Predictive Approximate Bayesian Computation via Saddle Points

Yingxiang Yang* Bo Dai* Negar Kiyavash† Niao He*†
{yyang172,kiyavash,niaohe} @illinois.edu
bohr.dai@gmail.com

Abstract

Approximate Bayesian computation (ABC) is an important methodology for Bayesian inference when the likelihood function is intractable. Sampling-based ABC algorithms such as rejection- and K2-ABC are inefficient when the parameters have high dimensions, while the regression-based algorithms such as K- and DR-ABC are hard to scale. In this paper, we introduce an optimization-based ABC framework that addresses these deficiencies. Leveraging a generative model for posterior and joint distribution matching, we show that ABC can be framed as saddle point problems, whose objectives can be accessed directly with samples. We present *the predictive ABC algorithm (P-ABC)*, and provide a probabilistically approximately correct (PAC) bound for its learning consistency. Numerical experiment shows that P-ABC outperforms both K2- and DR-ABC significantly.

1 Introduction

Approximate Bayesian computation (ABC) is an important methodology to perform Bayesian inference on complex models where likelihood functions are intractable. It is typically used in large-scale systems where the generative mechanism can be simulated with high accuracy, but a closed form expression for the likelihood function is not available. Such problems arise routinely in modern applications including population genetics [Excoffier, 2009, Drovandi and Pettitt, 2011], ecology and evolution [Csilléry et al., 2012, Huelsenbeck et al., 2001, Drummond and Rambaut, 2007], state space models [Martin et al., 2014], and image analysis [Kulkarni et al., 2014].

Formally, ABC aims to estimate the posterior distribution $p(\theta|y) \propto \pi(\theta)p(y|\theta)$. The word "approximate" refers to the fact that the joint distribution $\pi(\theta)p(y|\theta)$ is only available through fitting simulated data $\{(\theta_j, y_j)\}_{j=1}^N \sim p(y|\theta)\pi(\theta)$. Based on how the fitting is performed, existing ABC methods can be summarized into two main categories: sampling- and regression-based algorithms.

Sampling-based algorithms. A sampling-based algorithm directly approximates the likelihood function using simulated samples "similar" to the true observations according to certain choices of similarity measurement based on informative summary statistics, e.g., [Joyce and Marjoram, 2008, Nunes and Balding, 2010, Blum and François, 2010, Wegmann et al., 2009, Blum et al., 2013]. More recent representative algorithms include rejection ABC, indirect score ABC [Gleim and Pigorsch], K2-ABC [Park et al., 2016], distribution regression ABC (DR-ABC) [Mitrovic et al., 2016], expectation propagation ABC (EP-ABC) [Barthelmé and Chopin, 2011], random forest ABC [Raynal et al., 2016], Wasserstein ABC [Bernton et al., 2017], Copula ABC [Li et al., 2017], and ABC aided by neural network classifiers [Gutmann et al., 2014, 2016]. The aforementioned work can be viewed under a unified framework that approximates the posterior $p(\theta|y)$ with

$$p_\epsilon(\theta|y) \propto \int_{\mathcal{Y}} K_\epsilon(s_x, s_y)p(x|\theta)\pi(\theta)\mathrm{d}x \approx \frac{\pi(\theta)}{N}\sum_{i=1}^{N} K_\epsilon(s_{x_i}, s_y), \tag{1}$$

*Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign.
†Department of Industrial and Enterprise Systems Engineering, University of Illinois at Urbana-Champaign.
⋆Google Brain.

32nd Conference on Neural Information Processing Systems (NIPS 2018), Montréal, Canada.

where $\mathcal{Y}$ is the domain for the samples x and y, and the x_i's are drawn from the model $p(x_i|\theta)$. We use s_y to denote the summary statistics for y, and let $K_\epsilon(s_x, s_y)$ be an appropriate weighting kernel that measures similarity. For example, when $K_\epsilon(s_x, s_y) = \mathbf{1}\{s_x = s_y\}$ with $s_x = x$ and θ and y discrete, (1) recovers the true posterior asymptotically. When $K_\epsilon(s_x, s_y) = \mathbf{1}\{\rho(s_x, s_y) \leq \epsilon\}$ for some metric ρ, (1) reduces to rejection-ABC. When $K_\epsilon(s_x, s_y) = \exp(-\rho(s_x, s_y)/\epsilon)$, (1) reduces to soft-ABC [Park et al., 2016], a variation of the synthetic likelihood inference [Wood, 2010, Price et al., 2018] under the Bayesian setting. Finally, when $K_\epsilon(s_x, s_y)$ is the zero/one output of a neural network classifier, (1) reduces to ABC via classification [Gutmann et al., 2018].

Note that, most of the aforementioned algorithms require summary statistics and a smoothing kernel, which suffer from information loss when the summary statistics are insffucient, and introduce bias. To address the issue of insufficient summary statistics, Park et al. [2016] proposed K2-ABC, in which $K_\epsilon(s_x, s_y)$ is replaced by a smoothing kernel over the empirical maximum mean discrepancy (MMD) obtained from kernel embedding of the empirical distributions of the samples. When a characteristic kernel is selected, the kernel embedding of the distribution will be a sufficient statistics, therefore, without information loss. Meanwhile, Rodrigues et al. [2018] proposed recalibration techniques to debias the estimates of ABC algorithms. However, despite their simplicity and continuous improvements, sampling-based ABC algorithms suffer from bias caused by the weighting kernel K_ϵ, and the potential need of large amount of samples when the dimensions of θ and y are large.

Regression-based algorithms. Regression-based ABC algorithms establish regression relationships between the model parameter and the simulated data within an appropriate function space $\mathcal{F}$. Representative algorithms in this category include high-dimensional ABC [Nott et al., 2014], Kernel-ABC (K-ABC) [Blum et al., 2013], DR-ABC [Mitrovic et al., 2016]. In DR-ABC, the posterior is obtained by performing a distribution regression using the given samples. In contrary to the sampling-based algorithms, regression-based algorithms mitigate the bias introduced by the smoothing kernel. However, they do not provide an estimation for the posterior density. Meanwhile, it is often hard for such algorithms to scale. For example, the distribution regression involved in DR-ABC requires computing the inverse of an $N \times N$ kernel matrix, which has $\mathcal{O}(N^3)$ computation cost as the dataset scales.

Neither sampling- nor regression-based algorithms are satisfactory: while regression-based algorithms have better performances compared to the sampling-based algorithms, they are not scalable to high dimensions. Therefore, an important question is whether one can design an algorithm that can perform well on large datasets? In this paper, we propose an optimization-based ABC algorithm that can successfully address the deficiencies of both sampling- and regression-based algorithms. In particular, we show that ABC can be formulated under a unified optimization framework: finding the saddle point of a minimax optimization problem, which allows us to leverage powerful gradient-based optimization algorithms to solve ABC. More specifically, our contributions are three-fold: **First**, we start with a generative model for posterior approximation and show that the ABC problem can be formulated as a saddle point optimization through both joint distribution matching and posterior matching. This approach circumvents the difficulties associated with choosing sufficient summary statistics or computing kernel matrices, as needed in K2- and DR-ABC. More critically, the saddle point objectives can be evaluated based purely on samples, without assuming any implicit form of the likelihood. **Second**, we provide an efficient SGD-based algorithm for finding the saddle point, and provide a probabilistically approximately correct (PAC) bound guaranteeing the consistency of the solution to the problem. **Numerically**, we compare the proposed algorithm to K2- and DR-ABC. The experiment shows that our algorithm outperforms K2- and DR-ABC significantly and is close to optimal on the toy example dataset.

2 Approximate Bayesian Computation via Saddle Point Formulations

When the likelihood function is given, the true posterior $p(\theta|y)$ given observation y can be obtained by optimizing the *evidence lower bound* (ELBO) in the space $\mathcal{P}$ that contains all probability density functions [Zellner, 1988],

$$\min_{q(\theta)\in\mathcal{P}} \ \mathrm{KL}(q||\pi) - \mathbb{E}_{\theta\sim q}[\log p(y|\theta)], \tag{2}$$

where KL denotes the Kullback-Leibler divergence: $\mathrm{KL}(q\|\pi) = \mathbb{E}_{\theta\sim q}[\log \frac{q(\theta)}{\pi(\theta)}]$. When dealing with an intractable likelihood, this conventional optimization approach cannot work without combining it with methods that fit $p(y|\theta)$ with samples. In this paper, we introduce a new class of saddle point optimization objectives that allow the learner to directly leverage the samples from the likelihood $p(y|\theta)$, which is available under the ABC setting, for estimating the posterior. The method we

propose does not merely find $\theta^* = \operatorname{argmax}_\theta p(\theta|y)$ for a given data point y, but rather finds the optimal $p(\theta|y)$, or a representation of θ generated from $p(\theta|y)$ from a transportation reparametrization $\theta = f(y, \xi)$ for any data y (an idea inspired by Kingma and Welling [2013]), with asymptotically diminishing statistical error. We introduce our method below.

