Hamiltonian ABC

Abstract

Approximate Bayesian computation (ABC) is a powerful and elegant framework for performing inference in simulation-based models. However, due to the difficulty in scaling likelihood estimates, ABC remains useful for relatively low-dimensional problems. We introduce Hamiltonian ABC (HABC), a set of likelihood-free algorithms that apply recent advances in scaling Bayesian learning using Hamiltonian Monte Carlo (HMC) and stochastic gradients. We find that a small number forward simulations can effectively approximate the ABC gradient, allowing Hamiltonian dynamics to efficiently traverse parameter spaces. We also describe a new simple yet general approach of incorporating random seeds into the state of the Markov chain, further reducing the random walk behavior of HABC. We demonstrate HABC on several typical ABC problems, and show that HABC performs comparably to regular Bayesian inference on a high-dimensional problem from machine learning.

1 INTRODUCTION

In simulation-based science, models are defined by a simulator and its parameters. These are called *likelihood-free* models because, in contrast to probabilistic models, their likelihoods are either intractable to compute or must be approximated by simulations. To perform inference in likelihood-free models, a broad class of algorithms called Approximate Bayesian Computation [2, 10, 13, 14, 9, 6] are employed.

At the core of every ABC algorithm is simulation. To evaluate the quality of a parameter vector $\boldsymbol{\theta}$, a simulation is run using $\boldsymbol{\theta}$ as inputs and producing outputs $\mathbf{x}$. If the pseudo-data $\mathbf{x}$ is "close" to observations $\mathbf{y}$, then $\boldsymbol{\theta}$ is kept as a sample from the approximate posterior. Parameters $thetav$ are then adjusted, depending upon the algorithm, to obtain the next sample.

In ABC, there is a fundamental trade-off between the computation required to obtain independent samples and the approximation to the true posterior. If the parameter measuring closeness is too small, then samplers "mix" poorly; on the other hand, if it is too large, then the approximation is poor. As the dimension of the parameters grows, the problem worsens, just as it does for general Bayesian inference with probabilistic models, but it is more acute for ABC due to its simulation requirement. There is therefore a deep interest in improving the efficiency of ABC samplers (in terms of computation per independent sample). In this paper we address this issue directly by using Hamiltonian dynamics to approximately sample from likelihood-free models with high-dimensional parameters.

Hamiltonian Monte Carlo (HMC) [5, 12] is perhaps the only Bayesian inference algorithm that scales to high-dimensional parameter spaces. The core computation of HMC is the gradient of the log-likelihood. Two problems arise if we consider HMC for ABC: one, how can the gradients be computed for high-dimensional likelihood-free models, and two, given a stochastic approximation to the gradient, can a valid HMC algorithm be derived?

To answer the latter, we turn to recent developments in scaling Bayesian inference using HMC and stochastic gradients [19, 3, 4]. We call these *stochastic gradient Hamiltonian dynamics* (SGHD) algorithms. SGHD are computationally efficient for two reasons. First, they avoid computing the gradient of the log-likelihood over the entire data set, instead approximating it using small batches of data, i.e. computing stochastic gradients. Second, they can maintain reasonable approximations to the Hamiltonian dynamics and therefore avoid a Metropolis-Hastings correction step involving the full data set. Different strategies are employed to do this: small step-sizes combined with Langevin dynamics [19], using friction to prevent accumulation of errors in the Hamiltonian [3], and using a thermostat to control the temperature of the Hamiltonian [4]. Each of these strategies can be used by HABC.

In HABC, we use forward simulations to approximate the likelihood-free gradient. The key difference between SGHD methods and HABC is that the stochasticity of the gradient does not come from approximating the full data gradient with a batch gradient, but by the stochasticity of the simulator. It is therefore not the expense of the simulator (though this could very well be the case for many interesting simulation-based models – see Section 8) that requires an approximation to the gradient, but the likelihood-free nature of the problem.

There are several difficulties in estimating gradients of likelihood-free models that we address with HABC. The first is do to the form of the ABC log-likelihood. As we show in Section 2, using a conditional model for $\pi(\mathbf{x}|\boldsymbol{\theta})$ provides an estimate of the ABC likelihood that is less sensitive to ϵ and therefore is more conducive to stochastic gradient computations. The second difficulty is that for high-dimensional parameter spaces, computing the gradients naively (i.e. by finite differences [7]) can squash the gains brought by the Hamiltonian dynamics. Fortunately, we can use existing stochastic approximation algorithms [15, 16] that can be used to compute unbiased estimators of the gradient with a small number of forward simulations that is *independent* of the parameter dimension. The *stochastic perturbation stochastic approximation* (SPSA) [15] is described in Section 4

A further innovation of this paper is the use of common random numbers (CRN) to improve the efficiency of the Hamiltonian dynamics. The idea behind CRNs is to use the same set of random seeds for estimating a gradient by FD or SPSA, i.e. when simulating $\pi(\mathbf{x}|\theta+d\theta)$ and $\pi(\mathbf{x}|\theta-d\theta)$ use the same random seeds. This was applied successfully to SPSA [8] (and is analogous to using the same mini-batch in stochastic gradient methods). We extend and simplify this approach by including the random seeds ω into the state of the Markov chain; by keeping the random seeds fixed for several consecutive steps, the second order gradient stochasticity is greatly reduced. We show that this doing this produces is a valid MCMC procedure. This approach is not exclusive to HABC; we apply successfully to ABC-MCMC as well.

