All-in-one simulation-based inference

Manuel Gloeckler [1] **Michael Deistler** [1] **Christian Weilbach** [2] **Frank Wood** [2] **Jakob H. Macke** [1 3]

Abstract

Amortized Bayesian inference trains neural networks to solve stochastic inference problems using model simulations, thereby making it possible to rapidly perform Bayesian inference for any newly observed data. However, current simulation-based amortized inference methods are simulation-hungry and inflexible: They require the specification of a fixed parametric prior, simulator, and inference tasks ahead of time. Here, we present a new amortized inference method—the Simformer—which overcomes these limitations. By training a probabilistic diffusion model with transformer architectures, the Simformer outperforms current state-of-the-art amortized inference approaches on benchmark tasks and is substantially more flexible: It can be applied to models with function-valued parameters, it can handle inference scenarios with missing or unstructured data, and it can sample arbitrary conditionals of the joint distribution of parameters and data, including both posterior and likelihood. We showcase the performance and flexibility of the Simformer on simulators from ecology, epidemiology, and neuroscience, and demonstrate that it opens up new possibilities and application domains for amortized Bayesian inference on simulation-based models.

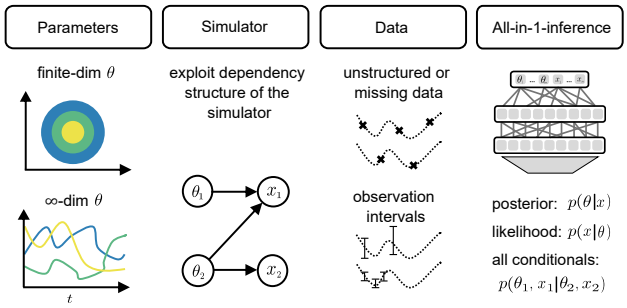

Figure 1. Capabilities of the Simformer: It can perform inference for simulators with a finite number of parameters or function-valued parameters (first column), it can exploit dependency structures of the simulator to improve accuracy (second column), it can perform inference for unstructured or missing data, for observation intervals (third column), and it provides an 'all-in-one' inference method that can sample all conditionals of the joint distribution, including posterior and likelihood (fourth column).

1. Introduction

Numerical simulators play an important role across various scientific and engineering domains, offering mechanistic insights into empirically observed phenomena (Gonçalves et al., 2020; Dax et al., 2021; Marlier et al., 2022). A fundamental challenge in these simulators is the identification of unobservable parameters based on empirical data, a task addressed by simulation-based inference (SBI) (Cranmer et al., 2020), which aims to perform Bayesian inference using samples from a (possibly blackbox) simulator, without requiring access to likelihood evaluations. A common approach in SBI is to train a neural network on pairs of parameters and corresponding simulation outputs: After an initial investment in simulations and network training, inference for any observation can then be performed without further simulations. These methods thereby *amortize* the cost of Bayesian inference.

Many methods for amortized SBI have been developed recently (Papamakarios & Murray, 2016; Lueckmann et al., 2017; Le et al., 2017; Greenberg et al., 2019; Papamakarios et al., 2019; Radev et al., 2020; Hermans et al., 2020; Glöckler et al., 2022; Boelts et al., 2022; Deistler et al., 2022a; Simons et al., 2023). While these methods have different strengths and weaknesses, most of them also share limitations. First, they often rely on structured, tabular data (typically $\boldsymbol{\theta}, \boldsymbol{x}$ vectors). Yet, real-world datasets are often more messy (Shukla & Marlin, 2021): Irregularly sampled time series naturally arise in domains like ecology, climate

[1]Machine Learning in Science, University of Tübingen and Tübingen AI Center, Tübingen, Germany [2]Department of Computer Science, University of British Columbia, Vancouver, Canada [3]Max Planck Institute for Intelligent Systems, Department Empirical Inference, Tübingen, Germany. Correspondence to: Manuel Gloeckler <manuel.gloeckler@uni-tuebingen.de>, Jakob H. Macke <jakob.macke@uni-tuebingen.de>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

science, and health sciences. Missing values often occur in real-world observations and are not easily handled by existing approaches. Second, the inputs of a simulator can correspond to a function of time or space, i.e., ∞-dimensional parameters (Chen et al., 2020; Ramesh et al., 2022). Existing amortized methods typically necessitate discretization, thereby limiting their applicability to a specific, often dense grid and precludes the evaluation of the parameter posterior beyond this grid. Third, they require specification of a fixed approximation task: the neural network can either target the likelihood (neural likelihood estimation, NLE, Papamakarios et al. (2019)) or the posterior (neural posterior estimation, NPE, Papamakarios & Murray (2016)). In practice, users might want to interactively explore both conditional distributions, investigate posteriors conditioned on subsets of data and parameters, or even explore different prior configurations. Fourth, while neural-network based SBI approaches are more efficient than classical ABC-methods (Lueckmann et al., 2021), they are still simulation-hungry. In part, this is because they target blackbox simulators, i.e., they do not require any access to the model's inner workings. However, in practice, one has at least *partial* knowledge (or assumptions) about the structure of the simulator (i.e., its conditional independencies), but common SBI methods cannot exploit such knowledge. These limitations have prevented the application of SBI in *interactive* applications, in which properties of the task need to be changed on the fly.

Here, we develop a new method for amortized Bayesian inference—the Simformer—which overcomes these limitations (Fig. 1), using a combination of transformers and probabilistic diffusion models (Peebles & Xie, 2022; Hatamizadeh et al., 2023), based on the idea of graphically structure diffusion models proposed by Weilbach et al. (2023). Our method can deal with unstructured and missing data and handles both parametric and nonparametric simulators (i.e., with function-valued ∞-dimensional) parameters. In addition, the method returns a single network that can be queried to sample *all* conditionals of the joint distribution (including the posterior, likelihood, and arbitrary parameter conditionals) and can also perform inference if the observations are intervals as opposed to specific values. We show that our method has higher accuracy than previous SBI methods on benchmark tasks (for a given simulation budget). Moreover, by using attention masks, one can use domain knowledge to adapt the Simformer to the dependency structure of the simulator (Weilbach et al., 2023) to further improve simulation efficiency. Thus, the Simformer provides an 'all-in-one' inference method that encapsulates posterior- and likelihood-estimation approaches and expands the space of data, simulators, and tasks for which users can perform simulation-based amortized Bayesian inference.

2. Preliminaries

2.1. Problem setting and approach

We consider a simulator with parameters $\boldsymbol{\theta}$ (potentially nonparametric) which stochastically generates samples $\boldsymbol{x}$ from its implicit likelihood $p(\boldsymbol{x}|\boldsymbol{\theta})$. After having observed data $\boldsymbol{x}_o$, we aim to infer the posterior distribution $p(\boldsymbol{\theta}|\boldsymbol{x}_o)$ of parameters given data, but also retain the flexibility to capture any other conditional of the full joint $p(\boldsymbol{\theta}, \boldsymbol{x})$. We, therefore, introduce the joint $\hat{\boldsymbol{x}} = (\boldsymbol{\theta}, \boldsymbol{x})$, that will serve as input for a transformer together with a mask indicating which values are *observed*. The transformer will then use attention mechanisms to compute the corresponding sequence of output scores of equal size. The scores corresponding to *unobserved* variables will then form the basis of a diffusion model representing the distribution over these variables.We first give background on the main ingredients (transformers and score-based diffusion models) in this section before giving a detailed description in Sec. 3.

2.2. Transformers and attention mechanisms

Transformers overcome limitations of feed-forward networks in effectively dealing with sequential inputs. They incorporate an attention mechanism which, for a given sequence of inputs, replaces individual hidden states with a weighted combination of all hidden states (Vaswani et al., 2017). Given three learnable linear projections of each hidden state (Q, K, V) this is computed as

$$\text{attention}(Q, K, V) = \text{softmax}(QK)^T/\sqrt{d})V.$$

2.3. Score-based diffusion models

Score-based diffusion models (Song et al., 2021b; Song & Ermon, 2019) describe the evolution of data through stochastic differential equations (SDEs). Common SDEs for score-based diffusion models can be expressed as

$$d\hat{\boldsymbol{x}}_t = f(\hat{\boldsymbol{x}}_t, t)dt + g(t)d\boldsymbol{w},$$

with $\boldsymbol{w}$ being a standard Wiener process, and f and g representing the drift and diffusion coefficients, respectively. The solution to this SDE defines a diffusion process that transforms an initial data distribution $p_0(\hat{\boldsymbol{x}}_0) = p(\hat{\boldsymbol{x}})$ into a simpler noise distribution $p_T(\hat{\boldsymbol{x}}_T) \approx \mathcal{N}(\hat{\boldsymbol{x}}_T; \boldsymbol{\mu}_T, \boldsymbol{\sigma}_T)$.

Samples from the generative model are then generated by simulating the reverse diffusion process (Anderson, 1982)

$$d\hat{\boldsymbol{x}}_t = \left[f(\hat{\boldsymbol{x}}_t, t) - g(t)^2 s(\hat{\boldsymbol{x}}_t, t)\right] dt + g(t)d\tilde{\boldsymbol{w}},$$

where $\tilde{\boldsymbol{w}}$ is a backward-in-time Wiener process. This relies on the knowledge of the score function $s(\hat{\boldsymbol{x}}_t, t) = \nabla_{\hat{\boldsymbol{x}}_t} \log p_t(\hat{\boldsymbol{x}}_t)$ at each step. The exact marginal score is typically intractable but can be estimated through time-dependent denoising score-matching (Hyvärinen & Dayan,

2005; Song et al., 2021b). Given that the conditional score is known, $p_t(\hat{\boldsymbol{x}}_t|\hat{\boldsymbol{x}}_0) = \mathcal{N}(\hat{\boldsymbol{x}}_t; \mu_t(\hat{\boldsymbol{x}}_0), \sigma_t(\hat{\boldsymbol{x}}_0))$, the score model $s_\phi(\hat{\boldsymbol{x}}_t, t)$ is trained to minimize the loss

$$\mathcal{L}(\phi) = \mathbb{E}_{t,\hat{\boldsymbol{x}}_0,\hat{\boldsymbol{x}}_t}\left[\lambda(t)\left\|s_\phi(\hat{\boldsymbol{x}}_t, t) - \nabla_{\hat{\boldsymbol{x}}_t}\log p_t(\hat{\boldsymbol{x}}_t|\hat{\boldsymbol{x}}_0)\right\|_2^2\right],$$

where λ denotes a positive weighting function. This objective, hence only requires samples from the original distribution $\hat{\boldsymbol{x}}_0 \sim p(\hat{\boldsymbol{x}})$.

3. Methods

The Simformer is a probabilistic diffusion model that uses a transformer to estimate the score (Weilbach et al. (2023); Hatamizadeh et al. (2023); Peebles & Xie (2022), Fig. 2). Unlike most previous approaches for simulation-based inference, which employ conditional density estimators to model either the likelihood or the posterior, the Simformer is trained on the *joint* distribution of parameters and data $p(\boldsymbol{\theta}, \boldsymbol{x}) =: p(\hat{\boldsymbol{x}})$. The Simformer encodes parameters and data (Sec. 3.1) such that arbitrary conditional distributions of the joint density (including posterior and likelihood) can still be sampled efficiently. The Simformer can encode known dependencies in the attention mask of the transformer (Sec. 3.2) and thereby ensures efficient training of arbitrary conditionals (Sec. 3.3). Finally, the Simformer uses guided diffusion to produce samples given arbitrary constraints (Sec. 3.4).

3.1. A Tokenizer for SBI

Transformers process sequences of uniformly sized vectors called tokens. Designing effective tokens is challenging and specific to the data at hand (Gu et al., 2022). The tokenizer represents each variable as an identifier that uniquely identifies the variable, a representation of the value of the variable, and a condition state (Fig. 2). The condition state is a binary variable and signifies whether the variable is conditioned on or not. It is resampled for every $(\boldsymbol{\theta}, \boldsymbol{x}) \in \mathbb{R}^d$ pair at every iteration of training. We denote the condition state of all variables as $M_C \in \{0, 1\}^d$. Setting $M_C = (0, \ldots, 0)$ corresponds to an unconditional diffusion model (Song et al., 2021b), whereas adopting $M_C^{(i)} = 1$ for data and $M_C^{(i)} = 0$ for parameters corresponds to training a conditional diffusion model of the posterior distribution (Simons et al., 2023; Geffner et al., 2023). In our experiments, we uniformly at random sample either the masks for the joint, the posterior, the likelihood, or two randomly sampled masks (details in Appendix Sec. A2). To focus on specific conditional distributions, one can simply change the distribution of condition masks.

The Simformer uses learnable vector embeddings for identifiers and condition states, as proposed in Weilbach et al. (2023). In cases where parameters or data are functions

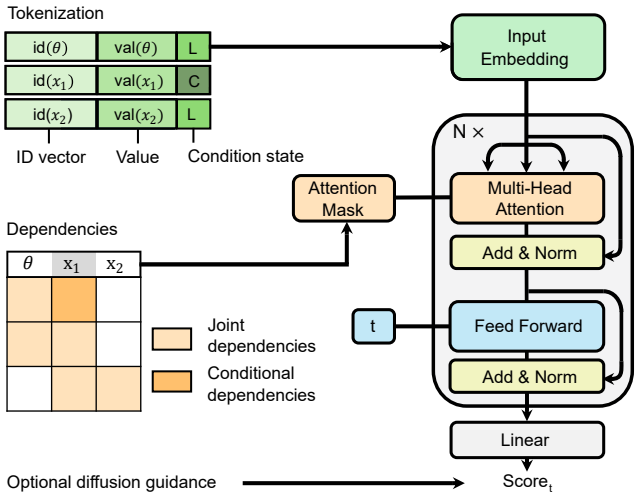

Figure 2. Simformer architecture. All variables (parameters and data) are reduced to a token representation which includes the variables' identity, the variables' value (val) as well as the conditional state (latent (L) or conditioned (C)). This sequence of tokens is processed by a transformer model; the interaction of variables can be explicitly controlled through an attention mask. The transformer architecture returns a score that is used to generate samples from the score-based diffusion model and can be modified (e.g. to guide the diffusion process).

of space or time, the node identifier will comprise a shared embedding vector and a random Fourier embedding of the elements in the index set. Finally, specialized embedding networks, commonly used in SBI algorithms and trained end-to-end (Lueckmann et al., 2017; Chan et al., 2018; Radev et al., 2020), can be efficiently integrated here by condensing complex data into a single token (e.g. we demonstrate this on a gravitational waves example in Appendix Sec. A3.2). This reduces computational complexity but loses direct control over dependencies and condition states for individual data elements.

3.2. Modelling dependency structures

For some simulators, domain scientists may have knowledge of (or assumptions about) the conditional dependency structures between parameters and data. For example, it may be known that all parameters are independent, or each parameter might only influence a single data value. The Simformer can exploit these dependencies by representing them in the attention mask M_E of the transformer (Weilbach et al., 2023). These constraints can be implemented as undirected (via a symmetric attention mask) or as directed dependencies (via a non-symmetric attention mask), that allow to enforce causal relations between parameters and observations. The latter, however, requires updating the mask if dependencies change i.e., due to conditioning (Webb et al., 2018) (Fig. 2, Appendix Sec. A1.1).

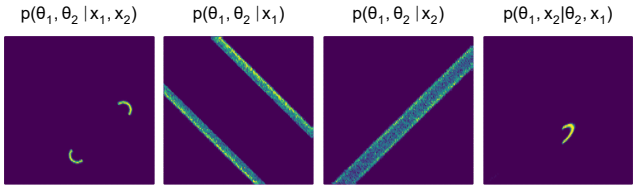

Figure 3. Examples of arbitrary conditional distributions of the Two Moons simulator, estimated by the Simformer.

A key advantage over masking weights directly (Germain et al., 2015) is that the attention mask can be easily dynamically adapted at train or inference time, allowing to enforce dependency structures that are dependent on input values and condition state (details in Appendix Sec. A1). We note that the attention mask M_E alone generally cannot ensure specific conditional independencies and marginalization properties in multi-layer transformer models. We describe the properties that can be reliably guaranteed and also explore how M_E can be effectively employed to learn certain desired properties in Appendix Sec. A1.

3.3. Simformer training and sampling

Having defined the tokenizer which processes every $(\boldsymbol{\theta}, \boldsymbol{x})$ pair and the attention mask to specify dependencies within the simulator, the Simformer can be trained using denoising score-matching (Hyvärinen & Dayan, 2005; Song et al., 2021b): We sample the noise level t for the diffusion model uniformly at random and generate a (partially) noisy sample $\hat{\mathbf{x}}_t^{M_C} = (1 - M_C) \cdot \hat{\mathbf{x}}_t + M_C \cdot \hat{\mathbf{x}}_0$ i.e. variables that we want to condition on remain clean. The loss can then be defined as

$$\ell(\phi, M_C, t, \hat{\mathbf{x}}_0, \hat{\mathbf{x}}_t) = \\ (1 - M_C) \cdot \left(s_\phi^{M_E}(\hat{\mathbf{x}}_t^{M_C}, t) - \nabla_{\hat{\mathbf{x}}_t} \log p_t(\hat{\mathbf{x}}_t | \hat{\mathbf{x}}_0) \right),$$

where $s_\phi^{M_E}$ denotes the score model equipped with a specific attention mask M_E. In expectation across noise levels t and the data, this results in

$$\mathcal{L}(\phi) = \mathbb{E}_{M_C, t, \hat{\mathbf{x}}_0, \hat{\mathbf{x}}_t}[\|\ell(\phi, M_C, t, \hat{\mathbf{x}}_0, \hat{\mathbf{x}}_t)\|_2^2].$$

We note that to simplify notation, M_E remains fixed here, but as stated in Sec. 3.2, it might depend on the condition state or input.

