

---

# Automatic Variational ABC

---

Alexander Moreno<sup>\*1</sup>, Tameem Adel<sup>2,3</sup>, Edward Meeds<sup>2,4</sup>, James M. Rehg<sup>1</sup> and Max Welling<sup>2,5</sup>

<sup>1</sup>College of Computing, Georgia Institute of Technology

<sup>2</sup>Machine Learning Group, University of Amsterdam

<sup>3</sup>Data Science, Radboud University

<sup>4</sup>Center for Integrative Bioinformatics IBIVU, VU University

<sup>5</sup>Canadian Institute for Advanced Research (CIFAR)

## Abstract

Approximate Bayesian Computation (ABC) is a framework for performing likelihood-free posterior inference for simulation models. Stochastic Variational inference (SVI) is an appealing alternative to the inefficient sampling approaches commonly used in ABC. However, SVI is highly sensitive to the variance of the gradient estimators, and this problem is exacerbated by approximating the likelihood. We draw upon recent advances in variance reduction for SVI [6][13] and likelihood-free inference using deterministic simulations [12] to produce low variance gradient estimators of the variational lower-bound. By then exploiting automatic differentiation libraries [8] we can avoid nearly all model-specific derivations. We demonstrate performance on three problems and compare to existing SVI algorithms. Our results demonstrate the correctness and efficiency of our algorithm.

## 1 Introduction

In many areas of science complex simulators are used as models of the underlying phenomena of interest. For example, in computational biology, models could be simulations of embryonic morphogenesis or cancer development; in environmental science, they could model earthquakes or climate change. Further examples of simulation-based science can be found in computational chemistry, physics, and neuroscience. The most fundamental ingredient and computational bottleneck is the ability to match a (simulation) model to given observations. Bayesian methods provide an elegant solution to this problem where posterior distributions provide insightful information about the correlations and sensitivities of the parameters. However, current methods are based on sampling, such as MCMC, which tends to converge slowly and requires many calls to the simulator. In this work we introduce a variational Bayesian alternative.

In Bayesian inference, one wants to infer the posterior distribution  $p(\boldsymbol{\theta}|\mathbf{y})$  of some parameters  $\boldsymbol{\theta}$ , given observations  $\mathbf{y}$ .

$$p(\boldsymbol{\theta}|\mathbf{y}) = \frac{p(\mathbf{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{\int p(\mathbf{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta}} \quad (1)$$

where  $p(\mathbf{y}|\boldsymbol{\theta})$  is the likelihood,  $p(\boldsymbol{\theta})$  is a prior, and  $\int p(\mathbf{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta}$  is the normalizing constant. The normalizing constant is often intractable, so that sampling methods or variational inference should be used. However, both assume a tractable expression for the likelihood. For many problems, the likelihood is intractable or extremely expensive to compute. Approximate Bayesian Computation (ABC) deals with how to do Bayesian inference by approximating the likelihood using simulator outputs. The simulator generates synthetic data  $\boldsymbol{x}$  according to the parameters  $\boldsymbol{\theta}$ , i.e. we can simulate

---

\*corresponding author, alexander.f.moreno@gmail.com

$\mathbf{x} \sim p(\mathbf{x}|\boldsymbol{\theta})$ . These are compared to the true data via an  $\epsilon$ -kernel or ABC-kernel  $K_\epsilon(y, x)$ , which is an approximation to  $p(\mathbf{y}|\mathbf{x})$  and is a density that measures the discrepancy between  $\mathbf{x}$  and  $\mathbf{y}$ , where  $\epsilon$  is the bandwidth. Often,  $K_\epsilon(\mathbf{y}, \mathbf{x}) = \mathcal{N}(\mathbf{y}|\mathbf{x}, \epsilon^2)$ , but one may take other distributions. The likelihood,  $p(\mathbf{y}|\boldsymbol{\theta})$ , is approximated by the ABC likelihood

$$p_\epsilon(\mathbf{y}|\boldsymbol{\theta}) = \int K_\epsilon(\mathbf{y}, \mathbf{x})p(\mathbf{x}|\boldsymbol{\theta}) dx \quad (2)$$

$$\approx \frac{1}{S} \sum_{s=1}^S K_\epsilon(\mathbf{y}, \mathbf{x}^{(s)}) \quad (3)$$

The bias of  $p_\epsilon(\mathbf{y}|\boldsymbol{\theta})$  goes to 0 as  $\epsilon \rightarrow 0$ . With large and/or high-dimensional observations, it may be beneficial to use summary statistics  $S(\mathbf{y})$  and  $S(\mathbf{x})$  instead of the entire data set, i.e.  $K_\epsilon(S(\mathbf{y}), S(\mathbf{x}))$  instead of  $K_\epsilon(\mathbf{y}, \mathbf{x})$ .  $\epsilon$  introduces a trade-off between bias with a large  $\epsilon$  and variance with a small  $\epsilon$ . We will assume that  $\mathbf{y}$  and  $\mathbf{x}$  will, depending on the context, represent either raw data or statistics.

Sampling methods are widely used in the likelihood-free and ABC literature: rejection sampling [17], Markov Chain Monte Carlo (MCMC) via a modified Metropolis Hastings algorithm [9], and population-based sampling [2, 3, 16]. However, these have slow mixing rates and make many calls to the simulator, making them ineffective for large-scale problems, although some recent methods have been proposed to improve this, including [11][21]. Stochastic variational inference (SVI) typically has a faster rate of convergence but might still suffer from a high variance in the estimated gradients, as we will see.

## 2 Related Work

[6] [14] showed that for certain classes of continuous latent variables and variational posteriors, by reparametrizing the parameters or latent variables to be estimated, one can obtain a lower variance Monte Carlo approximation to the gradient of the lower bound. [18] found a more general variance reduction method that is conceptually similar to Gibbs sampling, but tends to perform worse when using the same variational posterior.

