Journal of Machine Learning Research 27 (2026) 1-48 Submitted 7/23; Revised 10/25; Published 2/26

Generative Bayesian Inference with GANs

Yuexi Wang YXWANG99@ILLINOIS.EDU
Department of Statistics
University of Illinois Urbana-Champaign
Champaign, IL 61820, USA

Veronika Ročková VERONIKA.ROCKOVA@CHICAGOBOOTH.EDU
Booth School of Business
University of Chicago
Chicago, IL 60637, USA

Editor: Daniel Roy

Abstract

In the absence of explicit or tractable likelihoods, Bayesians often resort to approximate Bayesian computation (ABC) for inference. Our work bridges ABC with deep neural implicit samplers based on generative adversarial networks (GANs) and adversarial variational Bayes. Both ABC and GANs compare aspects of observed and fake data to simulate from posteriors and likelihoods, respectively. We develop a Bayesian GAN (B-GAN) sampler that directly targets the posterior by solving an adversarial optimization problem. B-GAN is driven by a deterministic mapping learned on the ABC reference by *conditional* GANs. Once the mapping has been trained, iid posterior samples are obtained by filtering noise at a negligible additional cost. We propose two post-processing local refinements using (1) data-driven proposals with importance reweighting, and (2) variational Bayes. We support our findings with frequentist-Bayesian results, showing that the typical total variation distance between the true and approximate posteriors converges to zero for certain neural network generators and discriminators. Our findings on simulated data show highly competitive performance relative to some of the most recent likelihood-free posterior simulators.

Keywords: Approximate Bayesian Computation, Generative Adversarial Networks, Implicit Models, Likelihood-free Bayesian Inference, Variational Bayes

1. ABC and Beyond

For a practitioner, much of the value of the Bayesian inferential approach hinges on the ability to compute the entire posterior distribution. Very often, it is easier to infer data-generating probability distributions through simulator models rather than likelihood functions. However, Bayesian computation with simulator models can be particularly grueling.

We assume that $\theta \in \Theta \subset \mathbb{R}^d$ is a parameter controlling a simulator-based model that gives rise to a data vector $X_\theta^{(n)} = (X_1, \dots, X_n)' \sim P_\theta^{(n)}$ which is *not* necessarily iid. The model may be provided by a probabilistic program that can be easily simulated but its implicit likelihood $p_\theta^{(n)} = \pi(X^{(n)} \,|\, \theta)$ cannot be evaluated. For an unknown inferential target $\theta_0 \in \Theta$, our goal is to approximate the post-data inferential density (i.e. the posterior)

$$\pi(\theta \mid X_0^{(n)}) \propto p_\theta^{(n)}(X_0^{(n)})\pi(\theta), \tag{1}$$

where $X_0^{(n)} \sim P_{\theta_0}^{(n)}$ denotes the observed data. We allow for the possibility that *both* the likelihood $p_\theta^{(n)}(\cdot)$ and/or the prior $\pi(\theta)$ are analytically intractable but easy to draw from.

Without the obligation to build a model, Approximate Bayesian Computation (ABC) (Beaumont et al., 2002; Sisson et al., 2018) provides an approximation to the posterior (1) by matching aspects of observed and fake data. This is accomplished via forward simulation of the so-called ABC reference table $\{(\theta_j, X_j^{(n)})\}_{j=1}^T$ where θ_j's have been sampled from the prior $\pi(\theta)$ and fake data $X_j^{(n)}$'s have been sampled from the likelihood $p_{\theta_j}^{(n)}(\cdot)$. In order to keep only plausible parameter draws, this table is then filtered through an accept/reject mechanism to weed out parameter values θ_j for which the summary statistics of the fake and observed data were too far. Our work, albeit not being an ABC method per-se, builds off of recent ABC and simulation-based Bayesian inference innovations described below.

ABC Regression adjustment (Beaumont et al., 2002; Beaumont, 2003; Blum and François, 2010) is a post-processing step that re-weights and re-adjusts the location of θ_j's gathered by rejection ABC by fitting a (weighted) regression model of θ_j's onto summary statistics $s_j = s(X_j^{(n)})$. Such a model can be regarded as provisional density estimator of $\pi(\theta \mid X^{(n)})$ derived from $s(X^{(n)})$ under certain regression distributional assumptions. More flexible conditional density estimators, such as neural mixture density networks (Papamakarios and Murray, 2016; Lueckmann et al., 2017), have been successfully integrated into ABC without the burden of choosing summary statistics. Our approach is related to these developments. However, we do not attempt to learn a flexible parametric approximation to the posterior (or the likelihood (Lueckmann et al., 2019; Papamakarios et al., 2019)). Instead, we find an *implicit* neural sampler from an approximation to $\pi(\theta \mid X^{(n)})$ by training Generative Adversarial Networks (Goodfellow et al., 2016) on the ABC reference table. GANs have been originally conceived to simulate from complex likelihoods by contrasting observed and fake data. ABC, on the other hand, contrasts observed and fake data to simulate from complex posteriors. Bringing together these two approaches, we propose the B-GAN posterior sampler, an incarnation of conditional GANs (Gauthier, 2014; Mirza and Osindero, 2014; Athey et al., 2021; Zhou et al., 2022) for likelihood-free Bayesian simulation. By contrasting the ABC reference table with a fake dataset under the same marginal distribution $\pi(X^{(n)})$, B-GAN learns to simulate from an approximation to the conditional distribution $\pi(\theta \mid X^{(n)})$. Similarly as Papamakarios and Murray (2016) and Lueckmann et al. (2017), our method is also global in the sense that it learns $\pi(\theta \mid X^{(n)})$ for *any* $X^{(n)}$, not necessarily $X_0^{(n)}$. More perfected posterior reconstructions can be obtained with post-processing steps that zoom in onto the posterior distribution evaluated at $X_0^{(n)}$. We consider two such refinements based on: (1) reinforcement learning with importance sampling, and (2) adversarial variational Bayes. We describe each approach below.

Simple rejection ABC may require exceedingly many trials to obtain only a few accepted samples when the posterior $\pi(\theta \mid X_0^{(n)})$ is much narrower than the prior $\pi(\theta)$ (see e.g. Marjoram et al. (2003); Sisson et al. (2007); Beaumont et al. (2009)). This has motivated query-efficient ABC techniques which intelligently decide where to propose next (see Jarvenpaa et al. (2020); Hennig and Schuler (2012) for decision-theoretic reasoning or Järvenpää et al. (2019) and Gutmann and Corander (2016) for implementations based on Bayesian optimization and surrogate models). Alternatively, Lueckmann et al. (2017) learn a Bayesian mixture density network approximating the posterior over multiple rounds of adaptively chosen simulations and use more flexible proposal distributions (not necessarily the prior) with a built-in importance-reweighting scheme. A similar strategy was used in Papamakarios et al. (2019) who used a pilot run of mixture density networks to learn the proposal distribution for the next round. Although $X_0^{(n)}$ *is not* used in B-GAN training, it can be used in the proposal inside the ABC reference table. Similarly as in Papamakarios et al. (2019), we use $X_0^{(n)}$ to construct a flexible proposal (i.e. an empirical Bayes prior) and convert the draws to posterior samples under the original prior by importance reweighting. This 'reinforcement learning' refinement substantially improves the reconstruction accuracy and can be justified by theory.

Our vanilla B-GAN sampler uses contrastive learning (Gutmann et al., 2018; Durkan et al., 2020) to estimate the conditional distribution $\pi(\theta \mid X^{(n)})$ for *any* $X^{(n)}$. Since $X_0^{(n)}$ is used only at the evaluation stage (not the training stage), we can custom-make the sampler to $X_0^{(n)}$ by using the B-GAN output (or the output after reinforcement learning) as an initialization for implicit variational Bayes optimization (Tran et al., 2017; Huszár, 2017). Implicit variational Bayes attempts to approximate the posterior using densities which are defined implicitly by a push-forward mapping. B-GAN also trains such a mapping but the generator will have never seen observed data. At later stages of B-GAN training, we can thereby modify the objective function for the generator so that it minimizes a lower bound to the marginal likelihood. Since the likelihood cannot be evaluated, we use contrastive learning inside the variational objective to compute the lower bound (Huszár, 2017; Tran et al., 2017). We consider the joint-contrastive form (Huszár, 2017; Durkan et al., 2020), where the classifier is still trained to learn the joint likelihood ratio using the ABC reference table (similarly as in B-GAN). However, the generator is now trained on $X_0^{(n)}$ by maximizing the evidence lower bound. This algorithm is related to the B-GAN simulator, but uses $X_0^{(n)}$ during the training stage.

Contrastive learning has been used inside Bayesian likelihood-free sampling algorithms before (see e.g. Wang et al. (2022); Gutmann et al. (2018); Kaji and Ročková (2022)). Both Wang et al. (2022) and Kaji and Ročková (2022) assume iid data with a large enough sample size n to be able to apply classification algorithms for each iteration of Metropolis Hastings and ABC, respectively. Our approach does not require iid data and can flexiblely accommodate almost any shape of data. We also do not require to run classification at each posterior simulation step.

We show highly competitive performance of our methods (relative to state-of-the art likelihood-free Bayesian methods) on several simulated examples. While conceptually related methodology has occurred before (Papamakarios and Murray, 2016; Lueckmann et al., 2017; Ramesh et al., 2022), theory supporting these likelihood-free Bayesian approaches has been lacking. We provide new frequentist-Bayesian theoretical results for the typical variational distance between the true and approximated posteriors. We analyze Wasserstein versions of both the B-GAN algorithm as well as adversarial variational Bayes. With properly tuned neural networks, we show that this distance goes to zero as $n \to \infty$ with large enough ABC reference tables.

The outline of our paper is as follows. Section 2 reviews conditional GANs and introduces the Bayesian GAN sampler together with the reinforcement adjustments. In Section 3, we describe another local enhancement strategy inspired by implicit variational Bayes. In Section 4, we investigate the theoretical guarantees of the B-GAN posteriors. The performance of our methods is illustrated on simulated datasets in Section 5 and a real data application in Section 6. In Section 7, we conclude with a discussion.

2. Adversarial Bayes

Generative Adversarial Networks (GANs) (Goodfellow et al., 2016) are a game-theoretic construct in artificial intelligence designed to simulate from likelihoods over complex objects. GANs involve two machines playing a game against one another. A *Generator* aims to deceive a *Discriminator* by simulating fake samples that resemble observed data while, at the same time, the Discriminator learns to tell the fake and real data apart. This process iterates until the generated data are indistinguishable by the Discriminator and can be regarded as genuine likelihood samples. Below, we review several recent GAN innovations and propose an incarnation for simulation from a posterior as opposed to a likelihood.

2.1 Vanilla GANs

In its simplest form, GANs learn how to implicitly simulate from the likelihood $p_{\theta_0}^{(n)}(\cdot)$ using only its realizations $X_0^{(n)} \in \mathcal{X}$ where $X_0^{(n)} \sim p_{\theta_0}^{(n)}$ when θ_0 is unknown. Recall that draws from implicit distributions can be obtained by passing a random noise vector $Z^{(n)} \in \mathcal{Z}$ through a non-stochastic pushforward mapping $g_{\boldsymbol{\beta}}(\cdot) : \mathcal{Z} \to \mathcal{X}$. The original GANs formulation (Goodfellow et al., 2016) involves a Generator, specified by the mapping $g_{\boldsymbol{\beta}}(\cdot)$, that attempts to generate samples similar to $X_0^{(n)}$ by filtering $Z^{(n)}$, i.e. $Z^{(n)} \sim \pi_Z^{(n)}, X^{(n)} = g_\beta(Z^{(n)}) \sim p_\theta^{(n)}$.

The coefficients $\boldsymbol{\beta}$ in the Genrator are iteratively updated depending on the feedback received from the Discriminator. The Discriminator, specified by a mapping $d(\cdot) : \mathcal{X} \to (0,1)$, gauges similarity between $X^{(n)}$ and $X_0^{(n)}$ with a discrepancy between their (empirical) distributions. Hereafter we use X to denote a generic dataset as $X \in \mathcal{X}$ for simplicity of notation. At a population level, a standard way of comparing two distributions, say $P_{\theta_0}^{(n)}$ and $P_\theta^{(n)}$, is with the symmetrical Jensen-Shannon divergence which can be written as a solution to a particular optimization problem

$$\mathrm{JS}(P_\theta^{(n)}, P_{\theta_0}^{(n)}) = \ln 2 + 0.5 \times \sup_{d:\mathcal{X}\to(0,1)} \left\{ E_{X\sim P_\theta^{(n)}} \ln\left[d(X)\right] + E_{X\sim P_{\theta_0}^{(n)}} \ln\left[1 - d(X)\right] \right\}. \quad (2)$$

The optimal Discriminator $d^*(\cdot)$, solving the optimization (2), is $d^*(X) = p_\theta^{(n)}(X)/[p_\theta^{(n)}(X) + p_{\theta_0}^{(n)}(X)]$ (Goodfellow et al., 2014, Proposition 1). The optimal Generator is then defined through the optimal value $\boldsymbol{\beta}^*$ which leaves the Discriminator maximally confused, i.e. $d^*(X) = 1/2$ and therefore $p_{\theta_0}^{(n)}(X) = p_\theta^{(n)}(X)$ uniformly over $\mathcal{X}$.

Despite the nice connection to likelihood ratios, original GANs (Goodfellow et al., 2014) may suffer from training difficulties when the discriminator becomes too proficient early on (Gulrajani et al., 2017; Arjovsky and Bottou, 2017). Alternative divergences have been implemented inside GANs that are less prone to these issues. For example, the Wasserstein distance (Arjovsky and Bottou, 2017) also admits a dual representation

$$d_W\left(P_\theta^{(n)}, P_{\theta_0}^{(n)}\right) = \sup_{f \in \mathcal{F}_W} \left| E_{X \sim P_\theta^{(n)}} f(X) - E_{X \sim P_{\theta_0}^{(n)}} f(X) \right| \tag{3}$$

where $\mathcal{F}_W = \{f : \|f\|_L \leq 1\}$ are functions with a Lipschitz semi-norm $\|f\|_L$ at most one. The function $f(\cdot)$ is often referred to as the *Critic*. In our implementations, we will concentrate on the Wasserstein version of GANs (Arjovsky et al., 2017).

2.2 Conditional GANs for Bayes

While originally intended for simulating from likelihoods underlying observed data, GANs can be extended to simulating from distributions *conditional on* observed data. Certain aspects of conditional GANs (cGANs) have been investigated earlier (Gauthier, 2014; Mirza and Osindero, 2014) in various contexts including causal inference (Athey et al., 2021) or non-parametric regression (Zhou et al., 2022). Our work situates conditional GANs firmly within the context of ABC and likelihood-free posterior simulation. Before we describe our development in Section 2.3, we first introduce the terminology of cGANs within a Bayesian context. We will intentionally denote with X the conditioning variables and focus on the conditional distribution $\pi(\theta \mid X)$ for the inferential parameter $\theta \in \Theta$ with a prior $\pi(\theta)$. Similarly as with vanilla GANs in Section 2.1, cGANs again involve two adversaries represented by two mappings. We focus on the Wasserstein version here. For readers that are less familiar with GANs and cGANs, we include the JS-based cGAN in Appendix A, which is more intuitive but less stable in training.

Definition 1 *(Generator) We define a deterministic* generative model *as a mapping $g : (\mathcal{Z} \times \mathcal{X}) \to \Theta$ that filters noise random variables $Z \in \mathcal{Z}$ to obtain samples from an implicit conditional density $\pi_g(\theta \mid X)$. This conditional model then defines an implicit joint model $\pi_g(X, \theta) = \pi_g(\theta \mid X)\pi(X)$, where $\pi(X) = \int_\Theta p_\theta^{(n)}(X)\pi(\theta)\mathrm{d}\theta$ is the marginal likelihood.*

Definition 2 *(Critic) We define a deterministic* critic model *as a mapping $f : (\mathcal{X} \times \Theta) \to \mathbb{R}$, which estimates the Wasserstein distance between the data pairs (X, θ) from $\pi(X, \theta)$ and $\pi_g(X, \theta)$.*

The main distinguishing feature, compared to vanilla GANs, is that the conditioning random vector X enters *both* mappings. The task is to flexibly parametrize $g_{\boldsymbol{\beta}}(\cdot)$, e.g. using neural networks as will be seen later, in order to approximate the joint density model $\pi(X, \theta)$ as closely as possible. Ideally, we would like to recover an (oracle) function $g^* : \mathcal{Z} \times \mathcal{X} \to \Theta$ such that the conditional distribution of $g^*(Z, X)$ given X is the same as $\pi(\theta \mid X)$. The existence of such an oracle g^* is encouraged by the noise-outsourcing lemma from probability theory (Kallenberg, 2002; Zhou et al., 2022) which we reiterate as Lemma 9 in Section A for Gaussian Z. Although it is not necessary for Θ and Z to be of the same dimension d, we choose $\pi_Z = N(0, I_d)$ to balance the expressiveness of the generator and the discriminator.

The premise of conditional GANs rests in the fact that matching two joint distributions, while fixing a marginal distribution, is equivalent to matching conditional distributions. This implies that $g^*(Z, X)$, given X, is indeed distributed according to $\pi(\theta \mid X)$. The question remains how to find the oracle mapping g^* in practice. In the Wasserstein cGAN, the minimax game is characterized as

$$(g^*, f^*) = \arg\min_{g\in\mathcal{G}} \max_{f\in\mathcal{F}} \left| E_{X\sim\pi(X), Z\sim\pi_Z} f(X, g(Z, X)) - E_{(\theta,X)\sim\pi(X,\theta)} f(X, \theta) \right|, \qquad (4)$$

where, using $\mathcal{F} = \mathcal{F}_W$ (1-Lipschitz in θ), g^* minimizes the Wasserstein distance between $\pi_g(X,\theta)$ and $\pi(X,\theta)$. When $\mathcal{G}$ and $\mathcal{F}$ are expressive enough, the minimum is achieved if and only if the Wasserstein distance between $\pi_{g^*}(X,\theta)$ and $\pi(X,\theta)$ is zero, which equivalently means $\pi_{g^*}(X,\theta) = \pi(X,\theta)$. With the same marginal $\pi(X)$, the solution g^* essentially satisfies $\pi_{g^*}(\theta \mid X) = \pi(\theta \mid X)$.

2.3 Generative Bayesian Inference with GANs

To implement the adversarial game (4) in practice, one needs to (a) parametrize $\mathcal{F}$ and $\mathcal{G}$ (for instance using neural networks) and (b) to replace the expectations in (4) with empirical counterparts. Both of these steps will introduce approximation error. We provide theoretical insights later in Section 4. We assume that the generator class $\mathcal{G} = \{g_{\boldsymbol\beta} : (\mathcal{Z}\times\mathcal{X}) \to \Theta \text{ where } \boldsymbol\beta \in \mathbb{R}^G\}$ is parametrized with $\boldsymbol\beta$ and the critic class $\mathcal{F} = \{f_{\boldsymbol\omega} : (\mathcal{X}\times\Theta) \to \mathbb{R} \text{ where } \boldsymbol\omega \in \mathbb{R}^C\}$ is parametrized with $\boldsymbol\omega$. We use neural networks (with ReLU activations) and support this choice with theory (Corollary 6).

For the empirical version, one can use the ABC reference table which consists of simulated data pairs $\{(\theta_j, X_j^{(n)})\}_{j=1}^T$ generated from the joint model $\pi(X^{(n)},\theta) = p_\theta^{(n)}(X^{(n)})\pi(\theta)$ under the prior $\pi(\theta)$. Specifically, we use n to refer the total data dimensionality. Each draw $X_j^{(n)} \sim P_\theta^{(n)}$ can be either n^* iid q-dimensional observations, or q-dimensional times-series of length n^*, such that $n = n^* * q$. From now on, we will simply denote $X_j^{(n)}$ with X_j, similarly for $X_0^{(n)}$ and $X^{(n)}$.

We can break the relationship between θ and X in the ABC reference table by contrasting these data pairs with another dataset consisting of $\{(g(Z_j, X_j), X_j)\}_{j=1}^T$ where Z_j's have been sampled from $\pi_Z(\cdot)$. Keeping the same X_j's essentially means that we are keeping the same marginal. The dataset $\{(g(Z_j, X_j), X_j)\}_{j=1}^T$ encapsulates iid draws from $\pi_g(X,\theta)$. These two datasets can be then used to approximate the expectations in (22) and (4). A high-level description of an algorithm for solving the Jensen-Shannon (JS) version of the game from Lemma 10 is outlined in Algorithm 4 (Appendix Section D). As mentioned earlier, the JS version may suffer from training issues (Arjovsky and Bottou, 2017). We provide an illustration of such issues using convergence diagnostics on a toy example in Section D.

In our implementations, we thereby consider the empirical version of (4) which again involves simulated datasets $\{(\theta_j, X_j)\}_{j=1}^T$ and $\{Z_j\}_{j=1}^T \overset{\text{iid}}{\sim} \pi_Z(\cdot)$ to obtain

$$\widehat{\boldsymbol\beta}_T = \arg\min_{\boldsymbol\beta: g_{\boldsymbol\beta}\in\mathcal{G}} \left[\max_{\boldsymbol\omega: f_{\boldsymbol\omega}\in\mathcal{F}_W} \left| \sum_{j=1}^T f_{\boldsymbol\omega}(X_j, g_{\boldsymbol\beta}(Z_j, X_j)) - \sum_{j=1}^T f_{\boldsymbol\omega}(X_j, \theta_j) \right| \right]. \qquad (5)$$

One particular way of solving this problem is summarized in Algorithm 1. In terms of the constraint on $\boldsymbol\omega$ to ensure the Lipschitz condition $\|f_{\boldsymbol\omega}\|_L \le 1$, the original Wasserstein GANs implementation (Arjovsky and Bottou, 2017) used gradient clipping, which may lead to computational issues. Alternatively, Gulrajani et al. (2017) imposed a soft version of the

Algorithm 1 *Vanilla B-GAN*

```
Input
Prior $\pi(\theta)$, observed data $X_0$ and noise distribution $\pi_Z(\cdot)$
Training
Initialize network parameters $\boldsymbol{\omega}^{(0)} = 0$ and $\boldsymbol{\beta}^{(0)} = 0$
Reference Table
For $j = 1, \dots, T$:    Generate $(X_j, \theta_j)$ where $\theta_j \sim \pi(\theta)$ and $X_j \sim P_{\theta_j}^{(n)}$.
Wasserstein GAN
For $t = 1, \dots, N$:
  Critic Update ($N_{\text{critic}}$ steps): For $k = 1, \dots, N_{\text{critic}}$
      Generate $Z_j \sim \pi_Z(z)$ for $j = 1, \dots, T$.
      Update $\boldsymbol{\omega}^{(t)}$ by applying stochastic gradient descent on (5).
  Generator Update (single step)
      Generate noise $Z_j \sim \pi_Z(z)$ for $j = 1, \dots, N$.
      Update $\boldsymbol{\beta}^{(t)}$ by applying stochastic gradient descent on (5).
Posterior Simulation:
For $i = 1, \dots, M$:    Simulate $Z_i \sim \pi_Z(z)$ and set $\widetilde{\theta}_i = g_{\boldsymbol{\beta}^{(N)}}(Z_i, X_0)$.
```

constraint with a penalty on the gradient of $f_{\boldsymbol{\omega}}$ with respect to a convex combination of the two contrasting datasets. We adopt the one-sided penalty same as Athey et al. (2021), with details in Appendix E. To stabilize gradients, the critic is updated multiple times (ideally until convergence) before each update of the generator, which is different from Algorithm 4 where such a stabilization may not be feasible (Arjovsky et al., 2017).