After having trained the Simformer, it can straightforwardly sample arbitrary conditionals (Fig. 3). We draw samples from the noise distribution and run the reverse diffusion process on all unobserved variables, while keeping observed variables constant at their conditioning value (Weilbach et al., 2023). Having access to all conditional distributions also allows us to combine scores and thereby perform inference for simulators with i.i.d. datapoints (Geffner et al., 2023). Similarly, we can use other score transformations to adapt to other prior or likelihood configurations post-hoc (see Appendix Sec. A1.4).

3.4. Conditioning on intervals with diffusion guidance

Guided diffusion makes it possible to sample from the generative model with an additional context $\mathbf{y}$, and has been used in tasks such as image inpainting, super-resolution, and image deblurring (Song et al., 2021b; Chung et al., 2022). It modifies the backward diffusion process to align it with a given context $\mathbf{y}$. Guided diffusion modifies the estimated score as

$$s(\hat{\boldsymbol{x}}_t, t | \mathbf{y}) \approx s_\phi(\hat{\boldsymbol{x}}_t, t) + \nabla_{\hat{\boldsymbol{x}}_t} \log p_t(\mathbf{y} | \hat{\boldsymbol{x}}_t).$$

Various strategies for guiding the diffusion process have been developed, mainly differing in how they estimate $\nabla_{\hat{\boldsymbol{x}}_t} \log p_t(\mathbf{y} | \hat{\boldsymbol{x}}_t)$ (Dhariwal & Nichol, 2021; Chung et al., 2023; Jalal et al., 2021; Song et al., 2022; Chung et al., 2022; Bansal et al., 2023; Lugmayr et al., 2022).

We here use diffusion guidance to be able to allow the Simformer to not only condition on fixed observations, but also on observation *intervals* (or, similarly, intervals of the prior). Bansal et al. (2023) demonstrated that diffusion models can be guided by arbitrary functions. In that line, we use the following general formulation to guide the diffusion process:

$$s_\phi(\hat{\boldsymbol{x}}_t, t | c) \approx s_\phi(\hat{\boldsymbol{x}}_t, t) + \nabla_{\hat{\boldsymbol{x}}_t} \log \sigma(-s(t) c(\hat{\boldsymbol{x}}_t))$$

Here σ denotes the sigmoid function, $s(t)$ is an appropriate scaling function satisfying $s(t) \to \infty$ as $t \to 0$, depending on the choice of SDE, and c denotes a constraint function $c(\hat{\boldsymbol{x}}) \leq 0$. For example, to enforce an interval upper bound u, we use $c(\hat{\boldsymbol{x}}) = \hat{\boldsymbol{x}} - u$. We detail the algorithm used for guiding the diffusion process in Alg. 1.

4. Results

4.1. Benchmark tasks

We evaluated performance in approximating posterior distributions across four benchmark tasks (Lueckmann et al., 2021). For each task, samples for ten ground-truth posteriors are available (Appendix Sec. A2.2), and we assessed performance as classifier two-sample test (C2ST) accuracy to these samples. Here, a score of 0.5 signifies perfect alignment with the ground truth posterior, and 1.0 indicates that a classifier can completely distinguish between the approximation and the ground truth. All results are obtained using the Variance Exploding SDE (VESDE); additional results using the Variance Preserving SDE (VPSDE) can be found in Appendix Sec. A3. See Appendix Sec. A2 for details on the parameterization.

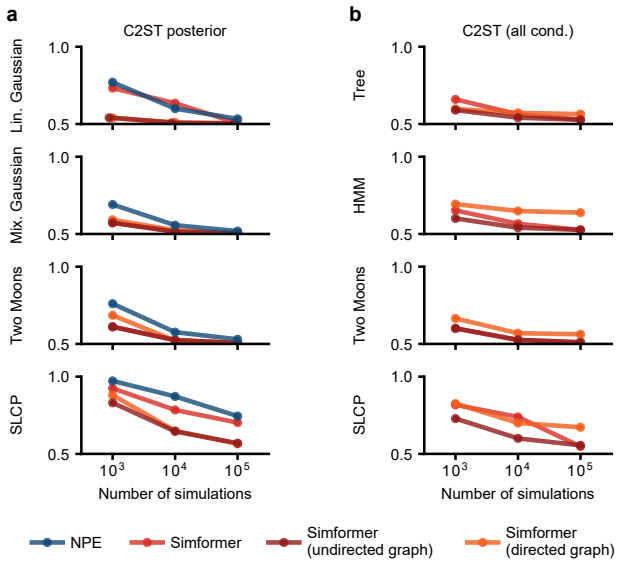

Figure 4. Simformer performance on benchmark tasks. The suffices "undirected graph" and "directed graph" denote Simformer variants with structured attention based on the respective graphical models. **(a)** Classifier Two-Sample Test (C2ST) accuracy between Simformer- and ground-truth posteriors. **(b)** C2ST between arbitrary Simformer-conditional distributions and their ground truth.

Across all four benchmark tasks, the Simformer outperformed neural posterior estimation (NPE), even when the Simformer used a dense attention mask (Fig. 4a). The only exception was the Gaussian linear task with 10k simulations; we show an extended comparison with NRE and NLE in Appendix Fig. A5, results with VPSDE in Appendix Fig. A6). Incorporating domain knowledge into the attention mask of the transformer led to further improvements in the accuracy of the Simformer, particularly in tasks with sparser dependency structures, such as the Linear Gaussian (fully factorized) and SLCP (4 i.i.d. observations). Averaged across all benchmark tasks and observations, the Simformer required about 10 times fewer simulations than NPE, leading to a vast reduction of computational cost for amortized inference.

Next, we evaluated the ability of the Simformer to evaluate arbitrary conditionals. Arbitrary parameter and data conditions often vastly differ from the form of the posterior distribution, leading to a challenging inference task (Fig. 3). We performed inference on two of the benchmark tasks and established two new tasks with particularly interesting dependencies (Tree and HMM, details in Appendix Sec. A2.2). For each of the tasks, we generated ground truth posterior samples with Markov-Chain Monte-Carlo on 100 randomly selected conditional or full joint distributions. We found that, despite the complexity of these tasks, Simformer was able to accurately model all conditionals across all tasks (Fig. 4b). We note that training solely on the posterior mask does not enhance performance relative to learning all conditional distributions (Appendix Sec. A3). Further, Simformer is well calibrated (Appendix Fig. A9, Fig. A10, Fig. A11, Fig. A12) and, in most cases, also superior with respect to the loglikelihood (Appendix Fig. A8).

4.2. Lotka-Volterra: Inference with unstructured observations

Many measurements in science are made in an unstructured way. For example, measurements of the populations of prey and predator species in ecology might not always be made at the same time points, and even the number of observations that were made might differ between species. To demonstrate that Simformer can deal with such 'unstructured' datasets, we applied the method to the ecological Lotka-Volterra model (Lotka, 1925; Volterra, 1926). The Lotka-Volterra model is a classic representation of predator-prey dynamics and is characterized by four global parameters, which govern the growth, hunting, and death rates of prey and predator. These populations evolve over time, guided by a set of coupled ordinary differential equations with Gaussian observation noise (details in Sec. A2.2). We

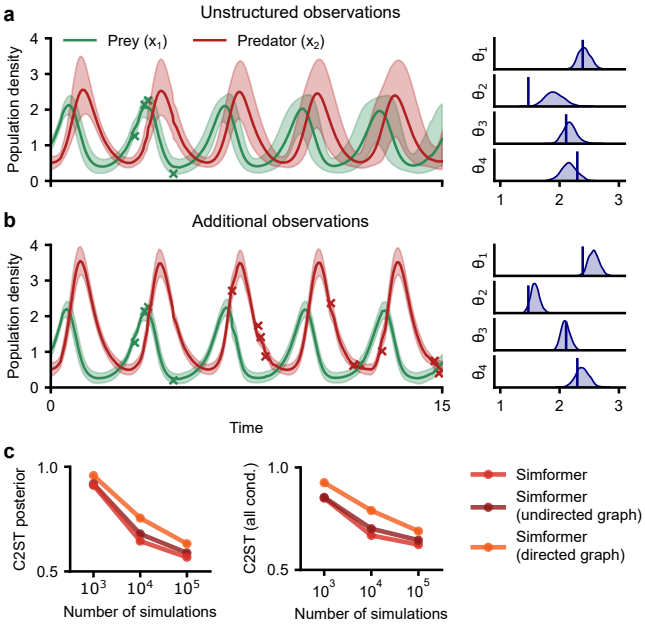

Figure 5. Inference with unstructured observations in the Lotka-Volterra model. **(a)** Posterior predictive (left) and posterior distribution (right) based on four unstructured observations of the prey population density (green crosses), using Simformer with 10^5 simulations. True parameters in dark blue. **(b)** Same as (a) with nine additional observations of the predator population density. **(c)** C2ST-performance in estimating arbitrary conditionals (right) or the posterior distribution (left) using the C2ST metric.

note that, unlike Lueckmann et al. (2021), we perform inference for the *full* time-series and do not rely on summary statistics.

We trained Simformer on 10^5 simulations and, after training, generated several synthetic observations. The first of these observations contained four measurements of the prey population, placed irregularly in time (green crosses in Fig. 5a).

Using Simformer, we inferred the posterior distribution given this data. We found that the ground truth parameter set was indeed within regions of high posterior probability, and the Simformer posterior closely matched the ground truth posterior generated with MCMC (Fig. 5c, Appendix Sec. A2.2). We then used the ability of Simformer to sample from arbitrary conditional distribution to simultaneously generate posterior and posterior predictive samples without additional runs of the simulator. The posterior predictives of Simformer capture data and uncertainty in a realistic manner (Fig. 5a).

As a second synthetic observation scenario, we used nine additional observations of the predator population, also irregularly placed in time (Fig. 5b). As expected, including these measurements reduces the uncertainty in both the posterior (Fig. 5b, right) and posterior predictive distributions (Fig. 5b left, posterior predictive again generated by the Simformer).

4.3. SIRD-model: Inference in infinite dimensional parameters

Next, we show that Simformer can perform inference on functional data, i.e., ∞-dimensional parameter spaces, and that it can incorporate measurements of a subset of parameters into the inference process. In many simulators, parameters of the system may depend on time or space, and amortized inference methods should allow to perform parameter inference at *any* (potentially infinitely many) points in time or space. We will demonstrate the ability of Simformer to solve such inference tasks in an example from epidemiology, the Susceptible-Infected-Recovered-Deceased (SIRD) model (Kermack & McKendrick, 1927).

The SIRD simulator has three parameters: recovery rate, death rate, and contact rate. To simplify the inference task, these parameters are sometimes assumed to be constant in time, but treating the parameters as time-dependent allows to incorporate factors such as social distancing measures, public health interventions, and natural changes in human behavior (Chen et al., 2020; Schmidt et al., 2021). This is in contrast to Lueckmann et al. (2021), which only considered a two-parameter SIR variant on a discrete-time grid. To demonstrate that Simformer can deal with a mixture of time-dependent and constant-in-time parameters, we assumed that the contact rate varied over time, whereas the recovery and death rate where constant in time.

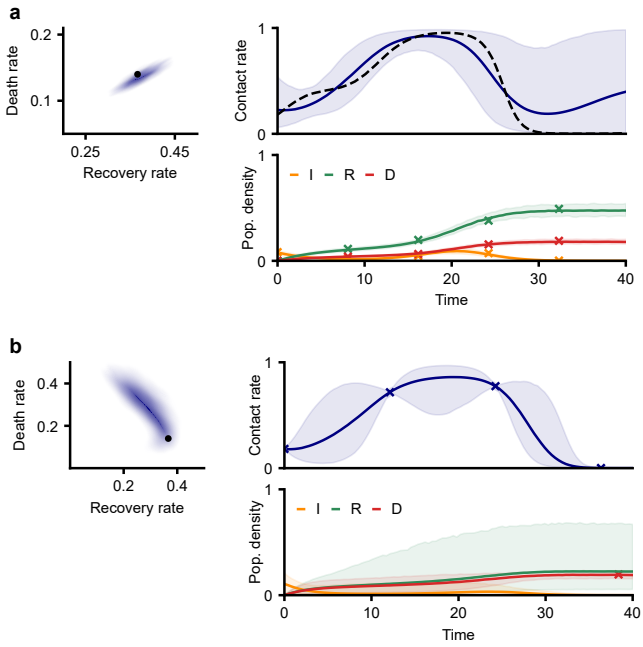

Figure 6. Inference of ∞-dim parameter space in the SIRD model. **(a)** Inferred posterior for global parameters (upper left) and time-dependent local parameters (upper right) based on five observations (crosses) of infected (I), recovered (R), and death (D) population densities. The black dot and dashed line indicate the true parameter, bold lines indicate the mean, and shaded areas represent 99% quantiles. **(b)** Inference with parameter measurements and a single measurement of fatalities.

We generated synthetic measurements from infected, recovered, and deceased individuals at irregularly spaced time points and applied the Simformer to estimate the posterior distribution of parameters. The Simformer estimated realistic death and recovery rates and successfully recovers a time-dependent contact rate that aligns with ground truth observations (Fig. 6a). Indeed, as measurements of infections tend towards zero (around timestamp 25, Fig. 6a, orange), the Simformer-posterior for the contact rate increases its uncertainty. This is expected, as we cannot obtain conclusive insights about the contact rate in scenarios with negligible infections. Additionally, as we already demonstrated on the Lotka-Volterra task, the ability of the Simformer to sample any conditional distribution allows us to generate posterior predictive samples without running the simulator. These samples closely match the observed data, further demonstrating the accuracy of the Simformer.

Next, we demonstrate that the Simformer can accurately sample parameter-conditioned posterior distributions (Fig. 6b). We generated a synthetic observation consisting of four measurements of the time-dependent contact rate

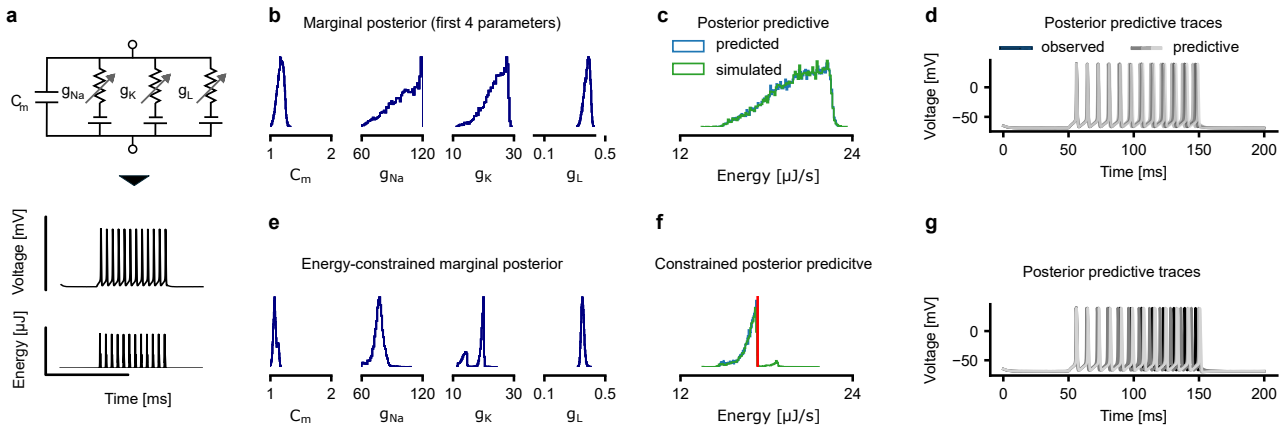

Figure 7. Inference in the Hodgkin-Huxley model. **(a)** Model schematic, observed voltage trace, and associated energy consumption. **(b)** Marginals of inferred posterior for four parameters. **(c)** Posterior predictive energy consumption from Simformer (blue) and from simulator outputs (green). **(d)** Posterior predictive samples from the posterior in (c) using the simulator. **(e)** Marginals of inferred energy constrained posterior for four parameters. **(f)** Posterior predictive energy consumption from Simformer (blue) and from simulator outputs (green). Energy constraint as red line. **(g)** Posterior predictive samples from posterior in (e) using the simulator.

parameter and a single measurement of infected people. The resulting Simformer-posterior closely aligns with the parameter measurements, and its posterior predictives are aligned with the data. We evaluate the performance quantitatively by computing the expected coverage, which verified that the conditional distributions estimated by Simformer are indeed well-calibrated (see Fig. A13).

Overall, these results demonstrate that the Simformer can tackle function-valued parameter spaces and that its ability to sample arbitrary conditionals allows the incorporation of parameter measurements or assumptions into the inference procedure.

