Truncated proposals for scalable and hassle-free simulation-based inference

Anonymous Author(s)
Affiliation
Address
email

Abstract

Simulation-based inference (SBI) solves statistical inverse problems by repeatedly running a stochastic simulator and learning posterior distributions from model-simulations. To improve simulation efficiency, several inference methods take a sequential approach and iteratively adapt the proposal distributions from which model simulations are generated. However, many of these sequential methods are difficult to use in practice, both because the resulting optimisation problems can be challenging and efficient diagnostic tools are lacking. To overcome these issues, we present Truncated Sequential Neural Posterior Estimation (TSNPE). TSNPE performs sequential inference with truncated proposals, sidestepping the optimisation issues of alternative approaches. We also develop an efficient coverage test for TSNPE that can scale to complex models with many parameters. We demonstrate that TSNPE performs on par with previous methods on established benchmark tasks. We then apply TSNPE to two challenging problems from neuroscience and show that TSNPE can successfully obtain the posterior distributions, whereas previous methods fail. Overall, our results demonstrate that TSNPE is an efficient, robust, and accurate inference method that can scale to problems that were previously inaccessible to neural posterior estimation.

1 Introduction

Computational models are an important tool to understand physical processes underlying empirically observed phenomena. These models, often implemented as numerical *simulators*, incorporate mechanistic knowledge about the physical process underling data generation, and thereby provide an interpretable model of empirical observations. In many cases, several parameters of the simulator have to be inferred from data, e.g., with Bayesian inference. However, performing Bayesian inference in these models can be difficult: Running the simulator may be computationally expensive, evaluating the likelihood-function might be computationally infeasible, and the model might not be differentiable. In order to overcome these limitations, Approximate Bayesian Computation (ABC) methods [Beaumont et al., 2002, 2009], synthetic likelihood approaches [Wood, 2010], and neural network-based methods [e.g., Papamakarios and Murray, 2016] have been developed.

A subset of neural network-based methods, known as neural posterior estimation (NPE) [Papamakarios and Murray, 2016, Lueckmann et al., 2017, Greenberg et al., 2019], train a neural density estimator on simulated data such that the density estimator directly approximates the posterior. Unlike other methods, NPE does not require any further Markov-chain Monte-Carlo (MCMC) or variational inference (VI) steps. As it provides an *amortized* approximation of the posterior, which can be used to quickly evaluate and sample the approximate posterior for any observation, this allows the application in time-critical and high-throughput inference scenarios [Gonçalves et al., 2020, Dax et al., 2021], and fast application of diagnostic methods which require posterior samples

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

for many different observations [Cook et al., 2006, Talts et al., 2018]. In addition, unlike methods targeting the likelihood (e.g., neural likelihood estimation, NLE [Papamakarios et al., 2019]), NPE can learn summary statistics from data and it can use equivariances in the simulations to improve the quality of inference [Dax et al., 2021, 2022].

If inference is performed for a particular observation $\mathbf{x}_o$, sampling efficiency of NPE can be improved with *sequential* training schemes: Instead of drawing parameters from the prior distribution, they are drawn adaptively from a proposal prior (e.g., a posterior estimate obtained with NPE) in order to optimize the posterior accuracy for a particular $\mathbf{x}_o$. These procedures are called Sequential Neural Posterior Estimation (SNPE) [Papamakarios and Murray, 2016, Lueckmann et al., 2017, Greenberg et al., 2019] and have been reported to be more simulation-efficient than training the neural network only on parameters sampled from the prior, across a set of benchmark tasks [Lueckmann et al., 2021].

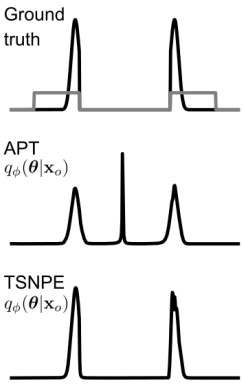

Figure 1: **APT vs TSNPE.** Top: Prior (gray) and true posterior (black). APT matches true posterior within the prior bounds but 'leaks' into region without prior support. TSNPE (ours) matches true posterior.

Despite the potential to improve simulation-efficiency, two limitations have impeded a more widespread adoption of SNPE by practitioners: First, the sequential scheme of SNPE can be unstable. SNPE requires a modification of the loss function compared to NPE, which suffers from issues that can limit its effectiveness on (or even prevent their application to) complex problems (see Sec. 2). Second, several commonly used diagnostic tools for SBI [Cook et al., 2006, Talts et al., 2018, Dalmasso et al., 2020, Hermans et al., 2021] rely on performing inference across multiple observations. In SNPE (in contrast to NPE), this requires generating new simulations and network retraining for each observation, which often prohibits the use of such diagnostic tools [Lueckmann et al., 2021].

Here, we introduce Truncated Sequential Neural Posterior Estimation (TSNPE) to overcome these limitations. TSNPE follows the SNPE formalism, but uses a proposal prior which is a *truncated* version of the prior: TSNPE draws simulations from the prior, but rejects them *before simulation* if they lie outside of the support of the approximate posterior. Thus, the proposal prior is (within its support) proportional to the prior, which allows us to train the neural network with maximum-likelihood in every round and, therefore, sidesteps the instabilities (and hence 'hassle') of previous SNPE methods. Our use of truncated proposals is strongly inspired by Blum and François [2010] (see Discussion), who proposed truncated proposals for regression-adjustment approaches in ABC. Similarly to Miller et al. [2021], TSNPE is 'locally amortized' and thus allows to run diagnostic tools at every round, without having to retrain networks. We exploit this property to develop a diagnostic tool, which we name 'simulation-based coverage calibration' (SBCC), given its close connection with simulation-based calibration (SBC, Talts et al. [2018]). SBCC evaluates the coverage of the posterior distribution computed with TSNPE without relying on integrating the approximate posterior on a grid [Dalmasso et al., 2020, Hermans et al., 2021] and without MCMC sampling. Therefore, SBCC can be run quickly and can evaluate the coverage of the approximate posterior even for high-dimensional parameter spaces.

We show that TSNPE is as efficient as SNPE on several established benchmark problems (Sec. 4.1). We then demonstrate that for two challenging neuroscience problems, TSNPE—but not previous methods—can robustly identify the posterior distributions (Sec. 4.2).

2 Background

In Neural Posterior Estimation (NPE), parameters are sampled from the prior $p(\boldsymbol{\theta})$ and simulated (i.e., $\mathbf{x}$ is sampled from $p(\mathbf{x}|\boldsymbol{\theta})$). Then, a neural density estimator $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x})$ (in our case a normalizing flow), with learnable parameters $\boldsymbol{\phi}$, is trained to minimize the loss:

$$\min_{\boldsymbol{\phi}} \mathcal{L} = \min_{\boldsymbol{\phi}} \mathbb{E}_{\boldsymbol{\theta}\sim p(\boldsymbol{\theta})} \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x}|\boldsymbol{\theta})}[-\log q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x})],$$

which is minimized if and only if, for a sufficiently expressive density estimator, $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x}) = p(\boldsymbol{\theta}|\mathbf{x})$ for all $\mathbf{x} \in \text{supp}(p(\mathbf{x}))$ [Paige and Wood, 2016, Papamakarios and Murray, 2016].

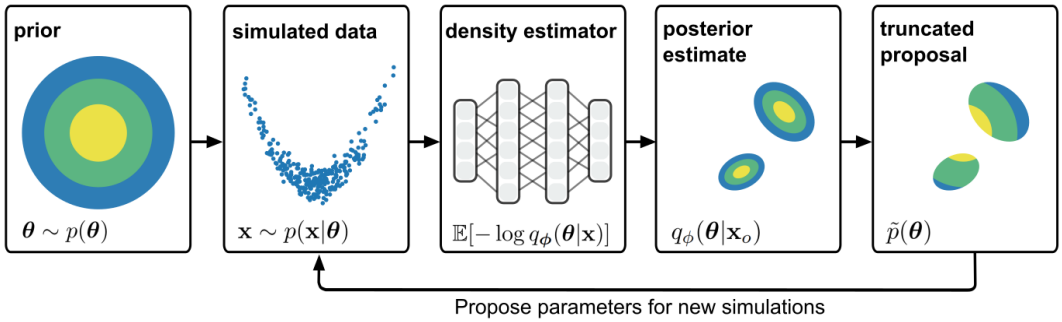

Figure 2: **Truncated Sequential Neural Posterior Estimation (TSNPE).** The method starts by sampling from the prior, running the simulator, and training a neural density estimator with maximum likelihood to approximate the posterior. In subsequent rounds, parameters are sampled from the prior, but rejected if they lie outside of the support of the approximate posterior. With these proposals, the neural density estimator can be trained with maximum likelihood in all rounds.

Sequential Neural Posterior Estimation (SNPE) aims to infer the posterior distribution $p(\boldsymbol{\theta}|\mathbf{x}_o)$ for a particular observation $\mathbf{x}_o$. SNPE initially performs NPE and, thereby, obtains an initial estimate of the posterior distribution. It then samples parameters from a proposal $\tilde{p}(\boldsymbol{\theta})$, which is often chosen to be the previously obtained estimate of the posterior $\tilde{p}(\boldsymbol{\theta}) = q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x}_o)$, and retrains the neural density estimator [Papamakarios and Murray, 2016]. This procedure can be repeated for several rounds.

Importantly, if parameters $\boldsymbol{\theta}$ are sampled from the proposal $\tilde{p}(\boldsymbol{\theta})$ rather than from the prior $p(\boldsymbol{\theta})$, the estimator $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x})$ that minimizes the maximum-likelihood loss function no longer converges to the true posterior. If one used the maximum-likelihood loss on data sampled from $\tilde{p}(\boldsymbol{\theta})$, i.e., $\mathcal{L} = \mathbb{E}_{\boldsymbol{\theta}\sim\tilde{p}(\boldsymbol{\theta})}\,\mathbb{E}_{\mathbf{x}\sim p(\mathbf{x}|\boldsymbol{\theta})}[-\log q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x})]$, then $\mathcal{L}$ would be minimized by $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x}) \propto p(\boldsymbol{\theta}|\mathbf{x})\frac{\tilde{p}(\boldsymbol{\theta})}{p(\boldsymbol{\theta})}$, which is not the true posterior. Multiple schemes have been developed to overcome this [Papamakarios and Murray, 2016, Lueckmann et al., 2017]. The most recent of these methods, Automatic Posterior Transformation (APT, or SNPE-C, in its atomic version) [Greenberg et al., 2019, Durkan et al., 2020] employs a loss that aims to classify the parameter set that generated a particular data point among other parameter sets (details in Appendix Sec. 6.5).

While APT has been reported to significantly outperform previous methods, several studies have also described cases in which the approach exhibits performance issues: Both the original APT paper [Greenberg et al., 2019] and Durkan et al. [2020] reported that APT can show 'leakage' of posterior mass outside of bounded priors. We demonstrate this issue on a simple 1-dimensional simulator with bounded prior (Fig. 1). The posterior estimated by APT is only required to match the true posterior density *within the support of the prior* (details in Appendix Sec. 6.5). Thus, after ten rounds of APT, while the approximate posterior matches the true posterior within the bounds of the prior, a substantial fraction of posterior mass lies in regions with zero prior probability. In simple models, approximate posterior samples that lie outside of the prior bounds can be efficiently rejected. However, in models with high numbers of parameters, the rejection rate can become so large that drawing posterior samples which lie inside of the prior bounds is prohibitive. For example, Glöckler et al. [2022] reported a rejection rate of more than 99.9999% in a model with 31 parameters, thus requiring approximately one minute to draw a single posterior sample from within the prior bounds.

We overcome these limitations by using 'truncated' proposal distributions. This allows us to train with maximum-likelihood at every round, thereby sidestepping issues of previous SNPE methods.

3 Methodology

3.1 Truncated proposals for SNPE

Given a particular observation $\mathbf{x}_o$, we suggest to restrict the proposals $\tilde{p}(\boldsymbol{\theta})$ to be proportional to the prior $p(\boldsymbol{\theta})$ at least in the $1-\epsilon$ highest-probability-region (HPR_ϵ, the smallest region that contains $1-\epsilon$ of the mass) of $p(\boldsymbol{\theta}|\mathbf{x}_o)$, i.e.

$$\tilde{p}(\boldsymbol{\theta}) \propto p(\boldsymbol{\theta}) \cdot \mathbb{1}_{\boldsymbol{\theta}\in\mathcal{M}}$$

Algorithm 1: TSNPE

Inputs: prior $p(\boldsymbol{\theta})$, observation $\mathbf{x}_o$, simulations per round N, number of rounds R, ϵ that defines the highest-probability region (HPR_ϵ)
Outputs: Approximate posterior $q_{\boldsymbol{\phi}}$.
Initialize: Proposal $\tilde{p}(\boldsymbol{\theta}) = p(\boldsymbol{\theta})$, dataset $\mathcal{X} = \{\}$
for $r \in [1, ..., R]$ **do**
  for $i \in [1, ..., N]$ **do**
    $\boldsymbol{\theta}_i \sim \tilde{p}(\boldsymbol{\theta})$
    simulate $\mathbf{x}_i \sim p(\mathbf{x}|\boldsymbol{\theta}_i)$
    add $(\boldsymbol{\theta}_i, \mathbf{x}_i)$ to $\mathcal{X}$
  $\boldsymbol{\phi}^* = \arg\min_{\boldsymbol{\phi}} -\frac{1}{N} \sum_{(\boldsymbol{\theta}_i, \mathbf{x}_i) \in \mathcal{X}} \log q_{\boldsymbol{\phi}}(\boldsymbol{\theta}_i|\mathbf{x}_i)$
  Optionally: compute Coverage($\tilde{p}(\boldsymbol{\theta}), q_{\boldsymbol{\phi}}$) ; `// see Alg. 2`
  $\tilde{p}(\boldsymbol{\theta}) \propto p(\boldsymbol{\theta}) \cdot \mathbb{1}_{\boldsymbol{\theta} \in \text{HPR}_\epsilon}$; `// see Alg. 3`

with $\text{HPR}_\epsilon(p(\boldsymbol{\theta}|\mathbf{x}_o)) \subseteq \mathcal{M}$. Thus, $\tilde{p}(\boldsymbol{\theta})$ is a 'truncated' proposal. The key insight is that, when using such a proposal and $\epsilon = 0$, one can train $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x})$ with maximum likelihood:

$$\min_{\boldsymbol{\phi}} \mathcal{L} = \min_{\boldsymbol{\phi}} \mathbb{E}_{\boldsymbol{\theta} \sim \tilde{p}(\boldsymbol{\theta})} \mathbb{E}_{\mathbf{x} \sim p(\mathbf{x}|\boldsymbol{\theta})}[-\log q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x})],$$

and $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x}_o)$ will still converge to $p(\boldsymbol{\theta}|\mathbf{x}_o)$ (Proof in Appendix Sec. 6.2).