Our GAN framework for Bayesian posterior simulation (i.e. Algorithm 1 further referred to as B-GAN) consists of a neural sampler $g_{\widehat{\boldsymbol{\beta}}_T}(Z, X)$ which generates samples from an approximate posterior, i.e. conditioning on the observed data X_0, by filtering iid noise as follows

$$\widetilde{\theta}_j = g_{\widehat{\boldsymbol{\beta}}_T}(Z_j, X_0) \quad \text{where} \quad Z_j \overset{\text{iid}}{\sim} \pi_Z(\cdot) \quad \text{for} \quad j = 1, \dots, M. \tag{6}$$

When $g_{\widehat{\boldsymbol{\beta}}_T}$ is close to g^*, the samples $\widetilde{\theta}_j$ will arrive approximately from $\pi(\theta \,|\, X_0)$. One of the practical appeals of this sampling procedure is that, once the generator has been trained, the simulation cost is negligible. Note that our observed data $X_0 \sim p_{\theta_0}^{(n)}(X)$ are *not involved* in the training stage, *only* in the simulation stage (6). We illustrate Algorithm 1 on a toy example. The configurations of our B-GAN networks and optimization hyperparameters are described in Appendix E.2. Note that our B-GAN approach is different from the Bayesian GAN of Saatci and Wilson (2017), which places a probabilistic structure over the GAN parameters and update the parameteters via sampling instead of pure optimization. Our work utilizes the GAN framework for likelihood-free Bayesian inference and the training scheme is optimization-based.

Example 1 (Toy Example) *This toy example (analyzed earlier in (Papamakarios et al., 2019)) exposes the fragility of ABC methods in a relatively simple setting. The experiment entails $n = 4$ two-dimensional Gaussian observations $X = (x_1, x_2, x_3, x_4)'$ with $x_j \sim \mathcal{N}(\mu_{\boldsymbol{\theta}}, \Sigma_{\boldsymbol{\theta}})$ parametrized by $\boldsymbol{\theta} = (\theta_1, \theta_2, \theta_3, \theta_4, \theta_5)'$, where*

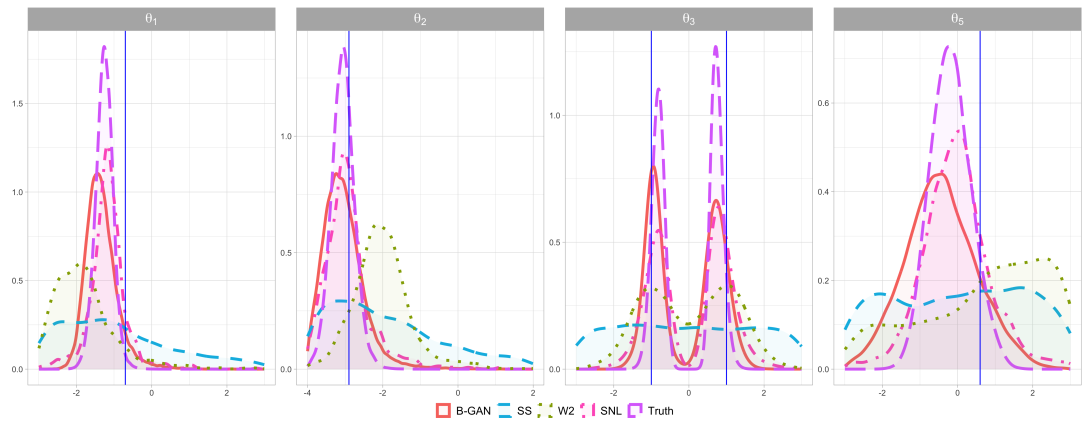


Figure 1: The approximate d posteriors given by B-GAN, SNL, SS, and W2 for the toy example. The results for θ_4 are similar to θ_3 and thus not shown here.

$$\mu_{\boldsymbol{\theta}} = (\theta_1, \theta_2)' \quad \textit{and} \quad \Sigma_{\boldsymbol{\theta}} = \begin{pmatrix} s_1^2 & \rho s_1 s_2 \\ \rho s_1 s_2 & s_2^2 \end{pmatrix}$$

with $s_1 = \theta_3^2, s_2 = \theta_4^2$ *and* $\rho = tanh(\theta_5)$. *The parameters* $\boldsymbol{\theta}$ *are endowed with a uniform prior on* $[-3,3] \times [-4,4] \times [-3,3] \times [-3,3] \times [-3,3]$. *Approximating the posterior can be tricky because the signs of parameters* θ_3 *and* θ_4 *are not identifiable, yielding multimodality. We generate* X_0 *with parameters* $\boldsymbol{\theta}_0 = (-0.7, -2.9, -1.0, -0.9, 0.6)'$. *Since we have access to the true posterior, we can directly compare our posterior reconstructions with the truth. We compare B-GAN with (1) ABC using naive summary statistics (SS) (mean and variance), (2) 2-Wasserstein distance ABC (Bernton et al., 2019), and (3) Sequential Neural Likelihood (SNL) (Papamakarios et al., 2019) with the default setting suggested by the authors. We provide all implementation details in Appendix E.2. For each method, we obtained* $M = 1\,000$ *samples and plotted them in Figure 1. Our B-GAN approach as well as SNL nicely capture the multimodality. Although the B-GAN posterior quickly locates a neighborhood around true values* θ_0, *its variance appears to be bigger than the true posterior, suggesting an approximation gap. This is expected because B-GAN is trained to perform well on average for any* X, not *necessarily for* X_0. *This motivates our two refinement strategies: one based on active learning (Section 2.4) and one based on variational Bayes (Section 3).*

Similarly as with default ABC techniques, our B-GAN approach is not query-efficient, i.e. many prior guesses θ_j in the training dataset may be too far from the interesting areas with a likelihood support leaving only a few observations to learn about the conditional $\pi(\theta \,|\, X_0)$. The next section presents a two-step approach which uses X_0 for proposal construction in the ABC reference table to obtain more valuable data-points in the reference table.

2.4 Two-step Refinement

Our chief goal is to find a high-quality approximation to the conditional distribution $\pi(\theta \,|\, X)$ evaluated at the observed data $X = X_0$, not necessarily uniformly over the entire domain $\mathcal{X}$. However, the ABC reference table $\{(\theta_j, X_j)\}_{j=1}^T$ may not contain enough data points X_j

Algorithm 2 *2-Step Refinement (B-GAN 2step)*

```
INPUT
    Prior $\pi(\theta)$, observed data $X_0$ and noise distribution $\pi_Z(z)$
Training
    Initialize network parameters $\boldsymbol{\omega}^{(0)} = 0$ and $\boldsymbol{\beta}^{(0)} = 0$
    Pilot Run
        Apply Algorithm 1 with $\pi(\theta)$ to learn $\widehat{g}_{pilot}(\cdot)$
    Reference Table
        Generate pairs $\{(X_j, \theta_j)\}_{j=1}^T$ where $\theta_j = \widehat{g}_{pilot}(Z_j, X_0)$ for $Z_j \sim \pi_Z$ and $X_j \sim P_{\theta_j}^{(n)}$.
    Refinement
        Apply Wasserstein GAN step in Algorithm 1 on $\{(X_j, \theta_j)\}_{j=1}^T$ and return $g_{\widehat{\boldsymbol{\beta}}_T}(\cdot)$
    Posterior Simulation
        Simulate $\{Z_i\}_{i=1}^M \overset{\text{iid}}{\sim} \pi_Z(z)$ and set $\widetilde{\theta}_i = g_{\widehat{\boldsymbol{\beta}}_T}(Z_i, X_0)$.
        Estimate $\hat{w}_i$ using either (7) or (8).
OUTPUT
    Pairs of posterior samples and weights $(\widetilde{\theta}_1, \hat{w}_1), \dots, (\widetilde{\theta}_M, \hat{w}_M)$
```

in the vicinity of X_0 to train the simulator when the prior $\pi(\theta)$ is too vague. This can be remedied by generating a reference table using an auxiliary proposal distribution $\widetilde{\pi}(\theta)$ which is more likely to produce pseudo-observations X_j that are closer to X_0. For example, a pilot simulator $g_{\widehat{\boldsymbol{\beta}}_T}(Z, X_0)$ in (6) obtained from Algorithm 1 under the original prior $\pi(\theta)$ can be used to guide simulations in the next round to sharpen the reconstruction accuracy around X_0 (Papamakarios and Murray, 2016). Training the generator $\widetilde{g}_{\widehat{\boldsymbol{\beta}}_T}$, simulating from an approximation to $\widetilde{\pi}(\theta \mid X_0) \propto p_{\theta_0}^{(n)}(X_0)\widetilde{\pi}(\theta)$, under the 'wrong' prior can be corrected for by importance re-weighting with weights $r(\theta) = \pi(\theta)/\widetilde{\pi}(\theta)$. Since the posterior $\widetilde{\pi}(\theta \mid X_0)r(\theta)$ is proportional to $\pi(\theta \mid X_0)$, reweighting the resulting samples $\widetilde{\theta}_j = \widetilde{g}_{\widehat{\boldsymbol{\beta}}_T}(Z, X_0^{(n)})$ with weights $w_j = r(\theta_j)$ will produce samples from an approximation to the original posterior (after normalization). Algorithm 2 summarizes this two-step strategy, referred to as B-GAN-2S.

Since the proposal density $\pi_{g_{\widehat{\boldsymbol{\beta}}_T}}(\theta \mid X_0)$ obtained in the pilot run *may not* have an analytical form, computing the importance weights $w_j = \pi(\theta_j)/\pi_{g_{\widehat{\boldsymbol{\beta}}_T}}(\theta_j \mid X_0)$ directly may not feasible. The density ratio $r(\theta)$ can be approximated, however. For example, with a tractable prior $\pi(\theta)$ the importance weights w_j can be estimated by

$$\hat{w}_j = \frac{\pi(\theta_j)}{\hat{\pi}_{g_{\widehat{\boldsymbol{\beta}}_T}}(\theta_j \mid X_0)} \tag{7}$$

where $\hat{\pi}_{g_{\widehat{\boldsymbol{\beta}T}}}(\theta_j \mid X_0)$ is a plugged-in kernel density estimator (KDE) estimator (Terrell and Scott, 1992). This is particularly useful and efficient when the parameter dimension is low. When the prior is also not tractable but simulatable, the weights w_j can be directly estimated from classification by contrasting datasets $\widetilde{\theta}_j \sim \widetilde{\pi}(\theta)$ (label '0') and $\theta_j \sim \pi(\theta)$ (label '1'). In particular, training a classifier $\widetilde{D}$ (see e.g. Cranmer et al. (2015); Durkan et al. (2020); Gutmann and Hyvärinen (2012) for the explanation of the 'likelihood-ratio-trick'

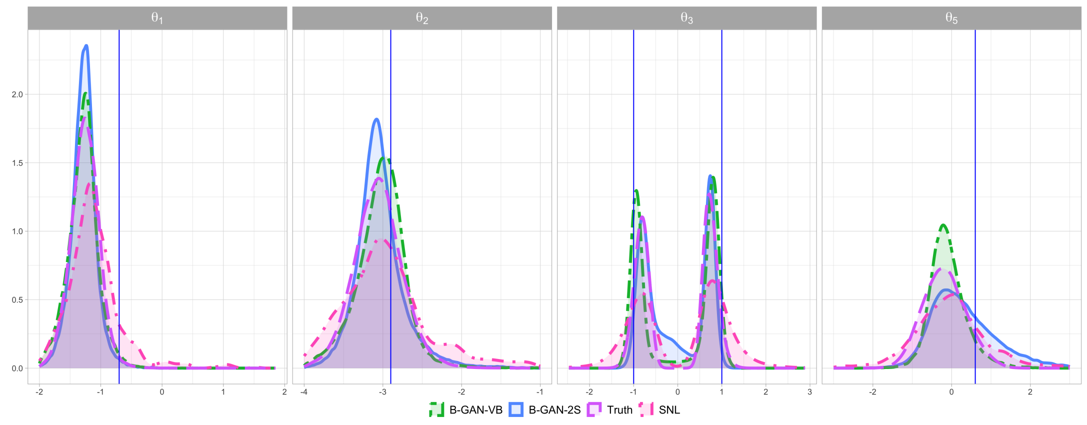


Figure 2: Posterior densities under the Gaussian model. The true parameter is $\boldsymbol{\theta}_0 = (-0.7, -2.9, -1.0, -0.9, 0.6)'$, while the signs of θ_3 and θ_4 are not identifiable.

for classification based estimators), we can obtain

$$\hat{w}_j = \frac{\tilde{D}(\widetilde{\theta}_j)}{1 - \tilde{D}(\widetilde{\theta}_j)}. \tag{8}$$

Papamakarios and Murray (2016) used mixture density networks estimators of the conditional distribution $\pi(\theta \mid x)$ after a pilot run to learn the proposal distribution $\widetilde{\pi}(\theta)$. In order to obtain an analytically tractable Gaussian mixture representation, their proposal $\widetilde{\pi}(\theta)$ has to be Gaussian and it cannot be narrower than any of the mixture components. We do not require such assumptions. Lueckmann et al. (2019) instead propose to directly incorporate the weights $r(\theta)$ inside training and relax the Gaussianity assumption to avoid the variance instability. Similarly as in Lueckmann et al. (2019), we could also incorporate weights $\hat{w}_j$ inside the objective function, e.g. multiplying each summand in (5) by $\hat{w}_j$.

In the two-step refinement, the observed data X_0 only contribute to the proposal distribution $\widetilde{\pi}(\theta)$, not the training of the simulator $g_{\widehat{\boldsymbol{\beta}}}(\cdot)$. In Section 3, we consider a variational Bayes variant which does involve X_0 in training.

Example 2 (Toy Example Continued) *We continue our exploration of the Toy Example 1. We now use the output from B-GAN (Algorithm 1) under the original uniform prior as a proposal distribution $\widetilde{\pi}(\theta)$ and generate training data $\{(X_j, \theta_j)\}_{j=1}^T$ with a marginal $\widetilde{\pi}(X) = \int_{\mathcal{X}} p_\theta^{(n)}(X)\widetilde{\pi}(\theta)d\theta$. To revert the generated posterior samples to the original uniform prior, we perform reweighting by $r(\theta) = \pi(\theta)/\widetilde{\pi}(\theta)$ using the kernel density estimator of $\widetilde{\pi}(\theta)$ obtained from the pilot B-GAN run in Algorithm 1. The number of training points used in the second step is $T = 50\,000$. We note that much smaller T could be used if one were to perform more sequential refinements, not just one. The re-weighted (and normalized) posterior is plotted against the truth, SNL and a variational Bayesian variant (introduced later in Section 3) in Figure 2. Both B-GAN and B-GAN-2S provide tighter approximations to the true posterior We repeated the experiment 10 times and report the Maximum Mean Discrepancies (MMD) (Gretton et al., 2012) between the true posterior and*

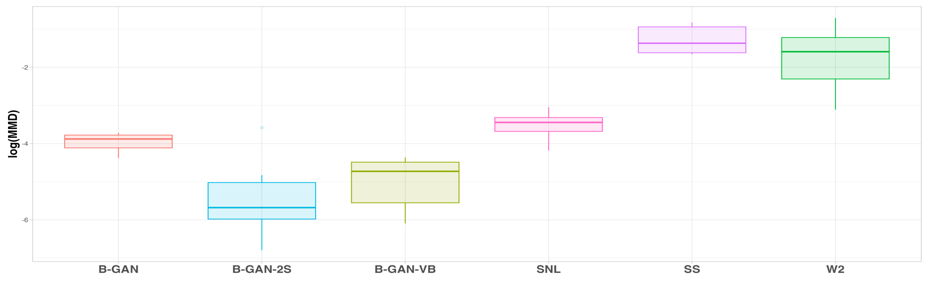


Figure 3: Maximum Mean Discrepancies (MMD, log scale) between the true posteriors and the approximated posteriors. The box-plots are computed from 10 repetitions.

its approximations obtained from $M = 1\,000$ posterior draws for each method in Figure 3. Satisfyingly, B-GAN-2S yields the smallest MMD. We support this encouraging finding with our theoretical results in Section 4.

2.5 The Case of i.i.d. Observations

In this subsection, we begin by reviewing three key challenges that arise when the total data dimensionality n is large. We then demonstrate how these challenges can be mitigated more effectively when the data consists of i.i.d. observations with large n^* but moderate q.

The first challenge stems from the fact that a large n demands more complex and expressive neural networks, which are often harder to train and require generating even bigger ABC reference tables. Second, as n^* increases, the posterior becomes more concentrated, making it difficult to efficiently identify high posterior density regions starting from a non-informative prior. Lastly, the storage and memory requirements grow substantially when simulating or processing the same number of ABC draws with larger n^*.

The most straightforward and naive implementation is to flatten the $n^* \times q$ data matrix into a vector and condition on the entire vector in our GAN sampler. However, this approach inherits all three challenges and should be avoided when $n = n^* * q$ is large. Instead, we propose two alternative implementations tailored to i.i.d. datasets with large n^*, which help alleviate some of these computational burdens.

The first solution is to split the dataset into multiple batches, and use sequential update to parse only one batch at a time. For example, for i.i.d. dataset with very large n^*, we partition the data into batches of size B, parse the data into vectors of size $B \times q$, then use the posterior from the previous batch as a prior for the next. This is coherent with the Bayesian framework. The neural network can also be recycled from previous batch, which should further relieve the computational burden. This can also be applied for some dependent datasets, such as the Lotka-Volterra model, whose dependency is solely determined by the observation from last timestamp. For this type of model, sequential parsing is still valid with partition along the time horizon. For this solution, it helps with second challenge and the third challenge, and can mildly relieve the first challenge. The sequential update allows us to gradually approach the concentrated high posterior density region and reduces the size of datasets that needs to be generated each time to be $T * B * q$

instead of $T * n^* * q$. Since the length of the flatten data vector is lower than $n^* * q$, it requires less complex network structrues compared to the naive implementation, but this can still quickly grow overwhelming when B also becomes really large.

The second solution leverages the exchangeable nature of i.i.d. observations. For an ideal parameter-efficient generator network, it should output the same conditional distribution regardless of the order of exchangeable inputs. In another words, it should satisfy

$$g_\beta(Z, [X_1, X_2, \ldots, X_n]) = g_\beta(Z, [X_{\pi(1)}, X_{\pi(2)}, \ldots, X_{\pi(n)}])$$

where $\pi(\cdot)$ can be any permutation. Similarly for the critic function, it should be able to provide the same Wasserstein distance estimate under any permutation of the conditional inputs.

For this parameter-efficient network, we adopt the deep set architecture from Zaheer et al. (2017), whose output is invariant to the order of inputs. To accommodate for the dependency between the parameter and each individual observation, we parametrize the generator and critic networks as

$$g_\beta(Z, [X_1, X_2, \ldots, X_n]) = g^{(1)}_{\beta_1}(Z, \sum_{i=1}^{n} g^{(2)}_{\beta_2}(Z, X_i))$$
$$f_\omega(\theta, [X_1, X_2, \ldots, X_n]) = f^{(1)}_{\omega_1}(\theta, \sum_{i=1}^{n} f^{(2)}_{\omega_2}(\theta, X_i))$$

where $g^{(1)}_{\beta_1}, g^{(2)}_{\beta_2}, f^{(1)}_{\omega_1}, f^{(2)}_{\omega_2}$ are sub-network structures inside the generator and critic functions. These structures allow us to efficiently learn conditional distribution on an exchangeable dataset without exploding network complexity.

We illustrate the efficiency of the three different implementations on the benchmark SBI example, M/G/1-queuing model. We follow the setting in Fearnhead and Prangle (2011). Each observation is a 5-dimensional vector consisting of the first five inter-departure times $x_i = (x_{i1}, x_{i2}, x_{i3}, x_{i4}, x_{i5})'$. In the model, the service times u_{ik} follow a uniform distribution $U[\theta_1, \theta_2]$, and the arrival times w_{ik} are exponentially distributed with the rate θ_3. We only observe the interdeparture times x_i, given by the process $x_{ik} = u_{ik} + \max(0, \sum_{j=1}^{k} w_{ij} - \sum_{j=1}^{k-1} x_{ij})$. We set true parameter to be $\theta_0 = (1, 5, 0.2)$ and the prior on $(\theta_1, \theta_2 - \theta_1, \theta_3)$ to be uniform on $[0, 10]^2 \times [0, 0.5]$.

To show how different implementations scale with n, we show three examples with $n = 50, 100, 200$ in Figure 4. Here we consider three treatments: (1) deep set structure (deep set), (2) sequential update by splitting n^* observations into 5 batches (sequential), and (3) stack all n observations into one long vector (stacked). To show how the performance of these implementations vary with sample size n^*, each implementation uses the same network structure, i.e., same network complexity across different n^*. Details of the implementations are provided in Appendix E.3.