4.4. Hodgkin-Huxley model: Inference with observation intervals

Finally, we demonstrate that the Simformer can perform inference in a highly nonlinear model and that it can constrain the parameters to observation *intervals* with guided diffusion. For example, in neuroscience, it is desirable to obtain parameter configurations conditioned to experimental voltage measurements but also restricted by constraints such as lowering the metabolic cost (energy) below a particular threshold. Such additional constraints can be formalized as observation *intervals*.

We demonstrate the ability of Simformer to perform such inferences in an example from neuroscience, the Hodgkin-Huxley simulator (Hodgkin & Huxley, 1952). This simulator describes the time course of voltage along the membrane of neurons (Fig. 7a). The simulator has 7 parameters and generates a noisy time series, which we reduced to summary statistics as in previous work (Gonçalves et al., 2020). In addition, we also record the metabolic cost consumed by the circuit and add it as an additional statistic (Appendix Sec. A2.2).

We first inferred the posterior distribution given only the summary statistics of the voltage (not the energy) with the Simformer, and we found that, consistent with prior work (Gonçalves et al., 2020), the posterior distribution has wide marginals for some parameters and narrow marginals for others (Fig. 7b). We then used Simformer's ability to sample arbitrary conditionals and generate posterior predictives for energy consumption (Fig. 7c). The posterior predictive distribution of Simformer closely matched the posterior predictive distribution obtained by running the simulator (Fig. 7cd), and the energy cost of different posterior samples varied significantly (Deistler et al., 2022b).

To identify energy-efficient parameter sets, we then defined an observation *interval* for the energy consumption (energy must be within the lowest 10% quantile of posterior predictives), and we used Simformer with guided diffusion to infer the posterior given voltage summary statistics and this constraint on energy consumption. The additional constraint on energy consumption significantly constrained the parameters posterior, in particular the maximal sodium and potassium conductances (Fig. 7e). We generated posterior predictive samples from this new posterior (via Simformer and by running the simulation) and found that their energy consumption indeed lies below the desired threshold (Fig. 7f). Furthermore, the corresponding predictive voltage trace is still in agreement with observations (Fig. 7g). Additional details and results on guidance are in Appendix

Sec. A3.3 (e.g. Fig. A15 for benchmarks on the accuracy of guidance).

Overall, Simformer can successfully recover the posterior distribution of highly nonlinear simulators. Simformer can condition on exact observations but also, using guided diffusion, on nearly arbitrary constraints (see Appendix Fig. A3, Fig A16).

5. Discussion

We developed the Simformer, a new method for simulation-based amortized inference. The Simformer outperforms previous state-of-the-art methods (NPE) for posterior inference and simultaneously estimates all other conditionals. On tasks with notable independent structures, Simformer can be (on average across tasks and observations), one order of magnitude more simulation-efficient if equipped with a proper attention mask. The Simformer is significantly more flexible than previous out-of-the box inference frameworks and allows us to perform inference in ∞-dimensional parameter spaces, on unstructured and missing data. The Simformer makes it possible to sample arbitrary (or specified) conditional distributions of the joint distribution of parameters and data, including posterior and likelihood, thereby providing an 'all-in-one' inference method. These conditional distributions can be used to perform inference with parameter conditionals, or to obtain posterior predictive samples without running the simulator. Using diffusion guidance, one can also condition on intervals, which, e.g., can be used to modify the prior without the need for retraining. Overall, the Simformer is an accurate and highly flexible inference method that opens up new possibilities for amortized inference methods in science and engineering.

Related Work The Simformer is designed to solve a range of problems in simulation-based inference, but its backbone, a probabilistic diffusion model on top of a transformer architecture, has also been used for generative models of images (Peebles & Xie, 2022; Hatamizadeh et al., 2023), and the task of generating arbitrary conditionals has been explored in various other generative models (Ivanov et al., 2019; Li et al., 2020; Strauss & Oliva, 2021; 2022). In addition, integrating structural knowledge about the inference tasks has been previously explored for discrete diffusion models or continuous normalizing flows (Weilbach et al., 2020; Harvey et al., 2022; Weilbach et al., 2023) and has also been explored for neural processes and meta-learning (Nguyen & Grover, 2022a;b; Müller et al., 2023; Maraval et al., 2023).

The benefits of diffusion models for simulation-based inference have also been explored: Simons et al. (2023) demonstrated that diffusion models can improve inference performance, and Geffner et al. (2023) showed that score decomposition can be used to perform inference for i.i.d. data. The usage of diffusion models in the Simformer inherits these benefits. Wildberger et al. (2023) demonstrated that flow-matching can largely reduce the number of trainable parameters needed for accurate inference results. Schmitt et al. (2023) proposed multi-head attention for integrating heterogeneous data from diverse sources. Rozet & Louppe (2023) use a score-based model to learn the joint distribution of a dynamical system, approximately restricting their network to the Markovian structure, and then use guidance to condition it on specific observations.

The Simformer overcomes many limitations of current amortized inference methods, several of which have previously been tackled separately: First, Chen et al. (2020); Ramesh et al. (2022); Moss et al. (2023) also estimated posteriors over parameters that depended on space, but they relied on predefined discretizations to do so. Second, Dyer et al. (2021) inferred the posterior distribution for irregularly sampled time series via approximate Bayesian computation, and Radev et al. (2020) amortized inference across a flexible number of i.i.d. trials (without considering irregularly sampled data). Third, Wang et al. (2023) proposed an approach to infer the posterior when data is missing, achieved through data augmentation and employment of recurrent neural networks. Forth, whereas the Simformer inherently returns likelihood, posterior, and all other conditionals, Radev et al. (2023) and Glöckler et al. (2022) learned separate networks for the likelihood and posterior and investigated features unlocked by having access to both distributions, and Deistler et al. (2022b) used MCMC to sample parameter conditionals of the learned posterior. Finally, Rozet & Louppe (2021) proposed to estimate arbitrary marginal distributions for neural ratio estimation, whereas the Simformer can be used to estimate all conditional distributions. All of the above works tackle the respective problem in isolation, whereas the architecture of the Simformer allows us to overcome all of these limitations at once.

Limitations Our method inherits the limitations of transformers and diffusion models: Generating samples is slower than for NPE, which is typically based on normalizing flows that permit fast sampling (Greenberg et al., 2019), whereas we have to solve the reverse SDE. On the other hand, sampling is much faster than methods that rely on MCMC (Papamakarios et al., 2019; Hermans et al., 2020). In our experiments, accurate inference is achievable with as few as 50 evaluation steps, leading to sampling times of a few seconds for 10k samples. Further improvements may be possible by adapting the model (Song et al., 2021a), the underlying SDE (Albergo et al., 2023) or SDE solver for sampling (Gonzalez et al., 2023).

Moreover, unlike normalizing flows, transformer evaluations scale quadratically with the number of input tokens, presenting significant memory and computational chal-

lenges during training. To mitigate this, various strategies have been proposed (Lin et al., 2022). Naturally, using a sparse attention mask (e.g. due to many independencies) can reduce computational complexity (Jaszczur et al., 2021; Weilbach et al., 2023).

In this work, we focus on estimating all conditionals, a task that, within our framework, is roughly as complex as learning the joint distribution. In problems with a few parameters but high dimensional data (i.e. images or long time series), estimating the joint might be harder than just the posterior. In such cases, Simformer can simply be queried to target specific conditionals of interest (e.g., posterior and missing data posteriors, see Appendix Sec. A3.2 for an example on gravitational waves).

Lastly, normalizing flows enable rapid and precise assessments of the log-probability for posterior (or likelihood) approximations. This efficiency facilitates their integration into MCMC frameworks and aids the computation of point estimates, such as the Maximum A Posteriori (MAP) estimate. The score-based diffusion model employed by the Simformer also allows to evaluate log-probabilities (of any conditional of the joint), but this requires solving the corresponding probability flow ODE, which presents a computational burden (Song et al., 2021b). Fortunately, for tasks such as MAP computation or integrating the Simformer likelihood into an MCMC scheme, there's no need to frequently assess log-probabilities. Instead, the score function can be utilized for gradient ascent to optimize the MAP or to perform Langevin-MCMC sampling, seamlessly incorporating the Simformer likelihood with such MCMC methods.

Conclusion We developed the Simformer, a new method for amortized simulation-based inference. On benchmark tasks, it performs at least as well or better as existing methods that only target the posterior, although the Simformer estimates all conditional distributions. The Simformer is highly flexible and can jointly tackle multiple amortized inference tasks more effectively than previous methods.

Software and Data

We used JAX (Bradbury et al., 2018) as backbone and hydra (Yadan, 2019) to track all configurations. Code to reproduce results is available at `https://github.com/mackelab/simformer`. We use SBI (Tejero-Cantero et al., 2020) for reference implementations of baselines.

Impact Statement

Simulation-based inference (SBI) holds immense promise for advancing science across various disciplines. Our work enhances the accuracy and flexibility of SBI, thereby allowing scientists to apply SBI to previously unattainable simulators and inference problems. However, it is crucial to acknowledge the potential for the application of our method in less desirable contexts. Careful consideration of ethical implications is necessary to ensure the responsible use of our method.

Acknowledgements

This work was supported by the German Research Foundation (DFG) through Germany's Excellence Strategy – EXC-Number 2064/1 – Project number 390727645, the German Federal Ministry of Education and Research (Tübingen AI Center, FKZ: 01IS18039A), the 'Certification and Foundations of Safe Machine Learning Systems in Healthcare' project funded by the Carl Zeiss Foundation, and the European Union (ERC, DeepCoMechTome, 101089288). MG and MD are members of the International Max Planck Research School for Intelligent Systems (IMPRS-IS). CW and FW acknowledge the support of the Natural Sciences and Engineering Research Council of Canada (NSERC), the Canada CIFAR AI Chairs Program, Inverted AI, MITACS, the Department of Energy through Lawrence Berkeley National Laboratory, and Google. This research was enabled in part by technical support and computational resources provided by the Digital Research Alliance of Canada Compute Canada (alliancecan.ca), the Advanced Research Computing at the University of British Columbia (arc.ubc.ca), Amazon, and Oracle.

References

Albergo, M. S., Boffi, N. M., and Vanden-Eijnden, E. Stochastic interpolants: A unifying framework for flows and diffusions. *arXiv preprint arXiv:2303.08797*, 2023.

Anderson, B. D. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12(3):313–326, 1982.

Bansal, A., Chu, H.-M., Schwarzschild, A., Sengupta, S., Goldblum, M., Geiping, J., and Goldstein, T. Universal guidance for diffusion models. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 843–852, 2023. doi: 10.1109/CVPRW59228.2023.00091.

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

Boelts, J., Lueckmann, J.-M., Gao, R., and Macke, J. H. Flexible and efficient simulation-based inference for models of decision-making. *Elife*, 11:e77220, 2022.

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J.,

Wanderman-Milne, S., and Zhang, Q. *JAX: composable transformations of Python+NumPy programs*, 2018.

Chan, J., Perrone, V., Spence, J. P., Jenkins, P. A., Mathieson, S., and Song, Y. S. A Likelihood-Free inference framework for population genetic data using exchangeable neural networks. *Adv Neural Inf Process Syst*, 31: 8594–8605, December 2018.

Chen, Y.-C., Lu, P.-E., Chang, C.-S., and Liu, T.-H. A time-dependent sir model for covid-19 with undetectable infected persons. *IEEE Transactions on Network Science and Engineering*, 7(4):3279–3294, October 2020. ISSN 2334-329X. doi: 10.1109/tnse.2020.3024723.

Chung, H., Sim, B., Ryu, D., and Ye, J. C. Improving diffusion models for inverse problems using manifold constraints. *Advances in Neural Information Processing Systems*, 35:25683–25696, 2022.

Chung, H., Kim, J., Mccann, M. T., Klasky, M. L., and Ye, J. C. Diffusion posterior sampling for general noisy inverse problems. In *The Eleventh International Conference on Learning Representations*, 2023.

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

Dax, M., Green, S. R., Gair, J., Macke, J. H., Buonanno, A., and Schölkopf, B. Real-time gravitational wave science with neural posterior estimation. *Phys. Rev. Lett.*, 127:241103, Dec 2021. doi: 10.1103/PhysRevLett.127.241103.

Deistler, M., Goncalves, P. J., and Macke, J. H. Truncated proposals for scalable and hassle-free simulation-based inference. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022a.

Deistler, M., Macke, J. H., and Gonçalves, P. J. Energy-efficient network activity from disparate circuit parameters. *Proceedings of the National Academy of Sciences*, 119(44):e2207632119, 2022b.

Dhariwal, P. and Nichol, A. Diffusion models beat gans on image synthesis. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 8780–8794. Curran Associates, Inc., 2021.

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

Dyer, J., Cannon, P., and Schmon, S. M. Approximate bayesian computation with path signatures. *arXiv preprint arXiv:2106.12555*, 2021.

Elsemüller, L., Olischläger, H., Schmitt, M., Bürkner, P.-C., Köthe, U., and Radev, S. T. Sensitivity-aware amortized bayesian inference. *arXiv preprint arXiv:2310.11122*, 2023.

Geffner, T., Papamakarios, G., and Mnih, A. Compositional score modeling for simulation-based inference. In *International Conference on Machine Learning*, pp. 11098–11116. PMLR, 2023.

Germain, M., Gregor, K., Murray, I., and Larochelle, H. Made: Masked autoencoder for distribution estimation. In Bach, F. and Blei, D. (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 881–889, Lille, France, 07–09 Jul 2015. PMLR.

Glöckler, M., Deistler, M., and Macke, J. H. Variational methods for simulation-based inference. In *International Conference on Learning Representations*, 2022.

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

Gonzalez, M., Fernandez, N., Tran, T., Gherbi, E., Hajri, H., and Masmoudi, N. Seeds: Exponential sde solvers for fast high-quality sampling from diffusion models, 2023.

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

Gu, Y., Wang, X., Ge, Y., Shan, Y., Qie, X., and Shou, M. Z. Rethinking the objectives of vector-quantized tokenizers for image synthesis. *arXiv preprint arXiv:2212.03185*, 2022.

Harvey, W., Naderiparizi, S., Masrani, V., Weilbach, C., and Wood, F. Flexible diffusion modeling of long videos. *Advances in Neural Information Processing Systems*, 35: 27953–27965, 2022.

Hatamizadeh, A., Song, J., Liu, G., Kautz, J., and Vahdat, A. Diffit: Diffusion vision transformers for image generation, 2023.

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

Hermans, J., Delaunoy, A., Rozet, F., Wehenkel, A., Begy, V., and Louppe, G. A trust crisis in simulation-based inference? your posterior approximations can be unfaithful, 2022.

Ho, J. and Salimans, T. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021.

Hodgkin, A. L. and Huxley, A. F. A quantitative description of membrane current and its application to conduction and excitation in nerve. *J Physiol*, 117(4):500–544, Aug 1952. doi: 10.1113/jphysiol.1952.sp004764.

Hyvärinen, A. and Dayan, P. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(4), 2005.

Ivanov, O., Figurnov, M., and Vetrov, D. Variational autoencoder with arbitrary conditioning. In *International Conference on Learning Representations*, 2019.

Jalal, A., Arvinte, M., Daras, G., Price, E., Dimakis, A. G., and Tamir, J. Robust compressed sensing mri with deep generative priors. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 14938–14954. Curran Associates, Inc., 2021.

Jaszczur, S., Chowdhery, A., Mohiuddin, A., Łukasz Kaiser, Gajewski, W., Michalewski, H., and Kanerva, J. Sparse is enough in scaling transformers, 2021.

Kermack, W. O. and McKendrick, A. G. A contribution to the mathematical theory of epidemics. *Proceedings of the royal society of london. Series A, Containing papers of a mathematical and physical character*, 115(772):700–721, 1927.

Le, T. A., Baydin, A. G., and Wood, F. Inference compilation and universal probabilistic programming. In *Artificial Intelligence and Statistics*, pp. 1338–1348. PMLR, 2017.

Li, Y., Akbar, S., and Oliva, J. Acflow: Flow models for arbitrary conditional likelihoods. In *International Conference on Machine Learning*, pp. 5831–5841. PMLR, 2020.

Lin, T., Wang, Y., Liu, X., and Qiu, X. A survey of transformers. *AI Open*, 2022.

Lotka, A. J. *Elements of physical biology*. Williams & Wilkins, 1925.

Lueckmann, J.-M., Goncalves, P. J., Bassetto, G., Öcal, K., Nonnenmacher, M., and Macke, J. H. Flexible statistical inference for mechanistic models of neural dynamics. *Advances in neural information processing systems*, 30, 2017.

Lueckmann, J.-M., Boelts, J., Greenberg, D., Goncalves, P., and Macke, J. Benchmarking simulation-based inference. In *International Conference on Artificial Intelligence and Statistics*, pp. 343–351. PMLR, 2021.

Lugmayr, A., Danelljan, M., Romero, A., Yu, F., Timofte, R., and Van Gool, L. Repaint: Inpainting using denoising diffusion probabilistic models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11461–11471, 2022.

Maathuis, M., Drton, M., Lauritzen, S., and Wainwright, M. *Handbook of graphical models*. CRC Press, 2018.

Maraval, A., Zimmer, M., Grosnit, A., and Ammar, H. B. End-to-end meta-bayesian optimisation with transformer neural processes. *arXiv preprint arXiv:2305.15930*, 2023.