We estimate $\mathcal{M}$ as the HPR_ϵ of the approximate posterior $\mathcal{M} = \text{HPR}_\epsilon(q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x}_o))$. Since the maximum likelihood loss employed to train $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x})$ is support-covering, the HPR_ϵ of $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x}_o)$ tends to cover the HPR_ϵ of $p(\boldsymbol{\theta}|\mathbf{x}_o)$ [Bishop and Nasrabadi, 2006].

In order to obtain the HPR_ϵ of $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x}_o)$, we define a threshold $\boldsymbol{\tau}$ on the likelihood of $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x}_o)$. To do so, we use a normalizing flow as $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x})$, which allows for closed-form density estimation and fast sampling. We then approximate the HPR_ϵ of the approximate posterior $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x}_o)$ as

$$\text{HPR}_\epsilon(q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x}_o)) \approx \mathbb{1}_{q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x}_o) > \boldsymbol{\tau}}.$$

We chose $\boldsymbol{\tau}$ as the ϵ-quantile of likelihoods of samples from $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x}_o)$, and evaluated TSNPE for $\epsilon = 10^{-3}$, 10^{-4}, and 10^{-5}. Using values of $\epsilon > 0$ yields a proposal prior which has smaller support than the current estimate of the posterior, e.g., using $\epsilon = 10^{-3}$ neglects 0.1% of mass from the support of the approximate posterior. Thus, this approach can lead to errors in posterior estimation, e.g., to 'under-covered' posteriors. However, as we describe later, the error introduced by this truncation is negligible, as we will demonstrate empirically on several benchmark tasks. We note that TSNPE can be trained on data pooled from all rounds (Appendix Sec. 6.11). The full algorithm is summarized in Alg. 1 (Fig. 2).

3.2 Sampling from the truncated proposal

To generate training data for subsequent rounds, we have to draw samples from the truncated proposal $\tilde{p}(\boldsymbol{\theta})$. We explored two strategies for sampling this distribution: Rejection sampling and sampling importance resampling (SIR) [Rubin, 1988]. For rejection sampling, we sample the prior $\boldsymbol{\theta} \sim p(\boldsymbol{\theta})$ and accept samples only if their probability under the approximate posterior is above the threshold $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x}) > \boldsymbol{\tau}$.

This strategy samples from the truncated proposal exactly, but can fail if the rejection rate becomes too high. To deal with these situations, we used SIR. For each sample from the truncated proposal, SIR draws K samples from the approximate posterior, computes weights $w_{i=1...K} = p(\boldsymbol{\theta}_i)\mathbb{1}_{\boldsymbol{\theta}_i \in \mathcal{M}}/q_{\boldsymbol{\phi}}(\boldsymbol{\theta}_i|\mathbf{x})$, normalizes w_i such that they sum to one, draws from a categorical distribution with weights $s \sim \text{Categorical}(w_i)$, and selects the posterior sample with index s. SIR requires a fixed sampling budget of K posterior samples per sample from the truncated proposal and returns exact samples from the truncated proposal for $K \to \infty$. In our runs, we use $K = 1024$.

3.3 Coverage diagnostic

In order for the estimated posterior $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x}_o)$ to converge to $p(\boldsymbol{\theta}|\mathbf{x}_o)$, TSNPE requires $\text{supp}(p(\boldsymbol{\theta}|\mathbf{x}_o)) \subseteq \text{HPR}_\epsilon(q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x}_o))$, i.e., the estimated posterior must be broader than the true

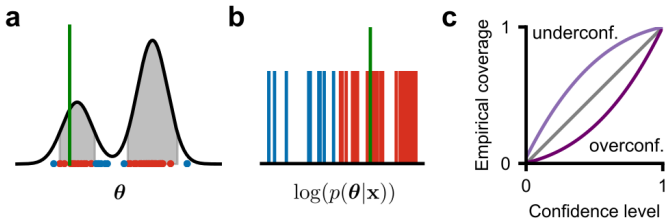

Figure 3: **Diagnostic tool.** (a) Parameter θ^* (green) lies within the 1-α confidence region (gray) of the estimated posterior. (b) $\log(p(\theta^*|x))$ is above the 1-α quantile of posterior samples. (c) 1-α versus empirical coverage, averaged over θ^*.

posterior (proof in Appendix Sec. 6.2). This can be diagnosed with the empirical coverage [Dalmasso et al., 2020, Hermans et al., 2021]. To obtain the empirical coverage, we need to compute the credible region with the smallest volume such that

$$\int_{\boldsymbol{\theta}} q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x}) d\boldsymbol{\theta} = 1 - \alpha \tag{1}$$

with confidence level $1 - \alpha$. If the posterior is well-calibrated, the true data-generating parameter $\boldsymbol{\theta}^*$ should lie within this confidence interval with probability $1 - \alpha$. In order to determine the credible region, Hermans et al. [2021] and Dalmasso et al. [2020] integrate the approximate posterior on a grid, an approach which does not scale to high-dimensional posteriors. Here, we exploit the fact that, with TSNPE, the approximate posterior can be sampled from and evaluated in closed-form, leading to a computationally efficient and scalable diagnostic.

Our diagnostic tool, simulation-based coverage calibration (SBCC), proceeds in the following way: We sample $\boldsymbol{\theta}^*$ from the prior and simulate it to generate $\mathbf{x}^*$. We then sample N times from the approximate posterior $\boldsymbol{\theta}_{i=1...N} \sim q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x}^*)$ and evaluate the log-probability of each sample $l_{i=1...N} = \log(q_{\boldsymbol{\phi}}(\boldsymbol{\theta}_{i=1...N}|\mathbf{x}^*))$ (Fig. 3a). In order to evaluate whether $\boldsymbol{\theta}^*$ lies within the credible region, we evaluate whether the log-probability of the true data-generating parameter $l^* = \log(q_{\boldsymbol{\phi}}(\boldsymbol{\theta}^*|\mathbf{x}^*))$ is larger or smaller than the $1 - \alpha$ quantile of l_i (Fig. 3b). The coverage of the approximate posterior should match the confidence level for all confidence levels $\alpha \in [0, 1]$ (Fig. 3c). The resulting method is summarized in Appendix Alg. 2.

SBCC requires to sample and evaluate the approximate posterior for multiple observations $\mathbf{x}$, which in principle could be computationally expensive. However, the use of normalizing flows for $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x})$ allows for fast sampling and density evaluation and does not require MCMC sampling (as would be required by methods based on learning the likelihood(-ratio) [Papamakarios et al., 2019, Hermans et al., 2020, Thomas et al., 2022]). Therefore, this tool can be run efficiently after every round of training, with parameters $\boldsymbol{\theta}^*$ sampled from the truncated proposal.

4 Results

We evaluated TSNPE on several benchmark tasks and on two complex problems from neuroscience. We found that TSNPE performs as well as APT on the benchmark tasks and that it is robust to choices of ϵ. In addition, we found that, in contrast with APT, TSNPE can successfully infer the posterior distribution for complex models with large numbers of parameters.

4.1 Performance on benchmark tasks

We compared TSNPE with NPE and APT on six benchmark tasks for which samples from the ground-truth posterior are available (see Appendix Sec. 6.7 for tasks) [Lueckmann et al., 2021]. We quantified the performance with a classifier two-sample test (C2ST), for which 0.5 indicates that the approximate posterior is identical to the ground-truth posterior, whereas 1.0 implies that the distributions can be completely separated by a classifier. Overall, APT and TSNPE perform similarly well and both outperform NPE (Fig. 4, left three columns). On two of the six tasks (Gaussian Linear and SLCP), APT has slightly better performance than TSNPE, whereas on two other tasks (SIR and Lotka-Volterra), TSNPE outperforms APT. Overall, TSNPE and APT perform similarly well, demonstrating that TSNPE is competitive with previous methods on benchmark tasks.

In order to get insights into the improved performance of TSNPE as compared to NPE, we computed the fraction of prior samples that lie within the HPR_ϵ of the approximate posterior (Fig. 4, fourth

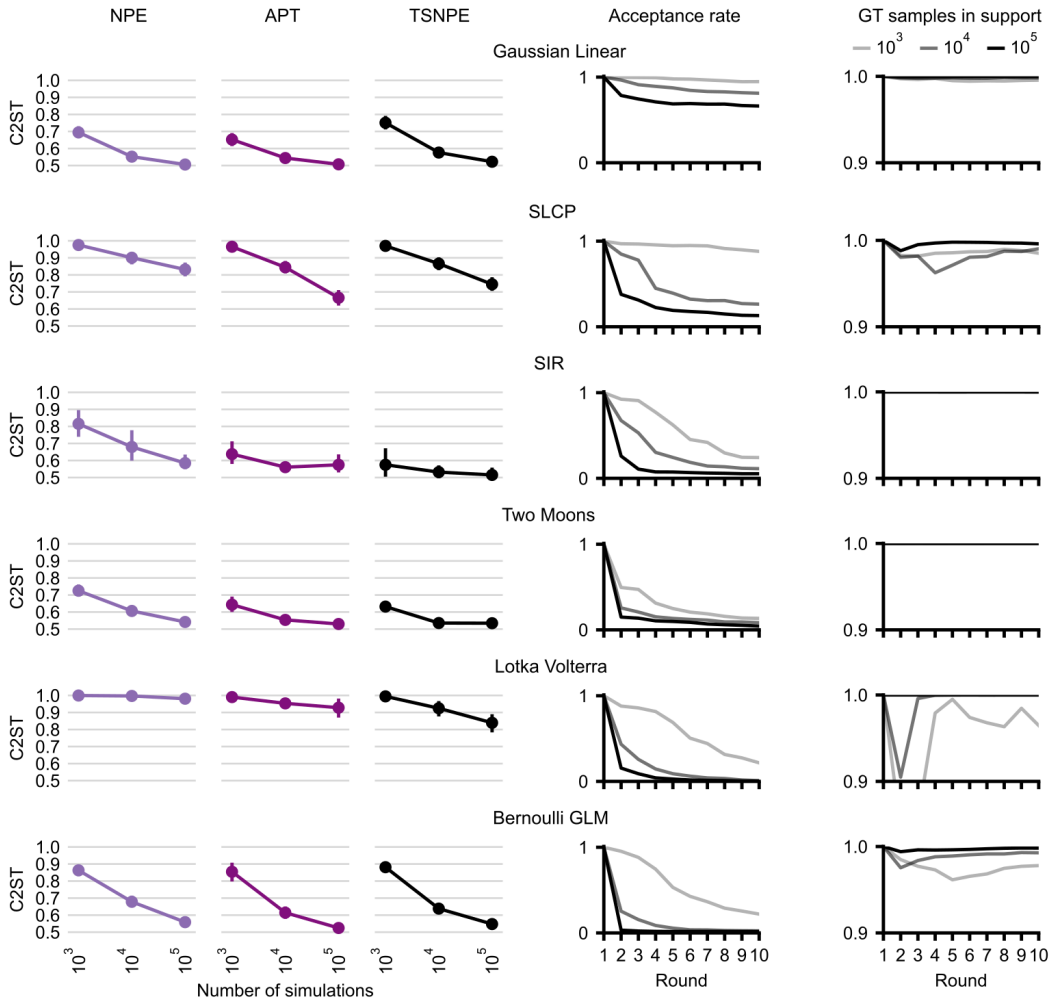

Figure 4: **Performance on six benchmark tasks.** Left three columns: classifier two-sample test accuracy (C2ST) of NPE (left), APT (middle), and TSNPE (right) for three simulation budgets. Forth column: Fraction of prior samples within the approximate-posterior HPR_ϵ in each round for each simulation budget. Fifth column: Fraction of true-posterior samples within the approximate-posterior HPR_ϵ. TSNPE with $\epsilon = 10^{-4}$ and rejection sampling from truncated proposal.

column). In tasks with broad posteriors and few simulations, the HPR_ϵ is almost as wide as the prior and thus the performances of NPE and TSNPE are similar (e.g., SLCP with 1k simulations). In other tasks and with more simulations, the HPR_ϵ is much narrower than the prior, leading to an improvement in simulation efficiency (e.g., Lotka-Volterra with 100k simulations).