2.1 Saddle Point Objectives

Joint distribution matching. Recall that $p(y|\theta)\pi(\theta) = p(\theta|y)p(y)$, a natural idea for estimating the posterior is to match the empirical joint distributions, given the availability of sampling from the joint distribution $p(y|\theta)\pi(\theta)$ and from the posterior $p(\theta|y)$. Using an f-divergence associated with some convex function ν, defined by $D_\nu(p, q) = \int q(x)\nu\left(p(x)/q(x)\right) \mathrm{d}x$, as our loss function, we have the following divergence minimization problem for ABC:

$$p(\theta|y) = \operatorname*{argmin}_{q(\theta|y)\in\mathcal{P}} D_\nu\left(p(y|\theta)\pi(\theta), q(\theta|y)p(y)\right), \tag{3}$$

in which the left-hand side is a posterior distribution while the right-hand side integrates over θ and y. The above optimization problem is difficult to solve since D_ν is nonlinear with respect to $q(\theta|y)$. This nonlinearity makes gradient computation hard as the f-divergence cannot be computed directly through samples obtained from the joint distribution. To address this issue, we apply Fenchel duality and the interchangeability principle as introduced in Dai et al. [2017], which yield an equivalent saddle point reformulation:

$$\min_{q(\theta|y)\in\mathcal{P}} \max_{u(\theta,y)\in\mathcal{U}} \Phi(f, u) := \mathbb{E}_{(\theta,y)\sim p(y|\theta)\pi(\theta)}\left[u(\theta, y)\right] - \mathbb{E}_{\theta\sim q(\theta|y), y\sim p(y)}\left[\nu^*\left(u(\theta, y)\right)\right]. \tag{4}$$

In (4), $\mathcal{U}$ is a function space containing $u^*(\theta, y) = \nu'(\frac{p(y|\theta)p(\theta)}{p(\theta|y)p(y)})$ and ν^* is the Fenchel dual of ν. When $\mathcal{P}$ is expressive enough, a saddle point solution to (4) recovers the posterior distribution.

The class of f-divergence covers many common divergences, including the KL divergence, Pearson χ^2 divergence, Hellinger distance, and Jensen-Shannon divergence. Apart from f-divergences, we can also employ other metrics to measure the distance between the joint distributions $p(y|\theta)p(\theta)$ and $p(\theta|y)p(y)$, e.g., the Wasserstein distance. If the training data come with labels, we can also choose the objective function to be the mean square error between the label and the maximum a posterior estimate from $p(\theta|y)$. [2] From a density ratio estimation perspective, the optimal solution of the dual variable, $u(\theta, y)$, is a discriminator that tells the true and synthetic joint distributions by computing their density ratios, which is related to the ratio matching in Mohamed and Lakshminarayanan [2016].

Posterior matching. Another way to learn the posterior representation is by directly matching the posterior distributions. Similar to the objective function defined in K-ABC, we have

$$\min_{q(\theta|y)\in\mathcal{P}} \max_{h(\theta)\in\mathcal{H}} \mathbb{E}_y\left[\left(\mathbb{E}_{\theta|y}[h(\theta)] - \mathbb{E}_{\theta\sim q(\theta|y)}[h(\theta)]\right)^2\right]. \tag{5}$$

Directly solving the optimization (5) is difficult due to the inner conditional expectation, but a saddle point formulation can be obtained by applying the same technique we used to obtain (4) (see Appendix B for detailed derivations):

$$\min_{q(\theta|y)\in\mathcal{P}} \max_{\substack{h(\theta)\in\mathcal{H}\\ v(y)\in\mathcal{V}}} \mathbb{E}_{(\theta,y)\sim p(y|\theta)\pi(\theta)}\left[v(y)h(\theta)\right] - \mathbb{E}_{(\theta,y)\sim q(\theta|y)p(y)}\left[v(y)h(\theta)\right] - \frac{1}{4}\mathbb{E}_y\left[v^2(y)\right] \tag{6}$$

where $\mathcal{V}$ is the entire space of functions on $\mathcal{Y}$. The resulting saddle point objective (6) is much easier to solve than (5) and stochastic gradient-based methods could be applied in particular.

2.2 Representations of $u(\theta, y)$ and $q(\theta|y)$

Under the most general setting where $\mathcal{P}$ and $\mathcal{U}$ are closed and bounded function spaces, (4) is convex-concave for which a unique solution can be obtained. Practically, different representation methods can be used for $u(\theta, y)$ and $q(\theta|y)$, for which different optimization techniques can be applied to solving (4). Below, we discuss several commonly used options.

Gaussian mixtures. Consider the following Gaussian mixture representation for $q(\theta|y)$ and $u(\theta, y)$:

$$q(\theta|y) = \sum_{i=1}^{m} c_i^{(q)}(y) \cdot \mathcal{N}(\mu_i^{(q)}, \Sigma^{(q)}; \theta) \quad \text{and} \quad u(\theta, y) = \sum_{i=1}^{m} c_i^{(u)} \cdot \mathcal{N}(\mu_i^{(u)}, \Sigma^{(u)}; (\theta, y)). \tag{7}$$

[2] Table 2 in Appendix provides some examples of divergences and the derivation of their corresponding saddle point objectives.

The coefficients $c_1^{(u)}, \ldots, c_m^{(u)}$ are positive real numbers while $c_1^{(q)}(y), \ldots, c_m^{(q)}(y)$ are y-dependent coefficients. A simple way to guarantee that the summation of $c_i^{(q)}(y)$ is one for any y is to assume that they take the form of softmax functions:

$$c_i^{(q)}(y) = \frac{\exp([1, y^\top] \cdot c_i^{(q)})}{\sum_{j=1}^m \exp([1, y^\top] \cdot c_j^{(q)})}, \quad \forall i \in \{1, \ldots, m\}, \tag{8}$$

with $c_1^{(q)} = 0$. This results (4) to be convex for $c_i^{(q)}$ and concave for $c_i^{(u)}$.

Transportation reparametrization. When the dimensions of θ and y increase, the conditional distribution $q(\theta|y)$ quickly becomes difficult to represent using parametric models. An effective way to implicitly represent $q(\theta|y)$ is to use a sampler $f(\xi, y) \in \mathcal{F}$ for a function space $\mathcal{F}$, in which θ is sampled using $\theta = f(\xi, y)$ using a pre-determined distribution $\xi \sim p_0(\xi)$. This idea is inspired by the reparametrization technique used in variational autoencoders (VAEs) and neural networks. In our case, both f and u can be represented using functions in reproducing kernel Hilbert spaces (RKHSs) or neural networks.

2.3 Discussions

The saddle point framework is closely related to both regression- and GAN-based ABC algorithms.

Relationship with regression-based ABC algorithms. Regression-based ABC algorithms, such as K-ABC, aim to compute the conditional expectation of the posterior by finding its conditional kernel embedding $C(y) : \mathcal{Y} \to \mathcal{H}$ in an RKHS. With such parametrization, the objective (5) becomes

$$\min_{C:\mathcal{Y}\to\mathcal{H}} L(C) := \sup_{\|h\|_{\mathcal{H}}\le 1} \mathbb{E}_y[(\mathbb{E}_{\theta|y}[h(\theta)] - \langle h, C(y)\rangle_{\mathcal{H}})^2].$$

This problem is further relaxed to a distribution regression problem by swapping the square operator with the inner expectation, which leads to minimizing $\mathbb{E}_{\theta,y}[\|K(\cdot,\theta) - C(y)\|^2]$, an upper bound of $L(C)$. Specifically, we have

$$\begin{aligned}
& \sup_{\|h\|_{\mathcal{H}}\le 1} \mathbb{E}_y \left[\left(\mathbb{E}_{\theta|y} \left[h(\theta) \right] - \langle h, C(y) \rangle_{\mathcal{H}} \right)^2 \right] \le \sup_{\|h\|_{\mathcal{H}}\le 1} \mathbb{E}_y \left[\left(\mathbb{E}_{\theta|y} \left[\langle h, k(\cdot,\theta) \rangle \right] - \langle h, C(y) \rangle_{\mathcal{H}} \right)^2 \right] \\
\le & \sup_{\|h\|_{\mathcal{H}}\le 1} \mathbb{E}_{\theta,y} \left[\langle h, k(\cdot,\theta) - C(y) \rangle^2 \right] \le \sup_{\|h\|_{\mathcal{H}}\le 1} \|h\|_{\mathcal{H}}^2 \, \mathbb{E}_{\theta,y} \left[\|k(\cdot,\theta) - C(y)\|^2 \right] \\
= & \mathbb{E}_{\theta,Y} \left[\|k(\cdot,\theta) - C(y)\|^2 \right].
\end{aligned}$$

In contrast, the proposed optimization framework for posterior matching does not restrict $h \in \mathcal{H}$. Moreover, the saddle point objective (6) is an exact reformulation of (5), rather than an upper bound.

Relationship with GAN-based ABC algorithms. GAN-based algorithms leverage the representation power of the neural networks to optimize the ELBO. One example is the use of variational autoencoder (VAE), where both q and p in (2) are represented by Gaussian distributions parameterized by neural networks. Better performances have been observed in Mescheder et al. [2017] by embedding the optimal value of $q(\theta|y)$ as the optimal solution of a real-valued discriminator network, equivalent to performing reparametrization. However, compared to the saddle point formulation, Mescheder et al. [2017] requires computing an additional layer of optimization due to the embedding performed. Meanwhile, when the underlying parameter is discrete, the saddle point formulation can be viewed as a special case of conditional GAN (CGAN) [Mirza and Osindero, 2014].