We briefly review ABC in Section 2. In Section 3 we review three approaches to stochastic gradient inference using Hamiltonian dynamics: SGLD, SGHMC, and SGNHT. We then introduce Hamiltonian ABC in Section 4, where we will show how to improve the stability of the gradient estimates by using CRNs and local density estimators of the simulator. Extensions to high-dimensional parameter spaces are also discussed. In Section 5 we show how HABC behaves on a simple one-dimensional problem, then in Section 6 we compare HABC with ABC-MCMC for two problems: a low-dimensional model of chaotic population dynamics and a high-dimensional problem.

2 APPROXIMATE BAYESIAN COMPUTATION

Consider the Bayesian inference task of either drawing samples from or learning an approximate model of the following (usually intractable) posterior distribution:

$$\pi(\boldsymbol{\theta}|\mathbf{y}_1,\ldots,\mathbf{y}_N) \propto \pi(\boldsymbol{\theta})\pi(\mathbf{y}_1,\ldots,\mathbf{y}_N|\boldsymbol{\theta}) \qquad (1)$$

where $\pi(\boldsymbol{\theta})$ is a prior distribution over parameters $\boldsymbol{\theta} \in \mathbb{R}^D$ and $\pi(\mathbf{y}_1,\ldots,\mathbf{y}_N|\boldsymbol{\theta})$ is the likelihood of N data observations, where $\mathbf{y}_i \in \mathbb{R}^J$. In ABC, the vector of J observations are typically informative statistics of the raw observations. It can be shown that if the statistics used in the likelihood function are sufficient, then these algorithms sample correctly from an approximation to the true posterior ([KEEP?]). The simulator is treated as generator of random pseudo-observations, i.e. $\mathbf{x} \overset{\text{sim}}{\sim} \pi(\mathbf{x}|\boldsymbol{\theta})$ is a draw from the simulator. Discrepancies between the simulator outputs $\mathbf{x}$ and the observations $\mathbf{y}$ are scaled by a closeness parameter ϵ and treated as likelihoods. This is the equivalent to putting an ϵ-kernel around the observations, and using a Monte Carlo estimate of the likelihood using S draws of $\mathbf{x}$:

$$\pi_{\boldsymbol{\epsilon}}(\mathbf{y}|\boldsymbol{\theta}) = \int \pi_{\boldsymbol{\epsilon}}(\mathbf{y}|\mathbf{x})\pi(\mathbf{x}|\boldsymbol{\theta})d\mathbf{x} \approx \frac{1}{S}\sum_{s=1}^{S}\pi_{\boldsymbol{\epsilon}}(\mathbf{y}|\mathbf{x}^{(s)}) \qquad (2)$$

In ABC Markov chain Monte Carlo (MCMC) [10, 20, 14] the Metropolis-Hastings (MH) proposal distribution is composed of the product of the proposal for the parameters $\boldsymbol{\theta}$ and the proposal for the simulator outputs:

$$q(\boldsymbol{\theta}',\mathbf{x}^{(1)'},\ldots,\mathbf{x}^{(S)'}|\boldsymbol{\theta}) = q(\boldsymbol{\theta}'|\boldsymbol{\theta})\prod_s \pi(\mathbf{x}^{(s)'}|\boldsymbol{\theta}') \qquad (3)$$

Using this form of the proposal distribution, and using the Monte Carlo approximation eq 2, we arrive at the following Metropolis-Hastings accept-reject probability,

$$\alpha = \min\left(1, \frac{\pi\left(\boldsymbol{\theta}'\right)\sum_{s=1}^{S}\pi_{\boldsymbol{\epsilon}}(\mathbf{y}|\mathbf{x}'^{(s)})q(\boldsymbol{\theta}|\boldsymbol{\theta}')}{\pi\left(\boldsymbol{\theta}\right)\sum_{s=1}^{S}\pi_{\boldsymbol{\epsilon}}(\mathbf{y}|x)q(\boldsymbol{\theta}'|\boldsymbol{\theta})}\right) \qquad (4)$$

If the simulations are part of the Markov chain, the algorithm corresponds to the pseudo-marginal (PM) sampler [1], otherwise it is a marginal sampler [10, 14]. For this paper we will be interested in the PM sampler because this is equivalent to having the random states that generated the simulation outputs in the state of the Markov chain, which we will use within a valid ABC sampling algorithm in Section 4.