Marlier, N., Brüls, O., and Louppe, G. Simulation-based bayesian inference for robotic grasping. In *IROS 2022 Workshop Probabilistic Robotics in the Age of Deep Learning*, 2022.

Moss, G., Višnjević, V., Eisen, O., Oraschewski, F. M., Schröder, C., Macke, J. H., and Drews, R. Simulation-based inference of surface accumulation and basal melt rates of an antarctic ice shelf from isochronal layers, 2023.

Müller, S., Hollmann, N., Arango, S. P., Grabocka, J., and Hutter, F. Transformers can do bayesian inference, 2023.

Nguyen, T. and Grover, A. Transformer neural processes: Uncertainty-aware meta learning via sequence modeling. *arXiv preprint arXiv:2207.04179*, 2022a.

Nguyen, T. and Grover, A. Transformer neural processes: Uncertainty-aware meta learning via sequence modeling. *arXiv preprint arXiv:2207.04179*, 2022b.

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

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

Peebles, W. and Xie, S. Scalable diffusion models with transformers. *arXiv preprint arXiv:2212.09748*, 2022.

Pospischil, M., Toledo-Rodriguez, M., Monier, C., Piwkowska, Z., Bal, T., Frégnac, Y., Markram, H., and Destexhe, A. Minimal hodgkin–huxley type models for different classes of cortical and thalamic neurons. *Biological cybernetics*, 99:427–441, 2008.

Radev, S. T., Mertens, U. K., Voss, A., Ardizzone, L., and Köthe, U. Bayesflow: Learning complex stochastic models with invertible neural networks. *IEEE transactions on neural networks and learning systems*, 33(4):1452–1466, 2020.

Radev, S. T., Schmitt, M., Pratz, V., Picchini, U., Köthe, U., and Bürkner, P.-C. Jana: Jointly amortized neural approximation of complex Bayesian models. In Evans, R. J. and Shpitser, I. (eds.), *Proceedings of the Thirty-Ninth Conference on Uncertainty in Artificial Intelligence*, volume 216 of *Proceedings of Machine Learning Research*, pp. 1695–1706. PMLR, 31 Jul–04 Aug 2023.

Ramesh, P., Lueckmann, J.-M., Boelts, J., Tejero-Cantero, Á., Greenberg, D. S., Goncalves, P. J., and Macke, J. H. GATSBI: Generative adversarial training for simulation-based inference. In *International Conference on Learning Representations*, 2022.

Rozet, F. and Louppe, G. Arbitrary marginal neural ratio estimation for simulation-based inference. *arXiv preprint arXiv:2110.00449*, 2021.

Rozet, F. and Louppe, G. Score-based data assimilation. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Schmidt, J., Krämer, N., and Hennig, P. A probabilistic state space model for joint inference from differential equations and data. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 12374–12385. Curran Associates, Inc., 2021.

Schmitt, M., Radev, S. T., and Bürkner, P.-C. Fuse it or lose it: Deep fusion for multimodal simulation-based inference, 2023.

Shukla, S. N. and Marlin, B. M. A survey on principles, models and methods for learning from irregularly sampled time series, 2021.

Simons, J., Sharrock, L., Liu, S., and Beaumont, M. Neural score estimation: Likelihood-free inference with conditional score based diffusion models. In *Fifth Symposium on Advances in Approximate Bayesian Inference*, 2023.

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

Song, J., Meng, C., and Ermon, S. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021a.

Song, J., Vahdat, A., Mardani, M., and Kautz, J. Pseudoinverse-guided diffusion models for inverse problems. In *International Conference on Learning Representations*, 2022.

Song, Y. and Ermon, S. Generative modeling by estimating gradients of the data distribution. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021b.

Strauss, R. and Oliva, J. B. Arbitrary conditional distributions with energy. *Advances in Neural Information Processing Systems*, 34:752–763, 2021.

Strauss, R. and Oliva, J. B. Posterior matching for arbitrary conditioning. *Advances in Neural Information Processing Systems*, 35:18088–18099, 2022.

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

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Volterra, V. Fluctuations in the abundance of a species considered mathematically. *Nature*, 118(2972):558–560, 1926.

Wang, Z., Hasenauer, J., and Schälte, Y. Missing data in amortized simulation-based neural posterior estimation. *bioRxiv*, 2023. doi: 10.1101/2023.01.09.523219.

Webb, S., Golinski, A., Zinkov, R., N, S., Rainforth, T., Teh, Y. W., and Wood, F. Faithful inversion of generative models for effective amortized inference. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

Weilbach, C., Beronov, B., Wood, F., and Harvey, W. Structured conditional continuous normalizing flows for efficient amortized inference in graphical models. In *International Conference on Artificial Intelligence and Statistics*, pp. 4441–4451. PMLR, 2020.

Weilbach, C. D., Harvey, W., and Wood, F. Graphically structured diffusion models. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J. (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 36887–36909. PMLR, 23–29 Jul 2023.

Wildberger, J. B., Dax, M., Buchholz, S., Green, S. R., Macke, J. H., and Schölkopf, B. Flow matching for scalable simulation-based inference. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Yadan, O. Hydra - a framework for elegantly configuring complex applications. Github, 2019.

Zhang, J. Causal reasoning with ancestral graphs. *Journal of Machine Learning Research*, 9(7), 2008.

Appendix

A1. Conditional and marginalization properties

In this section, we want to clarify what independence structures are exactly imposed by the Simformer equipped with a specific attention mask at the target distribution ($t = 0$) and intermediate marginals ($t > 0$) (Appendix Sec. A1.1). We further state what marginalization properties you can expect a priori and how to adapt the training procedure to additionally enforce certain marginalization constraints (Appendix Sec. A1.2). We then discuss how to extend to include post-hoc adaption of prior or likelihood (Appendix Sec. A1.3) and demonstrate the content on a toy example (Appendix Sec. A1.4).

A1.1. Conditional dependencies

We assume that the diffusion process (i.e. the underlying SDE) does not introduce any additional correlations, which is valid for VPSDE and VESDE. The attention mask, denoted by M_E, represents a graph $\mathcal{G}(\hat{\mathbf{x}}, M_E)$, with a total of N vertices. We assume that $p(\hat{\boldsymbol{x}})$ follows this graphical model. In this graph, if there exists a path from node $\hat{\mathbf{x}}_i$ to node $\hat{\mathbf{x}}_j$, then the transformer model $s_{\phi^*}^{M_E}$ is capable of attending $\hat{\mathbf{x}}_j$ to $\hat{\mathbf{x}}_i$, given it has enough layers. Conversely, the absence of such a path implies the transformer must estimate the score of $\hat{\mathbf{x}}_i$ independent of $\hat{\mathbf{x}}_j$. For an l-layer transformer, the matrix $D = \mathbb{I}(M_E^l > 0)$ succinctly represents all explicitly enforced conditional independencies, given a constant attention mask M_E. This is a classical result from graph theory i.e. that the n'th power of the adjacency matrix describes the number of walks from any node i to any node j. The i'th row of this matrix delineates the variables upon which $\hat{\mathbf{x}}_i$ can attend and, therefore, potentially depend (see Fig. A1a).

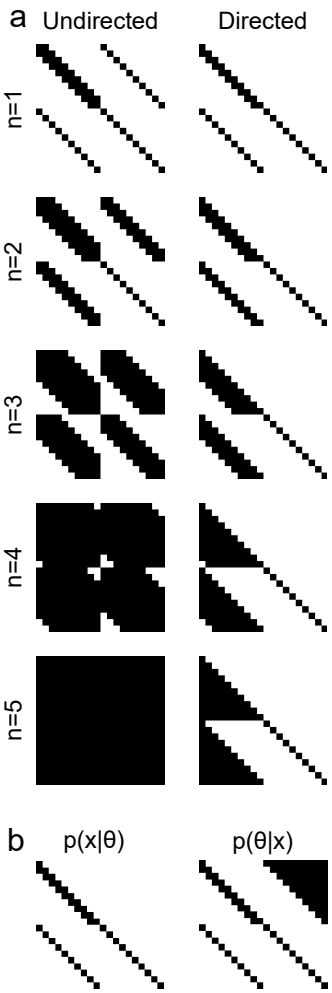

Figure A1. **(a)** Evolution of dependencies through $n = 1, \dots, 5$ transformer layers, given a constant attention mask for the HMM task ($n = 1$). **(b)** Necessary adaption of the directed attention mask to faithfully capture conditional dependencies.

Dependencies at $\mathbf{t = 0}$: For an undirected, connected graph, all variables can depend on each other (given l is large enough). This is a core argument by Weilbach et al. (2023) that an undirected graphical representation, given enough layers, is enough to faithfully represent all dependencies for any condition. Yet, this also diminishes any chance of correctly enforcing correct independencies beyond separating disconnected components. On the other hand, a directed acyclic graph will stay directed and acyclic. This property disallows modeling arbitrary dependencies, and this is why we have to dynamically adapt the mask to faithfully represent dependencies for arbitrary conditionals. We use the algorithm as proposed by Webb et al. (2018), which returns a minimal amount of edges we have to add to the directed graph to faithfully represent present dependencies (under certain topological ordering constraints). This is shown in Figure A1b. As expected for modeling the likelihood, no additional edges have to be introduced. On the other hand, to model the posterior distribution, we have to insert additional edges into the upper right corner. Note that this mask is sufficient to represent dependencies with a 1-layer transformer and thus adds too many edges in general. For Gaussian linear tasks, where M_E stands as an idempotent matrix (i.e. $M_E^2 = M_E$), resulting in $D = M_E$, this implies that all conditional independencies are correctly enforced, explaining the substantial enhancement in accuracy. Even if dependencies are not exactly enforced, as observed by both our work and Weilbach et al. (2023), structured masks can enhance performance and computational complexity, particularly in the presence of notable independence structures. It is important to note that these dependencies are what is enforced by the model, not what is necessarily learned.

Dependencies at $\mathbf{t > 0}$: The score estimator does target the score of $p_t(\hat{\boldsymbol{x}}_t) = \int p(\hat{\boldsymbol{x}}_t|\hat{\boldsymbol{x}})p(\hat{\boldsymbol{x}})d\hat{\boldsymbol{x}}$. Notably, the imposed graphical model $\mathcal{G}$ is assumed to be valid at $p(\hat{\boldsymbol{x}})$ but is generally invalid for $p_t(\hat{\boldsymbol{x}}_t)$. Directed graphical models are not closed under marginalization (beyond leave nodes) (Maathuis et al., 2018). Undirected graphical models are closed but become fully connected in the case of diffusion models (for each connected component) (Weilbach et al., 2020). As highlighted by Rozet & Louppe (2023), one rationale

for overlooking this concern is that for small values of t, indicating minimal noise, this assumption holds approximately true. Further, as t grows and noise accumulates, the mutual information between variables must decrease to zero by construction, implying that dependencies must be transformed from M_E at $t = 0$ to the identity mask I at $t = T$. As also discussed before, the actual constraints imposed on the transformer score model is D, which does have an increased "receptive field". For undirected graphical models, this can be seen as equivalent to the notion of "pseudo-markov blanckets" introduced in Rozet & Louppe (2023). Given enough layers, this is sufficient to model all $p_t(\hat{\boldsymbol{x}}_t)$ (Weilbach et al., 2023), at the cost of explicitly enforcing known constraints at $t = 0$. This is generally not true for the directed graphical model. It can faithfully represent all dependencies at time $t = 0$, but can not necessarily exactly represent it at time $t > 0$. Only if all connected components become autoregressive, it similarly can represent all dependencies. For further work, if it is desired to preserve the causal flow of information, it might be interesting to also consider more expressive graph representations. The class of ancestral graphs, for example, is closed under marginalization and can preserve the causal flow of information (Zhang, 2008).

A1.2. Marginalization Properties

Transformers, with their capability to process sequences of arbitrary lengths, present a compelling opportunity to exclude non-essential variables directly from the input. This is not merely a convenience but a method to reduce computational complexity, which is directly influenced by the length of the sequence. Therefore, omitting non-essential variables at the input stage is more efficient than removing them post hoc. Another unique ability, which is usually not possible for other models, is the possibility to compute marginal densities.

However, this selective exclusion comes with a specific prerequisite. The ability to drop variables is guaranteed only if, for any subset of variables $\{\hat{\mathbf{x}}_i, \hat{\mathbf{x}}_j, \dots\}$, the dependency matrix D satisfies $D_{ni} = 0, D_{nj} = 0, \dots$ for all $n \neq i, j$. In simpler terms, this means that this subset of variables should not be able to attend to any outside variables. When examining the mask depicted in Fig. A1, it becomes evident that for a transformer with five layers and an undirected mask, we cannot safely omit any of the variables. Conversely, with a directed mask in place, we are able to safely sample $p(\boldsymbol{\theta})$ (first 10 elements) independently from $p(\boldsymbol{x})$ (last 10 elements).

Particularly in cases where the dependency matrix D is densely populated, dropping out certain variables can change the output in an unexpected manner. This challenge can be addressed by training a transformer model to accurately estimate correct marginal distributions, which can be done using two techniques:

- **Subsampling:** When we subsample $\hat{\boldsymbol{x}}$ to a subset S, resulting in $\hat{\boldsymbol{x}}_S$, we effectively shift our target distribution to the specific marginal distribution $p(\hat{\boldsymbol{x}}_S)$. This technique is particularly valuable for representing objects of infinite dimensionality. According to the Kolmogorov Extension Theorem, such objects can be characterized through their finite-dimensional marginal distributions. Therefore, our approach involves learning the distributions $p(\hat{x}_{\tau_1}, \dots, \hat{x}_{\tau_N})$ for a series of random samples $\tau_1, \dots, \tau_N$ from the corresponding index set, typically represented by random time points. We can efficiently learn all finite-dimensional marginal distributions by randomly subsampling realizations of the process at these random time points. Additionally, it is particularly efficient because it reduces the sequence of variables during training. Importantly, this may necessitate modifying the attention mask, namely by ensuring that variables that were connected through a now-dropped node must be connected.

- **Modifying the attention mask:** Interestingly, altering the attention mask by a marginalization operation on the graph it represents is analogous to subsampling. For example, we may employ the identity mask to estimate all one-dimensional marginal distributions. The impact on the loss function can be reformulated as:

$$\mathcal{L}(\phi) = \mathbb{E}_{\hat{\boldsymbol{x}}_0, \hat{\boldsymbol{x}}_t}\left[\| s^I_{\phi^*}(\hat{\boldsymbol{x}}_t) - s(\hat{\boldsymbol{x}}_0, \hat{\boldsymbol{x}}_t)\|_2^2\right] = \sum_{i=1}^{d} \mathbb{E}_{\hat{\boldsymbol{x}}_0, \hat{\boldsymbol{x}}_t}\left[(s^I_{\phi^*}(\hat{\boldsymbol{x}}_t)^{(i)} - s(\hat{\boldsymbol{x}}_0, \hat{\boldsymbol{x}}_t)^{(i)})^2\right].$$

  As each variable is processed independently, thus $s^I_{\phi^*}(\hat{\boldsymbol{x}}_t)^{(i)} = s^I_{\phi^*}(\hat{\boldsymbol{x}}_t^{(i)})$ and for the family of SDEs (uncorrelated) we have $s(\hat{\boldsymbol{x}}_0, \hat{\boldsymbol{x}}_t)^{(i)} = s(\hat{\boldsymbol{x}}_0^{(i)}, \hat{\boldsymbol{x}}_t^{(i)})$. Consequently,

$$\mathcal{L}(\phi) = \sum_{i=1}^{d} \mathbb{E}_{\hat{\boldsymbol{x}}_0, \hat{\boldsymbol{x}}_t}\left[(s^I_{\phi^*}(\hat{\boldsymbol{x}}_t^{(i)}) - s(\hat{\boldsymbol{x}}_0, \hat{\boldsymbol{x}}_t^{(i)}))^2\right] = \sum_{i=1}^{d} \mathbb{E}_{\hat{\boldsymbol{x}}_0^{(i)}, \hat{\boldsymbol{x}}_t^{(i)}}\left[(s^I_{\phi^*}(\hat{\boldsymbol{x}}_t^{(i)}) - s(\hat{\boldsymbol{x}}_0, \hat{\boldsymbol{x}}_t^{(i)}))^2\right],$$

  This is essentially a sum of denoising score-matching losses for each one-dimensional marginal, verifying that it indeed aims to learn the correct marginal score. We can easily extend this result to other marginal distributions.

While we employed *subsampling* in the Lotka Volterra and SIR example. We do provide an example of the latter technique in Sec. A1.4.

