with code and training hyperparameter settings provided in the sbi toolbox (Tejero-Cantero et al., 2020).

Hyperparameters

MNLE requires a number of hyperparameter choices regarding the neural network architectures, for example, number of hidden layers, number of hidden units, number of stacked NSF transforms, kind of base distribution, among others (Durkan et al., 2019). With our implementation building on the sbi package we based our hyperparameter choices on the default settings provided there. This resulted in likelihood accuracy similar to LAN, but longer evaluation times due to the complexity of the underlying normalizing flow architecture.

To reduce evaluation time of MNLE, we further adapted the architecture to the example model (DDM). In particular, we ran a cross-validation of the hyperparameters relevant for evaluation time, that is, number of hidden layers, hidden units, NSF transforms, spline bins, and selected those that were optimal in terms of Huber loss and MSE between the approximate and the analytical likelihoods, as well as evaluation time. This resulted in an architecture with performance and evaluation time similar to LANs (more details in Appendix: Architecture and training hyperparameters). The cross-validation relied on access to the analytical likelihoods which is usually not given in practice, for example, for simulators with intractable likelihoods. However, we note that in cases without access to analytical likelihoods a similar cross-validation can be performed using quality measures other than the difference to the analytical likelihood, for example, by comparing the observed data with synthetic data and SLs provided by MNLE.

Acknowledgements

We thank Luigi Acerbi, Michael Deistler, Alexander Fengler, Michael Frank, and Ingeborg Wenger for discussions and comments on a preliminary version of the manuscript. We also acknowledge and thank the Python and Julia communities for developing the tools enabling this work, including DiffErentialEquations.jl, DiffModels.jl, NumPy, pandas, Pyro, PyTorch, sbi, sbibm, and Scikit-learn (see Appendix for details).

Additional information

Funding

Funder	Grant reference number	Author
Deutsche Forschungsgemeinschaft	SFB 1233	Jan-Matthis Lueckmann Jakob H Macke
Deutsche Forschungsgemeinschaft	SPP 2041	Jan Boelts Jakob H Macke
Deutsche Forschungsgemeinschaft	Germany's Excellence Strategy MLCoE	Jan Boelts Jan-Matthis Lueckmann Richard Gao Jakob H Macke
Bundesministerium für Bildung und Forschung	ADIMEM	Jan-Matthis Lueckmann Jakob H Macke
HORIZON EUROPE Marie Skłodowska-Curie Actions	101030918	Richard Gao
Bundesministerium für Bildung und Forschung	Tübingen AI Center	Jan Boelts Jakob H Macke
Bundesministerium für Bildung und Forschung	FKZ 01IS18052 A-D	Jan-Matthis Lueckmann Jakob H Macke
Bundesministerium für Bildung und Forschung	KZ 01IS18039A	Jan Boelts Jakob H Macke

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


Author contributions

Jan Boelts, Conceptualization, Resources, Data curation, Software, Formal analysis, Validation, Investigation, Visualization, Methodology, Writing – original draft, Writing – review and editing; Jan-Matthis Lueckmann, Conceptualization, Supervision, Visualization, Methodology, Writing – original draft, Writing – review and editing; Richard Gao, Conceptualization, Software, Validation, Visualization, Writing – original draft, Writing – review and editing; Jakob H Macke, Conceptualization, Supervision, Funding acquisition, Methodology, Project administration, Writing – review and editing

Author ORCIDs

Jan Boelts  <http://orcid.org/0000-0003-4979-7092>

Richard Gao  <http://orcid.org/0000-0001-5916-6433>

Jakob H Macke  <http://orcid.org/0000-0001-5154-8912>

Decision letter and Author response

Decision letter <https://doi.org/10.7554/eLife.77220.sa1>

Author response <https://doi.org/10.7554/eLife.77220.sa2>

Additional files

Supplementary files

- Transparent reporting form

Data availability

We implemented MNLE as part of the open source package for SBI, *sbi*, available at <https://github.com/mackelab/sbi>, copy archived at [swh:1:rev:d72fc6d790285c7779afb9a5f6b640691d4560](https://swh.io/rev:d72fc6d790285c7779afb9a5f6b640691d4560). Code for reproducing the results presented here, and tutorials on how to apply MNLE to other simulators using *sbi* can be found at <https://github.com/mackelab/mnle-for-ddms>, copy archived at [swh:1:rev:5e6cf714c223ec5c414b76ac70f7dc88d4fbd321](https://swh.io/rev:5e6cf714c223ec5c414b76ac70f7dc88d4fbd321).

References

- An Z**, South LF, Nott DJ, Drovandi CC. 2019. Accelerating bayesian synthetic likelihood with the graphical lasso. *Journal of Computational and Graphical Statistics* **28**:471–475. DOI: <https://doi.org/10.1080/10618600.2018.1537928>
- Bezanson J**, Edelman A, Karpinski S, Shah VB. 2017. Julia: A fresh approach to numerical computing. *SIAM Review* **59**:65–98. DOI: <https://doi.org/10.1137/141000671>
- Bingham E**, Chen JP, Jankowiak M, Obermeyer F, Pradhan N, Karaletsos T, Singh R, Szerlip P, Horsfall P, Goodman ND. 2019. Pyro: Deep universal probabilistic programming. *Journal of Machine Learning Research* **20**:973–978.
- Boelts J**. 2022a. *sbi*: simulation-based inference. [swh:1:rev:d72fc6d790285c7779afb9a5f6b640691d4560](https://swh.io/rev:d72fc6d790285c7779afb9a5f6b640691d4560). Software Heritage. <https://archive.softwareheritage.org/swh:1:dir:d327a49bdf0127726927c70c6b216405196653d6;origin=https://github.com/mackelab/sbi;visit=swh:1:snp:ff58cc8344d88bbe9952872597690748adb5798b;anchor=swh:1:rev:d72fc6d790285c7779afb9a5f6b640691d4560>
- Boelts J**. 2022b. Mixed neural likelihood estimation for models of decision-making. [swh:1:rev:5e6cf714c223ec5c414b76ac70f7dc88d4fbd321](https://swh.io/rev:5e6cf714c223ec5c414b76ac70f7dc88d4fbd321). Software Heritage. <https://archive.softwareheritage.org/swh:1:dir:e20ac3b3d5324d29f262cc52388b3916526d2518;origin=https://github.com/mackelab/mnle-for-ddms;visit=swh:1:snp:b9707619efaa06519bb97d54db256cc99f78df3f;anchor=swh:1:rev:5e6cf714c223ec5c414b76ac70f7dc88d4fbd321>
- Brooks S**, Gelman A, Jones G, Meng XL. 2011. MCMC Using Hamiltonian Dynamics. Brooks S (Ed). *Handbook of Markov Chain Monte Carlo*. Elsevier. p. 1–2. DOI: <https://doi.org/10.1201/b10905>
- Chandrasekaran C**, Hawkins GE. 2019. ChaRTr: An R toolbox for modeling choices and response times in decision-making tasks. *Journal of Neuroscience Methods* **328**:108432. DOI: <https://doi.org/10.1016/j.jneumeth.2019.108432>, PMID: 31586868
- Churchland PS**, Sejnowski TJ. 1988. Perspectives on cognitive neuroscience. *Science* **242**:741–745. DOI: <https://doi.org/10.1126/science.3055294>, PMID: 3055294
- 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>
- Cranmer K**, Brehmer J, Louppe G. 2020. The frontier of simulation-based inference. *PNAS* **117**:30055–30062. DOI: <https://doi.org/10.1073/pnas.1912789117>, PMID: 32471948
- Dax M**, Green SR, Gair J, Deistler M, Schölkopf B, Macke JH. 2022. Group equivariant neural posterior estimation. In International Conference on Learning Representations. .
- Drugowitsch J**. 2016. Fast and accurate Monte Carlo sampling of first-passage times from Wiener diffusion models. *Scientific Reports* **6**:1–13. DOI: <https://doi.org/10.1038/srep20490>, PMID: 26864391
- Durkan C**, Bekasov A, Murray I, Papamakarios G. 2019. Neural spline flows. *Advances in Neural Information Processing Systems*. 7511–7522.
- Durkan C**, Murray I, Papamakarios G. 2020. On contrastive learning for likelihood-free inference. In International Conference on Machine Learning. 2771–2781.
- Feltgen Q**, Daunizeau J. 2021. An overcomplete approach to fitting drift-diffusion decision models to trial-by-trial data. *Frontiers in Artificial Intelligence* **4**:531316. DOI: <https://doi.org/10.3389/frai.2021.531316>, PMID: 33898982
- Fengler A**, Govindarajan LN, Chen T, Frank MJ. 2021. Likelihood approximation networks (LANs) for fast inference of simulation models in cognitive neuroscience. *eLife* **10**:e65074. DOI: <https://doi.org/10.7554/eLife.65074>, PMID: 33821788
- Glöckler M**, Deistler M, Macke JH. 2021. Variational methods for simulation-based inference. In International Conference on Learning Representations. .
- Gonçalves PJ**, Lueckmann JM, Deistler M, Nonnenmacher M, Öcal K, Bassetto G, Chintaluri C, Podlaski WF, Haddad SA, Vogels TP, Greenberg DS, Macke JH. 2020. Training deep neural density estimators to identify mechanistic models of neural dynamics. *eLife* **9**:e56261. DOI: <https://doi.org/10.7554/eLife.56261>, PMID: 32940606
- Greenberg D**, Nonnenmacher M, Macke J. 2019. Automatic posterior transformation for likelihood-free inference. In Proceedings of the 36th International Conference on Machine Learning of Proceedings of Machine Learning Research. 2404–2414.
- 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.
- Harris CR**, Millman KJ, van der Walt SJ, Gommers R, Virtanen P, Cournapeau D, Wieser E, Taylor J, Berg S, Smith NJ, Kern R, Picus M, Hoyer S, van Kerkwijk MH, Brett M, Haldane A, Del Río JF, Wiebe M, Peterson P,

- Gérard-Marchant P, et al. 2020. Array programming with NumPy. *Nature* **585**:357–362. DOI: <https://doi.org/10.1038/s41586-020-2649-2>, PMID: 32939066
- Hawkins GE, Forstmann BU, Wagenmakers EJ, Ratcliff R, Brown SD. 2015. Revisiting the evidence for collapsing boundaries and urgency signals in perceptual decision-making. *The Journal of Neuroscience* **35**:2476–2484. DOI: <https://doi.org/10.1523/JNEUROSCI.2410-14.2015>, PMID: 25673842
- Heathcote A, Lin YS, Reynolds A, Strickland L, Grettton M, Matzke D. 2019. Dynamic models of choice. *Behavior Research Methods* **51**:961–985. DOI: <https://doi.org/10.3758/s13428-018-1067-y>, PMID: 29959755
- Hermans J, Begy V, Louppe G. 2020. Likelihood-free mcmc with approximate likelihood ratios. In Proceedings of the 37th International Conference on Machine Learning of Proceedings of Machine Learning Research. .
- Hoffman MD, Gelman A. 2014. The No-U-turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research: JMLR* **15**:1593–1623.
- Järvenpää M, Gutmann MU, Pleska A, Vehtari A, Marttinen P. 2019. Efficient acquisition rules for model-based approximate bayesian computation. *Bayesian Analysis* **14**:595–622. DOI: <https://doi.org/10.1214/18-BA1121>
- Kangasrääsiö A, Jokinen JP, Oulasvirta A, Howes A, Kaski S. 2019. Parameter inference for computational cognitive models with approximate bayesian computation. *Cognitive Science* **43**:e12738. DOI: <https://doi.org/10.1111/cogs.12738>
- Kingma DP, Ba J. 2015. Adam: A method for stochastic optimization. In Proceedings of the 3rd International Conference on Learning Representations, ICLR. .
- Lee MD. 2008. Three case studies in the Bayesian analysis of cognitive models. *Psychonomic Bulletin & Review* **15**:1–15. DOI: <https://doi.org/10.3758/pbr.15.1.1>, PMID: 18605474
- Lee MD. 2011. How cognitive modeling can benefit from hierarchical Bayesian models. *Journal of Mathematical Psychology* **55**:1–7. DOI: <https://doi.org/10.1016/j.jmp.2010.08.013>
- Lee MD, Wagenmakers EJ. 2014. Bayesian Cognitive Modeling. Cambridge university press. DOI: <https://doi.org/10.1017/CBO9781139087759>
- Lee HS, Betts S, Anderson JR. 2016. Learning problem-solving rules as search through a hypothesis space. *Cognitive Science* **40**:1036–1079. DOI: <https://doi.org/10.1111/cogs.12275>, PMID: 26292648
- Lopez-Paz D, Oquab M. 2017. Revisiting classifier two-sample tests. In 5th International Conference on Learning Representations, ICLR. .
- Lueckmann JM, Goncalves PJ, Bassetto G, Öcal K, Nonnenmacher M, Macke JH. 2017. Flexible Statistical Inference for Mechanistic Models of Neural Dynamics. *arXiv*. <https://arxiv.org/abs/1711.01861>
- Lueckmann JM, Bassetto G, Karaletsos T, Macke JH. 2019. Likelihood-free inference with emulator networks. In Proceedings of The 1st Symposium on Advances in Approximate Bayesian Inference of Proceedings of Machine Learning Research. 32–53.
- Lueckmann JM, Boelts J, Greenberg D, Goncalves P, Macke J. 2021. Benchmarking simulation-based inference. Proceedings of The 24th International Conference on Artificial Intelligence and Statistics of Proceedings of Machine Learning Research. 343–351.
- McClelland JL. 2009. The place of modeling in cognitive science. *Topics in Cognitive Science* **1**:11–38. DOI: <https://doi.org/10.1111/j.1756-8765.2008.01003.x>, PMID: 25164798
- Navarro DJ, Fuss IG. 2009. Fast and accurate calculations for first-passage times in Wiener diffusion models. *Journal of Mathematical Psychology* **53**:222–230. DOI: <https://doi.org/10.1016/j.jmp.2009.02.003>
- Neal RM. 2009. Slice sampling. *The Annals of Statistics* **31**:e62461. DOI: <https://doi.org/10.1214/aos/1056562461>
- Ong VMH, Nott DJ, Tran M-N, Sisson SA, Drovandi CC. 2009. Variational Bayes with synthetic likelihood. *Statistics and Computing* **28**:971–988. DOI: <https://doi.org/10.1007/s11222-017-9773-3>
- Palestro JJ, Sederberg PB, Osth AF, Van Zandt T, Turner BM. 2009. Likelihood-Free Methods for Cognitive Science. Cham: Springer. DOI: <https://doi.org/10.1007/978-3-319-72425-6>
- Palestro JJ, Weichart E, Sederberg PB, Turner BM. 2018. Some task demands induce collapsing bounds: Evidence from a behavioral analysis. *Psychonomic Bulletin & Review* **25**:1225–1248. DOI: <https://doi.org/10.3758/s13423-018-1479-9>, PMID: 29845433
- pandas development team. 2020. Pandas-dev/pandas: pandas. 6e1a040. Github. <https://github.com/pandas-dev/pandas>
- Papamakarios G, Murray I. 2016. Fast ϵ -free Inference of Simulation Models with Bayesian Conditional Density Estimation. In Advances in Neural Information Processing Systems. .
- Papamakarios G, Nalisnick E, Rezende DJ, Mohamed S, Lakshminarayanan B. 2019a. Normalizing Flows for Probabilistic Modeling and Inference. *arXiv*. <https://arxiv.org/abs/1912.02762>
- Papamakarios G, Sterratt D, Murray I. 2019b. Sequential neural likelihood: Fast likelihood-free inference with autoregressive flows. In Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS) of Proceedings of Machine Learning Research. 837–848.
- Paszke A, Gross S, Massa F, Lerer A, Bradbury J, Chanan G, Killeen T, Lin Z, Gimelshein N, Antiga L, Desmaison A, Kopf A, Yang E, DeVito Z, Raison M, Tejani A, Chilamkurthy S, Steiner B, Fang L, Bai J, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. In Advances in Neural Information Processing Systems. 8024–8035.
- Patil U, Hanne S, Burchert F, De Bleser R, Vasishth S. 2016. A computational evaluation of sentence processing deficits in aphasia. *Cognitive Science* **40**:5–50. DOI: <https://doi.org/10.1111/cogs.12250>, PMID: 26016698
- Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay E. 2011. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research* **12**:2825–2830.

- Price LF**, Drovandi CC, Lee A, Nott DJ. 2018. Bayesian Synthetic Likelihood. *Journal of Computational and Graphical Statistics* **27**:1–11. DOI: <https://doi.org/10.1080/10618600.2017.1302882>
- Priddle JW**, Sisson SA, Frazier DT, Turner I, Drovandi C. 2022. Efficient bayesian synthetic likelihood with whitening transformations. *Journal of Computational and Graphical Statistics* **31**:50–63. DOI: <https://doi.org/10.1080/10618600.2021.1979012>
- Rackauckas C**, Nie Q. 2017. DifferentialEquations.jl – a performant and feature-rich ecosystem for solving differential equations in julia. *Journal of Open Research Software* **5**:15. DOI: <https://doi.org/10.5334/jors.151>
- Radev ST**, Mertens UK, Voss A, Ardizzone L, Kothe U. 2022. BayesFlow: learning complex stochastic models with invertible neural networks. *IEEE Transactions on Neural Networks and Learning Systems* **33**:1452–1466. DOI: <https://doi.org/10.1109/TNNLS.2020.3042395>, PMID: 33338021
- Ratcliff R**, Rouder JN. 1998. Modeling response times for two-choice decisions. *Psychological Science* **9**:347–356. DOI: <https://doi.org/10.1111/1467-9280.00067>
- 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
- Reynolds AM**, Rhodes CJ. 2009. The Lévy flight paradigm: random search patterns and mechanisms. *Ecology* **90**:877–887. DOI: <https://doi.org/10.1890/08-0153.1>, PMID: 19449680
- Schad DJ**, Betancourt M, Vasishth S. 2021. Toward a principled Bayesian workflow in cognitive science. *Psychological Methods* **26**:103–126. DOI: <https://doi.org/10.1037/met0000275>, PMID: 32551748
- Shiffrin RM**, Lee MD, Kim W, Wagenmakers EJ. 2008. A survey of model evaluation approaches with A tutorial on hierarchical bayesian methods. *Cognitive Science* **32**:1248–1284. DOI: <https://doi.org/10.1080/03640210802414826>, PMID: 21585453
- Shinn M**, Lam NH, Murray JD. 2020. A flexible framework for simulating and fitting generalized drift-diffusion models. *eLife* **9**:e56938. DOI: <https://doi.org/10.7554/eLife.56938>, PMID: 32749218
- Sisson SA**. 2018. Overview of abc. Sisson SA, Fan Y, Beaumont MA (Eds). In *Handbook of Approximate Bayesian Computation, Chapter 1*. CRC Press, Taylor & Francis Group. p. 1–678. DOI: <https://doi.org/10.1201/9781315117195>
- 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>
- Tejero-Cantero A**, Boelts J, Deistler M, Lueckmann JM, 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>
- Tran D**, Vafa K, Agrawal K, Dinh L, Poole B. 2019. Discrete flows: Invertible generative models of discrete data. *Advances in Neural Information Processing Systems*. 14719–14728.
- Turner BM**, Van Zandt T. 2012. A tutorial on approximate bayesian computation. *Journal of Mathematical Psychology* **56**:69–85. DOI: <https://doi.org/10.1016/j.jmp.2012.02.005>
- Turner BM**, van Maanen L, Forstmann BU. 2015. Informing cognitive abstractions through neuroimaging: the neural drift diffusion model. *Psychological Review* **122**:312–336. DOI: <https://doi.org/10.1037/a0038894>, PMID: 25844875
- Turner BM**, Van Zandt T. 2018. Approximating bayesian inference through model simulation. *Trends in Cognitive Sciences* **22**:826–840. DOI: <https://doi.org/10.1016/j.tics.2018.06.003>, PMID: 30093313
- Usher M**, McClelland JL. 2001. The time course of perceptual choice: the leaky, competing accumulator model. *Psychological Review* **108**:550–592. DOI: <https://doi.org/10.1037/0033-295x.108.3.550>, PMID: 11488378
- van Opheusden B**, Acerbi L, Ma WJ. 2020. Unbiased and efficient log-likelihood estimation with inverse binomial sampling. *PLOS Computational Biology* **16**:e1008483. DOI: <https://doi.org/10.1371/journal.pcbi.1008483>, PMID: 33362195
- Van Rossum G**, Drake FL. 1995. Python Tutorial. Centrum Voor Wiskunde En Informatica.
- von Krause M**, Radev ST, Voss A. 2022. Mental speed is high until age 60 as revealed by analysis of over a million participants. *Nature Human Behaviour* **6**:700–708. DOI: <https://doi.org/10.1038/s41562-021-01282-7>, PMID: 35177809
- Voss A**, Voss J. 2007. Fast-dm: A free program for efficient diffusion model analysis. *Behavior Research Methods* **39**:767–775. DOI: <https://doi.org/10.3758/bf03192967>, PMID: 18183889
- Wagenmakers EJ**, van der Maas HLJ, Grasman R. 2007. An EZ-diffusion model for response time and accuracy. *Psychonomic Bulletin & Review* **14**:3–22. DOI: <https://doi.org/10.3758/bf03194023>, PMID: 17546727
- Wiecki TV**, Sofer I, Frank MJ. 2013. HDDM: Hierarchical bayesian estimation of the drift-diffusion model in python. *Frontiers in Neuroinformatics* **7**:14. DOI: <https://doi.org/10.3389/fninf.2013.00014>, PMID: 23935581
- Wiqvist S**, Frellsen J, Picchini U. 2021. Sequential Neural Posterior and Likelihood Approximation. *arXiv*. <https://arxiv.org/abs/2102.06522>
- Wood SN**. 2010. Statistical inference for noisy nonlinear ecological dynamic systems. *Nature* **466**:1102–1104. DOI: <https://doi.org/10.1038/nature09319>, PMID: 20703226

Appendix 1

Code availability

We implemented MNLE as part of the open-source package for SBI, *sbi*, available at <https://github.com/mackelab/sbi>, (Boelts, 2022a copy archived at [swh:1:rev:d72fc6d790285c7779afb9a5f6b640691d4560](https://doi.org/10.1101/2022.05.11.491456)). Code for reproducing the results presented here, and tutorials on how to apply MNLE to other simulators using *sbi* can be found at <https://github.com/mackelab/mnle-for-ddms>, (Boelts, 2022b copy archived at [swh:1:rev:5e6cf714c223ec5c414b76ac70f7dc88d4fbd321](https://doi.org/10.1101/2022.05.11.491456)). The implementation of MNLE relies on packages developed by the Python (Van Rossum and Drake, 1995) and Julia (Bezanson et al., 2017) communities, including DifferentialEquations.jl (Rackauckas and Nie, 2017), DiffModels.jl (Drugowitsch, 2016), NumPy (Harris et al., 2020), pandas (pandas development team, 2020), Pyro (Bingham et al., 2019), PyTorch (Paszke et al., 2019), *sbi* (Tejero-Cantero et al., 2020), *sbibm* (Lueckmann et al., 2021), and Scikit-learn (Pedregosa et al., 2011).