GAN-baseed ABC algorithms can also be modified to adapt to the saddle point framework. One example is to parametrize $q(\theta|y)$ with a generative model $\theta = f(y, \xi)$, and try to find a discriminator u to tell the "synthetic" θ, generated from $f(y, \xi)$, from the "real" θ, generated from $q(\theta|y)$. More specifically, we can use a neural network to solve

$$\min_{f(y,\xi)\in\mathcal{F}} \max_{u(\theta,y)\in\mathcal{U}} \mathbb{E}_{\xi\sim p_0(\xi),(\theta,y)\sim p(y|\theta)\pi(\theta)}[L(u(\theta, y), u(f(y, \xi), y))],$$

where L is some loss function, e.g. $L(a, b) = |a - b|$, and $\mathcal{F}$ is an appropriate function space, such as one that is parameterized by a neural network.

3 Algorithm and Theory

In the last section, we introduced a stochastic saddle point framework to tackle the ABC problem. In this section, we introduce a concrete algorithm named *predictive-ABC* (P-ABC) that solves the

finite-sample approximation (i.e., empirical risk) of this problem. In particular, we consider the empirical risk of (4), where the empirical expectations are taken over N samples $\{(\theta_i, y_i)\}_{i=1}^N$:

$$\min_{q(\theta|y)\in\mathcal{P}} \max_{u(\theta,y)\in\mathcal{U}} \widehat{\Phi}_N(q,u) := \widehat{\mathbb{E}}_{(\theta,y)\sim p(y|\theta)\pi(\theta)} u(\theta,y) - \widehat{\mathbb{E}}_{y\sim p(y)} \left\{ \mathbb{E}_{\theta\sim q(\theta|y)} [\nu^*(u(\theta,y))|y] \right\}. \quad (9)$$

We denote the optimal solution as q_N^* and u_N^*. In the following, we first introduce a general form of P-ABC, followed by customizations to different representation methods for $q(\theta|y)$ and $u(\theta, y)$. We then derive a probabilistically approximately correct (PAC) learning bound on the statistical error

$$\epsilon_N = D_\nu(p(y|\theta)\pi(\theta), q_N^*(\theta|y)p(y)) - D_\nu(p(y|\theta)\pi(\theta), q^*(\theta|y)p(y)),$$

which holds for closed and bounded function spaces $\mathcal{P}$ and $\mathcal{U}$ in general, with q^* and u^* denoting the solution to (4). Lastly, we present the convergence results of P-ABC. For representations of q and u such that the objective function is convex-concave, e.g. Gaussian mixture representations, we present the convergence of Algorithm 1. For transportation reparametrization, on the other hand, the convergence behavior of P-ABC remains largely an open problem.

3.1 The P-ABC Algorithm

We introduce P-ABC for solving (9), the empirical counterpart of (4), in Algorithm 1. This algorithm, in its general form, performs iterative updates to q and u using first-order methods. For the sake of presenation, we use projected gradient descent as the update rule. In practice, depending on the representations of q and u, other forms of updates can be used, such as the mirror descent. Below, we specify the form of the gradients for the representation methods presented in Section 2.

To compute the gradient, we note that for any $q \in \mathcal{P}$ such that the integral can be interchanged with the gradient, $\nabla_q \mathbb{E}_q[f(X)] = \mathbb{E}_q[f(X)\nabla \log q(X)]$. This allows us to take gradient with respect to q and use stochastic gradients for (9) assuming that a set of synthetic samples $\{\widetilde{\theta}_i\}_{i=1}^N$ can be drawn from $q(\theta|y)$ given $\{y_i\}_{i=1}^N$.

Gaussian mixtures. Denote the coefficient vectors for $q(\theta|y)$ and $u(\theta, y)$ in (7) and (8) as $\mathbf{c}^{(q)}$ and $\mathbf{c}^{(u)}$. Then, the stochastic gradients of $\widehat{\Phi}_N(q,u)$ with respect to $q(\theta|y)$ and $u(\theta, y)$ reduce to its gradients with respect to $\mathbf{c}^{(q)}$ and $\mathbf{c}^{(u)}$:

$$\nabla_{\mathbf{c}^{(q)}} \widehat{\Phi}_N = -\widehat{\mathbb{E}}_{y\sim p(y)} \left\{ \widehat{\mathbb{E}}_{\theta\sim q(\theta|y)} \left[\frac{\nu^*(u(\theta,y))}{q(\theta,y)} \cdot \begin{bmatrix} \nabla_{c_1^{(q)}} q(\theta|y) \\ \vdots \\ \nabla_{c_m^{(q)}} q(\theta|y) \end{bmatrix} \right] \right\},$$

$$\nabla_{\mathbf{c}^{(u)}} \widehat{\Phi}_N = \widehat{\mathbb{E}}_{p(y|\theta)\pi(\theta)} [\mathcal{N}(\boldsymbol{\mu}^{(u)}, \Sigma^{(u)}; (\theta,y))] - \widehat{\mathbb{E}}_{q(\theta|y)p(y)} \left[\frac{\mathrm{d}\nu^*(u(\theta,y))}{\mathrm{d}u(\theta,y)} \cdot \mathcal{N}(\boldsymbol{\mu}^{(u)}, \Sigma^{(u)}; (\theta,y)) \right].$$

where $\mathcal{N}(\boldsymbol{\mu}^{(u)}, \Sigma^{(u)}; (\theta, y))$ denotes a vector containing the values of Gaussian proability density functions whose means are specified in (7) and evaluated at (θ, y), while

$$\nabla_{c_i^{(q)}} q(\theta|y) = \sum_{i=1}^m \frac{[1;y] \cdot \exp([1,y^\top] \cdot c_i^{(q)}) \cdot \sum_{j\neq i} \exp([1,y^\top] \cdot c_j^{(q)})}{\left(\sum_{j=1}^m \exp([1,y^\top] \cdot c_j^{(q)})\right)^2} \cdot \mathcal{N}(\mu_i^{(q)}, \Sigma^{(q)}; \theta).$$

Transportation reparametrization. Consider the transportation reparametrization $\theta = f(y, \xi)$ for $\theta \sim p(\theta|y)$. The stochastic gradients of $\widehat{\Phi}_N(f,u)$ in (9) can be computed from the chain rule:

$$\nabla_f \widehat{\Phi}_N = -\widehat{\mathbb{E}}_{(y,\xi)\sim p(y)p_0(\xi)} \left\{ \frac{\mathrm{d}\nu^*(u(f(y,\xi),y))}{\mathrm{d}u(f(y,\xi),y)} \cdot \frac{\partial u(f(y,\xi),y)}{\partial f(y,\xi)} \cdot \nabla_f f(y,\xi) \right\},$$

$$\nabla_u \widehat{\Phi}_N = \widehat{\mathbb{E}}_{(\theta,y)\sim p(y|\theta)\pi(\theta)} \nabla_u u(\theta,y) - \widehat{\mathbb{E}}_{(y,\xi)\sim p(y)p_0(\xi)} \left[\frac{\mathrm{d}\nu^*(u(f(y,\xi),y))}{\mathrm{d}u(f(y,\xi),y)} \cdot \nabla_u u(f(y,\xi),y) \right].$$

When $\mathcal{F}$ and $\mathcal{U}$ are RKHSs, we have $\nabla_f f(y,\xi) = K_{\mathcal{F}}((y,\xi),\cdot)$ and $\nabla_u u(\theta,y) = K_{\mathcal{U}}((\theta,y),\cdot)$ with $K_{\mathcal{F}}$ and $K_{\mathcal{U}}$ being the reproducing kernels of $\mathcal{F}$ and $\mathcal{U}$, respectively, while $\Pi_{\mathcal{F}}$ and $\Pi_{\mathcal{U}}$ denote the projections onto $\mathcal{F}$ and $\mathcal{U}$. When f and u are represented by neural networks, ∇_f and ∇_u are the gradients with respect to the coefficients representing those neural networks, which can be efficiently calculated through back propagation.

Algorithm 1 Predictive ABC

Input: Maximum number of iterations T. Prior distribution $\pi(\theta)$, model $p(y|\theta)$. Step sizes $\{\eta_k^u\}_{k=1}^T$ and $\{\eta_k^q\}_{k=1}^T$, samples $\{(\theta_i, y_i)\}_{i=1}^N$, objective function $\widehat{\Phi}_N$.
Initialize: q_1, u_1.
for $k = 1$ **to** T **do**
  Given $\{y_1, \ldots, y_N\}$, sample $\{\widetilde{\theta}_1, \ldots, \widetilde{\theta}_N\}$ from $q(\theta|y)$.
  Update q: $q_{k+1} \leftarrow \Pi_{\mathcal{P}}(q_k - \eta_k^q \cdot \nabla_q \widehat{\Phi}_N(q_k, u_k))$.
  Update u: $u_{k+1} \leftarrow \Pi_{\mathcal{U}}(u_k - \eta_k^u \cdot \nabla_u \widehat{\Phi}_N(q_k, u_k))$.
end for
Output: $\bar{q} = \frac{\sum_{k=1}^T \eta_k^q q_k}{\sum_{k=1}^T \eta_k^q}$, and $\bar{u} = \frac{\sum_{k=1}^T \eta_k^u u_k}{\sum_{k=1}^T \eta_k^u}$.

3.2 Theoretical Properties

Learning bounds. By invoking the tail inequality in Antos et al. [2008] and the ϵ-net argument, we have the following theorem, the proof of which can be found in Appendix A.