A1.3. Post-hoc modifications

Altering the model configurations, such as employing different priors and likelihoods, is a consideration. Elsemüller et al. (2023) incorporated these modifications directly into their model. This is also possible here, but this method necessitates simulations across all configurations for training. Remarkably, our model allows a wide range of post-hoc adjustments after training on a single configuration, thus enabling it to represent a wide array of configurations. This flexibility is rooted in Bayes' rule, allowing for the decomposition of the score as

$$\nabla_{\boldsymbol{\theta}_t} \log p_t(\boldsymbol{\theta}_t|\mathbf{x}_t) = \nabla_{\boldsymbol{\theta}_t} \log p_t(\boldsymbol{\theta}_t) + \nabla_{\boldsymbol{\theta}_t} \log p_t(\mathbf{x}_t|\theta_t). \tag{1}$$

Our model can estimate scores for the model it is trained on i.e. as described in Eq. A1.4, but not for others. To address this limitation, we first can approximate

$$\nabla_{\boldsymbol{\theta}_t} \log p_t(\mathbf{x}_t|\boldsymbol{\theta}_t) \approx s_\phi(\boldsymbol{\theta}_t, t|\mathbf{x}_t) - s_\phi(\boldsymbol{\theta}_t, t), \tag{2}$$

and then adapt to a new family of model configurations using, for instance,

$$\nabla_{\boldsymbol{\theta}_t} \log p_t^{\alpha_1,\beta_1,\alpha_2,\beta_2}(\boldsymbol{\theta}_t|\mathbf{x}_t) \approx \underbrace{\alpha_1 \cdot (s_\phi(\boldsymbol{\theta}_t, t) + \beta_1)}_{\text{Prior change}} + \underbrace{\alpha_2 \cdot (s_\phi(\boldsymbol{\theta}_t, t|\mathbf{x}_t) - s_\phi(\boldsymbol{\theta}_t, t) + \beta_2)}_{\text{Likelihood change}}. \tag{3}$$

This decomposition is also the main mechanism behind classifier-free guidance methods (Ho & Salimans, 2021), which only act on the likelihood term. In general, α can temper the prior or likelihood, while β can shift the location. Yet, the exact influence can only be inferred with the precise knowledge of the corresponding distribution at hand.

In a similar line, we are able to impose almost arbitrary constraints by manipulating the score accordingly.

$$s_\phi(\hat{\boldsymbol{x}}_t, t|c) \approx s_\phi(\hat{\boldsymbol{x}}_t, t) + \nabla_{\hat{\boldsymbol{x}}_t} \sum_{i=1}^{K} \log \sigma(-s(t)c_i(\hat{\boldsymbol{x}}_t))$$

for a set of K equations $c_i(\hat{\boldsymbol{x}}_t) \leq 0$, specifying a specific constraint, and a scaling function s. More details on the exact implementation and choices in Sec. A3.3.

A1.4. Toy example

To demonstrate some of the above that we did not consider in the main paper, we consider a simple toy example of the form.

$$\theta \sim \mathcal{N}(0, 3^2) \qquad x_1 \sim \mathcal{N}(2 \cdot \sin(\theta), 0.5^2) \qquad x_2 \sim \mathcal{N}(0.1 \cdot \theta^2, 0.5 \cdot |x_1|)$$

We train the Simformer using the following masks: (1) a dense mask for joint estimation, (2) an identity mask for accurate one-dimensional marginal estimation, and (3) two-dimensional marginal masks for precise two-dimensional marginal estimation. Indeed, in contrast to a model trained solely with a dense mask, our approach correctly estimates the marginals even in the absence of other variables, as shown in Fig. A2. While both models can accurately capture the joint distribution (and consequently the marginals), this accuracy is contingent on receiving the complete sequence of variables as input.

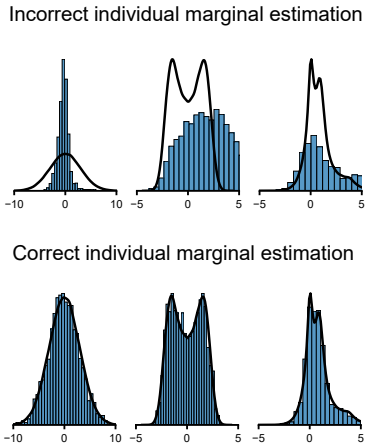

Figure A2. A model trained on a dense attention mask will predict the wrong marginal distribution without all other variables (top). A model trained also on the identity mask will provide correct marginals in the absence of all other variables (bottom)

Next, we aim to impose certain constraints on a simplified version of diffusion guidance. Which are:

- Interval: $c_1(x_1) = (x_1 - 2)$ and $c_2(x_1) = (3 - x_1)$.
- Linear: $c_1(x_1, \theta) = (x_1 + \theta)$ and $c_2(x_1, \theta) = -(x_1 + \theta)$.

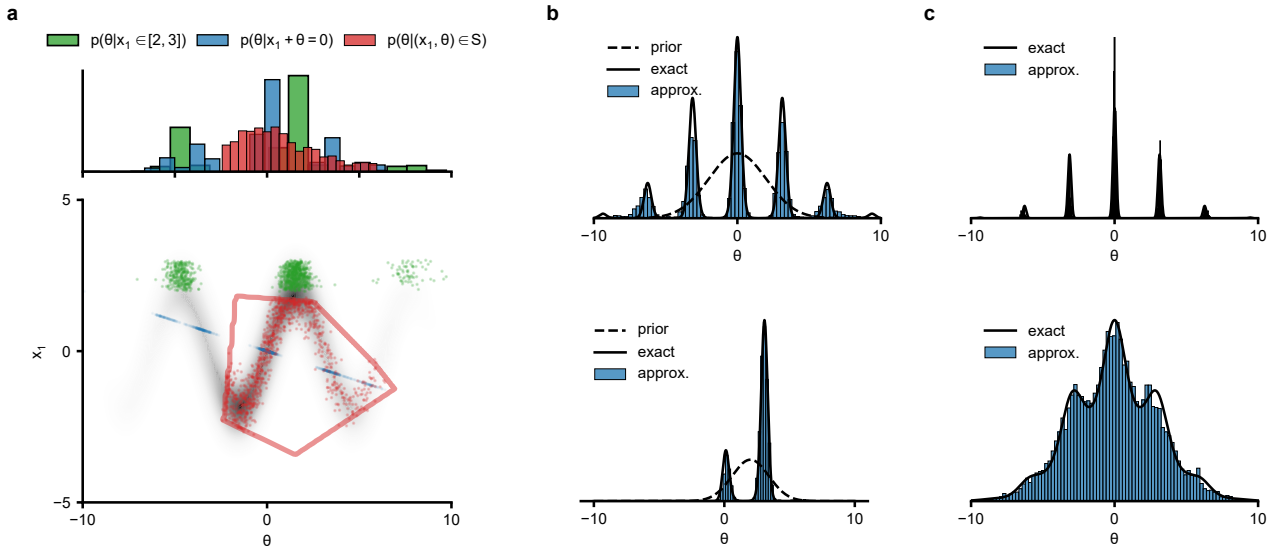

Figure A3. Illustration of the impact of post-hoc modifications on the 2d marginal posterior distribution for various model configurations, given the observation $x_1 = 0$. **(a)** Black shade shows ground-truth joint distributions. Scatter plots show samples with imposed constraints. **(b)** Posterior distribution with post-hoc modification to the prior i.e. increasing variance (top) or decreasing and shifting location. **(c)** Posteriors obtained by decreasing (top) or increasing (bottom) the variance of the likelihood

- Polytope: $c(x_1, \theta) = (A(x_1, \theta)^T - 1)$.

As visible in Fig. A3, we indeed can enforce this constraint while predicting the correct associated θ distribution.

Last but not least, we want to explore the capability to generalize to different generative models. In this example, with Gaussian distributions, affine transformations of approximate Gaussian scores will maintain their Gaussian nature, but we can alter the mean and variance.

In the Gaussian scenario, we have

$$\nabla_x \log \mathcal{N}(x; \mu_0, \sigma_0^2) = \frac{x - \mu_0}{\sigma_0^2},$$

thus, to adjust this score to a specific mean μ and variance σ^2, the appropriate choices would be

$$\alpha = \frac{\sigma_0^2}{\sigma^2}, \quad \text{and} \quad \beta = \frac{\mu - \mu_0}{\sigma_0^2}.$$

As demonstrated in Fig. A3, these post hoc modifications indeed enable the computation of the posterior distribution for the same observation $x_1 = 0$ across diverse configurations. It is crucial to acknowledge, however, that these modifications have limitations, particularly if the changes are significantly divergent from the distributions of the initially trained model. This is evident in the figure, as increasing the prior variance works less well than decreasing it.

A2. Experiment details

A2.1. Training and model configurations:

In our experiments, we adhere to the Stochastic Differential Equations (SDEs) as proposed by Song et al. (2021b), specifically the Variance Exploding SDE (VESDE) and the Variance Preserving SDE (VPSDE). These are defined as follows:

For VESDE:

$$f_{\text{VESDE}}(x,t) = 0, \qquad g_{\text{VESDE}}(t) = \sigma_{min} \cdot \left(\frac{\sigma_{max}}{\sigma_{min}}\right)^t \cdot \sqrt{2 \log \frac{\sigma_{max}}{\sigma_{min}}} \tag{4}$$

For VPSDE:

$$f_{\text{VPSDE}}(x,t) = -0.5 \cdot (\beta_{\text{min}} + t \cdot (\beta_{\text{max}} - \beta_{\text{min}})), \qquad g_{\text{VPSDE}}(t) = \sqrt{\beta_{\text{min}} + t \cdot (\beta_{\text{max}} - \beta_{\text{min}})} \tag{5}$$

We set $\sigma_{\text{max}} = 15$, $\sigma_{\text{min}} = 0.0001$, $\beta_{\text{min}} = 0.01$, and $\beta_{\text{max}} = 10$ for all experiments. Both for the time interval $[1e-5, 1.]$.

For implementing Neural Posterior Estimation (NPE), Neural Ratio Estimation (NRE), and Neural Likelihood Estimation (NLE), we utilize the sbi library (Tejero-Cantero et al., 2020), adopting default parameters but opting for a more expressive neural spline flow for NPE and NLE. Each method was trained using the provided training loop with a batch size of 1000 and an Adam optimizer. Training ceased upon convergence, as indicated by early stopping based on validation loss.

The employed transformer model features a token dimension of 50 and represents diffusion time through a 128-dimensional random Gaussian Fourier embedding. It comprises 6 layers and 4 heads with an attention size of 10, and a widening factor of 3, implying that the feed-forward block expands to a hidden dimension of 150. For the Lotka-Volterra, SIR, and Hodgkin-Huxley tasks, we increased the number of layers to 8. Similar to the above, we used a training batch size of 1000 and an Adam optimizer.

In all our experiments, we sampled the condition mask M_C as follows: At every training batch, we selected uniformly at random a mask corresponding to the joint, the posterior, the likelihood or two random masks. The random masks were drawn from a Bernoulli distribution with $p = 0.3$ and $p = 7$. In our experiments, we found this to work slightly better than just random sampling and sufficiently diverse to still represent all the conditionals. The edge mask M_E is chosen to match the generative process (see Fig. A4). The undirected variant was obtained by symmetrization. Note that this is the only input we provide; additional necessary dependencies, e.g., due to conditioning, are algorithmically determined (see Sec. A1.1).

For inference, we solved the reverse SDE using an Euler-Maruyama discretization. We use 500 steps by default; accuracy for different budgets is shown in Fig. A7.

A2.2. Tasks:

The tasks Gaussian Linear, Gaussian Mixture, Two Moons, and SLCP were used in Lueckmann et al. (2021).

Gaussian Linear: The prior for the parameter $\boldsymbol{\theta}$ is a normal distribution $\mathcal{N}(0, 0.1 \cdot \mathbf{I})$. The data $\boldsymbol{x}$ given $\boldsymbol{\theta}$ is generated by a Gaussian distribution $\mathcal{N}(\boldsymbol{x}; \boldsymbol{\theta}, 0.1 \cdot \mathbf{I})$. Both $\boldsymbol{\theta}, \boldsymbol{x} \in \mathbb{R}^{10}$.

Gaussian Mixture This task, commonly referenced in Approximate Bayesian Computation (ABC) literature (Sisson et al., 2007; Beaumont et al., 2009), involves inferring the common mean of a mixture of two-dimensional Gaussian distributions with distinct covariances. The task is defined as follows. The prior for the parameters $\boldsymbol{\theta}$ is a uniform distribution, denoted as $\mathcal{U}(-10, 10)$. The data $\boldsymbol{x}$ given $\boldsymbol{\theta}$ is modeled as a mixture of two Gaussian distributions:

$$\boldsymbol{x}|\boldsymbol{\theta} \sim 0.5 \cdot \mathcal{N}(\boldsymbol{x}; \boldsymbol{\theta}, \mathbf{I}) + 0.5 \cdot \mathcal{N}(x; \boldsymbol{\theta}, 0.01 \cdot \mathbf{I})$$

The parameter space $\boldsymbol{\theta}$ and the data space $\boldsymbol{x}$ are both in $\mathbb{R}^2$.

Two Moons : The Two Moons task is designed to test inference algorithms in handling multimodal distributions (Greenberg et al., 2019). The prior is a Uniform distribution $U(\boldsymbol{\theta}; -1, 1)$. The data $\boldsymbol{x}$ is generated from $\boldsymbol{\theta}$ as:

$$\boldsymbol{x}|\boldsymbol{\theta} = \begin{bmatrix} r\cos(\alpha) + 0.25 \\ r\sin(\alpha) \end{bmatrix} + \begin{bmatrix} -|\theta_1 + \theta_2|/\sqrt{2} \\ (-\theta_1 + \theta_2)/\sqrt{2} \end{bmatrix},$$

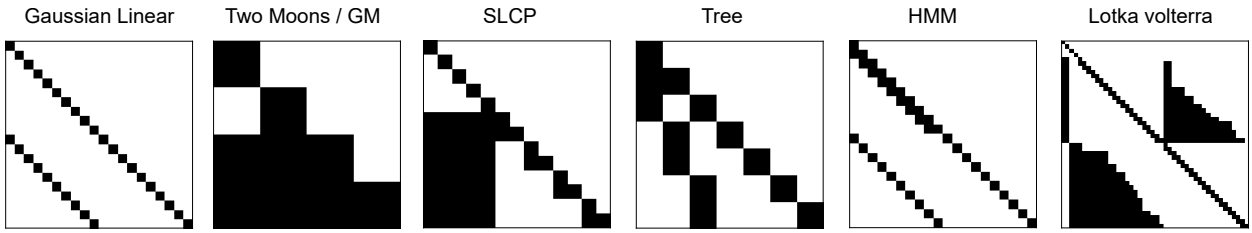

Figure A4. Directed *base* masks for each of the tasks.The Lotka Volterra mask dynamically adapts to different input times, here just for randomly selected times.

where $\alpha \sim \mathcal{U}(-\pi/2, \pi/2)$ and $r \sim \mathcal{N}(0.1, 0.012)$. Leading to a dimensionality $\boldsymbol{\theta} \in \mathbb{R}^2$, $\boldsymbol{x} \in \mathbb{R}^2$.

To obtain reference samples for all possible conditionals, we run the following procedure:

- We initialized N Markov chains with samples from the joint distribution.
- We run 1000 steps of a random direction slice sampling algorithm.
- We run an additional 3000 steps of MHMCMC with step size of 0.01.
- Only the last samples of each chain were considered, yielding N reference samples.

This procedure yielded samples in agreement with the reference posterior provided by Lueckmann et al. (2021) (C2ST ~ 0.5). Other conditionals did also look correct, but were not extensively investigated.

SLCP Task: The SLCP (Simple Likelihood Complex Posterior) task is a challenging inference task designed to generate a complex posterior distribution (Papamakarios et al., 2019; Greenberg et al., 2019; Hermans et al., 2020; Durkan et al., 2020). The setup is as follows. The prior over $\boldsymbol{\theta}$ is a uniform distribution $\mathcal{U}(-3, 3)$. The data $\boldsymbol{x}$ given $\boldsymbol{\theta}$ is $\boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_4)$, where each $\boldsymbol{x}_i \sim \mathcal{N}(\mu_\theta, \boldsymbol{\Sigma}_\theta)$ with:

$$\mu_\theta = \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix},$$
$$\boldsymbol{\Sigma}_\theta = \begin{bmatrix} \theta_3^2 & \tanh(\theta_5) \cdot \theta_3^2 \cdot \theta_4^2 \\ \tanh(\theta_5) \cdot \theta_3^2 \cdot \theta_4^2 & \theta_4^2 \end{bmatrix}.$$

Leading to a dimensionality of $\boldsymbol{\theta} \in \mathbb{R}^5$, $\boldsymbol{x} \in \mathbb{R}^8$.

To obtain reference samples for all possible conditionals, we run the following procedure:

- We initialized N Markov chains with samples from the joint distribution.
- We run 600 steps of a random direction slice sampling algorithm.
- We run an additional 2000 steps of MHMCMC with a step size of 0.1.
- Only the last samples of each chain was considered, yielding N reference samples.

This procedure yielded samples in agreement with the reference posterior provided by Lueckmann et al. (2021) (C2ST ~ 0.5). Other conditionals did also look correct, but were not extensively investigated.

Tree: This is a nonlinear tree-shaped task:

$$\theta_0 \sim \mathcal{N}(\theta_0; 0, 1.) \qquad \theta_1 \sim \mathcal{N}(\theta_0; 1.) \qquad \theta_2 \sim \mathcal{N}(\theta_2; \theta_0, 1.).$$

Observable data is obtained through

$$x_0 \sim \mathcal{N}(x_1; \sin(\theta_1)^2, 0.2^2) \quad x_1 \sim \mathcal{N}(0.1 \cdot \theta_1^2, 0.2^2) \quad x_2 \sim \mathcal{N}(x_2; 0.1 \cdot \theta_2^2, 0.6^2) \quad x_3 \sim \mathcal{N}(x_3; \cos(\theta_2)^2; 0.1^2)$$

which leads to a tree-like factorization with highly multimodal conditionals.