Architecture and training hyperparameters

For the Bernoulli neural network we used three hidden layers with 10 units each and sigmoid activation functions. For the neural spline flow architecture (Durkan et al., 2019), we transformed the reaction time data to the log-domain, used a standard normal base distribution, 2 spline transforms with 5 bins each and conditioning networks with 3 hidden layers and 10 hidden units each, and rectified linear unit activation functions. The neural network training was performed using the *sbi* package with the following settings: learning rate 0.0005; training batch size 100; 10% of training data as validation data, stop training after 20 epochs without validation loss improvement.

The emulator property of MNLE

Being based on the neural likelihood estimation framework, MNLE naturally returns an emulator of the simulator that can be sampled to generate synthetic data without running the simulator. We found that the synthetic data generated by MNLE accurately matched the data we obtained by running the DDM simulator (Figure 2—figure supplement 1). This has several potential benefits: it can help with evaluating the performance of the density estimator, it enables almost instantaneous data generation (one forward-pass in the neural network) even if the simulator is computationally expensive, and it gives full access to the internals of the emulator, for example, to gradients w.r.t. to data or parameters.

There is variant of the LAN approach which allows for sampling synthetic data as well: In the ‘Histogram-approach’ (Fengler et al., 2021) LANs are trained with a convolutional neural network (CNN) architecture using likelihood targets in form of two-dimensional empirical histograms. The output of the CNN is a probability distribution over a discretized version of the data space which can, in principle, be sampled to generate synthetic DDM choices and reaction times. However, the accuracy of this emulator property of CNN-LANs is limited by the number of bins used to approximate the continuous data space (e.g., 512 bins for the examples shown in Fengler et al., 2021).

1 Simulation-based inference for 2 efficient identification of generative 3 models in computational 4 connectomics

5 Jan Boelts ^{1,2} , Philipp Harth ³, Richard Gao ^{1,2}, Daniel Udvary ⁴, Felipe
6 Yáñez ⁴, Daniel Baum ³, Hans-Christian Hege ³, Marcel Oberlaender ⁴, Jakob
7 H. Macke ^{1,2,5}

8 ¹Machine Learning in Science, University of Tübingen; ²Tübingen AI Center, University
9 of Tübingen; ³Department of Visual and Data-centric Computing, Zuse Institute Berlin;
10 ⁴In Silico Brain Sciences, Max Planck Institute for Neurobiology of Behavior – caesar,
11 Bonn; ⁵Empirical Inference, Max Planck Institute for Intelligent Systems, Tübingen

12

 **For correspondence:**
jan.boelts@uni-tuebingen.de
(JB)

Code availability: Source
code for reproducing the
presented results is available
at [GitHub](#).

Competing interests: The
authors declare no competing
interests.

13 Abstract

14 Recent advances in connectomics research enable the acquisition of increasing amounts of data
15 about the connectivity patterns of neurons. How can we use this wealth of data to efficiently
16 derive and test hypotheses about the principles underlying these patterns? A common approach
17 is to simulate neural networks using a hypothesized wiring rule in a generative model and to
18 compare the resulting synthetic data with empirical data. However, most wiring rules have at
19 least some free parameters, and identifying parameters that reproduce empirical data can be
20 challenging as it often requires manual parameter tuning. Here, we propose to use
21 simulation-based Bayesian inference (SBI) to address this challenge. Rather than optimizing a
22 single rule to fit the empirical data, SBI considers many parametrizations of a wiring rule and
23 performs Bayesian inference to identify the parameters that are compatible with the data. It uses
24 simulated data from multiple candidate wiring rules and relies on machine learning methods to
25 estimate a probability distribution (the ‘posterior distribution over rule parameters conditioned
26 on the data’) that characterizes all data-compatible rules. We demonstrate how to apply SBI in
27 connectomics by inferring the parameters of wiring rules in an *in silico* model of the rat barrel
28 cortex, given *in vivo* connectivity measurements. SBI identifies a wide range of wiring rule
29 parameters that reproduce the measurements. We show how access to the posterior distribution
30 over all data-compatible parameters allows us to analyze their relationship, revealing biologically
31 plausible parameter interactions and enabling experimentally testable predictions. We further
32 show how SBI can be applied to wiring rules at different spatial scales to quantitatively rule out
33 invalid wiring hypotheses. Our approach is applicable to a wide range of generative models used
34 in connectomics, providing a quantitative and efficient way to constrain model parameters with
35 empirical connectivity data.

36

37 Author summary

38 The brain is composed of an intricately connected network of cells—what are the factors that con-
39 tribute to constructing these patterns of connectivity, and how? To answer these questions, amass-
40 ing connectivity data alone is not enough. We must also be able to efficiently develop and test our
41 ideas about the underlying connectivity principles. For example, we could simulate a hypothetical
42 wiring rule like “neurons near each other are more likely to form connections” in a computational
43 model and generate corresponding synthetic data. If the synthetic, simulated data resembles the
44 real, measured data, then we have some confidence that our hypotheses might be correct. The
45 challenge, however, lies in finding all the potential wiring rules, or equivalently, all the parameters
46 of the computational model that can reproduce the observed data, as this process is often idiosyn-
47 cratic and labor-intensive. To tackle this challenge, we introduce an approach that combines com-
48 putational modeling in connectomics, deep learning, and Bayesian statistical inference in order to
49 automatically infer a probability distribution over the model parameters likely to explain the data.
50 We demonstrate our approach by inferring wiring rules in a detailed model of the rat barrel cortex
51 and find that the inferred distribution identifies multiple data-compatible model parameters, re-
52 veals biologically plausible parameter interactions, and allows us to make experimentally testable
53 predictions.

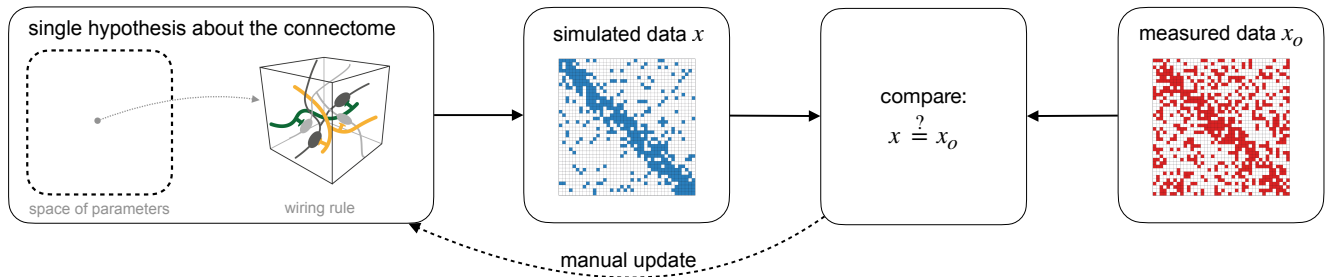
54 Introduction

55 Connectomics investigates the structural and functional composition of neural networks to distill
56 principles of the connectivity patterns underlying brain function (*Chklovskii et al., 2004; Sporns*
57 *et al., 2005*). Over the last years, advances in imaging and tracing techniques enabled the acquisi-
58 tion of increasingly detailed connectivity data (*Osten and Margrie, 2013; Kornfeld and Denk, 2018;*
59 *Macrina et al., 2021*) and led to significant insights (*Motta et al., 2019; Valdes-Aleman et al., 2021;*
60 *Loomba et al., 2022*). These advances in data acquisition necessitate new computational tools for
61 analyzing the data and testing hypotheses derived from it (*Jain et al., 2010; Sporns and Bassett,*
62 *2018; Peyser et al., 2019*). A recent computational approach for testing hypotheses in connectomics
63 has been to use so-called *generative models* (*Betzell and Bassett, 2017; Váša and Mišić, 2022; Luppi*
64 *et al., 2022*). The idea of generative modeling is to develop a computational model capable of gen-
65 erating synthetic connectivity data according to a specific hypothesis, e.g., a wiring rule (Fig. 1a,
66 left). Subsequently, one can validate and refine the wiring rule (or the underlying computational
67 model) by comparing the simulated with measured connectivity data (Fig. 1a, right). Examples for
68 this approach range from large-scale generative models of functional connectivity in the human
69 cortex (*Vértes et al., 2012; Betzell et al., 2016*), system-level network models of the mouse visual
70 cortex (*Billeh et al., 2020*), and generative models of cortical microcircuits (*Reimann et al., 2015*).

71 As a specific example, we here consider a generative model for simulating hypothesized wiring
72 rules in the rat barrel cortex (*Udvary et al., 2022*). The model is based on reconstructions of axon
73 and dendrite morphologies from *in vivo* recordings (*Narayanan et al., 2015*) and reconstructions
74 of the barrel cortex geometry, cytoarchitecture, and cellular organization (*Meyer et al., 2010, 2013*).
75 These anatomical features were combined into a 3D model to obtain a quantitative and realistic
76 estimate of the dense neuropil structure for a large volume of the rat barrel cortex (*Egger et al.,*
77 *2014; Udvary et al., 2022*). Thus, by applying a hypothesized wiring rule to the structural features
78 of the model, one can generate a corresponding synthetic barrel cortex connectome and compare
79 it to empirical data to test the validity of the wiring rule. For example, *Udvary et al. (2022)* used
80 the barrel cortex model to show that a wiring rule that only takes into account neuron morphology
81 predicts connectivity patterns that are consistent with those observed empirically in the barrel
82 cortex.

83 However, building generative models that accurately reproduce connectivity measurements
84 can be challenging: Suppose a hypothesized wiring rule does not reproduce the data. In that case,
85 a common approach would be manually refining the rule, e.g., by introducing parameters and

a Conventional generative modeling



b Automated model identification with SBI

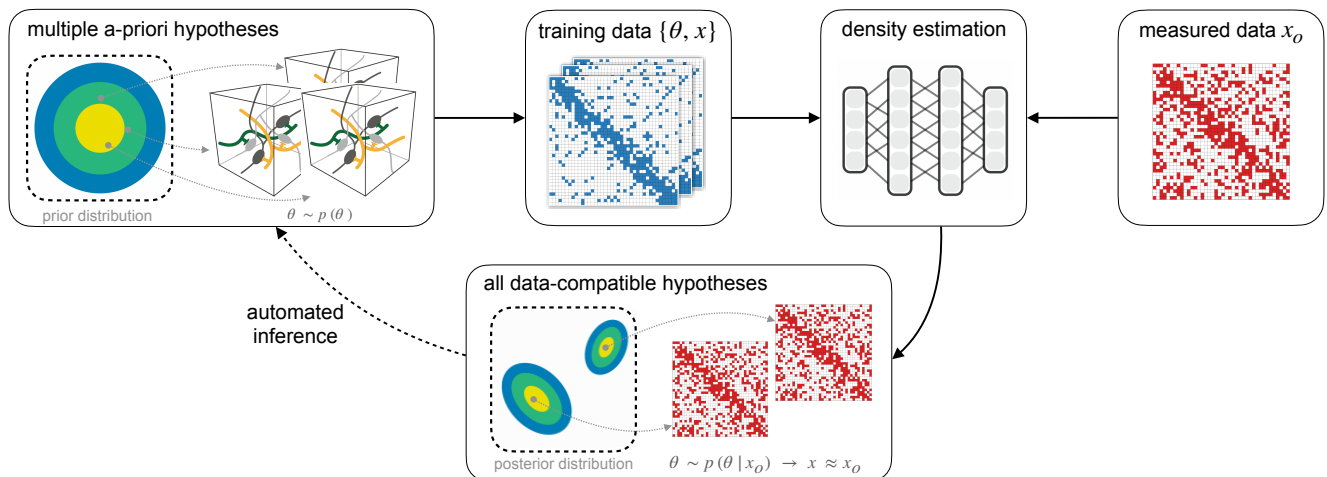


Figure 1. Enhancing generative modeling in connectomics with simulation-based inference. (a) Generative modeling is a common approach for testing hypotheses about the connectome: One implements a hypothesized wiring rule as a computational model that simulates connectivity data x (left) and then tests and manually refines the rule by comparing simulated with measured data x_o (right). (b) Our goal is to make this approach more efficient using simulation-based Bayesian inference (SBI): By equipping the generative model with parameters θ , we define a space of multiple *a-priori* hypotheses (left) from which we can generate multiple simulated data x (middle). We then use the simulated data to perform density estimation with artificial neural networks to estimate the *posterior distribution* over model parameters conditioned on the measured data, i.e., $p(\theta|x_o)$. The inferred posterior distribution characterizes *all* wiring rule parameters compatible with the measured data, replacing the manual refinement of single wiring rules in the conventional approach (bottom).

86 repeating the simulate-and-compare-to-measurements loop (Fig. 1a), which can be laborious and
 87 inefficient. Additionally, identifying one specific wiring rule configuration for which simulated and
 88 empirical data match might not be enough: Given that the available empirical connectivity data
 89 is sparse compared to the structural and functional complexity of the connectome, it is likely that
 90 there are many data-compatible wiring rules and we would need to repeat the search to identify
 91 them all.

92 To address these challenges, we propose a new approach that employs Bayesian inference to
 93 replace the manual comparison of individual wiring rules (Fig. 1a) with the automated inference of
 94 multiple wiring rules (Fig. 1b). We achieve this by taking two conceptual steps: First, we equip the
 95 generative model with parameters θ and interpret different parameter combinations as variants of
 96 the underlying hypothesis, e.g., variants of the wiring rule. Second, we define a probability distribu-
 97 tion over the model parameters such that each parameter configuration corresponds to a different
 98 candidate wiring rule, i.e., a *prior distribution* $p(\theta)$ (Fig. 1b, left), and use Bayesian inference to infer
 99 all data-compatible parameters. Given measured connectivity data x_o and a parametrized genera-
 100 tive model, we infer the conditional probability distribution over the model parameters given the

101 measured data, i.e., the *posterior distribution* $p(\theta|x_o)$. The posterior distribution characterizes *all*
102 parameter configurations (wiring rules) likely to explain the measured data, in contrast to conven-
103 tional approaches that often optimize for one best-fitting parameter. For example, by sampling
104 different parameters from the inferred posterior we would obtain different wiring rule configura-
105 tions all of which are likely to generate data similar to the measured data (Fig. 1 b, bottom). Addition-
106 ally, the posterior distribution also allows us to quantify the correlations between the parameters,
107 which can help to reveal parameter interactions and potential compensation mechanisms in the
108 model.

109 On a technical level, standard Bayesian inference methods usually require access to the like-
110 lihood function of the model. However, generative models employed in computational connec-
111 tomics are often defined as computer simulations for which the likelihood may not be easily accessi-
112 ble. Therefore, we propose using simulation-based inference (SBI, *Cranmer et al., 2020; Gonçalves*
113 *et al., 2020; Papamakarios and Murray, 2016; Lueckmann et al., 2017; Greenberg et al., 2019*). SBI
114 enables Bayesian inference using only simulated data from the model, i.e., without requiring ac-
115 cess to the likelihood. In particular, SBI performs conditional density estimation with artificial neu-
116 ral networks: It uses data simulated from the model to train an artificial neural network that takes
117 data as input and predicts an approximation to the posterior distribution. Once trained on simu-
118 lated data, the neural network can be applied to the measured data to obtain the desired posterior
119 distribution $p(\theta|x_o)$ (Fig. 1 b, right).

120 We demonstrate our approach using the example of constraining wiring rules in the structural
121 model of the rat barrel cortex introduced above. First, we show how to reformulate wiring rules
122 as parametrized models to make them amenable to Bayesian inference. The resulting generative
123 model consists of the parametrized wiring rule applied to the structural model to generate a simu-
124 lated connectome of the rat barrel cortex. Second, we show that SBI can identify all parameter
125 configurations that agree with measured connectivity data. When testing our approach in a scen-
126 ario with simulated data and a known reference solution, we find that SBI performs accurately. In
127 the realistic setting with measured connectivity data, SBI identifies a large set of rule configurations
128 that reproduce observed and predict unobserved features of the connectome. Importantly, ana-
129 lyzing the inferred posterior reveals that this set of plausible rules is highly structured and reflects
130 biologically interpretable interactions of the parameters. Finally, we illustrate the flexibility of the
131 SBI approach by inferring two proximity-based wiring rules at different spatial scales to quantita-
132 tively show that Peters' rule cannot explain connectivity measurements in the barrel cortex.

133 Our approach provides a new quantitative and efficient tool for constraining model paramet-
134 ers with connectivity measurements and is applicable to many generative models used in connec-
135 tomics. For example, it sets the stage for building generative models based on dense reconstruc-
136 tions of brain tissue (e.g., *MICrONS-Consortium et al., 2021; Shapson-Coe et al., 2021; Turner et al.,*
137 *2022*) and inferring underlying connectivity principles using SBI. We are making all software tools
138 required for applying SBI available in an open-source software package (*sbi, Tejero-Cantero* et al.,*
139 *2020*), facilitating its use by researchers across the field.

140 Results

141 Formulating wiring rules in the rat barrel cortex as simulation-based models

142 To demonstrate the potential of simulation-based inference (SBI) for connectomics, we selected
143 the problem of constraining wiring rules in the rat barrel cortex with empirical connectivity data.
144 Applying SBI requires three ingredients: a simulation-based model with free parameters, a prior
145 distribution over the parameters, and measured data (see *Methods & Materials* for details). Our
146 analyses are based on a digital model of the dense neuropil structure of the rat barrel cortex (*Eg-*
147 *ger et al., 2014; Udvary et al., 2022*), which we extended to obtain a simulation-based model. The
148 model contains reconstructions of the number and distribution of somata, axon and dendrite mor-
149 phologies, and subcellular features like pre-synaptic boutons and post-synaptic dendritic spines.