Theorem 1. Suppose $(\theta_i, y_i)_{i=1}^N$ is a β-mixing sequence [3] with $\beta_m \leq \bar{\beta}\exp(-bm^\kappa)$ for constants $\bar{\beta}$, b and κ, and suppose that function class $\mathcal{U} \times \mathcal{P}$ has a finite pseudo dimension D. [4] In addition, suppose that $u \in [-C_u, C_u]$ and the Fenchel dual satisfies $\nu^*(u) \leq C_\nu$. Then, with probability $1 - \delta$,

$$\epsilon_N \leq \sqrt{\frac{C_1(\max(C_1/b, 1))^{1/\kappa}}{C_2 N}},$$

where $C_1 = \log N^{\frac{D}{2}} e\delta^{-1} + [\log(2\max(16e(D+1)C_2^{\frac{D}{2}}, \bar{\beta}))]_+$ and $C_2 = (512(C_\nu + C_u)^2)^{-1}$.

Theorem 1 applies to all the formulations we introduced in Section 2, for which learning is consistent at a rate of $\mathcal{O}(N^{-1/2}\log N)$, with N being the number of samples, when the empirical saddle point approximation can be exactly solved. Below, we discuss the convergence of Algorithm 1.

Convergence of P-ABC. From a theoretical perspective, global convergence of first-order methods such as stochastic gradient descent (SGD) can be achieved when the objective function Φ is convex-concave. For example, when u and q are Gaussian mixtures or belong to RKHSs. More often than not, the objective function is not convex-concave, for which stochastic gradient descent (SGD) based algorithms are only guaranteed to converge towards a stationary point in certain restricted cases [Sinha et al., 2017, Li and Yuan, 2017, Kodali et al., 2018]. Below, we provide the convergence results for Algorithm 1 when $\widehat{\Phi}_N$ is convex-concave.

Consider the standard metric for evaluating the quality of any pair of estimates $\bar{q}$ and $\bar{u}$:

$$\varepsilon(\bar{q}, \bar{u}) = \max_{u \in \mathcal{U}} \widehat{\Phi}_N(\bar{q}, u) - \min_{q \in \mathcal{P}} \widehat{\Phi}_N(q, \bar{u}),$$

for which we have the following result (See Appendix C for proof).

Theorem 2 (Convergence of P-ABC)**.** Suppose that $\mathcal{P}$ and $\mathcal{U}$ are closed and bounded function spaces with diameters $D_{\mathcal{P}}$ and $D_{\mathcal{U}}$, respectively. Let $\widehat{\Phi}_N$ be convex-concave, L_N-Lipschitz, and suppose that there exists $q_{\min}$ such that $\inf_{\theta,y} q_N^*(\theta, y) \geq q_{\min}$. Then, for the outputs of Algorithm 1 with T iterates and whose step sizes satisfy $\eta_k^q = \eta_k^u = \eta_k$, denoted by $\bar{q}$ and $\bar{u}$, we have

$$\varepsilon(\bar{q}, \bar{u}) \leq \frac{D_{\mathcal{P}}^2 + D_{\mathcal{U}}^2 + \sum_{k=0}^T 2\eta_k^2 L_N^2}{2\sum_{k=0}^T \eta_k}.$$

Theorem 2 applies to the cases when $\mathcal{P}$ and $\mathcal{U}$ are spaces for Gaussian mixture coefficients or RKHSs, in which case $\widehat{\Phi}_N$ is convex-concave. It suggests that if there exist positive constants C_3 and C_4 such

[3] A discrete time stochastic process is mixing if widely separated events are asymptotically independent. Here, β_m provides an upper bound on the dependency of two events separated by n intervals of time. See Meir [2000] for a detailed definition.

[4] Pseudo dimension, also known as the Pollard dimension, is a generalization of VC dimension to the function class (see chapter 11 of Anthony and Bartlett [2009]).

that $C_3 k^{-1/2} \leq \eta_k \leq C_4 k^{-1/2}$, then $\lim_{T\to\infty} \varepsilon(\bar{q}, \bar{u}) = 0$. Together with Theorem 1, we know that the overall error, contributed by the summation of the learning error and the optimization error, can be bounded by $\mathcal{O}(N^{-1/2} \log N)$ upon selecting $T = \Theta(\exp(N))$.

4 Numerical Experiment

We test the performance of P-ABC under different objectives and representations, and compare the result with K2- and DR-ABC as representitives from sampling- and regression-based ABC algorithms.

4.1 Synthetic Dataset I: Superpoisition of Uniform Distributions

Consider a toy example where we observe a set of samples $Y^* := \{y_i^*\}_{i=1}^n$ where all y_i^*'s correspond to the same underlying parameter $\theta^* \in \mathbb{R}^p$. We assume the generating model is $y_i^* = \theta^* + u_i$ with $\{u_i\}_{i=1}^n$ being i.i.d. random vectors and each dimension of u_i as well as each dimension of θ^* is uniformly distributed on $[-0.5, 0.5]$. Denote the p coordinates of each observed sample y_i^* and θ^*, respectively, as $y_{i1}^*, \ldots, y_{ip}^*$ and $\theta_1^*, \ldots, \theta_p^*$, then the posterior can be written as

$$p(\theta^*|Y^*) \propto \prod_{j=1}^{p} \mathbb{1}\left[\max\{-0.5, \max_{i\in\{1,\ldots,n\}} y_{ij}^* - 0.5\} \leq \theta_j^* \leq \min\{0.5, \min_{i\in\{1,\ldots,n\}} y_{ij}^* + 0.5\}\right],$$

which is a uniform distribution whose boundary on the j-th dimension is defined by the values of the maximum and minimum values of the j-th coordinate among all y_i^*'s. Due to the fact that K2- and DR-ABC evaluate their performances using the mean square error, we use predictive ABC (P-ABC) to find the optimal minimum mean square error (MMSE) estimator for θ^*. We denote the optimal estimator by $\widehat{\theta}_{\text{opt}}$, which has a closed form solution with the j-th coordinate being

$$\widehat{\theta}_j = \begin{cases} \frac{1}{2} \cdot \max_{i\in\{1,\ldots,n\}} y_{ij}^*, & \min_{i\in\{1,\ldots,n\}} y_{ij}^* \geq 0 \\ \frac{1}{2} \cdot \min_{i\in\{1,\ldots,n\}} y_{ij}^*, & \max_{i\in\{1,\ldots,n\}} y_{ij}^* \leq 0 \\ \frac{1}{2}\left(\max_{i\in\{1,\ldots,n\}} y_{ij}^* + \min_{i\in\{1,\ldots,n\}} y_{ij}^*\right), & \min_{i\in\{1,\ldots,n\}} y_{ij}^* \leq 0 \leq \max_{i\in\{1,\ldots,n\}} y_{ij}^* \end{cases},$$

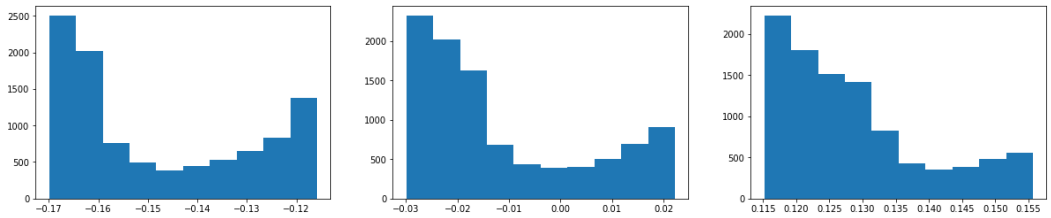
for all $j \in \{1, \ldots, p\}$. A sub-optimal estimator for this example is $\widehat{\theta}_{\text{ave}} = n^{-1} \sum_{i=1}^n (y_{i1}^*, \ldots, y_{ip}^*)$, which exploits the information that the expectation of the noise is a zero vector. We include these two closed-form estimators in our benchmarks in addition to K2- and DR-ABC.

The scalar case. We first examine the case when $\theta^* \in \mathbb{R}$ and $n = 1$, and compare the performance of P-ABC under different objectives and representations. The results are compared to that of the theoretically optimal estimator, for which $\widehat{\theta}_{\text{opt}} = y^*/2$.

We tested the performance of P-ABC, when the neural networks representing f and u in (4) were trained on 1000 samples. Each neural network contained two fully connected layers of size 8 with exponential linear unit (ELU) activation functions, and the final output layer for f was activated by the hyperbolic tangent. We chose ξ to be a one-dimensional uniform distribution on $[-1, 1]$ and used a learning rate of 10^{-4}. In 2×10^5 iterations, P-ABC achieved 0.0413 MSE on the training set and 0.0416 MSE on the test set. The optimal MSE corresponding to $\widehat{\theta}_{\text{opt}}$ was 0.0411 for the training set. Figure 1 shows the histogram of $f(y, \xi)$ for different values of y in $[-0.5, 0.5]$, using 10^4 trials of ξ. We see that the empirical probability distribution concentrates tightly around $y^*/2$, demonstrating that the output of P-ABC was nearly optimal. The training and testing errors are reported in Table 1, and the result shows that the performance of P-ABC was close to the theoretical optimum. By comparison, since there is only one observation available, K2- and DR-ABC do not output meaningful results as the computation of the MMD statistics requires at least two observations.

