Title

Variational Hamiltonian Monte Carlo via Score Matching

Permalink

Journal

ISSN

Authors

Zhang, Cheng
Shahbaba, Babak
Zhao, Hongkai

Publication Date

DOI

Peer reviewed

Variational Hamiltonian Monte Carlo via Score Matching

Cheng Zhang   CHENGZ4@UCI.EDU
Department of Mathematics, UC Irvine, Irvine, CA 92697

Babak Shahbaba   BABAKS@UCI.EDU
Department of Statistics, UC Irvine, Irvine, CA 92697

Hongkai Zhao   ZHAO@MATH.UCI.EDU
Department of Mathematics, UC Irvine, Irvine, CA 92697

Abstract

Traditionally, the field of computational Bayesian statistics has been divided into two main subfields: variational methods and Markov chain Monte Carlo (MCMC). In recent years, however, several methods have been proposed based on combining variational Bayesian inference and MCMC simulation in order to improve their overall accuracy and computational efficiency. This marriage of fast evaluation and flexible approximation provides a promising means of designing scalable Bayesian inference methods. In this paper, we explore the possibility of incorporating variational approximation into a state-of-the-art MCMC method, Hamiltonian Monte Carlo (HMC), to reduce the required gradient computation in the simulation of Hamiltonian flow, which is the bottleneck for many applications of HMC in big data problems. To this end, we use a *free-form* approximation induced by a fast and flexible surrogate function based on single-hidden layer feedforward neural networks. The surrogate provides sufficiently accurate approximation while allowing for fast exploration of parameter space, resulting in an efficient approximate inference algorithm. We demonstrate the advantages of our method on both synthetic and real data problems.

1. Introduction

Bayesian inference has been successful in modern data analysis. Given a probabilistic model for the underlying mechanism of the observed data, Bayesian methods properly quantify uncertainty and reveal the landscape or global structure of parameter space. While conceptually simple, exact posterior inference in many Bayesian models is often intractable. Therefore, in practice, people often resort to approximation methods among which Markov chain Monte Carlo (MCMC) and variational Bayesian (VB) are the two most popular choices.

The MCMC approach is based on drawing a series of correlated samples with guaranteed convergence to the target distribution. Therefore, MCMC methods are asymptotically unbiased. Simple methods such as random-walk Metropolis (Metropolis et al., 1953), however, often suffer from slow mixing (due to their random walk nature) when encountering complicated models with strong dependencies among parameters. Introducing an auxiliary momentum variable, Hamiltonian Monte Carlo (HMC) (Duane et al., 1987; Neal, 2011) reduces the random walk behavior by proposing states following a Hamiltonian flow which preserves the target distribution. By incorporating the geometric information of the target distribution, e.g., the gradient, HMC is able to generate distant proposals with high acceptance probabilities, enabling more efficient exploration of the parameter space than standard random-walk proposals.

A major bottleneck of HMC, however, is the computation of the gradient of the potential energy function in order to simulate the Hamiltonian flow. As the datasets involved in many practical tasks, such as "big data" problems, usually have millions to billions of observations, such gradient computations are infeasible since they need full scans of the entire dataset. In recent years, many attempts have been made to develop scalable MCMC algorithms that can cope with very large data sets (Welling & Teh, 2011; Ahn et al., 2012; Chen et al., 2014; Ding et al., 2014). The key idea of these methods stems from stochastic optimization where noisy estimates of the gradient based on small

subsets of the data are utilized to scale up the algorithms. The noise introduced by subsampling, however, could lead to non-ignorable loss of accuracy, which in turn hinders the exploration efficiency of standard MCMC approaches (Betancourt, 2015).

The main alternative to MCMC is variational Bayes inference (Jordan et al., 1999; Wainwright & Jordan, 2008). As a deterministic approach, VB transforms Bayesian inference into an optimization problem where a parametrized distribution is introduced to fit the target posterior distribution by minimizing the Kullback-Leibler (KL) divergence with respect to the variational parameters. Compared to MCMC methods, VB introduces bias but is usually faster.

A natural question would be: can we combine both methods to mitigate the drawbacks and get the best of both worlds? The first attempt in this direction was proposed by (de Freitas et al., 2001) where a variational approximation was used as proposal distribution in a block Metropolis-Hasting (MH) MCMC kernel to locate the high probability regions quickly, thus facilitating convergence. Recently, a new synthesis of variational inference and Markov chain Monte Carlo methods has been explored in (Salimans et al., 2015) where one or more steps of MCMC are integrated into variational approximation. The extra flexibility from MCMC steps provides a rich class of distributions to find a closer fit to the exact posterior.

In this work, we explore the possibility of utilizing variational approximation to speed up HMC for problems with large scale datasets, by reducing the cost of gradient computation. The idea is to incorporate the fast variational approximation into the simulation of Hamiltonian flow so that the number of potential energy (or likelihood) evaluations required to achieve a reasonably accurate approximation can be drastically reduced. To this end, we approximate the potential energy function by training a computationally fast neural network surrogate via score matching (Hyvärinen, 2005). The training data are collected while the "modified" HMC sampler (defined based on the surrogate induced Hamiltonian flow) explores the parameter space. This variational based training and optimization allows an implicit subsampling procedure that can guarantee effective approximation of the large scale landscape while removing redundancy and noise in the data. Rather than annealing the stepsizes, as commonly used in subsampling-based methods, the stepsizes in simulating the surrogate induced Hamiltonian flow can be the same as that of standard HMC while keeping a comparable acceptance probability. Therefore, we do not have to sacrifice the exploration efficiency of standard HMC. Compared to traditional *fixed-form* variational approximations, the surrogate induced distribution serves as a *free-form* variational approximation that is more flexible and thus can fit the target distribution better.

Our paper is organized as follows. In section 2, we introduce the two ingredients related to our method: Hamiltonian Monte Carlo and *fixed-form* variational Bayesian. Section 3 presents our method, termed Variational Hamiltonian Monte Carlo (VHMC). We demonstrate the efficiency of VHMC in a number of experiments in section 4 and conclude in section 5.