Finally, we evaluated whether the HPR_ϵ of the approximate posterior contains the support of the true posterior (Fig. 4, fifth column). We computed the fraction of true-posterior samples within the HPR_ϵ of the approximate posterior. For most tasks, fewer than 0.1% of samples were excluded, and the rate of erroneously rejected samples decreased as more simulations were used. In the Lotka-Volterra task with 1k and 10k simulations, many ground-truth samples were rejected and TSNPE performed poorly, but NPE and APT also failed to solve the task. Thus, while truncated proposals can potentially induce posterior biases, these only have a negligible effect on the performance of TSNPE. We note that TSNPE performance is qualitatively unaffected by the choice of $\epsilon \leq 10^{-3}$ and proposal sampling scheme (Appendix Fig. 7, Fig. 8, Fig.9).

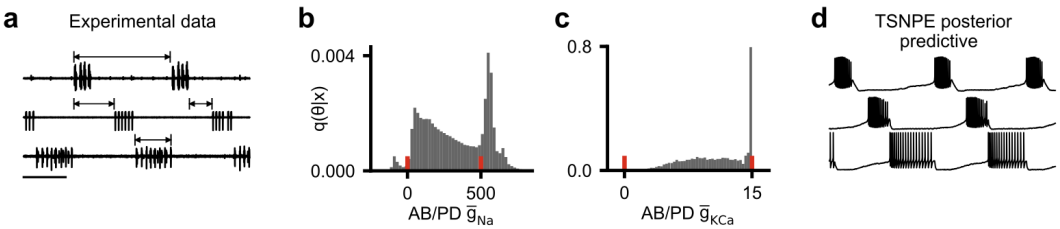

Figure 5: **Pyloric network inference.** (a) Data [Haddad and Marder, 2021]. (b) APT leakage (red: prior bounds). (c) APT failure in constrained space. (d) TSNPE posterior predictive sample.

4.2 Efficient and robust inference in two complex neuroscience problems

Next, we evaluate the performance of TSNPE on two challenging neuroscience problems, where the competitive advantage of TSNPE is fully realized.

Pyloric network We applied TSNPE to a challenging real-world simulator from neuroscience: The pyloric network of the stomatogastric ganglion in the crab *Cancer Borealis* [Prinz et al., 2003, 2004]. The model has 31 parameters and simulates 3 voltage traces that we reduce to 18 summary statistics. The prior distribution is uniform within previously described parameter ranges [Prinz et al., 2004, Gonçalves et al., 2020]. We identify the posterior distribution given experimentally observed data [Haddad and Marder, 2021] (Fig. 5a) with APT and TSNPE (13 rounds, 30k simulations per round).

When applying APT 'out of the box' (from 'sbi' toolbox [Tejero-Cantero et al., 2020]), the rate of approximate-posterior samples within the prior bounds was 0.02% after the second round and 0.0000% after the third round (Fig. 5b), which rendered a fourth round too computationally expensive.

We attempted to overcome these issues by appending a transformation T to the density estimator $q_\phi(\boldsymbol{\theta}|\mathbf{x})$ such that its support is constrained to match the support of the prior. In practice, we used a sigmoid transformation. While the resulting approximate posterior exhibited no leakage, this setup revealed another problem when running APT: In transformed (i.e., unbounded) space, the density estimator $q_\phi(\boldsymbol{\theta}|\mathbf{x})$ can put significant mass in regions outside of the training data. When forced into constrained space, these 'leaking' regions lead to spikes at the bounds of the parameter space (Fig. 5c, further details in Appendix Sec. 6.5; illustration, additional tests and full posterior in Appendix Figs. 12, 11 and 13). These spikes are at odds with previously published posterior distributions [Gonçalves et al., 2020, Deistler et al., 2021, Glöckler et al., 2022] and samples from these parameter regions do not produce good simulations (Appendix Fig. 14). This demonstrates that leakage occurs in APT even when the density estimator is forced into constrained space and that these issues lead to an incorrect posterior approximation as well as to poor predictive samples.

We applied TSNPE to this task for 13 rounds without any issue. The resulting posterior produces samples that closely match the observed data (Fig. 5d, more samples in Appendix Fig. 10, posterior distribution across all 31 parameters in Appendix Fig. 15). The obtained posterior is similar to previously published posteriors [Gonçalves et al., 2020, Deistler et al., 2021, Glöckler et al., 2022].

Multicompartment model of a single neuron Finally, we turn to a landmark problem in neuroscience for which the posterior has not yet been identified: A morphologically detailed model of a thick-tufted layer 5 pyramidal cell (L5PC) from the neocortex [Ramaswamy et al., 2015, Markram et al., 2015, Van Geit et al., 2016]. The model describes the response of a neuron to current stimuli of different strengths. The model has approximately 7000 separate compartments which compose the anatomy of the cell (Fig. 6a). Each compartment has dynamics based on the Hodgkin-Huxley equations [Hodgkin and Huxley, 1952] and contains multiple ion channels (details in Van Geit et al. [2016]). The model has 20 free parameters which are the maximal channel conductances and time constants of the ion channels. The simulation consists of four separate simulations corresponding to experimental protocols which describe the voltage response to different stimuli. For the first three protocols (Step 1, Step 2, Step 3), each voltage response is characterized by 10 summary statistics. The fourth protocol models the back-propagation of the voltage response through the dendritic tree and is captured by 5 additional summary statistics (bAP soma, bAP dend. 1, bAP dend. 2). In total,

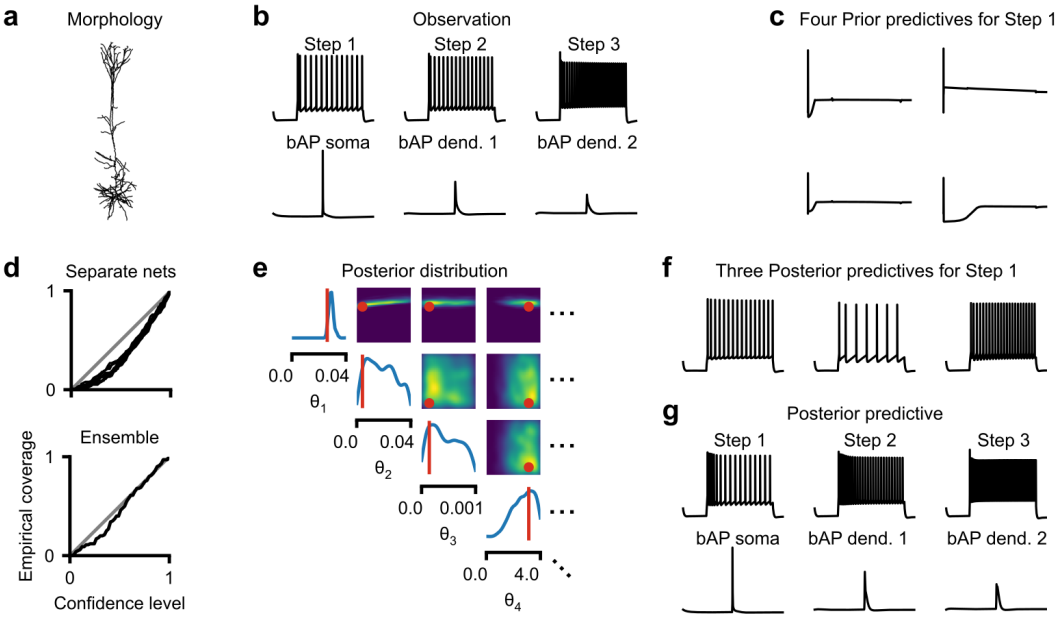

Figure 6: **TSNPE on L5PC.** (a) Cell morphology. (b) Observation. (c) Four prior samples for the step protocol. (d) Coverage for 1 and 10 neural nets. (e) Posterior. True parameter in red. (f) Three posterior predictives for the Step 1 protocol. (g) One posterior predictive for all protocols.

the model produces 35 summary statistics, to which we add Gaussian noise with diagonal covariance matrix capturing the response variability of previously reported measurements [Hay et al., 2011].

Our goal is to infer the posterior distribution over 20 parameters given 35 summary statistics that were simulated—and thus have a known ground-truth parameter set— to resemble experimentally observed activity (Fig. 6b). The prior is a uniform distribution within previously established bounds [Van Geit et al., 2016]. A major difficulty in fitting this model is that a large fraction of prior samples generate summary statistics that are very different from the observed data. In particular, about 99.98% of prior predictives contain at least one summary statistic that is undefined (e.g., time to first spike is undefined in the absence of spikes) (Fig. 6c). When a summary statistic is undefined, we assign it a value that is substantially outside the range of the observed data (Appendix Sec. 6.9).

We ran TSNPE over six rounds (see Appendix Sec. 6.10 for details on hyperparameters). In each round, we ran 30k simulations, leading to a total of 180k simulations. After every round, we evaluated the coverage with SBCC. After the first round, the approximate posterior exhibited poor coverage (Fig. 6d top). Therefore, as suggested by Hermans et al. [2021], we used an ensemble of 10 neural density estimators to ensure that the approximate posterior is sufficiently broad (Fig. 6d bottom).

The TSNPE-posterior has several parameters with broad marginals, demonstrating that this model exhibits 'degeneracy', a widespread phenomenon in biological systems [Marder and Taylor, 2011] (Fig. 6e, Appendix Sec. 6.9 for parameter names). Other marginals are narrow, demonstrating that the model is sensitive to changes in these parameters. Posterior predictive samples closely match the observed data (Fig. 6f,g; more samples in Appendix Fig. 17). We emphasize that fitting such morphologically-detailed neuron models is a challenging and widespread problem in neuroscience, one for which commonly used methods [e.g., genetic algorithms, Druckmann et al., 2007, Van Geit et al., 2016] are often simulation-inefficient or do not estimate the full posterior distribution. We show promising results, suggesting that TSNPE could be applied to other complex single-neuron models.

In contrast, with 'out of the box' APT, none of the 10M approximate-posterior samples was within the prior bounds after the second round. This rendered a third round too computationally expensive.

Overall, the results on the pyloric network model and on the multicompartment model demonstrate that TSNPE is an efficient and robust method that scales to complex and high-dimensional models that were inaccessible to previous neural posterior estimation methods.

5 Discussion

We presented a new method to perform Bayesian inference in implicit models, which we call Truncated Sequential Neural Posterior Estimation (TSNPE). Like previous methods, TSNPE adaptively selects parameters to improve simulation-efficiency and allow posterior inference in complex models with many parameters. The key ingredient is that TSNPE samples parameters from a 'truncated' region of the prior, and thus overcomes instabilities of previous methods while maintaining simulation efficiency. In order to diagnose potential errors of TSNPE, we developed a coverage test that can be run quickly and at every round of TSNPE. TSNPE presents a new variant of SNPE which is at least as powerful as previous variants on benchmark tasks, but provides a powerful alternative which is able to solve inference problems on which previous methods have failed.

Related work TSNPE differs from automatic posterior transformation (APT, SNPE-C) in its proposal and its loss function: TSNPE uses a truncated prior as proposal, while APT can flexibly use any proposal, which, e.g., allows for more sophisticated active learning rules [Lueckmann et al., 2019, Järvenpää et al., 2019]. However, APT's flexibility requires a modification of NPE loss function, which can be an impediment to its usage in practice: First, the modification can lead to 'leakage', which can make it prohibitive to draw samples within prior bounds. Second, APT loss requires an explicit prior and thus, cannot be applied to models in which the prior can only be sampled [Ramesh et al., 2022]. Third, current formulations of APT cannot discard parameters leading to invalid simulations as the posterior mass would 'leak' into parameter regions which only produce invalid simulations (Appendix Sec. 6.5). It might be possible that these issues are resolved using a modified formulation of APT, e.g., by combining its atomic loss with additional loss terms, or preventing leakage by penalizing 'bad' parameters [Greenberg et al., 2019]. In cases in which leakage prevents application of APT (in particular, in high-dimensional problems), TSNPE provides an alternative.

Our method is inspired by previous work that introduced a mechanism to post-hoc correct samples obtained by an Approximate Bayesian Computation (ABC) algorithm [Blum and François, 2010], i.e., 'regression adjustment ABC'. Their method draws samples from a truncated region of the prior to avoid correction terms, but estimates the posterior density with ABC samples—rather than using a flexible neural density estimator—and estimates the support by training a dedicated support-vector machine. In addition, the method runs a single round of truncation and retraining, whereas we demonstrate that TSNPE can be robustly applied across 10 rounds.

Truncated proposals have also recently been proposed with an orthogonal aim to TSNPE [Truncated Marginal Neural Ratio Estimation (TMNRE) Miller et al., 2021]: TMNRE uses truncated proposals to efficiently infer selected posterior marginals while being amortised around the observation ('local amortization'), allowing to test the coverage properties of the selected marginals, e.g., with [simulation-based calibration Cook et al., 2006, Talts et al., 2018]. In contrast, TSNPE aims at efficiently inferring the full posterior distribution by proposals that avoid the correction of SNPE loss function. While TSNPE shares the properties of 'local amortization' of TMNRE, its full posterior coverage can also be tested (with SBCC).