Performance under higher dimensions. We examined the performance of P-ABC when the dimension of θ^* was higher. For illustration purpose, we chose $\dim(\theta) = 16, 128, 256$, and we assumed that the set of observations, Y^*, contained 10 samples for each parameter value. Once again, we used neural networks to represent f and u in P-ABC, for which we trained with 1000 sets of samples.

To reduce the input dimension of the neural networks, for each input set of samples $Y := \{y_1, \ldots, y_n\}$, we set $f(Y, \xi) = \frac{1}{n}\sum_{i=1}^n f(y_i, \xi)$ and $u(\theta, Y) = \frac{1}{n}\sum_{i=1}^n u(\theta, y_i)$. More specifically, rather than taking the entire set of samples as the input, the neural network representing f took each sample individually, and used their average as the final value of $f(Y, \xi)$. Under this setting, for 2×10^5 iterations, the obtained results are shown in Table 1. We can see that P-ABC outperformed both K2- and DR-ABC in all four cases, and when the dimension of θ^* was small, the performance of P-ABC was close to that of $\widehat{\theta}_{\text{ave}}$.



$y^* = -0.25$ $y^* = 0$ $y^* = 0.25$

Figure 1: Histogram of $f(y, \xi)$ computed from 10^4 samples of $y \sim p(y)$ and $\xi \sim p_0(\xi)$. The distribution shows that the estimated θ concentrates closely around $\widehat{\theta}_{\text{opt}} = y^*/2$, suggesting that the P-ABC estimate is near optimal.

MSE	P-ABC [test,train]	K2-ABC	DR-ABC	$\widehat{\theta}_{\text{opt}}$	$\widehat{\theta}_{\text{ave}}$
$\dim(\theta^*) = 1$	[0.009,0.010]	0.011	0.083	0.003	0.008
$\dim(\theta^*) = 16$	[0.182, 0.155]	1.283	1.143	0.050	0.134
$\dim(\theta^*) = 128$	[2.749,1.793]	21.478	10.730	0.409	1.064
$\dim(\theta^*) = 256$	[4.266,1.399]	41.830	21.324	0.818	2.119

Table 1: MSE for estimating θ^* with different dimensions using K2-, DR- and P-ABC. For K2- and DR-ABC, we set $\epsilon = 0.01$ when computing MMD. For P-ABC, the hidden layer sizes are 8,32,128,256 for different values of $\dim(\theta)$, and the dimension of ξ's are 1,4,4,4, respectively.

4.2 Synthetic Dataset II: Gaussian Mixtures

Consider a model where the underlying parameter $\theta^* \in \mathbb{R}$ is uniformly distributed on $[-0.5, 0.5]$, and the observation y^* was sampled from a Gaussian mixture: $y^* = (0.5 + \theta^*)\mathcal{N}(-1, 1) + (0.5 - \theta^*)\mathcal{N}(1, 1)$. In this example, we compared the performances between K2-, DR-, EP-, and the proposed P-ABC. For P-ABC, we adopted the same network structures for the neural networks representing f and g as in the previous example, and trained them with 4000 sets of samples. Each set of samples contained 250 samples corresponding to the same θ^* and the same amount of samples were used for the benchmarks. P-ABC achieved an MSE of 0.004, and EP-ABC achiieved an MSE of 0.06. [5] Note that the implementation of EP-ABC requires Cholesky factorization for each iteration, which is computationally expensive and particularly sensitive to initialization. In fact, the run time of EP-ABC was significantly longer than P-ABC. While P-ABC took less than 5 minutes to average its performance over 1000 sets of test samples, EP-ABC took 10 minutes to average its performance over 100 sets of samples. K2- and DR-ABC, by comparison, were unable to run 100 trials within 1 hour. This experiment demonstrated the efficiency of implementing the P-ABC algorithm.

Although P-ABC has demonstrated superior numerical performances over the benchmarks, we point out that it suffers from some of the defficiencies of the other existing ABC algorithms. One such defficiency is that the algorithm is prone to mismatched priors. To see this, we plotted the histogram of $f(y, \xi)$ for $\theta^* = 0$ when P-ABC was trained on a mismatched prior. In particular, we applied the transformation $\widetilde{\theta} = (\theta + a)/(2a + 1)$, and used $\widetilde{\theta}$ as the sampled parameter from the prior. This transformation introduced bias between the true prior and the prior used for training, and as can be seen in Figure 2, the range of the estimated parameter by P-ABC shifted away from θ^* as a increased.

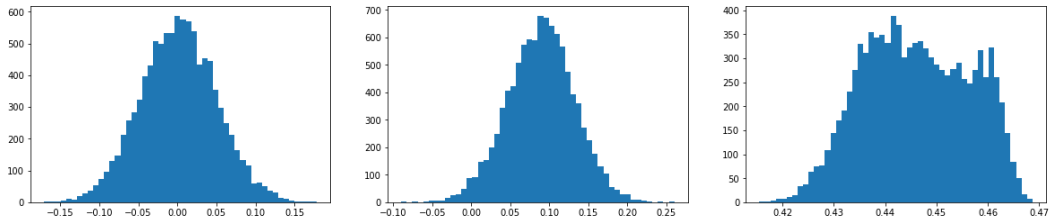
4.3 Ecological Dynamic System

Time series observations are an important application scenario for ABC. We compared the performances of K2-, DR- and P-ABC over the example of an ecological dynamic system studied in Park et al. [2016], whose population dynamics follow the relationship

$$y_{t+1} = P y_{t-\tau} \exp\left(-\frac{y_{t-\tau}}{y_0}\right) e_t + y_t \exp(-\delta \epsilon_t).$$

Let $Y = (y_1, \ldots, y_t)$ denote the set of samples that contains the population size data up to time t. The noise $e_t \sim \Gamma(\sigma_p^{-2}, \sigma_p^2)$, $\epsilon_t \sim \Gamma(\sigma_d^{-2}, \sigma_d^2)$, while $\theta = (P, y_0, \sigma_d^2, \sigma_p^2, \tau, \delta)$. Similar to Park et al.

[5] Per implementation of the code made available online by Barthelmé and Chopin [2011].



$\theta^* = 0, a = 0$ $\theta^* = 0, a = 1$ $\theta^* = 0, a = 10$

Figure 2: Impact of improper prior on P-ABC. Consider finding uniformly distributed $\theta \sim \mathcal{U}[-0.5, 0.5]$ from $y = (0.5 + \theta)\mathcal{N}(-1, 1) + (0.5 - \theta)\mathcal{N}(-1, 1)$. Improper priors are obtained by $\widetilde{\theta} = (\theta + a)/(2a + 1)$ with $a = 1, 10$. We see that training on improper prior injects bias into the output of P-ABC.

[2016], we sample each dimension of $\log \theta$ from a uniform distribution on $[-5, 2]$, and set $\tau = \lceil \tau \rceil$. For P-ABC, we implemented a recurrent neural network (RNN) with LSTM cells to capture the dynamics of the underlying time series. The output of the LSTM cell is then plugged into a fully connected layer along with θ or ξ. The structures of the neural networks representing f and u are shown in Figure 4 in Appendix D. When training, we set the size of each sample set Y to 30, and we used 1000 sets of samples to train the algorithms. For P-ABC, we set $\dim(\xi) = 4$, the size of the LSTM cell to 32 and the size of the fully connected layer to 16. For K2- and DR-ABC, the samples within each set Y were regarded as i.i.d.. The obtained result is shown in Figure 3, with the verticle axis denoting the MSE of the estimated parameter. P-ABC outperformed K2-ABC and DR-ABC on all aspects: the MSE was 12.9 for P-ABC, 24.7 for K2-ABC, and 16.4 for DR-ABC. In addition, P-ABC had the lowest average, quartile, and better performance on outliers.

5 Conclusion

In this paper, we presented a unifying optimization framework for ABC, named Predictive-ABC, under which we showed that ABC can be formulated as a saddle point problem for different objective functions. We presented a high-probability error bound that decays at the speed of $\mathcal{O}(N^{-1/2} \log N)$ with N being the number of samples and we presented a stochastic-gradient-descent-based algorithm, P-ABC, to find the solution. In practice, P-ABC significantly outperforms K2- and DR-ABC, representatives for the state-of-the-art sampling- and regression-based algorithms, respectively.

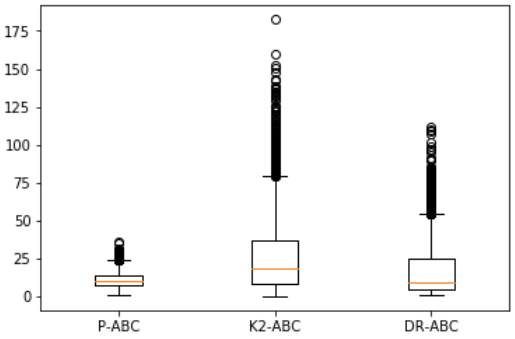


Figure 3: Statistics of MSEs for P-, K2- and DR-ABC trained on 1000 sequences of length 30.

Acknowledgement

This work was supported in part by MURI grant ARMY W911NF-15-1-0479, ONR grant W911NF-15-1-0479, NSF CCF-1755829 and NSF CMMI-1761699.

References

Martin Anthony and Peter L Bartlett. *Neural network learning: Theoretical foundations*. cambridge university press, 2009.

András Antos, Csaba Szepesvári, and Rémi Munos. Learning near-optimal policies with Bellman-residual minimization based fitted policy iteration and a single sample path. *Machine Learning*, 71 (1):89–129, 2008.