From Figure 4, we observe that for the deep set structure, the approximated posterior for θ_2 and θ_3 gets more and more concentrated when n^* increases, and the approximated posterior covers the true values in all three n^*, showing that using the parameter-efficient network structure can scale better with n^* for i.i.d. datasets. For the simple stacking implementation, since the input dimension becomes $5n^*$, it suffers from all three challenges

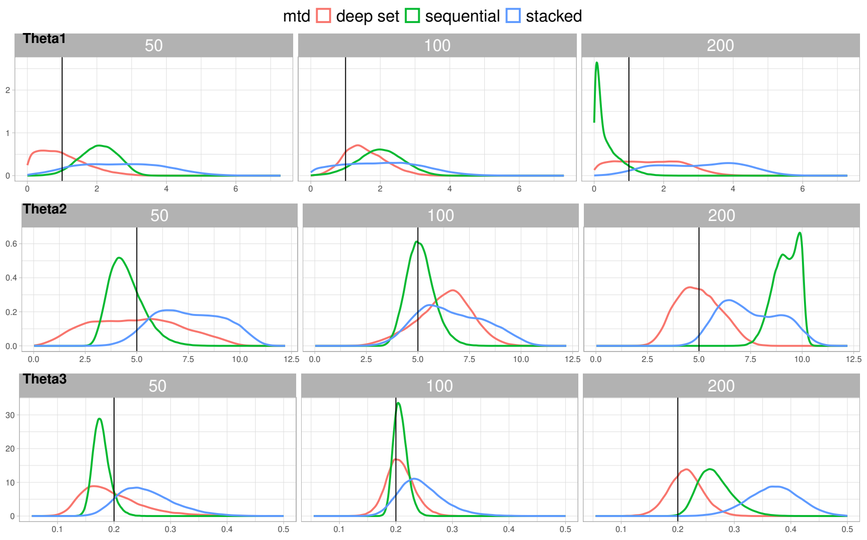


Figure 4: Posterior for M/G/1-queuing under different implementations of B-GAN

we mention earlier. We observe that the network is able to give some rough approximation when $n^* = 50$ and $n^* = 100$ but fails to learn when $n^* = 200$. For the batch sequential update, the input dimension is $5n^*/5 = n^*$, so it is less affected by the growing sample size compared with the stacking implementation. Since the batch updates also help gradually learning the high posterior density area, the batch update implementation produces the most concentrated posterior for $n^* = 50$ and $n^* = 100$. However, it also fails to learn when $n^* = 200$. This could happen due to the cumulated errors in the batch update.

For the deep set structure, it largely circumvent the first challenge of exploding network complexity and partially relieve the second challenge from incorporating the statistical knowledge of exchangeability into the design. It cannot avoid the last challenge of growing data size which could be overwhelming for users with limited storage or memory. We recommend combing the deep set structure with the sequential update for extremely large n to achieve maximum efficiency.

3. GAN Variational Bayes

Variational Bayes (VB) is an optimization-centric Bayesian inferential framework based on minimizing a divergence between approximate and real posteriors. VB typically reduces the infinite-dimensional optimization problem to a finite-dimensional one by forcing approximations into structured parametric forms. Implicit distributions (defined as probabilistic programs) have the potential to yield finer and tighter VB posterior approximations (Huszár, 2017; Kingma and Welling, 2013; Tran et al., 2017; Titsias and Ruiz, 2019). This section highlights the connection between the implicit variational Bayes inference and our B-GAN framework (Algorithm 1 and 2), both of which target the posterior.

The VB setup consists of an (intractable) likelihood $p_\theta^{(n)}(\cdot)$, prior $\pi(\theta)$ and a class of posterior approximations $q_{\boldsymbol{\beta}}(\theta \mid X_0)$ indexed by $\boldsymbol{\beta}$. We are recycling the notation of $\boldsymbol{\beta}$ here to highlight the connection between the GAN generator and the implicit variational generator.

The goal of the VB approach is to find a set of parameters $\boldsymbol{\beta}^*$ that maximize the evidence lower bound (ELBO) to the marginal likelihood

$$\log \pi(X_0) \geq \mathcal{L}(\boldsymbol{\beta}) \equiv \int \log \left(\frac{\pi(X_0, \theta)}{q_{\boldsymbol{\beta}}(\theta \mid X_0)} \right) q_{\boldsymbol{\beta}}(\theta \mid X_0) \mathrm{d}\theta. \tag{9}$$

The tightness of the inequality (for the best approximating model within the class) increases with expressiveness of the inference model $q_{\boldsymbol{\beta}}(\cdot)$, where the equality occurs when $q_{\boldsymbol{\beta}}(\theta \mid X_0) = \pi(\theta \mid X_0)$. Writing the ELBO $\mathcal{L}(\boldsymbol{\beta}) = -\mathrm{KL}\left(q_{\boldsymbol{\beta}}(\theta \mid X_0) \| \pi(\theta \mid X_0)\right) + C$ in terms of Kullback-Leibler discrepancy, we have

$$\boldsymbol{\beta}^* = \arg\max_{\boldsymbol{\beta}} \mathcal{L}(\boldsymbol{\beta}) = \arg\min E_{\theta \sim q_\beta(\theta \mid X_0)} \log \left(\frac{\pi(\theta \mid X_0)}{q_\beta(\theta \mid X_0)} \right) \tag{10}$$

We estimate this expectation using K Monte Carlo draws $\theta_k \sim q_\beta(\theta \mid X_0), k = 1, \ldots, K$, yielding the empirical objective

$$\hat{\boldsymbol{\beta}}_{\mathrm{VB}} = \arg\min_{\boldsymbol{\beta}} \frac{1}{K} \sum_{\theta_k \sim q_\beta(\theta \mid X_0)} \log \left(\frac{\pi(\theta_k \mid X_0)}{q_\beta(\theta_k \mid X_0)} \right) \tag{11}$$

where $q_\beta(\theta \mid X_0)$ is generated implicitly via $Z_k \sim \pi_Z$ and $\theta_k = g(Z_k, X_0)$.

However, contrary to conventional VB algorithms, we cannot directly evaluate (11) since neither $g_{\boldsymbol{\beta}}(\theta \mid X_0)$ nor $\pi(\theta \mid X_0)$ are tractable. To address this, we leverage contrastive learning (Bickel et al., 2007) to estimate the density ratio between two implicit distributions. Here we first show how the loss function in (11) can be re-written with density ratio estimator, and then we delve into details on how we actually implement that due to training stability concerns.

Recall the "oracle classifier" $d^*_{g_{\boldsymbol{\beta}}}$ in Theorem 10, we have

$$\frac{d^*_{g_{\boldsymbol{\beta}}}(X, \theta)}{1 - d^*_{g_{\boldsymbol{\beta}}}(X, \theta)} = \frac{\pi(X, \theta)}{q_{\boldsymbol{\beta}}(\theta \mid X) \pi(X)},$$

which allows us to re-write the VB loss in (11) as

$$\hat{\boldsymbol{\beta}}_{\mathrm{VB}} = \arg\min_{\boldsymbol{\beta}} \frac{1}{K} \sum_{k=1}^{K} \mathrm{logit}(d^*_{g_{\boldsymbol{\beta}}}(X_0, \theta_k)). \tag{12}$$

Since the "oracle classifier" $d^*_{g_{\boldsymbol{\beta}}}$ is unavilable, it has to be estimated by solving a classification problem using a particular class of classifiers $\mathcal{D} = \{d_{\boldsymbol{\phi}} : (\mathcal{X} \times \Omega) \to (0,1); \boldsymbol{\phi} \in \mathbb{R}^C\}$, for instance, neural networks parameterized by ϕ. This adversarial idea – replacing components of the ELBO with learned discriminators – has been explored in earlier works (Mescheder et al., 2017; Huszár, 2017; Tran et al., 2017), typically for latent variable models. Here, instead of performing maximum likelihood estimation (Mescheder et al., 2017), we focus purely on VB inference with approximate posteriors $q_{\boldsymbol{\beta}}(\theta \mid X_0)$ when θ is assigned a prior.

While (12) provides a clear and elegant formulation, it encounters two training challenges: (1) classifier estimation when θ_0 is unknown, and (2) instability under KL loss. To tackle these issues, we make two modifications: (1) we use the VB loss as a regularization term, and (2) we replace the discriminator function with the critic function. We elaborate the details below.

To address (1), note that we cannot train a classifier using the pair $\{\theta_0, X_0\}$ and $\{g_\beta(Z, X_0), X_0\}$ since θ_0 is unknown. Instead, we train $d_\phi(X,\theta)$ on samples from the joint $\pi(X,\theta)$ versus $q_{\boldsymbol{\beta}}(\theta \mid X)\pi(X)$, and then plug in X_0 as the condition variable. This gives rises to an adversarial game: the *Generator* $g_{\boldsymbol{\beta}}(Z,X)$ maximizes the ELBO conditioned on X_0, and the *Discriminator* $d_{\boldsymbol{\phi}}(X,\theta)$ distinguishes between true and generated joint distributions. However, such training schedule makes it challenging to maintain the balance between the generator network and the discriminator network, which is extremely intricate for GAN training. When the generator is learning locally and the critic function is learning globally, the balance is easily broken when the generator misbehaves away from the vicinity of X_0.

We resolve this issue by incorporating the ELBO a regularization term in addition to the original adversarial loss, which encourages the generator improves locally near X_0. Basically the training alternates between updating the discriminator to maximizing the GAN loss and updating the generator to minimize the $\{$GAN loss $+\lambda\cdot$ELBO loss$\}$, where λ controls the strength of the regularization.

To address (2), i.e., KL-based training instability, we consider a different Wasserstein formulation motivated by our theoretical results in Theorem 1. Our first term associated with the total variation bound

$$\mathcal{A}_1(\mathcal{F},\mathcal{G}) \equiv \mathbb{E} \inf_{\boldsymbol{\omega}: f_{\boldsymbol{\omega}} \in \mathcal{F}} \left\| \log \frac{\pi(\cdot \mid X)}{\pi_{\widehat{\boldsymbol{\beta}}_T}(\cdot \mid X)} - f_\omega(X,\cdot) \right\|_\infty$$

suggests that the critic function $f_\omega(X,\cdot)$ also approximates the log-likelihood ratio $\log \frac{\pi(\cdot \mid X)}{\pi_{\widehat{\boldsymbol{\beta}}_T}(\cdot \mid X)}$. This allows us to rewrite the ELBO term in (12) as

$$\widehat{\mathcal{L}}(\beta) = \frac{1}{K} \sum_{\theta_i = g_\beta(Z_i, X_0)} f_{\hat{\omega}}(X_0, \theta_i). \tag{13}$$

where $f_{\hat{\omega}}$ is a trained critic approximating the log-density ratio, based on samples from joint distributions $\pi(X,\theta)$ and $q_{\boldsymbol{\beta}}(\theta \mid X)\pi(X)$.

Putting it all together, we present the VB-regularized WGANs training procedure. We initialize the generator and critic networks at $\boldsymbol{\beta}^{(0)}$ and $w^{(0)}$ and generate the ABC reference table $\{(\theta_j, X_j)\}_{j=1}^T \overset{\text{iid}}{\sim} \pi(\theta, X)$ with $\{Z_j\}_{j=1}^T \overset{\text{iid}}{\sim} \pi_Z(\cdot)$. Then, for the t-th iteration, update

$$\begin{aligned} w^{(t)} &= \arg\max_{w: f_w \in \mathcal{F}} \sum_{j=1}^T f_w(X_j, g_{\beta^{(t)}}(Z_j, X_j)) - \sum_{j=1}^T f_w(X_j, \theta_j) \\ \beta^{(t)} &= \arg\min_{\beta: g_\beta \in \mathcal{G}} \underbrace{\sum_{j=1}^T f_{w^{(t)}}(X_j, g_\beta(Z_j, X_j))}_{\text{GAN Loss}} + \lambda \cdot \underbrace{\sum_{j=T+1}^{T+K} f_{w^{(t)}}(X_0, g_\beta(Z_j, X_0))}_{\text{approx VB loss in (13)}} \end{aligned} \tag{14}$$

This formulation effectively balances global learning from simulated datasets with local adaptation to the observed data X_0, offering advantages over purely adversarial schemes

Algorithm 3 *GAN Variational Bayes (Wasserstein Version)*

```
INPUT
    Prior $\pi(\theta)$, observed data $X_0$ and noise distribution $\pi_Z(z)$
Training
    Initialize network parameters $\boldsymbol{\omega}^{(0)} = 0$ and $\boldsymbol{\beta}^{(0)} = 0$
Pilot Run
    Apply Algorithm 1 with $\pi(\theta)$ to learn $\widehat{g}_{pilot}(\cdot)$
Reference Table
    Generate pairs $\{(X_j, \theta_j)\}_{j=1}^T$ where $\theta_j = \widehat{g}_{pilot}(Z_j, X_0)$ for $Z_j \sim \pi_Z$ and $X_j \sim P_{\theta_j}^{(n)}$.
WGAN Training
For $t = 1, \dots, N$:
    Critic Update ($N_{\text{critic}}$ steps): Same as in Critic Update of Algorithm 1
    Generator Update (single step)
        Generate noise $Z_j \sim \pi_Z(z)$ for $j = 1, \dots, N$.
        Update $\boldsymbol{\beta}^{(t)}$ by applying stochastic gradient descent on (14).
Posterior Simulation
    Simulate $\{Z_i\}_{i=1}^M \overset{\text{iid}}{\sim} \pi_Z(z)$ and set $\widetilde{\theta}_i = g_{\boldsymbol{\beta}^{(N)}}(Z_i, X_0)$.
    Estimate $\hat{w}_i$ using either (7) or (8).
OUTPUT
    Pairs of posterior samples and weights $(\widetilde{\theta}_1, \hat{w}_1), \dots, (\widetilde{\theta}_M, \hat{w}_M)$
```

such as Algorithm 1, which ignore X_0 altogether. Another perspective on (14) is that it can also be viewed as a weighted loss between simulated datasets $\{X_j\}_{j=1}^T$ and the observed dataset X_0. We would thereby expect this version to work better than Algorithm 1 which does not use X_0 at all.

Indeed, on the toy simulated example in Example 2 we can see that the VB variant produces tighter reconstructions relative to the B-GAN approach. The performance, however, is not uniformly better than Algorithm 2. We provide a snapshot from another repetition in Figure 9 (Appendix C), where the spikiness of B-GAN-VB (especially obvious when estimating θ_2) may explain why the MMDs between B-GAN-VB and the true posteriors are larger than for B-GAN-2S in Figure 3. The algorithm is described in Algorithm 3. There are also other variants of adversarial VB using the classification-based KL loss or the Wasserstein loss Mescheder et al. (2017); Huszár (2017), but we have found that they are less stable to train and thus exclude them from the comparisons.

4. Theory

The purpose of this section is to provide theoretical solidification for the implicit posterior simulators in Algorithm 1, 2 and 3. We will quantify the typical (squared) total variation (TV) distance between the actual posterior and its approximation and illustrate that with carefully chosen neural generators and discriminators, the expected total variation distance vanishes as $n \to \infty$. We will continue denoting $X^{(n)}$ simply by X.

We define with $\nu = P_\theta^{(n)} \otimes \Pi$ the joint measure on $\mathcal{X} \times \Theta$ with a density $\pi(X,\theta) = p_\theta^{(n)}(X)\pi(\theta)$. The goal is to approximate this measure with μ_g defined semi-implicitly through a density function $\pi_g(X,\theta) = \pi(X)\pi_g(\theta \mid X)$ where $\pi(X) = \int \pi(X,\theta)\mathrm{d}\theta$ is the marginal likelihood and where the samples from the density $\pi_g(\theta \mid X)$ are obtained by the *Generator* in Definition 1. Thus, by keeping the marginal distribution the same, the distribution $\pi_g(\theta \mid X)$ is ultimately approximating the conditional distribution $\pi(\theta \mid X)$. The quality of the approximation will be gauged under the integral probability metric[1] (IPM)

$$d_{\mathcal{F}}(\mu_g, \nu) = \sup_{f \in \mathcal{F}} \left| E_{(X,\theta)\sim\mu_g} f(X,\theta) - E_{(X,\theta)\sim\nu} f(X,\theta) \right|, \tag{15}$$

where $\mathcal{F}$ is a class of evaluation metrics (such as Lipschitz-1 functions that yield the Wasserstein-1). The IPM metric (15), due to shared marginals of the two distributions, satisfies

$$d_{\mathcal{F}}(\mu_g, \nu) \leq E_X d_{\mathcal{F}}(\mu_g(X), \nu(X)), \tag{16}$$

where $\mu_g(X)$ and $\nu(X)$ denote the *conditional* measures with densities $\pi_g(\theta \mid X)$ and $\pi(\theta \mid X)$. At the population level, the B-GAN (Algorithm 1) minimax game finds an equilibrium

$$g^* = \min_{g \in \mathcal{G}} d_{\mathcal{F}}(\mu_g, \nu),$$

where $\mathcal{G}$ is a class of generating functions (that underlie the implicit distribution μ_g).

Typically, both $\mathcal{F}$ and $\mathcal{G}$ would be parameterized by neural networks with the hope that the discriminator networks can closely approximate the metric $d_{\mathcal{F}}$ and that the generator networks can flexibly represent distributions. In practice, one would obtain a data-driven estimator based on the *empirical* distribution $\bar{\nu}_T$ of (θ_j, X_j) for $1 \leq j \leq T$ and the *empirical* distribution $\bar{\mu}_g$ of $(g(Z_j, X_j), X_j)$ where $Z_j \sim \pi_Z$ for $1 \leq j \leq T$. Assuming that $\mathcal{G} = \{g_{\boldsymbol{\beta}} : \boldsymbol{\beta} \in \mathbb{R}^G\}$, the B-GAN estimator can be written as

$$\widehat{\boldsymbol{\beta}}_T \in \arg\min_{\boldsymbol{\beta}: g_{\boldsymbol{\beta}} \in \mathcal{G}} \max_{\boldsymbol{\omega}: f_{\boldsymbol{\omega}} \in \mathcal{F}} \left| E_{\bar{\mu}_g} f_{\boldsymbol{\omega}}(X, g_{\boldsymbol{\beta}}(Z,X)) - E_{\bar{\nu}_T} f_{\boldsymbol{\omega}}(X,\theta) \right|. \tag{17}$$

For brevity, we will often denote the generator density $\pi_{g_{\boldsymbol{\beta}}}(\cdot)$ (see Definition 1) simply by $\pi_{\boldsymbol{\beta}}(\cdot)$ and similarly for $\mu_{g_{\boldsymbol{\beta}}}$. The next Theorem provides an upper bound on the typical total variation (TV) distance between true and the approximated posterior measures $\nu(X_0)$ and $\mu_{\widehat{\boldsymbol{\beta}}_T}(X_0)$ with densities $\pi(\theta \mid X_0)$ and $\pi_{\widehat{\boldsymbol{\beta}}_T}(\theta \mid X_0)$, respectively. The total variation distance can be upper bounded by three terms: (1) the ability of the critic to tell the true model apart from the approximating model

$$\mathcal{A}_1(\mathcal{F}, \mathcal{G}) \equiv \mathbb{E} \inf_{\boldsymbol{\omega}: f_{\boldsymbol{\omega}} \in \mathcal{F}} \left\| \log \frac{\pi(\cdot \mid X)}{\pi_{\widehat{\boldsymbol{\beta}}_T}(\cdot \mid X)} - f_\omega(X, \cdot) \right\|_\infty \tag{18}$$

(2) the ability of the generator to approximate the average true posterior

$$\mathcal{A}_2(\mathcal{G}) \equiv \inf_{\boldsymbol{\beta}: g_{\boldsymbol{\beta}} \in \mathcal{G}} \left[E_X \left\| \log \frac{\pi_{\boldsymbol{\beta}}(\cdot \mid X)}{\pi(\cdot \mid X)} \right\|_\infty \right]^{1/2}, \tag{19}$$

1. The absolute value can be removed due to the Monge-Rubinstein dual (Villani, 2008).

and (3) the complexity of the (generating and) critic function classes measured the pseudo-dimension $Pdim(\cdot)$ and defined in Definition 2 in Bartlett et al. (2017). Intuitively, pseudo dimension is the maximum number of points you can assign arbitrary real values to and still be able to fit them exactly using functions from the hypothesis space. For example, if the hypothesis space is the set of all linear functions, its pseudo dimension is 2, since you can fit exactly 2 arbitrary points (with real-valued outputs), but no more than that, with a single linear function.

Note that the L^∞ norm in both (18) and (19) are both taken with respect to θ. We use $\cdot$ in place of θ to avoid ambiguity. We denote with $\mathcal{H} = \{h_{\boldsymbol{\omega},\boldsymbol{\beta}} : h_{\boldsymbol{\omega},\boldsymbol{\beta}}(Z,X) = f_{\boldsymbol{\omega}}(g_{\boldsymbol{\beta}}(Z,X),X)\}$ a structured composition of networks $f_{\boldsymbol{\omega}} \in \mathcal{F}$ and $g_{\boldsymbol{\beta}} \in \mathcal{G}$.

Theorem 1 *Let $\widehat{\boldsymbol{\beta}}_T$ be as in (17) where $\mathcal{F} = \{f : \|f\|_\infty \le B\}$ for some $B > 0$. Denote with $\mathbb{E}$ the expectation with respect to empirical measure on $\{(\theta_j, X_j)\}_{j=1}^T$ and $\{Z_j\}_{j=1}^T$ in the reference table. Assume that the prior satisfies*

$$\Pi[B_n(\theta_0;\epsilon)] \ge \mathrm{e}^{-C_2 n\epsilon^2} \quad \textit{for some } C_2 > 0 \textit{ and } \epsilon > 0, \tag{20}$$

where the KL neighborhood $B_n(\theta_0;\epsilon)$ is defined as

$$B_n(\theta_0;\epsilon) = \{\theta \in \Theta : KL(P_{\theta_0}^{(n)} \| P_\theta^{(n)}) \le n\epsilon^2, V_{2,0}(P_{\theta_0}^{(n)}, P_\theta^{(n)}) \le n\epsilon^2\}. \tag{21}$$

Then for $T \ge Pdim(\mathcal{F}) \vee Pdim(\mathcal{H})$ we have for any $C > 0$

$$P_{\theta_0}^{(n)} \mathbb{E}\, d_{TV}^2 \left(\nu(X_0), \mu_{\widehat{\boldsymbol{\beta}}_T}(X_0)\right) \le \mathcal{C}_n^T(\epsilon, C),$$

where, for some $\widetilde{C} > 0$ and $Pmax \equiv Pdim(\mathcal{F}) \vee Pdim(\mathcal{H})$,

$$C_n^T(\epsilon, C) = \frac{1}{C^2 n\epsilon^2} + \frac{\mathrm{e}^{(1+C_2+C)n\epsilon^2}}{4}\left[2\mathcal{A}_1(\mathcal{F},\mathcal{G}) + \frac{B\,\mathcal{A}_2(\mathcal{G})}{\sqrt{2}} + 4\widetilde{C}\,B\sqrt{\frac{\log T \times Pmax}{T}}\right].$$

Proof Section B.1

Remark 3 *Zhou et al. (2022) provided theoretical results for the total variation distance between* joint *distributions using the Jensen-Shannon version of conditional GANs. Contrastingly, we provide frequentist-Bayesian results, quantifying the typical squared total variation distance between the true and approximate posteriors after integrating over the data generating process. In addition, we use the Wasserstein GANs, building on oracle inequalities established in Liang (2021).*

Interestingly, while the GAN approach is trained *without* the knowledge of X_0, Theorem 1 obtains a finite-sample (fixed n and d) upper bound for the typical TV distance *after plugging* X_0 into the sampler. This frequentist result has required the prior concentration condition (20) which could be avoided if we were willing to assume bounded likelihood ratios (Kaji and Ročková, 2022) or uniform estimation error (Frazier et al., 2024). With bounded likelihood ratios, we would then also not have the first term and the exponential multiplier in the bound.