2. Background

2.1. Hamiltonian Monte Carlo

In general formulation of Bayesian inference, a set of independent observations $Y = \{y_1, \ldots, y_N\}$ are modeled by an underlying distribution $p(y|\theta)$ with unknown parameter θ. Given a prior distribution of $\theta \sim p(\theta)$, the posterior distribution is given by Bayesian formula

$$p(\theta|Y) = \frac{p(Y|\theta)p(\theta)}{p(Y)} \propto \prod_{n=1}^{N} p(y_n|\theta) \cdot p(\theta) \quad (1)$$

To construct the Hamiltonian dynamical system, the position-dependent potential energy function is defined as the negative log unnormalized posterior density

$$U(\theta) = -\sum_{n=1}^{N} \log p(y_n|\theta) - \log p(\theta) \quad (2)$$

and the kinetic energy function is defined as a quadratic function of an auxiliary momentum variable r: $K(r) = r^T M^{-1} r$, where M is a mass matrix and is often set to identity, I. The fictitious Hamiltonian, therefore, is defined as the total energy function of the system $H(\theta, r) = U(\theta) + K(r)$. As one of the state-of-the-art MCMC methods, Hamiltonian Monte Carlo suppresses random walk behavior by simulating the Hamiltonian dynamical system to propose distant states with high acceptance probabilities. That is, in order to sample from the posterior distribution $p(\theta|Y)$, HMC augments the parameter space and generates samples from the joint distribution of (θ, r)

$$\pi(\theta, r) \propto \exp(-U(\theta) - K(r)) \quad (3)$$

Notice that θ and r are separated in (3), we can simply drop the momentum samples r and the θ samples follow the marginal distribution which is exactly the target posterior.

To generate proposals, HMC simulates the Hamiltonian flow governed by the following differential equations

$$\frac{d\theta}{dt} = \frac{\partial H}{\partial r} = M^{-1} r \quad (4)$$

$$\frac{dr}{dt} = -\frac{\partial H}{\partial \theta} = -\nabla_\theta U(\theta) \quad (5)$$

Algorithm 1 Hamiltonian Monte Carlo

Input: Starting position $\theta^{(1)}$ and step size ϵ
for $t = 1, 2, \ldots, T$ **do**
    Resample momentum r
    $r^{(t)} \sim \mathcal{N}(0, M)$
    $(\theta_0, r_0) = (\theta^{(t)}, r^{(t)})$
    Simulate discretization of Hamiltonian dynamics:
    for $l = 1$ **to** L **do**
        $r_{l-1} \leftarrow r_{l-1} - \frac{\epsilon}{2}\nabla_\theta U(\theta_{l-1})$
        $\theta_l \leftarrow \theta_{l-1} + \epsilon M^{-1} r_{l-1}$
        $r_l \leftarrow r_l - \frac{\epsilon}{2}\nabla_\theta U(\theta_l)$
    end for
    $(\theta^*, r^*) = (\theta_L, r_L)$
    Metropolis-Hasting correction:
    $u \sim \text{Uniform}[0, 1]$
    $\rho = \exp[H(\theta^{(t)}, r^{(t)}) - H(\theta^*, r^*)]$
    if $u < \min(1, \rho)$ **then**
        $\theta^{(t+1)} = \theta^*$
    else
        $\theta^{(t+1)} = \theta^{(t)}$
    end if
end for

Over a period t, also called trajectory length, (4) and (5) together define a map $\phi_t : (\theta_0, r_0) \mapsto (\theta^*, r^*)$ in the extended parameter space, from the starting state to the end state. As implied by a Hamiltonian flow, ϕ_t is reversible, volume-preserving and also preserves the Hamiltonian $H(\theta_0, r_0) = H(\theta^*, r^*)$. These allow us to construct π-invariant Markov chains whose proposals will always be accepted. In practice, however, (4) and (5) are not analytically solvable and we need to resort to numerical integrators. As a symplectic integrator, the *leapfrog* scheme (see Algorithm 1) maintains reversibility and volume preservation and hence is a common practice in HMC literatures. The bias introduced through the discretization needs to be corrected in an Metroplis-Hasting (MH) step. However, we can control the stepsizes to maintain high acceptance probabilities even for distant proposals.

In recent years, many variants of HMC have been developed to make the algorithm more flexible and generally applicable in a variety of settings. For example, methods proposed in (Hoffman & Gelman, 2011; Wang et al., 2013) enable automatically tuning of hyper-paramters such as the stepsize ϵ and the number of *leapfrog* steps L, saving the amount of tuning-related headaches. Riemannian Manifold HMC (Girolami & Calderhead, 2011) further improves standard HMC's efficiency by automatically adapting to local structures using Riemanian geometry of parameter space. These adaptive techniques could be potentially combined with our proposed method which focuses on reducing the computational complexity.

2.2. Fixed-form Variational Bayes

Instead of running a Markov chain, we can approximate the intractable posterior distribution with a more convenient and tractable distribution. A popular approach of obtaining such an approximation is *fixed-form variational Bayes* (Honkela et al., 2010; Saul & Jordan, 1996; Salimans & Knowles, 2013) where a parametrized distribution $q_\eta(\theta)$ is proposed to approximate the target posterior $p(\theta|Y)$ by minimizing the KL divergence

$$
\begin{aligned}
D_{KL}(q_\eta(\theta)||p(\theta|Y)) &= \int q_\eta(\theta) \log\left(\frac{q_\eta(\theta)}{p(\theta|Y)}\right) d\theta \\
&= \log(p(Y)) + \int q_\eta(\theta) \log\left(\frac{q_\eta(\theta)}{p(\theta, Y)}\right) d\theta \quad (6)
\end{aligned}
$$

since $\log(p(Y))$ is a constant (used extensively in model selection), it suffices to minimize the second term in (6). Usually, $q_\eta(\theta)$ is chosen from the exponential family of distributions with the following canonical form:

$$
q_\eta(\theta) = \exp[T(\theta)\eta - A(\eta)]\nu(\theta) \quad (7)
$$

where $T(\theta)$ is a row vector of sufficient statistics, $A(\eta)$ is for normalization and $\nu(\theta)$ is a base measure. The column vector η is often called the natural parameters of the exponential family distribution $q_\eta(\theta)$. Taking this approach and substituting into (6), we now have a parametric optimization problem in η:

$$
\hat{\eta} = \arg\min_\eta \mathbb{E}_{q_\eta(\theta)}[\log q_\eta(\theta) - \log p(\theta, Y)] \quad (8)
$$

The above optimization problem can be solved using gradient-based optimization or fix-point algorithms if $\mathbb{E}_{q_\eta(\theta)}[\log q_\eta(\theta)]$, $\mathbb{E}_{q_\eta(\theta)}[\log p(\theta, Y)]$ and its derivatives with respect to η can be evaluated analytically. Without assuming posterior independence and requiring conjugate exponential models, posterior approximations of this type are usually much more accurate than a factorized approximation following the mean-field assumptions. However, the requirement of being able to analytically evaluate those quantities mentioned above is also very restrictive. To mitigate these limitations, (Salimans & Knowles, 2013) proposed a new optimization algorithm which relates (8) to stochastic linear regression. To reveal the connection, the posterior approximate (7) is relaxed and rewritten in the unnormalized form

$$
\tilde{q}_{\tilde{\eta}}(\theta) = \exp[\tilde{T}(\theta)\tilde{\eta}]\nu(\theta) \quad (9)
$$

where the nonlinear normalizer $A(\eta)$ is removed and the vectors of sufficient statistics and natural parameters are augmented, i.e. $\tilde{T}(\theta) = (1, T(\theta))$, $\tilde{\eta} = (\eta_0, \eta')'$. The unnormalized version of KL divergence is utilized to deal with $\tilde{q}_{\tilde{\eta}}(\theta)$ and achieves its minimum at

$$
\tilde{\eta} = \mathbb{E}_q[\tilde{T}(\theta)'\tilde{T}(\theta)]^{-1} \mathbb{E}_q[\tilde{T}(\theta)' \log p(\theta, Y)] \quad (10)
$$

which resembles the maximum likelihood estimator for linear regression. Based on this observation, (Salimans & Knowles, 2013) derived a stochastic approximation algorithm using (10) as a fixed point update and approximating the involved expectations by weighted Monte Carlo.