[13] (BBVI) showed how to do stochastic variational inference in the non-conjugate case for both discrete and continuous latent variables with minimal model-specific derivation. This involves taking a Monte-Carlo approximation of the gradient of the expected lower bound, using ADAGRAD to avoid taking derivatives by hand, and using Rao Blackwellization with a mean-field assumption and control variates to reduce the variance of the gradients. [15] (O-BBVI) extended this work by adding importance sampling to further reduce the variance of the gradient estimator. [19] (VBIL) builds on the same naive estimator as BBVI, but uses natural gradient descent instead of ADAGRAD and treats the ABC case. They show that despite a bias being introduced by taking the log of a likelihood estimator, it is equivalent to using the true likelihood, and they apply this to several intractable likelihood problems. See the supplementary material for the similarity between BBVI and VBIL.

[7] used automatic differentiation to fully automate variational inference, which we refer to as AVI. AVI fits a Gaussian variational distribution in a transformed space, leading to a non-Gaussian approximation in the original space when transformed back. This limits the range of variational distributions. In the transformed space they make a fully-factorized mean-field assumption. AVI requires rewriting the simulator in STAN. With complex simulators, this is unrealistic and negates the main benefit: that it is automatic.

## 3 Automatic Variational ABC

### 3.1 The Variational Lower Bound

In this section we will derive the variational lower bound for our ABC problem and use a number of methods to reduce the variance in estimating its gradients.

Consider a simulator model  $p(\mathbf{x}|\boldsymbol{\theta})$  for which we do not have an analytic expression available but which is given to us as a (potentially complex) piece of computer code (otherwise known as a ‘‘probabilistic program’’). Here,  $\mathbf{x}$  is the output of our simulator, or some summary statistics thereof

(the computation of which we will assume to be a part of the simulator) and  $\theta$  are parameters of the model. The data's marginal likelihood is given by

$$\log p(\mathbf{y}) = \log \int p(\mathbf{y}|\theta)p(\theta)d\theta \quad (4)$$

$$= \log \int Q_\phi(\theta) \frac{p(\mathbf{y}|\theta)p(\theta)}{Q_\phi(\theta)} d\theta \quad (5)$$

$$\geq \int Q_\phi(\theta) \log \frac{p(\mathbf{y}|\theta)p(\theta)}{Q_\phi(\theta)} d\theta \text{ by Jensen's inequality} \quad (6)$$

this assumes a variational posterior  $Q_\phi(\theta)$  over the variable  $\theta$ . We denote the last term as  $\mathcal{L}$  to indicate a lower bound on the marginal probability. Maximizing  $\mathcal{L}$  is equivalent to minimizing the KL divergence between the variational posterior and the true posterior  $KL(Q_\phi(\theta)||p(\theta|\mathbf{y}))$ . We then have

$$\mathcal{L} = \int Q_\phi(\theta) \log \frac{p(\mathbf{y}|\theta)p(\theta)}{Q_\phi(\theta)} d\theta \quad (7)$$

$$= \int Q_\phi(\theta) \log p(\mathbf{y}|\theta) d\theta - KL(Q_\phi(\theta)||p(\theta)) \quad (8)$$

$$= \mathbb{E}_Q(\log p(\mathbf{y}|\theta)) - KL(Q_\phi(\theta)||p(\theta)) \quad (9)$$

We now replace the true likelihood with the ABC likelihood, giving

$$\mathcal{L}_{ABC} = \int Q_\phi(\theta) \log \int p_\epsilon(\mathbf{y}|\mathbf{x})p(\mathbf{x}|\theta)d\mathbf{x}d\theta - KL(Q_\phi(\theta)||p(\theta)) \quad (10)$$

$$= \int Q_\phi(\theta) \log \int p_\epsilon(\mathbf{y}|f(\theta, \mathbf{u}))p(\mathbf{u})d\mathbf{u}d\theta - KL(Q_\phi(\theta)||p(\theta)) \quad (11)$$

where  $p_\epsilon(\mathbf{y}|f(\theta, \mathbf{u}))$  is the  $\epsilon$ -kernel and we have used our first reparameterization [10, 12] where we replace

$$\int d\mathbf{x} p(\mathbf{x}|\theta)H[\mathbf{x}] = \int d\mathbf{u} p(\mathbf{u})H[f(\theta, \mathbf{u})] \quad (12)$$

with  $H$  some arbitrary function of  $\mathbf{x}$ . The simulator  $p(\mathbf{x}|\theta)$  is now written as a deterministic function  $f(\theta, \mathbf{u})$  where the randomness is now externalized into a variable  $\mathbf{u} \sim p(\mathbf{u})$ . For example, if  $p(\mathbf{x}|\theta)$  is normally distributed and  $\theta$  contains the mean and variance, then  $\mathbf{x} = \boldsymbol{\mu} + \boldsymbol{\Sigma}\mathbf{u}$  with  $\mathbf{u} \sim N(\mathbf{0}, \mathbf{I})$ . Once again we will assume that the simulator output is either the raw data or a vector of statistics; in both cases we will represent the deterministic version as  $\mathbf{x} = f(\theta, \mathbf{u})$ . To reduce the variance of the gradients, we apply a second reparameterization [6]:

$$\int d\theta Q_\phi(\theta)H[\theta] = \int d\boldsymbol{\nu} Q_0(\boldsymbol{\nu})H[g(\phi, \boldsymbol{\nu})] \quad (13)$$

$$(14)$$

where it is important to note that the distribution  $Q_0$  is constant w.r.t. any parameters of interest, and all dependency on the parameters  $\phi$  is transferred to the function  $g$ . This gives

$$\mathcal{L}_{ABC} = \int Q_0(\boldsymbol{\nu}) \log \int p_\epsilon(\mathbf{y}|f(g(\phi, \boldsymbol{\nu}), \mathbf{u}))p(\mathbf{u})d\mathbf{u}d\boldsymbol{\nu} - KL(Q_\phi(\theta)||p(\theta)) \quad (15)$$