From (16), it can be seen that Algorithm 1 targets a lower bound to the *average* Wasserstein distance between the posterior $\pi(\theta \mid X)$ and $\pi_g(\theta \mid X)$ after marginalizing over $\pi(X)$. In other words, Algorithm 1 *is not* necessarily targeting $\pi(\theta \mid X_0)$. The 2-step enhancement in Algorithm 2 provides more data draws in the ABC table that more closely resemble X_0. Theorem 1 applies to Algorithm 2 as well with slight modifications.

Corollary 4 *(2step B-GAN) Assume that $\widehat{\boldsymbol{\beta}}_T$ in* (17) *is learned under the proposal distribution $\widetilde{\pi}(\theta)$ and denote with $\widetilde{\mathbb{E}}$ the expectation of the reference table under $\widetilde{\pi}(\theta)$. Assume that the original prior $\pi(\theta)$ satisfies* (20)*. Then the importance re-weighted posterior reconstruction from Algorithm 2 satisfies the statement in Theorem 1 with $\mathbb{E}$ replaced by $\widetilde{\mathbb{E}}$ and with $\widetilde{\mathcal{A}}_2(\mathcal{G}) \equiv \inf_{\boldsymbol{\beta}: g_{\boldsymbol{\beta}} \in \mathcal{G}} \left[\int_{\mathcal{X}} \widetilde{\pi}(X) \left\| \log \frac{\pi_{\boldsymbol{\beta}}(\theta \mid X)}{\pi(\theta \mid X)} \right\|_\infty \mathrm{d}X \right]^{1/2}$ and $\widetilde{\mathcal{A}}_1(\mathcal{F}, \mathcal{G}) \equiv \widetilde{\mathbb{E}} \inf_{\boldsymbol{\omega}: f_{\boldsymbol{\omega}} \in \mathcal{F}} \left\| \frac{\pi(\theta)}{\widetilde{\pi}(\theta)} \log \frac{\pi(\theta \mid X)}{\pi_{\widehat{\boldsymbol{\beta}}_T}(\theta \mid X)} - f_\omega(X, \theta) \right\|_\infty$.*

Proof Section B.2

The (nonasymptotic) bounds for the *typical* TV distance in Theorem 1 and Corollary 4 are not refined enough to fully appreciate the benefits of the 2-step enhancement. In Remark 11 in Appendix (Section B.3), we provide an intuitive explanation for why the 2-step refinement version works so well in practice on each particular realization of X_0 (not only on average over many realizations X_0). We also provide a version of Theorem 1 for adversarial variational Bayes (Algorithm 3) in Theorem 2 (Section B.5).

Theorem 1 provides intuition for how the quality of the generator and discriminator networks affects the total variation distance. It also suggests how large T should be (relative to n and d) to assure that the average squared TV distance is arbitrarily small (vanishing with n). Recall that the training data size T can be chosen by the user. In Remark 7 below, we will focus on specific deep learning architectures and provide a condition for choosing T such that the upper bound in Theorem 1 is $o(1)$ as $T \to \infty$ for a suitable choice of ϵ and C (potentially depending on n) and for n, d fixed. Instead of fixing n and allowing T to grow, in Corollary 6 we fix T (as a function of n) and allow n to grow and show that the upper bound is $o(1)$ as $n \to \infty$. We use the ReLU activation function $\sigma_{ReLU}(x) = \max\{x, 0\}$ for both the critic and generator networks, which have good approximation properties (Schmidt-Hieber, 2020).

Definition 5 *We denote with $\mathcal{F}_L^B(S, W)$ a class of feed-forward ReLU neural networks f with depth L (i.e. the number of hidden layers plus one), width W and size S (total number of parameters in the network) such that $\|f\|_\infty \leq B$. The width is defined as $W = \max\{w_0, \ldots, w_L\}$ where w_l is the width of the l^{th} layer with w_0 the input data dimension and w_L the output dimension. With $\mathcal{G}_L^B(S, W)$, we denote the leaky ReLU neural networks with the same meaning of parameters.*

The following Corollary warrants optimism when using neural networks for the generator and the discriminator. We formulate the Corollary in context of Algorithm 1 and note that a similar conclusion holds for Algorithm 2 as well. This Corollary shows that there exist deep learning architectures such that for large enough T (depending on n), the typical squared TV distance vanishes as $n \to \infty$.

Corollary 6 *Assume that the joint distribution $\pi(\theta, X)$ is realizable in the sense that there exists $g_{\boldsymbol{\beta}_0} \in \mathcal{G}_{L_0}^{B_0}(S_0, W_0)$ such that $\pi(\theta, X) = \pi_{g_{\boldsymbol{\beta}_0}}(\theta, X)$. Assume that $\mathcal{G}_{L^*}^{B^*}(S^*, W^*) \subseteq$*

$G_{L_0}^{B_0}(S_0, W_0)$ is a class of leaky ReLU generative networks indexed by $\boldsymbol{\beta}$ where $\|\boldsymbol{\beta}_0\|_\infty \vee \|\boldsymbol{\beta}\|_\infty \leq b$ for some $b > 0$. Assume that $\mathcal{F} = \mathcal{F}_L^B(S, W)$ are ReLU discriminator networks and π_Z is uniform on $[0,1]^d$. Assume the prior concentration (20) is satisfied with $\epsilon_n > 0$ such that $\epsilon_n = O(1/\sqrt{n})$. For each arbitrarily slowly increasing sequence C_n, there exists $L, S, W > 0$ and training data size T (depending on n) such that we have $P_{\theta_0}^{(n)} \mathbb{E} d_{TV}^2 \left(\nu(X_0), \mu_{\widehat{\boldsymbol{\beta}}_T}(X_0)\right) = o(1)$ as $n \to \infty$ for d fixed.

Proof Section B.4 in the Appendix.

Remark 7 *Using the same architecture as in Corollary 6, given n and d we can find the smallest ϵ such that the prior concentration is satisfied. Choosing $1/C = o(1)$ as $T \to \infty$ such that $\mathrm{e}^C \sqrt{(\log T \times Pmax)/T} = o(1)$ will yield an upper bound $C_n^T(\epsilon, C)$ that is $o(1)$ as $T \to \infty$ for fixed n and d.*

5. Performance Evaluation

This section demonstrates very promising performance of our B-GAN approaches in Algorithm 1 (B-GAN), Algorithm 2 (B-GAN-2S) and Algorithm 3 (B-GAN-VB) and on simulated examples relative to other Bayesian likelihood-free methods (plain ABC using summary statistics (SS); 2-Wasserstein distance ABC by Bernton et al. (2019); Sequential Neural Likelihood (SNL) (Papamakarios et al., 2019) with default settings). SNL (Papamakarios et al., 2019) employs autoregressive flow to estimate the likelihood function from simulated ABC reference table. The implementation details of our methods and the counterparts are described in Appendix E.4 for the Lotka-Volterra model and Appendix E.5 for the Boom-and-Bust model.

We evaluate the performance of different approximated posteriors in terms of three metrics: bias $\mathbb{E}\left|\hat{\theta}_i - \theta_i\right|$, width of the 95% credible intervals and its coverage and the Posterior Mass Concentration (PMC) in a small δ neighborhood of the true values $\boldsymbol{\theta}_0$. PMC is defined coordinate-wise as $\mathbb{P}(\hat{\theta}_j \in (\boldsymbol{\theta}_{0,j} - \delta_j, \boldsymbol{\theta}_{0,j} + \delta_j))$ for every $j = 1, \ldots, d$. PMC incorporates both bias and variance of the posterior approximation, and it is also helpful in understanding how much uncertainty has reduced compared to the prior. Since the examples we consider here all use uniform priors, we choose δ_j to be 5% of the width of those uniform priors to visualize how much uncertainty has reduced after observing X_0.

5.1 Lotka-Volterra Model

The Lotka-Volterra (LV) predator-prey model (Wilkinson, 2018) is one of the classical likelihood-free examples and describes population evolutions in ecosystems where predators interact with prey. The state of the population is prescribed deterministically via a system of ordinary differential equations (ODEs). Inference for such models is challenging because the transition density is intractable. However, simulation from the model is possible, which makes it a natural candidate for simulator-based inference methods.

The model monitors population sizes of predators x_t and prey y_t over time t. The changes in states are determined by four parameters $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_4)'$ controlling: (1) the rate $r_1^t = \theta_1 x_t y_t$ of a predator being born; (2) the rate $r_2^t = \theta_2 x_t$ of a predator dying; (3) the rate $r_3^t = \theta_3 y_t$ of a prey being born; (4) the rate $r_4^t = \theta_4 x_t y_t$ of a prey dying. Given the initial

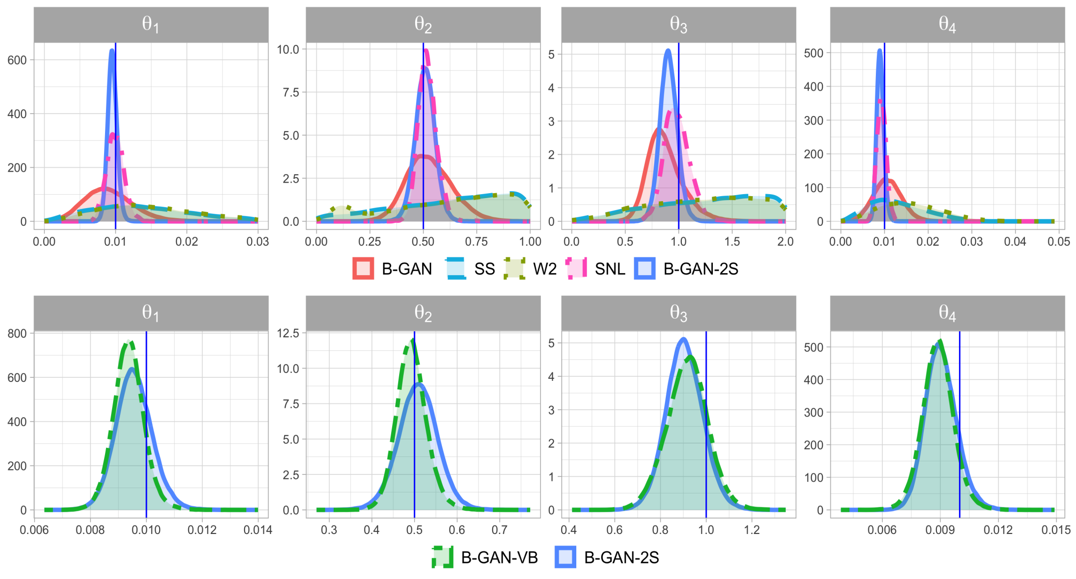


Figure 5: Approximate posterior densities under the Lotka-Volterra Model. The true parameter vector (marked by vertical lines) is $\boldsymbol{\theta}_0 = (0.01, 0.5, 1, 0.01)'$.

population sizes (x_0, y_0) at time $t = 0$, the dynamics can be simulated using the Gillespie algorithm (Gillespie, 1977), which is a stochastic discrete-time Markov chain model. The algorithm samples times to an event from an exponential distribution (with a rate $\sum_{j=1}^{4} r_j^t$) and picks one of the four reactions with probabilities proportional to their individual rates r_j^t. We use the same setup as Kaji and Ročková (2022) where each simulation is started at $x_0 = 50$ and $y_0 = 100$ and state observations are recorded every 0.1 time units for a period of 20 time units, resulting in a series of 201 observations each.

The real data X_0 are generated with true values $\boldsymbol{\theta}_0 = (0.01, 0.5, 1, 0.01)'$. The data vector X_0 is stretched into one $(201 \times 2 \times n^*)$ vector, where n^* is number of i.i.d. copies of time-series observations. The advantage of our approach is that it can be used even for $n^* = 1$ when other methods (such as Kaji and Ročková (2022)) cannot. We focus on the $n^* = 1$ case here. We use an informative prior $\theta \in U(\Xi)$ with a restricted domain $\Xi = [0, 0.1] \times [0, 1] \times [0, 2] \times [0, 0.1]$ to make it easier for classical ABC methods (see Kaji and Ročková (2022)) and to make the GAN training more efficient. Previous analyses (Papamakarios and Murray, 2016) suggested summary statistics as the mean, log-variance, autocorrelation (at lag 1 and 2) of each series as well as their correlation. Papamakarios et al. (2019) also built their sequential neural network on top of this set of summary statistics. We also build our model on the summary statistics, since it performs better than using the time series empirically. This example is quite challenging due to the spikiness of the likelihood in very narrow areas of the parameter space (as explained in Kaji and Ročková (2022)).

A typical snapshot (for one particular data realization) of the approximated posteriors is given in Figure 5 and the summary statistics averaged over 10 repetitions are reported in Table 1. Since we do not have access to the true posterior, we look at the width of

		B-GAN	B-GAN-RL	B-GAN-VB	SNL	SS	W2
	Bias	0.30	**0.08**	**0.08**	0.11	0.96	1.10
$\theta_1 = 0.01$	CI	1.34	0.28 (0.9)	**0.25 (0.9)**	0.44	3.80	4.02 (0.9)
	PMC	0.82	**1.00**	**1.00**	**1.00**	0.40	0.38
	Bias	0.91	0.46	**0.43**	0.45	2.49	2.42
$\theta_2 = 0.5$	CI	0.40	0.18	**0.14**	0.17	0.91	0.84
	PMC	0.33	0.62	**0.64**	**0.64**	0.10	0.10
	Bias	0.22	**0.12**	0.13	0.13	0.49	0.47
$\theta_3 = 1$	CI width	0.77	**0.41**	**0.41**	0.48	1.76	1.73
	PMC	0.27	**0.48**	**0.48**	0.47	0.11	0.11
	Bias	0.29	**0.12**	**0.12**	0.15	0.68	0.79
$\theta_4 = 0.01$	CI width	1.29	0.38	**0.35**	0.52	2.72	2.82
	PMC	0.83	**0.99**	**0.99**	0.98	0.51	0.42

Table 1: Summary statistics of the approximated posteriors under the Lotka-Volterra model (averaged over 10 repetitions). Bold fonts mark the best model of each column. The coverage of the 95% credible intervals are 1 unless otherwise noted in the parentheses.

the 95% credible interval, its coverage (proportion of the 10 replications such that the true value is inside the credible interval), bias of the posterior mean and the posterior mass concentration. Again, we observe that B-GAN-2S and B-GAN-VB outperforms B-GAN with smaller biases, tighter variances and higher probability mass concentrated near $\boldsymbol{\theta}_0$. In Figure 5, B-GAN-VB appears to have smaller bias than B-GAN-2S when estimating all parameters and also consistently outperforms SNL. The computation cost requirements are compared in Section E.7.

5.2 Simple Recruitment, Boom and Bust

Our second demonstration is on the simple recruitment, boom and bust model (Fasiolo et al., 2018). The model is prescribed by a discrete stochastic process, characterizing the fluctuation of the population size of a certain group over time. Given the population size N_t and parameter $\boldsymbol{\theta} = (r, \kappa, \alpha, \beta)'$, the population size at the next timestep N_{t+1} follows the following distribution

$$N_{t+1} \sim \begin{cases} \text{Poisson}\big(N_t(1+r)\big) + \epsilon_t, & \text{if } N_t \leq \kappa \\ \text{Binom}\big(N_t, \alpha\big) + \epsilon_t, & \text{if } N_t > \kappa \end{cases},$$

where $\epsilon_t \sim \text{Pois}(\beta)$ is a stochastic arrival process, with rate $\beta > 0$. The population grows stochastically at rate $r > 0$, but it crashes if the carrying capacity κ is exceeded. The survival probability $\alpha \in (0, 1)$ determines the severity of the crash. Over time the population fluctuates between high and low population sizes for several cycles.

This model has been shown to be extra challenging for both synthetic likelihood (SL) methods and ABC methods in Fasiolo et al. (2018). The distribution of the statistics is far from normal which breaks the normality assumption of SL. In addition, the authors show that ABC methods require exceedingly low tolerances and low acceptance rates to achieve satisfying accuracy.

We first run the simulation study using the setup in An et al. (2020). The real data X_0 is generated using parameters $r = 0.4, \kappa = 50, \alpha = 0.09$ and $\beta = 0.05$, and the prior distribution is uniform on $[0,1] \times [10,80] \times [0,1] \times [0,1]$. The observed data has 250 time-steps, with 50 burn-in steps to remove the transient phase of the process.

Previous analyses of the model suggested various summary statistics, including the mean, variance, skewness, kurtosis of the data, lag 1 differences, and lag 1 ratios (An et al., 2020). We use them in SS and SNL methods. We have explored three types of input: the time series itself, the time series in conjunction with their summary statistics, and the summary statistics only. We find that the network built on the summary statistics appears to perform the best, thus we only include the results from this network here.

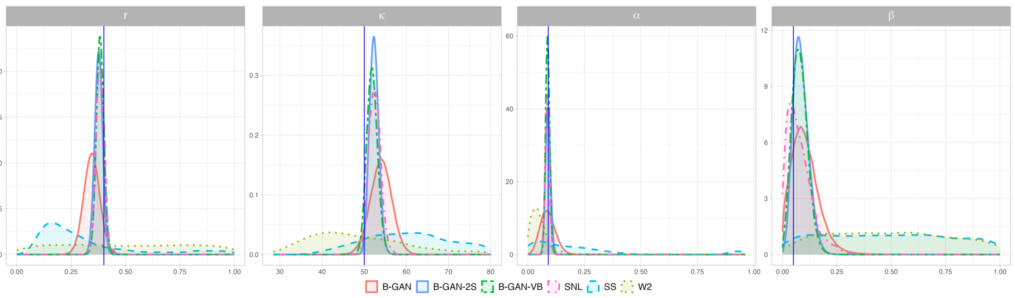


Figure 6: Approximate posterior densities under the Boom-and-Bust Model. The true parameter is $\boldsymbol{\theta}_0 = (0.4, 50, 0.09, 0.05)'$.

We include SS, W2, SNL as competitors for comparisons. One snapshot of the approximate posterior densities is provided in Figure 6. We report the performance summary averaged over 10 repetitions in Table 2. ABC methods struggle to identify the parameters and provide very flat posteriors. The vanilla B-GAN is able to identify the correct location of parameters but with rather wide credible intervals. We observe great improvements after applying either the 2-step refinements or the VB refinements. For this example, B-GAN-VB performs the best in estimating r and β, while B-GAN-RL approximates κ better and SNL estimates α slighly better than B-GAN-VB. The BnB model is more challenging to learn, compared with the LV model. Potentially the performance of two refinements can be further improved if we add more sequential refinement steps with fewer training epoches to gradually modify the proposal distribution.

6. Empirical Analysis: the Susceptible-Infected-Recovered (SIR) epidemic model with application to the common cold data

In this section, we illustrate our approach on the problem of estimating the three-parameter Susceptible-Infected-Recovered (SIR) epidemic model and apply our methods to the 21-day common-cold outbreak data in Tristan da Cunha island from October 1967 (Hammond & Tyrrell (1971); Shibli et al. (1971)). We provide the data in Table 3, which tracks the number of infected and recovered individuals.

The SIR model categorizes hosts into three statuses at time t. Individuals are considered susceptible (S), if they are able to be infected by the pathogen, infected (I) if currently

		B-GAN	B-GAN-RL	B-GAN-VB	SNL	SS	W2
$r = 0.4$	Bias ($\times 10^{-1}$)	0.44	0.26	**0.24**	**0.24**	2.16	2.59
	CI width ($\times 10^{-1}$)	1.65	0.81 (0.9)	**0.76 (0.9)**	0.93	8.26	9.49
	PMC	0.63	0.88	**0.91**	0.90	0.08	0.09
$\kappa = 50$	Bias	3.03	**1.42**	1.56	1.52	10.60	10.16
	CI width	11.00	**4.33 (0.9)**	5.09	5.37	37.17	43.20
	PMC	0.66	**0.96**	0.94	0.94	0.22	0.19
$\alpha = 0.1$	Bias ($\times 10^{-2}$)	2.92	1.29	1.05	**1.01**	15.08	5.46
	CI width ($\times 10^{-1}$)	1.37	0.50	0.39	**0.38**	9.18	2.77
	PMC	0.83	**1.00**	**1.00**	**1.00**	0.31	0.55
$\beta = 0.05$	Bias ($\times 10^{-1}$)	1.20	**0.96**	1.01	1.28	4.41	3.92
	CI width	0.39 (0.9)	0.24 (0.7)	**0.23 (0.8)**	0.39 (0.9)	0.95	0.86 (0.5)
	PMC	0.44	**0.54**	**0.54**	0.52	0.11	0.12

Table 2: Summary statistics of the approximated posteriors under the Boom-and-Bust model (averaged over 10 repetitions). Bold fonts mark the best model of each column. The coverage of the 95% credible intervals are 1 unless otherwise noted in the parentheses.

Day	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
Infected	1	1	3	7	6	10	13	13	14	14	17	10	6	6	4	3	1	1	1	1	0
Recovered	0	0	0	0	5	7	8	13	13	16	16	24	30	31	33	34	36	36	36	36	37

Table 3: Common cold outbreak data in Tristan da Cunha island (1967).

infected with the pathogen or recovered (R) if they have successfully cleared the pathogen. SIR models and their variants, in both deterministic and stochastic forms, are among the most fundamental epidemiological models and have found use describing diseases as diverse as influenza, herpes and malaria. We adopt the Model (3.12) in Toni et al. (2009), which has the highest probability, to describe the dynamics. This model introduces a latent state where individuals are infected but not yet infectious, to account for the delay between infection and the ability to infect others.

The deterministic SIR model without demographics can be written mathematically as

$$\frac{\mathrm{d}S(t)}{\mathrm{d}t} = -\beta I(t)S(t), \quad \frac{\mathrm{d}L(t)}{\mathrm{d}t} = \beta I(t)S(t) - \delta L(t), \quad \frac{\mathrm{d}I(t)}{\mathrm{d}t} = \delta L(t) - \gamma I(t), \quad \frac{\mathrm{d}R(t)}{\mathrm{d}t} = \gamma I(t),$$

where $S(t), L(t), I(t), R(t)$ are the numbers of susceptible, latent, infected and recovered individuals in the population at time t (days), and δ is the rate at which latent individuals become infectious. The parameter β is the transmission rate, and γ is the recovery rate.

Similar to the Lotka-Volterra model, we use stochastic discrete-time Markov chain to simulate the dynamics. The model is governed by four parameters $\theta = (\beta, \gamma, \delta, S(0))'$, where $S(0)$ is the number of susceptible individuals at week 0. Since the data in Table 3 only collected the number of infected and recovered individuals, the number of susceptible individuals is not directly observed and thus $S(0)$ needs to be estimated.