Simulation-based coverage calibration The main failure mode of our method occurs if the HPR_ϵ of the approximate posterior is too narrow relative to the support of the true posterior. In this case, TSNPE neglects relevant regions in parameter space. In order to diagnose such failure, we developed SBCC, a coverage test for TSNPE. SBCC can evaluate the coverage of the approximate posterior without integrating it on a grid [Dalmasso et al., 2020, Hermans et al., 2021] and, unlike diagnostic tools for methods based on likelihood(-ratio) learning, does not require MCMC runs for multiple observations $\mathbf{x}$. These properties allow SBCC to be run efficiently at every round of TSNPE and for models with many parameters. In addition, in contrast to diagnostic tools for likelihood-free inference with Approximate Bayesian Computation, SBCC does not require an additional step to estimate the density of approximate posterior samples [Prangle et al., 2014]. We note that SBCC is closely related to simulation-based calibration [Cook et al., 2006] (and is identical under certain conditions, Appendix Sec. 6.6). In principle, SBCC could be applied to other SNPE variants, although empirically the impact of arbitrary proposals on SBCC performance is currently unclear.

Conclusion TSNPE combines the simulation-efficiency of sequential neural posterior estimation with the robustness and coverage-tests of non-sequential methods, allowing to scale neural posterior estimation to complex and high-dimensional models that were inaccessible to previous methods.

References

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

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

Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.

Michael GB Blum and Olivier François. Non-linear regression models for approximate bayesian computation. *Statistics and Computing*, 20(1):63–73, 2010.

Bob Carpenter, Andrew Gelman, Matthew D Hoffman, Daniel Lee, Ben Goodrich, Michael Betancourt, Marcus Brubaker, Jiqiang Guo, Peter Li, and Allen Riddell. Stan: A probabilistic programming language. *Journal of statistical software*, 76(1), 2017.

Samantha R Cook, Andrew Gelman, and Donald B Rubin. Validation of software for bayesian models using posterior quantiles. *Journal of Computational and Graphical Statistics*, 15(3):675–692, 2006.

Niccolò Dalmasso, Taylor Pospisil, Ann B Lee, Rafael Izbicki, Peter E Freeman, and Alex I Malz. Conditional density estimation tools in python and r with applications to photometric redshifts and likelihood-free cosmological inference. *Astronomy and Computing*, 30:100362, 2020.

Maximilian Dax, Stephen R Green, Jonathan Gair, Jakob H Macke, Alessandra Buonanno, and Bernhard Schölkopf. Real-time gravitational wave science with neural posterior estimation. *Physical review letters*, 127(24):241103, 2021.

Maximilian Dax, Stephen R Green, Jonathan Gair, Michael Deistler, Bernhard Schölkopf, and Jakob H. Macke. Group equivariant neural posterior estimation. In *International Conference on Learning Representations*, 2022.

Michael Deistler, Jakob H Macke, and Pedro J Gonçalves. Energy efficient network activity from disparate circuit parameters. *bioRxiv*, 2021.

Shaul Druckmann, Yoav Banitt, Albert A Gidon, Felix Schürmann, Henry Markram, and Idan Segev. A novel multiple objective optimization framework for constraining conductance-based neuron models by experimental data. *Frontiers in Neuroscience*, 1:1, 2007.

Conor Durkan, Artur Bekasov, Iain Murray, and George Papamakarios. Neural spline flows. *Advances in neural information processing systems*, 32, 2019.

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

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

Pedro J Gonçalves, Jan-Matthis Lueckmann, Michael Deistler, Marcel Nonnenmacher, Kaan Öcal, Giacomo Bassetto, Chaitanya Chintaluri, William F Podlaski, Sara A Haddad, Tim P Vogels, et al. Training deep neural density estimators to identify mechanistic models of neural dynamics. *Elife*, 9:e56261, 2020.

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

Sara Ann Haddad and Eve Marder. Recordings from the c. borealis stomatogastric nervous system at different temperatures in the decentralized condition. *URL https://doi.org/10.5281/zenodo*, 5139650, 2021.

Etay Hay, Sean Hill, Felix Schürmann, Henry Markram, and Idan Segev. Models of neocortical layer 5b pyramidal cells capturing a wide range of dendritic and perisomatic active properties. *PLoS computational biology*, 7(7):e1002107, 2011.

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

Joeri Hermans, Arnaud Delaunoy, François Rozet, Antoine Wehenkel, and Gilles Louppe. Averting a crisis in simulation-based inference. *arXiv preprint arXiv:2110.06581*, 2021.

Alan L Hodgkin and Andrew F Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of physiology*, 117(4):500, 1952.

Marko Järvenpää, Michael U Gutmann, Arijus Pleska, Aki Vehtari, and Pekka Marttinen. Efficient acquisition rules for model-based approximate bayesian computation. *Bayesian Analysis*, 14(2): 595–622, 2019.

William Ogilvy Kermack and Anderson G McKendrick. 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.

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

Jan-Matthis Lueckmann, Giacomo Bassetto, Theofanis Karaletsos, and Jakob H Macke. Likelihood-free inference with emulator networks. In *Symposium on Advances in Approximate Bayesian Inference*, pages 32–53. PMLR, 2019.

Jan-Matthis Lueckmann, Jan Boelts, David Greenberg, Pedro Goncalves, and Jakob Macke. Benchmarking simulation-based inference. In *International Conference on Artificial Intelligence and Statistics*, pages 343–351. PMLR, 2021.

Eve Marder and Adam L Taylor. Multiple models to capture the variability in biological neurons and networks. *Nature neuroscience*, 14(2):133–138, 2011.

Henry Markram, Eilif Muller, Srikanth Ramaswamy, Michael W Reimann, Marwan Abdellah, Carlos Aguado Sanchez, Anastasia Ailamaki, Lidia Alonso-Nanclares, Nicolas Antille, Selim Arsever, et al. Reconstruction and simulation of neocortical microcircuitry. *Cell*, 163(2):456–492, 2015.

Benjamin Miller, Alex Cole, Patrick Forré, Gilles Louppe, and Christoph Weniger. Truncated marginal neural ratio estimation. *Advances in Neural Information Processing Systems*, 34, 2021.

Brooks Paige and Frank Wood. Inference networks for sequential monte carlo in graphical models. In *International Conference on Machine Learning*, pages 3040–3049. PMLR, 2016.

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

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

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

Dennis Prangle, Michael GB Blum, G Popovic, and SA Sisson. Diagnostic tools for approximate bayesian computation using the coverage property. *Australian & New Zealand Journal of Statistics*, 56(4):309–329, 2014.

Astrid A Prinz, Cyrus P Billimoria, and Eve Marder. Alternative to hand-tuning conductance-based models: construction and analysis of databases of model neurons. *Journal of neurophysiology*, 2003.

Astrid A Prinz, Dirk Bucher, and Eve Marder. Similar network activity from disparate circuit parameters. *Nature neuroscience*, 7(12):1345–1352, 2004.

Srikanth Ramaswamy, Jean-Denis Courcol, Marwan Abdellah, Stanislaw R Adaszewski, Nicolas Antille, Selim Arsever, Guy Atenekeng, Ahmet Bilgili, Yury Brukau, Athanassia Chalimourda, et al. The neocortical microcircuit collaboration portal: a resource for rat somatosensory cortex. *Frontiers in neural circuits*, 9:44, 2015.

Poornima Ramesh, Jan-Matthis Lueckmann, Jan Boelts, Álvaro Tejero-Cantero, David S. Greenberg, Pedro J. Goncalves, and Jakob H. Macke. GATSBI: Generative adversarial training for simulation-based inference. In *International Conference on Learning Representations*, 2022.

Donald B Rubin. Using the sir algorithm to simulate posterior distributions. *Bayesian statistics*, 3: 395–402, 1988.

Sean Talts, Michael Betancourt, Daniel Simpson, Aki Vehtari, and Andrew Gelman. Validating bayesian inference algorithms with simulation-based calibration. *arXiv preprint arXiv:1804.06788*, 2018.

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

Owen Thomas, Ritabrata Dutta, Jukka Corander, Samuel Kaski, and Michael U Gutmann. Likelihood-free inference by ratio estimation. *Bayesian Analysis*, 17(1):1–31, 2022.

Werner Van Geit, Michael Gevaert, Giuseppe Chindemi, Christian Rössert, Jean-Denis Courcol, Eilif B Muller, Felix Schürmann, Idan Segev, and Henry Markram. Bluepyopt: leveraging open source software and cloud infrastructure to optimise model parameters in neuroscience. *Frontiers in neuroinformatics*, page 17, 2016.

Peter J Wangersky. Lotka-volterra population models. *Annual Review of Ecology and Systematics*, 9 (1):189–218, 1978.

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

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

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

6 Appendix

6.1 Reproducibility statement

We used the configuration manager hydra to track the configuration and seeds of each run [Yadan, 2019]. All code to reproduce the results can be found at `https://github.com/tsnpeNeurips`. We implemented TSNPE on top of the publicly accessible sbi toolbox [Tejero-Cantero et al., 2020]. All simulations and runs were performed on a high-performance computer. For each run, we used between 8 and 48 CPU cores.

6.2 Proof of convergence

Below, we proof that the posterior distribution obtained with TSNPE converges to the true posterior distribution under the assumption that the HPR_ϵ of the approximate posterior covers the support (i.e., the HPR_ϵ for $\epsilon = 0$) of the true posterior.

$$\begin{aligned}\mathcal{L} &= -\frac{1}{N}\sum_i \log(q_{\boldsymbol{\phi}}(\boldsymbol{\theta}_i|\mathbf{x}_i)) \\ &\xrightarrow{N\to\infty} -\mathbb{E}_{p(\boldsymbol{\theta},\mathbf{x})}[\log(q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x}))] \\ &= -\mathbb{E}_{\tilde{p}(\boldsymbol{\theta})p(\mathbf{x}|\boldsymbol{\theta})}[\log(q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x}))],\end{aligned}$$

where $\tilde{p}(\boldsymbol{\theta})$ is the proposal distribution from which $\boldsymbol{\theta}$ are drawn. In the first round, we use the prior $\tilde{p}(\boldsymbol{\theta}) = p(\boldsymbol{\theta})$. In later rounds, we sample from the prior but reject samples that lie outside of the HPR_ϵ of the approximate posterior. Thus $\tilde{p}(\boldsymbol{\theta}) = U(\boldsymbol{\theta})p(\boldsymbol{\theta})/Z_1$, where $U(\boldsymbol{\theta})$ is 1 on the support of the approximate posterior and Z_1 being the normalization constant. Outside of the support of the approximate posterior, $U(\boldsymbol{\theta})$ is allowed to be anything for the proof to hold. Then:

$$\begin{aligned}\mathcal{L} &= -\iint \frac{1}{Z_1}U(\boldsymbol{\theta})p(\boldsymbol{\theta})p(\mathbf{x}|\boldsymbol{\theta})\log(q_\phi(\boldsymbol{\theta}|\mathbf{x}))\, d\boldsymbol{\theta}d\mathbf{x} \\ &= \int p(\mathbf{x})\int -\frac{1}{Z_1}U(\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathbf{x})\log(q_\phi(\boldsymbol{\theta}|\mathbf{x}))\, d\boldsymbol{\theta}d\mathbf{x}\end{aligned}$$

The term within the integral over $\boldsymbol{\theta}$ is proportional to the Kullback-Leibler-divergence between $U(\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathbf{x})/Z_2$ (with $Z_2 = \int U(\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathbf{x})d\boldsymbol{\theta}$) and the approximate posterior $q_\phi(\boldsymbol{\theta}|\mathbf{x})$. Thus, $\mathcal{L}$ is minimized if and only if $q_\phi(\boldsymbol{\theta}|\mathbf{x}) = U(\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathbf{x})/Z_2$ for all $\mathbf{x}$ within the the support of $p(\mathbf{x})$.

Under the assumption that $U(\boldsymbol{\theta}) = 1$ at least on the support of the true posterior $p(\boldsymbol{\theta}|\mathbf{x}_o)$, we obtain $q_\phi(\boldsymbol{\theta}|\mathbf{x}_o) = p(\boldsymbol{\theta}|\mathbf{x}_o)$ because for all $\boldsymbol{\theta}$ where $p(\boldsymbol{\theta}|\mathbf{x}_o) > 0$, we have $U(\boldsymbol{\theta}) = 1$. Notice that, for any $\boldsymbol{\theta}$ for which $p(\boldsymbol{\theta}|\mathbf{x}_o) = 0$, $U(\boldsymbol{\theta})$ can be anything without affecting $U(\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathbf{x})/Z_2$. This allows us to use data pooled from several rounds of simulations (Sec. 6.11).

This proof also demonstrates that $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x})$ converges to the true posterior only for those $\mathbf{x}$ whose posterior support is contained within the support of the posterior given $\mathbf{x}_o$.

6.3 Simulation-based coverage calibration (SBCC)

The algorithm for computing the coverage (SBCC) is shown in Alg. 2.

6.4 Algorithm description for sampling from the HPR of the approximate posterior

The algorithm for sampling from the HPR_ϵ of the approximate posterior is shown in Alg. 3.

6.5 Alleviating issues of APT

We compared TSNPE to Automatic Posterior Transformation (APT) [Greenberg et al., 2019]. In this section, we will briefly review how APT works, and why 'leakage' occurs, and how we attempted to improve it.