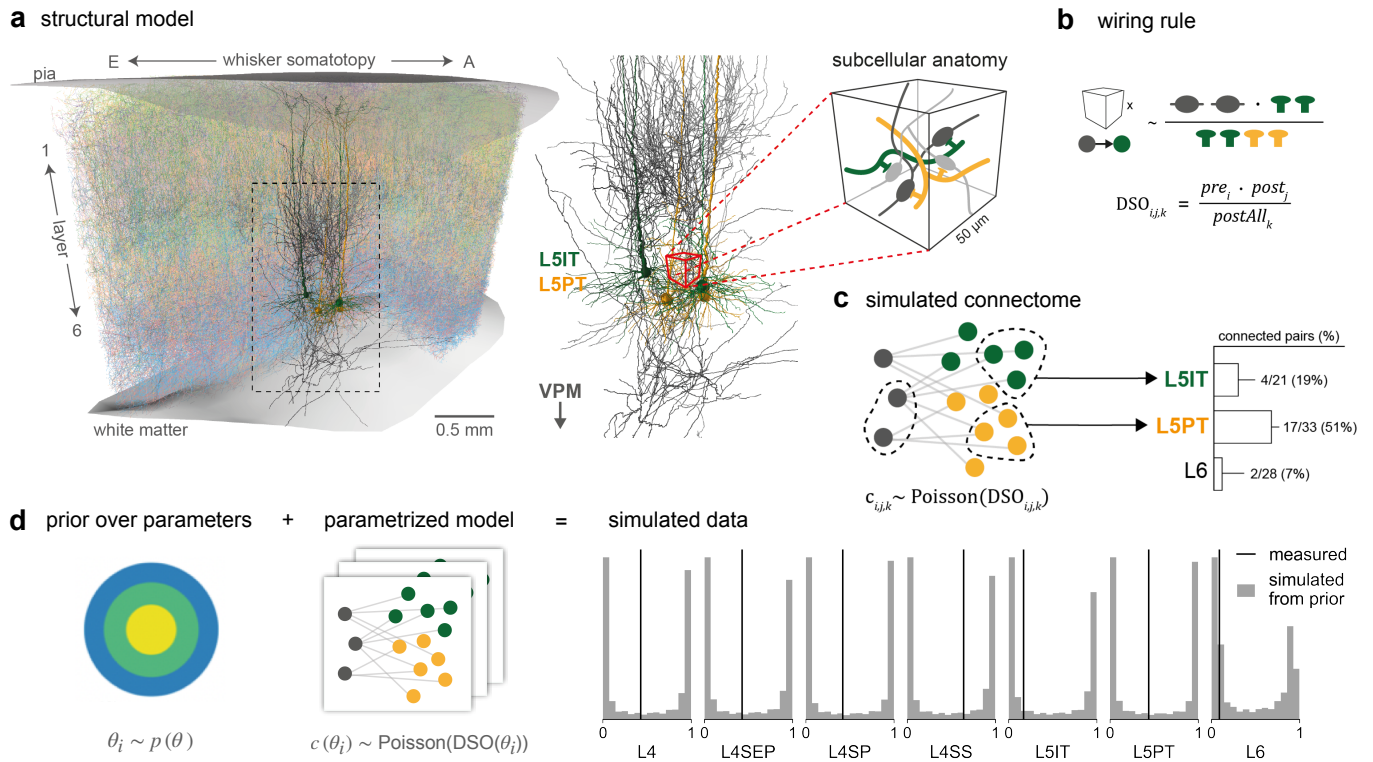


Figure 2. Formulating wiring rules in the rat barrel cortex as simulation-based models. (a) The structural model of the rat barrel cortex contains digital reconstructions of position, morphology, and subcellular features of several neuron types in the barrel cortex and the ventral posterior medial nucleus (VPM) of the thalamus. (b) We formulate a wiring rule that predicts the probability of a synapse between two neurons from their *dense structural overlap* (DSO), i.e., the product of the number of *pre*- and *postsynaptic* structural features, normalized by all postsynaptic features in a given subvolume (*postAll*). (c) By applying the wiring rule to every neuron-pair subvolume combination of the model to connection probabilities and then sampling corresponding synapse counts from a Poisson distribution (left), we can simulate a barrel cortex connectome. To compare the simulated data to measurements, we calculate population connection probabilities between VPM and barrel cortex cell types as they have been measured experimentally (right). (d) To obtain a simulation-based model, we introduce parameters to the rule and define a prior distribution over parameters (left) such that each parameter combination corresponds to a different rule. Simulating data with random parameters from the prior covers the entire range of probabilities (right, gray), including the measured data (black, *Bruno and Sakmann, 2006; Constantinople and Bruno, 2013*).

150 These anatomical features were collected for several neuron types in the barrel cortex and pro-
 151 jecting neurons from the ventral posterior medial nucleus (VPM) of the thalamus and arranged
 152 in 3D model (Fig. 2a, see *Methods & Materials* for details). Thus, by applying a wiring rule that
 153 predicts the connectivity of each neuron pair in the model from the structural features, one can
 154 construct a simulated connectome of the entire barrel cortex (*Egger et al., 2014*). *Udvary et al.*
 155 (*2022*) proposed a parameter-free wiring rule acting solely on structural features of the model, e.g.,
 156 the pre-synaptic boutons and postsynaptic dendritic spines. Here, we extended this wiring rule
 157 to a parameterized version that allows systematic analysis of how such structural features could
 158 interact and be predictive of connectivity.

159 A wiring rule for the rat barrel cortex

160 The parameter-free wiring rule introduced by *Udvary et al. (2022)* proposes that the probability of
 161 two neurons forming a synapse is proportional to a quantity called *dense structural overlap* (DSO).
 162 The DSO is defined as the product of the number of presynaptic boutons and postsynaptic contact
 163 sites (e.g., dendritic spines), normalized by the number of all postsynaptic targets in the neigh-
 164 borhood (denoted as pre_i , $post_j$, and $postAll_k$ for each neuron i , neuron j and subvolume k in the
 165 model, Fig. 2b):

$$DSO_{i,j,k} = \frac{pre_i \cdot post_j}{postAll_k}. \quad (1)$$

166 To simulate a connectome corresponding to the DSO wiring rule, one applies the rule to every
 167 subvolume-neuron-pair combination of the structural model and then samples synapse counts
 168 from a Poisson distribution using the calculated synapse probabilities (Fig. 2c, left, see [Methods &](#)
 169 [Materials](#) for details). The resulting simulated connectome can then be used to calculate summary
 170 statistics in the format recorded in experiments, e.g., *in vitro* or *in vivo* paired recordings or dense
 171 reconstructions at electron microscopic levels (Fig. 2c, right).

172 **Udvary et al. (2022)** showed that the DSO wiring rule can reproduce measured network charac-
 173 teristics at different scales. However, in its current form, the DSO rule assumes that the pre- and
 174 postsynaptic features have the same relative weight in determining the probability of a synapse
 175 and that these weights are the same across all barrel cortex cell types. Is this specific combination
 176 of pre- and postsynaptic features in the DSO rule is the only valid choice? A common approach to
 177 testing this question would be to iteratively modify the rule, e.g., by adding a scaling factor to the
 178 postsynaptic features or introducing different scaling factors for every cell type. However, this ap-
 179 proach can be inefficient because any changes to the rule would require rerunning the procedure
 180 of generating simulated data and manually comparing it to measured data.

181 Defining a wiring rule simulator

182 In order to test different variations of the DSO rule efficiently, we introduced three parameters to
 183 the DSO rule: θ_{pre} for scaling the presynaptic bouton counts, θ_{post} for scaling the postsynaptic target
 184 density, and $\theta_{postAll}$ for scaling the normalizing feature (Fig. 2d, left). The parametrized DSO rule for
 185 a presynaptic neuron i and postsynaptic neuron j positioned in a subvolume k is then given by

$$DSO_{i,j,k}(\theta) = \frac{pre_i^{\theta_{pre}} \cdot post_j^{\theta_{post}}}{postAll_k^{\theta_{postAll}}}, \quad (2)$$

186 (see [Methods & Materials](#) for details). The three parameters represent the relative weight with
 187 which each local subcellular feature contributes to forming connections.

188 The next step towards applying SBI is selecting measured data x_o to constrain the rule param-
 189 eters. We selected seven connection probabilities of neuronal populations mapping from the ven-
 190 tral posterior medial nucleus (VPM) of the thalamus to different layers and cell types in the barrel
 191 cortex, as proposed by **Udvary et al. (2022)**: layer four (L4), layer four septum (L4SEP), layer four
 192 star-pyramidal cells (L4SP), and layer four spiny stellate cells (L4SS) (**Bruno and Sakmann, 2006**),
 193 layer five slender-tufted intratelencephalic cells (L5IT), layer five thick-tufted pyramidal tract cells
 194 (L5PT), and layer six (**Constantinople and Bruno, 2013**).

195 Overall, one simulation of the wiring rule consisted of three steps: First, applying the rule with a
 196 given set of parameters to the structural features of every combination of neuron-pair-subvolume
 197 to obtain connection probabilities; second, sampling synapses from the Poisson distribution given
 198 the probabilities; and third, calculating the summary statistics matching the measured VPM-barrel
 199 cortex population connection probabilities (Fig. 2a-d; see [Methods & Materials](#) for details). As prior
 200 over the parameters $p(\theta)$, we selected a Gaussian distribution such that sampling random paramete-
 201 rer values from the prior resulted in simulated population connection probabilities that covered a
 202 broad range of possible values, including the measured values (Fig. 2d, right). This set of sampled
 203 model parameter values and their corresponding simulated connection probabilities constituted
 204 the training dataset for running SBI.

205 SBI performs accurately on simulated data

206 Before applying SBI to infer the parameters of the DSO rule given measured data, we validated its
 207 accuracy. As a first step, we considered a variant of the DSO rule simulator for which it was possi-
 208 ble to obtain a ground-truth reference posterior distribution (see section [Methods & Materials](#) for

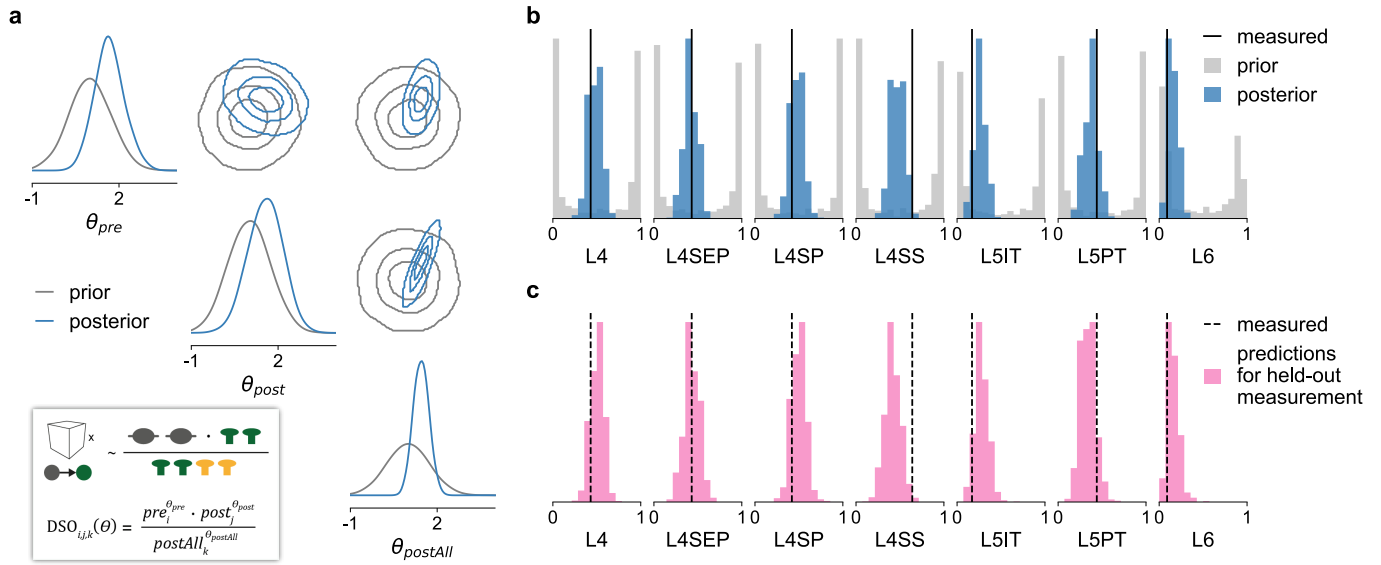


Figure 3. SBI posterior reveals parameter interactions and predicts unseen data. (a) The posterior over the three wiring rule parameters scaling the DSO features (see inset) inferred with SBI (blue) and the initial prior distribution over parameters (gray). The corner plot shows the one-dimensional marginal distribution of each parameter on the diagonal and the pairwise two-dimensional marginals on the off-diagonal (contour lines show the 34%, 68% and 95% credible regions). (b) Comparison of measured connection probabilities (black, *Bruno and Sakmann, 2006; Constantinople and Bruno, 2013*) with those simulated with parameter values sampled from the inferred posterior (blue) versus from the prior (gray). (c) Each panel shows the predictions for one held-out measurement generated from a posterior that was trained and conditioned only on the other six measurements, i.e., each panel refers to a different posterior.

209 details). Using this reference solution, we checked whether SBI infers the posterior accurately and
 210 how many training simulations it requires. We compared three SBI algorithms: Sequential Neural
 211 Posterior Estimation (SNPE, *Papamakarios and Murray, 2016; Lueckmann et al., 2017; Greenberg*
 212 *et al., 2019*), which performs SBI sequentially over multiple rounds focusing inference on one par-
 213 ticular observation; its non-sequential variant NPE; and a classical rejection-sampling-based ap-
 214 proach called Sequential Monte Carlo (SMC, *Sisson et al., 2007; Beaumont et al., 2009*). We found
 215 that all three methods can accurately infer the reference posterior distribution (Suppl. Fig. S1a,b)
 216 but that they differ in terms of simulation efficiency: SNPE was slightly more efficient than NPE,
 217 and both were substantially more efficient than SMC (Suppl. Fig. S1b). As a second step, we per-
 218 formed two checks to validate SBI on the realistic version of the simulator for which no reference
 219 solution was available. First, we used simulated-based calibration (SBC, *Talts et al., 2020*) to check
 220 whether the variances of posterior distributions inferred with SBI were well-calibrated, i.e., nei-
 221 ther too narrow (overconfident) nor too wide (conservative). We found that SNPE and NPE run with *sim-*
 222 *ulated* observed data inferred well-calibrated posteriors for all three parameters (Suppl. Fig. S1c).
 223 Second, we checked the predictive performance of SBI by generating simulated data using param-
 224 eter values sampled from the inferred posterior. We found that the predicted data resembles the
 225 (simulated) observed data (Suppl. Fig. S1d, see *Methods & Materials* for details).

226 SBI identifies many possible wiring rules and reveals parameter interactions

227 After evaluating SBI with simulated data, we applied it to infer the posterior over DSO rule param-
 228 eters given the seven measured VPM-barrel cortex connection probabilities (*Bruno and Sakmann,*
 229 *2006; Constantinople and Bruno, 2013*). Our analysis of the inferred posterior distribution revealed
 230 three key insights.

231 The posterior identifies many data-compatible wiring rule configurations.
232 We found that the inferred posterior distribution was relatively broad, suggesting many param-
233 eter combinations with a high probability of explaining the measured data (Fig. 3a): The one-
234 dimensional posterior marginal distributions of the three parameters (Fig. 3a, blue in diagonal sub-
235 plots) showed maxima at parameter values of $\theta_{pre} = 1.57$, $\theta_{post} = 1.51$ and $\theta_{postAll} = 1.38$, and marginal
236 variances of $\hat{\sigma}_{pre}^2 = 0.26$, $\hat{\sigma}_{post}^2 = 0.35$ and $\hat{\sigma}_{postAll}^2 = 0.07$. These values indicated that $\theta = [1.57, 1.51, 1.38]$
237 was the most likely weighting of the DSO rule features (see Fig. 3a, inset) but that there are also
238 several other parameter combinations with a high posterior probability. For example, sampling
239 parameter values from the posterior for θ_{post} would return values lying mostly in an interval as
240 broad as $[0.33, 2.69]$ (95% posterior credible interval). Despite this relatively broad range of plausi-
241 ble parameters values, we still found that simulating these parameters with the DSO rule resulted
242 in connection probabilities close to the measured data and substantially different from those sim-
243 ulated with the prior (Fig. 3b, blue versus gray). How can so many parameter configurations from
244 such broad ranges all result in similar data?

245 Posterior analysis reveals biologically plausible parameter interactions.

246 Having access to the full posterior distribution and its covariance structure allowed us to answer
247 this question. Inspection of the two-dimensional marginals of each parameter pair indicated a
248 correlation structure substantially different from the uncorrelated prior distribution (Fig. 3a, off-
249 diagonal subplots). To quantify this, we estimated the Pearson correlation coefficients of 10,000
250 parameter values sampled from the posterior distribution. We found a negative correlation be-
251 tween θ_{pre} and θ_{post} (Pearson correlation coefficient $\rho = -0.23$) and positive correlations between
252 $\theta_{postAll}$ and θ_{pre} as well as θ_{post} ($\rho = 0.35$ and $\rho = 0.81$, respectively). These correlations are plausible
253 given the design of the DSO rule (see equation 2). For example, the negative correlation between
254 θ_{pre} and θ_{post} indicated that when increasing the value of θ_{pre} , we would have to decrease θ_{post}
255 in order to obtain the same overall number of connections for a particular cell type. This suggests
256 that a stronger influence of presynaptic boutons on the connection probability requires a weaker
257 influence of postsynaptic target targets on the connection probability.

258 The correlations further suggested that all three structural features are relevant in predicting
259 the connection probabilities: Once one parameter is fixed, the values of the other parameters are
260 strongly constrained. Having access to the full posterior distribution allowed us to quantify this
261 by calculating the *conditional* correlations between the parameters. We obtained the conditional
262 correlations by conditioning the posterior on one parameter dimension—i.e., holding it at a fixed
263 value—and calculating the correlation between samples drawn from the resulting two-dimensional
264 conditional posterior, once for each of the three parameters θ_{pre} , θ_{post} and $\theta_{postAll}$ (see Methods &
265 Materials for details). The resulting correlation coefficients were substantially higher than without
266 conditioning: -0.99 between θ_{pre} and θ_{post} and 0.99 between the other two parameter combinations
267 (see Suppl. Fig. S3 for a visualization of the conditional posteriors). This result confirmed that while
268 the overall range of data-compatible wiring rule parameters is relatively large (Fig. 3a), once one
269 parameter is fixed, the other two are constrained to a very small range of values. Furthermore,
270 the strong conditional posterior correlations indicated that the DSO rule with three parameters is
271 overparametrized, i.e., a parametrization of the DSO rule with only two parameters likely suffices to
272 explain the measured data (see Supplementary material for details). Collectively, the inferred pos-
273 terior suggested that the number of presynaptic boutons and the number of postsynaptic contact
274 sites (and, by extension, axonal and dendritic path length) are sensitive and strongly interdepend-
275 ent structural features for predicting synaptic connectivity.

276 SBI posterior predicts unobserved connection probabilities.

277 To demonstrate the utility of SBI-enabled generative models as a tool for hypothesis generation,
278 we investigated how one can make predictions on unobserved data. Above, we used SBI to con-
279 strain the wiring rule parameters with only the seven measured connection probabilities. However,

280 in principle, the structural model provides access to the entire (simulated) connectome of one bar-
281 rel cortex column. Thus, it allows us to make predictions about other features of the connectome
282 that were not measured yet. To test this approach, we repeated the SBI training procedure seven
283 times, holding out each connectivity measurement once from the training data set, i.e., we trained
284 the posterior estimator on pairs of parameters and data, (θ, x) , where x has six entries instead
285 of seven. After training, we obtained seven different posteriors, each conditioned on six of the
286 seven measured connection probabilities. We then sampled parameter values from every poste-
287 rior, simulated the corresponding barrel cortex connectomes, and calculated *all seven* connection
288 probabilities.

289 The predictions for held-out measurements clustered around the actual measurement values
290 for most of the seven connection probabilities (Fig 3c) and closely resembled the predictive distri-
291 butions of the posterior inferred given all measurements (Fig 3b). Quantitatively, we found that
292 the predictions were within one standard deviation of the measurements (given the sample size
293 used in the experiments, see *Methods & Materials* for details). A classifier trained to distinguish
294 between the predictions of the posterior inferred from all measurements and predictions for held-
295 out measurements achieved an accuracy of 0.68 for L4SS, 0.58 for L5PT, and < 0.55 accuracy for
296 all other measurements (0.5 being the chance level). This cross-validation approach indicated that
297 the structural model paired with the SBI-enabled wiring rule enables us to make experimentally
298 testable predictions. For example, one could predict connection probabilities of cell types different
299 from the seven measured here or other connectivity features of the rat barrel cortex available in
300 the structural model (see below).

301 **Using SBI to rule out invalid wiring hypotheses**

302 Above, we demonstrated that SBI provides a quantitative way to identify valid wiring rule configu-
303 rations from a large set of hypothesized wiring rules. SBI can also be used to rule out invalid hy-
304 potheses, e.g., to show that an existing hypothesis does not agree with empirical data. One such
305 debated hypothesis in connectomics is the so-called *Peters' rule* (*Peters and Feldman, 1976; Brait-*
306 *enberg and Schüz, 1991*). According to this hypothesis, neurons form connections whenever their
307 axons and dendrites are in close proximity, i.e., Peters' rule can be formulated as “axo-dendritic
308 proximity predicts connectivity” (*Udvary et al., 2022*). However, several empirical and theoretical
309 approaches found substantial evidence against Peters' rule (e.g., *Mishchenko et al., 2010; Kasthuri*
310 *et al., 2015; Rees et al., 2017; Udvary et al., 2022*).

311 Here, we show that SBI provides an alternative, quantitative way to discard this hypothesis for
312 the rat barrel cortex. We formulated two wiring rules that implement the proximity hypothesis in
313 the structural model of the barrel cortex at two spatial scales: one predicting connections on the
314 neuron-to-neuron level (Fig. 4) and one predicting synapse counts at the subcellular level (Fig. 5).
315 Both wiring rules have one free parameter, i.e., they incorporate many different proximity hypothe-
316 ses, but there is one particular parameter value corresponding to Peters' rule. We used SBI to infer
317 the posterior distribution over the rule parameters given the seven measured VPM-barrel cortex
318 connection probabilities. Subsequently, we compared inferred parameter values and their predic-
319 tions with those corresponding to Peters' rule.