In the next section, we will discuss how the variational Bayes approach can be actually utilized to accelerate HMC. For this, we construct a fast and accurate approximation for the computationally expensive potential energy function. The approximation is provided by variational Bayes and is incorporated in the simulation of Hamiltonian flow.

3. Variational Hamiltonian Monte Carlo

Besides subsampling, an alternative approach that can save computation cost is to construct fast and accurate surrogate functions for the expensive potential energy functions (Liu, 2001; Neal, 2011). As one of the commonly used models for emulating expensive-to-evaluate functions, Gaussian process (GP) is used in (Rasmussen, 2003) to approximate the potential energy and its derivatives based on true values of these quantities collected during an initial exploratory phase. However, a major drawback of GP-based surrogate methods is that inference time grows cubically in the size of training set due to the necessity of inverting a dense covariance matrix. This is especially crucial in high dimensional spaces, where large training sets are often needed before a reasonable level of approximation accuracy is achieved. Our goal, therefore, is to develop a method that can scale to large training set while still maintaining a desired level of flexibility. For this purpose, we propose to use neural networks along with efficient training algorithms to construct surrogate functions. A typical single-hidden layer feedforward neural network (SLFN) with scalar output is defined as

$$z(\theta) = \sum_{i=1}^{s} v_i \sigma(w_i \cdot \theta + d_i) + b \tag{11}$$

where w_i, d_i and v_i are the input weight vector, bias and output weight for the ith hidden neuron, σ is a nonlinear activation function and b is the output bias. Given a training dataset

$$\mathcal{T} := \{(\theta_n, t_n)\}_{n=1}^{N} \in \mathbb{R}^d \times \mathbb{R} \tag{12}$$

the estimates of weights and bias can be obtained by minimizing the mean square error (MSE) cost function. To save training time, randomly assigned input weights $\{w_i\}_{i=1}^{s}$ and bias $\{d_i\}_{i=1}^{s}$ are suggested in (Ferrari & Stengel, 2005; Huang et al., 2006b) where the optimization is reduced to a linear regression problem which has a fast least square solution. Unlike a standard Gaussian process, the above neural network based surrogate scales linearly in the size of training data, and cubically in the number of hidden neurons. This allows us to explicitly balance evaluation time and model capacity.

3.1. Surrogate Induced Hamiltonian Flow

The neural network surrogate can be used to define a surrogate induced Hamiltonian flow which satisfies the following equations:

$$\frac{d\theta}{dt} = \frac{\partial \tilde{H}}{\partial r} = M^{-1}r \tag{13}$$

$$\frac{dr}{dt} = -\frac{\partial \tilde{H}}{\partial \theta} = -\nabla_\theta z(\theta) \tag{14}$$

where the modified Hamiltonian is $\tilde{H}(\theta, r) = z(\theta) + K(r)$. Similar to the true Hamiltonian flow, the surrogate induced Hamiltonian flow generates proposals from the joint distribution $\tilde{\pi}(\theta, r) \propto \exp(-z(\theta) - K(r))$ and θ thus follows the marginal distribution

$$q_{\tilde{v}}(\theta) \propto \exp(-z(\theta)) = \exp[-\sum_{i=1}^{s} v_i \sigma(w_i \theta + d_i) - b] \tag{15}$$

where $\tilde{v} = (-b, -v')'$.

3.2. Free-form Variational Bayes

Since our neural network surrogates approximate the potential energy function, the underlying distribution $q_{\tilde{v}}(\theta)$ then approximates the target posterior distribution. Denote the vector of outputs from the hidden layer by $\Psi(\theta) = [\Psi_1(\theta), \dots, \Psi_s(\theta)]$, $\Psi_i = \sigma(w_i \theta + d_i)$, $i = 1, \dots, s$. Then, (15) can be rewritten in a similar form to the unnormalized *fixed-form* approximation (9)

$$q_{\tilde{v}}(\theta) \propto \exp[\tilde{\Psi}(\theta)\tilde{v}] \tag{16}$$

where $\tilde{\Psi}(\theta) = (1, \Psi(\theta))$. Here, the vector of outputs from the hidden layer plays a similar role as the vector of sufficient statistics. Moreover, a set of randomly assigned input weights and bias composed linearly inside the nonlinear activation function forms a set of basis functions whose spanning space has been shown to approximate a rich class of functions arbitrarily well (Huang et al., 2006a). As a result, the surrogate induced approximation (16) is often more flexible than the *fixed-form* approximation. Unlike the *fixed-form* approximation, the surrogate induced approximation method generally does not allow for drawing samples directly. However, we can simulate the surrogate induced Hamiltonian flow (13) and (14) to generate proposals and collect the values of interest, such as the potential energy function and its derivatives, as training data to improve the surrogate approximation. Since approximation (16) does not take any specific form of the exponential family of distributions, this really leads to a *free-form*

variational Bayesian approach. By choosing a proper number of hidden neurons, the *free-form* variational Bayesian approach provides an implicit subsampling procedure that can effectively remove redundancy and noise in the data while striking a good balance between computation cost and approximation accuracy of the underlying distribution.

3.3. Score Matching

Note that both the surrogate induced distribution and the posterior distribution are known up to a constant. Therefore, we use score matching (Hyvärinen, 2005), a well known strategy to estimate unnormalized models, to train our *free-form* variational approximation. Suppose that we have collected training data of size t from the Markov chain history

$$\mathcal{T}_s^{(t)} := \{(\theta_n, \nabla_\theta U(\theta_n))\}_{n=1}^t \in \mathbb{R}^d \times \mathbb{R}^d \quad (17)$$

where θ_n is the n-th sample. The estimator of the output weight vector can be obtained by optimizing the empirical square distance between the gradients of surrogate and potential energy, so-called score functions, plus an additional regularization term:

$$\hat{v} = \arg\min_v \sum_{n=1}^t \|\nabla_\theta z(\theta_n) - \nabla_\theta U(\theta_n)\|^2 + \lambda\|v\|^2 \quad (18)$$

which has an online updating formula summarized in the following proposition 1, see Appendix A in the supplementary material for a detailed proof and a brief introduction to score matching as well.