To obtain reference samples for all possible conditionals, we run the following procedure:

- We initialized N Markov chains with samples from the joint distribution.
- We run 5000 steps of a HMC sampler.
- Only the last samples of each chain were considered, yielding N reference samples.

HMM: This is a task in which the parameters have a Markovian factorization.

$$\theta_0 \sim \mathcal{N}(\theta_0; 0., 0.5^2) \qquad \theta_{i+1} \sim \mathcal{N}(\theta_{i+1}; \theta_i, 0.5^2)$$

for $i = 0, ..., 9$. Observations are generated according to $x_i = \mathcal{N}(x_i; \theta_i^2, 0.5^2)$, leading to a nonlinear hidden Markov model with bimodal correlated posterior and leading to a dimensionality of $\boldsymbol{\theta} \in \mathbb{R}^{10}$, $\boldsymbol{x} \in \mathbb{R}^{10}$.

To obtain reference samples for all possible conditionals, we run the following procedure:

- We initialized N Markov chains with samples from the joint distribution.
- We run 5000 steps of an HMC sampler.
- Only the last samples of each chain were considered, yielding N reference samples.

Lotka Volterra The Lotka-Volterra equations, a foundational model in population dynamics, describe the interactions between predator and prey species (Volterra, 1926; Lotka, 1925). This model is parameterized as follows: the prior is chosen to be a sigmoid-transformed Normal distribution, scaled to a range from one to three. Data then evolves according to the following differential equations:

$$\begin{aligned} \frac{dx}{dt} &= \alpha x - \beta xy, \\ \frac{dy}{dt} &= \delta xy - \gamma y. \end{aligned} \tag{6}$$

Here, x and y represent the population sizes of the prey and predator species, respectively. The parameters α, β, γ, and δ are positive real numbers that describe the two species' interaction rates and survival rates. To each simulation, we add Gaussian observation noise with $\sigma = 0.1$.

SIRD Model with Time-Dependent Contact Rate The SIRD (Susceptible, Infected, Recovered, Deceased) model extends the classical SIR framework by incorporating a Deceased (D) compartment. Similar models were explored by Chen et al. (2020); Schmidt et al. (2021). This addition is crucial for modeling diseases with significant mortality rates. The dynamics of the SIRD model, considering a time-dependent contact rate, are governed by the following set of differential equations:

$$
\begin{aligned}
\frac{dS}{dt} &= -\beta(t)SI, \\
\frac{dI}{dt} &= \beta(t)SI - \gamma I - \mu I, \\
\frac{dR}{dt} &= \gamma I, \\
\frac{dD}{dt} &= \mu I.
\end{aligned}
$$

Here, S, I, R, and D denote the number of susceptible, infected, recovered, and deceased individuals, respectively. The term $\beta(t)$ represents the time-varying contact rate, while γ and μ signify the recovery and mortality rates among the infected population, respectively.

Incorporating a time-dependent contact rate $\beta(t)$ is pivotal for capturing the effects of public health interventions and societal behavioral changes over time. This feature is essential for accurately simulating the real-world dynamics of a disease's spread, particularly in the context of varying public health policies and community responses.

We impose a Uniform prior on the global variables, γ and δ, denoted as $\gamma, \delta \sim \text{Unif}(0, 0.5)$. For the time-dependent contact rate, we first sample $\hat{\beta} \sim \mathcal{G}(0, k)$ from a Gaussian process prior, with k representing an RBF kernel defined as $k(t_1, t_2) = 2.5^2 \exp\left(-\frac{1}{2}\frac{\|t_1 - t_2\|^2}{7^2}\right)$. This is further transformed via a sigmoid function to ensure $\beta(t) \in [0, 1]$ for all t. Observational data is modeled with log-normal noise, characterized by a mean of $S(t)$ and a standard deviation of $\sigma = 0.05$.

Hodgkin-Huxley Model: In our study, we adhere to the implementation guidelines set forth by Pospischil et al. (2008) for the Hodgkin-Huxley model. The initial membrane voltage is established at $V_0 = -65.0\,\text{mV}$. Simulations are conducted over a duration of $200\,\text{ms}$, during which an input current of $4\,\text{mA}$ is applied in the interval between $50\,\text{ms}$ and $150\,\text{ms}$.

The rate functions are defined by the following equations:

$$
\begin{aligned}
\alpha_m(V) &= 0.32 \times \frac{\text{efun}\left(-0.25(V - V_0 - 13.0)\right)}{0.25}, \\
\beta_m(V) &= 0.28 \times \frac{\text{efun}\left(0.2(V - V_0 - 40.0)\right)}{0.2}, \\
\alpha_h(V) &= 0.128 \times \exp\left(-\frac{(V - V_0 - 17.0)}{18.0}\right), \\
\beta_h(V) &= \frac{4.0}{1.0 + \exp\left(-\frac{(V - V_0 - 40.0)}{5.0}\right)}, \\
\alpha_n(V) &= 0.032 \times \frac{\text{efun}\left(-0.2(V - V_0 - 15.0)\right)}{0.2}, \\
\beta_n(V) &= 0.5 \times \exp\left(-\frac{(V - V_0 - 10.0)}{40.0}\right)
\end{aligned}
$$

where $\text{efun}(x) = \begin{cases} 1 - \frac{x}{2} & \text{if } x < 1e-4 \\ \frac{x}{\exp(x) - 1.0} & \text{otherwise} \end{cases}$.

This formulation leads to the comprehensive Hodgkin-Huxley differential equations:

$$
\begin{aligned}
\frac{dV}{dt} &= \frac{I_{\text{inj}}(t) - g_{\text{Na}} m^3 h (V - E_{\text{Na}}) - g_{\text{K}} n^4 (V - E_{\text{K}}) - g_{\text{L}} (V - E_{\text{L}})}{C_m} + 0.05\, dw_t, \\
\frac{dm}{dt} &= \alpha_m(V)(1 - m) - \beta_m(V) m, \\
\frac{dh}{dt} &= \alpha_h(V)(1 - h) - \beta_h(V) h, \\
\frac{dn}{dt} &= \alpha_n(V)(1 - n) - \beta_n(V) n, \\
\frac{dH}{dt} &= g_{\text{Na}} m^3 h (V - E_{\text{Na}}).
\end{aligned}
$$

Notably, there exist multiple methodologies for estimating energy consumption in neuronal models, as discussed by Deistler et al. (2022b). In our approach, we opt to calculate energy consumption based on sodium charge, which can be converted into $\mu J/s$ as detailed by Deistler et al. (2022b). For observational data, we employ summary features consistent with those used by Gonçalves et al. (2020).

A3. Additional experiments

In Sec. A3.1, we include additional experiments, i.e., investigating different SDEs, comparing to more methods, adding additional metrics, and reviewing efficiency. In Sec. A3.2, we demonstrate target inference with embedding nets on a complex task for gravitational wave data. Finally, in Sec. A3.3, we review how good guidance methods can compute arbitrary conditionals, as well as general constraints.

A3.1. Extended benchmark

Overview of benchmark results: Comprehensive benchmark results have been obtained for both the Variance Exploding SDE (VESDE) and the Variance Preserving SDE (VPSDE) models, as well as for several SBI methods. These methods include Neural Posterior Estimation (NPE) (Papamakarios & Murray, 2016), Neural Likelihood Estimation (NLE) (Papamakarios et al., 2019), and Neural Ratio Estimation (NRE) (Hermans et al., 2020). The outcomes of these benchmarks are depicted in Figure A5 and Figure A6.

Furthermore, we have implemented a baseline Neural Posterior Score Estimation (NPSE) method (Simons et al., 2023; Geffner et al., 2023), where the score network is a conditional MLP in contrast to the transformer architecture. Additionally, a variant named the 'Simformer (posterior only)' was tested, in which the training focuses exclusively on the associated posterior masks, rendering its neural network usage akin to NPSE (up to different architectures). As expected, these two approaches do perform similarly. Furthermore, this shows that targeting all conditionals does not hurt (but can even improve) the performance even when evaluating the posterior only.

Comparative performance of SDE variants: Overall, the different SDE variants exhibit comparably high performance, with some notable exceptions. Specifically, the VESDE model demonstrates superior performance in the Two Moons task, whereas the VPSDE model shows a slight edge in the SLCP task.

Impact of training only on posterior masks: Interestingly, training solely on the posterior mask does not enhance performance relative to learning all conditional distributions. This observation confirms our initial hypothesis that the desired property of efficient learning of all conditionals is inherently 'free' in our framework. In cases like the SLCP, where the likelihood is relatively simple, there appears to be an added advantage in learning both the posterior and the likelihood distributions. Traditionally, likelihood-based methods such as NLE outperform direct posterior estimation techniques on this task. As the Simformer approach estimates both quantities jointly, it may benefit from this additional information.

Model evaluations for reverse diffusion: In Figure A7, we illustrate how the C2ST varies with the number of model evaluations used in solving the reverse SDE. This variation is observed by examining different uniform discretizations of the time interval $[0, 1]$ with varying numbers of elements. Notably, the performance improvement of the method with an increasing number of evaluations is not gradual. Rather, there is a sharp transition from suboptimal to near-perfect performance when the number of evaluations exceeds 50. This finding is particularly favorable for diffusion models, as opposed to methods like NLE or Neural Ratio Estimation NRE, which necessitate a subsequent Markov Chain Monte Carlo (MCMC) run. It is important to note that these MCMC runs typically require significantly more than 50 evaluations, highlighting the efficiency of diffusion models in this context. This is especially important as transformer models are usually more expensive to evaluate than the network architectures used in NLE and NRE.

Average negative loglikelihood: The average negative loglikelihood (NLL) for the true posterior is a metric suitable for evaluation on an increased number of different observations (Lueckmann et al., 2021; Hermans et al., 2022). We evaluate the average on 5000 samples from the joint distribution. We did this for both the posterior and likelihood, as estimated by Simformer, and compared it to the corresponding NPE and NLE baseline. Note that NPE and NLE are trained to minimize the NLL, giving it a natural advantage. In contrast, Simformer only indirectly minimizes negative loglikelihood through the score-matching objective. Notably, to evaluate the loglikelihood for the Simformer, we have to use the probability flow ODE (Song et al., 2021b). Hence, the loglikelihood is also based on the probability flow ODE, not the corresponding SDE formulation (which does not necessarily exactly agree for a finite number of steps). We show the corresponding result in Fig A8. In most cases, the results agree with the C2ST evaluation (which only evaluates SDE sampling quality). However, in some cases NLE or NPE does perform better with respect to this metric. The difference is due to the discrepancy between SDE sampling and ODE log probability evaluation and the fact that Simformer is not trained to minimize loglikelihood, which is not necessarily at odds with producing good samples.

Calibration: To check whether the distributions estimated by Simformer are well-calibrated, we performed an expected coverage analysis (Hermans et al., 2022), again both for the posterior and likelihood. Intuitively, this test checks whether the ground-truth parameter lies within the top $\alpha\%$ highest density region in $\alpha\%$ of all cases (which is what the true posterior must satisfy). The same analysis was performed for NPE as a reference (see Fig. A9). In cases in which the likelihood is significantly easier to learn than the posterior (i.e., SLCP), we can observe that, indeed, the estimate of the simple likelihood becomes well calibrated earlier than the posterior (see Fig. A10, Fig. A11, Fig. A12, upper right corner). Overall, Simformer is well-calibrated and, similar to NPE, tends to more *conservative* approximations (coverage plots tend to be above the diagonal).

We also perform a coverage analysis on the SIR task (Fig. A13). Note that because this model is nonparametric, there are infinitely many distributions we could evaluate (i.e. by selecting different times for observations or parameters). We opt to run an evaluation on 20 random time points for each time-dependent parameter (contact rate) or observation (S, I, D).

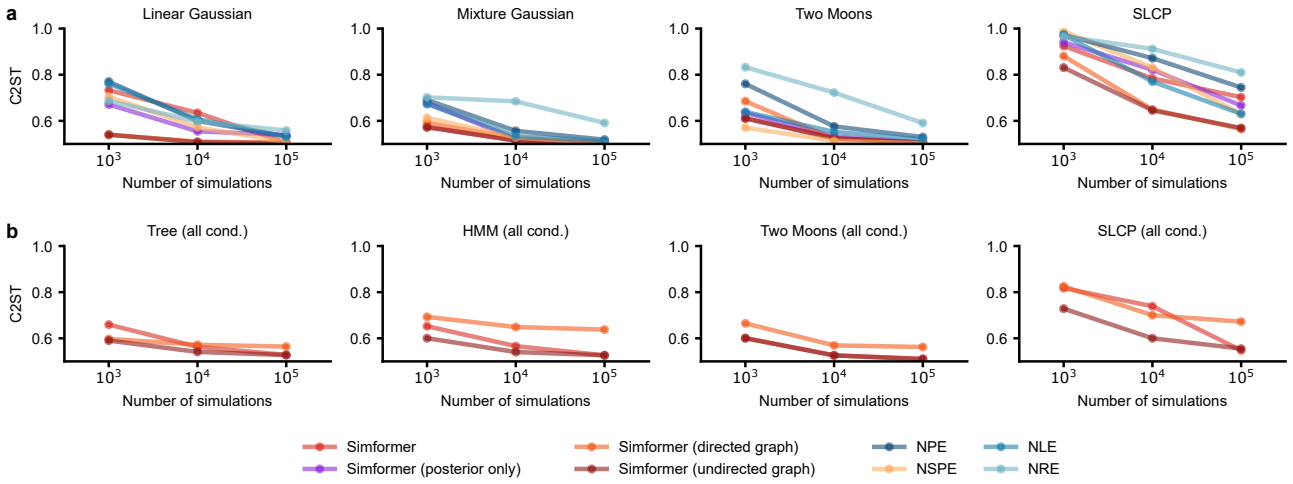

Figure A5. Extended benchmark results for the VESDE. In addition to NPE, we also run NRE, NLE, and NSPE. **(a)** Shows performance in terms of C2ST for SBIBM tasks. **(b)** Shows performance in terms of C2ST for all conditional distributions.

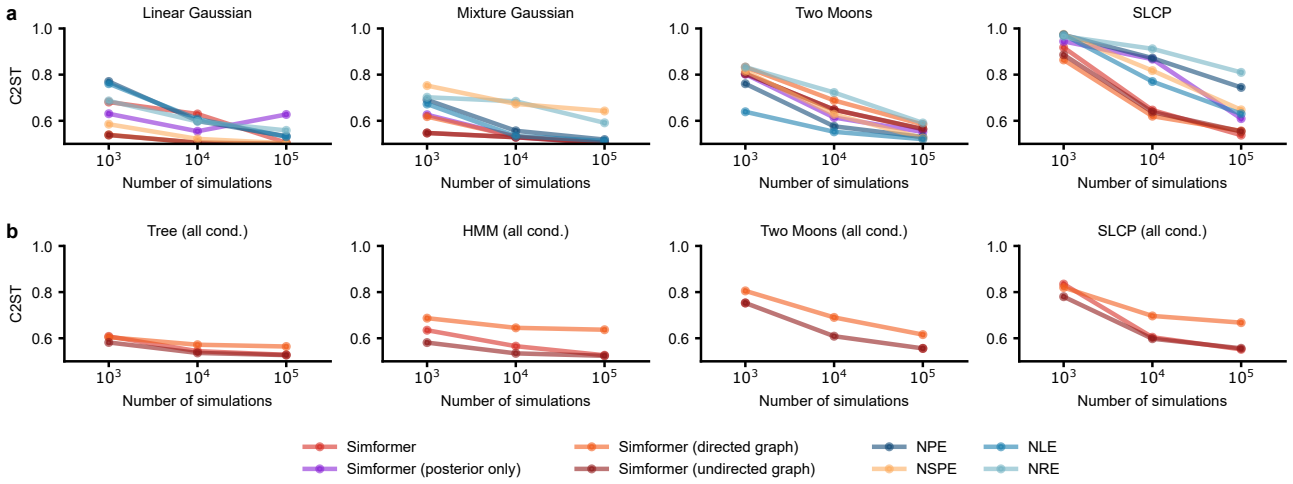

Figure A6. Extended benchmark results for the VPSDE. In addition to NPE, we also run NRE, NLE, and NSPE. **(a)** Shows performance in terms of C2ST for SBIBM tasks. **(b)** Shows performance in terms of C2ST for all conditional distributions.

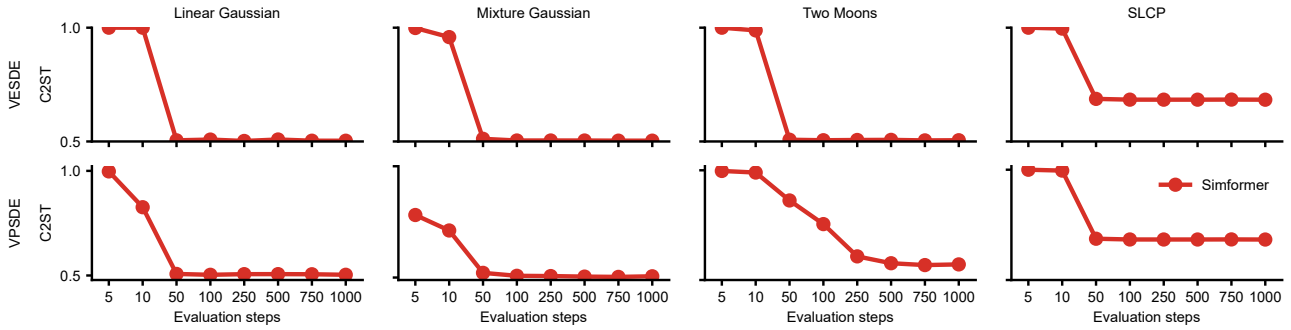

Figure A7. For all tasks as well as the VPSDE and VESDE, we show how the performance as measured in C2ST increases as we increase the evaluation steps to solve the reverse SDE. For all tasks, except Two Moons on the VPSDE, 50 evaluations are sufficient to reach best performance.