Assuming that  $KL(q(\theta)||p(\theta))$  can be calculated analytically, we replace expectations by samples to obtain

$$\mathcal{L}_{ABC} \approx \frac{1}{S} \sum_{s=1}^S \log \frac{1}{L} \sum_{l=1}^L p_\epsilon(\mathbf{y}|f(g(\phi, \boldsymbol{\nu}^{(s)}), \mathbf{u}^{(l)})) - KL(Q_\phi(\theta)||p(\theta)) \quad (16)$$

where  $\boldsymbol{\nu}^{(s)} \sim Q_0(\boldsymbol{\nu})$  and  $\mathbf{u}^{(l)} \sim p(\mathbf{u})$ .

When  $KL(Q_\phi(\theta)||p(\theta))$  cannot be calculated analytically, we can apply the second reparameterization to it and approximate it via sampling.

---

**Algorithm 1** Automatic Variational ABC

---

Specify simulation code  $f(\boldsymbol{\theta}, \mathbf{u})$  and simulator randomness  $p(u)$   
Specify reparametrization  $Q(\boldsymbol{\theta}|\phi)$  in terms of parameterless  $Q_0(\boldsymbol{\nu})$   
 $\phi \leftarrow$  initialize parameters  
**repeat**  
   $\boldsymbol{\nu} \leftarrow$  samples from  $Q_0(\boldsymbol{\nu})$   
   $\mathbf{u} \leftarrow$  samples from  $p(u)$   
   $g \leftarrow \nabla \tilde{\mathcal{L}}_\phi(\phi, \boldsymbol{\nu}, u)$   
   $\phi \leftarrow$  run one step of an adaptive gradient algorithm using  $g$ .  
**until** convergence  
return  $\phi$

---

One issue is that even assuming that the ABC likelihood is an unbiased estimator of the true likelihood (which it is not), taking the log introduces a bias, so that we now have a biased estimate of the lower bound and thus biased gradients. We show in the appendix that the theory developed in VBIL applies to our estimator, and thus taking the log of an estimator is not a problem. In the appendix we extend this model to learning a variational distribution for latent variables per datapoint.

### 3.2 Automatic Differentiation and Averaged Gradients

We now wish to take derivatives w.r.t.  $\phi$  to compute stochastic gradients and ascent on the bound eq. 16. This requires differentiating through  $\log p_\epsilon(\mathbf{y}|f(g(\phi, \boldsymbol{\nu}^{(s)}), \mathbf{u}))$ , and via the chain rule, also through the simulator function  $f(\boldsymbol{\theta}, \mathbf{u})$ . Because we write our simulator code within an auto-differentiation environment (in our case, Python, where we can apply Autograd [8]), this is handled automatically by the optimization algorithm. This avoids the model specific derivation of taking derivatives by hand, while allowing us to use a wider range of variational posteriors than [7]. A strong requirement that will be discussed further in the experiments, is that the function  $g$  can deterministically generate  $\boldsymbol{\theta}$  and  $\mathbf{u}$ , which may not be possible for all distributions of interest.

In our experiments we use adaptive gradient algorithms which automatically tune the learning rates per parameter according to the history of gradients and their variances. Two such algorithms are ADAM [5] and ADAGRAD [4]: we used the latter in our experiments. As we will discuss later, using these algorithms greatly reduces the effect of large variance gradients and is especially helpful for VBIL experiments. Algorithm 1 describes the procedure for sampling from parameterless distributions, differentiating the lower bound, and applying an adaptive gradient-step.

Another important challenge is choosing  $\epsilon$ . As we increase  $\epsilon$ , the bias of the likelihood and thus our gradient estimator goes up, but as we decrease it, the variance goes up. Wilkinson [20] showed that in the likelihood-free setting,  $\epsilon$  could be interpreted as model or simulation noise. We can empirically take advantage of this interpretation and set  $\epsilon$  according to the standard deviation of the mean of the raw simulation data, depending upon whether this is appropriate for the statistics used. For example, if  $\mathbf{x}$  is the length  $M$  vector of raw data from the simulator we can set  $\epsilon = \sigma(\mathbf{x})/\sqrt{M}$ .

## 4 Experiments

We perform three experiments: 1) inferring the success probability from Bernoulli trials 2) inferring the rate of an exponential distribution 3) inferring the parameters of a stochastic biological system of blowfly populations. For each experiment, we use a Gaussian kernel for the ABC-likelihood, and adaptively set  $\epsilon$  depending on the simulation problem. We compare to BBVI, which as mentioned before, is the same as VBIL but using ADAGRAD [4] instead of natural gradient descent. We use ADAM for both our method and BBVI (in the original paper, they used ADAGRAD). Because of the posterior limitations and requirement of rewriting a potentially complex simulator in STAN, we do not compare to AVI. Due to higher variance, running BBVI occasionally leads to invalid parameter values; when that happens, we reinitialize.