Proposition 1 *Suppose our current estimator of the output weight vector is $v^{(t)}$ based on the current training dataset $\mathcal{T}_s^{(t)} := \{(\theta_n, \nabla_\theta U(\theta_n))\}_{n=1}^t \in \mathbb{R}^d \times \mathbb{R}^d$ using s hidden neurons. Given a new training data point $(\theta_{t+1}, \nabla_\theta U(\theta_{t+1}))$, the updating formula for the estimator is given by*

$$v^{(t+1)} = v^{(t)} + W^{(t+1)}(\nabla_\theta U(\theta_{t+1}) - A_{t+1}v^{(t)}) \quad (19)$$

where

$$W^{(t+1)} = C^{(t)}A'_{t+1}\left[I_d + A_{t+1}C^{(t)}A'_{t+1}\right]^{-1}$$
$$A_{t+1} = (A_1(\theta_{t+1}), \dots, A_s(\theta_{t+1}))$$

with $A_i(\theta_{t+1}) := \sigma'(w_i \cdot \theta_{t+1} + d_i)w_i$, *and* $C^{(t)}$ *can be updated by* Sherman-Morrison-Woodbury *formula:*

$$C^{(t+1)} = C^{(t)} - W^{(t+1)}A_{t+1}C^{(t)} \quad (20)$$

The estimator and inverse matrix can be initialized as $v^{(0)} = 0$, $C^{(0)} = \frac{1}{\lambda}I_s$. The online learning can be achieved by storing the inverse matrix C and performing

the above updating formulas which cost $\mathcal{O}(d^3 + ds^2)$ computation and $\mathcal{O}(s^2)$ storage, independent of t.

There are two main advantages of using score matching. First, the drift term b in our neural network surrogate is automatically removed. Notice that b is only an auxiliary variable to improve approximation and is not necessary in neither the simulation of surrogate induced Hamiltonian flow (13) and (14) nor the surrogate induced distribution (16). Eliminating b could save some computation. Second, the gradient gives more information than a single function value and thus reduces the required number of training data points to achieve reasonable accuracy.

3.4. Variational HMC in Practice

The neural network based surrogate is capable of approximating the potential energy function well when there is enough training data. However, the approximation could be poor when only few training data are available which is true in the early stage of the Markov chain simulations. To alleviate this issue, we propose to add an auxiliary regularizer which provides enough information for the sampler at the beginning and gradually diminishes as the surrogate becomes increasingly accurate. Here, we use the Laplace's approximation to the potential energy function but any other fast VB approximations could be used. The regularized surrogate approximation then takes the form

$$V_t(\theta) = \mu_t z_t(\theta) + \frac{1}{2}(1 - \mu_t)(\theta - \theta^L)'\nabla_\theta^2 U(\theta^L)(\theta - \theta^L)$$

where $\mu_t \in [0, 1]$ is a smooth monotone function monitoring the transition from the Laplace's approximation to the surrogate approximation. Refining the surrogate approximation by acquiring training data from simulating the regularized surrogate induced Hamiltonian flow, we arrive at an efficient approximate inference method: *Variational Hamiltonian Monte Carlo (VHMC)* (Algorithm 2).

In practice, the surrogate approximation may achieve sufficient quality and an early stopping could save us from inefficient updating of the output weight vector. In fact, the stopping time t_0 serves as a knob to control the desired approximation quality. Before stopping, VHMC acts as a *free-form* variational Bayes method that keep improving itself by collecting training data from the history of the Markov chain. After stopping, VHMC performs as a standard HMC algorithm which samples from the surrogate induced distribution. VHMC successfully combines the advantages of both variational Bayes and Hamiltonian Monte Carlo, resulting in higher computational efficiency (compared to HMC) and better approximation (compared to VB).

Algorithm 2 Variational Hamiltonian Monte Carlo

    Input: Regularization coefficient λ, transition function μ_t, number of hidden neurons s, starting position $\theta^{(1)}$ and HMC parameters
    Find the Maximum A Posterior θ^L and compute the Hessian matrix $\nabla_\theta^2 U(\theta^L)$
    Randomly assign the input weights and bias: $\{w_i\}_{i=1}^s$ and $\{d_i\}_{i=1}^s$
    for $t = 1, 2, \ldots, T$ **do**
        Propose (θ^*, r^*) with regularized surrogate induced Hamiltonian flow, using $\nabla_\theta V_t(\theta)$
        Perform Metropolis-Hasting step according to the underlying distribution $\pi_t \sim \exp(-V_t(\theta) - K(r))$
        if New state is accepted & $t < t_0$ **then**
            Acquire new training data point $(\theta_{t+1}, \nabla_\theta U(\theta_{t+1}))$
            Update the output weight estimate $v^{(t+1)} \leftarrow$ (23) and the inverse matrix $C^{(t+1)} \leftarrow$ (20)
        else
            $v^{(t+1)} = v^{(t)}, \; C^{(t+1)} = C^{(t)}$
        end if
    end for

---

4. Experiments

4.1. A Beta-binomial Model for Overdispersion

We first demonstrate the performance of our variational Hamiltonian Monte Carlo method on a toy example from (Albert, 2009), which considers the problem of estimating the rates of death from stomach cancer for the largest cities in Missouri. The data is available from the R package LearnBayes which consists of 20 pairs (n_j, y_j) where n_j records the number of individuals that were at risk for cancer in city j, and y_j is the number of cancer deaths that occurred in that city. The counts y_j are overdispersed compared to what would be expected under a binomial model with a constant probability, so (Albert, 2009) assumes a beta-binomial model with mean m and precision K:

$$p(y_j|m, K) = \binom{n_j}{y_j} \frac{B(Km + y_j, K(1-m) + n_j - y_j)}{B(Km, K(1-m))}$$

and assigns the parameters the following improper prior:

$$p(m, K) \propto \frac{1}{m(1-m)} \frac{1}{(1+K)^2}$$

The resulting posterior is extremely skewed and a reparameterization $x_1 = \text{logit}(m), \; x_2 = \text{logit}(K)$ is proposed to ameliorate this issue.