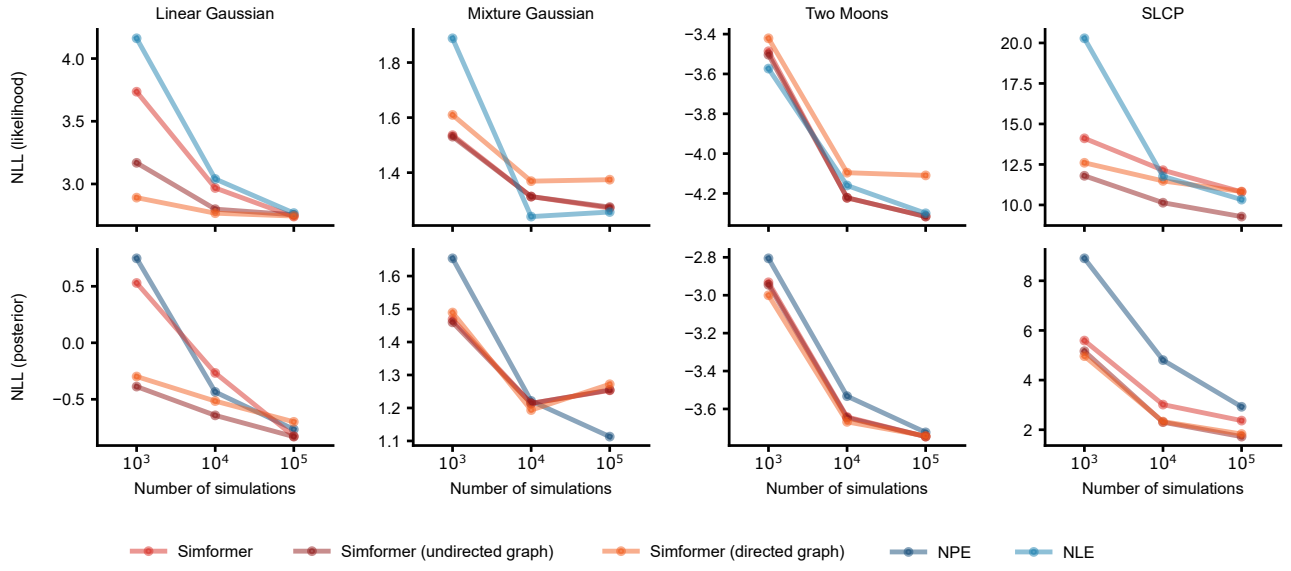

Figure A8. Average negative loglikelihood of the true parameter for NPE, NLE, and all Simformer variants. Evaluating both the likelihood (top row) and posterior (bottom row).

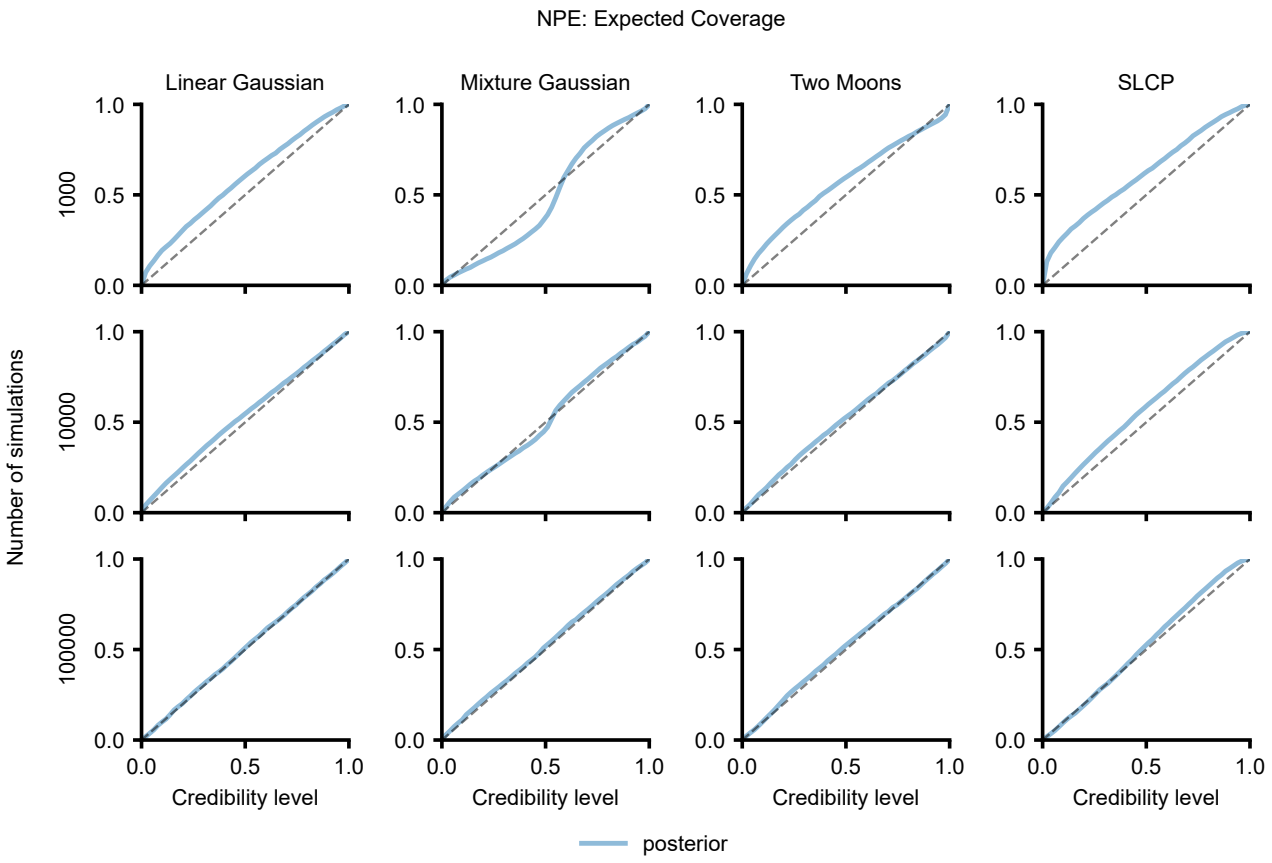

Figure A9. Calibration analysis for NPE using *expected coverage* (Hermans et al., 2022). Each row corresponds to training simulation sizes of 1k, 10k, 100k. Each column represents a task.

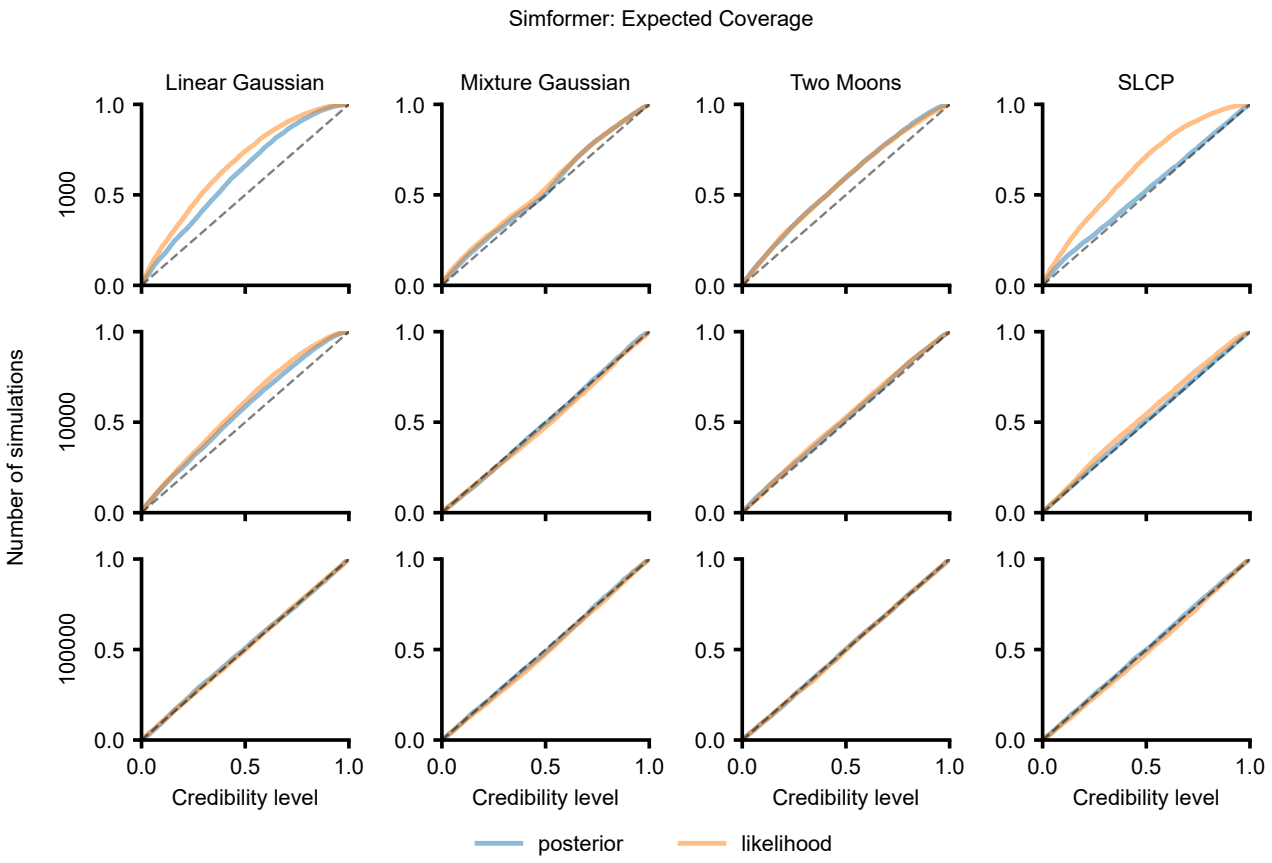

Figure A10. Calibration analysis for Simformer using *expected coverage* (Hermans et al., 2022), both for the posterior and likelihood. Each row corresponds to training simulation sizes of 1k, 10k, 100k. Each column represents a task.

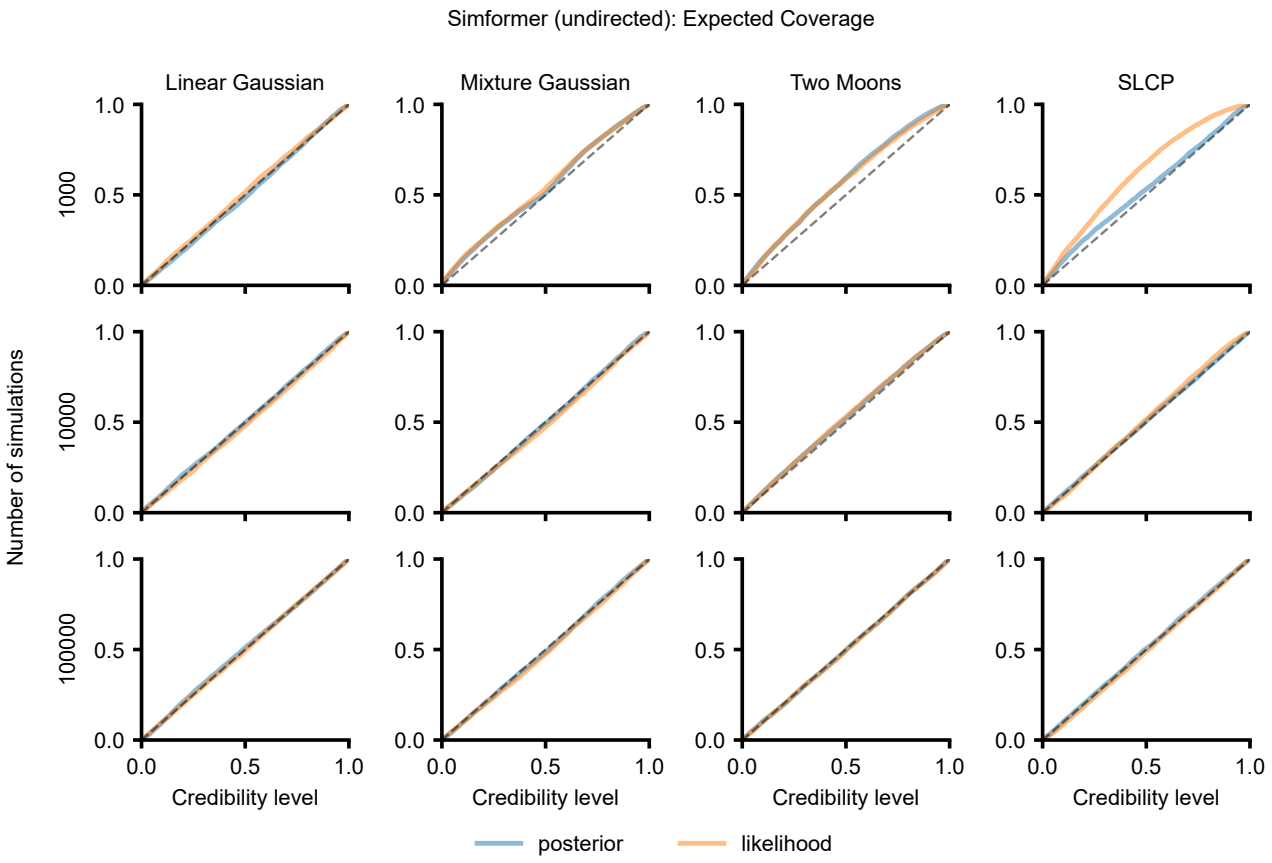

Figure A11. Calibration analysis for Simformer (undirected) using *expected coverage* (Hermans et al., 2022), both for the posterior and likelihood. Each row corresponds to training simulation sizes of 1k, 10k, 100k. Each column represents a task.

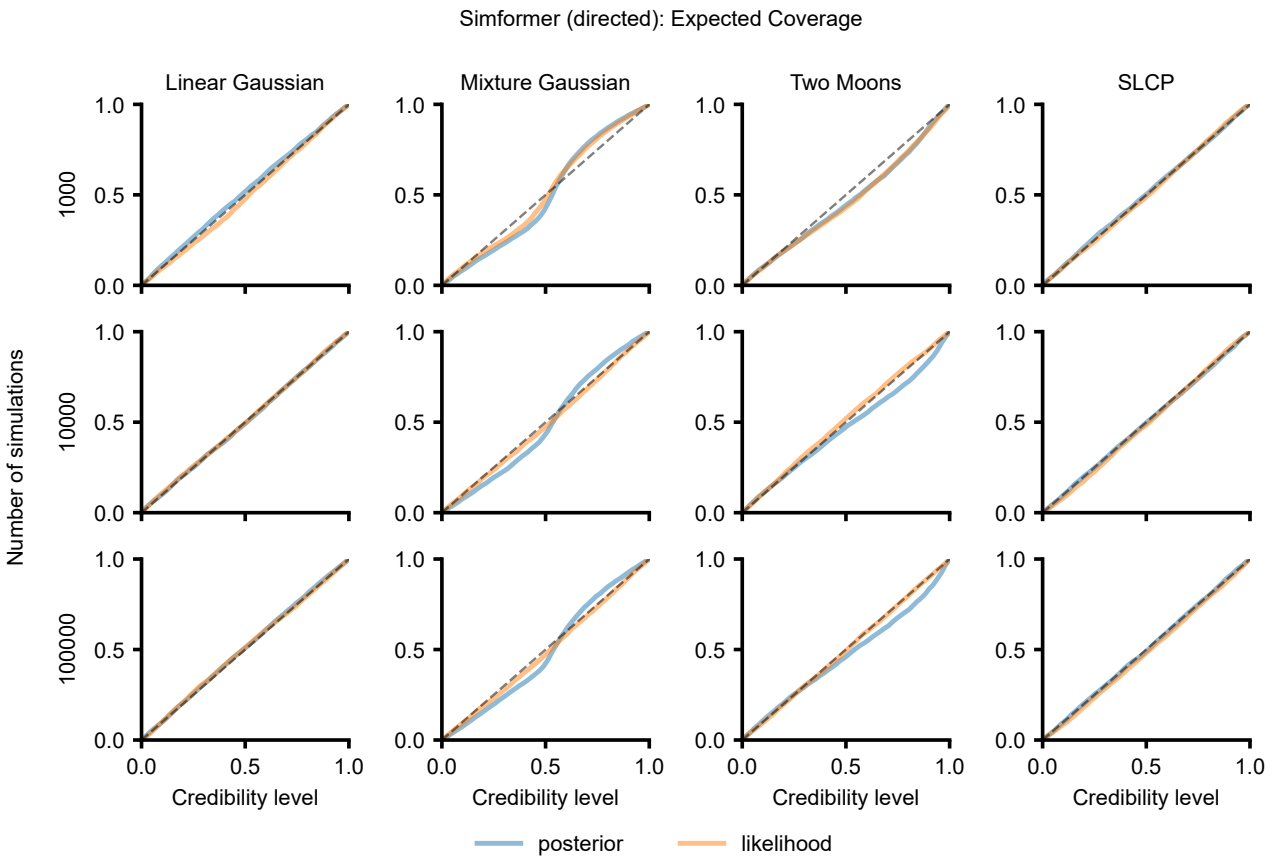

Figure A12. Calibration analysis for Simformer (directed) using *expected coverage*, both for the posterior and likelihood. Each row corresponds to training simulation sizes of 1k, 10k, 100k. Each column represents a task.