Following the setting in Toni et al. (2009), we place a prior on the parameters θ as $S(0) \sim \mathrm{U}[37, 100], \beta \sim \mathrm{U}[0, 3], \gamma \sim \mathrm{U}[0, 3], \delta \sim \mathrm{Unif}[0, 5]$. We include the plot of approximated posterior in Figure 7. For β and $S(0)$, the posterior seems to be unimodal, similar to the findings in Toni et al. (2009), and B-GAN-VB provides the tightest credible intervals. The posterior of γ and δ appears to be multimodal, which makes the inference more challenging and the posterior we obtain here is more dispersed than that in Toni et al. (2009).

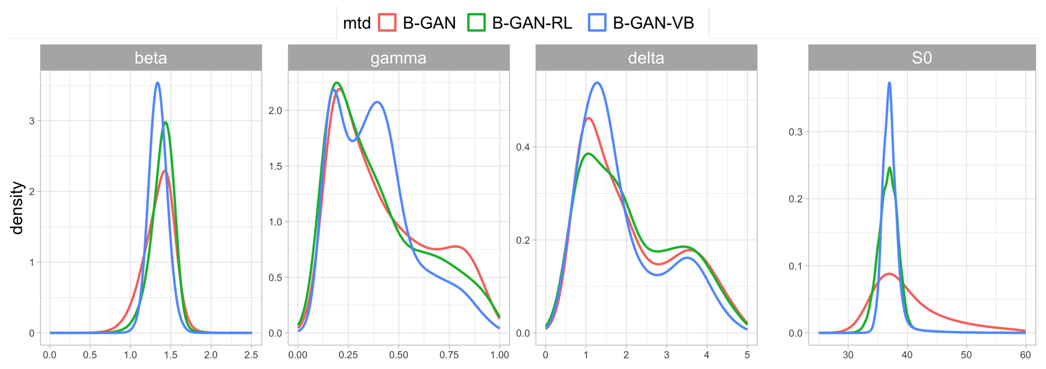


Figure 7: Approximated posterior densities under the SIR model.

To further investigate the quality of the approximated posteriors, we examine the predictive performance of the fitted models. We simulate 100 datasets from the posterior predictive distribution and compare them with the observed data X_0. The results are provided in Figure 8. We observe that for all methods, the simulated datasets cover the observed data, with B-GAN-VB providing the best fit.

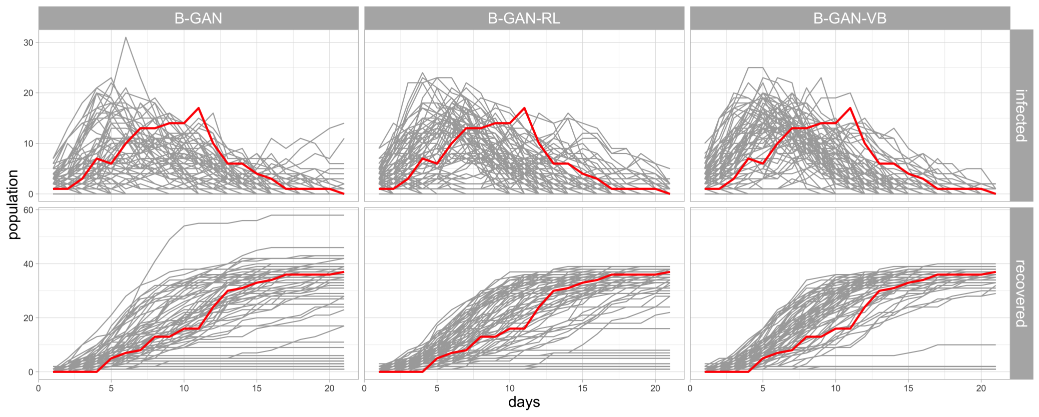


Figure 8: Simulated datasets under the posterior distributions. The red curves are the observed data. Each grey line represents one simulated time series of the infected/recovered population.

7. Discussion

This paper proposes strategies for Bayesian simulation using generative networks. We have formalized several schemes for implicit posterior simulation using GAN conditional density regression estimators as well as implicit variational Bayes. The common denominator behind our techniques is (joint) contrastive adversarial learning (Tran et al., 2017; Huszár, 2017). We have provided firm theoretical support in the form of bounds for the typical total variation distance between the posterior and its approximation. We have highlighted the potential of our adversarial samplers on several simulated examples with very encouraging findings. We hope that our paper will embolden practitioners to implement neural network posterior samplers in difficult situations when likelihood (and prior) are implicit.

One advantage of using conditional GANs over other conditional density estimators, such as the autoregressive flow in Papamakarios et al. (2019), lies in their expressive flexibility and ability to incorporate statistical structures. While the two methods share a similar purpose, they are fundamentally different. SNL provides a closed-form representation for the approximated density function, while the generator $g(\cdot, X^{(n)})$ in WGAN is approximating a pushforward mapping from π_Z to the conditional distribution $\pi(\theta \mid X^{(n)})$ and no closed-form of the density function is available. Due to the difference, SNL is more likely to suffer from the curse of dimensionality due to the invertibility constraint (Papamakarios et al., 2021). In addition, the invertibility constraint also imposes restrictions on choices of neural network architecture. In constrast, the generator in our WGAN framework can be designed to accommodate various structures that exploit the statistical properties of the data. For instance, in the case of i.i.d. data as discussed in Section 2.5 or partially exchanegable data as in Luciano et al. (2025), the generator can be designed in a parameter-efficient manner that exploits the exchangeability of the data, thereby avoiding exploding network complexity as the number of observations increases. Moreover, when the posterior is approximately Gaussian, the generator can be constructed as two neural networks that separately parameterize the posterior mean and variance. More recently, (Baptista et al., 2024) consider the monotonicity constraint for the generator, which could offer extra inferential benefits with its connection to the optimal transport map.

Another interesting direction is to extend our approach to the case of mis-specified models, where the true data is not consistent with the model. Understanding the behavior of the posterior in such cases is crucial for practical applications. We leave the exploration of this topic for future work. Some possible directions include incorporating the idea of Bayesian predictive check (Guttman, 1967; Rubin, 1984) to diagnose the model mis-specification and estimating the conditional distribution given robust summary statistics.

Acknowledgments

The authors gratefully acknowledge support from the James S. Kemper Research Fund at the Booth School of Business and the National Science Foundation (DMS: 1944740, DMS: 2515542). This work used the DeltaAI system at the National Center for Supercomputing Applications through allocation MTH250021 from the Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support (ACCESS) program, which is supported by National Science Foundation grants #2138259, #2138286, #2138307, #2137603, and #2138296. We also thank the editor and the anonymous reviewers for their constructive comments and suggestions, which have significantly improved the quality of this paper.

References

Ziwen An, David J Nott, and Christopher Drovandi. Robust Bayesian synthetic likelihood via a semi-parametric approach. *Statistics and Computing*, 30(3):543–557, 2020.

Martin Anthony and Peter L Bartlett. *Neural network learning: Theoretical foundations*, volume 9. Cambridge University Press, 1999.

Martin Arjovsky and Léon Bottou. Towards principled methods for training generative adversarial networks. In *International Conference on Learning Representations*, 2017.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International Conference on Machine Learning*, pages 214–223. PMLR, 2017.

Susan Athey, Guido W Imbens, Jonas Metzger, and Evan Munro. Using Wasserstein generative adversarial networks for the design of Monte Carlo simulations. *Journal of Econometrics*, 2021.

Ricardo Baptista, Bamdad Hosseini, Nikola B Kovachki, and Youssef M Marzouk. Conditional sampling with monotone gans: From generative models to likelihood-free inference. *SIAM/ASA Journal on Uncertainty Quantification*, 12(3):868–900, 2024.

Peter L Bartlett, Dylan J Foster, and Matus J Telgarsky. Spectrally-normalized margin bounds for neural networks. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Mark A Beaumont. Estimation of population growth or decline in genetically monitored populations. *Genetics*, 164(3):1139–1160, 2003.

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

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

Espen Bernton, Pierre E Jacob, Mathieu Gerber, and Christian P Robert. Approximate Bayesian computation with the Wasserstein distance. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 81(2):235–269, 2019.

Steffen Bickel, Michael Brückner, and Tobias Scheffer. Discriminative learning for differing training and test distributions. In *Proceedings of the 24th International Conference on Machine Learning*, pages 81–88, 2007.

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

Rainer Burkard, Mauro Dell'Amico, and Silvano Martello. *Assignment problems*. SIAM, 2009.

Xiaohong Chen and Halbert White. Improved rates and asymptotic normality for non-parametric neural network estimators. *IEEE Transactions on Information Theory*, 45 (2):682–691, 1999.

Kyle Cranmer, Juan Pavez, and Gilles Louppe. Approximating likelihood ratios with calibrated discriminative classifiers. *arXiv preprint arXiv:1506.02169*, 2015.

Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in Neural Information Processing Systems*, volume 26, 2013.

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.

Matteo Fasiolo, Simon N Wood, Florian Hartig, and Mark V Bravington. An extended empirical saddlepoint approximation for intractable likelihoods. *Electronic Journal of Statistics*, 12(1):1544–1578, 2018.

Paul Fearnhead and Dennis Prangle. Constructing ABC summary statistics: semi-automatic ABC. *Nature Precedings*, pages 1–1, 2011.

David T Frazier, Ryan Kelly, Christopher Drovandi, and David J Warne. The statistical accuracy of neural posterior and likelihood estimation. *arXiv preprint arXiv:2411.12068*, 2024.

Jon Gauthier. Conditional generative adversarial nets for convolutional face generation. *Class project for Stanford CS231N: convolutional neural networks for visual recognition, Winter semester*, 2014(5):2, 2014.

Subhashis Ghosal and Aad van der Vaart. Convergence rates of posterior distributions for noniid observations. *The Annals of Statistics*, 35(1):192–223, 2007.

Daniel T Gillespie. Exact stochastic simulation of coupled chemical reactions. *The Journal of Physical Chemistry*, 81(25):2340–2361, 1977.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, volume 27, 2014.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning.* MIT press, 2016.

Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1):723–773, 2012.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of Wasserstein gans. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Michael U Gutmann and Jukka Corander. Bayesian optimization for likelihood-free inference of simulator-based statistical models. *Journal of Machine Learning Research*, 2016.

Michael U Gutmann and Aapo Hyvärinen. Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *Journal of Machine Learning Research*, 13(2), 2012.

Michael U Gutmann, Ritabrata Dutta, Samuel Kaski, and Jukka Corander. Likelihood-free inference via classification. *Statistics and Computing*, 28(2):411–425, 2018.

Irwin Guttman. The use of the concept of a future observation in goodness-of-fit problems. *Journal of the Royal Statistical Society: Series B (Methodological)*, 29(1):83–100, 1967.

Philipp Hennig and Christian J Schuler. Entropy search for information-efficient global optimization. *Journal of Machine Learning Research*, 13(6), 2012.

Ferenc Huszár. Variational inference using implicit distributions. *arXiv preprint arXiv:1702.08235*, 2017.

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.

Marko Jarvenpaa, Aki Vehtari, and Pekka Marttinen. Batch simulations and uncertainty quantification in Gaussian process surrogate approximate Bayesian computation. In *Proceedings of the 36th Conference on Uncertainty in Artificial Intelligence*, volume 124, pages 779–788. PMLR, 2020.

Tetsuya Kaji and Veronika Ročková. Metropolis-Hastings via classification. *Journal of the American Statistical Association*, pages 1–33, 2022.

Olav Kallenberg. *Foundations of Modern Probability*, volume 2. Springer, 2002.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Diederik P Kingma and Max Welling. Auto-encoding variational Bayes, 2013.

Tengyuan Liang. How well generative adversarial networks learn distributions. *Journal of Machine Learning Research*, 22:1–41, 2021.

Antoine Luciano, Charly Andral, Christian P Robert, and Robin J Ryder. Permutations accelerate approximate bayesian computation. *arXiv preprint arXiv:2507.06037*, 2025.

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. In *Advances in Neural Information Processing Systems*, volume 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.

Paul Marjoram, John Molitor, Vincent Plagnol, and Simon Tavaré. Markov chain Monte Carlo without likelihoods. *Proceedings of the National Academy of Sciences*, 100(26): 15324–15328, 2003.

Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. Adversarial variational Bayes: Unifying variational autoencoders and generative adversarial networks. In *International Conference on Machine Learning*, pages 2391–2400. PMLR, 2017.

Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

George Papamakarios and Iain Murray. Fast ε-free inference of simulation models with Bayesian conditional density estimation. In *Advances in Neural Information Processing Systems*, pages 1028–1036, 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.

George Papamakarios, Eric T Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 22(57):1–64, 2021.

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

Donald B Rubin. Bayesianly justifiable and relevant frequency calculations for the applied statistician. *The Annals of Statistics*, pages 1151–1172, 1984.

Yunus Saatci and Andrew G Wilson. Bayesian GAN. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Johannes Schmidt-Hieber. Nonparametric regression using deep neural networks with ReLU activation function. *The Annals of Statistics*, 48(4):1875–1897, 2020.

Bodhisattva Sen. A gentle introduction to empirical process theory and applications. *Lecture Notes, Columbia University*, 2018.

Scott A Sisson, Yanan Fan, and Mark M Tanaka. Sequential Monte Carlo without likelihoods. *Proceedings of the National Academy of Sciences*, 104(6):1760–1765, 2007.

Scott A Sisson, Yanan Fan, and Mark A Beaumont. Overview of ABC. In *Handbook of Approximate Bayesian Computation*, pages 3–54. Chapman and Hall/CRC, 2018.

George R Terrell and David W Scott. Variable kernel density estimation. *The Annals of Statistics*, pages 1236–1265, 1992.

Michalis K Titsias and Francisco Ruiz. Unbiased implicit variational inference. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 167–176. PMLR, 2019.

Tina Toni, David Welch, Natalja Strelkowa, Andreas Ipsen, and Michael PH Stumpf. Approximate Bayesian computation scheme for parameter inference and model selection in dynamical systems. *Journal of the Royal Society Interface*, 6(31):187–202, 2009.

Dustin Tran, Rajesh Ranganath, and David Blei. Hierarchical implicit models and likelihood-free variational inference. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Ramon Van Handel. Probability in high dimension. Technical report, Princeton Univ NJ, 2014.

Yuexi Wang, Tetsuya Kaji, and Veronika Ročková. Approximate Bayesian computation via classification. *The Journal of Machine Learning Research*, 23(1):15837–15885, 2022.

Darren J Wilkinson. *Stochastic modelling for systems biology.* Chapman and Hall/CRC, 2018.

Xingyu Zhou, Yuling Jiao, Jin Liu, and Jian Huang. A deep generative approach to conditional sampling. *Journal of the American Statistical Association*, pages 1–12, 2022.

Appendix A. Theory for Jensen-Shannon Conditional GANs

Definition 8 *(Discriminator) We define a deterministic* discriminative model *as a mapping $d : (\mathcal{X} \times \Theta) \to (0,1)$ which predicts whether the data pair (X, θ) came from $\pi(X,\theta)$ (label 1) or from $\pi_g(X,\theta)$ (label 0).*

Lemma 9 *(Zhou et al., 2022, Lemma 2.1) Let (X,θ) be a random pair taking values in $\mathcal{X} \times \Theta$ with a joint distribution $\pi(X,\theta)$. Then, for any given $d_z \geq 1$, there exists a random vector $Z \sim \pi_Z = N(0, I_{d_z})$ and a Borel-measurable function $g^* : \mathbb{R}^d \times \mathcal{X} \to \Theta$ such that Z is independent of X and $(X,\theta) = (X, g^*(Z,X))$ almost surely.*

Lemma 10 *Consider a minimax game $(g^*, d^*) = \arg\min_{g\in\mathcal{G}} \max_{d\in\mathcal{D}} D(g,d)$ prescribed by*

$$D(g,d) = E_{(X,\theta)\sim\pi(X,\theta)} \log d(X,\theta) + E_{X\sim\pi(X), Z\sim\pi_Z} \log[1 - d(X, g(Z,X)]. \tag{22}$$

Assume that $\mathcal{G}$ and $\mathcal{D}$ are universal approximators capable of representing any function $g : (\mathcal{Z} \times \mathcal{X}) \to \Theta$ and $d : (\mathcal{X} \times \Theta) \to (0,1)$, respectively. Then, uniformly on $\mathcal{X}$ and Θ, the solution (g^*, d^*) satisfies

$$\pi_{g^*}(\theta \mid X) = \frac{\pi(X,\theta)}{\pi(X)} = \pi(\theta \mid X) \quad \text{and} \quad d^*_g(X,\theta) = \frac{\pi(X,\theta)}{\pi(X,\theta) + \pi_g(X,\theta)} \quad \text{for any } g \in \mathcal{G}.$$

Proof The expression for $d^*_g(X,\theta)$ is an immediate consequence of Proposition 1 in Goodfellow et al. (2014). Plugging-in this expression into (22), we find that

$$g^* = \arg\min_{g\in\mathcal{G}} \left(E_{(X,\theta)\sim\pi(X,\theta)} \log d^*_g(X,\theta) + E_{X\sim\pi(X), z\sim\pi_Z} \log[1 - d^*_g(X, g(Z,X))] \right),$$

According to Theorem 1 of Goodfellow et al. (2014), the minimum is achieved if and only if $\pi_{g^*}(X,\theta) = \pi(X,\theta) = \pi(\theta \mid X)\pi(X)$. The fact that $\pi(X,\theta)$ and $\pi_{g^*}(X,\theta)$ have the same marginal $\pi(X)$ implies the expression for $\pi_{g^*}(\theta \mid X)$. ■

Appendix B. Proofs from Section 4

B.1 Proof of Theorem 1

Proof We continue denoting $X^{(n)}$ simply by X. Recall the definition of the KL neighborhood

$$B_n(\theta_0;\epsilon) = \{\theta \in \Theta : \mathrm{KL}(P^{(n)}_{\theta_0} \| P^{(n)}_\theta) \leq n\epsilon^2, V_{2,0}(P^{(n)}_{\theta_0}, P^{(n)}_\theta) \leq n\epsilon^2\}, \tag{23}$$

where $\mathrm{KL}(P^{(n)}_{\theta_0} \| P^{(n)}_\theta) = P^{(n)}_{\theta_0} \log[p^{(n)}_{\theta_0}/p^{(n)}_\theta]$ and

$$V_{2,0}(P^{(n)}_{\theta_0}, P^{(n)}_\theta) = P^{(n)}_{\theta_0} \left| \log[p^{(n)}_{\theta_0}/p^{(n)}_\theta] - \mathrm{KL}(P^{(n)}_{\theta_0} \| P^{(n)}_\theta) \right|^2. \tag{24}$$

We define an event, for some fixed $C > 0$ and $\epsilon > 0$,

$$\mathcal{A}_n(\epsilon) = \left\{ X : \int_{B_n(\theta_0;\epsilon)} \frac{p^{(n)}_\theta(X)}{p^{(n)}_{\theta_0}(X)} \pi(\theta) \mathrm{d}\theta > \mathrm{e}^{-(1+C)n\varepsilon^2} \Pi[B_n(\theta_0;\epsilon)] \right\}.$$

We denote with $\mathbb{E}$ the expectation with respect to $\{(\theta_j, X_j)\}_{j=1}^T$ from the ABC reference table sampled from the joint $\pi(\theta, X)$. For simplicity of notation, we use $\mathbb{E}_{\widehat{\boldsymbol{\beta}}_T}$ interchangeably with $\mathbb{E}$, since it is equivalently accounting for the randomness in $\widehat{\boldsymbol{\beta}}_T$. Using the fact that the total variation distance is bounded by 2, we have

$$P_{\theta_0}^{(n)}\mathbb{E}_{\hat{\beta}_T} d_{TV}^2\left(\nu(X_0), \mu_{\widehat{\boldsymbol{\beta}}_T}(X_0)\right) = \int_{\mathcal{A}_n(\epsilon)} p_{\theta_0}^{(n)}(X_0)\mathbb{E}_{\hat{\beta}_T} d_{TV}^2\left(\nu(X_0), \mu_{\widehat{\boldsymbol{\beta}}_T}(X_0)\right)\mathrm{d}X_0 \tag{25}$$

$$+ 4\,\mathbb{P}_{\theta_0}^{(n)}[\mathcal{A}_n^c(\epsilon)]. \tag{26}$$

According to Lemma 10 in Ghosal and van der Vaart (2007), we have $\mathbb{P}_{\theta_0}^{(n)}[\mathcal{A}_n^c(\epsilon)] \leq \frac{1}{C^2 n\epsilon^2}$. Denoting with $\pi(X) = \int_\theta p_\theta^{(n)}(X)\pi(\theta)\mathrm{d}\theta$ the marginal likelihood, we can rewrite the term in (25) as

$$\int_{\mathcal{X}} \mathbb{I}_{\mathcal{A}_n(\epsilon)}(X)\pi(X)r(X)\mathbb{E}_{\hat{\beta}_T} d_{TV}^2\left(\nu(X), \mu_{\widehat{\boldsymbol{\beta}}_T}(X)\right)\mathrm{d}X$$

where

$$\frac{1}{r(X)} \equiv \int_\Theta \frac{p_\theta^{(n)}(X)}{p_{\theta_0}^{(n)}(X)}\pi(\theta)\mathrm{d}\theta \geq \int_{B_n(\theta_0;\epsilon)} \frac{p_\theta^{(n)}(X)}{p_{\theta_0}^{(n)}(X)}\pi(\theta)\mathrm{d}\theta.$$

On the event $\mathcal{A}_n(\epsilon)$, we can thus write

$$r(X) < \frac{\mathrm{e}^{(1+C)n\epsilon^2}}{\Pi[B_n(\theta_0;\epsilon)]}.$$

Under the assumption (20), the term in (25) can be upper bounded by

$$\mathrm{e}^{(1+C+C_2)n\epsilon^2}\int_{\mathcal{X}} \pi(X)\mathbb{E}_{\hat{\beta}_T} d_{TV}^2\left(\nu(X), \mu_{\widehat{\boldsymbol{\beta}}_T}(X)\right)\mathrm{d}X \leq \frac{\mathrm{e}^{(1+C+C_2)n\epsilon^2}}{4}\mathbb{E}_{\hat{\beta}_T}[\mathrm{KL}(\nu\|\mu_{\widehat{\boldsymbol{\beta}}_T}) + \mathrm{KL}(\mu_{\widehat{\boldsymbol{\beta}}_T}\|\nu)]. \tag{27}$$