320 Neuron-level rule

321 At the neuron-to-neuron level, we defined the proximity of two neurons as the number of subvol-
322 umes v they share in the structural model and introduced a threshold parameter acting on the
323 proximity: Neurons i and j form a connection $c_{i,j}$ if the number of subvolumes v_{ij} that contain
324 presynaptic structures of neuron i and postsynaptic structures of neuron j , exceeds a threshold
325 parameter θ_{thres} :

$$c_{i,j}(\theta_{thres}) = 1 \text{ if } v_{ij} > \theta_{thres} \text{ else } 0. \quad (3)$$

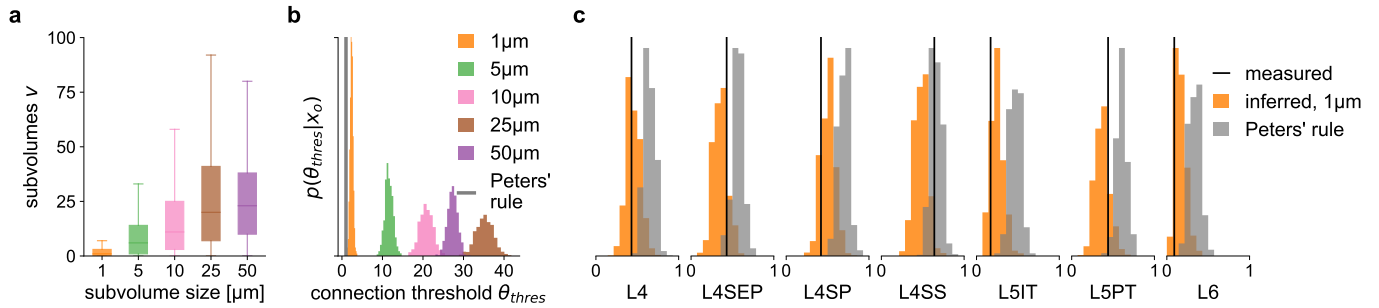


Figure 4. Neuron-level wiring rule inferred with SBI differs from Peters' rule. We used SBI to infer a proximity-based wiring rule at different spatial resolutions of the rat barrel cortex model and compared its predictions to that of Peters' rule. **(a)** Distributions of the shared subvolumes v between neurons in the barrel cortex model for each spatial resolution (subvolume edge length, see legend in (b)). **(b)** SBI posteriors inferred over the connection threshold parameter of the wiring rule (θ_{thres} , number of shared subvolumes required to form a connection), shown for each spatial resolution (colors), and for Peters' rule assuming $\theta_{thres} = 1$ (gray). **(c)** Connection probabilities simulated from the inferred posterior (orange) and Peters' rule (gray) compared to the measured connection probabilities (black).

326 To compare the resulting pair-wise connections between neurons in the barrel cortex model to the
 327 measured connection probabilities, we mapped them to the corresponding population connection
 328 probabilities as described above (see section *Methods & Materials* for details).

329 The structural feature used in this rule is the number of shared subvolumes between two neu-
 330 rons (in contrast to the *subcellular* features used in the DSO rule above). This feature directly
 331 depends on the spatial resolution of the structural model, i.e., the edge length of the subvolume
 332 used to construct the model. Therefore, we calculated the structural features at five different spa-
 333 tial resolutions: 50, 25, 10, 5, and 1 μm . We found that the overall number of shared subvolumes among
 334 neurons increased with edge length, reflecting the increase in subvolume size (Fig. 4a).

335 When applying Peters' rule to the barrel cortex model at the neuron level, a connection oc-
 336 curs whenever two neurons share at least one subvolume, i.e., the connection threshold would be
 337 $\theta_{thres} = 1$. Does this assumption hold for the barrel cortex at the neuron-to-neuron level as well? To
 338 answer this question quantitatively, we used SBI to infer the threshold parameter θ_{thres} of the neu-
 339 ron level rule for each edge length. We observed that the inferred threshold parameters shifted
 340 to larger values with increasing edge length, e.g., the higher spatial resolution, the fewer common
 341 subvolumes were required to obtain a connection (Fig. 4b). This was in line with our observation
 342 that the overall number of shared subvolumes available in the structural model increased with in-
 343 creasing edge length (Fig. 4a). However, irrespective of the spatial resolution, all inferred threshold
 344 parameters were substantially larger than the $\theta_{thres} = 1$ of Peters' rule, reaching from $\theta_{thres} \approx 3$ (pos-
 345 terior mean) for the 1 μm -subvolume model (Fig. 4b, orange) to $\theta_{thres} \approx 28$ for the 50 μm -subvolume
 346 model (Fig. 4b, violet). Accordingly, the comparison of the predictive performance of the inferred
 347 rule and Peters' rule showed that only the data simulated from the inferred rule centered around
 348 the measured data (Fig. 4c, orange and gray, respectively).

349 Synapse-level rule

350 We repeated this test of Peters' rule at the subcellular level as well. Here, we defined a probabilistic
 351 rule: Whenever a presynaptic structure of neuron i and a postsynaptic structure of neuron j are
 352 present within the same subvolume k , they form a synapse with probability θ_{prob} :

$$c_{i,j,k}(\theta_{prob}) \sim \text{Bernoulli}(\theta_{prob}) \text{ if axon of } i \text{ and dendrite of } j \text{ are present in } k. \quad (4)$$

353 This rule predicts synapses for every neuron-pair-subvolume combination using the structural
 354 model with subvolumes of 1 μm edge length. To compare the simulated synapse counts to the
 355 measured connection probabilities, we calculated simulated connection probabilities as described
 356 above. The posterior distribution over the connection probability parameter θ_{prob} inferred with SBI

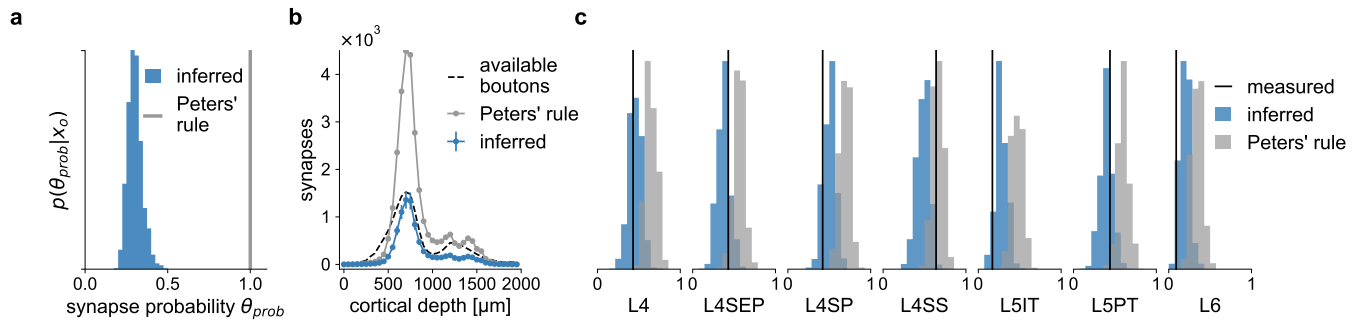


Figure 5. Synapse-level wiring rule inferred with SBI differs from Peters' rule. We compared an SBI-inferred parametrized wiring rule predicting synapse counts on the subcellular level with a corresponding formulation of Peters' rule. **(a)** SBI posterior for the wiring rule parameter θ (probability of forming a synapse if two neurons are close), compared to Peters' rule assuming $\theta = 1$ (gray). **(b)** Number of synapses predicted by the inferred posterior (blue) and Peters' rule (gray) compared to the number of presynaptic boutons realistically available in the structural model (dashed black), plotted over the entire cortical depth of the barrel cortex column. **(c)** Connection probabilities simulated from the inferred synapse level posterior (blue) and Peters' rule (gray) compared to the measured connection probabilities (black).

357 centered around $\theta_{prob} \approx 0.3$ (Fig. 5a). This result suggests that in only thirty percent of the loca-
 358 tions where axon and dendrite are close to each other (shared $1 \mu\text{m}$ subvolume), the rule predicts
 359 a synapse, which is substantially lower than the value of $\theta_{prob} = 1$ corresponding to Peters' rule.
 360 Accordingly, simulating parameter values sampled from the posterior resulted in connection prob-
 361 abilities closer to the measured ones than those predicted by Peters' rule (Fig. 5c, blue vs. gray).

362 For another comparison of Peters' rule with the inferred wiring rule at the synapse level, we
 363 leveraged the predictive properties of the structural model and the SBI posterior. In particular,
 364 the structural model provides access to estimates of the number of biologically available boutons
 365 across the cortical depth of the barrel cortex column (Udvary et al., 2022). The SBI posterior al-
 366 lowed us to simulate data according to the inferred wiring rule parameters. Thus, it was possible
 367 to compare the estimate of the number of empirically available boutons of each presynaptic VPM
 368 neuron with the number of simulated synapses from the inferred wiring rule and Peters' rule. We
 369 found that the inferred rule predicted synapses close to or below the total number of available
 370 boutons (Fig. 5b), in contrast to Peters' rule, which predicted more synapses than biologically plau-
 371 sible.

372 Our results demonstrate how SBI can be applied to different wiring rules to quantitatively rule
 373 out a specific invalid hypothesis: One incorporates the hypothesis into a parametrized model and
 374 compares the SBI-inferred parameters to those corresponding to the hypothesis. In the case of Pe-
 375 ters' rule, the inferred posteriors showed that axo-dendritic proximity alone cannot predict connec-
 376 tivity observed empirically in the rat barrel cortex—it consistently predicts too many connections
 377 (Fig. 4c,d). This finding is in line with previous results showing that the number of dendrites and
 378 axons close to each other exceeds the number of synapses by 1-2 orders of magnitude (Udvary
 379 et al., 2022). Thus, we can conclude that to explain connectivity in the rat barrel cortex, wiring rules
 380 cannot be based solely on axo-dendritic proximity. They also have to take into account subcellular
 381 features like pre- and post-synaptic structures along axons and dendrites.

382 Discussion

383 What principles are behind the complex connectivity patterns of neural networks that shape brain
 384 function? Connectomics aims to answer this question by acquiring detailed data about the struc-
 385 tural and functional composition of the brain. Over the last few years, the development of new
 386 computational approaches for analyzing the resulting large amounts of data and testing the de-
 387 rived hypotheses gained momentum (Triesch and Hilgetag, 2016; Peyser et al., 2019). One compu-
 388 tational approach is to leverage generative models for testing hypotheses about the connectome,
 389 e.g., to implement a hypothesized wiring rule in a computational model and ask whether model

390 simulations can reproduce the measured connectivity patterns of a specific brain region (*Váša and*
391 *Mišić, 2022*). However, identifying the free parameters in the wiring rule that reproduce measured
392 connectivity data can be challenging.

393 We introduced a method that renders the generative modeling approach to connectomics more
394 efficient, enabling us to systematically infer *all* data-compatible parameters of a given compu-
395 tational model. Instead of manually refining a specific generative model to match the data, we
396 equipped it with parameters such that it represents several candidate hypotheses. We then used
397 Bayesian parameter inference to infer the posterior distribution over model parameters condi-
398 tioned on the measured data. The inferred distribution represents all candidate parameter con-
399 figurations, i.e., hypotheses, capable of reproducing the measured data. By relying on simulation-
400 based inference (SBI) methods that do not require access to the likelihood function of the model,
401 we were able to apply our approach to the simulation-based generative models commonly used
402 in computational connectomics.

403 To demonstrate the utility of this approach, we employed it to constrain several wiring rules—
404 at different spatial scales—with connectivity measurements from the rat barrel cortex. We first
405 showed that the inference method is accurate in a scenario with a ground-truth reference solution.
406 Next, in the realistic setting with measured connectivity data, we retrieved many different wiring
407 rule configurations that could explain the measured data equally well. Analyzing the geometrical
408 structure of the inferred posterior distribution revealed strong correlations between wiring rule
409 parameters that are in line with their biological interpretations. Importantly, we were able to accu-
410 rately predict held-out connectivity measurements, demonstrating the method's utility in making
411 experimentally testable predictions. Finally, we used our approach to quantitatively show that a
412 wiring rule based solely on axo-dendritic proximity cannot explain barrel cortex connectivity mea-
413 surements. Overall, these results demonstrate the potential benefits of the Bayesian inference
414 approach, i.e., having access to the full posterior distribution over model parameters rather than
415 manually optimizing for individual parameters one hypothesis at a time and the flexibility of SBI in
416 requiring only simulated data to perform inference.

417 **Related work**

418 The problem of identifying parameters of computational models that reproduce experimentally
419 observed data has been addressed in computational connectomics before. For example, *Vértes*
420 *et al. (2012)* built a model of functional connections between brain regions and used optimiza-
421 tion methods to find single best-fitting parameters capturing the functional MRI data measured in
422 humans. *Klimm et al. (2014)* and *Betz et al. (2016)* used Monte Carlo sampling methods for op-
423 timizing the parameters of synthetic networks of structural connectivity to match the topological
424 properties of human connectomes recorded with MRI. In contrast to our approach, these studies
425 do not perform Bayesian inference but rely on optimization techniques that identify single best-
426 fitting solutions, potentially ignoring other parameters that fit the data equally well.

427 While there have been Bayesian approaches to computational connectomics, they differed
428 from the approach we proposed here. *Jonas and Kording (2015)* built a probabilistic model of
429 cell type-dependent connectivity in the mouse retina and proposed a non-parametric Bayesian al-
430 gorithm that automatically predicts cell types and microcircuitry from connectomics data. *Klinger*
431 *et al. (2021)* performed Bayesian model comparison using a rejection-sampling approach (*Toni*
432 *et al., 2009*) to compare a set of competing local circuit models in layer 4 of the mouse primary
433 somatosensory cortex based on purely structural connectomics data. In contrast to our approach,
434 they inferred the probabilities of several models whose parameters are fixed (i.e., model com-
435 parison) and did not infer the parameter of individual models. Moreover, their approach relied on
436 classical rejection-sampling techniques, which are less simulation-efficient compared to the neural-
437 network-based SBI we employed and will likely not scale to higher-dimensional inference problems
438 (Figure S1b; for a detailed comparison, see *Lueckmann et al., 2021*). However, combining both ap-
439 proaches, e.g., using more efficient neural-network-based model comparison techniques (*Boelts*

440 *et al., 2019; Radev et al., 2021*) followed by parameter inference with SBI, would be a promising
441 direction for future research.

442 Aside from the above examples, computational models in connectomics are often implemented
443 as complicated computer simulations that can generate simulated data but for which the underlying
444 likelihood functions are not accessible, thus limiting our ability to perform Bayesian inference.
445 To account for this limitation, we employed simulation-based inference (SBI, *Cranmer et al., 2020*)
446 methods which only require simulations from the model to perform Bayesian inference. SBI has
447 been applied previously in various fields, ranging from genomics (*Bernstein et al., 2021*), evolution-
448 ary biology (*Ratmann et al., 2007; Vecilla et al., 2022*), computational and cognitive neuroscience
449 (*Gonçalves et al., 2020; Oesterle et al., 2020; Deistler et al., 2022b; Groschner et al., 2022; Sab-
450 bagh et al., 2020; Hashemi et al., 2022*), to robotics (*Marlier et al., 2021*), global health (*de Witt
451 et al., 2020*) and astrophysics (*Alsing et al., 2018; Dax et al., 2021*). For the wiring rule exam-
452 ples presented here, we used sequential neural posterior estimation (SNPE, *Papamakarios and
453 Murray, 2016; Lueckmann et al., 2017; Greenberg et al., 2019*), which performs neural-network-
454 based conditional density estimation to estimate the posterior distribution from simulated data.
455 Neural-network-based SBI approaches build on recent advances in probabilistic machine learning
456 (*Papamakarios and Murray, 2016; Le et al., 2017; Papamakarios et al., 2021*) and exhibit several
457 advantages compared to more classical rejection-sampling-based techniques, commonly termed
458 Approximate Bayesian Computation (ABC, *Sisson et al., 2018*). For example, in contrast to ABC
459 approaches, they do not require selecting a criterion for quantitatively comparing simulated with
460 measured data. Furthermore, they can leverage the ability of neural networks to exploit continu-
461 ities in the parameter space or automatically learn informative summary features from raw data.
462 As a consequence, they are often more simulation-efficient (Suppl. Fig. S1b) and scale better to
463 problems with more model parameters and high-dimensional data (*Lueckmann et al., 2021*).

464 **Applicability and limitations**

465 In this work, we applied SBI to the specific problem of inferring wiring rules that can generate ob-
466 served connectivity data. A key advantage of SBI is that only data simulated from the model are
467 required to infer the posterior distribution over model parameters. Thus, it is generally applicable
468 to any computational model capable of simulating data from a set of parameters. To give spe-
469 cific examples, one could make the algorithms for generating synthetic functional connectomes
470 proposed by *Vértes et al. (2012)*, *Klimm et al. (2014)*, or *Betzel et al. (2016)* amenable to SBI by
471 introducing parameters of interests and then use SBI to efficiently identify all data-compatible pa-
472 rameter values.

473 Posterior-targeting SBI approaches, like the NPE algorithm we used, have the additional ad-
474 vantage that they can obtain the posterior distribution for new data points without retraining the
475 underlying artificial neural networks, i.e., they perform amortized inference (see *Papamakarios
476 and Murray, 2016; Gonçalves et al., 2020*, for details). Another advantage of SBI is that it can
477 leverage the ability of neural networks to automatically learn informative summary features from
478 observations (see *Lueckmann et al., 2017; Chan et al., 2018; Greenberg et al., 2019; Gonçalves
479 et al., 2020; Ramesh et al., 2022*, for examples). While we did not exploit this feature for the low-
480 dimensional measured data in the wiring rule examples presented here, we believe that it will
481 be essential for future applications of SBI in computational connectomics, e.g., when dealing with
482 high-dimensional dense reconstructions of electron-microscopy data (*Shapson-Coe et al., 2021;
483 MICrONS-Consortium et al., 2021; Turner et al., 2022*).

484 SBI's dependency on simulated data and neural network training also entails several limitations:
485 the inferred posterior distributions are only approximations of the unknown actual posterior distri-
486 bution. Therefore, applying SBI requires careful evaluation of every problem at hand. In theory, SBI
487 does recover the unknown posterior distribution when given enough training data (*Papamakarios
488 and Murray, 2016*). In practice, however, the complexity and dimensionality of the posterior dis-
489 tribution determine how many training simulations are required for an accurate approximation

490 of the posterior. Previous studies have successfully applied SBI in scenarios where the simulator
491 has a runtime on the order of seconds and with up to thirty parameters (*Gonçalves et al., 2020*;
492 *Deistler et al., 2022b*; *Ramesh et al., 2022*), but these numbers strongly depend on the problem
493 and available computational resources.

494 Another limitation of SBI is the problem of model misspecification. SBI generally assumes that
495 the generative model is well-specified, i.e., that it can simulate data that is very similar to the mea-
496 sured data. If this is not the case, then the inferred posterior can be substantially biased (*Frazier*
497 *et al., 2019*; *Cannon et al., 2022*). We recommend performing prior predictive checks to detect
498 model misspecification, i.e., generating a large set of simulated data with parameters sampled
499 from the prior distribution and checking whether the measured data lies inside the distribution
500 of simulated data, as demonstrated in the wiring rule example (Fig. 2d, see Supplementary mate-
501 rial for details). Recent methodological work in SBI addresses this problem, e.g., by automatically
502 detecting model misspecification (*Schmitt et al., 2022*) or by explicitly incorporating the model mis-
503 match into the generative model (*Ward et al., 2022*).

504 More generally, applying SBI to new inference problems requires several choices by the prac-
505 titioner, from prior predictive checks and model-checking to selecting suitable neural network ar-
506 chitectures and validating the inferred posterior distribution. As a general guideline, we recom-
507 mend following the steps we performed for the wiring rule example: First, we investigated the ac-
508 curacy of SBI and estimated the required number of training simulations by testing it in a scenario
509 with a known reference solution. Second, we ensured that the inferred posterior distribution has
510 well-calibrated uncertainty estimates using simulation-based calibration (*Talts et al., 2020*). Third,
511 we checked whether the parameter values identified by SBI accurately reproduced the measured
512 data (see *Methods & Materials* for details). Additionally, we recommend guiding hyperparameter
513 choices by resorting to well-tested heuristics and default settings available in open-source soft-
514 ware packages developed and maintained by the community. We performed all our experiments,
515 evaluation steps, and visualization using the *sbi toolkit* (*Tejero-Cantero* et al., 2020*).

516 Conclusion

517 We present SBI as a method for constraining the parameters of generative models in computa-
518 tional connectomics with measured connectivity data. The key idea of our approach is to initially
519 define a probability distribution over many possible model parameters and then use Bayesian pa-
520 rameter inference to identify all those parameter values that reproduce the measured data. We
521 thereby replace the iterative refinement of individual model configurations with the systematic
522 inference of all data-compatible solutions. Our approach will be applicable to many generative
523 modeling scenarios in computational connectomics, providing researchers with a quantitative tool
524 to evaluate and explore hypotheses about the connectome.