#### 4.1 Bernoulli Problem

We perform the first VBIL experiment. We have  $M$  Bernoulli samples, each with success probability  $\theta$ , and we infer a distribution for  $\theta$ . Our observed data  $\mathbf{y}$ , is a vector of 1's and 0's. Let  $k = \sum_{m=1}^M y_m$ . With a Beta(1, 1) prior, the true posterior is Beta( $k + 1, M - k + 1$ ). Because we need to differentiate with respect to the simulator, instead of simulating  $M$  Bernoulli trials, which gives discrete outputs, we use the normal approximation to the binomial with  $M$  trials and success probability  $\theta$ , and sample from that. Thus  $S(\mathbf{x})$  is simply the output of the simulator. We set  $M = 100$  and  $k = 70$ . For this problem, we cannot calculate the sample variance as in the  $\epsilon$ -selection method described. However, we note that under the Gaussian approximation to the binomial, the simulator variance is  $M \cdot \theta(1 - \theta)$ , so we set  $\epsilon = \sqrt{\theta(1 - \theta)}$ .

For both AVABC and BBVI, we use a Kumaraswamy distribution as the variational posterior for  $\theta$ , since the Kumaraswamy distribution is similar to the Beta, but has a closed form inverse CDF, allowing us to generate samples deterministically given samples from the uniform distribution. Thus,  $Q_0(\boldsymbol{\nu}) = U(0, 1)$ . For the Gaussian approximation in the simulation, we use  $\mathcal{N}(\mathbf{0}, \mathbf{I})$ . We initialize by setting the Kumaraswamy parameters to be (1, 1).

Figure 1(a) shows the lower bound for the Beta problem. Both methods have their lower bounds stabilize in under 100 iterations. 1(b) shows the posteriors: both are reasonable. 1(c) shows the naive gradient distribution taking 100 samples at the convergence parameters. We see that for 10 samples and simulations per sample, the gradients are approximately equal, but for 1 sample and simulation, AVABC has far lower variance than BBVI. 1(d) shows the naive gradient distributions for a full run of the algorithm. That is, we run each algorithm from start until convergence, store the gradients, and plot the distributions. AVABC has far lower variance. ADAM drastically reduces the variance of both methods so that they are almost equivalent, leading to the similar convergence.

#### 4.2 Exponential Problem

We wish to infer a distribution for the rate  $\lambda$  of an exponential distribution. In this setting, with a Gamma( $\alpha, \beta$ ) prior and a data vector  $y$  of  $M$  samples from the exponential distribution, the true posterior is Gamma( $\alpha + M, \beta + \bar{y}M$ ). The simulator draws samples from the exponential distribution. We use statistics  $S(\mathbf{x}) = \bar{x}$  and  $S(\mathbf{y}) = \bar{y}$ . We set the true  $\lambda = 1$ , and  $M = 15$ .

For both methods, we use a log-normal variational posterior model. At each iteration, we can take samples  $\boldsymbol{\nu}^{(s)}$  from  $Q_0(\boldsymbol{\nu}) \sim \mathcal{N}(0, \mathbf{I})$  and then deterministically calculate

$$\theta = g(\boldsymbol{\phi}, \boldsymbol{\nu}^{(s)}) = \exp(\mu + \Sigma \cdot \boldsymbol{\nu}_s) \quad (17)$$

We initialize both  $\mu$  and  $\sigma$  by drawing from  $U(2, 3)$ . The convergence parameters are approximately  $\mu = 0$  and  $\sigma = 0.3$ , so given that we use the lognormal distribution, these parameters are fairly far away. We use 10 samples and 10 particles/simulations per sample. That is,  $S = 10$  and  $L = 10$ .

Figures 2(a) shows the lower bound plot for the exponential problem. AVABC, in blue, has much lower variance and faster convergence. AVABC stabilizes at around 4,000 iterations, while BBVI takes over 30,000 iterations to stabilize. 2(b) shows the posteriors against the true posterior (green). Both have relatively good estimates. 2(c) shows the distribution of gradients for both a single sample and 10 samples. For 10 samples, the AVABC gradient distribution has far lower variance and is thus more peaked.

#### 4.3 Blowfly Problem

Performing well-founded statistical inference in biological dynamic models representing chaotic and near-chaotic systems is difficult. We apply our method to a problem where the dynamic behavior of adult blowfly populations is analyzed. We use the observed data of [22] where discretized differential equations are used to model the blowfly population dynamics. Parameter settings resulting from this modeling can have some chaotic behavior. We base our simulation on the following -equation (1) in Section 1.2.3 of the supplementary material in [22]-:

$$N_{t+1} = PN_{t-\tau} \exp\left(\frac{-N_{t-\tau}}{N_0}\right)e_t + N_t \exp(-\delta\epsilon_t) \quad (18)$$