We choose $\mu_t = 1 - \exp(-t/200)$ as our transition schedule and set up the HMC parameter to achieve around 85% acceptance. We run the variational Hamiltonian Monte Carlo long enough so that we can estimate the full approximation qualify of our surrogate. We then train the neural

network based surrogate using different numbers of hidden neurons and examine the resulting KL-divergence and score matching squared distance to the true posterior density. As we can see from Figures 1 and 2, the neural network based surrogate indeed offers a high quality approximation and becomes more accurate as the number of hidden neurons increases. The surrogate induced Hamiltonian flow effectively explores the parameter space and transfers information from the posterior to the surrogate.
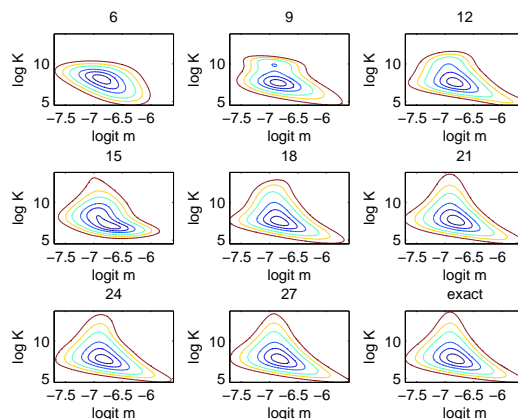

Figure 1. Approximate posteriors for a varying number of hidden neurons. Exact posterior at bottom right.
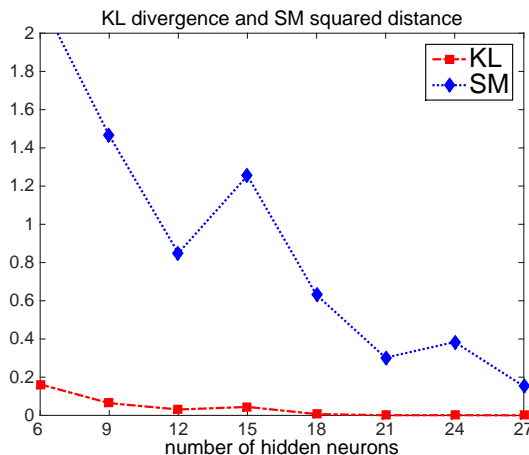

Figure 2. KL-divergence and score matching squared distance between the surrogate approximation and the exact posterior density using an increasing number of hidden neurons.

4.2. Bayesian Probit Regression

Next, we demonstrate the approximation performance of our Variational HMC algorithm relative to existing varia-
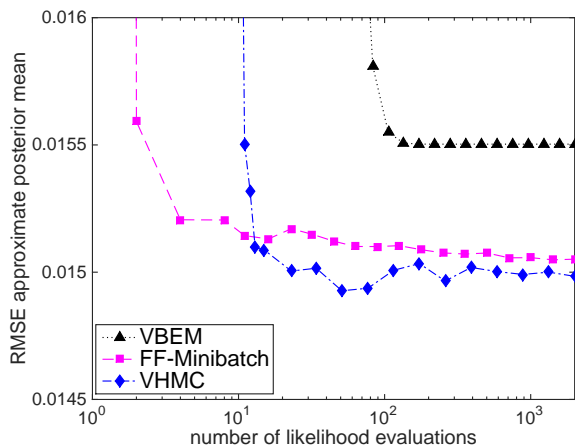
Figure 3. RMSE approximate posterior mean as a function of the number of likelihood evaluations for difference variational Bayesian approaches and our Variational HMC algorithm.

tional approaches on a simple Bayesian classification problem, binary probit regression, as a running example. Given N observed data pairs $\{(y_n, x_n)|y_n \in \{0,1\}, \ x_n \in \mathbb{R}^d\}_{n=1}^N$, the model comprised a probit likelihood function $P(y_n = 1|\theta) = \Phi(\theta^T x_n)$ and a Gaussian prior over the parameter $p(\theta) = \mathcal{N}(0,100)$, where Φ is the standard Gaussian cdf. A full covariance multivariate normal approximation is used for all variational approaches. The synthetic data we use are simulated from the model, with $N = 10000$ and $d = 5$. We show the performance averaged over 50 runs for all methods. We compare our algorithm to Variational Bayesian Expectation Maximization (VBEM) (Beal & Ghahramani, 2002; Ormerod & Wand, 2010), and the fixed-form variational approximation of (Salimans & Knowles, 2013). For all variational approaches, we initialize the posterior approximation to the prior. For our Variational HMC algorithm, we choose $s = 100$ random hidden units for the surrogate and set the starting point to be the origin. The number of hidden units is chosen in such a way that the surrogate is flexible enough to fit the target well and remain fast in computation. The HMC parameters are set to make the acceptance probability around 70%. The target density is almost Gaussian, and a fast transition $\mu_t = 1 - \exp(-t/5)$ is enough to stabilize our algorithm. The approximation performance is accessed in terms of the root mean squared error (RMSE) between the estimate (variational mean for VB and sample mean for VHMC) and the true parameter that is used to generate the dataset.

Figure 3 shows the performance of our Variational HMC algorithm, as well as the performance of the other two variational Bayes methods. As we can see from the graph,

VHMC and the subsampling based fixed-form variational approach (FF-minibatch) achieve lower RMSE than the VBEM algorithm. That is because of the extra factorization assumptions made by VBEM when introducing the auxiliary variables (Ormerod & Wand, 2010). Even though Gaussian approximation is already sufficiently accurate on this simple example, VHMC can still arrive at a lower RMSE due to the extra flexibility provided by the *free-form* neural network surrogate function.

4.3. Bayesian Logistic Regression

Now, we apply our Variational HMC method to a Bayesian logistic regression model. Given the i-th input vector x_i, the corresponding output (label) $y_i = \{0,1\}$ is assumed to follow the probability $p(y_i = 1|x_i, \beta) = 1/(1 + \exp(-x_i^T \beta))$ and a Gaussian prior $p(\beta) = \mathcal{N}(0,100)$ is used for the model parameter β. We test our proposed algorithm on the `a9a` dataset (Lin et al., 2008). The original dataset, which is compiled from the UCI `adult` dataset, has 32561 observations and 123 features. We use a 50 dimension random projection of the original features. We choose $s = 2000$ hidden units for the surrogate and set a transition schedule $\mu_t = 1 - \exp(-t/500)$ for our VHMC algorithm. We then compare the algorithm to HMC (Duane et al., 1987; Neal, 2011) and to SGLD (Welling & Teh, 2011). For HMC and VHMC, we set the *leap-frog* stepsize such that the acceptance rate is around 70%. For SGLD we choose batch size of 500 and use a range of fixed stepsizes.