An alternative approach to computing the ABC likelihood is to estimate the parameters of a conditional model $\pi(\mathbf{x}|\boldsymbol{\theta})$, e.g. kernel density estimate [17] or a Gaussian model [22]. While either approach should be adequate and both have their own limits and advantages, for this paper we will use a Gaussian model. In ABC, using a conditional Gaussian model for $\pi(\mathbf{x}|\boldsymbol{\theta})$ is called a *synthetic likelihood* (SL) model [22]. For a SL log-likelihood model, we compute estimators of the first and second moments of $\pi(\mathbf{x}|\boldsymbol{\theta})$. The advantage is that for a Gaussian ϵ-kernel, we can convolve the two densities

$$\begin{aligned} \pi_{\boldsymbol{\epsilon}}(\mathbf{y}|\boldsymbol{\theta}) &= \int \mathcal{N}(\mathbf{y}|\mathbf{x}, \epsilon^2)\mathcal{N}(\mathbf{x}|\mu_{\boldsymbol{\theta}}, \sigma^2_{\boldsymbol{\theta}})d\mathbf{x} \quad (5) \\ &= \mathcal{N}(\mathbf{y}|\mu_{\boldsymbol{\theta}}, \sigma^2_{\boldsymbol{\theta}} + \epsilon^2) \quad (6) \end{aligned}$$

Of particular concern to this paper is the behavior of the log-likelihoods for different values of ϵ. In the ϵ-kernel case, the log-likelihood is very sensitive to small values of ϵ:

$$\begin{aligned} \log \pi_{\boldsymbol{\epsilon}}(\mathbf{y}|\boldsymbol{\theta}) &= \log \sum_s \mathcal{N}(\mathbf{y}|\mathbf{x}^{(s)}, \epsilon^2) \quad (7) \\ &= \log \mathcal{N}(\mathbf{y}|\mathbf{x}^{(s)}, \epsilon^2) + \log(1+H) \quad (8) \\ &\approx -\log\epsilon - \frac{1}{2\epsilon^2}(\mathbf{y} - \mathbf{x}^{(m)})^2 \quad (9) \end{aligned}$$

where m is the simulation that is closest to $\mathbf{y}$, H is a sum over terms close to 0. We can see that the log-likelihood can be set arbitrarily small by decreasing ϵ. On the other hand, by using a model of the simulation at $\boldsymbol{\theta}$

$$\log \pi_{\boldsymbol{\epsilon}}(\mathbf{y}|\boldsymbol{\theta}) \approx -\frac{1}{2}\log(\sigma^2_{\boldsymbol{\theta}} + \epsilon^2) - \frac{(\mathbf{y} - \mu_{\boldsymbol{\theta}})^2}{2(\sigma^2_{\boldsymbol{\theta}} + \epsilon^2)} \quad (10)$$

For the SL model, ϵ acts as a smoothing term and can be set to small values with little change to the log-likelihood, as long as the SL estimators are fit appropriately. This insensitivity to ϵ will be used in Section 4 for estimating gradients of the ABC likelihood. Before describing HABC in full detail however, we now explain how scaling Hamiltonian dynamics in Bayesian learning can be accomplished using stochastic gradients from batched data.

3 SCALING BAYESIAN INFERENCE USING HAMILTONIAN DYNAMICS

Scaling Bayesian inference algorithms to massive datasets is necessary for their existing relevance in the so-called *big data* era. We now review the role stochastic gradient methods combined with Hamiltonian dynamics have played in recent advances in scaling Bayesian inference. Most importantly, these methods have combined the ability of HMC to explore high-dimensional parameter spaces with the computational efficiency of using stochastic gradients based on small batches of the full dataset. After an overview of HMC, we will briefly describe stochastic gradient Hamiltonian dynamics (SGHDs), starting with using Langevin dynamics [19], then HMC with friction [3], and finally HMC with thermostats [4]. We will then make the connection between SGHDs and HABC in Section 4.

3.1 Hamiltonian Monte Carlo

Hamiltonian dynamics are often necessary to adequately explore the target distribution of high-dimensional parameter spaces. By proposing parameters that are far from the current location and yet have high acceptance probability, Hamiltonian Monte Carlo [5, 12] can efficiently avoid random walk behavior that can render proposals in high-dimensions painfully slow to mix.

HMC simulates the trajectory of a particle along a frictionless surface, using random initial momentum $\boldsymbol{\rho}$ and position $\boldsymbol{\theta}$. The Hamiltonian function computes the energy of the system and the dynamics govern how the momentum and position change over time. The continuous Hamiltonian dynamics can be simulated by discretizing time into small steps η. If η is small, the value of $\boldsymbol{\theta}$ at the end of a simulation can be used as proposals within the Metropolis-Hastings algorithm. Hamiltonian dynamics should propose $\boldsymbol{\theta}$ that are always accepted, but errors due to discretization may require a Metropolis-Hastings correction. It is this correction step that SGHD algorithms want to avoid as it requires computing the log-likelihood over the full data set.