Simon Barthelmé and Nicolas Chopin. ABC-EP: Expectation propagation for likelihoodfree Bayesian computation. In *ICML*, pages 289–296, 2011.

Peter L Bartlett, Nick Harvey, Chris Liaw, and Abbas Mehrabian. Nearly-tight VC-dimension and pseudodimension bounds for piecewise linear neural networks. *arXiv preprint arXiv:1703.02930*, 2017.

Espen Bernton, Pierre E Jacob, Mathieu Gerber, and Christian P Robert. Inference in generative models using the Wasserstein distance. *arXiv preprint arXiv:1701.05146*, 2017.

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

Michael GB Blum, Maria Antonieta Nunes, Dennis Prangle, Scott A Sisson, et al. A comparative review of dimension reduction methods in approximate Bayesian computation. *Statistical Science*, 28(2):189–208, 2013.

Katalin Csilléry, Olivier François, and Michael G B Blum. Abc: An R package for approximate Bayesian computation (ABC). *Methods in Ecology and Evolution*, 3(3):475–479, 2012. ISSN 2041210X. doi: 10.1111/j.2041-210X.2011.00179.x.

Bo Dai, Niao He, Yunpeng Pan, Byron Boots, and Le Song. Learning from conditional distributions via dual embeddings. In *Artificial Intelligence and Statistics*, pages 1458–1467, 2017.

C. C. Drovandi and A. N. Pettitt. Estimation of parameters for macroparasite population evolution using approximate Bayesian computation. *Biometrics*, 67(1):225–233, 2011. ISSN 0006341X. doi: 10.1111/j.1541-0420.2010.01410.x.

Alexei J Drummond and Andrew Rambaut. Beast: Bayesian evolutionary analysis by sampling trees. *BMC evolutionary biology*, 7(1):214, 2007.

Christoph Leuenberger Daniel Wegmann Laurent Excoffier. Bayesian computation and model selection in population genetics. *Genetics*, page 18, 2009. doi: 10.1534/genetics.109.102509. URL `http://arxiv.org/abs/0901.2231`.

Alexander Gleim and Christian Pigorsch. Approximate Bayesian computation with indirect summary statistics.

Michael U Gutmann, Ritabrata Dutta, Samuel Kaski, and Jukka Corander. Statistical inference of intractable generative models via classification. *arXiv preprint arXiv:1407.4981*, 2014.

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

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

David Haussler. Sphere packing numbers for subsets of the boolean n-cube with bounded Vapnik-Chervonenkis dimension. *Journal of Combinatorial Theory, Series A*, 69(2):217–232, 1995.

John P Huelsenbeck, Fredrik Ronquist, Rasmus Nielsen, and Jonathan P Bollback. Bayesian inference of phylogeny and its impact on evolutionary biology. *science*, 294(5550):2310–2314, 2001.

Paul Joyce and Paul Marjoram. Approximately sufficient statistics and Bayesian computation. *Statistical applications in genetics and molecular biology*, 7(1), 2008.

Diederik P Kingma and Max Welling. Auto-encoding variational Bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Naveen Kodali, James Hays, Jacob Abernethy, and Zsolt Kira. On convergence and stability of GANs. 2018.

T Kulkarni, Ilker Yildirim, Pushmeet Kohli, W Freiwald, and Joshua B Tenenbaum. Deep generative vision as approximate Bayesian computation. In *NIPS 2014 ABC Workshop*, 2014.

Jingjing Li, David J Nott, Yanan Fan, and Scott A Sisson. Extending approximate Bayesian computation methods to high dimensions via a Gaussian copula model. *Computational Statistics & Data Analysis*, 106:77–89, 2017.

Yuanzhi Li and Yang Yuan. Convergence analysis of two-layer neural networks with ReLU activation. In *Advances in Neural Information Processing Systems*, pages 597–607, 2017.

Gael M. Martin, Brendan P. M. McCabe, Worapree Maneesoonthorn, and Christian P. Robert. Approximate Bayesian computation in state space models. *arXiv:1409.8363*, pages 1–38, 2014. URL `http://arxiv.org/abs/1409.8363`.

Ron Meir. Nonparametric time series prediction through adaptive model selection. *Machine learning*, 39(1):5–34, 2000.

Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. Adversarial variational Bayes: Unifying variational autoencoders and generative adversarial networks. *arXiv preprint arXiv:1701.04722*, 2017.

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

Jovana Mitrovic, Dino Sejdinovic, and Yee Whye Teh. DR-ABC: Approximate Bayesian computation with kernel-based distribution regression. 2016.

Shakir Mohamed and Balaji Lakshminarayanan. Learning in implicit generative models. *arXiv preprint arXiv:1610.03483*, 2016.

David J Nott, Y Fan, L Marshall, and SA Sisson. Approximate Bayesian computation and Bayes' linear analysis: toward high-dimensional ABC. *Journal of Computational and Graphical Statistics*, 23(1):65–86, 2014.

Matthew A Nunes and David J Balding. On optimal selection of summary statistics for approximate Bayesian computation. *Statistical applications in genetics and molecular biology*, 9(1), 2010.

Mijung Park, Wittawat Jitkrittum, and Dino Sejdinovic. K2-ABC: Approximate Bayesian computation with kernel embeddings. 2016.

Leah F Price, Christopher C Drovandi, Anthony Lee, and David J Nott. Bayesian synthetic likelihood. *Journal of Computational and Graphical Statistics*, 27(1):1–11, 2018.

Louis Raynal, Jean-Michel Marin, Pierre Pudlo, Mathieu Ribatet, Christian P Robert, and Arnaud Estoup. ABC random forests for Bayesian parameter inference. *arXiv preprint arXiv:1605.05537*, 2016.

GS Rodrigues, Dennis Prangle, and Scott A Sisson. Recalibration: A post-processing method for approximate Bayesian computation. *Computational Statistics & Data Analysis*, 126:53–66, 2018.

Aman Sinha, Hongseok Namkoong, and John Duchi. Certifiable distributional robustness with principled adversarial training. *arXiv preprint arXiv:1710.10571*, 2017.

Dustin Tran, Rajesh Ranganath, and David M Blei. Hierarchical implicit models and likelihood-free variational inference. *arXiv preprint arXiv:1702.08896*, 2017.

Daniel Wegmann, Christoph Leuenberger, and Laurent Excoffier. Efficient approximate Bayesian computation coupled with Markov chain Monte Carlo without likelihood. *Genetics*, 182(4): 1207–1218, 2009.

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

Arnold Zellner. Optimal information processing and Bayes's theorem. *The American Statistician*, 42 (4):278–280, 1988.

Appendix

A Proof of Theorem 1

In this proof, we find the concentration bound for the statistical error for (4), which is defined as

$$\epsilon_N = D_\nu(p(y|\theta)\pi(\theta), q_N^*(\theta|y)p(y)) - D_\nu(p(y|\theta)\pi(\theta), q^*(\theta|y)p(y)),$$

where $q_N^*(\theta|y)$ is the distribution that optimizes (9) (for transportation reparametrization $q_N^*(\theta|y)$ is the distribution induced by the optimal empirical solution f_N^* with N samples in (4)), and $q^*(\theta|y)$ is the solution to (4). Let $\widehat{u} := \text{argmax}_{u\in\mathcal{U}}\, \Phi(f_N^*, u)$, we can bound the learning error ϵ_N by

$$\begin{aligned}\epsilon_N &= \max_{u\in\mathcal{U}} \Phi(q_N^*, u) - \max_{u\in\mathcal{U}} \Phi(q^*, u) = \Phi(q_N^*, \widehat{u}) - \Phi(q^*, u^*) \\ &= \Phi(q_N^*, \widehat{u}) - \Phi(q^*, \widehat{u}) + \Phi(q^*, \widehat{u}) - \Phi(q^*, u^*) \\ &\leq \Phi(q_N^*, \widehat{u}) - \Phi(q^*, \widehat{u}) \leq 2 \sup_{q\in\mathcal{P}, u\in\mathcal{U}} |\widehat{\Phi}_N(q, u) - \Phi(q, u)|.\end{aligned}$$

In the following, we provide a high probability upper bound for $\sup_{q,u} |\widehat{\Phi}_N(q, u) - \Phi(q, u)|$.

A.1 Technical Lemmas

The concentration bound requires Lemma 5 in Antos et al. [2008]:

Lemma 1 (Lemma 5 [Antos et al., 2008])**.** Suppose that $Z_1, \ldots, Z_N \in \mathcal{Z}$ is a sequence that is stationary and β-mixing, and $\mathcal{G}$ is a class of bounded functions, then

$$\mathbb{P}\left(\sup_{g\in\mathcal{G}} \left|\frac{1}{N}\sum_{i=1}^{N} g(Z_i) - \mathbb{E}[g(Z_1)]\right| > \varepsilon\right) \leq 16\mathbb{E}\left[\mathcal{N}\left(\varepsilon/8, \mathcal{G}, (Z_i'; i \in H)\right)\right] \exp\left(\frac{-N\varepsilon^2}{256C^2}\right) + \\ + 2m_N\beta_{k_N+1}$$

where Z_i' is the "ghost" sample that mirrors Z_i, $\mathcal{N}(\epsilon/8, \mathcal{G}, (Z_i'; i \in H))$ is the covering number for $\mathcal{G}$, and $H = \cup_{j=1}^{m_N} H_i$ is the union of blocks in the sampling path.