525 Methods & Materials

526 Code availability

527 Data and code for reproducing the results are available at [https://github.com/mackelab/sbi-for-](https://github.com/mackelab/sbi-for-connectomics)
528 [connectomics](https://github.com/mackelab/sbi-for-connectomics), including a tutorial on how to apply SBI in computational connectomics in general.
529 For running SBI using SNPE, posterior visualization, and posterior validation, we used the *sbi* pack-
530 age at <https://github.com/mackelab/sbi> (*Tejero-Cantero* et al., 2020*). The benchmarking of SNPE
531 and SMC-ABC methods, including the generation of reference posteriors, was performed using the
532 *sbibm* package at <https://github.com/sbi-benchmark> (*Lueckmann et al., 2021*).

533 Bayesian inference for computational connectomics

534 We introduced Bayesian inference as a tool to identify model parameters of generative models
535 in computational connectomics, given experimentally observed data. Bayesian inference takes a
536 probabilistic view and defines the model parameters and data as random variables. It aims to infer

537 the conditional probability distribution of the model parameters conditioned on the observed data,
538 i.e., the *posterior distribution*. Bayes' rule defines the posterior distribution as

$$p(\theta|x_{\text{obs}}) = \frac{p(x_{\text{obs}}|\theta)p(\theta)}{p(x_{\text{obs}})}, \quad (5)$$

539 where $p(x_{\text{obs}}|\theta)$ is the likelihood of the data given model parameters, $p(\theta)$ is the prior distribu-
540 tion over model parameters and $p(x) = \int_{\theta} p(x|\theta)p(\theta)d\theta$ is the so-called *evidence*. Thus, performing
541 Bayesian inference requires three components:

- 542 1. Experimentally observed data x_{obs} .
- 543 2. A *likelihood* $p(x_{\text{obs}}|\theta)$, which defines the relationship between model parameters and data.
544 In our setting, the likelihood is implicitly defined by the computational model, i.e., by the
545 simulator generating connectomics data x given model parameters θ . The simulator needs
546 to be stochastic, i.e., when repeatedly executed with a fixed parameter θ , it should generate
547 varying data. Technically, given a fixed parameter value θ , the likelihood defines a probability
548 distribution over x , and simulating data corresponds to sampling $x \sim p(x|\theta)$.
- 549 3. A *prior* distribution $p(\theta)$. The prior incorporates prior knowledge about the parameters θ , e.g.,
550 biologically plausible parameter ranges or known parameter correlations.

551 The posterior distribution $p(\theta|x_{\text{obs}})$ inferred through Bayes' rule characterizes all model parameters
552 likely to reproduce the observed data. For example, model parameters with a high probability un-
553 der the posterior distribution will result in data close to the observed data. In contrast, parameters
554 from low probability density regions will likely generate data different from the observed data.

555 In most practical applications, it is hard to obtain an analytical solution to Bayes' rule because
556 the evidence $p(x) = \int p(x|\theta)p(\theta)d\theta$ is challenging to calculate. There exists a large set of methods to
557 perform approximate inference, e.g., Markov Chain Monte Carlo sampling (MCMC, *Rosenbluth and*
558 *Rosenbluth, 1955; Hogg and Foreman-Mackey, 2018*). MCMC methods can be used to obtain sam-
559 ples from the posterior distribution. However, they require evaluation of the likelihood function of
560 the model, and computational models in connectomics are usually defined as scientific simulators
561 for which no analytical form of the underlying likelihood is available or numerical approximations
562 are computationally expensive.

563 Simulation-based inference

564 Simulation-based inference (SBI, *Cranmer et al., 2020*) allows us to perform Bayesian inference
565 without numerical evaluation of the likelihood by requiring only access to simulations from the
566 model. The idea of SBI is to generate a large set of pairs of model parameters and corresponding
567 simulated data and use it as training data for artificial neural networks (ANN). The employed ANNs
568 are designed to approximate complex probability distributions. Thus, they can be used to approxi-
569 mate the likelihood to then obtain posterior samples via MCMC (*Papamakarios et al., 2017; Lueck-*
570 *mann et al., 2019; Hermans et al., 2020; Boelts et al., 2022*) or the posterior distribution directly
571 (*Papamakarios and Murray, 2016; Lueckmann et al., 2017; Greenberg et al., 2019*). Once trained,
572 the neural networks are applied to the experimentally observed data to obtain the approximate
573 posterior. In our work, we used an SBI approach called *sequential neural posterior estimation* (SNPE,
574 *Papamakarios and Murray, 2016; Greenberg et al., 2019*).

575 Neural posterior estimation

576 Neural posterior estimation (NPE) uses an artificial neural network $F(x)$ to learn an approximation
577 of the posterior from training data pairs $\{(\theta_i, x_i)\}_{i=1}^N$, where θ is sampled from a prior $\theta_i \sim p(\theta)$, and
578 x is simulated from the model $x_i \sim \text{simulator}(\theta_i)$. The density estimator $F(x)$ is trained to construct
579 a distribution that directly approximates the *posterior*. It is usually defined as a parametric family
580 q_{ϕ} with parameters ϕ , e.g., a mixture density network (MDN, *Bishop, 1994*), or a normalizing flow
581 (*Papamakarios et al., 2021*). For example, suppose q is a mixture of Gaussians, then F would

582 take the data as input and predict the parameters ϕ , $\phi = F(x)$, where ϕ contains the means, the
 583 covariance matrix, and the mixture weight of each mixture component. $F(x)$ is trained to predict
 584 the parameters ϕ from x by minimizing

$$-\frac{1}{N} \sum_{i=1}^N \log q_{\phi=F(x_i)}(\theta_i | x_o).$$

585 This training loss implicitly minimizes the Kullback-Leibler divergence between the true poste-
 586 rior and the approximation $q_{\phi}(\theta|x)$. It will converge to zero, i.e., NPE will infer the true posterior, in
 587 the limit of infinite training data and given density estimator that is flexible enough (**Papamakari-**
 588 **os and Murray, 2016; Le et al., 2017**). Algorithm 1 summarizes the algorithmic steps of NPE; see
 589 **Greenberg et al. (2019)** for details.

590 Once NPE is trained on simulated data, it can be applied to the actual observed data x_{obs} , e.g.,
 591 $\phi = F(x_{\text{obs}})$, to obtain an approximation to the desired posterior:

$$q_{\phi}(\theta|x_{\text{obs}}) \approx p(\theta|x_{\text{obs}}). \quad (6)$$

592 Importantly, NPE applies to any newly observed data without retraining the density estimator, i.e.,
 593 the inference with NPE is *amortized*. There is also a sequential variant of NPE called SNPE, where
 594 the training is performed over several rounds to focus the density estimator on a specific obser-
 595 vation x_{obs} . In each new round of SNPE, the new training data is not generated with parameters
 596 sampled from the prior but from the posterior estimate of the previous round. While the sequen-
 597 tial approach can be substantially more sampling-efficient compared to NPE, i.e., requiring fewer
 598 training simulations to obtain a good posterior approximation for a given x_{obs} (**Lueckmann et al.,**
 599 **2021**), it comes with two caveats. First, it requires retraining for every new x_{obs} . Second, using a pro-
 600 posal distribution different from the prior for simulating new data requires a correction, resulting in
 601 additional algorithmic choices and challenges. Over the last few years, different approaches have
 602 been proposed to perform this correction (**Papamakarios and Murray, 2016; Lueckmann et al.,**
 603 **2017; Greenberg et al., 2019; Deistler et al., 2022a**). We used SNPE with the correction proposed
 604 by **Greenberg et al. (2019)**.

Algorithm 1: Single round Neural Posterior Estimation as in **Papamakarios and Murray (2016)**

```

input simulator  $p(\mathbf{x}|\theta)$ , prior  $p(\theta)$ , observed data  $x_{\text{obs}}$ 
for  $j = 1 : N$  do
  | Sample  $\theta_i \sim p(\theta)$ 
  | Simulate  $\mathbf{x}_i \sim p(\mathbf{x}|\theta_i)$ 
end
 $\phi \leftarrow \arg \min -1/N \sum_i \log q_{F(\mathbf{x}_i, \phi)}(\theta_i)$ 
Set  $\hat{p}(\theta|x_o) = q_{F(x_{\text{obs}}, \phi)}(\theta)$ 
return Samples from  $\hat{p}(\theta|x_{\text{obs}})$ ; density estimator  $q_{F(\mathbf{x}, \phi)}(\theta)$ 

```

605 **Posterior validation**

606 In theory and with unlimited training data, NPE will converge to the true (unknown) posterior dis-
 607 tribution. However, training data is limited in practice, and the underlying posteriors can be high-
 608 dimensional and complex. Thus, it is essential to validate the approximate posterior. There are
 609 two common techniques for validating SBI even in the absence of a reference posterior: predictive
 610 checks (**Gelman et al., 2020**) and calibration checks, e.g., simulation-based calibration (SBC, **Cook**
 611 **et al., 2006; Talts et al., 2020**).

612 Predictive checks

613 Predictive checks can be applied to either the prior or the posterior. The *prior predictive check* is
614 applied before the inference. It checks whether the model can produce data close to the experi-
615 mentally observed data x_{obs} , i.e., that the distribution obtained by sampling from the prior and sim-
616 ulating the corresponding data contains x_{obs} . If this is not the case, the model or the prior could be
617 misspecified and should be refined before applying SBI. We performed the prior predictive check
618 for the DSO rule simulator by sampling 100,000 parameters from the prior and ensuring that the
619 resulting distribution of simulated connection probabilities covers the seven measured values (see
620 *Prior predictive checks* for details).

621 The posterior predictive check tests the predictive performance of the posterior. It should be
622 applied after the inference by simulating data using parameters sampled from the posterior:

$$x_p \sim \text{simulator}(\theta_p) \text{ where } \theta_p \sim p(\theta|x_{\text{obs}}).$$

623 The simulated data should cluster around the observed data with a variance on the order of the
624 variance expected from the simulator. We performed this check for all inferred wiring rules by sim-
625 ulating 1,000 data points using 1,000 parameters sampled from the corresponding SBI posterior.

626 Simulation-based calibration

627 The variance of the posterior distribution expresses the uncertainty in the parameters. Simulation-
628 based calibration (SBC) provides a way to check whether these uncertainties are, on average, well-
629 calibrated, i.e., that the posterior is (on average) neither too broad (under-confident) nor too nar-
630 row (over-confident). The basic idea of SBC is the following. Suppose one uses an SBI method
631 to obtain $i = 1, \dots, N$ different posteriors $p(\theta|x_i)$ for different observations x_i generated from dif-
632 ferent parameters θ_i sampled from the prior. If one determines the rank of each parameter θ_i
633 among samples from its corresponding posterior $p(\theta|x_i)$, then the posteriors obtained with this
634 SBI method have well-calibrated uncertainties if the collection of all N ranks follows a uniform dis-
635 tribution (*Talts et al., 2020*). To check whether the posterior obtained with SBI is well-calibrated,
636 we repeated the inference with NPE $N = 1000$ times (no retraining required), using data generated
637 from the simulator with parameters sampled from the prior. Subsequently, we performed a vi-
638 sual check for uniformity of the corresponding SBC ranks by comparing their empirical cumulative
639 density function against that of a uniform distribution.

640 A generative structural model of the rat barrel cortex

641 We demonstrated the utility of SBI for computational connectomics by constraining wiring rules in
642 a structural model of the rat barrel cortex with connectivity measurements. To fulfill the prereq-
643 uisites of Bayesian inference defined above, we set up a simulation-based model for simulating
644 wiring rules in the barrel cortex model. The wiring rule simulator has three components:

- 645 1. a *structural model* that provides features (Fig.2a),
- 646 2. a parametrized *wiring rule* that is applied to the features to simulate a connectome (Fig.2b),
- 647 3. calculation of *summary statistics* from the simulated connectome to match the available mea-
648 surements (Fig.2c).

649 The structural model

650 The structural model is a digital reconstruction of the rat barrel cortex constructed from detailed
651 measurements of cell types and their morphologies, locations, and sub-cellular features, includ-
652 ing single boutons and dendrites, obtained from several animals (*Meyer et al., 2010, 2013; Egger*
653 *et al., 2012, 2014; Narayanan et al., 2015; Udvary et al., 2022*). These measurements and their
654 digital reconstructions were copied and arranged according to measured cell type distributions
655 in all cortical layers to obtain a realistic estimate of the structural composition of a large part of

656 the entire barrel cortex. The model contains around 477,000 excitatory and 77,000 inhibitory neu-
 657 rons, resulting in more than 5.5 billion synaptic sites. Furthermore, it incorporates the projections
 658 from the ventral posterior medial nucleus (VPM) of the thalamus to all cortical layers. The model
 659 gives access to several cellular and subcellular structural features, including presynaptic boutons
 660 and postsynaptic target counts (spine densities). These features were collected for every neuron
 661 segment in every subvolume of the model. The model does not contain synaptic connections but
 662 only the structural features. Thus, it allows to simulate the effect of different wiring rules by apply-
 663 ing the rule to the structural features and comparing the resulting connectome to experimental
 664 measurements (Udvary et al., 2022).

665 Dense structural overlap wiring rule

666 We applied a wiring rule to the structural model to turn it into a simulation-based model that can
 667 generate simulated connectomes of the rat barrel cortex. As a wiring rule, we used the dense struc-
 668 tural overlap (DSO) rule introduced by (Egger et al., 2014; Udvary et al., 2022), which proposes that
 669 two neurons form a synapse depending on their locally available structural subcellular features
 670 summarized as DSO. The DSO is the product of the numbers of pre- and postsynaptic structures,
 671 pre and $post$, that a presynaptic neuron i and a postsynaptic neuron j contribute to a subvolume k
 672 relative to the total number of postsynaptic structures contributed by all neurons, $postAll$ (Fig.2b):

$$DSO_{i,j,k} = \frac{pre_i \cdot post_j}{postAll_k}. \quad (7)$$

673 We assumed that the number of connections between any neuron pair (i,j) within a subvolume k
 674 is given by a Poisson distribution with the DSO as the rate parameter (Egger et al., 2014, Fig.2c):

$$c_{ijk} \sim \text{Poisson}(DSO_{i,j,k}).$$

675 The DSO rule is stochastic, i.e., it samples different synapse counts every time it is applied to the
 676 structural model. However, the contribution of each structural feature to the DSO is fixed. To
 677 allow for more flexibility in the relative weighting of each feature, we generalized the DSO rule by
 678 introducing three scaling parameters for the three structural features, θ_{pre} , θ_{post} and $\theta_{postAll}$:

$$DSO_{i,j,k}(\theta) = \frac{pre_i^{\theta_{pre}} \cdot post_j^{\theta_{post}}}{postAll_k^{\theta_{postAll}}}. \quad (8)$$

679 We rewrote the parametrized rule as a Poisson *generalized linear model* (GLM, **Nelder and Wedder-**
 680 **burn, 1972**) by transforming the features to the logarithmic space, stacking them as column vectors
 681 in a feature matrix $X_{ijk} = [\log pre(i, k); \log post(j, k); -\log postAll(k)]$ and arranging the scaling param-
 682 eters in a vector θ :

$$\begin{aligned} DSO_{i,j,k}(\theta) &= \exp \left(\log \left(\frac{pre_i^{\theta_{pre}} \cdot post_j^{\theta_{post}}}{postAll_k^{\theta_{postAll}}} \right) \right) \\ &= \exp \left(\theta_{pre} \log pre_i + \theta_{post} \log post_j - \theta_{postAll} \log postAll_k \right) \\ &= \exp(\theta^T X_{ijk}) \\ c_{ijk}(\theta) &\sim \text{Poisson}(\exp(\theta^T X_{ijk})). \end{aligned}$$

683 The generalized version of the DSO rule takes parameter combination θ and generates simulated
 684 connectomes in the format of synapse counts. Each new parameter setting corresponds to a dif-
 685 ferent variant of the DSO rule that could have generated the measured data. Note that setting
 686 $\theta = [1, 1, 1]^T$ would result in the expression introduced in equations 7 and 8.

687 Mapping from simulated connectomes to measured connectivity data
 688 The generalized DSO rule and the chosen prior distribution jointly define a space of simulated con-
 689 nectomes associated with the DSO rule. The last step for constructing a simulation-based model
 690 is to map the simulated connectomes to the type of data that can be measured empirically. The
 691 measurements available for the barrel cortex are connection probabilities estimated from pair-
 692 wise recordings between neurons in the ventral posterior medial nucleus (VPM) of the thalamus
 693 and different layers and cell types in the cortex: to layer 4 (L4), layer 4 septum (L4SEP), layer 4 star
 694 pyramidal cells (L4SP), and layer 4 spiny stellate cells (L4SS) (all measured by *Bruno and Sakmann, 2006*),
 695 layer 5 slender-tufted intratelencephalic cells (L5IT), layer 5 thick-tufted pyramidal tract cells
 696 (L5PT), and layer 6 (all measured by *Constantinople and Bruno, 2013*).

697 While the simulated connectome provided access to all neuron-pair-subvolume combinations
 698 in the structural model, the measurements were given only as estimated connection probabilities
 699 for different cell types. To calculate these connection probabilities from the simulated connectome,
 700 we identified the pairs from the presynaptic and postsynaptic neuron populations used in the ex-
 701 periments and calculated the connection probabilities from the corresponding simulated synapse
 702 counts. The number of probed neuron pairs was relatively small in the experiments, e.g., around
 703 50 (*Bruno and Sakmann, 2006*). We tried to mimic this experimental setting in the simulation by
 704 selecting a random sample of 50 pairs from the thousands of possible pairs available in the simu-
 705 lated connectome. We then checked how many of those neuron pairs connected with at least one
 706 synapse for each of the seven populations. Algorithm 2 summarizes the steps for calculating the
 707 summary statistics.

Algorithm 2: Wiring rule simulator and summary statistics calculation

input structural model S , wiring rule $R(\theta)$, parameter $\theta \sim p(\theta)$, measured connectivity data

x_{obs}

Simulate:

generate synapse counts for each neuron-pair i, j in each subvolume k :

$c_{ijk} \sim \text{simulator}(S, R, \theta)$

Summarize:

for each population m measured in x_{obs} **do**

Find index set of neuron pairs Π_m belonging to population m

Sample 50 random neuron-pair indices $\pi \sim \Pi_m$

Estimate population connection probability as average over connected pairs:

$$\sum_{(i,j) \in \pi} \frac{\mathbb{1}(i \rightarrow j)}{|\pi|}$$

end

return simulated connectivity data x

708 Overall, this provided us with a setup to perform Bayesian inference as defined above, to con-
 709 strain the parameters of a hypothesized wiring rule in the rat barrel cortex:

- 710 1. observed data x_{obs} given by seven measured connection probabilities between VPM and bar-
 711 rel cortex
- 712 2. a stochastic simulation-based model that generates simulated data x according to the hy-
 713 pothesized DSO rule, given parameters θ
- 714 3. a prior over model parameters θ

715 This setup readily extends to other observed data, other simulation-based models, or priors.

716 Experimental settings

717 Simulation and SBI settings

718 For performing SBI on the DSO rule parameters, we used a Gaussian prior over the three parameters:
719

$$\theta \sim \mathcal{N}(\mu_0 = [1, 1, 1]^\top, \Sigma_0 = 0.5 I). \quad (9)$$

720 We chose the variance of the prior such that the distribution of simulated connection probabilities covered the range $[0, 1]$ densely (see [Supplementary material](#) for details).

722 For the inference on the distance-based wiring rules, we used a uniform prior over the shared subvolume threshold parameter θ_{thres} of neuron-level rule

$$\theta_{thres} \sim \mathcal{U}(0, 100),$$

724 and a Beta distribution prior over the synapse probability parameter θ_{prob} of the synapse level rule

$$\theta_{prob} \sim \text{Beta}(\alpha = 2, \beta = 2).$$

725 The settings for generating training simulations for SBI were the same for all results: We drew
726 1,000,000 parameter values from the prior and simulated the corresponding synapse counts or
727 neuron-level connections, followed by the summary step (except for the simulator used for bench-
728 marking, see below).

729 To perform SBI, we used (S)NPE with Neural Spline Flows (NSF, [Durkan et al., 2019](#)) as den-
730 sity estimator. The NSF hyperparameters were: five transforms, two residual blocks of 50 hidden
731 units each, ReLU non-linearity, and ten spline bins, all as implemented in the public *sbi* toolbox
732 ([Tejero-Cantero* et al., 2020](#)). The training was performed with a training batch size of 1,000, a
733 validation set proportion of 10%, and a convergence criterion of 20 epochs without improvement
734 of validation loss. We used the different versions of NPE or SNPE as follows: For the benchmark of
735 SBI against the MCMC reference posterior, we compared the non-sequential (NPE) and sequential
736 version (SNPE). For evaluating SBI with simulated data, we used the NPE to leverage its ability to
737 perform inference repeatedly for many different observations without retraining as needed for
738 running simulation-based calibration. For the inference on the DSO rule, we used SNPE with ten
739 rounds to focus the posterior on the measured data. For the inference on the distance-based rules,
740 we used NPE.