More formally, the Hamiltonian $H(\boldsymbol{\theta}, \boldsymbol{\rho}) = U(\boldsymbol{\theta}) + K(\boldsymbol{\rho})$ is a function of the current potential energy $U(\boldsymbol{\theta})$ and kinetic energy $K(\boldsymbol{\rho}) = \boldsymbol{\rho}^T M^{-1}\boldsymbol{\rho}/2$ (M is a diagonal matrix of masses which for presentation are set to 1). The potential energy is defined by negative log joint density of the data and prior:

$$U(\boldsymbol{\theta}) = -\log\pi(\boldsymbol{\theta}) - \sum_{i=1}^{N} \log\pi(\mathbf{y}_i|\boldsymbol{\theta}) \quad (11)$$

The Hamiltonian dynamics follow

$$d\boldsymbol{\theta} = \boldsymbol{\rho}dt \qquad d\boldsymbol{\rho} = -\nabla U(\boldsymbol{\theta})dt \quad (12)$$

in simulation $dt = \eta$.

3.2 Stochastic Gradient Hamiltonian Dynamics

If the log-likelihood over the full data set is replaced with a batch estimate, as is done for the following *stochastic gradient Hamiltonian dynamics* (SGHDs) algorithms, then the error in simulating the Hamiltonian dynamics comes not only from the discretization, but from the variance of the stochastic gradient. As long as this error is controlled, either by using small steps η (SGLD), or adding friction terms A (SGHMC), or using a thermostat ξ (SGNHT), the expensive MH correction step can be avoided and values of $\boldsymbol{\theta}$ from the Hamiltonian dynamics can be used as samples from the posterior.

SGHDs replace the full potential energy and its gradient with a batch approximation:

$$\hat{U}(\boldsymbol{\theta}) = -\log \pi(\boldsymbol{\theta}) - \frac{N}{n} \sum_{i=h_1}^{h_n} \log \pi(\mathbf{y}_i|\boldsymbol{\theta}) \quad (13)$$

$$\nabla \hat{U}(\boldsymbol{\theta}) = -\nabla \log \pi(\boldsymbol{\theta}) - \frac{N}{n} \sum_{i=h_1}^{h_n} \nabla \log \pi(\mathbf{y}_i|\boldsymbol{\theta}) \quad (14)$$

where n is the batch size, and h_i are indices chosen randomly without replacement from $[1, N]$ (i.e. it defined a random batch).

Stochastic gradient Langevin dynamics [19] performs one full leap-frog step of HMC. Starting with a half step for the momentum, the update for $\boldsymbol{\theta}$ is

$$\boldsymbol{\rho}_t \sim \mathcal{N}(0, \boldsymbol{I}_p) \quad (15)$$

$$\boldsymbol{\rho}_{t+\frac{1}{2}} = \boldsymbol{\rho}_t - \eta \nabla \hat{U}(\boldsymbol{\theta}_t)/2 \quad (16)$$

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \eta \boldsymbol{\rho}_{t+\frac{1}{2}} \quad (17)$$

Tt is not necessary to include $\boldsymbol{\rho}$ in the updates since there is only one step:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \eta \mathcal{N}(0, \boldsymbol{I}_p) - \eta^2 \nabla \hat{U}(\boldsymbol{\theta}_t)/2 \quad (18)$$

One of the potential drawbacks of SGLD is that the momentum term is *refreshed* for every update of the parameters, and since this means the parameter update only uses the current gradient approximation, it limits the benefits of using Hamiltonian dynamics. On the other hand, this also prevents SGLD from accumulating errors in the Hamiltonian dynamics.

Stochastic Gradient HMC (SGHMC) [3] avoids $\boldsymbol{\rho}$ refreshment altogether. By applying HMC directly using the stochastic approximation $\hat{U}$ and $\nabla \hat{U}$, which the authors call *naive SGHMC*, the variance of the gradient will introduce errors that left unaddressed will result in sampling from the incorrect target distribution. Under the assumption that $\nabla \hat{U}(\boldsymbol{\theta}) = \nabla U(\boldsymbol{\theta}) + \mathcal{N}(\mathbf{0}, \boldsymbol{V}_{\boldsymbol{\theta}})$, where $\boldsymbol{V}_{\boldsymbol{\theta}}$ is the covariance of the gradient approximation, and updates $\boldsymbol{\rho}_{t+1} = \boldsymbol{\rho}_t + \Delta \boldsymbol{\rho}_t$ and $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \eta \boldsymbol{\rho}_{t+1}$, the change in momenta $\Delta \boldsymbol{\rho}$ from one full step is

$$-\eta \left(\nabla U(\boldsymbol{\theta}) + \mathcal{N}(\mathbf{0}, \boldsymbol{V}_{\boldsymbol{\theta}}) \right) = -\eta \nabla U(\boldsymbol{\theta}) + \mathcal{N}\left(\mathbf{0}, \eta^2 \boldsymbol{V}_{\boldsymbol{\theta}}\right) \quad (19)$$