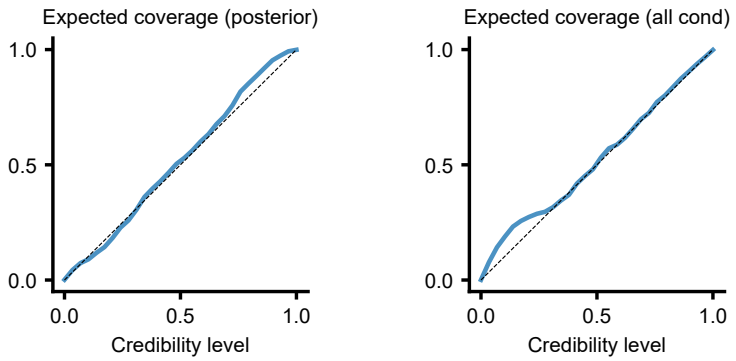

Figure A13. Calibration analysis for the SIR task using *expected coverage* (Hermans et al., 2022). On the left, we evaluate the posterior (for randomly selected time points). On the right, we have the coverage for different conditional distributions (also for randomly selected time points).

A3.2. Targeteted inference and embedding nets

In the main manuscript, we focus on estimating all conditionals of a certain task. However, in certain scenarios, it might simply not be wanted or way harder to do so. In this case, we can query Simformer to simply target only a subset of conditionals by restricting the number of condition masks M_C to whatever conditionals we deem worth estimating. Secondly, in tasks were data is high dimensional, it becomes computationally demanding to consider each scalar as a variable. In this case, we should encode whole vectors into a single token.

As a test case, we will consider the Gravitational Waves benchmark tasks as presented in Hermans et al. (2022). In this case, we have low dimensional $\theta \in \mathbb{R}^2$, i.e., the masses of the two black holes, and two high dimensional $\boldsymbol{x} \in \mathbb{R}^{8192}$ measurements of the corresponding gravitational waves from two different detectors. In this case, it is clear that learning the likelihood, i.e., a conditional generative model for the high dimensional observations, is harder than just learning the posterior over the two parameters. A common practice for high dimensional observations is to use an *embedding network*, i.e., a neural network that compresses it to a lower dimensional vector. Hermans et al. (2022) did use a convolutional embedding net for NPE on this task. As already hinted in the manuscript, we can do the same for Simformer, i.e., we compress the detector measurements using a convolutional neural network into a single token. Additionally to the full posterior distribution, we are still interested in the partial posterior distributions as, e.g., there might only be measurements from one of the detectors (notably, the measurements are not independent). We hence only target the conditionals $p(\boldsymbol{\theta}|\boldsymbol{x}_1, \boldsymbol{x}_2)$, $p(\boldsymbol{\theta}|\boldsymbol{x}_1)$ and $p(\boldsymbol{\theta}|\boldsymbol{x}_2)$. We use 100k simulations for training. For two examples, we show the estimated (partial) posterior(s) in Fig. A14a Fig. A14b. Simformer can combine the information from both detectors in a meaningful way (as verified by a calibration analysis, Fig. A14c.

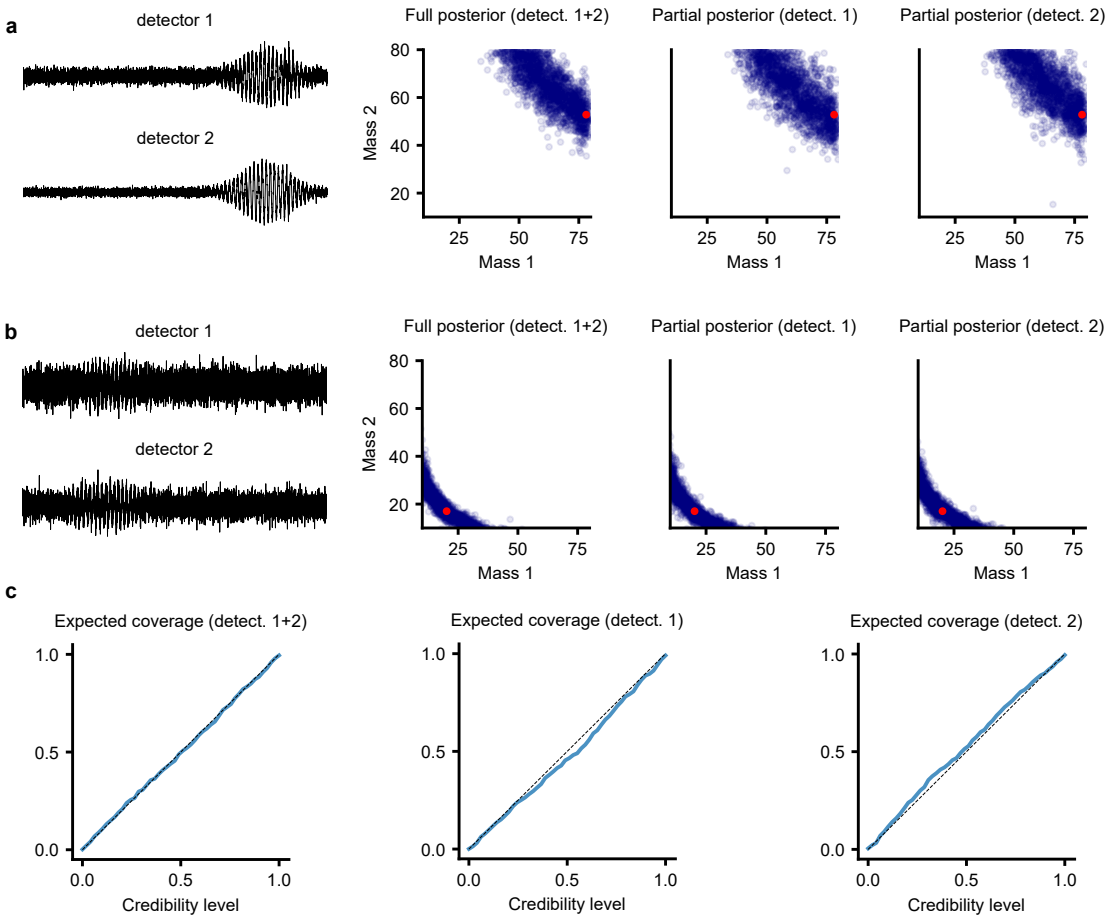

Figure A14. Inference on gravitational wave measurements. **(a)** Shows the detector measurements of a gravitational wave (first column). The associated posterior and partial posteriors for the detector measurements. **(b)** Shows the same as in (a) but for different observations. **(c)** Calibration analysis of the posterior and partial posteriors in terms of *expected coverage*.

A3.3. Details on general guidance

Diffusion guidance can vary in its implementation from less rigorous to highly rigorous approaches. Achieving rigor in this context typically necessitates a known likelihood function. However, in the realm of SBI, this likelihood function is often either intractable or challenging to compute (Chung et al., 2023). Consequently, our focus is directed towards universally applicable approximations, as discussed in the works of Lugmayr et al. (2022) and Bansal et al. (2023).

In our methodology, we integrate two principal strategies that have demonstrated efficacy in practical scenarios. The first of these strategies is self-recurrence, as advocated by Lugmayr et al. (2022). This might also be interpreted as a predictor-corrector algorithm (Song et al., 2021b) with a pseudo-Gibbs sampling corrector. This approach has been shown to substantially improve performance, though it necessitates an increase in computational resources. The second strategy entails adjusting the estimated score with a general constraint function, which we evaluate on a *denoised* estimate of the variables (Bansal et al., 2023; Chung et al., 2023; Rozet & Louppe, 2021). Overall, this is remarkable flexibility and supports almost any constraint to be incorporated. We provide a pseudo-code in Algorithm 1. In our experimental assessments, it proved to be sufficiently accurate. For comparative purposes, we also implemented the RePaint method as proposed by Lugmayr et al. (2022). However, it is important to note that this method primarily applies to normal conditioning and does not readily extend to general constraints. On the other hand, *General guidance* requires the specification of a scaling function, which up and down scales the constrained score at different diffusion times t. As the magnitude of the marginal score does depend on the SDE, this scaling function should also. In our experiment, we generally used a scaling function of the form $s(t) = \frac{1}{\sigma(t)^2}$, i.e., which is inversely proportional to the variance of the approximate marginal SDE scores.

Algorithm 1 General Guidance

Require: Number of steps T, Min time T_{min}, Max time T_{max},self-recurrence steps r, scaling function $s(t)$ and constraint function $c(x)$, drift coefficient $f(x,t)$, diffusion coefficient $g(t)$, associated mean and standard deviation functions μ, σ such that $\hat{\boldsymbol{x}}_t = \mu(t)\hat{\boldsymbol{x}}_0 + \sigma(t)\epsilon$.

Set time step $\Delta t = \frac{T_{\text{max}} - T_{\text{min}}}{T}$

Sample $\hat{\boldsymbol{x}}_T \sim \mathcal{N}(\mu_T, \sigma_T \mathbf{I})$ *// Initialize at terminal distribution*

for $i = 1$ **downto** T **do**

  $t_i = T_{\text{max}} - i \cdot \Delta t$

  for $j = 1$ **to** r **do**

    $\epsilon \sim \mathcal{N}(0, \mathbf{I})$

    $s = s_\phi(\hat{\boldsymbol{x}}_{t_{i+1}}, t_i)$ *// Marginal score estimate*

    $\hat{\boldsymbol{x}}_{\sim 0} = \frac{\hat{\boldsymbol{x}}_{t_{i+1}} + \sigma(t_{i+1})^2 \cdot s}{\mu(t_{i+1})}$ *// Denoise*

    $\tilde{s} = s + \nabla_{\hat{\boldsymbol{x}}} \log \sigma(s(t) c(\hat{\boldsymbol{x}}_{\sim 0}))$ *// Constraint score*

    $\hat{\boldsymbol{x}}_{t_i} = \hat{\boldsymbol{x}}_{t_{i+1}} - \left(f(\hat{\boldsymbol{x}}_{t_{i+1}}, t_i) - g(t_i)^2 \cdot \tilde{s}\right) \Delta t - g(t_i)\sqrt{\Delta t} \cdot \epsilon$

    if $r > 0$ **then**

      // Resample future point using SDE equations

      $\epsilon \sim \mathcal{N}(0, \mathbf{I})$

      $\hat{\boldsymbol{x}}_{t_{i+1}} = \hat{\boldsymbol{x}}_{t_i} + f(\hat{\boldsymbol{x}}_{t_{i+1}}, t_i)\Delta t + g(t_i)\sqrt{\Delta t} \cdot \epsilon$

    end if

  end for

end for

return $\hat{\boldsymbol{x}}_{T_{min}}$

Benchmarking the Guidance Methods: In this experiment, we diverged from traditional approaches by training the Simformer exclusively for joint estimation. The primary distinction from a conditional distribution lies in the condition mask distribution, which in this case is a point mass centered at the all-zero vector. Our comparative analysis, as depicted in Figure A15, reveals that diffusion guidance-based methods fall short in performance when operating within the same computational budget and without self-recurrence. A notable observation is that the application of self-recurrence markedly improves the results, aligning them closely with those achieved through model-based conditioning. This enhancement, however, incurs a fivefold increase in computational demand.

Arbitrary Constraints: The above benchmarks have demonstrated the high accuracy potential of diffusion guidance. The effectiveness of diffusion guidance in accurately reconstructing distributions is evident from Figure A16a. Despite its general efficacy, the model exhibits minor issues, such as the slightly excessive noise observed in the two-moon scenario. These issues, however, can be mitigated through the application of self-recurrence. Figure A16b further illustrates our approach's capability to concurrently address multiple constraints while also being able to integrate model-based conditioning (every exact constrained is model-based).

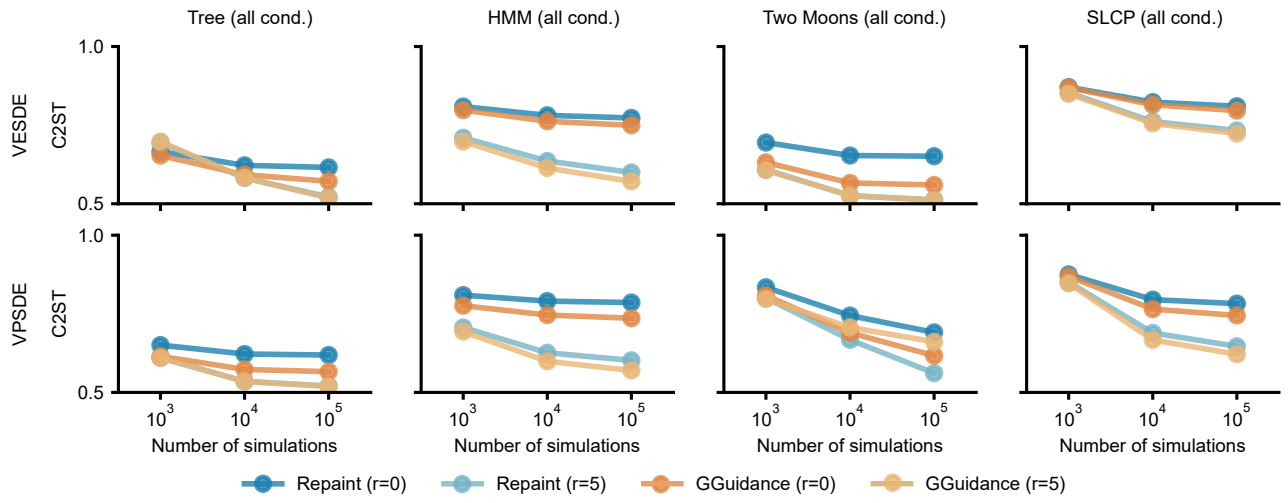

Figure A15. The Simformer exclusively trained for joint distribution estimation (i.e., M_C is always zero and thereby disables model-based conditioning). As model-based conditioning is not feasible, conditioning is implemented through diffusion guidance. This figure demonstrates the application of varying levels of self-recurrence, denoted as r, to enforce different conditions.

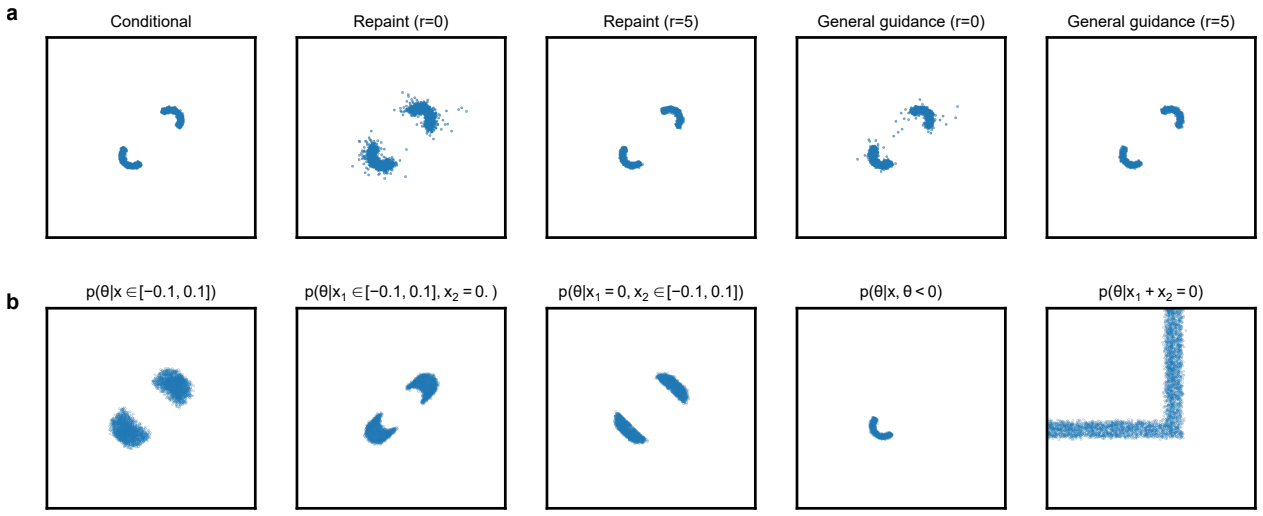

Figure A16. **(a)** Shortcomings of diffusion guidance without self recurrence $r = 0$, which can be fixed using $r = 5$. This, however, also increases the computational cost by five. **(b)** General set constraints enforced using diffusion guidance for the Two Moons tasks. The (conditional) Simformer model was trained on 10^5 simulations. Any exact condition was model-based, and any set constraint was enforced through guidance.