741 Validating SBI on the wiring rule simulator

742 We performed two validation steps to ensure that SBI performs reliably when inferring wiring rules
743 in the structural model of the rat barrel cortex. First, we set up a simplified version of the DSO
744 rule simulator for which it was possible to obtain a high-quality reference posterior. In particu-
745 lar, we reduced the number of neuron-pair-subvolume combinations from 130 million available
746 in the original structural model to only ten. Additionally, we omitted the summary step, such that
747 running one simulation corresponded to applying the rule to the structural features of the ten
748 neuron-pair-subvolume combinations and sampling synapse counts from the corresponding Pois-
749 son distribution. As a consequence, the likelihood of this simplified simulator was accessible, i.e., it
750 was given by the Poisson distribution, and it was possible to obtain accurate posterior samples us-
751 ing standard approximate Bayesian inference MCMC sampling ([Hogg and Foreman-Mackey, 2018](#)).

752 We implemented this reduced simulator in the SBI benchmarking framework `sbi_bm` ([Lueck-
753 mann et al., 2021](#)) and obtained reference posterior samples via slice sampling MCMC ([Neal,
754 2003](#)) using ten parallel chains and sequential importance reweighting ([Rosenbluth and Rosen-
755 bluth, 1955; Lueckmann et al., 2021](#)), all as implemented in `sbi_bm`. We compared three algorithms:

756 SNPE, which estimates the posterior in multiple rounds focusing on one specific observation; the
757 single-round variant NPE, which works for many observations without retraining; and a classical se-
758 quential rejection-sampling-based algorithm called SMC-ABC (Sisson *et al.*, 2007; Beaumont *et al.*,
759 2009). As a measure of posterior accuracy, we used the classifier-2-sample-test score (C2ST Lopez-
760 Paz and Oquab, 2018), defined by the classification accuracy of an artificial-neural-network clas-
761 sifier trained to distinguish the approximate and reference posterior samples. For running SNPE
762 and NPE, we used the `sbi` toolbox (Tejero-Cantero *et al.*, 2020), and for SMC-ABC, we used the im-
763 plementations provided in `sbi`. Generating the training data for the SBI algorithm by simulation
764 data from the model can be a crucial computational factor. To investigate the simulation efficiency
765 of different SBI algorithms for our inference problem, we performed a quantitative comparison of
766 the number of training simulations and the resulting accuracy of the approximate posterior by re-
767 peating inference with SMC-ABC, NPE, and SNPE for a simulation budget of 1,000; 10,000; 100,000
768 and 1,000,000 simulations.

769 As a second validation step, we applied SBI to the original version of the DSO rule for which
770 no reference posterior was available. For this setting, we tested the validity of NPE applied to
771 simulated observed data. First, we performed simulation-based calibration (SBC) to check the
772 calibration of the posterior uncertainties inferred by NPE. To run SBC, we trained NPE once on
773 1,000,000 simulations and then obtained 1,000 different posteriors $p(\theta|x_i)$ for different observa-
774 tions x_i , where x_i was generated from different parameters θ_i sampled from the prior. We
775 then collected the individual ranks of the underlying parameter θ_i under their posterior and tested
776 whether these ranks were uniformly distributed by visually inspecting their empirical cumulative
777 density functions. Second, we checked whether the parameters identified by the NPE posterior
778 distribution could reproduce the (simulated) observed data. To perform this check, we sampled
779 1,000 parameters from the posterior inferred given a simulated example observation, simulated
780 corresponding connection probabilities using the DSO rule simulator, and compared them to the
781 observed data.

782 Acknowledgment

783 We thank Auguste Schulz, Guy Moss, and Zinovia Stefanidi for their discussions and comments on
784 a preliminary version of the manuscript.

785 This work was supported by the German Research Foundation (DFG; SFB 1233 PN 276693517;
786 SPP 2041; Germany's Excellence Strategy MCoE-EXC number 2064/1 PN 390727645), the German
787 Federal Ministry of Education and Research (BMBF; project ADIMEM, FKZ 01IS18052 A-D; Tübing-
788 en AI Center, FKZ: 01IS18039A), and the European Union's Horizon 2020 research and innovation
789 program under the Marie Skłodowska-Curie grant agreement No. 101030918 (to R.G.).

790 Author contributions

791 J.B. and J.H.M. conceived the methodology; J.B. conducted the experiments, data analysis, and soft-
792 ware development with help from P.H. and F.Y.; D.U. created the rat barrel cortex model; J.B. cre-
793 ated the visualizations with help from P.H., R.G., and J.H.M.; J.B. wrote the paper with help from
794 R.G., J.H.M., and all other authors. J.H.M. provided the principal supervision with help from M.O.,
795 H.H., and D.B.

796 References

- 797 Alsing J, Wandelt B, Feeney S. Massive optimal data compression and density estimation for scalable, likelihood-
798 free inference in cosmology. *Monthly Notices of the Royal Astronomical Society*. 2018; 477(3):2874–2885. doi:
799 .10.1093/mnras/sty819.
- 800 Avecilla G, Chuong JN, Li F, Sherlock G, Gresham D, Ram Y. Neural networks enable efficient and accurate
801 simulation-based inference of evolutionary parameters from adaptation dynamics. *PLoS biology*. 2022;
802 20(5):e3001633. doi: .10.1371/journal.pbio.3001633.

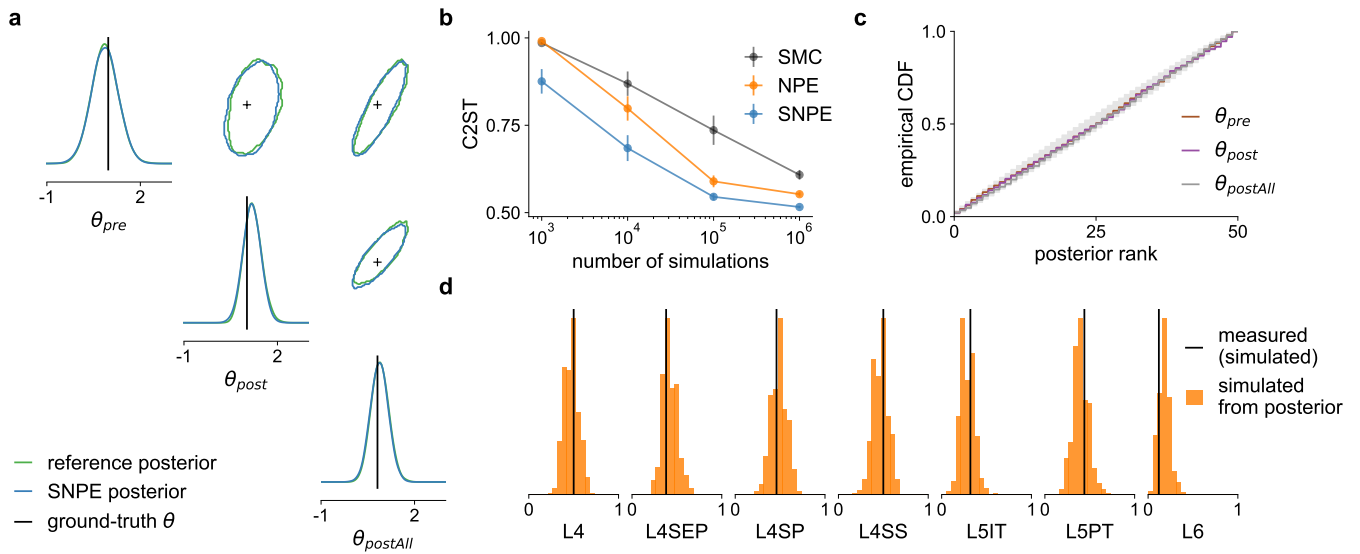
- 803 **Beaumont MA**, Cornuet JM, Marin JM, Robert CP. Adaptive approximate Bayesian computation. *Biometrika*.
804 2009 Dec; 96(4):983–990. doi: .10.1093/biomet/asp052.
- 805 **Bernstein DB**, Sulheim S, Almaas E, Segrè D. Addressing uncertainty in genome-scale metabolic model recon-
806 struction and analysis. *Genome Biology*. 2021 Feb; 22(1):64. doi: .10.1186/s13059-021-02289-z.
- 807 **Betzl RF**, Avena-Koenigsberger A, Goñi J, He Y, de Reus MA, Griffa A, Vértés PE, Mišić B, Thiran JP, Hagmann
808 P, van den Heuvel M, Zuo XN, Bullmore ET, Sporns O. Generative models of the human connectome. *Neu-
809 roImage*. 2016 Jan; 124:1054–1064. doi: .10.1016/j.neuroimage.2015.09.041.
- 810 **Betzl RF**, Bassett DS. Generative models for network neuroscience: prospects and promise. *Journal of The
811 Royal Society Interface*. 2017 Nov; 14(136):20170623. doi: .10.1098/rsif.2017.0623.
- 812 **Billeh YN**, Cai B, Gratiy SL, Dai K, Iyer R, Gouwens NW, Abbasi-Asl R, Jia X, Siegle JH, Olsen SR, et al. Systematic
813 integration of structural and functional data into multi-scale models of mouse primary visual cortex. *Neuron*.
814 2020; 106(3):388–403. doi: .10.1016/j.neuron.2020.01.040.
- 815 **Bishop CM**. Mixture density networks. Aston University; 1994.
- 816 **Boelts J**, Lueckmann JM, Gao R, Macke JH. Flexible and efficient simulation-based inference for models of
817 decision-making. *eLife*. 2022 Jul; 11:e77220. doi: .10.7554/eLife.77220, publisher: eLife Sciences Publications,
818 Ltd.
- 819 **Boelts J**, Lueckmann JM, Goncalves PJ, Sprekeler H, Macke JH. Comparing neural simulations by neural density
820 estimation. In: *2019 Conference on Cognitive Computational Neuroscience Berlin, Germany: Cognitive Compu-
821 tational Neuroscience*; 2019. p. 578–581. doi: .10.32470/CCN.2019.1291-0.
- 822 **Braitenberg V**, Schüz A. Peters' Rule and White's Exceptions. In: Braitenberg V, Schüz A, editors. *Anatomy of
823 the Cortex: Statistics and Geometry Studies of Brain Function*, Berlin, Heidelberg: Springer; 1991.p. 109–112.
824 doi: .10.1007/978-3-662-02728-8_21.
- 825 **Bruno RM**, Sakmann B. Cortex is driven by weak but synchronously active thalamocortical synapses. *Science*.
826 2006; 312(5780):1622–1627. doi: .10.1126/science.1124593.
- 827 **Cannon P**, Ward D, Schmon SM, Investigating the Impact of Model Misspecification in Neural Simulation-based
828 Inference. *arXiv*; 2022. doi: .10.48550/arXiv.2209.01845.
- 829 **Chan J**, Perrone V, Spence J, Jenkins P, Mathieson S, Song Y. A Likelihood-Free Inference Framework for Popula-
830 tion Genetic Data using Exchangeable Neural Networks. In: *Advances in Neural Information Processing Systems*,
831 vol. 31 Curran Associates, Inc.; 2018. doi: .10.1101/267211.
- 832 **Chklovskii DB**, Mel BW, Svoboda K. Cortical rewiring and information storage. *Nature*. 2004 Oct;
833 431(7010):782–788. doi: .10.1038/nature03012.
- 834 **Constantinople CM**, Bruno RM. Deep cortical layers are activated directly by thalamus. *Science*. 2013;
835 340(6140):1591–1594. doi: .10.1126/science.1236425.
- 836 **Cook SR**, Gelman A, Rubin DB. Validation of software for Bayesian models using posterior quantiles. *Journal
837 of Computational and Graphical Statistics*. 2006; 15(3):675–692. doi: .10.1198/106186006X136976.
- 838 **Cranmer K**, Brehmer J, Louppe G. The frontier of simulation-based inference. *Proceedings of the National
839 Academy of Sciences*. 2020; doi: .10.1073/pnas.1912789117.
- 840 **Dax M**, Green SR, Gair J, Macke JH, Buonanno A, Schölkopf B. Real-Time Gravitational Wave Science with Neural
841 Posterior Estimation. *Physical review letters*. 2021; 127(24):241103. doi: .10.1103/physrevlett.127.241103.
- 842 **Deistler M**, Goncalves PJ, Macke JH. Truncated proposals for scalable and hassle-free simulation-based infer-
843 ence. In: *Advances in Neural Information Processing Systems 35 (NeurIPS 2022)* Curran Associates, Inc.; 2022.
844 doi: .10.48550/arXiv.2210.04815.
- 845 **Deistler M**, Macke JH, Gonçalves PJ. Energy-efficient network activity from disparate circuit parameters. *PNAS*.
846 2022 Nov; 119(44):e2207632119. doi: .10.1073/pnas.2207632119.
- 847 **Durkan C**, Bekasov A, Murray I, Papamakarios G. Neural spline flows. In: *Advances in Neural Information Pro-
848 cessing Systems* Curran Associates, Inc.; 2019. p. 7509–7520. doi: .10.48550/arxiv.1906.04032.
- 849 **Egger R**, Dercksen VJ, Udvary D, Hege HC, Oberlaender M. Generation of dense statistical connectomes from
850 sparse morphological data. *Front Neuroanat*. 2014; 8:129. doi: .10.3389/fnana.2014.00129.

- 851 **Egger R**, Narayanan RT, Helmstaedter M, de Kock CPJ, Oberlaender M. 3D Reconstruction and Standardization
852 of the Rat Vibrissal Cortex for Precise Registration of Single Neuron Morphology. *PLoS Computational Biology*.
853 2012; 8(12):e1002837. doi: .10.1371/journal.pcbi.1002837.
- 854 **Frazier DT**, Robert CP, Rousseau J. Model Misspecification in ABC: Consequences and Diagnostics; 2019. <http://arxiv.org/abs/1708.01974>, doi: .10.1111/369-7412/20/82421, arXiv:1708.01974 [math, q-fin, stat].
- 856 **Gelman A**, Vehtari A, Simpson D, Margossian CC, Carpenter B, Yao Y, Kennedy L, Gabry J, Bürkner PC, Modrák
857 M, Bayesian Workflow. arXiv; 2020. doi: .10.48550/arXiv.2011.01808.
- 858 **Gonçalves PJ**, Lueckmann JM, Deistler M, Nonnenmacher M, Öcal K, Bassetto G, Chintaluri C, Podlaski WF, Had-
859 dad SA, Vogels TP, Greenberg DS, Macke JH. Training deep neural density estimators to identify mechanistic
860 models of neural dynamics. *eLife*. 2020 Sep; 9:e56261. doi: .10.7554/eLife.56261.
- 861 **Greenberg D**, Nonnenmacher M, Macke J. Automatic Posterior Transformation for Likelihood-Free Inference.
862 In: *Proceedings of the 36th International Conference on Machine Learning* PMLR; 2019. p. 2404–2414. doi:
863 .10.48550/arxiv.1905.07488.
- 864 **Groschner LN**, Malis JG, Zuidinga B, Borst A. A biophysical account of multiplication by a single neuron. *Nature*.
865 2022 Mar; 603(7899):119–123. doi: .10.1038/s41586-022-04428-3.
- 866 **Hashemi M**, Vattikonda AN, Jha J, Sip V, Woodman MM, Bartolomei F, Jirsa VK, Simulation-Based Inference
867 for Whole-Brain Network Modeling of Epilepsy using Deep Neural Density Estimators. medRxiv; 2022. doi:
868 .10.1101/2022.06.02.22275860.
- 869 **Hermans J**, Begy V, Louppe G. Likelihood-free MCMC with Amortized Approximate Ratio Estimators. In: *Pro-
870 ceedings of the 37th International Conference on Machine Learning* PMLR; 2020. p. 4239–4248.
- 871 **Hogg DW**, Foreman-Mackey D. Data Analysis Recipes: Using Markov Chain Monte Carlo*. *The Astrophysical
872 Journal Supplement Series*. 2018 May; 236(1):11. doi: .10.3847/1538-4365/aab76e.
- 873 **Jain V**, Seung HS, Turaga SC. Machines that learn to segment images: a crucial technology for connectomics.
874 *Current Opinion in Neurobiology*. 2010 Oct; 20(5):653–666. doi: .10.1016/j.conb.2010.07.004.
- 875 **Jonas E**, Kording K. Automatic discovery of cell types and microcircuitry from neural connectomics. *eLife*. 2015
876 Apr; 4:e04250. doi: .10.7554/eLife.04250.
- 877 **Kasthuri N**, Hayworth KJ, Berger DR, Schalek RL, Conchello JA, Knowles-Barley S, Lee D, Vázquez-Reina A, Kaynig
878 V, Jones TR, Roberts M, Morgan JL, Tapia JC, Seung HS, Roncal WG, Vogelstein JT, Burns R, Sussman DL, Priebe
879 CE, Pfister H, et al. Saturated Reconstruction of a Volume of Neocortex. *Cell*. 2015 Jul; 162(3):648–661. doi:
880 .10.1016/j.cell.2015.06.054.
- 881 **Klimm F**, Bassett DS, Carlson JM, Mucha PJ. Resolving Structural Variability in Network Models and the Brain.
882 *PLOS Computational Biology*. 2014 Mar; 10(3):e1003491. doi: .10.1371/journal.pcbi.1003491.
- 883 **Klinger E**, Motta A, Marr C, Theis FJ, Helmstaedter M. Cellular connectomes as arbiters of local circuit models
884 in the cerebral cortex. *Nature Communications*. 2021 May; 12(1):2785. doi: .10.1038/s41467-021-22856-z.
- 885 **Kornfeld J**, Denk W. Progress and remaining challenges in high-throughput volume electron microscopy. *Cur-
886 rent Opinion in Neurobiology*. 2018 Jun; 50:261–267. doi: .10.1016/j.conb.2018.04.030.
- 887 **Le TA**, Baydin AG, Zinkov R, Wood F. Using synthetic data to train neural networks is model-based reason-
888 ing. In: *2017 International Joint Conference on Neural Networks (IJCNN)*; 2017. p. 3514–3521. doi: .10.1109/I-
889 JCNN.2017.7966298.
- 890 **Lomba S**, Straehle J, Gangadharan V, Heike N, Khalifa A, Motta A, Ju N, Sievers M, Gempt J, Meyer HS, Helm-
891 staedter M. Connectomic comparison of mouse and human cortex. *Science*. 2022 Jun; 377(6602):eabo0924.
892 doi: .10.1126/science.abo0924.
- 893 **Lopez-Paz D**, Oquab M, Revisiting Classifier Two-Sample Tests. arXiv; 2018. doi: .10.48550/arXiv.1610.06545.
- 894 **Lueckmann JM**, Bassetto G, Karaletsos T, Macke JH. Likelihood-free inference with emulator networks. In:
895 *Proceedings of The 1st Symposium on Advances in Approximate Bayesian Inference* PMLR; 2019. p. 32–53. doi:
896 .10.48550/arxiv.1805.09294.
- 897 **Lueckmann JM**, Boelts J, Greenberg D, Goncalves P, Macke J. Benchmarking Simulation-Based Inference. In:
898 *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics* PMLR; 2021. p. 343–351.
899 doi: .10.48550/arXiv.2101.04653.