The inequality above stems from the Pinsker's inequality (Van Handel, 2014, Theorem 4.8) and the fact that the joint measures ν and $\mu_{\widehat{\boldsymbol{\beta}}_T}$ have the same marginal distribution $\pi(X)$. In particular, using Fubini's theorem, we can write

$$\begin{aligned}
&\int_{\mathcal{X}} \pi(X)\mathbb{E}_{\hat{\beta}_T} d_{TV}^2\left(\nu(X), \mu_{\widehat{\boldsymbol{\beta}}_T}(X)\right)\mathrm{d}X \\
&\leq \frac{1}{4}\int_{\mathcal{X}} \pi(X)\mathbb{E}_{\hat{\beta}_T}[\mathrm{KL}(\nu(X)\|\mu_{\widehat{\boldsymbol{\beta}}_T}(X)) + \mathrm{KL}(\mu_{\widehat{\boldsymbol{\beta}}_T}(X)\|\nu(X))]\mathrm{d}X \\
&= \frac{1}{4}\mathbb{E}_{\hat{\beta}_T}\int_{\mathcal{X}} \pi(X)\int_\Theta \log\frac{\pi(\theta\,|\,X)}{\pi_{\widehat{\boldsymbol{\beta}}_T}(\theta\,|\,X)}\left[\pi(\theta\,|\,X) - \pi_{\widehat{\boldsymbol{\beta}}_T}(\theta\,|\,X)\right]\mathrm{d}\theta\mathrm{d}X \\
&= \frac{1}{4}\mathbb{E}_{\hat{\beta}_T}\left[\mathrm{KL}(\nu\|\mu_{\widehat{\boldsymbol{\beta}}_T}) + \mathrm{KL}(\mu_{\widehat{\boldsymbol{\beta}}_T}\|\nu)\right] \equiv \frac{1}{4}\mathbb{E}_{\hat{\beta}_T} d_{\mathrm{KL}}^S(\nu, \mu_{\widehat{\boldsymbol{\beta}}_T}).
\end{aligned}$$

The above inequality is essential for understanding how the average squared total variation distance between the posterior and its approximation (with the average taken with respect to the observed data generating process) can be related to the 'symmetrized' KL divergence $d_{\mathrm{KL}}^S(\nu, \mu_{\widehat{\boldsymbol{\beta}}_T})$ between the *joint* distribution and its approximation. We now continue to bound the symmetrized KL divergence. For simplicity, we denote with $\widehat{\boldsymbol{\beta}}$ the estimator $\widehat{\boldsymbol{\beta}}_T$ in (17). We have the following decomposition, for any $\boldsymbol{\omega}$ such that $f_{\boldsymbol{\omega}} \in \mathcal{F}$,

$$\begin{aligned} d_{\mathrm{KL}}^S(\nu, \mu_{\widehat{\boldsymbol{\beta}}}) &= \int_{\mathcal{X}} \pi(X) \int_{\Theta} f_\omega(\theta, X)[\pi(\theta \mid X) - \pi_{\widehat{\boldsymbol{\beta}}}(\theta \mid X)]\mathrm{d}\theta \mathrm{d}X \\ &\quad + \int_{\mathcal{X}} \pi(X) \int_{\Theta} \left[\log \frac{\pi(\theta \mid X)}{\pi_{\widehat{\boldsymbol{\beta}}}(\theta \mid X)} - f_\omega(\theta, X)\right] [\pi(\theta \mid X) - \pi_{\widehat{\boldsymbol{\beta}}}(\theta \mid X)]\mathrm{d}\theta \mathrm{d}X \\ &\leq d_{\mathcal{F}}\left(\nu, \mu_{\widehat{\boldsymbol{\beta}}}\right) + 2\left\|\log \frac{\pi(\theta \mid X)}{\pi_{\widehat{\boldsymbol{\beta}}}(\theta \mid X)} - f_\omega(\theta, X)\right\|_\infty, \end{aligned}$$

where we have used the inequality $\int fg \leq \|f\|_\infty \|g\|_1$ and the fact that $\pi(\theta \mid X)$ and $\pi_{\widehat{\boldsymbol{\beta}}}(\theta \mid X)$ are both non-negative and integrate to one. Then, choosing $f_{\boldsymbol{\omega}} \in \mathcal{F}$ that minimizes the second term we obtain

$$d_{\mathrm{KL}}^S(\nu, \mu_{\widehat{\boldsymbol{\beta}}}) \leq 2\,\mathcal{A}_1(\mathcal{F}, \mathcal{G}, \widehat{\boldsymbol{\beta}}) + d_{\mathcal{F}}\left(\nu, \mu_{\widehat{\boldsymbol{\beta}}}\right),$$

where $\mathcal{A}_1(\mathcal{F}, \mathcal{G}, \widehat{\boldsymbol{\beta}})$ is defined as

$$\mathcal{A}_1(\mathcal{F}, \mathcal{G}, \widehat{\boldsymbol{\beta}}) \equiv \inf_{\boldsymbol{\omega}: f_{\boldsymbol{\omega}} \in \mathcal{F}} \left\|\log \frac{\pi(\theta \mid X)}{\pi_{\widehat{\boldsymbol{\beta}}_T}(\theta \mid X)} - f_\omega(X, \theta)\right\|_\infty$$

and $\mathcal{A}_1(\mathcal{F}, \mathcal{G}) = \mathbb{E}_{\widehat{\boldsymbol{\beta}}} \mathcal{A}_1(\mathcal{F}, \mathcal{G}, \widehat{\boldsymbol{\beta}})$ was defined in (18).

We now apply a mild modification of the oracle inequality in (Liang, 2021, Lemma 12). As long as $\mathcal{F}$ and $\mathcal{H}$ are symmetric[2], then for any $\boldsymbol{\beta}$ such that $g_{\boldsymbol{\beta}} \in \mathcal{G}$ we have

$$d_{\mathcal{F}}(\nu, \mu_{\widehat{\boldsymbol{\beta}}}) \leq d_{\mathcal{F}}(\mu_{\boldsymbol{\beta}}, \nu) + 2d_{\mathcal{F}}(\bar{\nu}_T, \nu) + d_{\mathcal{F}}(\bar{\mu}_T^{\boldsymbol{\beta}}, \mu_{\boldsymbol{\beta}}) + d_{\mathcal{H}}(\bar{\pi}_T, \pi), \tag{28}$$

where $\bar{\nu}_T$ and $\bar{\mu}_T^{\boldsymbol{\beta}}$ are the empirical counterparts of $\nu, \mu_{\boldsymbol{\beta}}$ based on T iid samples (ABC reference table $\{(\theta_j, X_j)\}_{j=1}^T$ for ν and $\{(g_{\boldsymbol{\beta}}(Z_j, X_j), X_j)\}_{j=1}^T$ with $Z_j \overset{\text{iid}}{\sim} \pi_Z$ for $\mu_{\boldsymbol{\beta}}$). In addition $\bar{\pi}_T$ is the empirical version for the distribution π_Z for Z and $\mathcal{H} = \{h_{\boldsymbol{\omega},\boldsymbol{\beta}} : h_{\boldsymbol{\omega},\boldsymbol{\beta}}(Z, X) = f_{\boldsymbol{\omega}}(X, g_{\boldsymbol{\beta}}(Z, X))\}$. This oracle inequality can be proved analogously as (Liang, 2021, Lemma 12) the only difference being that due to the conditional structure of our GANs the function class $\mathcal{H}$ is not entirely a composition of networks $f_{\boldsymbol{\omega}}$ and $g_{\boldsymbol{\beta}}$ but has a certain nested structure. Similarly as in (Liang, 2021) (proof of Theorem 13), we can write for any $\boldsymbol{\beta}$ such that $g_{\boldsymbol{\beta}} \in \mathcal{G}$

$$\begin{aligned} d_{\mathcal{F}}(\mu_{\boldsymbol{\beta}}, \nu) &\leq B \times d_{TV}(\mu_{\boldsymbol{\beta}}, \nu) \leq B\sqrt{\frac{1}{4} d_{\mathrm{KL}}^S(\mu_{\boldsymbol{\beta}}, \nu)} \\ &\leq \frac{B}{\sqrt{2}} \left[\int_{\mathcal{X}} \left\|\log \frac{\pi_{\boldsymbol{\beta}}(\theta \mid X)}{\pi(\theta \mid X)}\right\|_\infty \pi(X)\mathrm{d}X\right]^{1/2}. \end{aligned}$$

Choosing $\boldsymbol{\beta}$ that minimizes the expectation on the right side, we obtain $d_{\mathcal{F}}(\mu_{\boldsymbol{\beta}}, \nu) \leq \frac{B}{\sqrt{2}} \mathcal{A}_2(\mathcal{G})$, where the term $A_2(\mathcal{G})$ was defined in (19). Denote with

$$R_T(\mathcal{F}) = E_{\boldsymbol{\varepsilon}} \sup_{f \in \mathcal{F}} \frac{1}{T} \sum_{j=1}^T \varepsilon_j f(X_j, \theta_j)$$

2. i.e. if $f \in \mathcal{F}$ then also $-f \in \mathcal{F}$

the Rademacher complexity with $\boldsymbol{\varepsilon} = (\varepsilon_1, \dots, \varepsilon_T)'$ iid Rademacher[3] variables. For the second term in (28), the symmetrization property (see e.g. Lemma 26 in Liang (2021) or Theorem 3.17 in Sen (2018)) yields for $T \geq Pdim(\mathcal{F})$

$$\mathbb{E}\, d_{\mathcal{F}}(\bar{\nu}_T, \nu) \leq 2\,\mathbb{E}\, R_T(\mathcal{F}) \leq \widetilde{C} \times B\sqrt{\frac{Pdim(\mathcal{F}) \log T}{T}}$$

for some $\widetilde{C} > 0$, where $Pdim(\mathcal{F})$ is the pseudo-dimension of the function class $\mathcal{F}$ (Definition 2 in (Bartlett et al., 2017)). The second inequality follows, for example, from Lemma 29 in (Liang, 2021). The bounds on $\mathbb{E}\, d_{\mathcal{H}}(\bar{\pi}_T, \pi)$, and $\mathbb{E}\, d_{\mathcal{F}}(\bar{\mu}_T^{\beta}, \mu_\beta)$ in (28) are analogous. Putting the pieces together from the oracle inequality in (28) we can upper-bound $P_{\theta_0}^{(n)} \mathbb{E} d_{TV}^2 \left(\nu(X_0), \mu_{\widehat{\boldsymbol{\beta}}_T}(X_0)\right)$ with

$$\frac{1}{C^2 n\varepsilon^2} + \frac{\mathrm{e}^{(1+C+C_2)n\varepsilon^2}}{4}\left[2\mathcal{A}_1(\mathcal{F}, \mathcal{G}) + \frac{B}{\sqrt{2}}\mathcal{A}_2(\mathcal{G}) + 4\widetilde{C}\, B\sqrt{\frac{\log T}{T}}\,(Pdim(\mathcal{F}) \vee Pdim(\mathcal{H}))^{1/2}\right]$$

which yields the desired statement. ∎

B.2 Proof of Corollary 4

Proof We continue to use the shorthand notation X for $X^{(n)}$ and $\widehat{\boldsymbol{\beta}}$ for $\widehat{\boldsymbol{\beta}}_T$. Denote with $\widetilde{\pi}(\theta)$ the proposal distribution. Then, the posterior $\widetilde{\pi}(\theta \mid X)$ under $\widetilde{\pi}(\theta)$ satisfies

$$\pi(\theta \mid X) = \widetilde{\pi}(\theta \mid X) \times R(X) \times r(\theta), \quad \text{where} \quad R(X) = \frac{\widetilde{\pi}(X)}{\pi(X)} \text{ and } r(\theta) = \frac{\pi(\theta)}{\widetilde{\pi}(\theta)}. \tag{29}$$

Our reconstruction in Algorithm 2 works by first approximating the joint distribution $\widetilde{\pi}(\theta, X)$ and then reweighting by the prior ratio, namely

$$\pi_{\widehat{\boldsymbol{\beta}}}(\theta \mid X) = \widetilde{\pi}_{\widehat{\boldsymbol{\beta}}}(\theta \mid X) \times R(X) \times r(\theta), \tag{30}$$

where $\widehat{\boldsymbol{\beta}}$ has been learned by B-GAN (Algorithm 1) by matching the joint $\widetilde{\pi}(\theta, X) = \widetilde{\pi}(\theta \mid X)\widetilde{\pi}(X)$ under the prior $\widetilde{\pi}(\theta)$. We denote the joint measure with this density by $\widetilde{\nu}$. Denote with $\mu_{\widehat{\boldsymbol{\beta}}_T}(X)$ the approximating conditional measure with a density (30). We can apply the same steps as in the proof of Theorem 1 until the step in (27). Similarly, we denote with $\widetilde{\mathbb{E}}$ the expectation with respect to $\{(\theta_j, X_j)\}_{j=1}^T$ from the ABC reference table sampled from the joint $\widetilde{\pi}(\theta, X)$, and we use $\widetilde{\mathbb{E}}_{\widehat{\boldsymbol{\beta}}_T}$ interchangeably with $\widetilde{\mathbb{E}}$. The next steps will have minor modifications. Notice that

$$\log \frac{\pi(\theta \mid X)}{\pi_{\widehat{\boldsymbol{\beta}}}(\theta \mid X)} = \log \frac{\widetilde{\pi}(\theta \mid X)}{\widetilde{\pi}_{\widehat{\boldsymbol{\beta}}}(\theta \mid X)}$$

3. taking values $\{-1, +1\}$ with probability 1/2.

and thereby

$$\begin{aligned}
&\int_{\mathcal{X}} \pi(X)\widetilde{\mathbb{E}}_{\widehat{\boldsymbol{\beta}}} d_{TV}^2 \left(\nu(X), \mu_{\widehat{\boldsymbol{\beta}}_T}(X)\right) \mathrm{d}X \\
&\leq \frac{1}{4}\int_{\mathcal{X}} \pi(X)\widetilde{\mathbb{E}}_{\widehat{\boldsymbol{\beta}}}\Big[\mathrm{KL}(\nu(X)\|\mu_{\widehat{\boldsymbol{\beta}}}(X)) + \mathrm{KL}(\mu_{\widehat{\boldsymbol{\beta}}}(X)\|\nu(X))\Big]\mathrm{d}X \\
&= \frac{1}{4}\widetilde{\mathbb{E}}_{\widehat{\boldsymbol{\beta}}}\int_{\mathcal{X}} \pi(X)\int_{\Theta} \log\frac{\pi(\theta\,|\,X)}{\pi_{\widehat{\boldsymbol{\beta}}}(\theta\,|\,X)}\left[\pi(\theta\,|\,X) - \pi_{\widehat{\boldsymbol{\beta}}}(\theta\,|\,X)\right]\mathrm{d}\theta\mathrm{d}X \\
&= \frac{1}{4}\widetilde{\mathbb{E}}_{\widehat{\boldsymbol{\beta}}}\int_{\mathcal{X}} \widetilde{\pi}(X)\int_{\Theta} r(\theta)\log\frac{\pi(\theta\,|\,X)}{\pi_{\widehat{\boldsymbol{\beta}}}(\theta\,|\,X)}\left[\widetilde{\pi}(\theta\,|\,X) - \widetilde{\pi}_{\widehat{\boldsymbol{\beta}}}(\theta\,|\,X)\right]\mathrm{d}\theta\mathrm{d}X \\
&\equiv \frac{1}{4}\widetilde{\mathbb{E}}_{\widehat{\boldsymbol{\beta}}} d_{\mathrm{KL}}^S(\widetilde{\nu}, \widetilde{\mu}_{\widehat{\boldsymbol{\beta}}}).
\end{aligned}$$

Using similar arguments as in the proof of Theorem 1, we have the following decomposition, for any $\boldsymbol{\omega}$ such that $f_{\boldsymbol{\omega}} \in \mathcal{F}$,

$$\begin{aligned}
d_{\mathrm{KL}}^S(\widetilde{\nu}, \widetilde{\mu}_{\widehat{\boldsymbol{\beta}}}) &= \int_{\mathcal{X}} \widetilde{\pi}(X)\int_{\Theta} f_\omega(\theta, X)[\widetilde{\pi}(\theta\,|\,X) - \widetilde{\pi}_{\widehat{\boldsymbol{\beta}}}(\theta\,|\,X)]\mathrm{d}\theta\mathrm{d}X \\
&\quad + \int_{\mathcal{X}} \widetilde{\pi}(X)\int_{\Theta}\left[r(\theta)\log\frac{\pi(\theta\,|\,X)}{\pi_{\widehat{\boldsymbol{\beta}}}(\theta\,|\,X)} - f_\omega(\theta, X)\right][\widetilde{\pi}(\theta\,|\,X) - \widetilde{\pi}_{\widehat{\boldsymbol{\beta}}}(\theta\,|\,X)]\mathrm{d}\theta\mathrm{d}X \\
&\leq d_{\mathcal{F}}\left(\widetilde{\nu}, \widetilde{\mu}_{\widehat{\boldsymbol{\beta}}}\right) + 2\left\|r(\theta)\log\frac{\pi(\theta\,|\,X)}{\pi_{\widehat{\boldsymbol{\beta}}}(\theta\,|\,X)} - f_\omega(\theta, X)\right\|_\infty.
\end{aligned}$$

The rest of the proof is analogous. The only difference is that $\widehat{\boldsymbol{\beta}}$ now minimizes the empirical version of $d_{\mathcal{F}}\left(\widetilde{\nu}, \widetilde{\mu}_{\widehat{\boldsymbol{\beta}}}\right)$ under the proposal distribution $\widetilde{\pi}(\theta)$.

B.3 Motivation for the Sequential Refinement

Remark 11 (2step Motivation) *For the proposal distribution $\widetilde{\pi}(\theta)$, using similar arguments as in the proof of Theorem 1, the TV distance of the posterior at X_0 (not averaged over $P_{\theta_0}^{(n)}$) can be upper-bounded by*

$$4\, d_{TV}^2\left(\nu(X_0), \mu_{\widehat{\boldsymbol{\beta}}}(X_0)\right) \leq 2\,\mathcal{A}_1(\mathcal{F}, \mathcal{G}, X_0) + \frac{B}{\sqrt{2}}\mathcal{A}_2(\mathcal{G}) + 4\widetilde{C}\, B\sqrt{\frac{\log T \times Pmax}{T}} + A_3(\widetilde{\pi})$$

where $\mathcal{A}_1(\mathcal{F}, \mathcal{G}, X_0) \equiv \sup_{\boldsymbol{\beta}: g_{\boldsymbol{\beta}} \in \mathcal{G}} \inf_{\boldsymbol{\omega}: f_{\boldsymbol{\omega}} \in \mathcal{F}} \left\|\log\frac{\pi(\theta\,|\,X_0)}{\pi_{\boldsymbol{\beta}}(\theta\,|\,X_0)} - \frac{f_\omega(X_0,\theta)}{r(\theta)}\right\|$ *is the discriminability* evaluated at X_0 *(as opposed to* (18)*) and where*

$$A_3(\widetilde{\pi}) = 2\int_{\mathcal{X}} \widetilde{\pi}(X)\left[\|f_{\boldsymbol{\omega}}(X_0, \theta) - f_{\boldsymbol{\omega}}(X, \theta)\|_\infty + B\|g_{\widehat{\boldsymbol{\beta}}}(\theta)(X) - g_{\widehat{\boldsymbol{\beta}}}(\theta)(X_0)\|_1\right]\mathrm{d}X$$

and $g_{\widehat{\boldsymbol{\beta}}}(\theta)(X) \equiv \pi(\theta\,|\,X) - \pi_{\widehat{\boldsymbol{\beta}}}(\theta\,|\,X)$. This decomposition reveals how the TV distance can be related to discriminability around X_0 and an average discrepancy between the true and approximated posterior densities relative to their value at X_0 where the average is taken over the marginal $\widetilde{\pi}(X)$. These averages will be smaller since the marginal $\widetilde{\pi}(X)$ produces *datasets more similar to X_0. For example, an approximation to the the posterior predictive distribution $\widetilde{\pi}(X)=\int_{\mathcal{X}} p_\theta^{(n)}(X)\pi_{\widehat{\boldsymbol\beta}}(\theta\,|\,X_0)$ where $\widehat{\boldsymbol\beta}$ has been learned by B-GAN (Algorithm 1) is likely to yield datasets similar to X_0, thereby producing a tighter upper bound than a flat prior.*

We provide clarifications of the calculations and reasoning in Remark 11. We assume that a prior distribution $\widetilde{\pi}(\theta)$ has been used in the ABC reference table that yields the marginal $\widetilde{\pi}(X)=\int_X p_\theta^{(n)}(X)\widetilde{\pi}(\theta)\mathrm{d}\theta$. Recall the definition of the reweighted posterior reconstruction in (30) and (29). Denote with

$$g_{\widehat{\boldsymbol\beta}}(\theta)(X)\equiv\pi(\theta\,|\,X)-\pi_{\widehat{\boldsymbol\beta}}(\theta\,|\,X)=R(X)\times r(\theta)\times[\widetilde{\pi}(\theta\,|\,X)-\widetilde{\pi}_{\widehat{\boldsymbol\beta}}(\theta\,|\,X)]$$

the difference between true and approximated posteriors at X, where $\widehat{\boldsymbol\beta}$ has been trained using the proposal $\widetilde{\pi}(\theta)$ and where $R(X)=\widetilde{\pi}(X)/\pi(X)$ and $r(\theta)=\pi(\theta)/\widetilde{\pi}(\theta)$. Using similar arguments as in the proof of Theorem 1, the squared TV distance of the posterior and its approximation satisfies, for any element $f_{\boldsymbol\omega}\in\mathcal{F}=\{f:\|f\|_\infty\leq B\}$,

$$\begin{aligned}
4\,d_{TV}^2\left(\nu(X_0),\mu_{\widehat{\boldsymbol\beta}}(X_0)\right)\leq&\int_\Theta\log\frac{\pi(\theta\,|\,X_0)}{\pi_{\widehat{\boldsymbol\beta}}(\theta\,|\,X_0)}g_{\widehat{\boldsymbol\beta}}(\theta)(X_0)\mathrm{d}\theta\\
=&\int_\Theta\left[\log\frac{\pi(\theta\,|\,X_0)}{\pi_{\widehat{\boldsymbol\beta}}(\theta\,|\,X_0)}-\frac{f_\omega(X_0,\theta)}{r(\theta)}\right]g_{\widehat{\boldsymbol\beta}}(\theta)(X_0)\mathrm{d}\theta\\
&+\int_{\mathcal{X}}\pi(X)\int_\Theta\frac{f_\omega(X,\theta)}{r(\theta)}g_{\widehat{\boldsymbol\beta}}(\theta)(X_0)\mathrm{d}\theta\mathrm{d}X\\
&+\int_{\mathcal{X}}\frac{\pi(X)}{r(\theta)}\int_\Theta[f_\omega(X_0,\theta)-f_\omega(X,\theta)]g_{\widehat{\boldsymbol\beta}}(\theta)(X_0)\mathrm{d}\theta\mathrm{d}X\\
\leq&2\inf_{\boldsymbol\omega}\left\|\log\frac{\pi(\theta\,|\,X_0)}{\pi_{\widehat{\boldsymbol\beta}}(\theta\,|\,X_0)}-\frac{f_\omega(X_0,\theta)}{r(\theta)}\right\|_\infty+d_{\mathcal{F}}\left(\widetilde{\nu},\widetilde{\mu}_{\widehat{\beta}}\right)\\
&+2\int_{\mathcal{X}}\widetilde{\pi}(X)\|f_{\boldsymbol\omega}(X_0,\theta)-f_{\boldsymbol\omega}(X,\theta)\|_\infty\mathrm{d}X\\
&+2B\times\int_{\mathcal{X}}\widetilde{\pi}(X)\left\|\frac{g_{\widehat{\boldsymbol\beta}}(\theta)(X)}{R(X)r(\theta)}-\frac{g_{\widehat{\boldsymbol\beta}}(\theta)(X_0)}{R(X_0)r(\theta)}\right\|_1\mathrm{d}X.
\end{aligned}$$

The term $d_{\mathcal{F}}\left(\widetilde{\nu},\widetilde{\mu}_{\widehat{\boldsymbol\beta}}\right)$ can be bounded as in the proof of Corollary 4 by

$$\frac{B}{\sqrt{2}}\widetilde{\mathcal{A}}_2(\mathcal{G})+4\widetilde{C}\,B\sqrt{\frac{\log T\times Pmax}{T}}$$

Compared to Corollary 4, the upper bound on $d_{TV}^2\left(\nu(X_0),\mu_{\widehat{\boldsymbol\beta}}(X_0)\right)$ now involves the discriminability *evaluated at* X_0 (not averaged over the marginal $\widetilde{\pi}(X)$), i.e.

$$\mathcal{A}_1(\mathcal{F},\mathcal{G},X_0)\equiv\sup_{\boldsymbol\beta:g_{\boldsymbol\beta}\in\mathcal{G}}\inf_{\boldsymbol\omega:f_{\boldsymbol\omega}\in\mathcal{F}}\left\|\log\frac{\pi(\theta\,|\,X_0)}{\pi_{\boldsymbol\beta}(\theta\,|\,X_0)}-\frac{f_\omega(X_0,\theta)}{r(\theta)}\right\|.$$

The additional two terms in the upper bound involve integration over $\widetilde{\pi}(X)$.