By adding a friction term $\boldsymbol{B}$ to $\Delta \boldsymbol{\rho}$ proportional to $\boldsymbol{V}_{\boldsymbol{\theta}}$, the correction step can be avoided

$$\Delta \boldsymbol{\rho} = -\eta \boldsymbol{B} \boldsymbol{\rho}_t - \eta \nabla U(\boldsymbol{\theta}_t) + \mathcal{N}(\mathbf{0}, 2\eta \boldsymbol{B}) \quad (20)$$

where $\boldsymbol{B} = \frac{1}{2} \eta \boldsymbol{V}_{\boldsymbol{\theta}_t}$. In practice, since we can only estimate $\boldsymbol{B}$ by some $\hat{\boldsymbol{B}}$ and can only compute $\hat{U}$, a user defined friction term $\boldsymbol{C}$ is used (with $\boldsymbol{C} - \hat{\boldsymbol{B}}$ is semi-positive definite). Thus the updates used for $\Delta \boldsymbol{\rho}$ for SGHMC:

$$-\eta \boldsymbol{C} \boldsymbol{\rho}_t - \eta \nabla \hat{U}(\boldsymbol{\theta}_t) + \mathcal{N}\left(\mathbf{0}, 2\eta(\boldsymbol{C} - \hat{\boldsymbol{B}})\right) \quad (21)$$

In our experiments we compute an online estimate $\hat{\boldsymbol{V}}$ and set $\boldsymbol{C} = c\boldsymbol{I}_p + \hat{\boldsymbol{V}}$.

Stochastic Gradient thermostats (SGT) [4] addresses the difficulty of estimating $\hat{\boldsymbol{B}}$ by introducing a scalar variable ξ who's addition to the Hamiltonian dynamics maintains the temperature of the system constant, i.e. it acts as a (Nose-Hoovier) thermostat [**?**]. The update equations remain simple: initialize $\xi = \boldsymbol{C}$ (or c), then for $t = 1 \ldots$

$$\boldsymbol{\rho}_{t+1} = \rho_t - \eta \xi_t \boldsymbol{\rho} - \eta \nabla \hat{U}(\boldsymbol{\theta}_t) + \mathcal{N}(\mathbf{0}, 2\eta_t \boldsymbol{C}) \quad (22)$$

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \eta \boldsymbol{\rho}_{t+1} \quad (23)$$

$$\xi_{t+1} = \xi_t + \eta \left(\boldsymbol{\rho}_{t+1}^T \boldsymbol{\rho}_{t+1}/p - 1 \right) \quad (24)$$

In summary, the hyperparameters required for these algorithms are η and $\boldsymbol{C}$ (for SGHMC and SGNHT only), and in practice, some way of estimating $\hat{\boldsymbol{V}}$ for SGHMC.

4 HAMILTONIAN ABC

In the application of stochastic gradient Hamiltonian dynamics to ABC, the stochasticity of the gradient due to batch sizes is instead caused by two other factors. The first factor, is due to the finite number of forward simulations that are used to approximate the log-likelihood function. We will explain below how to perform ABC gradient estimates based on finite difference approximations on two log-likelihood formulations: the ϵ-kernel density model and on a Gaussian conditional model, both are based on S random simulations. The second factor is due to the dimension of D. For large D, a finite difference estimate requires $2p$ forward simulations for each random simulation, by using a further stochastic approximation to the gradient, called Simultaneous Perturbation Stochastic Approximation (SPSA) [**?**], an unbiased estimate of the gradient can computed with $2q$ forward simulations, where ≥ 1 and usually $Q \ll D$.

To begin, we define $\boldsymbol{\theta}_d^+ = \boldsymbol{\theta} + c\boldsymbol{\Delta}_d$ and $\boldsymbol{\theta}_d^- = \boldsymbol{\theta} + c\boldsymbol{\Delta}_d$, where c is a small positive scalar, e.g. $c = 0.001$, and $\boldsymbol{\Delta}_d$ is a vector of size D that, for FD estimates is 1 for entry d and 0 otherwise. Our gradient estimate require simulating S times at $\boldsymbol{\theta}^+$ and $\boldsymbol{\theta}^-$, generating sets of pseudo-data $\{\mathbf{x}_{sd}^+\}$ and $\{\mathbf{x}_{sd}^-\}$. These are then plugged into the log-likelihood estimate for $\mathbf{y}$, which we call $\mathcal{L}$:

$$\frac{\partial \mathcal{L}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_d} \approx \frac{\mathcal{L}(\boldsymbol{\theta}_d^+) - \mathcal{L}(\boldsymbol{\theta}_d^-)}{2c} \quad (25)$$

For the ϵ-kernel density model it is easy to show that the gradient estimate is sensitive to ϵ in this case:

$$\frac{\partial \mathcal{L}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_d} \approx \frac{1}{2c\epsilon^2} \left(-\frac{1}{2}(\mathbf{y} - \mathbf{x}_{sd}^+)^2 + \frac{1}{2}(\mathbf{y} - \mathbf{x}_{sd}^-)^2 \right) \quad (26)$$

So by making ϵ arbitrarily small we can make the gradient extremely sensitive to our simulations. Intuitively, we want the gradient to be fairly stable no matter what values of ϵ we use. To do this we will replace log-likelihood with a conditional model. We can see that ϵ now can be set very small as it now acts as a smoother for our estimate, resulting in a robust gradient.