Algorithm 2: Simulation-based coverage calibration (SBCC)

Inputs: proposal $\tilde{p}(\boldsymbol{\theta})$, approximate posterior $q_{\boldsymbol{\phi}}$ trained on $\boldsymbol{\theta} \sim \tilde{p}(\boldsymbol{\theta})$, number of simulations M, number of posterior samples per simulation P.
Outputs: Coverage for different confidence levels $1 - \alpha$.
for $i \in [1, ..., M]$ **do**
  $\boldsymbol{\theta}_i^* \sim \tilde{p}(\boldsymbol{\theta})$
  simulate $\mathbf{x}_i^* \sim p(\mathbf{x}|\boldsymbol{\theta}_i^*)$
  $l_i^* = \log(q_{\boldsymbol{\phi}}(\boldsymbol{\theta}_i^*|\mathbf{x}_i^*))$
  $c = 0$
  for $j \in [1, ..., P]$ **do**
    $\boldsymbol{\theta}_j \sim q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x}_i)$
    $l_j = \log(q_{\boldsymbol{\phi}}(\boldsymbol{\theta}_j|\mathbf{x}_i)$
    if $l_j > l_i^*$ **then**
      $c = c + 1$
    $e = c/P$
    add e to $\mathcal{E}$
Plot CDF of e

Algorithm 3: Obtaining the HPR$_\epsilon$ of the approximate posterior with rejection sampling

Inputs: Prior $p(\boldsymbol{\theta})$, Approximate posterior $q_{\boldsymbol{\phi}}$, observation $\mathbf{x}_o$, number of posterior samples M, ϵ that defines the HPR, number of desired samples from $\tilde{p}(\theta)$ N.
Outputs: Samples $\mathcal{S}$ from $\tilde{p}(\boldsymbol{\theta})$.
for $i \in [1, ..., M]$ **do**
  $\boldsymbol{\theta}_i \sim q_{\boldsymbol{\phi}}(\boldsymbol{\theta}\ \mathbf{x}_o)$
  $l_i = \log(q_{\boldsymbol{\phi}}(\boldsymbol{\theta}_i|\mathbf{x}_o)$
  add l_i to $\mathcal{P}$
$\boldsymbol{\kappa}$ = quantile$(\mathcal{P}, n)$
while $s < N$ **do**
  $\boldsymbol{\theta} \sim p(\boldsymbol{\theta})$
  $l = \log(q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x}_o))$
  if $l > \boldsymbol{\kappa}$ **then**
    add $\boldsymbol{\theta}$ to $\mathcal{S}$
    s += 1

APT review Broadly, APT exists in two versions. Its first version can be applied only if the density estimator $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x})$ is a mixture of Gaussians, the proposal $\tilde{p}(\boldsymbol{\theta})$ is a mixture of Gaussians, and the prior $p(\boldsymbol{\theta})$ is either uniform or Gaussian. We did not compare TSNPE to APT in this form because we wanted to use more expressive density estimators for $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x})$. The second version of APT, known as atomic APT, allows to use any density estimator $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x})$, any proposal $\tilde{p}(\boldsymbol{\theta})$ and any explicit prior $p(\boldsymbol{\theta})$. One must be able to evaluate the density estimator and the prior, but the proposal can be implicit (i.e. have no closed form density). Atomic APT minimizes the loss:

$$\mathcal{L}_{\boldsymbol{\phi}} = -\frac{1}{N}\sum_{i=1}^{N} \log \frac{q_{\boldsymbol{\phi}}(\boldsymbol{\theta}_i|\mathbf{x}_i)/p(\boldsymbol{\theta}_i)}{q_{\boldsymbol{\phi}}(\boldsymbol{\theta}_i|\mathbf{x}_i)/p(\boldsymbol{\theta}_i) + \sum_{j=1...A-1} q_{\boldsymbol{\phi}}(\boldsymbol{\theta}_j|\mathbf{x}_i)/p(\boldsymbol{\theta}_j)}$$

with number of atoms A. In this loss, $\boldsymbol{\theta}_j$ and $\boldsymbol{\theta}_i$ can be sampled from *any* proposal distribution. $\mathbf{x}_i$ is sampled from $p(\mathbf{x}_i|\boldsymbol{\theta}_i)$.

Leakage issue Notice that the above loss will be the same for $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x})$ and for $c \cdot q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x})$. In other words, the approximate posterior has to be correct only up to proportionality [Greenberg et al., 2019, Durkan et al., 2020]. Thus, an approximate posterior $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x})$ which is $c \cdot p(\boldsymbol{\theta}|\mathbf{x})$ will have a minimal loss. Since the dataset on which $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x})$ is trained contains no $\boldsymbol{\theta}$ that lie outside of the prior bounds, the approximate posterior can be anything without affecting the value of the loss function. This is what is called 'leakage', and which has been pointed out as a potential problem both in the original APT paper Greenberg et al. [2019] and work studying the relationship of APT with contrastive learning approaches Durkan et al. [2020]: While the approximate posterior might be proportional to

the true posterior within the bounds of the prior $p(\boldsymbol{\theta})$, it can put significant mass outside of the prior bounds (because the loss does not penalize this behaviour).

Transforming the parameter space We tried to fix the leakage issue by appending a transformation such that the density estimator has constrained support. This requires that the bounds of the parameter space are known and that such a transformation can be implemented. If this fix can be applied, the leakage will be zero by definition. In all our experiments, such a transformation substantially helped with the leakage problem.

Leakage into regions of no training data However, even when possible, such transformation does not completely prevent 'leakage': When inspecting the loss of APT, we see that the approximate posterior can put mass into any region of the parameter space in which no parameter sets $\boldsymbol{\theta}$ in the training data lie. For example, if the prior distribution is standard Gaussian and one trains APT on 100 parameter sets sampled from the prior, the training dataset will unlikely contain values that are smaller than -3 or larger than 3. Therefore, APT can 'leak' into the regions below -3 or above 3 and still have an optimal loss. While, in the limit of infinite data, APT would correct its 'leakage' as soon as parameters from these regions are used as training data, given a finite number of simulations the approximate posterior can 'leak' into new regions which have not been explored (yet). The (potential) problem of leakage does not affect all applications equally: High-dimensional parameter spaces suffer more from this behavior than low-dimensional ones, as there are many regions into which the mass of the approximate posterior can 'leak'. This behaviour is illustrated in Appendix Fig. 12.

Leakage because of invalid data Another way in which leakage can occur is if the simulator produces invalid data (e.g. NaN or infinity). Often, such invalid simulations are discarded from the training dataset [Lueckmann et al., 2017] and the approximate posterior is trained only on samples that produce valid outputs. For example, assume that a small region of the prior always produces invalid simulations. In this case, the approximate posterior will never be trained on simulations from this parameter region and APT can 'leak' into this region. Thus, the approximate posterior might contain significant mass in parameter regions that produce invalid simulations.

Explicit recommendations for running APT We will now give explicit recommendations for running APT. These modification greatly improved the performance of APT in our experiments, but were not able to avoid the failure of the algorithm on challenging real-world problems such as the pyloric network (Fig. 5).

1. For priors with bounded supports: If possible, transform the parameter space into unbounded space.
2. Do not discard invalid simulations. Instead, replace invalid entries (such as NaN) with an unreasonably large (or small) value and train on all available simulations.
3. Check if the posterior contains a lot of mass in regions with very low prior probability. If this is the case, it can hint at a failure of APT through leakage into regions of no training data.
4. If you transformed the parameter space, check if many posterior samples lie very close to the bounds. Again, this can hint a failure of APT through leakage into regions of no training data.

6.6 Relation between simulation-based calibration and our diagnostic

As written in the discussion, our method is closely related to simulation-based calibration (SBC) [Cook et al., 2006, Talts et al., 2018]. Briefly, SBC samples $\boldsymbol{\theta}^*$ from the prior, samples the likelihood given this $\mathbf{x} \sim p(\mathbf{x}|\boldsymbol{\theta}^*)$ and then draws samples from the posterior $\boldsymbol{\theta}_i \sim p(\boldsymbol{\theta}|\mathbf{x})$ (e.g. with MCMC). It then projects the (potentially high-dimensional) parameters $\boldsymbol{\theta}_i$ into a one-dimensional space $T(\boldsymbol{\theta}_i)$. Often, this projection is the 1D-marginal distribution of parameters [Carpenter et al., 2017]. It then ranks $T(\boldsymbol{\theta}^*)$ under all posterior samples $T(\boldsymbol{\theta}_{1...N})$. Repeated across several prior samples, the distribution of ranks should be uniform. For high-dimensional parameter spaces, the marginal distribution of each parameter is checked independently.

Notably, other projections into a one-dimensional space are possible. Below, we explain that our method is identical to running SBC with projection $T : \boldsymbol{\theta} \rightarrow q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x})$ and with posterior samples exactly following $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x})$.

In SBC and in our diagnostic method, samples are drawn from the prior $\boldsymbol{\theta}^* \sim p(\boldsymbol{\theta})$ and simulated $\mathbf{x}^* \sim p(\mathbf{x}|\boldsymbol{\theta})$. In our diagnostic method, as well as in SBC with projection being the log-probability of the approximate posterior, one then samples the posterior to obtain $\boldsymbol{\theta}_i$ and evaluates the log-probability of all samples, i.e. $l_i = T(\boldsymbol{\theta}_i) = \log(q(\boldsymbol{\theta}_i|\mathbf{x}^*)$ as well as of the initial parameter set $l^* = \log(q(\boldsymbol{\theta}^*|\mathbf{x}^*)$. SBC then ranks l^* under all l_i, which is equivalent to computing its quantile (as in our method). In our diagnostic tool, one then evaluates whether this quantile is above or below several confidence levels (evaluation on a 1D evenly spaced grid, same as binning of the ranks in SBC). This generates a step-function with the step occuring at the quantile of l^*. This step function is the cumulative distribution function of a dirac at the quantile of l^*. Therefore, repeated across several $\boldsymbol{\theta}^*$, our coverage plots (e.g. Fig 3c) correspond to the cumulative distribution function of the histograms generated by SBC (with projection $T : \boldsymbol{\theta} \rightarrow q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x})$ and posterior samples exactly following $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x})$).

6.7 Benchmark tasks

Below, we briefly describe the benchmark tasks. For details, please see Lueckmann et al. [2021].

Gaussian linear: 10 parameters which are the mean of a Gaussian model. The prior is Gaussian, resulting in a Gaussian posterior.

Bernoulli GLM: Generalized linear model with Bernoulli observations. Inference is performed on 10-dimensional sufficient summary statistics of the originally 100 dimensional raw data. The resulting posterior is 10-dimensional, unimodal, and concave.

Lotka Volterra: A traditional model in ecology [Wangersky, 1978], which describes a predator-prey interaction between species, illustrating a task with complex likelihood and unimodal posterior.

SLCP: A task introduced by Papamakarios et al. [2019] with a simple likelihood and complex posterior. The prior is uniform, the likelihood has Gaussian noise but is non-linearly related to the parameters, resulting in a posterior with four symmetrical modes.

Two moons: This model has two parameters with a uniform prior. The simulator is non-linear, generating a posterior with both local and global (bimodal) structure [Greenberg et al., 2019].

SIR: Epidemiological model with two parameters and ten summary statistics [Kermack and McKendrick, 1927].

6.8 Toy model

The toy model shown in Fig. 1 is given by a uniform prior within $[-2, -1]$ and $[1, 2]$. The simulator is $x \sim \theta^2 + \epsilon$, where ϵ is a Gaussian distribution with mean zero and standard deviation 0.2. We ran APT [Greenberg et al., 2019] and TSNPE for 10 rounds with 500 simulations per round. For APT, all hyperparameters are the default values from the sbi package [Tejero-Cantero et al., 2020], but we used a neural spline flow (NSF) for both APT and TSNPE [Durkan et al., 2019].

6.9 Computational model of single-neuron dynamics

We performed Bayesian inference in a complex model of single-neuron dynamics. The model is the same as used in Van Geit et al. [2016]. We added observation noise with standard deviations taken from previously published measurements [Hay et al., 2011]. The prior is a uniform distribution within the same bounds as previously used [Van Geit et al., 2016]. The summary statistics are also the same as in Van Geit et al. [2016].

We replaced NaN values with the value of the minimal value among prior samples minus two times the standard deviation of prior samples. Several summary statistics had heavy tailed distributions, which lead to a very high standard deviation. For these summary statistics, we picked the replacement value by hand. The final values are shown in table 6.9.

The parameters are shown in table 6.9.