Following (Ahn et al., 2012), we investigate the time normalized effective sample size (ESS)[1] averaged over the 51 parameters and compare this with the relative error after a fixed amount of computation time. The relative error of mean (REM) and relative error of covariance (REC) is defined as

$$\text{REM}_t = \frac{\sum_i |\overline{\beta_i^t} - \beta_i^o|}{\sum_i |\beta_i^o|}, \quad \text{REC}_t = \frac{\sum_i |C_{ij}^t - C_{ij}^o|}{\sum_{ij} |C_{ij}^o|}$$ (21)

where $\overline{\beta^t} = \frac{1}{t} \sum_{t'=1}^t \beta_{t'}$, $C^t = \frac{1}{t} \sum_{t'=1}^t (\beta_{t'} - \overline{\beta_t})(\beta_{t'} - \overline{\beta_t})^T$ are the sample mean and sample covariance up to time t and the ground truth β^o, C^o are obtained using a long run (T = 500K samples) of HMC algorithm.

Figure 4 shows the relative error at time T = 300, T = 3000 as a function of the time normalized mean ESS, which is a measure of the mixing rate. The results for the mean are shown on the top, and those for the covariance are on the bottom. We run each algorithm with a different setting of parameters that control the mixing rate: number of *leap-frog* steps $L = [50, 40, 30, 20, 10, 5, 1]$ for HMC and $L = [50, 40, 30, 20, 10, 5]$ for VHMC, and stepsizes

---

[1] Given B samples, $\text{ESS} = B[1 + 2\sum_{k=1}^K \gamma(k)]^{-1}$, where $\gamma(k)$ is the sample autocorrelation at lag k (Geyer, 1992)
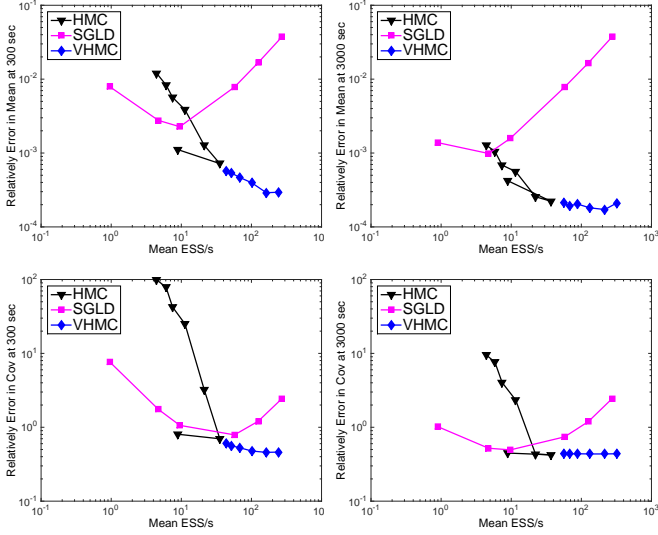
Figure 4. Final error of logistic regression at time T versus mixing rate for the mean (top) and covariance (bottom) estimates after 300 (left) and 3000 (right) seconds of computation. Each algorithm is run using different setting of parameters.

$\epsilon = [2e\text{-}3, 1e\text{-}3, 5e\text{-}4, 1e\text{-}4, 5e\text{-}5, 1e\text{-}5]$ for SGLD.

As we decrease the stepsize, SGLD becomes less biased in the gradient approximation, resulting in smaller relative error. However, the exploration efficiency drops at the same time and sampling variance gradually dominates the relative error. In contrast, HMC uses a fixed *leap-frog* stepsize and therefore maintains high exploration efficiency in parameter space. The down side is the expensive computation of the full gradient and the possible turning back of the trajectories when the number of *leap-frog* steps is unnecessarily large. Adopting a flexible neural network surrogate, VHMC balances the computation cost and approximation quality much better than subsampling and achieves lower relative error with high mixing rates.

4.4. Independent Component Analysis

Finally, we apply our method to sample from the posterior distribution of the unmixing matrix in Independent Component Analysis (ICA) (Hyvärinen & Oja, 2000). Given N d-dimensional observations $X = \{x_n \in \mathbb{R}^d\}_{n=1}^N$, we model the data as $p(x|W) = |\det(W)| \prod_{i=1}^d p_i(w_i^T x)$, where w_i is the i-th row of W and p_i is supposed to capture the true density of the i-th independent component. Following (Welling & Teh, 2011), we use a Gaussian prior over the unmixing matrix $p(w_{ij}) = \mathcal{N}(0, \sigma)$ and choose $p_i(y_i) = [4\cosh(\frac{1}{2}y_i)]^{-1}$ with $y_i = w_j^T x$. We evaluate our method using the MEG dataset (Vigário et al., 1997), which has 122 channels and 17730 observations. We ex-

tract the first 5 channels for our experiment which leads to samples with 25 dimensions. We then compare our algorithm to standard HMC and stochastic gradient Langevin dynamics (SGLD) (Welling & Teh, 2011). For SGLD, we use a natural gradient (Amari et al., 1996) which has been found to improve the efficiency of gradient descent significantly. We set $\sigma = 100$ for the Gaussian prior. For HMC and Variational HMC, we set the *leap-frog* stepsize to keep the acceptance ratio around 70% and set $L = 40$ to allow an efficient exploration in parameter space. For SGLD, we choose batch size of 500 and use stepsizes from a polynomial annealing schedule $a(b + t)^{-\delta}$, with $a = 5 \times 10^{-3}, b = 10^{-4}$ and $\delta = 0.5$. (This setting reduces the stepsize from 5×10^{-5} to 1×10^{-6} during 1e+7 iterations). We choose $s = 1000$ hidden units and set the transition schedule $\mu_t = 1 - \exp(-t/2000)$ for our Variational HMC algorithm. To measure the convergence of the samplers, we use the Amari distance (Amari et al., 1996) $d_A(\overline{W}, W_0)$, where $\overline{W}$ is the sample average and W_0 is the true unmixing matrix estimated using a long run (T = 100K samples) of standard HMC algorithm.
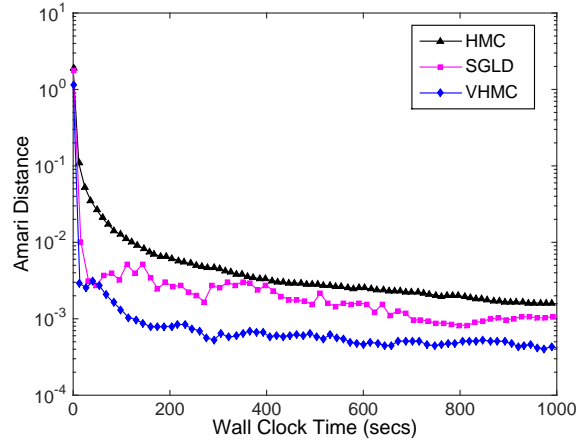

Figure 5. Convergence of Amari distance on the MEG data for HMC, SGLD and our Variational HMC algorithm.