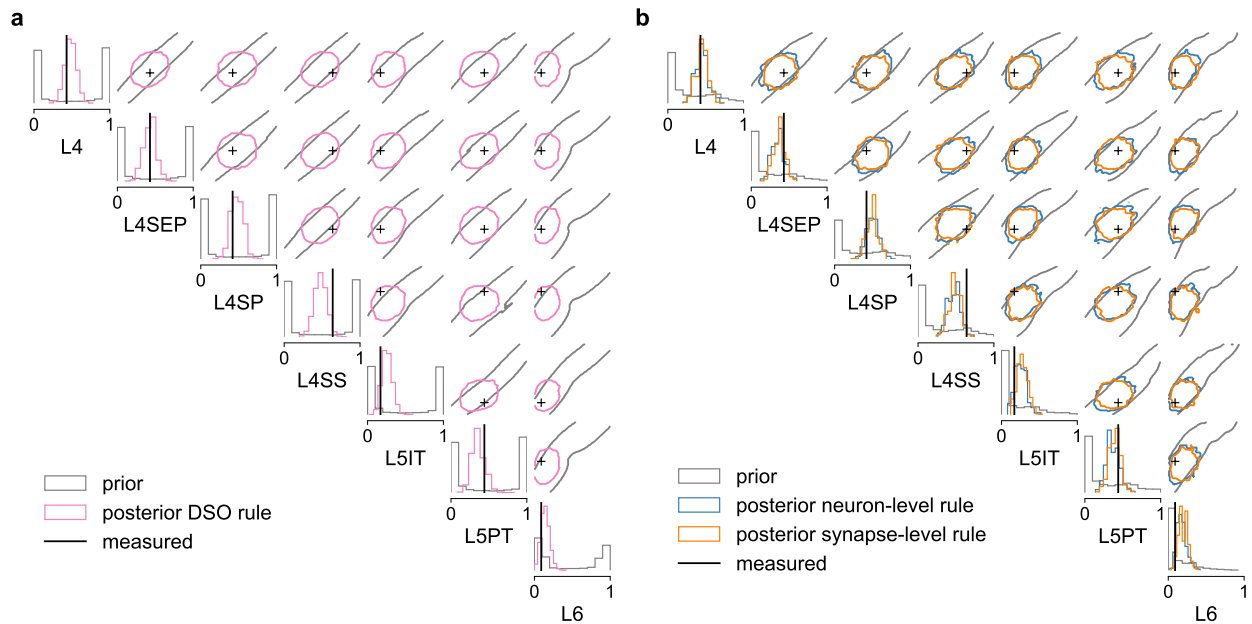
- 900 **Lueckmann JM**, Goncalves PJ, Bassetto G, Öcal K, Nonnenmacher M, Macke JH. Flexible statistical inference
901 for mechanistic models of neural dynamics. In: *Advances in Neural Information Processing Systems 30* Curran
902 Associates, Inc.; 2017.p. 1289–1299. doi: [.10.48550/arxiv.1711.01861](https://doi.org/10.48550/arxiv.1711.01861).
- 903 **Luppi AI**, Cabral J, Cofre R, Destexhe A, Deco G, Kringelbach ML. Dynamical models to evaluate structure-
904 function relationships in network neuroscience. *Nature Reviews Neuroscience*. 2022 Oct; p. 1–2. doi:
905 [.10.1038/s41583-022-00646-w](https://doi.org/10.1038/s41583-022-00646-w).
- 906 **Macrina T**, Lee K, Lu R, Turner NL, Wu J, Popovych S, Silversmith W, Kemnitz N, Bae JA, Castro MA, Dorkenwald
907 S, Halageri A, Jia Z, Jordan C, Li K, Mitchell E, Mondal SS, Mu S, Nehoran B, Wong W, et al., Petascale neural
908 circuit reconstruction: automated methods. *bioRxiv*; 2021. doi: [.10.1101/2021.08.04.455162](https://doi.org/10.1101/2021.08.04.455162).
- 909 **Marlier N**, Bröls O, Louppe G, Simulation-based Bayesian inference for multi-fingered robotic grasping. *arXiv*;
910 2021. doi: [.10.48550/arXiv.2109.14275](https://doi.org/10.48550/arXiv.2109.14275).
- 911 **Meyer HS**, Egger R, Guest JM, Foerster R, Reissl S, Oberlaender M. Cellular organization of cortical barrel
912 columns is whisker-specific. *Proceedings of the National Academy of Sciences*. 2013 Nov; 110(47):19113–
913 19118. doi: [.10.1073/pnas.1312691110](https://doi.org/10.1073/pnas.1312691110), publisher: Proceedings of the National Academy of Sciences.
- 914 **Meyer HS**, Wimmer VC, Oberlaender M, de Kock CPJ, Sakmann B, Helmstaedter M. Number and Laminar
915 Distribution of Neurons in a Thalamocortical Projection Column of Rat Vibrissal Cortex. *Cerebral Cortex*.
916 2010 Oct; 20(10):2277–2286. doi: [.10.1093/cercor/bhq067](https://doi.org/10.1093/cercor/bhq067).
- 917 **MICrONS-Consortium**, Bae JA, Baptiste M, Bodor AL, Brittain D, Buchanan J, Bumbarger DJ, Castro MA, Celii B,
918 Cobos E, Collman F, da Costa NM, Dorkenwald S, Elabbady L, Fahey PG, Fliss T, Froudarakis E, Gager J, Gamlin
919 C, Halageri A, et al. Functional connectomics spanning multiple areas of mouse visual cortex. *bioRxiv*. 2021;
920 doi: [.10.1101/2021.07.28.454025](https://doi.org/10.1101/2021.07.28.454025).
- 921 **Mishchenko Y**, Hu T, Spacek J, Mendenhall J, Harris KM, Chklovskii DB. Ultrastructural Analysis of Hip-
922 pocampal Neuropil from the Connectomics Perspective. *Neuron*. 2010 Sep; 67(6):1009–1020. doi:
923 [.10.1016/j.neuron.2010.08.014](https://doi.org/10.1016/j.neuron.2010.08.014).
- 924 **Motta A**, Berning M, Boergens KM, Staffler B, Beining M, Loomba S, Hennig P, Wissler H, Helmstaedter M. Dense
925 connectomic reconstruction in layer 4 of the somatosensory cortex. *Science*. 2019 Nov; 366(6469):eaay3134.
926 doi: [.10.1126/science.aay3134](https://doi.org/10.1126/science.aay3134).
- 927 **Narayanan RT**, Egger R, Johnson AS, Mansvelter HD, Sakmann B, de Kock CPJ, Oberlaender M. Beyond Colum-
928 nar Organization: Cell Type- and Target Layer-Specific Principles of Horizontal Axon Projection Patterns in
929 Rat Vibrissal Cortex. *Cerebral Cortex*. 2015 Nov; 25(11):4450–4468. doi: [.10.1093/cercor/bhv053](https://doi.org/10.1093/cercor/bhv053).
- 930 **Neal RM**. Slice sampling. *The Annals of Statistics*. 2003 Jun; 31(3):705–767. doi: [.10.1214/aos/1056562461](https://doi.org/10.1214/aos/1056562461).
- 931 **Nelder JA**, Wedderburn RWM. Generalized Linear Models. *Journal of the Royal Statistical Society Series A*
932 (General). 1972; 135(3):370–384. doi: [.10.2307/2344614](https://doi.org/10.2307/2344614).
- 933 **Oesterle J**, Behrens C, Schröder C, Hermann T, Euler T, Franke K, Smith RG, Zeck G, Berens P. Bayesian inference
934 for biophysical neuron models enables stimulus optimization for retinal neuroprosthetics. *eLife*. 2020 Oct;
935 9:e54997. doi: [.10.7554/eLife.54997](https://doi.org/10.7554/eLife.54997).
- 936 **Osten P**, Margrie TW. Mapping brain circuitry with a light microscope. *Nature Methods*. 2013 Jun; 10(6):515–
937 523. doi: [.10.1038/nmeth.2477](https://doi.org/10.1038/nmeth.2477).
- 938 **Papamakarios G**, Murray I. Fast ϵ -free Inference of Simulation Models with Bayesian Conditional Den-
939 sity Estimation. In: *Advances in Neural Information Processing Systems 29* Curran Associates, Inc.; 2016. p.
940 1028–1036. doi: [.10.48550/arxiv.1605.06376](https://doi.org/10.48550/arxiv.1605.06376).
- 941 **Papamakarios G**, Nalisnick E, Rezende DJ, Mohamed S, Lakshminarayanan B. Normalizing flows for
942 probabilistic modeling and inference. *Journal of Machine Learning Research*. 2021; 22(57):1–64. doi:
943 [.10.48550/arxiv.1912.02762](https://doi.org/10.48550/arxiv.1912.02762).
- 944 **Papamakarios G**, Pavlakou T, Murray I. Masked Autoregressive Flow for Density Estimation. In: *Ad-
945 vances in Neural Information Processing Systems 30* Curran Associates, Inc.; 2017.p. 2338–2347. doi:
946 [.10.48550/arxiv.1705.07057](https://doi.org/10.48550/arxiv.1705.07057).
- 947 **Peters A**, Feldman ML. The projection of the lateral geniculate nucleus to area 17 of the rat cerebral cortex. I.
948 General description. *Journal of neurocytology*. 1976; 5(1):63–84. doi: [.10.1007/bf01176183](https://doi.org/10.1007/bf01176183).

- 949 **Peysers A**, Diaz Pier S, Klijn W, Morrison A, Triesch J. Editorial: Linking experimental and computational connectomics. *Network Neuroscience*. 2019 Sep; 3(4):902–904. doi: [.10.1162/netn_e_00108](https://doi.org/10.1162/netn_e_00108).
- 951 **Radev ST**, D'Alessandro M, Mertens UK, Voss A, Köthe U, Bürkner PC. Amortized Bayesian Model Comparison With Evidential Deep Learning. *IEEE Transactions on Neural Networks and Learning Systems*. 2021; p. 1–15. doi: [.10.1109/TNNLS.2021.3124052](https://doi.org/10.1109/TNNLS.2021.3124052).
- 954 **Ramesh P**, Lueckmann JM, Boelts J, Tejero-Cantero A, Greenberg DS, Goncalves PJ, Macke JH. GATSBI: Generative Adversarial Training for Simulation-Based Inference. In: *International Conference on Learning Representations*; 2022. doi: [.10.48550/arxiv.2203.06481](https://doi.org/10.48550/arxiv.2203.06481).
- 957 **Ratmann O**, Jørgensen O, Hinkley T, Stumpf M, Richardson S, Wiuf C. Using Likelihood-Free Inference to Compare Evolutionary Dynamics of the Protein Networks of *H. pylori* and *P. falciparum*. *PLOS Computational Biology*. 2007 Nov; 3(11):e230. doi: [.10.1371/journal.pcbi.0030230](https://doi.org/10.1371/journal.pcbi.0030230).
- 960 **Rees CL**, Moradi K, Ascoli GA. Weighing the Evidence in Peters' Rule: Does Neuronal Morphology Predict Connectivity? *Trends in Neurosciences*. 2017 Feb; 40(2):63–71. doi: [.10.1016/j.tins.2016.11.007](https://doi.org/10.1016/j.tins.2016.11.007).
- 962 **Reimann MW**, King JG, Muller EB, Ramaswamy S, Markram H. An algorithm to predict the connectome of neural microcircuits. *Frontiers in Computational Neuroscience*. 2015; 9. doi: [.10.3389/fncom.2015.00120](https://doi.org/10.3389/fncom.2015.00120).
- 964 **Rosenbluth MN**, Rosenbluth AW. Monte Carlo calculation of the average extension of molecular chains. *The Journal of Chemical Physics*. 1955; 23(2):356–359. doi: [.10.1063/1.1741967](https://doi.org/10.1063/1.1741967).
- 966 **Sabbagh D**, Ablin P, Varoquaux G, Gramfort A, Engemann DA. Predictive regression modeling with MEG/EEG: from source power to signals and cognitive states. *NeuroImage*. 2020 Nov; 222:116893. doi: [.10.1016/j.neuroimage.2020.116893](https://doi.org/10.1016/j.neuroimage.2020.116893).
- 969 **Schmitt M**, Bürkner PC, Köthe U, Radev ST, Detecting Model Misspecification in Amortized Bayesian Inference with Neural Networks. *arXiv*; 2022. doi: [.10.48550/arXiv.2112.08866](https://doi.org/10.48550/arXiv.2112.08866).
- 971 **Shapson-Coe A**, Januszewski M, Berger DR, Pope A, Wu Y, Blakely T, Schalek RL, Li P, Wang S, Maitin-Shepard J, Karlupia N, Dorkenwald S, Sjostedt E, Leavitt L, Lee D, Bailey L, Fitzmaurice A, Kar R, Field B, Wu H, et al. A connectomic study of a petascale fragment of human cerebral cortex. *bioRxiv*. 2021; doi: [.10.1101/2021.05.29.446289](https://doi.org/10.1101/2021.05.29.446289).
- 975 **Sisson SA**, Fan Y, Tanaka MM. Sequential Monte Carlo without likelihoods. *Proceedings of the National Academy of Sciences*. 2007 Feb; 104(6):1760–1765. doi: [.10.1073/pnas.0607208104](https://doi.org/10.1073/pnas.0607208104).
- 977 **Sisson SA**, Fan Y, Beaumont M. Handbook of approximate Bayesian computation. Chapman and Hall/CRC; 2018. doi: [.10.48550/arxiv.1802.09720](https://doi.org/10.48550/arxiv.1802.09720).
- 979 **Sporns O**, Bassett DS. Editorial: New Trends in Connectomics. *Network Neuroscience*. 2018 Jun; 2(2):125–127. doi: [.10.1162/netn_e_00052](https://doi.org/10.1162/netn_e_00052).
- 981 **Sporns O**, Tononi G, Kötter R. The Human Connectome: A Structural Description of the Human Brain. *PLOS Computational Biology*. 2005 Sep; 1(4):e42. doi: [.10.1371/journal.pcbi.0010042](https://doi.org/10.1371/journal.pcbi.0010042).
- 983 **Talts S**, Betancourt M, Simpson D, Vehtari A, Gelman A, Validating Bayesian Inference Algorithms with Simulation-Based Calibration. *arXiv*; 2020. doi: [.10.48550/arXiv.1804.06788](https://doi.org/10.48550/arXiv.1804.06788).
- 985 **Tejero-Cantero* A**, Boelts* J, Deistler* M, Lueckmann* JM, Durkan* C, Gonçalves PJ, Greenberg DS, Macke JH. sbi: A toolkit for simulation-based inference. *Journal of Open Source Software*. 2020 Aug; 5(52):2505. doi: [.10.21105/joss.02505](https://doi.org/10.21105/joss.02505).
- 988 **Toni T**, Welch D, Strelkowa N, Ipsen A, Stumpf MPH. Approximate Bayesian computation scheme for parameter inference and model selection in dynamical systems. *Journal of The Royal Society Interface*. 2009 Feb; 6(31):187–202. doi: [.10.1098/rsif.2008.0172](https://doi.org/10.1098/rsif.2008.0172).
- 991 **Triesch J**, Hilgetag CC. Computational connectomics. *e-Neuroforum*. 2016 Sep; 7(3):43–44. doi: [.10.1007/s13295-016-0029-z](https://doi.org/10.1007/s13295-016-0029-z).
- 993 **Turner NL**, Macrina T, Bae JA, Yang R, Wilson AM, Schneider-Mizell C, Lee K, Lu R, Wu J, Bodor AL, Bleckert AA, Brittain D, Froudarakis E, Dorkenwald S, Collman F, Kemnitz N, Ih D, Silversmith WM, Zung J, Zlateski A, et al. Reconstruction of neocortex: Organelles, compartments, cells, circuits, and activity. *Cell*. 2022; 185:1082–1100.e24. doi: [.10.1016/j.cell.2022.01.023](https://doi.org/10.1016/j.cell.2022.01.023).

- 997 **Udvary D**, Harth P, Macke JH, Hege HC, de Kock CPJ, Sakmann B, Oberlaender M. The impact of
998 neuron morphology on cortical network architecture. *Cell Reports*. 2022 Apr; 39(2):110677. doi:
999 .10.1016/j.celrep.2022.110677.
- 1000 **Valdes-Aleman J**, Fetter RD, Sales EC, Heckman EL, Venkatasubramanian L, Doe CQ, Landgraf M, Cardona A,
1001 Zlatic M. Comparative Connectomics Reveals How Partner Identity, Location, and Activity Specify Synaptic
1002 Connectivity in *Drosophila*. *Neuron*. 2021 Jan; 109(1):105–122.e7. doi: .10.1016/j.neuron.2020.10.004.
- 1003 **Váša F**, Mišić B. Null models in network neuroscience. *Nature Reviews Neuroscience*. 2022 Aug; 23(8):493–504.
1004 doi: .10.1038/s41583-022-00601-9.
- 1005 **Vértes PE**, Alexander-Bloch AF, Gogtay N, Giedd JN, Rapoport JL, Bullmore ET. Simple models of human brain
1006 functional networks. *Proceedings of the National Academy of Sciences*. 2012; 109(15):5868–5873. doi:
1007 .10.1073/pnas.1111738109.
- 1008 **Ward D**, Cannon P, Beaumont M, Fasiolo M, Schmon SM. Robust Neural Posterior Estimation and Statistical
1009 Model Criticism. In: *Advances in Neural Information Processing Systems 30* Curran Associates, Inc.; 2022. doi:
1010 .10.48550/arXiv.2210.06564.
- 1011 **de Witt CS**, Gram-Hansen B, Nardelli N, Gambardella A, Zinkov R, Dokania P, Siddharth N, Espinosa-Gonzalez
1012 AB, Darzi A, Torr P, Baydin AG, Simulation-Based Inference for Global Health Decisions. *arXiv*; 2020. doi:
1013 .10.48550/arXiv.2005.07062.



Supplementary Figure S1. Validating SBI over wiring rule parameters with simulated data. (a) Comparison of the posterior over wiring rule parameters inferred with SBI (using the SNPE algorithm, blue) and the reference solution (ground-truth parameters in black). (b) Inference accuracy in terms of classifier-2-sample-test accuracy (C2ST) between reference solution and three SBI algorithms, plotted as a function of training simulations (0.5 is best, error bars show standard error over ten different observations). (c) Distributions of posterior ranks from Simulation-based calibration, obtained for each parameter separately from NPE applied to the full version of the DSO rule simulator. A well-calibrated posterior should have uniformly distributed ranks, as indicated by the area shaded gray. (d) Comparison of data simulated with samples drawn from the NPE posterior (posterior predictive distribution, orange) and the underlying observed data (simulated, black).



Supplementary Figure S2. Predictive distributions. Comparison of connection probabilities simulated from the prior (gray), the SBI posterior (colors), and the measured data (black). Shown for the dense structural overlap rule (DSO) (see Results) in (a) and for the proximity-based wiring rules on the neuron level and the synapse level in (b). The prior predictive distributions cover a wide range of values, whereas the posterior predictive distributions cluster around the measured connection probabilities taking into account the measurement noise.

1015 **Supplementary material**

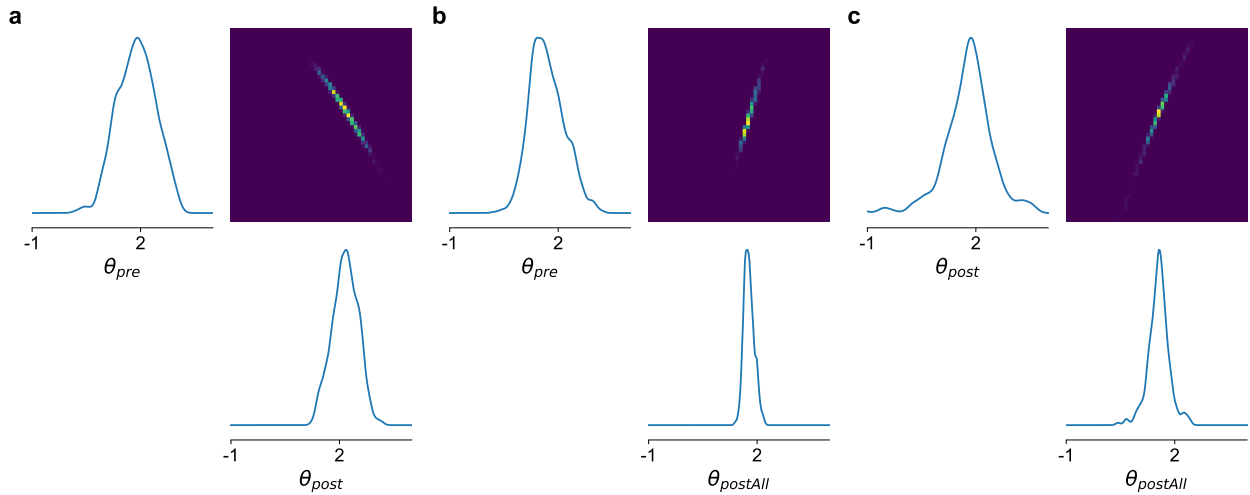
1016 **Prior predictive checks**

1017 An essential requirement for a generative model is that it can actually generate the measured
 1018 data, i.e., that the model is not misspecified. The prior predictive distribution shows all data that
 1019 can be generated by the model when sampling parameters from the prior. Thus, this distribution
 1020 provides a tool to check for misspecification. For the wiring rule simulator, the prior predictive is a
 1021 seven-dimensional distribution (Fig. S4b).

1022 We found that with the chosen setting of the prior, simulator, and summary statistics (see Al-
 1023 gorithm 2), the prior predictive distribution covers a large range of plausible values, including the
 1024 experimental measurements (Fig. S4b). The two-dimensional marginals show strong positive cor-
 1025 relations between all seven connection probabilities and additional blocks of stronger correlations
 1026 within layer 4 (L4, L4SEP, L4SP, L4SS) and layer 5 (L5PT, L5IT), visible in the correlations matrix
 1027 (Fig. S4a). These correlations are plausible because all populations share the same source popula-
 1028 tions in the thalamus, and the blocks of stronger correlations correspond to connection probabili-
 1029 ties within the same target layer in the cortex. Furthermore, depending on the number of neuron
 1030 pairs used to calculate connection probabilities (see Mapping from simulated connectomes to mea-
 1031 sured connectivity data), the simulator accurately matches the empirical variance expected from
 1032 the measured data (Fig. S4c). These results indicated that a subsampling of 50 pairs was adequate
 1033 to model the experiments.

1034 **Alternative parametrizations of the dense structural overlap rule**

The strong conditional correlations between the three parameters of the dense structural overlap
 rule (DSO, see Posterior analysis reveals biologically plausible parameter interactions) indicated
 that an alternative parametrization of the DSO rule, e.g., using only two parameters, could also be
 able to explain the measured data. We tested this hypothesis by changing the DSO rule as follows.



Supplementary Figure S3. Conditional posterior distributions. We obtained conditional posterior distributions for the DSO rule (see Results) by conditioning one of the three parameters to a value drawn from the posterior. The resulting two-dimensional posterior over θ_{pre} and θ_{post} (**a**), θ_{pre} and $\theta_{postAll}$ (**b**) and θ_{post} and $\theta_{postAll}$ (**c**), showed strong correlations as visible on the off-diagonal two-dimension marginals.