■

B.4 Proof of Corollary 6

With $\varepsilon_n = O(1/\sqrt{n})$ we need to verify that for some suitable choice of $C_n \to \infty$ we have as $n \to \infty$

$$A_1(\mathcal{F}, \mathcal{G}) = o(\mathrm{e}^{-C_n}) \tag{31}$$

$$A_2(\mathcal{G}) = o(\mathrm{e}^{-C_n}) \tag{32}$$

$$\frac{\log T}{T} \times \left[Pdim(\mathcal{F}) \vee Pdim(\mathcal{H})\right] = o(\mathrm{e}^{-2C_n}) \tag{33}$$

for T that is large enough, i.e. $T \geq Pdim(\mathcal{F}) \vee Pdim(\mathcal{H})$. The term $\mathcal{A}_2(\mathcal{G})$ equals zero from our assumption of representability of $\pi(\theta, X) = \pi_{g_{\boldsymbol{\beta}_0}}(\theta, X)$ for $g_{\boldsymbol{\beta}_0} \in \mathcal{G}$, which verifies (32). We assume that $X^{(n)}$ is a stacked vector of n observed vectors of length q, not necessarily iid, and denote $d^* = d + nq$. Using leaky ReLU networks and assuming representability, for any $\boldsymbol{\beta}$ such that $g_{\boldsymbol{\beta}} \in \mathcal{G}$ the log posterior ratio $r_{\boldsymbol{\beta}}(\theta, X^{(n)}) = \log \frac{\pi(\theta \mid X^{(n)})}{\pi_{\boldsymbol{\beta}}(\theta \mid X^{(n)})}$ is continuous and, due to boundedness of the network weights, satisfies

$$0 < \underline{C} \leq r_{\boldsymbol{\beta}}(\theta, X^{(n)}) \leq \bar{C} < \infty$$

for any fixed d^*. With large enough T and setting $E = [-\log T, \log T]^{d^*}$ and $R = \log T$, Lemma 12 yields that there exists a ReLU network $f_{\boldsymbol{\omega}} \in \mathcal{F}$ with a width

$$W = 3^{d^*+3} \max\{d^* \lfloor N^{1/d^*} \rfloor, N+1\}$$

and depth $L = 12 \log T + 14 + 2d^*$ such that

$$\mathcal{A}_1(\mathcal{F}, \mathcal{G}) \leq \sup_{\boldsymbol{\beta}: g_{\boldsymbol{\beta}} \in \mathcal{G}} \inf_{\boldsymbol{\omega}: f_{\boldsymbol{\omega}} \in \mathcal{F}} \| r_{\boldsymbol{\beta}}(\theta, X^{(n)}) - f_{\boldsymbol{\omega}}(X^{(n)}, \theta) \|_{L_\infty(E)} \leq 19\sqrt{d^*}\omega_f^E(2(\log T)^{1-2/d^*} N^{-2/d^*}),$$

where ω_f^E is the modulus of continuity of $f(t)$ satisfying $\omega_f^E(t) \to 0$ as $t \to 0^+$. Choosing N such that $2^{d^*/2}(\log T)^{d^*/2-1} = o(N)$ as $T \to \infty$, the right-hand side above goes to zero for any fixed $d^* = d + nq$. For each n, we can find T large enough (depending on the modulus of continuity) such that $\mathcal{A}_1(\mathcal{F}, \mathcal{G})\mathrm{e}^{C_n}\sqrt{d^*} \leq \eta_n$ for some $\eta_n = o(1)$, yielding (31). The smallest T that satisfies this will be denoted with T_n.

In order to verify (33), Theorem 14.1 in Anthony and Bartlett (1999) and Theorem 6 in Bartlett et al. (2017) show that for piecewise linear activation functions (including ReLU and leaky ReLU) there exist constants $c, C > 0$ such that

$$c \times SL \log(S/L) \leq Pdim(\mathcal{F}) \leq C \times SL \log S,$$

where $\mathcal{F}$ is a class of discriminator networks with L layers and S parameters. Since elements in $\mathcal{H}$ can be regarded as sparse larger neural networks with $L + L^*$ layers, $S + S^*$ parameters and piece-wise linear activations, we have

$$Pdim(\mathcal{F}) \vee Pdim(\mathcal{H}) \leq C \times (S + S^*)(L + L^*) \log(S + S^*).$$

Our assumption $T \geq Pdim(\mathcal{F}) \vee Pdim(\mathcal{H})$ will thus be satisfied, for instance, when

$$T > C \times (S + S^*)(L + L^*) \log(S + S^*). \tag{34}$$

Choosing $N = \lfloor 2^{d^*/2}(\log T)^{d^*/2}\rfloor$, which satisfies the requirement $2^{d^*/2}(\log T)^{d^*/2-1} = o(N)$ as $T \to \infty$, yields

$$W = 3^{d^*+3}\max\{d^*\lfloor N^{1/d^*}\rfloor, N+1\} = 3^{d^*+3}\lfloor 2^{d^*/2}(\log T)^{d^*/2} + 1\rfloor \tag{35}$$

for a sufficiently large n (and thereby d^*). Recall that in the feed-forward neural networks, the total number of parameters $S = \sum_{l=0}^{L-1}[w_l(w_l+1)]$ satisfies $S \leq LW(W+1)$. For any fixed d^* (and thereby n), assuming $L = 12\log T + 14 + 2d^*$ as before and W as in (35), we define $T(d^*)$ as the smallest T that satisfies

$$C \times [LW(W+1)+S^*](L+L^*)\log[LW(W+1)+S^*] \leq \frac{T}{\log T} \times \mathrm{e}^{-2C_n} \times \eta_n$$

for some $\eta_n = o(1)$. Any $T > T(d^*)$ satisfies $T > Pdim(\mathcal{F}) \vee Pdim(\mathcal{H})$ and $\mathrm{e}^{2C_n}\frac{\log T}{T}[Pdim(\mathcal{F}) \vee Pdim(\mathcal{H})] \leq \eta_n$. With $T \geq \max\{T_n, T(d^*)\}$, the condition (33) is verified.

Lemma 12 *(Zhou et al., 2022, Lemma B5) Let f be a uniformly continuous function defined on $E \subset [-R,R]^d$. For any $L, N \in \mathbb{N}^+$, there exists a ReLU network function $f_{\boldsymbol{\phi}}$ with width $3^{d+3}\max\{d\lfloor N^{1/d}\rfloor, N+1\}$ and depth $12L + 14 + 2d$ such that*

$$\|f - f_{\boldsymbol{\phi}}\|_{L_\infty(E)} \leq 19\sqrt{d}\,\omega_f^E(2RN^{-2/d}L^{-2/d}),$$

where $\omega_f^E(t)$ is the modulus of continuity of $f(t)$ satisfying $\omega_f^E(t) \to 0$ as $t \to 0^+$.

B.5 Theory for Adversarial Variational Bayes

Theorem 2 *Let $\widehat{\boldsymbol{\beta}}_T$ be as in (36) where $\mathcal{F} = \{f : \|f\|_\infty \leq B\}$ for some $B > 0$. Denote with $\mathbb{E}$ the expectation with respect to $\{Z_j\}_{j=1}^T$ in the reference table. Assume that the prior satisfies (20). Then for $T \geq Pdim(\mathcal{F}\circ\mathcal{G})$ we have for any $C_n > 0$*

$$P_{\theta_0}^{(n)}\mathbb{E}\, d_{TV}^2\left(\nu(X_0), \mu_{\widehat{\boldsymbol{\beta}}_T}(X_0)\right) \leq \mathcal{D}_n^T(\mathcal{F},\mathcal{G},\varepsilon_n,C_n),$$

where

$$D_n^T(\mathcal{F},\mathcal{G},\varepsilon_n,C_n) = \frac{\mathcal{A}_3(\mathcal{F},\mathcal{G})}{2} + \frac{1}{C_n^2 n\varepsilon_n^2} + \frac{1}{2}\widetilde{C}\sqrt{\frac{\log T}{T}Pdim(\mathcal{F}\circ\mathcal{G})} + \frac{\mathrm{e}^{(1+C_2+C_n)n\varepsilon_n^2}}{4}\frac{B}{\sqrt{2}}\mathcal{A}_2(\mathcal{G})$$

for some $\widetilde{C} > 0$ where $\mathcal{A}_2(\mathcal{G})$ was defined in (19) and where

$$\mathcal{A}_3(\mathcal{F},\mathcal{G}) = P_{\theta_0}^{(n)}\mathbb{E}\left\|\log\frac{\pi_{\widehat{\boldsymbol{\beta}}_T}(\theta \mid X_0)}{\pi(\theta \mid X_0)} - f_{\boldsymbol{\omega}(\widehat{\boldsymbol{\beta}}_T)}(X_0,\theta)\right\|_\infty.$$

Here $\mathbb{E}$ account for the nested randomness in the estimation process of $\boldsymbol{\omega}(\widehat{\boldsymbol{\beta}}_T)$ and $\hat{\boldsymbol{\beta}}_T$.

Proof We denote with $\bar{E}$ the expectation with respect to the empirical distribution. Because the class $\mathcal{F}$ is symmetrical (i.e. $f \in \mathcal{F}$ implies $-f \in \mathcal{F}$), the adversarial variational Bayes estimator is defined as

$$\widehat{\boldsymbol{\beta}}_T = \arg\min_{\boldsymbol{\beta}: g_{\boldsymbol{\beta}}\in\mathcal{G}}\left[\bar{E}_{Z\sim\pi_Z} f_{\boldsymbol{\omega}(\boldsymbol{\beta})}(X_0, g_{\boldsymbol{\beta}}(Z,X_0)) - E_{\theta\sim\pi(\theta \mid X_0)} f_{\boldsymbol{\omega}(\boldsymbol{\beta})}(X_0,\theta)\right] \tag{36}$$

where

$$\boldsymbol{\omega}(\boldsymbol{\beta}) = \arg\max_{\boldsymbol{\omega}: f_{\boldsymbol{\omega}} \in \mathcal{F}} \left[\bar{E}_{Z\sim\pi_Z, X\sim\pi(X)} f_{\boldsymbol{\omega}}\left(X, g_{\boldsymbol{\beta}}(Z,X)\right) - \bar{E}_{(\theta,X)\sim\pi(\theta,X)} f_{\boldsymbol{\omega}}(X,\theta)\right].$$

Note that the (stochastic) gradient descent update for $\boldsymbol{\beta}$, conditioning on the most recent value of $\boldsymbol{\omega}$, *does not* involve the second term $E_{\theta\sim\pi(\theta\,|\,X_0)} f_{\boldsymbol{\omega}(\boldsymbol{\beta})}(\theta, X_0)$ in (36) because it does not depend on $\boldsymbol{\beta}$. The minimization occurs only over the first term. We obtain theoretical results for $\widehat{\boldsymbol{\beta}}_T$ and note that our Algorithm 3 targets this estimator. In the sequel, we denote $\widehat{\boldsymbol{\beta}}_T$ simply by $\widehat{\boldsymbol{\beta}}$ and use the notation $\pi_{\boldsymbol{\beta}}(\theta, X) = \pi_{\boldsymbol{\beta}}(\theta\,|\,X)\pi(X)$ for the joint generator model. Using the Pinsker inequality we obtain

$$\begin{aligned} P_{\theta_0}^{(n)} \mathbb{E} 4\, d_{TV}^2\left(\nu(X_0), \mu_{\widehat{\boldsymbol{\beta}}}(X_0)\right) \le & P_{\theta_0}^{(n)} \mathbb{E} \int_\Theta \log \frac{\pi(\theta\,|\,X_0)}{\pi_{\widehat{\boldsymbol{\beta}}}(\theta\,|\,X_0)} [\pi(\theta\,|\,X_0) - \pi_{\widehat{\boldsymbol{\beta}}}(\theta\,|\,X_0)] \mathrm{d}\theta \\ \le & P_{\theta_0}^{(n)} \mathbb{E} 2 \left\| \log \frac{\pi_{\widehat{\boldsymbol{\beta}}}(\theta\,|\,X_0)}{\pi(\theta\,|\,X_0)} - f_{\boldsymbol{\omega}(\widehat{\boldsymbol{\beta}})}(X_0, \theta) \right\|_\infty \\ & + P_{\theta_0}^{(n)} \mathbb{E} d_{\widehat{\boldsymbol{\beta}}}(\nu_{\widehat{\boldsymbol{\beta}}}(X_0), \mu(X_0)), \end{aligned}$$

where we define (for any $\boldsymbol{\beta}, \widetilde{\boldsymbol{\beta}}$ such that $g_{\boldsymbol{\beta}} \in \mathcal{G}$ and $g_{\widetilde{\boldsymbol{\beta}}} \in \mathcal{G}$)

$$d_{\widetilde{\boldsymbol{\beta}}}(\nu_{\boldsymbol{\beta}}(X), \mu(X)) \equiv E_{\theta\sim\pi_{\boldsymbol{\beta}}(\theta\,|\,X)} f_{\boldsymbol{\omega}(\widetilde{\boldsymbol{\beta}})}(X,\theta) - E_{\theta\sim\pi(\theta\,|\,X)} f_{\boldsymbol{\omega}(\widetilde{\boldsymbol{\beta}})}(X,\theta).$$

From the definition of (36) and since $\mathcal{F}\circ\mathcal{G}$ is symmetrical, we have for any realization X_0 and for any $\boldsymbol{\beta}$

$$\begin{aligned} d_{\widehat{\boldsymbol{\beta}}}(\nu_{\widehat{\boldsymbol{\beta}}}(X_0), \mu(X_0)) &= d_{\widehat{\boldsymbol{\beta}}}(\nu_{\widehat{\boldsymbol{\beta}}}(X_0), \bar{\nu}_{\widehat{\boldsymbol{\beta}}}(X_0)) + d_{\widehat{\boldsymbol{\beta}}}(\bar{\nu}_{\widehat{\boldsymbol{\beta}}}(X_0), \mu(X_0)) \\ &\le d_{\widehat{\boldsymbol{\beta}}}(\nu_{\widehat{\boldsymbol{\beta}}}(X_0), \bar{\nu}_{\widehat{\boldsymbol{\beta}}}(X_0)) + d_{\boldsymbol{\beta}}(\bar{\nu}_{\boldsymbol{\beta}}(X_0), \mu(X_0)) \\ &\le d_{\mathcal{F}}(\nu_{\widehat{\boldsymbol{\beta}}}(X_0), \bar{\nu}_{\widehat{\boldsymbol{\beta}}}(X_0)) + d_{\boldsymbol{\beta}}(\bar{\nu}_{\boldsymbol{\beta}}(X_0), \nu_{\boldsymbol{\beta}}(X_0)) + d_{\boldsymbol{\beta}}(\nu_{\boldsymbol{\beta}}(X_0), \mu(X_0)) \\ &\le 2 d_{\mathcal{F}\circ\mathcal{G}}(\pi_Z, \bar{\pi}_Z) + d_{\boldsymbol{\beta}}(\nu_{\boldsymbol{\beta}}(X_0), \mu(X_0)). \end{aligned} \tag{37}$$

Next, using the same arguments as in the proof of Theorem 1, we obtain

$$P_{\theta_0}^{(n)} \mathbb{E} d_{\boldsymbol{\beta}}(\nu_{\boldsymbol{\beta}}(X_0), \mu(X_0)) \le 4\, \mathbb{P}_{\theta_0}^{(n)}[\mathcal{A}_n^c(\epsilon)] + \mathrm{e}^{(1+C_n+C_2)n\epsilon^2} \mathbb{E} \int_{\mathcal{X}} d_{\boldsymbol{\beta}}(\nu_{\boldsymbol{\beta}}(X), \mu(X)) \pi(X) \mathrm{d}X$$

Since $\|f_{\boldsymbol{\omega}(\boldsymbol{\beta})}\|_\infty \le B$, we have for any $\boldsymbol{\beta}$

$$\begin{aligned} E_X d_{\boldsymbol{\beta}}(\mu_{\boldsymbol{\beta}}(X), \nu(X)) &= \int_{\mathcal{X}} \pi(X) \int_\Theta f_{\boldsymbol{\omega}(\boldsymbol{\beta})}(X,\theta) [\pi(\theta\,|\,X) - \pi_{\boldsymbol{\beta}}(\theta\,|\,X)] \mathrm{d}\theta \mathrm{d}X \\ &\le B \times E_X d_{TV}(\mu_{\boldsymbol{\beta}}(X), \nu(X)) \\ &\le \frac{B}{\sqrt{2}} E_X \sqrt{\left\| \log \frac{\pi(\theta\,|\,X)}{\pi_{\boldsymbol{\beta}}(\theta\,|\,X)} \right\|_\infty}. \end{aligned}$$

In the sequel, we choose $\boldsymbol{\beta}$ which minimizes this term. Next, using the symmetrization techniques as before in the proof of Theorem 1 and denoting with $\mathbb{E}$ the expectation with respect to $\{Z_j\}_{j=1}^T$, we have

$$d_{\mathcal{F}\circ\mathcal{G}}(\pi_Z, \bar{\pi}_Z) \leq \mathbb{E}\mathcal{R}_n(\mathcal{F}\circ\mathcal{G}) \leq \widetilde{C}\sqrt{\frac{\log T}{T} Pdim(\mathcal{F}\circ\mathcal{G})}.$$

Putting the pieces together, we obtain an upper bound $D_n^T(\mathcal{F}, \mathcal{G}, \varepsilon_n, C_n)$ for $P_{\theta_0}^{(n)} \mathbb{E}\, d_{TV}^2 \left(\nu(X_0), \mu_{\widehat{\boldsymbol{\beta}}_T}(X_0)\right)$.

Appendix C. More on Adversarial Variational Bayes

The VB algorithm reported in the main paper follows the scheme in Algorithm 3. Figure 9 includes an example of performance of Algorithm 3 on another realization of Example 1.

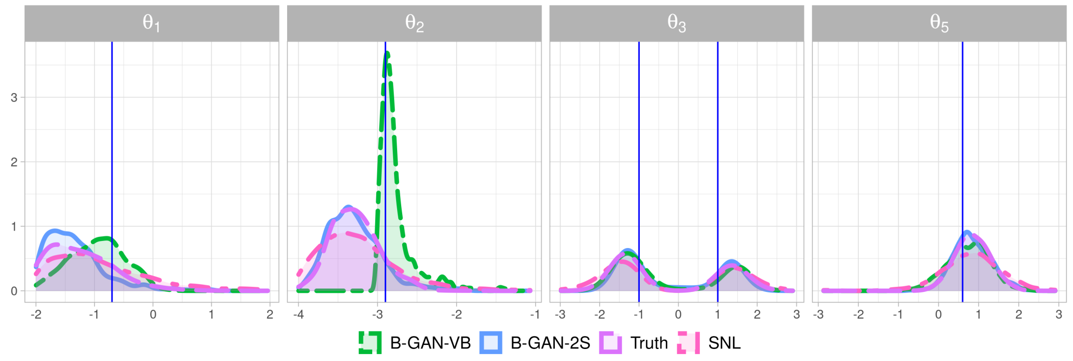


Figure 9: Posterior densities under the Gaussian model (another repetition). The true parameter is $\boldsymbol{\theta}_0 = (-0.7, -2.9, -1.0, -0.9, 0.6)'$, while the signs of θ_3 and θ_4 are not identifiable.

Appendix D. Jensen-Shannon Version of B-GAN

The empirical version of the minimax game using (22) involves the ABC reference table $\{(\theta_j, X_j)\}_{j=1}^T$ and noise realizations $\{Z_j\}_{j=1}^T \sim \pi_Z(\cdot)$ to solve

$$(g^*, d^*) = \arg\min_{g\in\mathcal{G}}\max_{d\in\mathcal{D}} \left(\sum_{j=1}^T \log d(X_j, \theta_j) + \sum_{j=1}^T \log\left[1 - d(X_j, g(Z_j, X_j))\right]\right). \tag{38}$$

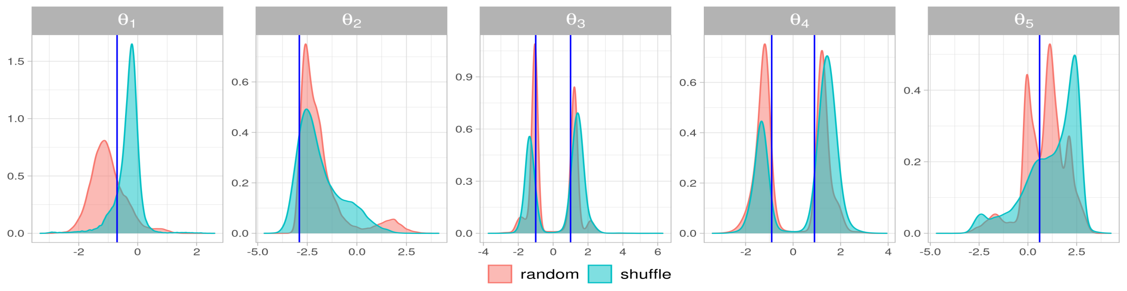
The convergence difficulties of Jensen-Shannon (JS) GANs (Goodfellow et al., 2014), similar to Algorithm 4, have been no secret. We provide a simple illustration of how it can fail on the toy example in Example 1. In particular, we show that the convergence is very sensitive to the choice of the learning rate (step size in stochastic gradient descent). To make comparisons with the Wasserstein version more fair, we use the same network architecture used for Algorithm 1 described in Appendix E.2 for the JS version as well. At the end of this section we point out that JS can work with more careful tuning at higher computational cost.