4.1 Gradients in High Dimensions

In the gradient-free setting, Spall [15] provides a stochastic approximate to the true gradient using only 2 forward simulations (function evaluations). This is in contrast to multivariate finite-difference stochastic approximation FDSA [7] requiring $2p$ evaluations.

The gradient estimate is

$$\hat{g}_t\left(\boldsymbol{\theta}_t\right) = \begin{bmatrix} \frac{y_t^+ - y_t^-}{2c_{t1}\Delta_{t1}} \\ \vdots \\ \frac{y_t^+ - y_t^-}{2c_{tp}\Delta_{tp}} \end{bmatrix} \tag{27}$$

where c_{tp} is a step-size that is usually constant for all dimensions p, but can be different (as shown in this case); $\Delta_{tp} \in -1, +1$ is a *perturbation mask* (called symmetric Bernouilli variables by Spall), i.e. $\Delta_{tp} \sim 2 * \text{Bernouilli}(0.5) - 1$; and $y_t^{\pm}$ are function evaluations:

$$y_t^+ = L\left(\boldsymbol{\theta} + c_t\Delta_t\right) \tag{28}$$
$$y_t^- = L\left(\boldsymbol{\theta} - c_t\Delta_t\right) \tag{29}$$

A way of reducing the noise in the gradient is by averaging over q draws of Δ_t:

$$\hat{g}_t\left(\boldsymbol{\theta}_t\right) = \frac{1}{q}\sum_{j=1}^{q} \hat{g}_t^j\left(\boldsymbol{\theta}_t\right) \tag{30}$$

where for each j new perturbation masks are drawn.

For ABC, the gradients we compute are stochastic because we must approximate $\pi(\mathbf{x}|\boldsymbol{\theta})$ with forward simulations. We can make the approximation arbitrarily good by running more simulations, but this is often unnecessary for sampling, because we can make use of the uncertainty in our current approximation to decide if our sample is correct or not (cite UAI2014?).

- In both cases, we rewrite the expression conditioning on a set of random seeds omega. Explain the reason for this. By observing the simulation at different theta but same seeds, we can eliminate much of the noise in the markov chain. By treating omega as part of the state, we can smoothly move along noiseless outputs for several iterations and then step.

4.2 Stochastic ABC Gradients

For low-dimensional problems a finite difference (FD) approximation to ∇U. For each dimension p, forward simulate for each random seed twice at plus c and minus c (only along dimension p): $x_{\pm}^{(s)} \sim \pi(\mathbf{x}|\boldsymbol{\theta} \pm c\Delta_p, \omega_s)$. These simulations are then plugged into the appropriate estimator (logsumsexp or SL). $\hat{U}_{K_\epsilon}$:

$$-\log\pi(\boldsymbol{\theta} \pm c\Delta_p) - \log\sum_s \pi_\epsilon(\mathbf{y}|f(\boldsymbol{\theta} \pm c\Delta_p, \omega_s)) \tag{31}$$

or $\hat{U}_{SL}$

$$-\log\pi(\boldsymbol{\theta} \pm c\Delta_p) - \log\mathcal{N}(\mathbf{y}|\mu_{\boldsymbol{\theta}\pm c\Delta_p}, \sigma^2_{\boldsymbol{\theta}\pm c\Delta_p} + \epsilon^2) \tag{32}$$

$$\hat{U}(\boldsymbol{\theta}) = -\log\pi(\boldsymbol{\theta}) - \log\pi_\epsilon(\mathbf{y}|\boldsymbol{\theta}) \tag{33}$$
$$\nabla\hat{U}(\boldsymbol{\theta}) = -\nabla\log\pi(\boldsymbol{\theta}) - \frac{N}{n}\sum_{i=h_1}^{h_n}\nabla\log\pi(\mathbf{y}_i|\boldsymbol{\theta}) \tag{34}$$

4.3 Common Random Numbers

[CITE NEAL How to view an MCMC...] Another variance reduction technique called the method of *common random numbers* (CRN) [8] sets a common seed for the random number generator (RNG) for both calls to the simulator. This technique can remove the effect of the simulator noise in the gradient estimate, leaving on the randomness of the perturbation masks as the source of noise. Consider using SPSA instead of SGD: the CRN technique is equivalent of using the same batch of data vectors to evaluate the log-likelihood which is a sensible approach.