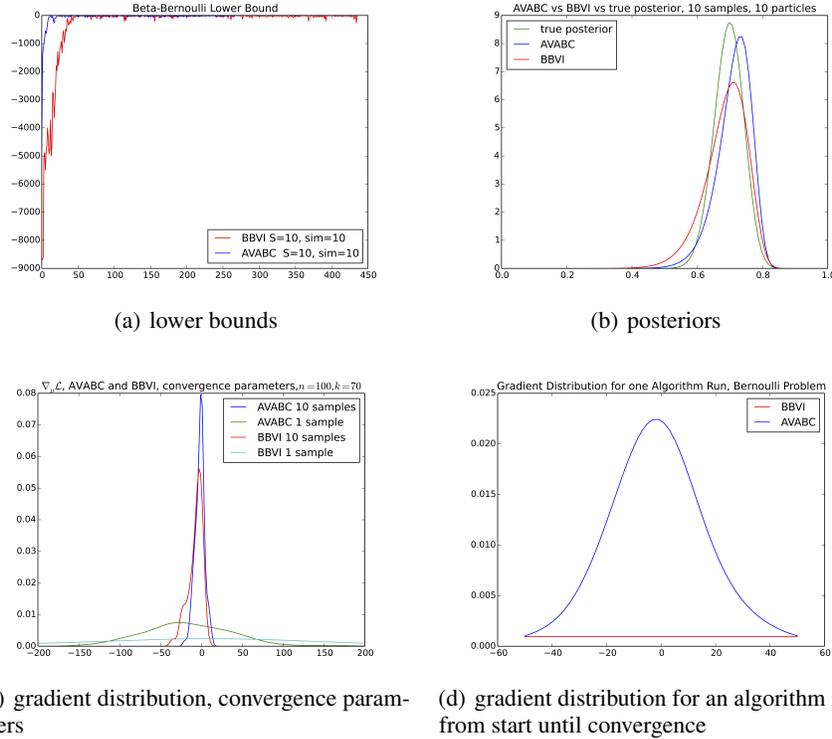


Figure 1: a) Lower bounds for the Bernoulli problem. Performance is similar between the two methods. Both methods stabilize in about 100 iterations. b) posteriors for both methods compared to the true posterior. Both achieve decent posteriors, despite using a different variational distribution from the true posterior family. c) Gradients for the Bernoulli problem for  $\mu$  (chosen because it has higher variance gradients than  $\sigma$ ) at convergence values. Our model with 10 samples is in blue, while for BBVI it is in red. For 10 samples, they have similar gradient variance. However, our model with 1 sample is in green, while BBVI with 1 sample is turquoise, showing that for a single sample our model has far lower variance. d) gradient distribution for the Bernoulli problem for  $\mu$  across a full run of the algorithm.

where  $e_t \sim \mathcal{G}(1/\sigma_p^2, 1/\sigma_p^2)$  and  $e_t \sim \mathcal{G}(1/\sigma_d^2, 1/\sigma_d^2)$  are noise sources, and  $\tau$  is an integer. There are 5 parameters to estimate;  $\theta = \{\log P, \log \delta, \log N_0, \log \sigma_d, \log \sigma_p\}$ . Similar to [11, 10], priors of  $\theta_{1...5}$  are Gaussian distributions. The vector of observations,  $\mathbf{y}$ , consists of a time-series of daily counts of a blowfly population. Generating data  $\mathbf{x}$ , from a simulator based on parameters,  $\theta$ , of a time-series, is an intriguing task, since, in addition to their chaotic nature, small changes in the parameter prior(s) might lead to degenerate  $\mathbf{x}$ . There are 10 statistics in use (again similar to [11]): the mean values of each of the 4 quartiles of the simulated samples, the mean values of the 4 quartiles of the first-order differences of the simulated samples and the number of maximal peaks under 2 different thresholds.

Our prior and Q distribution are Gaussian:

$$\begin{aligned}
 p(\theta) &= \mathcal{N}(\mu, \sigma) \\
 Q(\theta|\phi) &= \mathcal{N}(\mu_\phi, \sigma_\phi)
 \end{aligned}
 \tag{19}$$

where  $\mathcal{N}$  refers to a Gaussian distribution and therefore  $g$  is the transformation of standard normals.

For this experiment, we used control variates and ADAM as the only variance reduction for BBVI. However, both our method and BBVI can be easily extended to use Rao Blackwellization as well. For the 5 parameters,  $\theta_{1...5}$ , we obtain the posteriors shown in Figures 3(a)-3(f). In Figure 3(e), the Blowfly lower bound plot is shown. As can be seen in Figure 3(e), the AVABC lower bound has a lower variance and a faster convergence rate, i.e. number of required iterations for AVABC before it stabilizes is about 175, whereas BBVI requires nearly 2750 iterations in order to stabilize to some

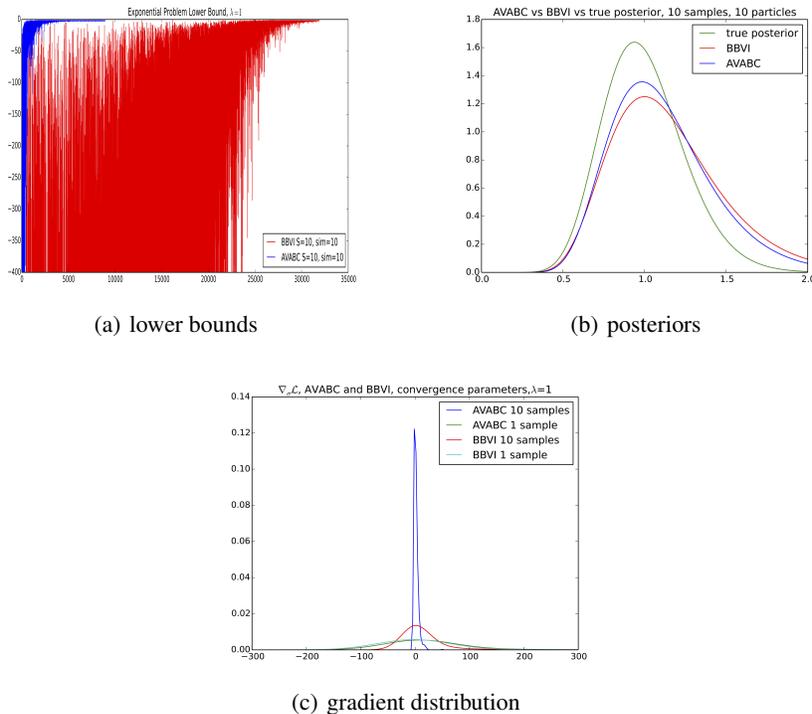


Figure 2: a) Lower bounds for the exponential problem with 10 samples and 10 particles/simulations per sample. BBVI (red) has drastically higher variance and takes substantially more iterations to converge than AVABC (blue). b) posteriors for both methods compared to the true posterior c) Gradients for the exponential problem at convergence values. Our model with 10 samples is in blue and is the most peaked, showing that with 10 samples, our model has far lower variance close to a local optimum.

extent. In fact as per the current set of experiments performed on blowfly, BBVI did not really reach a high level of stability.