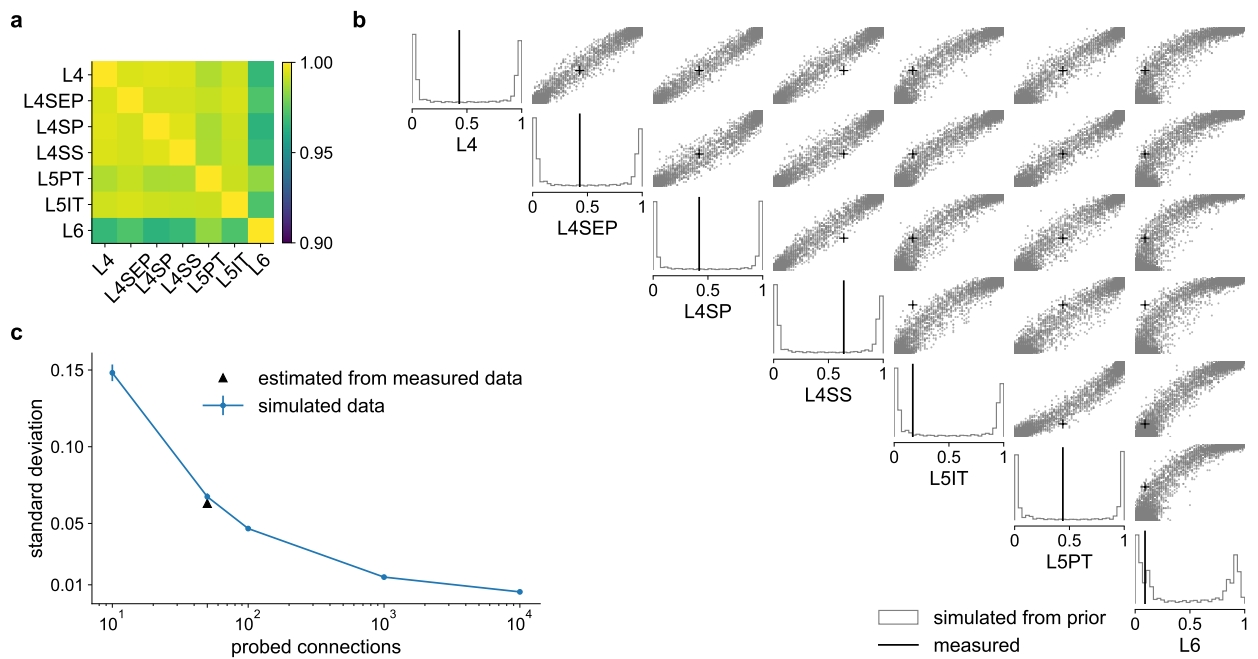
The initial formulation of the rule is given by

$$DSO_{i,j,k}(\theta) = \frac{pre_i^{\theta_{pre}} \cdot post_j^{\theta_{post}}}{postAll_k^{\theta_{postAll}}}. \quad (10)$$

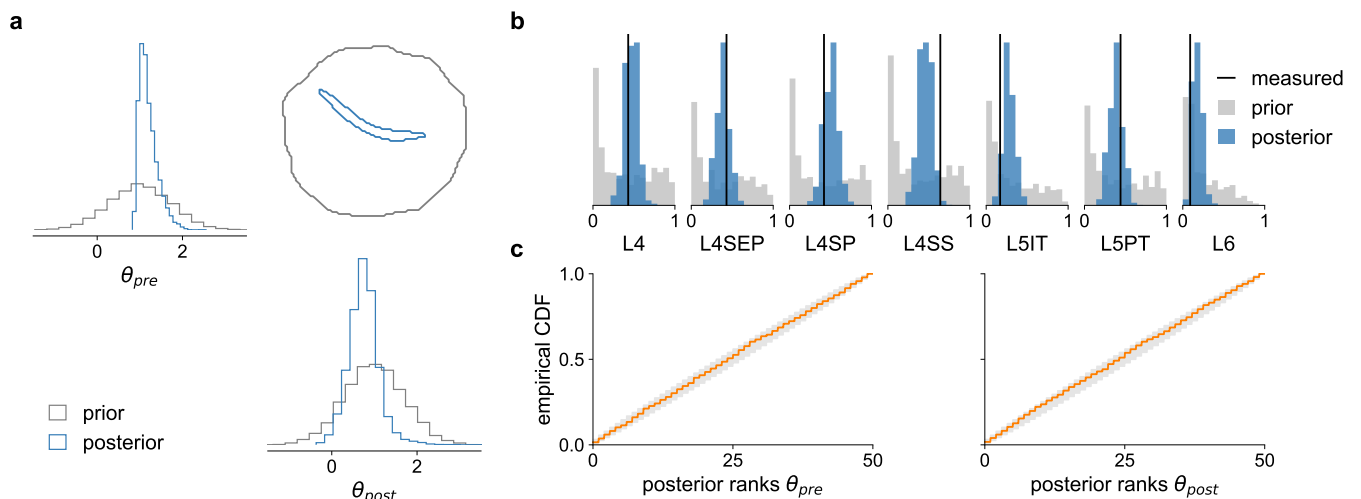
The *postAll* structural feature in the denominator contains the post-synaptic target densities of *all* post-synaptic structures in a given subvolume *k* of the structural model, including the *post_j* features. It acts as a normalizing factor for the rule, e.g., when the weight θ_{post} increases, a corresponding increase in $\theta_{postAll}$ can maintain a similar overall DSO value. We changed the DSO rule by removing the parameter $\theta_{postAll}$, but including the *scaled* features $post_j^{\theta_{post}}$ into *postAll*, instead of only taking *post*. By taking the scaled version, we maintained the normalizing property of the denominator of the DSO rule: If θ_{post} was high, in the initial three-parameter formulation of the DSO rule, this could be compensated by a corresponding increase in $\theta_{postAll}$. While this was not possible anymore in the two-param version, the compensation occurred implicitly by including the scaled *post_j* features:

$$DSO_{i,j,k}(\theta) = \frac{pre_i^{\theta_{pre}} \cdot post_j^{\theta_{post}}}{(post_j^{\theta_{post}} + postAll_{k,-j})}. \quad (11)$$

1035 We performed SBI using the same settings as before to obtain the posterior distribution over
 1036 the two parameters of the reduced DSO rule (Fig. S5a). The resulting posterior predictive distri-
 1037 bution showed similar accuracy to the one of the three-parameter DSO rule (compare Fig. S5b
 1038 versus Fig. 3b). This result indicates that the two-parameter DSO rule provides an alternative to
 1039 the three-parameter rule.



Supplementary Figure S4. Prior predictive distribution for the DSO rule. (a) Estimated correlation matrix of the seven connection probabilities generated from the model (note the restricted range of the color map). (b) Marginal plot of the prior predictive distribution showing data simulated from 10k samples from the prior; one-dimensional marginal as histograms on the diagonal and two-dimensional marginals as scatter plots on the upper triangular subplots; literature data in black. (c) Size of the random subset of neuron pairs used to calculate the connection probabilities plotted against the resulting standard deviation (std) in the simulated connection probabilities. Calculated from 1,000 simulations given the same parameters, averaged over the seven connection probabilities (blue); compared with the empirical variance expected from the sample sizes used in the literature (black).



Supplementary Figure S5. SBI results for the two-parameter DSO rule. (a) Posterior distribution (blue) over the two parameters of the reduced DSO rule, inferred with SBI given the seven measured connection probabilities and the corresponding prior distribution (gray). (b) Connection probabilities simulated with parameters sampled from the posterior (blue) and the prior (gray), compared to the measured connection probabilities (black, *Bruno and Sakmann, 2006; Constantinople and Bruno, 2013*). (c) Distribution of posterior ranks for every rule parameter, calculated with simulation-based calibration (orange), compared to the desired uniform distribution (gray area on the diagonal).

sbi: A toolkit for simulation-based inference

Alvaro Tejero-Cantero^{e, 1}, Jan Boelts^{e, 1}, Michael Deistler^{e, 1},
Jan-Matthis Lueckmann^{e, 1, 3}, Conor Durkan^{e, 2}, Pedro J. Gonçalves^{1, 3},
David S. Greenberg^{1, 4}, and Jakob H. Macke^{1, 5, 6}

^e Equally contributing authors **1** Computational Neuroengineering, Department of Electrical and Computer Engineering, Technical University of Munich **2** School of Informatics, University of Edinburgh **3** Neural Systems Analysis, Center of Advanced European Studies and Research (caesar), Bonn **4** Model-Driven Machine Learning, Centre for Materials and Coastal Research, Helmholtz-Zentrum Geesthacht **5** Machine Learning in Science, University of Tübingen **6** Empirical Inference, Max Planck Institute for Intelligent Systems, Tübingen

DOI: [10.21105/joss.02505](https://doi.org/10.21105/joss.02505)

Software

- [Review](#) ↗
- [Repository](#) ↗
- [Archive](#) ↗

Editor: [Dan Foreman-Mackey](#) ↗

Reviewers:

- [@EiffL](#)
- [@cranmer](#)

Submitted: 14 July 2020

Published: 21 August 2020

License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](#)).

Summary

Scientists and engineers employ stochastic numerical simulators to model empirically observed phenomena. In contrast to purely statistical models, simulators express scientific principles that provide powerful inductive biases, improve generalization to new data or scenarios and allow for fewer, more interpretable and domain-relevant parameters. Despite these advantages, tuning a simulator's parameters so that its outputs match data is challenging. Simulation-based inference (SBI) seeks to identify parameter sets that a) are compatible with prior knowledge and b) match empirical observations. Importantly, SBI does not seek to recover a single 'best' data-compatible parameter set, but rather to identify all high probability regions of parameter space that explain observed data, and thereby to quantify parameter uncertainty. In Bayesian terminology, SBI aims to retrieve the posterior distribution over the parameters of interest. In contrast to conventional Bayesian inference, SBI is also applicable when one can run model simulations, but no formula or algorithm exists for evaluating the probability of data given parameters, i.e. the likelihood.

We present *sbi*, a PyTorch-based package that implements SBI algorithms based on neural networks. *sbi* facilitates inference on black-box simulators for practising scientists and engineers by providing a unified interface to state-of-the-art algorithms together with documentation and tutorials.

Motivation

Bayesian inference is a principled approach for determining parameters consistent with empirical observations: Given a prior over parameters, a stochastic simulator, and observations, it returns a posterior distribution. In cases where the simulator likelihood *can* be evaluated, many methods for approximate Bayesian inference exist (e.g., Metropolis, Rosenbluth, Rosenbluth, Teller, & Teller, 1953; Baydin et al., 2019; Graham & Storkey, 2017; Le, Baydin, & Wood, 2017; Neal, 2003). For more general simulators, however, evaluating the likelihood of data given parameters might be computationally intractable. Traditional algorithms for this 'likelihood-free' setting (Cranmer, Brehmer, & Louppe, 2020) are based on Monte-Carlo rejection (Pritchard, Seielstad, Perez-Lezaun, & Feldman, 1999; Sisson, Fan, & Tanaka, 2007), an approach known as *Approximate Bayesian Computation* (ABC). More recently, algorithms based on neural networks have been developed (Greenberg, Nonnenmacher, & Macke, 2019; Hermans, Begy, & Louppe, 2020; Lueckmann et al., 2017; Papamakarios & Murray, 2016;

Papamakarios, Sterratt, & Murray, 2019). These algorithms are not based on rejecting simulations, but rather train deep neural conditional density estimators or classifiers on simulated data. To aid in effective application of these algorithms to a wide range of problems, `sbi` closely integrates with PyTorch and offers state-of-the-art neural network-based SBI algorithms (Greenberg et al., 2019; Hermans et al., 2020; Papamakarios et al., 2019) with flexible choice of network architectures and flow-based density estimators. With `sbi`, researchers can easily implement new neural inference algorithms, benefiting from the infrastructure to manage simulators and a unified posterior representation. Users, in turn, can profit from a single inference interface that allows them to either use their own custom neural network, or choose from a growing library of preconfigured options provided with the package.

Related software and use in research

We are aware of several mature packages that implement SBI algorithms. `elfi` (Lintusaari et al., 2018) is a package offering BOLFI, a Gaussian process-based algorithm (Gutmann & Corander, 2016), and some classical ABC algorithms. The package `carl` (Louppe, Cranmer, & Pavez, 2016) implements the algorithm described in Cranmer, Pavez, & Louppe (2015). Two other SBI packages, currently under development, are `hypothesis` (Hermans, 2019) and `pydelfi` (Alsing, 2019). `pyabc` (Klinger, Rickert, & Hasenauer, 2018) and `ABCpy` (Dutta, Schoengens, Onnela, & Mira, 2017) are two packages offering a diversity of ABC algorithms.

`sbi` is closely integrated with PyTorch (Paszke et al., 2019) and uses `nflows` (Durkan, Bekasov, Papamakarios, & Murray, 2019) for flow-based density estimators. `sbi` builds on experience accumulated developing `delfi` (mackelab.org, 2017), which it succeeds. `delfi` was based on `theano` (Al-Rfou et al., 2016) (development discontinued) and developed both for SBI research (Greenberg et al., 2019; Lueckmann et al., 2017) and for scientific applications (Gonçalves et al., 2019). The `sbi` codebase started as a fork of `lfi` (Durkan, 2020), developed for Durkan et al. (2020).

Description

`sbi` currently implements three families of neural inference algorithms:

- Sequential Neural *Posterior* Estimation (SNPE) trains a deep neural density estimator that directly estimates the posterior distribution of parameters given data. Afterwards, it can sample parameter sets from the posterior, or evaluate the posterior density on any parameter set. Currently, SNPE-C (Greenberg et al., 2019) is implemented in `sbi`.
- Sequential Neural *Likelihood* Estimation (SNLE) (Papamakarios et al., 2019) trains a deep neural density estimator of the likelihood, which then allows to sample from the posterior using e.g. MCMC.
- Sequential Neural *Ratio* Estimation (SNRE) (Durkan et al., 2020; Hermans et al., 2020) trains a classifier to estimate density ratios, which in turn can be used to sample from the posterior e.g. with MCMC.

The inference step returns a `NeuralPosterior` object that represents the uncertainty about the parameters conditional on an observation, i.e. the posterior distribution. This object can be sampled from —and if the chosen algorithm allows, evaluated— with the same API as a standard PyTorch probability distribution.

An important challenge in making SBI algorithms usable by a broader community is to deal with diverse, often pre-existing, complex simulators. `sbi` works with any simulator as long as it can be wrapped in a Python callable. Furthermore, `sbi` ensures that custom simulators

work well with neural networks, e.g. by performing automatic shape inference, standardizing inputs or handling failed simulations. To maximize simulator performance, `sbi` leverages vectorization where available and optionally parallelizes simulations using `joblib` (Varoquaux, 2008). Moreover, if dimensionality reduction of the simulator output is desired, `sbi` can use a trainable summarizing network to extract relevant features from raw simulator output and spare the user manual feature engineering.

In addition to the full-featured interface, `sbi` provides also a *simple* interface which consists of a single function call with reasonable defaults. This allows new users to get familiarized with simulation-based inference and quickly obtain results without having to define custom networks or tune hyperparameters.

With `sbi`, we aim to support scientific discovery and computational engineering by making Bayesian inference applicable to the widest class of models (simulators with no likelihood available), and practical for complex problems. We have designed an open architecture and adopted community-oriented development practices in order to invite other machine-learning researchers to join us in this long-term vision.

Acknowledgements

This work has been supported by the German Federal Ministry of Education and Research (BMBF, project 'ADIMEM', FKZ 01IS18052 A-D), the German Research Foundation (DFG) through SFB 1089 'Synaptic Microcircuits', SPP 2041 'Computational Connectomics' and Germany's Excellence Strategy – EXC-Number 2064/1 – Project number 390727645.

Conor Durkan was supported by the EPSRC Centre for Doctoral Training in Data Science, funded by the UK Engineering and Physical Sciences Research Council (grant EP/L016427/1) and the University of Edinburgh.

We are grateful to Artur Bekasov, George Papamakarios and Iain Murray for making `nflows` (Durkan et al., 2019) available, a package for normalizing flow-based density estimation which `sbi` leverages extensively.

References

- Al-Rfou, R., Alain, G., Almahairi, A., Angermueller, C., Bahdanau, D., Ballas, N., Bastien, F., et al. (2016). Theano: A python framework for fast computation of mathematical expressions. *arXiv*, arXiv-1605.
- Alsing, J. (2019). `pydelfi`: Density estimation likelihood-free inference. *GitHub repository*. <https://github.com/justinalsing/pydelfi>; GitHub.
- Baydin, A. G., Shao, L., Bhimji, W., Heinrich, L., Meadows, L., Liu, J., Munk, A., 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). doi:10.1145/3295500.3356180
- Cranmer, K., Brehmer, J., & Louppe, G. (2020). The frontier of simulation-based inference. *Proceedings of the National Academy of Sciences*. doi:10.1073/pnas.1912789117
- Cranmer, K., Pavez, J., & Louppe, G. (2015). Approximating likelihood ratios with calibrated discriminative classifiers. *arXiv preprint arXiv:1506.02169*.
- Durkan, C. (2020). `lfi`. *GitHub repository*. <https://github.com/conormdurkan/lfi>; GitHub.
- Durkan, C., Bekasov, A., Papamakarios, G., & Murray, I. (2019). `nflows`: Normalizing flows in PyTorch. *GitHub repository*. <https://github.com/bayesiains/nflows>; GitHub.

- Durkan, C., Murray, I., & Papamakarios, G. (2020). On contrastive learning for likelihood-free inference. *Proceedings of the 36th International Conference on Machine Learning*, Proceedings of machine learning research, 98.
- Dutta, R., Schoengens, M., Onnela, J.-P., & Mira, A. (2017). ABCpy: A user-friendly, extensible, and parallel library for approximate bayesian computation. In *Proceedings of the platform for advanced scientific computing conference* (pp. 8:1–8:9). doi:[10.1145/3093172.3093233](https://doi.org/10.1145/3093172.3093233)
- Gonçalves, P. J., Lueckmann, J.-M., Deistler, M., Nonnenmacher, M., Öcal, K., Bassetto, G., Chintaluri, C., et al. (2019). Training deep neural density estimators to identify mechanistic models of neural dynamics. *bioRxiv*, 838383. doi:[10.1101/838383](https://doi.org/10.1101/838383)
- Graham, M. M., & Storkey, A. J. (2017). Asymptotically exact inference in differentiable generative models. *Electronic Journal of Statistics*, 11(2), 5105–5164. doi:[10.1214/17-EJS1340SI](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*, Proceedings of machine learning research (Vol. 97, pp. 2404–2414). PMLR.
- 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.
- Hermans, J. (2019). Hypothesis. *GitHub repository*. <https://github.com/montefiore-ai/hypothesis>; GitHub.
- Hermans, J., Begy, V., & Louppe, G. (2020). Likelihood-free MCMC with approximate likelihood ratios. In *Proceedings of the 37th international conference on machine learning*, Proceedings of machine learning research (Vol. 98). PMLR.
- Klinger, E., Rickert, D., & Hasenauer, J. (2018). pyABC: Distributed, likelihood-free inference. *Bioinformatics*, 34(20), 3591–3593. doi:[10.1093/bioinformatics/bty361](https://doi.org/10.1093/bioinformatics/bty361)
- Le, T. A., Baydin, A. G., & Wood, F. (2017). Inference compilation and universal probabilistic programming. *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS) 2017*, 54.
- Lintusaari, J., Vuollekoski, H., Kangasrääsiö, A., Skytén, K., Järvenpää, M., Marttinen, P., Gutmann, M. U., et al. (2018). ELFI: Engine for likelihood-free inference. *The Journal of Machine Learning Research*, 19(1), 643–649.
- Louppe, G., Cranmer, K., & Pavez, J. (2016). carl: A likelihood-free inference toolbox. *Journal of Open Source Software*, 1(1), 11. doi:[10.21105/joss.00011](https://doi.org/10.21105/joss.00011)
- Lueckmann, J.-M., Goncalves, 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).
- mackelab.org. (2017). DELFI: Density estimation likelihood-free inference. *GitHub repository*. <https://github.com/mackelab/delfi>; GitHub.
- Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., & Teller, E. (1953). Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6), 1087–1092. doi:[10.1063/1.1699114](https://doi.org/10.1063/1.1699114)
- Neal, R. M. (2003). Slice sampling. *The Annals of Statistics*, 31(3), 705–741. doi:[10.1214/aos/1056562461](https://doi.org/10.1214/aos/1056562461)
- 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).

- Papamakarios, G., Sterratt, D., & Murray, I. (2019). Sequential neural likelihood: Fast likelihood-free inference with autoregressive flows. *Proceedings of Machine Learning Research*, Proceedings of machine learning research, 89, 837–848.
- Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., et al. (2019). PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32* (pp. 8024–8035).
- 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. doi:[10.1093/oxfordjournals.molbev.a026091](https://doi.org/10.1093/oxfordjournals.molbev.a026091)
- 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. doi:[10.1073/pnas.0607208104](https://doi.org/10.1073/pnas.0607208104)
- Varoquaux, G. (2008). Joblib. *GitHub repository*. <https://github.com/joblib/joblib>; GitHub.