Summary statistic	Observation	Replacement value
step1_soma_ahp_depth_abs	-62.1358	-110.974
step1_soma_ahp_depth_abs_slow	-62.2882	-151.52
step1_soma_ahp_slow_time	0.140599	-0.8473
step1_soma_ap_height	28.43591	-33.959
step1_soma_ap_width	0.67857	-2.132
step1_soma_isi_cv	0.03328	-1.202
step1_soma_adaptation_index2	-0.0039499	-0.3790
step1_soma_doublet_isi	67.00	-1.699
step1_soma_mean_frequency	7.106	-52.343
step1_soma_time_to_first_spike	33.3000	-719.1
step2_soma_ahp_depth_abs	-60.6933	-110.974
step2_soma_ahp_depth_abs_slow	-60.8186	-151.066
step2_soma_ahp_slow_time	0.1496	-0.8481
step2_soma_ap_height	26.5820	-33.958
step2_soma_ap_width	0.67058	-2.1747
step2_soma_isi_cv	0.03598	-1.0649
step2_soma_adaptation_index2	-0.001467	-0.47684
step2_soma_doublet_isi	44.600	-1.6999
step2_soma_mean_frequency	8.8444	-67.842
step2_soma_time_to_first_spike	23.000	-719.1
step3_soma_ahp_depth_abs	-56.759	-110.744
step3_soma_ahp_depth_abs_slow	-55.903	-149.126
step3_soma_ahp_slow_time	0.2168	-0.83586
step3_soma_ap_height	16.968	-34.01
step3_soma_ap_width	0.5968	-2.6932
step3_soma_isi_cv	0.09933	-1.1164
step3_soma_adaptation_index2	0.007206	-0.5326
step3_soma_doublet_isi	21.100	-1.699
step3_soma_mean_frequency	16.086	-125.30
step3_soma_time_to_first_spike	10.600	-719.1
bap_dend1_ap_amplitude_from_voltagebase	53.267	-60.701
bap_dend2_ap_amplitude_from_voltagebase	30.592	-31.779
bap_soma_ap_height	37.519	-33.95
bap_soma_ap_width	0.800	-1.583
bap_soma_spikecount	1.0	-0.8855

Index	Parameter	Ground truth
θ_1	gnats2_tbar_nats2_t_apical	0.026145
θ_2	gskv3_1bar_skv3_1_apical	0.004226
θ_3	gimbar_im_apical	0.000143
θ_4	gnata_tbar_nata_t_axonal	3.137968
θ_5	gk_tstbar_k_tst_axonal	0.089259
θ_6	gamma_cadynamics_e2_axonal	0.00291
θ_7	gnap_et2bar_nap_et2_axonal	0.006827
θ_8	gsk_e2bar_sk_e2_axonal	0.007104
θ_9	gca_hvabar_ca_hva_axonal	0.00099
θ_{10}	gk_pstbar_k_pst_axonal	0.973538
θ_{11}	gskv3_1bar_skv3_1_axonal	1.021945
θ_{12}	decay_cadynamics_e2_axonal	287.19873
θ_{13}	gca_lvastbar_ca_lvast_axonal	0.008752
θ_{14}	gamma_cadynamics_e2_somatic	0.000609
θ_{15}	gskv3_1bar_skv3_1_somatic	0.303472
θ_{16}	gsk_e2bar_sk_e2_somatic	0.008407
θ_{17}	gca_hvabar_ca_hva_somatic	0.000994
θ_{18}	gnats2_tbar_nats2_t_somatic	0.983955
θ_{19}	decay_cadynamics_e2_somatic	210.48529
θ_{20}	gca_lvastbar_ca_lvast_somatic	0.000333

6.10 Choices of hyperparameters

For the results on the benchmark tasks, we picked the same hyperparameters for TSNPE as those that were used in Lueckmann et al. [2021] for APT (called SNPE in Lueckmann et al. [2021]). For fitting the single-neuron model, we picked the same hyperparameters with the exception of the batchsize: We set it to 500 for this model, whereas the benchmarking tasks are performed with a batchsize of 10000.

For the threshold truncation threshold $\boldsymbol{\tau}$, we evaluated the $\epsilon = 10^{-3}$, 10^{-4}, and 10^{-5} quantiles of posterior samples on the benchmark tasks. When fitting the multicompartment model of single-neuron dynamics, we used the 10^{-3} quantile of posterior samples.

For both neuroscience tasks, we used a neural spline slow (NSF) as density estimator [Durkan et al., 2019]. The hyperparameters of the NSF are the defaults from the 'sbi' package [Tejero-Cantero et al., 2020]. On both of these tasks, we used $\epsilon = 10^{-3}$. On the pyloric network task, we replaced invalid summary statistics with a value that is 2 standard deviations (of prior predictives) below the observation.

For the pyloric network task, we ran APT with 2 atoms. We reduced the number of atoms from the default value in the 'sbi' package (10 atoms) because a larger number of atoms increased training time. When using 10 atoms, the training time exceeded the simulation time of the model on the pyloric network task. With 2 atoms, the training time of APT was comparable to the training time of TSNPE (albeit still a bit higher). We implemented the transformation of the parameter space with the pytorch method 'biject_to()' [Paszke et al., 2019]. We sampled from the truncated proposal with rejection sampling.

For the multicompartment model of single-neuron dynamics, we initially samples parameters from the truncated proposal with rejection sampling. After the third round of training, the rejection rate exceeded 99.9%, which renders rejection sampling more expensive than sampling importance resampling (SIR). We thus switched to SIR with $K = 1024$ for the final rounds (see Sec. 3.2).

6.11 Pooling data across several rounds

At every round, we use a truncated proposal which is $\tilde{p}(\boldsymbol{\theta}) = 0$ outside of the $\text{HPR}_\epsilon(q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x}_o))$. When running TSNPE across several rounds $R > 1$, we would like to be simulation-efficient by

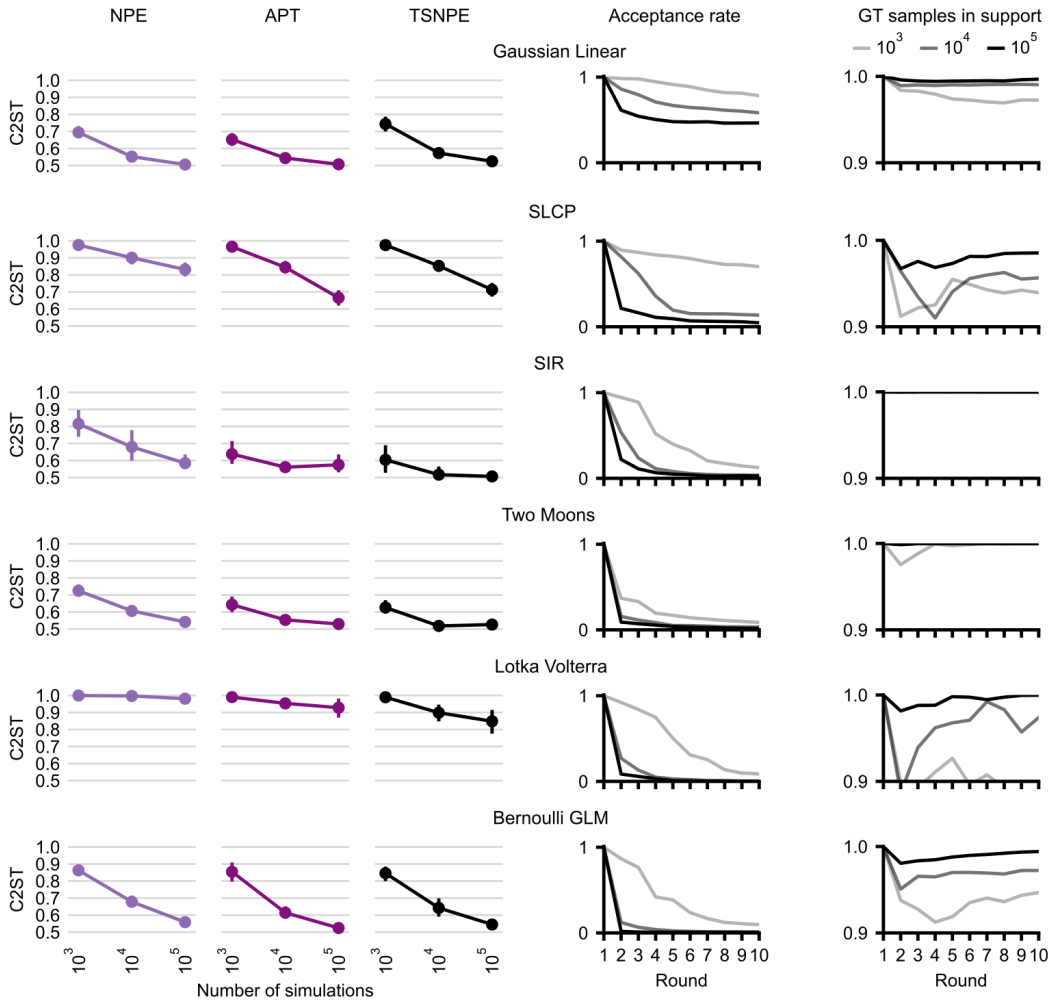

Figure 7: **Benchmark results for a less conservative threshold.** All hyperparameters are the same as in Fig. 4. The only difference is that that we used $\epsilon = 10^{-3}$ (compared to 10^{-4} in Fig. 4).

training on data collected across *all* rounds. After R rounds, the parameters will be sampled from:

$$\overline{p}(\boldsymbol{\theta}) = \frac{1}{N} \sum_{r=1}^{R} \tilde{p}_r(\boldsymbol{\theta}) \tag{2}$$

While this distribution is composed of several partially overlapping proposals, it remains proportional to the prior within $\text{HPR}_\epsilon(q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x}_o))$ (assuming that each individual $\tilde{p}_r(\boldsymbol{\theta})$ is proportional to the prior within $\text{HPR}_\epsilon(q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x}_o))$) (Appendix Fig. 18). This allows us to always train on data pooled across all rounds without requiring a change in the loss (see also proof of convergence in Appendix Sec. 6.2).

6.12 Diagnostic tool in a multi-round setting

We run our diagnostic tool after every round of training. If one trains only on simulations that were run in the most recent round, the diagnostic tool can be run as in the first round. However, if one wishes to train on simulations from all rounds, the deep neural density estimator is trained on simulations that follow the distribution:

$$\tilde{p} \propto p(\boldsymbol{\theta}) \cdot \sum_{r=1}^{R} \cdot \mathbb{1}_{\boldsymbol{\theta} \in \mathcal{M}_r}$$

In order to deal with this, when running the diagnostic tool in round r, we draw parameters from $\tilde{p}(\theta)$ (the truncated proposal), simulate them $\mathbf{x} \sim p(\mathbf{x}|\boldsymbol{\theta})$, sample from the posterior $q_{\boldsymbol{\phi}}(\boldsymbol{\theta}|\mathbf{x})$ and reject

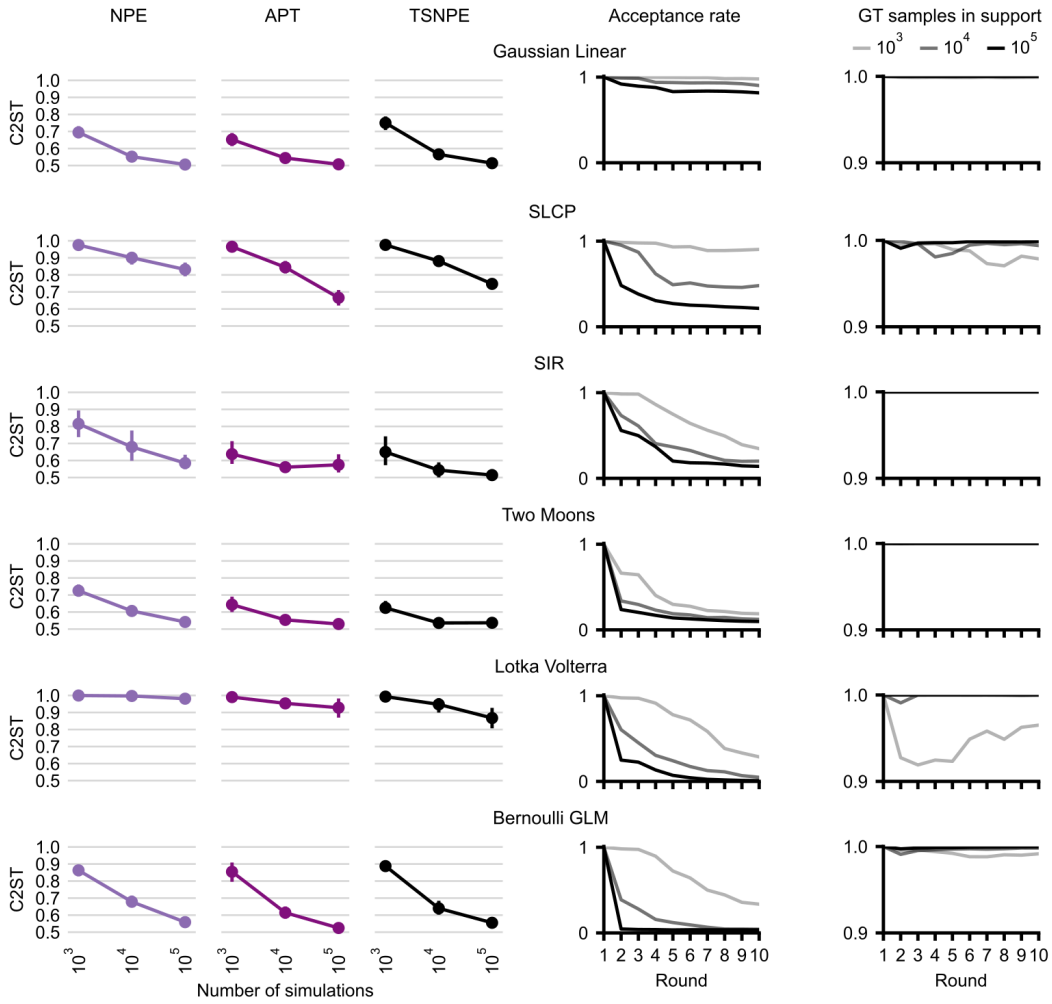

Figure 8: **Benchmark results for a more conservative threshold.** All hyperparameters are the same as in Fig. 4. The only difference is that that we used $\epsilon = 10^{-5}$ (compared to 10^{-4} in Fig. 4).

samples that lie outside of the support of $q_\phi(\boldsymbol{\theta}|\mathbf{x}_o)$. We then proceed as in the first round of running the diagnostic tool. This procedure ensures that the posterior regions which lie within the support of $q_\phi(\boldsymbol{\theta}|\mathbf{x}_o)$ are well-calibrated. It does not ensure that the full posterior ($q_\phi(\boldsymbol{\theta}|\mathbf{x})$ with $\mathbf{x} \sim p(\mathbf{x}|\boldsymbol{\theta})$ and $\boldsymbol{\theta} \sim \tilde{p}(\boldsymbol{\theta})$) is well-calibrated.