The covering number in the above lemma can be bounded using the following result:

Lemma 2 (Corollary 3 [Haussler, 1995])**.** For any set $\mathcal{X}$, and $x_1, \ldots, x_n \in \mathcal{X}$, assume $\mathcal{F}$ of functions on $\mathcal{X}$ are bounded within $[0, C]$ with pseudo-dimension $D_\mathcal{F} < \infty$. Then, for any ε,

$$\mathcal{N}(\varepsilon, \mathcal{F}, (x_1, \ldots, x_n)) \leq e(D_\mathcal{F} + 1)\left(\frac{2eC}{\varepsilon}\right)^{D_\mathcal{F}},$$

where e is the Euler's number.

With the above two lemmas, we are ready to prove the theorem.

A.2 Learning Error

For any sample (θ, y), let $\phi(q, u) = u(\theta, y) - \mathbb{E}_{\theta\sim q(\theta|y)}[\nu^*(u(\theta, y))|y]$. Then, the objective function in (4) can be written as $\Phi(q, u) = \mathbb{E}[\phi(q, u)]$, where the expectation is with respect to the joint distribution of $(\theta, y) \sim p(y|\theta)\pi(\theta)$.

By assumption, since $|u(\theta, y)| \leq C_u$, [6] the Fenchel dual in (4) can be bounded. In particular, we denote the upper bound as $\nu^*(u) = \sup_{u'\in[-C_u, C_u]}\langle u', u\rangle - \nu(u') \leq C_\nu$, [7] and hence

$$\phi(q, u) \leq C_\nu + C_u.$$

[6] Imagine Gaussian mixture representation with bounded coefficients, or neural networks whose last layer of activation is a bounded function, such as the hyperbolic tangent or the sigmoid function, amplified by a constant C_u such that $|u(\theta, y)| \leq C_u$.

[7] For example, for χ^2-divergence, we have $C_\nu = C_u^2 + C_u$.

Invoking Lemma 1, in which we let $\mathcal{G} = \mathcal{P} \times \mathcal{U}$, we have

$$\mathbb{P}\left(\sup_{(q,u)\in\mathcal{P}\times\mathcal{U}}\left|\widehat{\Phi}_N(q,u) - \Phi(q,u)\right| > \epsilon\right) \leq 2m_N\beta_{k_N+1} + \\ + 16\mathbb{E}\left[\mathcal{N}\left(\frac{\epsilon}{8}, \mathcal{P}\times\mathcal{U}, ((\theta_i, y_i), i \in H)\right)\right]\exp\left(\frac{-m_N\epsilon^2}{128(C_\nu + C_u)^2}\right).$$

Next, invoking Lemma 2 and by assumption, [8] we have

$$\mathbb{P}\left(\sup_{(q,u)\in\mathcal{P}\times\mathcal{U}}\left|\widehat{\Phi}_N(q,u) - \Phi(q,u)\right| > \epsilon\right) \leq 16e(D+1)\left(\frac{4eC}{\epsilon}\right)^D \exp\left(\frac{-m_N\epsilon^2}{128(C_\nu + C_u)^2}\right) + \\ + 2m_N\beta_{k_N+1}.$$

Lastly, by setting $k_N = (N\epsilon^2 C_2/b)^{1/(\kappa+1)}$, with $C_2 = 1/(512(C_\nu + C_u)^2)$ and $m_N = N/2k_N$, we have the right-hand side of the above equation bounded by δ with

$$\epsilon = \sqrt{\frac{C_1(\max(C_1/b, 1))^{1/\kappa}}{C_2 N}},$$

and $C_1 = 0.5D\log N + \log(e/\delta) + [\log(2\max(16e(D+1)C_2^{D/2}, \bar{\beta}))]_+$, with $\bar{\beta}$ is such that $\beta_m \leq \bar{\beta}\exp(-bm^\kappa)$.

B Derivation of objective functions

Divergence	Saddle point objective
χ^2 divergence	$\mathbb{E}_{(\theta,y)\sim p(y\|\theta)\pi(\theta)}[u(\theta,y)] - \mathbb{E}_{\theta\sim f(y,\xi),\xi\sim p_0(\xi),y\sim p(y)}[u(\theta,y) + u^2(\theta,y)/4]$
Wasserstein distance	$\mathbb{E}_{(\theta,y)\sim p(y\|\theta)\pi(\theta)}[u(\theta,y)] - \mathbb{E}_{\theta\sim f(y,\xi),\xi\sim p_0(\xi),y\sim p(y)}[u(\theta,y)]$
KL divergence	$\mathbb{E}_{(\theta,y)\sim p(y\|\theta)\pi(\theta)}[u(\theta,y)] - \mathbb{E}_{\theta\sim f(y,\xi),\xi\sim p_0(\xi),y\sim p(y)}[1 + \log(u(\theta,y))]$

Table 2: A list of divergences and their corresponding saddle point objective.

In this section, we present the detailed derivation of the objective functions in Section 2 and give examples for deriving (4) for several choices of f-divergences in Table 2.

Posterior Distribution Matching For the posterior distribution matching objective, we want to minimize

$$\min_{f\in\mathcal{F}}\max_{h\in\mathcal{H}}\mathbb{E}_y\left[\left(\mathbb{E}_{\theta|y}[h(\theta)] - \mathbb{E}_{\theta\sim p_0(\xi)}[h(f(y,\xi))]\right)^2\right].$$

By exploiting the dual embedding technique and the Fenchel duality, we have

$$\begin{aligned}
&\max_{h\in\mathcal{H}}\mathbb{E}_y\left[\left(\mathbb{E}_{\theta|y}\left[h(\theta)\right] - \mathbb{E}_{\theta\sim p_0(\xi)}\left[h(f\left(y,\xi\right))\right]\right)^2\right] \\
=\ &\max_{h\in\mathcal{H}}\mathbb{E}_y\left[\max_{v_y\in\mathbb{R}} v_y\left(\mathbb{E}_{\theta|y}\left[h(\theta)\right] - \mathbb{E}_{\xi\sim p_0(\xi)}\left[h(f\left(y,\xi\right))\right] - \frac{1}{2}v_y^2\right)\right] \\
=\ &\max_{h\in\mathcal{H}}\max_{v\in\mathcal{V}}\mathbb{E}_y\left[v\left(y\right)\cdot\left(\mathbb{E}_{\theta|y}\left[h(\theta)\right] - \mathbb{E}_{\xi\sim p(\xi)}\left[h(f(y,\xi))\right]\right) - \frac{1}{2}v^2(y)\right],
\end{aligned}$$

thus achieving the equivalence between (5) and (6).

Joint Distribution Matching

- χ^2**-divergence.** Recall that the divergence minimization objective (3) can be written in the saddle point formulation:

$$\min_{f\in\mathcal{F}}\max_{u\in\mathcal{U}}\mathbb{E}_{(\theta,y)\sim p(y|\theta)\pi(\theta)}\left[u(\theta,y)\right] - \mathbb{E}_{\theta\sim f(y,\xi),\xi\sim p_0(\xi),y\sim p(y)}\left[\nu^*\left(u(\theta,y)\right)\right].$$

[8] For transportation reparametrization with neural networks, it is known that when the neural network is feed forward and piecewise linear, there exists upper bounds for its pseudo dimension [Bartlett et al., 2017].

For χ^2 divergence, we have

$$\nu(x) = (x-1)^2.$$

Therefore,

$$\nu^*(x) = \sup_y(\langle x, y\rangle - \nu(y)) = \sup_y(xy - (y-1)^2) = \frac{x^2+4x+8}{4}.$$

Plugging in $x = u(\theta, y)$ gives the result of $\nu^*(u(\theta, y))$, and the expression for Φ follows immediately.

$$\nu^*(x) = \sup_y(\langle x, y\rangle - \nu(y)) = \sup_y(\langle x, y\rangle - (y-1)^2) = x - \frac{x^2}{4}.$$

Plugging in $x = u(\theta, y)$ gives the expression of $\nu^*(u(\theta, y)$, and hence we arrive at the conclusion.

- **KL divergence.** For KL divergence, $\nu(x) = \log x$. We thus have

$$\nu^*(x) = \sup_y(\langle x, y\rangle - \log y) = 1 + \log x.$$

Therefore, letting $x = u(\theta, y)$ gives the expression of $\nu^*(u(\theta, y))$, and hence we arrive at the conclusion.

C Proof of Theorem 2

Following the notations in Algorithm 1, we denote the updates of q and u at iteration k by q_k and u_k, respectively. Different representations of q and u will determine the detailed forms of q_k and u_k. For example, when represented by Gaussian mixtures, both u_k and q_k in Algorithm 1 will be a coefficient vector. However, for the purpose of proving convergence, using an abstract form u_k and q_k is sufficient barring that $\mathcal{U}$ and $\mathcal{P}$ are closed and bounded.