Using the same parameter vector $\boldsymbol{\theta}$ but different random seeds:

$$\pi\left(\mathbf{y}|\boldsymbol{\theta}\right) = \int \pi_{\boldsymbol{\epsilon}}\left(\mathbf{y}|\mathbf{x}\right)\pi\left(\mathbf{x}|\boldsymbol{\theta}\right)d\mathbf{x} \tag{35}$$
$$\approx \frac{1}{S}\sum_{s=1}^{S}\pi_{\boldsymbol{\epsilon}}\left(\mathbf{y}|\mathbf{x}^{(s)}\right) \tag{36}$$

where $\mathbf{x}^{(s)} = \pi\left(\mathbf{x}|\boldsymbol{\theta}, \omega_s\right)$. Note we have passed a unique random seed ω_s into the simulator, making $\mathbf{x}^{(s)}$ a deterministic function of the simulator with parameters $\{\boldsymbol{\theta}, \omega_s\}$. It will be useful to rewrite this deterministic simulator function as f: $\mathbf{x}^{(s)} = f\left(\boldsymbol{\theta}, \omega_s\right)$, allowing us to rewrite the likelihood as

$$\pi\left(\mathbf{y}|\boldsymbol{\theta}\right) \approx \frac{1}{S}\sum_{s=1}^{S}\pi_{\boldsymbol{\epsilon}}\left(\mathbf{y}|f\left(\boldsymbol{\theta}, \omega_s\right)\right) \tag{37}$$

4.4 Algorithms

5 Demonstration

[FROM UAI] In this illustrative problem we infer the rate of an exponential distribution under a gamma prior with shape α and rate β, having N observations at the true rate

$\theta^{\star}$; this is the *exponential example* in [18]. Let $\mathbf{w}$ be a vector of N draws from an exponential distribution, i.e. $w_n \sim \text{Exp}(\theta)$. The posterior is a gamma distribution with shape $\alpha + N$ and rate $\beta + \sum w_n$. To use this problem with ABC, we use the exponential draws as the simulator and the mean of $\mathbf{w}$ as the statistic y and assume that N is known. The inference problem is therefore to sample from $p(\theta|y, \alpha, \beta, N)$.

5.1 First order CRNs

Using same seed in a gradient estimate at time t (versus not using them). This is makes sense if we compare to SG-data where a single batch is used for computing the gradient. If a FD method was used, we would use the same batch.

5.2 Second order CRNs

Using same seed in a gradient estimate at multiple time steps (versus not changing them – the previous example). Analogous to keeping the same batch for consecutive steps in SG-data.

5.3 Comparing Gradient Estimates with SL

- Can estimate ABC posterior by CLT of Gammas (sum of exponential). For large S, the variance of the SL approximates the "true" CLT approximation. Thus we can set ϵ much smaller for the SL case. The kernel log-sumexp suffers when ϵ is small, using only the closest simulation to $\mathbf{y}$. Instead the SL estimates a density around the simulations and the likelihood is less sensitive to ϵ.
- Better tolerance to low ϵ using SL.

5.4 Using variance of gradients in SGHMC

Compute online $\hat{B}$ then fix.

5.5 Visualizing Noise from Gradient versus Injected Noise

6 Experiments

6.1 Blowfly

[FROM UAI] Adult blowfly populations exhibit dynamic behavior for which several competing population models exist. In this experiment, we use observational data and a simulation model from Wood [22], based on their improvement upon previous population dynamics theory. Population dynamics are modeled using (discretized) differential equations that can produce chaotic behavior for some parameter settings. An example blowfly replicate series is shown in Figure **??**, along with times-series generated by a sample from $\pi(\boldsymbol{\theta}|\mathbf{y})$ using GPS-ABC.

In [22] there are several explanations of the population dy namics, corresponding to different simulations and parameters. We concentrate on the equation (1) in section 1.2.3 of the supplementary information, considered "a better alternative model" by the author. The population dynamics equation generates $N_1, \ldots, N_T$ using the following update rule:

$$N_{t+1} = PN_{t-\tau} \exp(-N_{t-\tau}/N_0)e_t + N_t \exp(-\delta\epsilon_t)$$

where $e_t \sim \mathcal{G}(1/\sigma_p^2, 1/\sigma_p^2)$ and $\epsilon_t \sim \mathcal{G}(1/\sigma_d^2, 1/\sigma_d^2)$ are sources of noise, and τ is an integer (not to be confused with the τ used as the MH acceptance threshold in our algorithms). In total, there are 6 parameters $\theta = \{\log P, \log \delta, \log N_0, \log \sigma_d, \log \sigma_p, \tau\}$. See [22] for further details about the significance of the parameters. We put Gaussian priors over all the parameters (with Gaussian proposal distributions), except for τ which has a Poisson prior (and a left/right increment proposal). Time-series generated with parameters from this prior distribution produce extremely varied results, some are chaotic, some are degenerate, etc. Modeling this simulator is *very* challenging.