The Amari distance as a function of runtime is reported for each of these methods in Figure 5. From the graph, we can see that SGLD converges faster than standard HMC. The bias introduced by subsampling is compensated by the fast exploration in parameter space which reduce the sample variance. However, the exploration efficiency of SGLD decreases as the stepsize is annealed. By maintaining efficient exploration in parameter space (same stepsize as HMC) while reducing the computation in simulating the Hamiltonian flow, VHMC outperforms SGLD, arriving at a lower Amari distance much more rapidly.

5. Conclusion

We have presented a novel approach, Variational Hamiltonian Monte Carlo, for approximate Bayesian inference. Our approach builds on the framework of HMC, but using flexible and efficient neural network surrogate functions to approximate the expensive full gradient. The surrogate keeps refining its approximation by collecting training data while the sampler exploring the parameter space. This way, our algorithm can be viewed as a *free-form* variational approach. Unlike subsampling-based MCMC methods, VHMC maintains the relatively high exploration efficiency of its MCMC counterpart while reducing the computation cost. Compared to *fixed-form* variational approximation, VHMC is more flexible and thus can approximate the target distribution better.

References

Ahn, S., Korattikara, A., and Welling, M. Bayesian posterior sampling via stochastic gradient fisher scoring. In *International Conference on Machine Learning*, 2012.

Albert, J. *Bayesian Computing with R*. Springer Science, New York, 2009.

Amari, S. I., Cichocki, A., and Yang, H. A new learning algorithm for blind signal separation. In *Advances in Neual Information Processing Systems*, pp. 757–763, 1996.

Beal, M. J. and Ghahramani, Z. The variational Bayesian EM algorithm for incomplete data: with application to scoring graphical model structures. In *Bayesian Statistics 7: Proceeding of the 7th Valencia International Meeting*, pp. 453–463, 2002.

Betancourt, M. The fundamental incompatibility of scalable Hamiltonian Monte Carlo and naive data subsampling. In *Proceedings of the 32nd International Conference on Machine Learning (ICML 2015)*, 2015.

Chen, T., Fox, E. B., and Guestrin, C. Stochastic gradient hamiltonian monte carlo. In *Proceedings of 31st International Conference on Machine Learning (ICML 2014)*, 2014.

de Freitas, N., Højen-Sørensen, P., Jordan, M. I., and Russell, S. Variational mcmc. In *Proceedings of the 7th conference on Uncertainty in Artificial Intelligence (UAI 2001)*, pp. 120–127, San Francisco, 2001. Morgan Kaufmann.

Ding, N., Fang, Y., Babbush, R., Chen, C., Skell, R. D., and Neven, H. Bayesian sampling using stochastic gradient thermostats. In *Advances in Neural Information Processing Systems 27 (NIPS 2014)*, 2014.

Duane, S., Kennedy, A. D., Pendleton, B J., and Roweth, D. Hybrid Monte Carlo. *Physics Letters B*, 195(2):216 – 222, 1987.

Ferrari, S. and Stengel, R. F. Smooth function approximation using neural networks. *IEEE Trans. Neural Network*, 16(1):24–38, 2005.

Geyer, C. J. Practical Markov Chain Monte Carlo. *Statistical Science*, 7:473–483, 1992.

Girolami, M. and Calderhead, B. Riemann manifold Langevin and Hamiltonian Monte Carlo methods. *Journal of the Royal Statistical Society*, (with discussion) 73 (2):123–214, March 2011.

Hoffman, M. and Gelman, A. The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo. arxiv.org/abs/1111.4246, 2011.

Honkela, A., Raiko, T., Kuusela, M., Tornio, M., and Karhunen, J. Approximate Riemannian conjugate gradient learning for fixed-form variational Bayes. *Journal of Machine Learning Research*, 11:3235–3268, 2010.

Huang, G. B., Chen, L., and Siew, C. K. Universal approximation using incremental constructive feedforward networks with random hidden nodes. *IEEE Trans. Neural Networks*, 17(4):879–892, 2006a.

Huang, G. B., Zhu, Q. Y., and Siew, C. K. Extreme learning machine: Theory and applications. *Neurocomputing*, 70 (1-3):489–501, 2006b.

Hyvärinen, A. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6:695–709, 2005.

Hyvärinen, A. and Oja, E. Independent component analysis: algorithms and applications. *Neural networks*, 13: 411–430, 2000.

Jordan, M. I., Ghahramani, Z., Jaakkola, T. S., and Saul, L. K. An introduction to variational methods for graphical methods. In *Machine Learning*, pp. 183–233. MIT Press, 1999.

Lin, C. J., Weng, R. C., and Keerthi, S. S. Trust region Newton method for large-scale logistic regression. *Journal of Machine Learning Research*, 9:627–650, 2008.

Liu, J. S. *Monte Carlo Strategies in Scientific Computing*. Springer, 2001.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953.

Neal, R. M. MCMC using Hamiltonian dynamics. In Brooks, S., Gelman, A., Jones, G., and Meng, X. L. (eds.), *Handbook of Markov Chain Monte Carlo*, pp. 113–162. Chapman and Hall/CRC, 2011.

Ormerod, J. T. and Wand, M. P. Explaining Variational Approximations. *The American Statistician*, 2(64):140–153, 2010.

Rasmussen, C. E. Gaussian processes to speed up hybrid monte carlo for expensive bayesian integrals. *Bayesian Statistics*, 7:651–659, 2003.

Salimans, T. and Knowles, D. A. Fixed-form variational posterior approximation through stochastic linear regression. *Bayesian Analysis*, 8(4):837–882, 2013.

Salimans, T., Kingma, D. P., and Welling, M. Markov chain monte carlo and variational inference: bridging the gap. In *International Conference on Machine Learning (ICML 2015)*, pp. 1218–1226, 2015.

Saul, L. and Jordan, M. I. Exploiting tractable substructures in intractable networks. In Tesauro, G., Touretzky, D. S., and Leen, T. K. (eds.), *Advance in neural information processing systems 7 (NIPS 1996)*, pp. 486–492, Cambridge, MA, 1996. MIT Press.

Vigário, R., S'arel'a, J., and Oja, E. MEG data for studies using independent component analysis. http://research.ics.aalto.fi/ica/eegmeg/MEG_data.html. 1997.

Wainwright, M. and Jordan, M. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305, 2008.

Wang, Z., Mohamed, S., and Nando, D. Adaptive hamiltonian and riemann manifold monte carlo. In *Proceedings of the 30th International Conference on Machine Learning (ICML 2013)*, pp. 1462–1470, 2013.