Algorithm 4 *B-GAN (Jensen-Shannon Version)*

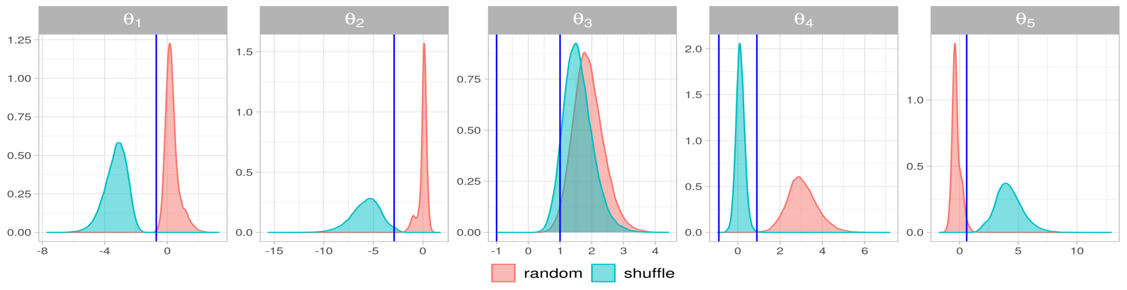
```
INPUT
Prior $\pi(\theta)$, observed data $X_0$ and noise distribution $\pi_Z(\cdot)$
PROCEDURE
Initialize networks $d^{(0)}$ and $g^{(0)}$
ABC Reference Table
For $j = 1, \dots, T$: Generate $(X_j, \theta_j)$ where $\theta_j \sim \pi(\theta)$ and $X_j \sim P_{\theta_j}^{(n)}$.
GAN Training
For $t = 1, \dots, N$:
    Generate noise $Z_j \sim \pi_Z(z)$ for $j = 1, \dots, T$.
    Update $d^{(t)}$ and $g^{(t)}$ using stochastic gradient descent applied to (38).
POSTERIOR SIMULATION
For $i = 1, \dots, M$: Simulate $Z_i \sim \pi_Z(z)$ and set $\widetilde{\theta}_i = g^{(N)}(Z_i, X_0^{(n)})$ .
```

One of the most common causes of failure is an overly-powerful discriminator. Because of the 'log loss', when the discriminator learns too fast and becomes too strong, the 'close-to-boundary' predictions (near 0 for fake data or near 1 for real data) cause vanishing gradients for the generator. We show how the training balance (between the generator and the discriminator) can be easily disturbed when we alter the learning rate of the generator. We consider two scenarios where the noise variables $\{Z_j\}_{j=1}^T$ are (a) refreshed for each stochastic gradient step, and (b) when they are sampled only once ahead of time and then random minibatches selected for each stochastic gradient steps. In Figure 10, we report the four approximated posteriors for these two scenarios considering two learning rates of the generator (weak $\mathrm{lr}_g = 10^{-4}$ versus strong $\mathrm{lr}_g = 10^{-3}$) while keeping the learning rate of the discriminator at 10^{-4}. When the generator is weak compared to the discriminator ($\mathrm{lr}_g = 10^{-4}$), the posterior reconstructions are not quite around the true values (bottom figure). When the generator is stronger ($\mathrm{lr}_g = 10^{-3}$), the posteriors at least cover the correct locations of the parameters, but they are not nearly as successful as the Wasserstein B-GAN reconstructions we have seen in Figure 1. It is also interesting to compare the results for the two treatments of the noise Z. Sampling Z's ahead of time and then sub-sampling for stochastic gradient yields less satisfactory reconstructions.

The JS version of B-GAN could work with a more delicate calibration and extensive training. The Gaussian example was also studied in Ramesh et al. (2022, Section 3.1), who constructed much bigger and deeper networks for both the discriminator and generator. They used a 4-layer network for the generator, with layer widths equal to $(d_X + d_Z, 128, 128, 128, 128, d_\theta)$, and a 5-layer network for the discriminator, with layer widths $(d_X + d_\theta, 2048, 2048, 2048, 2048, 2048, 1)$. They used leaky ReLU activations with a 0.1 negative slope. Using 100 000 pairs of (X_i, θ_i) (twice as many compared to what we used), the networks were trained for 20 000 epochs with 100 discriminator updates per each generator update. Spectral normalization is also applied to ensure stable training. Yet, the authors observed that the performance on the Gaussian example is inferior to SNL (Figure 2A in their paper). Our Wasserstein B-GAN implementation could outperform SNL with much simpler networks, smaller ABC reference table and significantly lower opti-



(a) $\mathrm{lr}_g = 10^{-3}$



(b) $\mathrm{lr}_g = 10^{-4}$

Figure 10: Approximated posteriors under different generator learning rates and different Z randomizations. The learning rate of the discriminator is fixed at 10^{-4} in both. The blue vertical lines mark the correct locations of the parameters.

mization costs. We explored other structures, which are different from the ones mentioned above and simpler than the ones used by the authors. For example, a generator network with input-output dimensionality as $(d_X + d_Z, 128, 128, 128, d_\theta)$ and a discriminator network with input-output dimensionality as $(d_X + d_\theta, 512, 512, 512, 512, 1)$. However, they did not produce satisfactory posterior approximations.

Appendix E. Implementation

E.1 Implementation details of the one-sided gradient penalty

For the Lipschitz constraint in (5), to avoid the ad hoc gradient clipping in Arjovsky and Bottou (2017), Gulrajani et al. (2017) proposed the soft constraint with a penalty on the gradient of f_ω with respect to a convex combination of the two constrasting datasets. They adopt the two-sided penalty encouraging the norm of the gradient to go towards 1 instead of just staying below 1 (one-sided penalty). This is inspired by the fact that the optimal critic function contains straight lines with gradient norm 1 connecting coupled points from the contrasted distributions (Gulrajani et al., 2017, Proposition1). Similarly as Athey et al. (2021), we adopt the one-sided penalty only with respect to θ_j

$$\lambda\Big\{\frac{1}{T}\sum_{j=1}^{T}\Big[\max\Big(0, \Big\|\nabla_{\bar\theta} f_{\boldsymbol\omega}(X_j, \bar\theta_j)\Big\|_2 - 1\Big)\Big]^2\Big\} \tag{39}$$

where $\bar{\theta}_j = \epsilon_j \theta_j + (1-\epsilon_j) g(Z_j, X_j)$ with the ϵ_j re-drawn from a uniform distribution at each step. The choice of λ is discussed in Appendix E.

E.2 Implementation Details for the Gaussian Example

	$\theta_1 = 0.7$		$\theta_2 = -2.9$		$\|\theta_3\| = 1.0$		$\|\theta_4\| = 0.9$		$\theta_5 = 0.6$	
	bias	CI width	bias	CI width	bias	CI width	bias	CI width	bias	CI width
Truth	0.63	1.50 (0.8)	0.39	1.04 (0.9)	0.37	2.85	0.27	2.39	0.83	1.68 (0.8)
SNL	0.73	3.50	0.49	2.38	0.40	4.11	0.28	3.47	0.88	3.33
SS	1.25	5.35	1.48	6.53	0.83	5.66	0.86	5.67	1.56	5.69
W2	1.14	3.74	0.98	3.29 (0.9)	0.46	4.20	0.45	3.70	1.36	5.52
	ReLU activations									
B-GAN	0.69	2.89	0.43	2.08	0.37	3.54	0.29	3.12	0.87	2.38 (0.8)
	Architecture (128, 128, 128)									
B-GAN-2S	0.57	1.86 (0.8)	0.32	1.15 (0.9)	0.31	2.91 (0.9)	0.23	2.42 (0.9)	0.84	2.16 (0.8)
	Architecture (256, 256)									
B-GAN-2S	0.57	1.73 (0.8)	0.32	1.18 (0.9)	**0.28**	**2.76 (0.8)**	0.20	**2.30 (0.9)**	0.73	1.79 (0.8)
	Leaky ReLU activations									
B-GAN	0.64	2.74	0.39	1.92	0.35	3.53	0.26	3.04	0.86	2.28 (0.8)
B-GAN-2S	**0.53**	1.77 (0.8)	0.30	**1.12 (0.9)**	0.33	2.98	0.27	2.52	0.83	2.12 (0.9)

Table 4: Summary statistics of the approximated posteriors under the Gaussian model (averaged over 10 repetitions). For θ_3 and θ_4, we compute the statistics using the absolute values of the parameters, since the posteriors have symmetric modes. Truth refers to the posterior calculated from the exact likelihood function. Bold fonts mark the best model of each column (excluding the true posterior). The coverage of the intervals are 1 unless otherwise stated in the parenthese.

Network Architectures. Our implementation of Algorithm 1 in python builds on the codes provided by (Athey et al., 2021).[4] We use fairly modest generator/critic networks. For the generator network, we use only 3 hidden layers totaling in dimensions $(d_Z + d_X, 128, 128, 128, d_\theta)$. For the critic network, we use a similar architecture with layer dimensions equal to $(d_\theta + d_X, 128, 128, 128, 1)$. A small amount of regularization (dropout = 0.1 for each layer) was applied to avoid over-fitting. All the weights are initialized at 0 and the $\pi_Z(\cdot)$ is specified as the mean zero Gaussian with identity covariance matrix, dimension d_Z equal to d_θ.

For the post-processing enhancements in Algorithm 2 and Algorithm 3, we choose shallower but wider networks with the hope that they will better capture local aspects (Chen and White, 1999). In particular, for both the generator and critic networks, we use only two-layer networks, resulting in layer dimensions $(d_Z + d_X, 256, 256, d_\theta)$ for the generator and $(d_\theta + d_X, 256, 256, 1)$ for the critic. We summarize comparisons of this setting with the previous 3 layers of 128 units for B-GAN-2S in Table 4. From the results, we see that wider and shallower networks are superior to the deeper and narrower ones for the two local refinements. We have thereby reported posteriors with two layers of 256 units networks in the main paper. In terms of activation functions, our previous analyses are conducted with a ReLU activation, given its expressibility and inherent sparsity (Goodfellow et al.,

4. https://github.com/evanmunro/dswgan-paper

2016). We have also considered the leaky ReLU with a negative slope $a = 0.1$ (see Table 4) and found no significant advantage of one versus the other. We report results for ReLUs in Figure 1.

Hyperparameters. Regarding the choice of T, for our ABC methods (with summary statistics and the Wasserstein distance), we use a reference table of size $T = 100\,000$. We construct approximated posteriors using the top 1% draws with the smallest ABC discrepancies. For Algorithm 1, we use the same reference table. However, for the stochastic gradient descent updates we use a batchsize $T = 6400$ (implementations in (Ramesh et al., 2022) suggest a batch size around 10% of the total sample size T). The data pairs (X_j, θ_j) are thus subsetted with replacement (not re-sampled) for each iteration of the stochastic gradient descent. The noise variables Z_j's, however, are refreshed (not pre-sampled and subsetted) for each batch. This is commonly used in existing GAN implementations, including (Athey et al., 2021).

We set $n_{\text{critic}} = 15, \lambda = 5$, and the learning rate for the two networks as $\text{lr}_g = \text{lr}_c = 10^{-4}$, which are used in both (Gulrajani et al., 2017) and (Athey et al., 2021). For optimization, we use the ADAptive Moment estimation algorithm (ADAM) by (Kingma and Ba, 2014). We train B-GAN for $N = 1\,000$ epochs or until convergence (the test loss stops decreasing).

For Algorithm 2 and 3, we use smaller reference tables ($T = 50\,000$) with a smaller batchsize 1280, again training the networks for $N = 1000$ epochs. We increase the n_{critic} and the penalty λ with the hope that a regularized and stabilized critic could help the generator to learn better in the local region. In general, a well-behaved critic network always helps but the extra training costs could be high when the sample size is large. Since these local enhancement variants are trained on a smaller reference table, we make these alterations so that they can converge faster with only a minor increase in computation costs. In addition, unlike the JS version of GAN as described in Algorithm 4, the Wasserstein version is less sensitive to the choice of hyper-parameters. Essentially, different hyper-parameters yield similar results and only lead to a trade-off between the convergence speed and computation costs.

Details of Other Methods. For the SNL model (Papamakarios et al., 2019), we adopt the configurations suggested by the authors.[5] They generated $1\,000$ pairs of (θ_j, X_j) in each round, with 5% randomly selected to be used as a validation set. They stopped training if the validation log likelihood did not improve after 20 epochs. They suggested 40 rounds for the Gaussian model. Each Masked Autoregressive Flow (MAF) network has two layers of 50 hidden units with hyperbolic tangent function (tanh) activations.

For ABC methods, we use the mean and variance as summary statistics for naive ABC (SS) and the Wasserstein version is implemented using the 2-Wasserstein (W2) distance under the Euclidean metric defined as $W2(X_i, X_j) = \min_\gamma \left[\sum_{k_1=1}^n \sum_{k_2=1}^n \gamma_{k_1 k_2} \|X_{i,k_1} - X_{j,k_2}\|^2\right]^{1/2}$ s.t. $\gamma' \mathbf{1}_n = \mathbf{1}_n, \gamma \mathbf{1}_n = \mathbf{1}_n$ with $0 \leq \gamma_{k_1 k_2} \leq 1$. Note that when we calculate the 2-Wasserstein distance, each X_i is treated as 4 pairs ($n = 4$) of bi-variate normal variables rather than a flat vector of length 8. For both methods, the approximated posteriors are constructed using the ABC draws with the top 1% smallest data discrepancies. Performance details (after 10 repetitions) are summarized in Table 4.

5. The authors provide their implementations on https://github.com/gpapamak/snl

E.3 Implementation Details for the I.I.D. M/G/1-Queuing Model

For the M/G/1-queuing example, we generate ABC reference table of size $T = 100\,000$ in all three examples. For the stacking and deep set structure, the entire table is generated upfront as $\{(\theta_i, X_i^{(n)})\}$. For the stacking implementation, $X_i^{(n)}$ is flatten as a long tensor of shape $(1, n)$, while the deep set structure takes a three-dimensional tensor of shape $(1, n^*, 5)$. For the sequential update implementation, we generate new ABC reference table with $T = 100\,000$ as $\{\theta_i, X_i^{n/5}\}$ (we use 5 batch update) with θ_i generated from the posterior distribution learned from last iteration and each dataset only contains $n^*/5$ i.i.d. observations and is flatten of shape $(1, n/5)$.

For our deep set architecture for learning the conditional distribution on i.i.d. datasets, it is structured as

$$g_\beta(Z, [X_1, X_2, \ldots, X_n]) = g_{\beta_1}^{(1)}(Z, \sum_{i=1}^n g_{\beta_2}^{(2)}(Z, X_i))$$
$$f_\omega(\theta, [X_1, X_2, \ldots, X_n]) = f_{\omega_1}^{(1)}(\theta, \sum_{i=1}^n f_{\omega_2}^{(2)}(\theta, X_i))$$

where $g_{\beta_1}^{(1)}, g_{\beta_2}^{(2)}, f_{\omega_1}^{(1)}, f_{\omega_2}^{(2)}$ are sub-network structures inside the generator and critic functions. For the M/G/1-queuing example, we use the same deep set structure for $g_{\beta_2}^{(2)}$ and $f_{\omega_2}^{(2)}$ as ReLU networks with 1 hidden layers with 64 units. We use 2 hidden layers with $(64, 64)$ units for $g_{\beta_1}^{(1)}$ and 1 hidden layer with 64 units for $f_{\omega_1}^{(1)}$. We set $\text{lr}_g = 1e^{-3}$ and $\text{lr}_c = 1e^{-4}$, and train for 2 000 epoches with batch size of 1 280.

For the sequential implementation, we use 3 hidden ReLU layers of $(128, 128, 128)$ units for both the generator and the critic functions. We set $\text{lr}_g = 5e^{-4}$ and $\text{lr}_c = 1e^{-4}$ and train the networks with batch size of 1 280. We train the network for 1 000 epoches for the first batch and 500 epoches for the remaining batches.

For the stacked implementation, we use 3 hidden ReLU layers of $(128, 128, 128)$ units for both the generator and the critic functions. We set $\text{lr}_g = 5e^{-4}$ and $\text{lr}_c = 1e^{-4}$ and train the networks for 2 000 epoches with batch size of 1 280.

E.4 Implementation Details for the Lotka-Volterra Example

For the Lotka-Volterra example, we adopt the 9-dimensional summary statistics used in Papamakarios and Murray (2016). For Algorithm 1, we use ReLU neural networks with $L-1 = 3$ hidden layers with $(128, 128, 128)$ units from bottom to top for both the generator and the critic functions. We generate $T = 1\,000\,000$ pairs of (X_i, θ_i) for the vanilla B-GAN, and the networks are trained with a batch size $B = 12\,80$ for 1 000 epochs. We adopt the same $n_{\text{critic}}, \lambda, \text{lr}_g = \text{lr}_c$ as the Gaussian example in Appendix E.2.

For learning from the adjusted prior $\widetilde{\pi}(\theta) = \pi_{\widehat{g}}(\theta \mid X_0)$ where $\widehat{g}$ was obtained from Algorithm 1, we choose shallower networks with $L - 1 = 2$ hidden layers and $W = 256$ hidden units in each layer. We generate $T = 50\,000$ samples for local enhancements and use a batch size $B = 1\,280$ and 1000 epochs for training. For the VB variant (B-GAN-VB in Algorithm 3), we use the same network architecture and training configuration as B-GAN-RL. We set the weights of the VB loss to be 0.2.

For the two ABC methods, we use the same $T = 1\,000\,000$ pairs used in the B-GAN training. We adopt the summary statistics described in Section 5.1 for the naive ABC (SS) and Wasserstein version is calculated on top of the pairs of predator-prey population $\{x_t, y_t\}$. For both models, we again accept ABC draws with the top 1% smallest data discrepancies.

For the SNL model, the author suggested building the network on top of the same set of summary statistics used in naive ABC, and 20 epochs of training. The network architecture and other training configuration remain the same as in Appendix E.2.

E.5 Implementation Details for the Boom-and-Bust Example

We generate $T = 500\,000$ pairs of (X_i, θ_i) for training the vanilla B-GAN in Algorithm 1. We use the same batch size, learning rate and network architectures as in Appendix E.2, except that we train the networks for $2\,000$ epochs this time. We have explored three types of different inputs: (1) the time-series itself; (2) the summary statistics suggested by previous literature such as (An et al., 2020) (described in Section 5.2); (3) the time-series together with the summary statistics. We find the one built on only the summary statistics works best.

For this example, we find that this model is more challenging than the Lotka-Volterra example and the vanilla posterior does not always learn the correct location of θ_0. Directly using the B-GAN posterior results in poor training samples for both the 2-step refinement and the VB implementation. To improve the robustness of our methods in repeated experiments, we have revised the proposal distribution in the second step to be a mixture of 50% prior and 50% posterior from the vanilla B-GAN. This ensures that we can be guided towards the area closer to the true values θ_0 while guarantees that θ_0 is absolutely covered by the proposal distribution.

For the local enhancement variants, B-GAN-2S in Algorithm 2 and B-GAN-VB in Algorithm 3, we generate $T = 50\,000$ samples. The network architectures and training configurations are the same as the ones in Appendix E.2.

For ABC methods, they are built on the same $T = 500\,000$ draws used in B-GAN. We adopt the summary statistics mentioned in Section 5.2 for the naive ABC (SS), and the 2-Wasserstein ABC is trained on the time series.

For SNL models, we use the summary statistics as the input to the networks and train the model for 20 epochs, similar to the setting used in Appendix E.4.

E.6 Implementation Details for the Common Cold example

We generate $T = 500\,000$ pairs of (X_i, θ_i) for training the vanilla B-GAN in Algorithm 1. We use the same batch size, learning rate as in Appendix E.2. We use ReLU networks with $L - 1 = 2$ hidden layers with $(128, 128)$ units from bottom to top for both the generator and the critic functions.

For B-GAN-RL and B-GAN-VB, we generate $T = 50\,000$ samples from the B-GAN posterior.The network architectures and training configurations are the same as B-GAN.

E.7 Computation Costs Comparison

We provide comparisons of computation times of each method in Table 5 for the Gaussian example and the Lotka-Volterra model. Note that SS, W2 and SNL were executed with CPUs and all B-GAN models were computed using GPUs. We report the time of B-GAN-2S and B-GAN-VB for computation using the adjusted prior only (i.e. without the pilot run). Since B-GAN model learns the joint distribution of (X, θ) and is universal regardless of the observed data X_0, we recycle the same pre-trained B-GAN model to recover the adjusted prior repeatedly for different X_0 in our simulation study. Although we only report the computation time for one repetition here, this feature saves a lot of computation costs when one wants to investigate average performance from multiple repetitions.

The complexity of computing the exact Wasserstein distance is $\mathcal{O}(T^3)$ (Burkard et al., 2009) and $\mathcal{O}(T^2)$ (Cuturi, 2013) for the approximate one. The computation costs of SNL and B-GAN depends on the network architecture and how fast the networks converge. We can only give a rough computation complexity estimate as $\mathcal{O}(T \times \#epochs \times \#weights)$ for these neural network based models. From Table 5, we observe that our methods could be more scalable than W2 and SS when the dimension of the dataset is high (8 of Gaussian model vs 402 of Lotka-Volterra (LV) model). The computation time of SNL on LV is smaller than for the Gaussian model due to fewer epochs in training (the author suggested 20 rounds for LV and 40 rounds for Gaussian model). In addition, SNL is trained on summary statistics ($q = 9$) rather than the time-series for the LV model. The computation costs of SS increase significantly on the LV example, resulting from both the increase in dimension and the computation costs of the selected summary statistics. The computation costs of the Wasserstein distance are unsurprisingly high as it is known that they do not scale well.

	SS	W2	SNL	B-GAN	B-GAN-2S	B-GAN-VB
Gauss	33.75	221.28	4790.56	2736.93	676.25	726.22
Lotka-Volterra	5846.95	162644.96	3080.96	1610.05	762.21	753.61

Table 5: Computation time of one repetition for each method on Gauss example and Lotka-Volterra (LV) example (in seconds). The time of B-GAN-2S and B-GAN-VB is for computation using the adjusted prior.