As with any ABC problem the choice of statistics is important as it directly affects the quality of the results. It is also non-trivial and requires careful thought and sometimes trial and error. In total there are 10 statistics: the log of the mean of all 25% quantiles of $N/1000$ (4 statistics), the mean of the 25% quantiles of the first-order differences of $N/1000$ (4 statistics), and the maximal peaks of smoothed N, with 2 different thresholds (2 statistics). With these statistics it is possible to reproduce time-series that appear similar to the observations. Note that these statistics are different from Wood's, but they capture similar time-series features and are sufficient to produce credible population dynamics. [END FROM UAI]

6.2 Bayesian Autoencoders

We perform Bayesian inference on an autoencoder neural network using MNIST images as observations. We apply two inference algorithms: SGNHT and HABC. For SGNHT, we use the known gradient and for HABC we apply 2SPSA. After 1000 iterations we collect parameter vectors every 100 iterations for a total of 100 sets. (Do we use SGHMC to estimate Bhat in first 1000 iterations?). Use same batches at each iteration for SGNHT and HABC?

results: convergence of training error (batches), convergence in test error using parameters collected (record nbr of forward or backward passes required), show mean and variance of filters (how?), compare with using the last parameter sample only (stochastic optimization).

7 DISCUSSION

8 CONCLUSION

- New set of ABC algorithms for efficient exploration of posterior distribution.
- Key: opens the door to high-dimensional ABC inference.

A further complication, which we do not address on this paper, is the forward simulation might be, and for many interesting problems is, expensive. We would therefore prefer to resort to some sort of surrogate modeling [11, 21**?**] to avoid simulations whenever possible. We address this scenario in Section **??**. Shown for relatively fast simulators. Extensions to surrogates with GPS (Meeds2014GpsUai, rasmussen:2003, wilkinson:2014) can use derivative of GP for likelihood as GP of gradient. These could be matched by occasional gradient estimates (Osborne).

References

References

[1] Andrieu, C. and Roberts, G. The pseudo-marginal approach for efficient monte carlo computations. *The Annals of Statistics*, 37(2):697–725, 2009.

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

[3] Chen, Tianqi, Fox, Emily B, and Guestrin, Carlos. Stochastic gradient hamiltonian monte carlo. 2014.

[4] Ding, Nan, Fang, Youhan, Babbush, Ryan, Chen, Changyou, Skeel, Robert D, and Neven, Hartmut. Bayesian sampling using stochastic gradient thermostats. In *Advances in Neural Information Processing Systems*, pp. 3203–3211, 2014.

[5] Duane, Simon, Kennedy, Anthony D, Pendleton, Brian J, and Roweth, Duncan. Hybrid monte carlo. *Physics letters B*, 195(2):216–222, 1987.

[6] Fan, Yanan, Nott, David J, and Sisson, Scott A. Approximate bayesian computation via regression density estimation. *Stat*, 2013.

[7] Kiefer, Jack, Wolfowitz, Jacob, et al. Stochastic estimation of the maximum of a regression function. *The Annals of Mathematical Statistics*, 23(3):462–466, 1952.

[8] Kleinman, Nathan L, Spall, James C, and Naiman, Daniel Q. Simulation-based optimization with stochastic approximation using common random numbers. *Management Science*, 45(11):1570–1578, 1999.

[9] Marin, J.-M., Pudlo, P., Robert, C.P., and Ryder, R.J. Approximate bayesian computational methods. *Statistics and Computing*, 22:1167–1180, 2012.

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

[11] Meeds, Edward and Welling, Max. GPS-ABC: Gaussian process surrogate approximate bayesian computation. *Uncertainty in AI*, 2014.

[12] Neal, Radford M. Mcmc using hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*, 2, 2011.

[13] Sisson, SA, Fan, Y, and Tanaka, Mark M. Sequential monte carlo without likelihoods. *Proceedings of the National Academy of Sciences*, 104(6):1760, 2007.

[14] Sisson, Scott A and Fan, Yanan. Likelihood-free markov chain monte carlo. *Arxiv preprint arXiv:1001.2058*, 2010.

[15] Spall, James C. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *Automatic Control, IEEE Transactions on*, 37(3):332–341, 1992.

[16] Spall, James C. Adaptive stochastic approximation by the simultaneous perturbation method. *Automatic Control, IEEE Transactions on*, 45(10):1839–1853, 2000.

[17] Turner, Brandon M. and Sederberg, Per B. A generalized, likelihood-free method for posterior estimation. *Psychonomic Bulletin & Review*, 21(2):227–250, 2014.

[18] Turner, Brandon M and Van Zandt, Trisha. A tutorial on approximate bayesian computation. *Journal of Mathematical Psychology*, 56(2):69–85, 2012.

[19] Welling, Max and Teh, Yee W. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pp. 681–688, 2011.

[20] Wilkinson, R. 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] Wilkinson, R. Accelerating abc methods using gaussian processes. *AISTATS*, 2014.

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