Welling, M. and Teh, Y. W. Bayesian learning via stochastic gradient Langevin dynamics. In *Proceedings of the International Conference on Machine Learning*, 2011.

Supplementary Material

A. Score Matching

We aim to approximate the target posterior $p(\theta|Y)$ by a *free-form* unnormalized probability

$$q_{\tilde{v}}(\theta) \propto \exp[-z(\theta)]$$

From (Hyvärinen, 2005), we optimize the expected squared distance between score functions [2]

$$J(\tilde{v}) = \frac{1}{2} \int_{\theta \in \mathbb{R}^d} q_{\tilde{v}}(\theta) \|\varphi_{\tilde{v}}(\theta) - \varphi_Y(\theta)\|^2 \, d\theta$$

where

$$\varphi_{\tilde{v}}(\theta) = -\nabla_\theta \log q_{\tilde{v}}(\theta) = \nabla_\theta z(\theta)$$
$$\varphi_Y(\theta) = -\nabla_\theta \log p(\theta|Y) = \nabla_\theta U(\theta)$$

It is easy to verify that $J(\tilde{v}) = 0 \Rightarrow \varphi_{\tilde{v}} = \varphi_Y + C \Rightarrow q_{\tilde{v}}(\theta) = p(\theta|Y)$, so $K(\theta)$ is a well defined squared distance and we refer to it as score matching distance.

Given the training data collected from the Markov chain history

$$\mathcal{T}_s^{(t)} := \{(\theta_n, \nabla_\theta U(\theta_n))\}_{n=1}^t \in \mathbb{R}^d \times \mathbb{R}^d$$

we can optimize the empirical version

$$\hat{J}(\tilde{v}) = \frac{1}{2t} \sum_{n=1}^t \|\nabla_\theta z(\theta_n) - \nabla_\theta U(\theta_n)\|^2$$

Proof of Proposition 1

$$v^{(t)} = \arg\min_v \sum_{n=1}^t \|\nabla_\theta z(\theta_n) - \nabla_\theta U(\theta_n)\|^2 + \lambda \|v\|^2 \tag{22}$$

As assumed the neural network surrogate takes the form

$$z(\theta) = \sum_{i=1}^s v_i \sigma(w_i \cdot \theta + d_i) + b$$

its derivative is

$$\nabla_\theta z(\theta) = \sum_{i=1}^s v_i \sigma'(w_i \cdot \theta + d_i) w_i = A(\theta) v$$

where $A(\theta) = (A_1(\theta), A_2(\theta), \ldots, A_s(\theta))$ and

$$A_i(\theta) = \sigma'(w_i \cdot \theta + d_i) w_i, \quad i = 1, \ldots, s$$

---

[2]Note that the samples for θ now are collected from the free-form approximation, so we change the integral weights accordingly.

Denote

$$A^{(t)} = \begin{bmatrix} A(\theta_1) \\ A(\theta_2) \\ \vdots \\ A(\theta_t) \end{bmatrix}, \quad B^{(t)} = \begin{bmatrix} \nabla_\theta U(\theta_1) \\ \nabla_\theta U(\theta_2) \\ \vdots \\ \nabla_\theta U(\theta_t) \end{bmatrix}$$

(22) can be simplified and solved as below

$$v^{(t)} = \arg\min_v \|A^{(t)}v - B^{(t)}\|^2 + \lambda\|v\|^2$$

$$= \arg\min_v \ v'\left((A^{(t)})'A^{(t)} + \lambda I\right)v - 2\left(B^{(t)}\right)'A^{(t)}v$$

$$= \left((A^{(t)})'A^{(t)} + \lambda I\right)^{-1}\left(A^{(t)}\right)'B^{(t)}$$

Similarly, given a new data point $(\theta_{t+1}, \nabla_\theta U(\theta_{t+1}))$, the new estimator is

$$v^{(t+1)} = \left((A^{(t+1)})'A^{(t+1)} + \lambda I\right)^{-1}\left(A^{(t+1)}\right)'B^{(t+1)}$$

where

$$A^{(t+1)} = \begin{bmatrix} A^{(t)} \\ A_{t+1} \end{bmatrix}, \quad B^{(t+1)} = \begin{bmatrix} B^{(t)} \\ B_{t+1} \end{bmatrix}$$

$A_{t+1} = A(\theta_{t+1})$, $B_{t+1} = \nabla_\theta U(\theta_{t+1})$. Therefore

$$v^{(t+1)} = \left((A^{(t+1)})'A^{(t+1)} + \lambda I\right)^{-1}\left(A^{(t+1)}\right)'B^{(t+1)}$$

$$= \left((A^{(t)})'A^{(t)} + A'_{t+1}A_{t+1} + \lambda I\right)^{-1}$$

$$\left[\left(A^{(t)}\right)'B^{(t)} + A'_{t+1}B_{t+1}\right] \tag{23}$$

Denote $C^{(t)} = \left[(A^{(t)})'A^{(t)} + \lambda I\right]^{-1}$, by Sherman-Morrison-Woodbury formula,

$$C^{(t+1)} = \left[(A^{(t)})'A^{(t)} + A'_{t+1}A_{t+1} + \lambda I\right]^{-1}$$

$$= C^{(t)} - C^{(t)}A'_{t+1}\left[I + A_{t+1}C^{(t)}A'_{t+1}\right]^{-1}A_{t+1}C^{(t)}$$

substitute into (23)

$$v^{(t+1)} = C^{(t)}\left[\left(A^{(t)}\right)'B^{(t)} + A'_{t+1}B_{t+1}\right] -$$

$$C^{(t)}A'_{t+1}\left[I + A_{t+1}C^{(t)}A'_{t+1}\right]^{-1}A_{t+1}$$

$$C^{(t)}\left[\left(A^{(t)}\right)'B^{(t)} + A'_{t+1}B_{t+1}\right]$$

$$= v^{(t)} + C^{(t)}A'_{t+1}\left(B_{t+1} - \right.$$

$$\left.\left[I + A_{t+1}C^{(t)}A'_{t+1}\right]^{-1}A_{t+1}C^{(t)}A'_{t+1}B_{t+1}\right)$$

$$- C^{(t)}A'_{t+1}\left[I + A_{t+1}C^{(t)}A'_{t+1}\right]^{-1}A_{t+1}v^{(t)}$$

$$= v^{(t)} + W^{(t+1)}(B_{t+1} - A_{t+1}v^{(t)})$$

where

$$W^{(t+1)} = C^{(t)}A'_{t+1}\left[I + A_{t+1}C^{(t)}A'_{t+1}\right]^{-1}$$

and the updating formula for $C^{(t+1)}$ can be simplified as

$$C^{(t+1)} = C^{(t)} - W^{(t+1)}A_{t+1}C^{(t)}$$