## 5 Discussion and Future Work

In this paper, we introduced Automatic Variational ABC, a low-variance variational likelihood-free inference algorithm. With our approach, all simulation code, prior and Q distributions, and the variational lower-bound are written within an autodifferentiation environment. Taking this route towards Bayesian inference of simulation models requires a initial overhead of rewriting simulation code, but reaps the benefits of allowing end-to-end optimization of the lower-bound in an automatic fashion using stochastic gradient ascent.

Our algorithm also uses several reparameterizations of both the simulation—making it a deterministic function of parameters and random latents—and the Q distributions. These reparameterizations not only allow one to perform automatic inference, but also greatly reduce the variance of the gradients of the lower-bound. This variance reduction makes variational inference for likelihood-free models feasible. Further, by using adaptive gradient-step algorithms, such as Adagrad or Adam, learning rates can automatically be adjusted to the variance of the gradients, which helps significantly for both our algorithm and alternatives. For variational inference of simulation models, variance reduction is the name of the game.

We demonstrated performance on three experiments and compared to alternatives BBVI and VBIL; we found that AVABC achieves much lower variance and faster convergence for both toy and real problems. Despite the positive preliminary results AVABC demonstrated, our algorithm in the current form has clear limitations. First, it assumes that one can write the variational posterior distributions

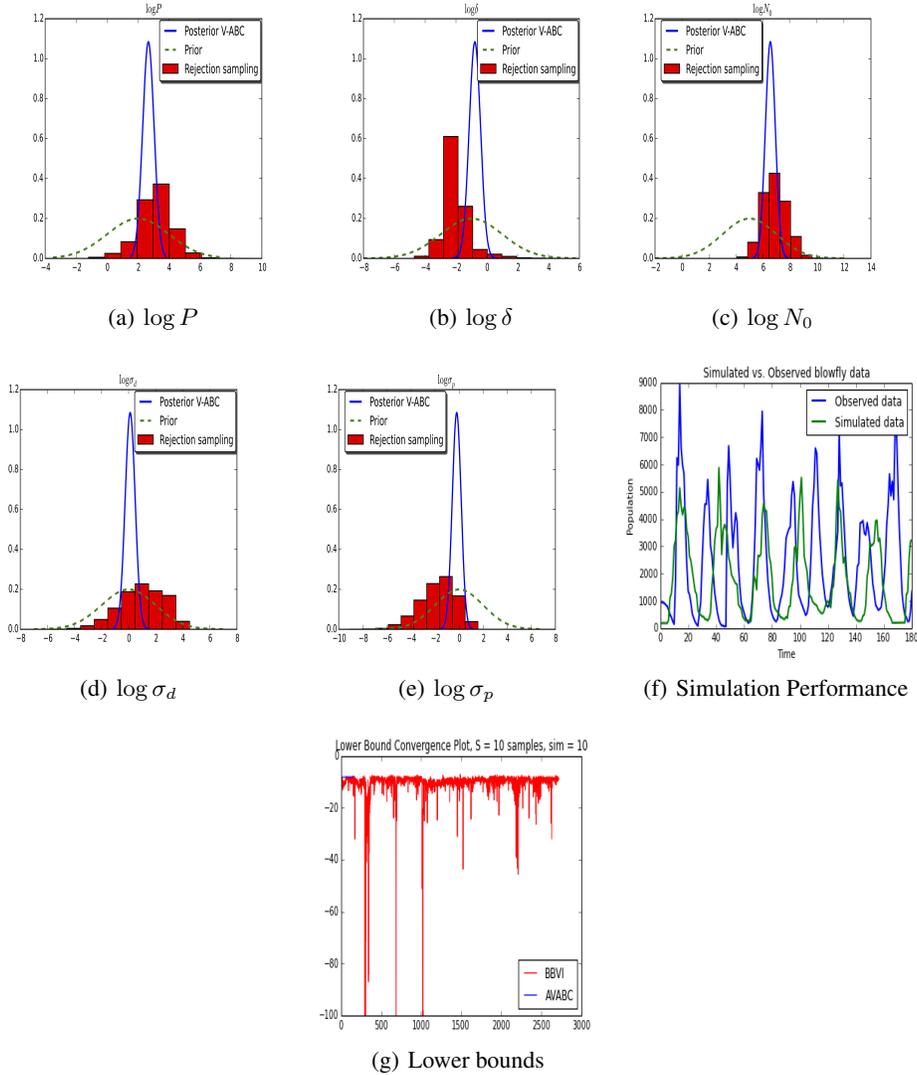


Figure 3: a-e: Plots for the first 5 parameters of the blowfly experiment: Posterior probability,  $p(\theta|\mathbf{y})$ , along with the corresponding prior probabilities and rejection samples produced by  $\epsilon = 0.25$ . For AVABC, 10 simulations per sample are used. f: Simulation performance, plot of simulated vs observed data at each time step g: Lower bounds with 10 samples and 10 simulations per sample. Variance of AVABC (blue) is considerably lower than BBVI (red). Convergence of AVABC requires fewer iterations than BBVI.

as a deterministic function of its parameters and a set of parameterless variables. Some distributions, such as the Beta or Gamma distribution, cannot be written this way since they involve rejection steps (this could be overcome but it requires further thought). Alternative distributions such as the Kumaraswamy for the Beta and the Weibull or log-normal for a Gamma must be used. Second, because we must differentiate the simulator, the outputs must be differentiable, which is a restrictive assumption. Often, we can approximate discrete outputs by a continuous distribution, as we did when approximating the binomial distribution with a normal distribution, but this may not always be the case. Most significantly, it requires writing possibly very complex simulation code within an auto-dif environment, which may only be possible on a limited set of simulations.