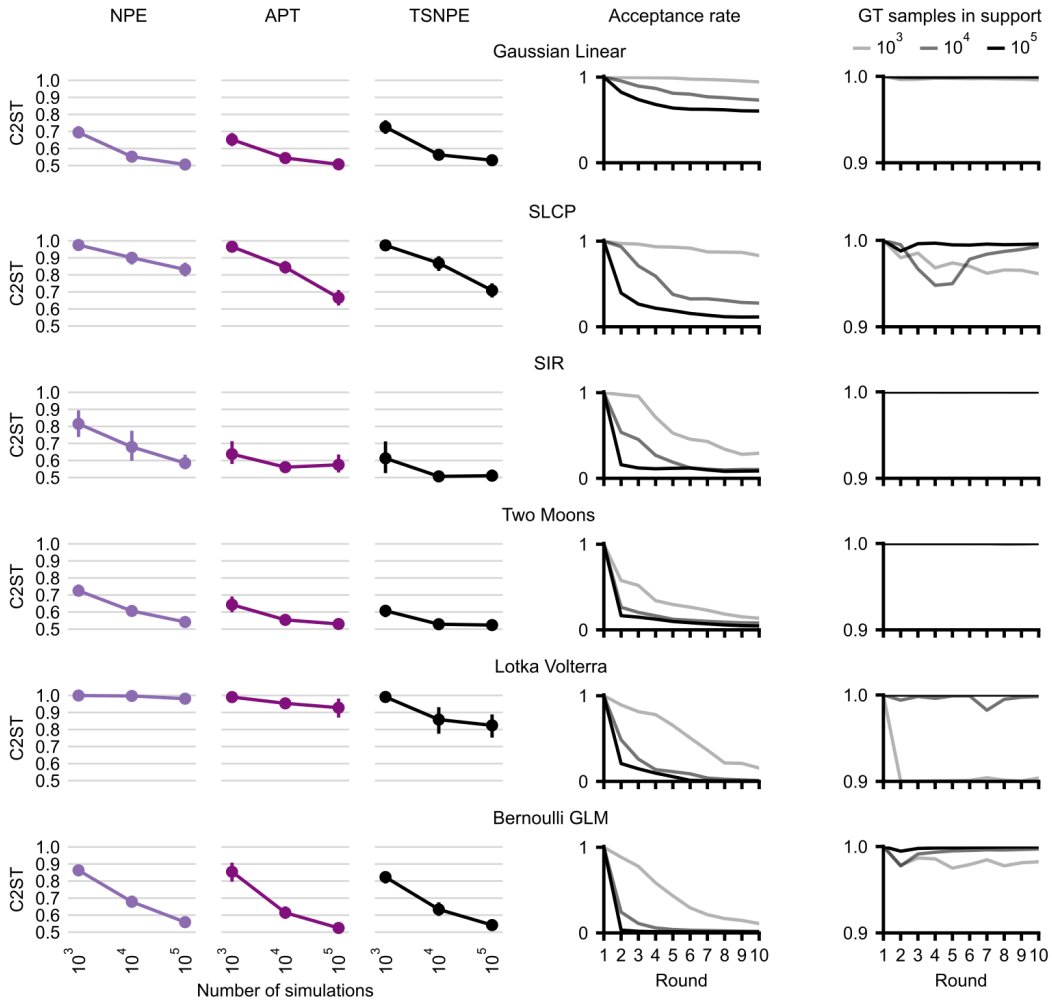

Figure 9: **Benchmark results for sampling importance resampling (SIR).** All hyperparameters are the same as in Fig. 4 ($\epsilon = 10^{-4}$) but we use SIR to sample from the truncated proposal (instead of rejection sampling in Fig. 4).

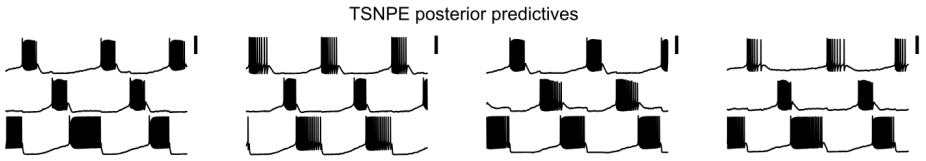

Figure 10: **Four posterior predictives of TSNPE applied to the pyloric network model.** The generated activity closely matches the summary statistics of the experimental data (Fig. 5a).

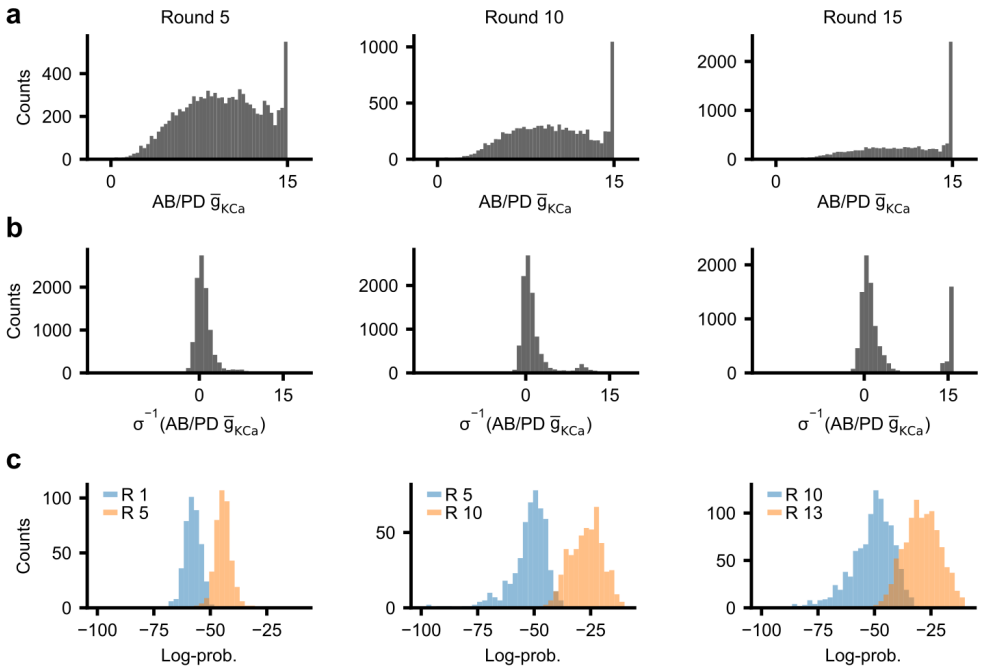

Figure 11: **Further explanation of issues when the parameter space is constrained.** (a) Marginal distribution of $\overline{g}$(KCa) of the AB/PD neuron when running APT for 5, 10, and 13 rounds. (b) Marginal distribution of this parameter, but when transformed into unconstrained space. This is the training data that the normalizing flow is 'effectively' seeing. (c) We evaluated those samples that were at the very right bounds of the marginal distribution under the current posterior (blue) as well as under the posterior from approximately 5 rounds before (orange). The samples have consistently lower log-probability under the previous posterior, which shows that the location of the leaking mass is moving.

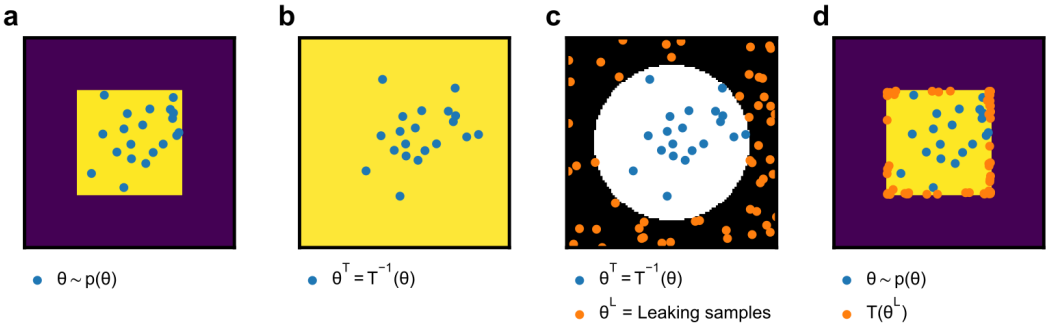

Figure 12: **Illustration of 'leakage' when prior is transformed to unconstrained space.** Note that these results are not based on an actual run of APT, but are purely for illustrative purposes. (a) The prior is a two-dimensional distribution on a constrained space (yellow region). Samples from the prior are in blue. (b) When the prior is transformed with an inverse sigmoid, its support becomes unconstrained. (c) APT will fit the posterior within the region of the training data (white region, blue samples). Outside of the training data, the trained density estimator can put arbitrarily much mass without affecting the loss (black region). Samples from this region are in orange. (d) When these 'leaking' samples (obtained with APT) are transformed back into constrained space, the 'leaking' mass ends up on the bounds of the prior.

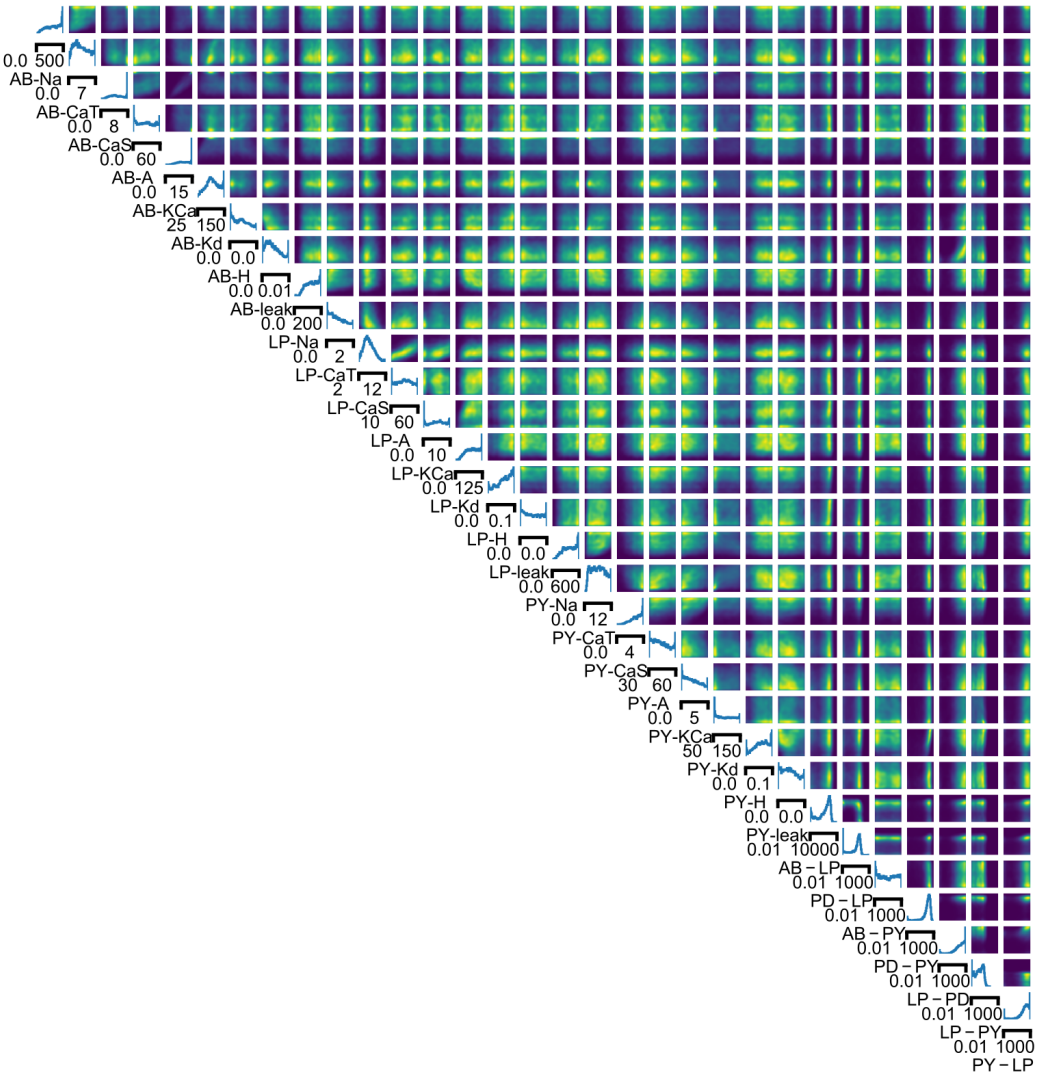

Figure 13: **Posterior distribution inferred by APT for the pyloric network model.**

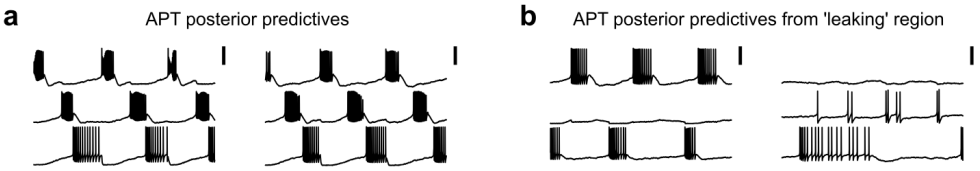

Figure 14: **Failure of APT when transformed to constrained space.** (a) Activity generated by two parameter sets sampled from the posterior distribution obtained with APT. We ensured that the shown parameter sets are not sampled from the very bounds of posterior distribution (i.e., that they were not sampled from the peak shown in Fig. 5c). (b) Activity generated by two parameter sets sampled from the posterior distribution obtained with APT. We specifically selected parameter sets whose value of $\overline{g}_{\mathrm{KCa}}$ in the AB-PD neuron was above 14.999 (i.e., those samples which are in the peak shown in Fig. 5c).