By convex-concavity of the empirical loss function, we have

$$\widehat{\Phi}_N(q_k, u_k) - \widehat{\Phi}_N(q, u_k) \leq \langle \nabla_q \widehat{\Phi}_N(q_k, u_k), q_k - q\rangle,$$

and

$$\widehat{\Phi}_N(q_k, u) - \widehat{\Phi}_N(q_k, u_k) \leq \langle \nabla_u \widehat{\Phi}_N(q_k, u_k), u - u_k\rangle$$

for any $q \in \mathcal{P}$ and $u \in \mathcal{U}$. Combining these two inequalities gives

$$\widehat{\Phi}_N(q_k, u) - \widehat{\Phi}_N(q, u_k) \leq \langle \nabla_q \widehat{\Phi}_N(q_k, u_k), q_k - q\rangle + \langle \nabla_u \widehat{\Phi}_N(q_k, u_k), u - u_k\rangle.$$

It is worth clarifying, at this point, that the gradient symbol we used for $\widehat{\Phi}_N$ so far refer to the actual gradient rather than the stochastic gradients given in Section 3.1. However, they are closely related by the fact that the expectation of the stochastic gradient is the gradient. To avoid confusion, we use $\widehat{\nabla}_q \widehat{\Phi}_N$ and $\widehat{\nabla}_u \widehat{\Phi}_N$ to represent the stochastic gradients, for which we have

$$\mathbb{E}\left[\widehat{\nabla}_u \widehat{\Phi}_N(q, u)\right] = \nabla_u \widehat{\Phi}_N(q, u) \qquad \text{and} \qquad \mathbb{E}\left[\widehat{\nabla}_q \widehat{\Phi}_N(q, u)\right] = \nabla_q \widehat{\Phi}_N(q, u),$$

where the expectation is taken over the the second term in (9), where we have used $\theta \sim q(\theta|y)$ to derive the stochastic gradients.

By convexity of $\widehat{\Phi}_N$, we have

$$\begin{aligned}
\varepsilon(\bar{q}, \bar{u}) &= \max_{u\in\mathcal{U}} \widehat{\Phi}_N(\bar{q}, u) - \min_{q\in\mathcal{P}} \widehat{\Phi}_N(q, \bar{u}) \\
&= \max_{u\in\mathcal{U}} \widehat{\Phi}_N\left(\frac{\sum_{k=1}^T \eta_k q_k}{\sum_{k=1}^T \eta_k}, u\right) - \min_{q\in\mathcal{P}} \widehat{\Phi}_N\left(q, \frac{\sum_{k=1}^T \eta_k u_k}{\sum_{k=1}^T \eta_k}\right) \\
&\leq \max_{u\in\mathcal{U}} \frac{\sum_{k=1}^T \eta_k \widehat{\Phi}_N(q_k, u)}{\sum_{k=1}^T \eta_k} - \min_{q\in\mathcal{P}} \frac{\sum_{k=1}^T \eta_k \widehat{\Phi}_N(q, u_k)}{\sum_{k=1}^N \eta_k} \\
&\leq \frac{\max_{u\in\mathcal{U}, q\in\mathcal{P}} \left\{\sum_{k=0}^T \left(\eta_k \langle \nabla_q \widehat{\Phi}_N(q_k, u_k), q_k - q\rangle - \eta_k \langle \nabla_u \widehat{\Phi}_N(q_k, u_k), u_k - u\rangle\right)\right\}}{\sum_{k=1}^T \eta_k}.
\end{aligned}$$

We now prove that the numerator is upper bounded by the numerator of the right-hand side of the bound in the statement of Theorem 2, which will bring us to the conclusion.

To prove this, we first note that, by the contractivity of the projection operator, we have

$$\begin{aligned}\mathbb{E}\|q_{k+1}-q\|_{\mathcal{P}}^2 &= \mathbb{E}\left\|\Pi_{\mathcal{P}}(q_k-\eta_k\widehat{\nabla}_q\widehat{\Phi}_N(q_k,u_k))-\Pi_{\mathcal{P}}(q)\right\|_{\mathcal{P}}^2\\ &\le \mathbb{E}\|q_k-q\|_{\mathcal{P}}^2+\mathbb{E}\left\|\eta_k\widehat{\nabla}_q\widehat{\Phi}_N(q_k,u_k)\right\|_{\mathcal{P}}^2-2\left\langle q_k-q,\eta_k\nabla_q\widehat{\Phi}_N(q_k,u_k)\right\rangle,\end{aligned}$$

which implies

$$2\left\langle q_k-q,\eta_k\nabla_q\widehat{\Phi}_N(q_k,u_k)\right\rangle \le \mathbb{E}\|q_k-q\|_{\mathcal{P}}^2+\mathbb{E}\left\|\eta_k\widehat{\nabla}_q\widehat{\Phi}_N(q_k,u_k)\right\|_{\mathcal{P}}^2-\mathbb{E}\|q_{k+1}-q\|_{\mathcal{P}}^2.$$

Similarly, we have

$$-2\left\langle u_k-u,\eta_k\nabla_u\widehat{\Phi}_N(q_k,u_k)\right\rangle \le \mathbb{E}\|u_k-u\|_{\mathcal{U}}^2+\mathbb{E}\left\|\eta_k\widehat{\nabla}_u\widehat{\Phi}_N(q_k,u_k)\right\|_{\mathcal{U}}^2-\mathbb{E}\|u_{k+1}-u\|_{\mathcal{U}}^2.$$

By the Lipschitz assumption, we have

$$\left\|\eta_k\widehat{\nabla}_q\widehat{\Phi}_N(q_k,u_k)\right\|_{\mathcal{P}}^2\le(\eta_k)^2L_N^2,\qquad\text{and}\qquad\left\|\eta_k\widehat{\nabla}_u\widehat{\Phi}_N(q_k,u_k)\right\|_{\mathcal{U}}^2\le(\eta_k)^2L_N^2.$$

Therefore,

$$\begin{aligned}&2\left\langle q_k-q,\eta_k\nabla_q\widehat{\Phi}_N(q_k,u_k)\right\rangle-2\left\langle u_k-u,\eta_k\nabla_u\widehat{\Phi}_N(q_k,u_k)\right\rangle\\ &\le\mathbb{E}\left(\|q_k-q\|_{\mathcal{P}}^2-\|q_{k+1}-q\|_{\mathcal{P}}^2+\|u_k-u\|_{\mathcal{U}}^2-\|u_{k+1}-u\|_{\mathcal{U}}^2\right)+\left[(\eta_k)^2+(\eta_k)^2\right]L_N^2.\end{aligned}$$

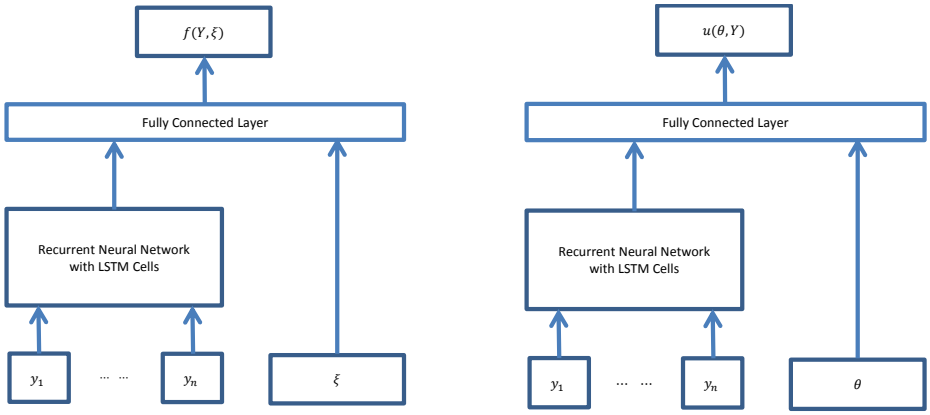
Lastly, by telescoping, we have

$$\begin{aligned}&\sum_{k=0}^{T}\left(2\left\langle q_k-q,\eta_k\nabla_q\widehat{\Phi}_N(q_k,u_k)\right\rangle-2\left\langle u_k-u,\eta_k\nabla_u\widehat{\Phi}_N(q_k,u_k)\right\rangle\right)\\ &\le\mathbb{E}\left(\|q_0-q\|_{\mathcal{P}}^2+\|u_0-u\|_{\mathcal{U}}^2-\|q_{T+1}-q\|_{\mathcal{P}}^2-\|u_{T+1}-u\|_{\mathcal{U}}^2\right)+\sum_{k=0}^{T}\left[(\eta_k)^2+(\eta_k)^2\right]L_N^2\\ &\le\mathbb{E}\left(\|q_0-q\|_{\mathcal{P}}^2+\|u_0-u\|_{\mathcal{U}}^2\right)+\sum_{k=0}^{T}\left[(\eta_k)^2+(\eta_k)^2\right]L_N^2\\ &\le D_{\mathcal{P}}^2+D_{\mathcal{U}}^2+\sum_{k=0}^{T}2\eta_k^2L_N^2.\end{aligned}$$

Hence we reached the conslusion.

D Neural Network Architecture for Biological Dynamic System Experiment

Below, we describe the structure for the neural network we used in the simulation of the ecological dynamical system. The network structure is shown in Figure 4. On the left-hand side, we have the network structure for $f(Y,\xi)$, for which we first input $y_1,\ldots,y_n$ into an RNN comprised of LSTM cells, and then use its output combined with ξ as the input for a fully connected layer. On the right-hand side, the network structure for $u(\theta,Y)$ is similar, except that we replace ξ with θ and change the dimension of the fully connected layer accordingly. When θ is generated from $f(Y,\xi)$, we concatenate the two neural networks, using the output of f as part of the input to u.



Generator Discriminator

Figure 4: Network architectures for u and f.