In the future we would like to expand upon the allowable Q distributions by writing the random number generators within an auto-dif environment. More challenging simulators, such as state-space models should also be ported. Along the lines of [11][21], it would be possible add surrogate functions

of the simulator and thus call the surrogate instead of the simulator when it has high confidence. Another, as suggested in O-BBVI [15], is to combine importance sampling with the reparametrization of the variational distribution for further variance reduction.

## Acknowledgments

We thank Facebook and Google for their support. This work was supported in part by the National Institute of Health grant number R01EY013178, the Amsterdam Academic Alliance Data Science (AAA-DS) Program Award to the UvA and VU Universities, the Georgia Tech Executive Vice President of Research Office and the Center for Computational Health, grants from the Gordon and Betty Moore Foundation and the National Science Foundation and contributions from OCC members like the University of Chicago. Finally, we thank the authors of VBIL for providing their code, and Christian A. Naesseth for helpful discussions that led us to realize the relationship between our method and VBIL.

## References

- [1] Shun-Ichi Amari. Natural gradient works efficiently in learning. *Neural computation*, 10(2):251–276, 1998.
- [2] Mark A Beaumont, Jean-Marie Cornuet, Jean-Michel Marin, and Christian P Robert. Adaptive approximate Bayesian computation. *Biometrika*, 96(4):983–990, 2009.
- [3] Pierre Del Moral, Arnaud Doucet, and Ajay Jasra. Sequential Monte Carlo samplers. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(3):411–436, 2006.
- [4] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, 12:2121–2159, 2011.
- [5] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [6] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2014.
- [7] Alp Kucukelbir, Rajesh Ranganath, Andrew Gelman, and David Blei. Automatic variational inference in stan. In *Advances in Neural Information Processing Systems*, pages 568–576, 2015.
- [8] D. Maclaurin and D. Duvenaud. Autograd. [github.com/HIPS/autograd](https://github.com/HIPS/autograd), 2015.
- [9] 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.
- [10] Edward Meeds, Robert Leenders, and Max Welling. Hamiltonian ABC. *Uncertainty in AI*, 2015.
- [11] Edward Meeds and Max Welling. GPS-ABC: Gaussian process surrogate approximate Bayesian computation. *Uncertainty in AI*, 2014.
- [12] Edward Meeds and Max Welling. Optimization monte carlo: Efficient and embarrassingly parallel likelihood-free inference. *arXiv preprint arXiv:1506.03693*, 2015.
- [13] Rajesh Ranganath, Sean Gerrish, and David M Blei. Black box variational inference. *AISTATS*, 2014.
- [14] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.
- [15] Francisco JR Ruiz, Michalis K Titsias, and David M Blei. Overdispersed black-box variational inference. *arXiv preprint arXiv:1603.01140*, 2016.
- [16] SA Sisson, Y Fan, and Mark M Tanaka. Sequential Monte Carlo without likelihoods. *Proceedings of the National Academy of Sciences*, 104(6), 2007.
- [17] S. Tavaré, D.J. Balding, R.C. Griffiths, and P. Donnelly. Inferring coalescence times from dna sequence data. *Genetics*, 145(2):505–518, 1997.
- [18] Michalis Titsias and Miguel Lázaro-Gredilla. Local expectation gradients for black box variational inference. In *Advances in Neural Information Processing Systems*, pages 2620–2628, 2015.
- [19] Minh-Ngoc Tran, David J Nott, and Robert Kohn. Variational bayes with intractable likelihood. *arXiv preprint arXiv:1503.08621*, 2015.
- [20] R. Wilkinson. Approximate bayesian computation (ABC) gives exact results under the assumption of model error. *Statistical Applications in Genetics and Molecular Biology*, 12(2):129–142, 2013.
- [21] R. Wilkinson. Accelerating abc methods using gaussian processes. *AISTATS*, 2014.
- [22] S. Wood. Statistical inference for noisy nonlinear ecological dynamic systems. *Nature*, 466(7310):1102–1104, 2010.

## 6 Relationship between AVABC and VBIL

In VBIL, they show that minimizing an augmented parameter KL divergence is equivalent to minimizing the KL divergence between the variational distribution and the true posterior. The KL divergence they minimize is given by

$$KL(\lambda) = \int q_\lambda(\theta) g_N(z|\theta) \log \frac{q_\lambda(\theta)}{p(\theta)p_N(y|\theta, z)} dz d\theta \quad (20)$$

where  $q_\lambda(\theta)$  is the variational distribution in the original parameter space,  $p_N(y|\theta, z)$  is an estimator of the likelihood (for example, in the ABC case  $p_N(y|\theta, z) = p_\epsilon(y|\theta)$ , the ABC likelihood),  $z = \log \frac{p_N(y|\theta, z)}{p(y|\theta)}$ , and  $g_N(z|\theta)$  is the density for  $z$ . The VBIL authors note that calculating  $p_N(y|\theta, z)$  implicitly generates  $z$ .