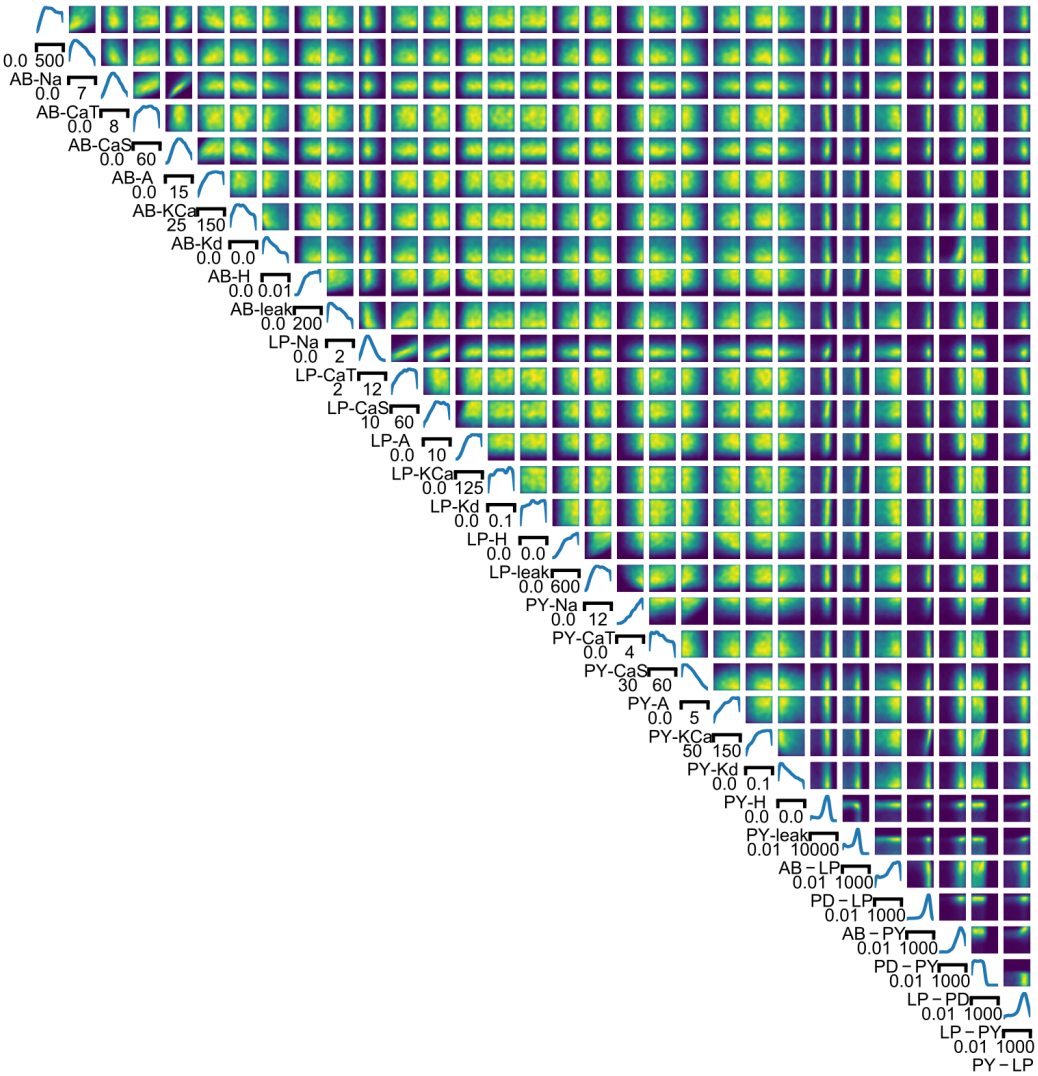

Figure 15: **Posterior distribution inferred by TSNPE for the pyloric network model.**

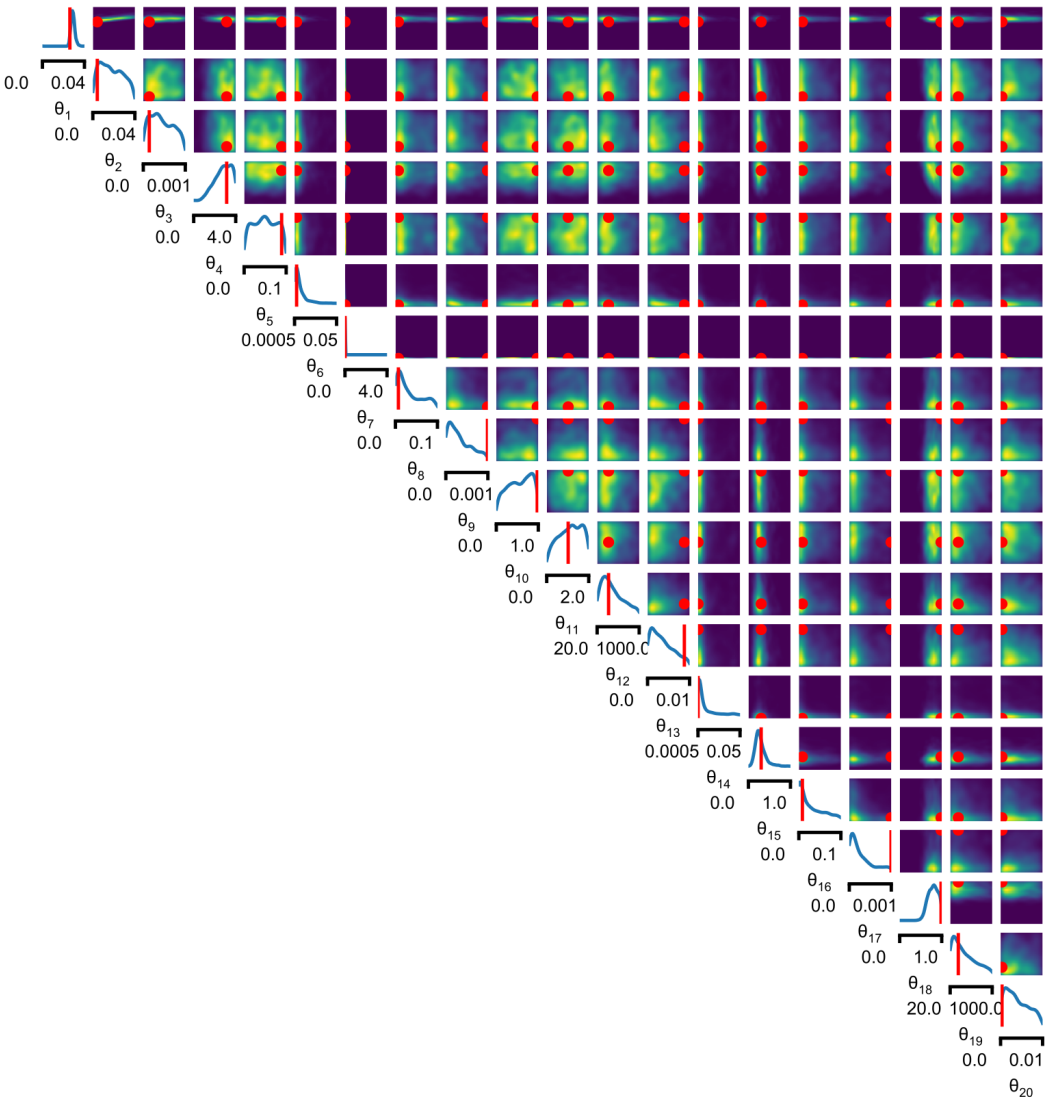

Figure 16: **Full posterior distribution over circuit parameters for the layer 5 pyramidal cell.** Parameter names can be found in Appendix Sec. 6.9. Red dot are the ground truth parameters

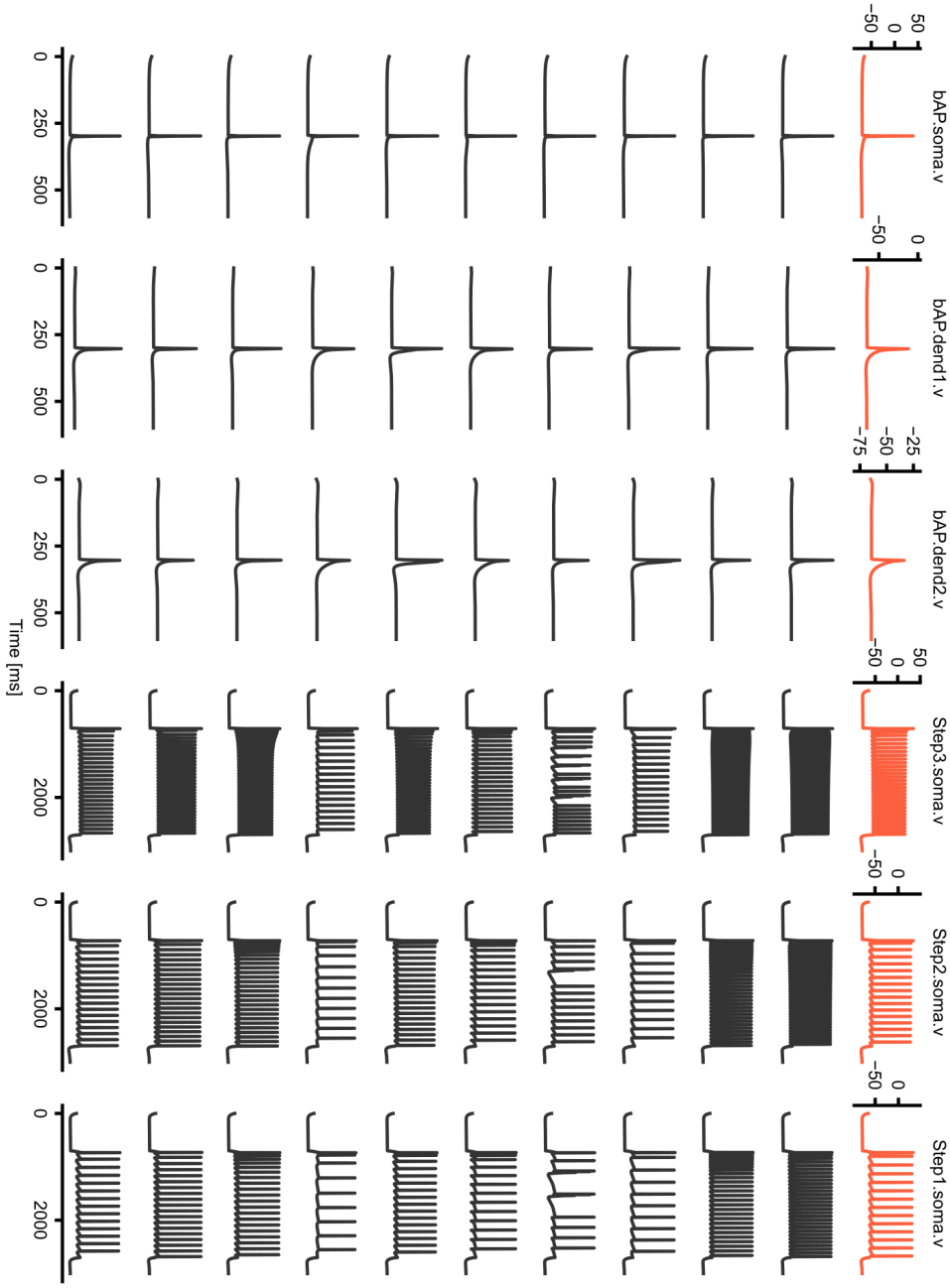

Figure 17: **Observed data (orange, top row) and ten posterior predictives obtained with TSNPE (black).**

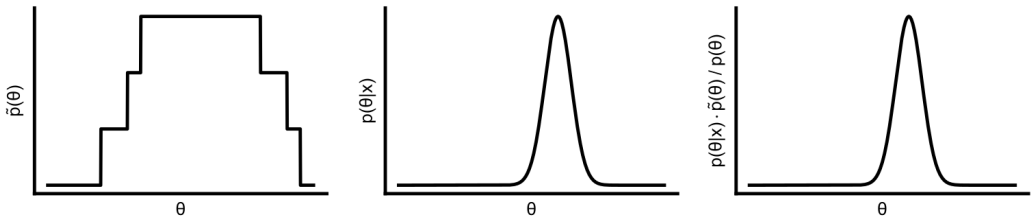

Figure 18: **Truncated proposals when pooling data from multiple rounds.** Assume a uniform prior. Left: Proposal that is the average of the proposal in three rounds. Middle: true posterior. Right: the approximate posterior converges to $p(\boldsymbol{\theta}|\mathbf{x}_o)\frac{\tilde{p}(\boldsymbol{\theta})}{p(\boldsymbol{\theta})}$ [Papamakarios and Murray, 2016, Lueckmann et al., 2017, Greenberg et al., 2019]. This matches the true posterior even though the proposal is not constant *outside* of the support of the true posterior.