Now note

$$KL(\lambda) = \int q_\lambda(\theta) g_N(z|\theta) \log \frac{q_\lambda(\theta)}{p(\theta)p_N(y|\theta, z)} dz d\theta \quad (21)$$

$$= \int q_\lambda(\theta) \log \frac{q_\lambda(\theta)}{p(\theta)} d\theta - \int q_\lambda(\theta) g_N(z|\theta) \log p_N(y|\theta, z) dz d\theta \quad (22)$$

$$= KL(q(\theta)||p(\theta)) - \int q_\lambda(\theta) g_N(z|\theta) \log p_N(y|\theta, z) dz d\theta \quad (23)$$

$$= KL(q(\theta)||p(\theta)) - E_{\theta \sim q(\theta), z \sim g_N(z|\theta)} (\log p_N(y|\theta, z)) \quad (24)$$

$$= -(E_{\theta \sim q(\theta), z \sim g_N(z|\theta)} (\log p_N(y|\theta, z)) - KL(q(\theta)||p(\theta))) \quad (25)$$

Note the similarity to equation 9 in the main paper. The only differences are that they include the density for  $z$ , and  $p_N(y|\theta, z)$  is any estimator of the likelihood. Since calculating  $p_N(y|\theta, z)$  implicitly generates  $z$ , our estimator for  $\mathcal{L}_{ABC}$  is an unbiased estimator of  $-KL(\lambda)$ . Thus the theory they developed in VBIL holds for our problem and the biased gradients due to taking the log of an estimator are not an issue. The bias introduced by the choice of  $\epsilon$  can in theory be an issue, but we find that in practice the choice we described in the main paper gives good results.

## 7 Similarity between VBIL and BBVI

In VBIL, we have  $\hat{h}(\theta, z) = \log(p(\theta)\hat{p}_N(y|\theta, z))$ , where  $\hat{p}_N(y|\theta, z)$  is an estimator of the likelihood as described above. Let  $q_\lambda(\theta)$  be the variational distribution in the original parameter space. Then, an estimator of the KL divergence between  $q_{\lambda, z}(\theta, z)$  and  $\pi_N(\theta, z)$  in the augmented parameter space is

$$\nabla_\lambda KL(\lambda)^{naive} = \frac{1}{S} \sum_{s=1}^S \nabla_\lambda [\log q_\lambda(\theta^{(s)})] (\log q_\lambda(\theta^{(s)}) - \hat{h}(\theta^{(s)}, z^{(s)})) \quad (26)$$

Note that  $z$  is never dealt with explicitly. It is only handled implicitly via the unbiased estimator to the likelihood. If the unbiased estimator equals the true likelihood, we get  $\hat{h}(\theta, z) = \log(p(\theta)p(y|\theta)) = \log(p(y, \theta))$ . We then have

$$\nabla_{\lambda_i} KL(\lambda)^{naive} = \frac{1}{S} \sum_{s=1}^S \nabla_\lambda [\log q_\lambda(\theta^{(s)})] (\log q_\lambda(\theta^{(s)}) - \log p(y, \theta^{(s)})) \quad (27)$$

$$(28)$$

The estimator for BBVI is

$$\nabla_\lambda \mathcal{L} \approx \frac{1}{S} \sum_{s=1}^S \nabla_\lambda \log q_\lambda(\theta^{(s)}) (\log p(y, \theta^{(s)}) - \log q_\lambda(\theta^{(s)})) \quad (29)$$

Note that the VBIL estimator under the true likelihood is the negative of the BBVI estimator. Building from this, VBIL has one major difference: they use natural gradient descent [1] instead of ADAGRAD [4] or ADAM [5].

## 8 Learning Latent Variables per Datapoint

We start with the marginal probability as before, but introduce latent variables  $z$  (no relationship with the  $z$  described for VBIL) and assume a factorized variational posterior  $Q(\theta, z|\phi, \xi) = Q_\phi(\theta) \prod_n Q_\xi(z_n)$  and factorized prior  $p(\theta, z) = p(\theta)p(z)$ .

$$\log p(y) = \log \int p(y|\theta, z)p(\theta)p(z)d\theta dz \quad (30)$$

$$= \log \int Q_\phi(\theta)Q_\xi(z) \frac{p(y|\theta, z)p(\theta)p(z)}{Q_\phi(\theta)Q_\xi(z)} d\theta dz \quad (31)$$

$$= \log \mathbb{E}_{Q_\phi(\theta), Q_\xi(z)} \left( \frac{p(y|\theta, z)p(\theta)p(z)}{Q_\phi(\theta)Q_\xi(z)} \right) \quad (32)$$

$$\geq \mathbb{E} \left( \log \frac{p(y|\theta, z)p(\theta)p(z)}{Q_\phi(\theta)Q_\xi(z)} \right) \quad (33)$$

$$= \mathbb{E}(\log p(y|\theta, z)) - KL(Q_\phi(\theta)||p(\theta)) - KL(Q_\xi(z)||p(z)) \quad (34)$$

$$(35)$$

We again apply the ABC likelihood and the reparametrization to make the simulator deterministic

$$\mathcal{L}_{ABC} = \mathbb{E}_{Q_\phi(\theta)Q_\xi(z)} (\log p_\epsilon(y|\theta, z)) - KL(Q_\phi(\theta)||p(\theta)) - KL(Q_\xi(z)||p(z)) \quad (36)$$

$$= \mathbb{E}(\log \int p_\epsilon(y|f(\theta, z, u))p(u)du) - KL(Q_\phi(\theta)||p(\theta)) - KL(Q_\xi(z)||p(z)) \quad (37)$$

$$= \mathbb{E}(\log \int p_\epsilon(y|f(g(\phi, \nu), h(\xi, w), u))p(u)du) - KL(Q_\phi(\theta)||p(\theta)) - KL(Q_\xi(z)||p(z)) \quad (38)$$

We can again replace all expectations with samples to obtain

$$\begin{aligned} \mathcal{L}_{ABC} \approx & \frac{1}{S} \frac{1}{K} \sum_s \sum_k \log \frac{1}{L} \sum_l p_\epsilon(y|f(g(\phi, \nu^{(s)}), h(\xi, w^{(k)}), u^{(l)})) - KL(Q_\phi(\theta)||p(\theta)) \\ & - KL(Q_\xi(z)||p(z)) \end{aligned} \quad (39)$$