Fast Inference for Intractable Likelihood Problems using Variational Bayes

David Gunawan, Minh-Ngoc Tran and Robert Kohn*
May 19, 2017

Abstract

Variational Bayes (VB) is a popular estimation method for Bayesian inference. However, most existing VB algorithms are restricted to cases where the likelihood is tractable, which precludes their use in many important situations. Tran et al. (2017) extend the scope of application of VB to cases where the likelihood is intractable but can be estimated unbiasedly, and name the method Variational Bayes with Intractable Likelihood (VBIL). This paper presents a version of VBIL, named Variational Bayes with Intractable Log-Likelihood (VBILL), that is useful for cases as Big Data and Big Panel Data models, where unbiased estimators of the gradient of the log-likelihood are available. We demonstrate that such estimators can be easily obtained in many Big Data applications. The proposed method is exact in the sense that, apart from an extra Monte Carlo error which can be controlled, it is able to produce estimators as if the true likelihood, or full-data likelihood, is used. In particular, we develop a computationally efficient approach, based on data subsampling and the MapReduce programming technique, for analyzing massive datasets which cannot fit into the memory of a single desktop PC. We illustrate the method

^{*}Gunawan and Kohn are with School of Economics, University of New South Wales Business School. Tran is with Discipline of Business Analytics, University of Sydney Business School. Email: minh-ngoc.tran@sydney.edu.au

using several simulated datasets and a big real dataset based on the arrival time status of U. S. airlines.

Keywords. Pseudo Marginal Metropolis-Hastings, Big Data, Panel Data, Difference Estimator.

1 Introduction

Given an observed dataset y and a statistical model with a vector of unknown parameters θ , a major aim of statistics is to carry out inference about θ , i.e., estimate the underlying θ that generated y and assess the associated uncertainty. The likelihood function $p(y|\theta)$, which is the density of the data y conditional on the postulated model and the parameter vector θ , is the basis of Bayesian methods, such as Markov chain Monte Carlo (MCMC) and Variational Bayes (VB). These methods require exact evaluation of the likelihood $p(y|\theta)$ at each value of θ . In many modern statistical applications, however, the likelihood function, and thus the log-likelihood function, is either analytically intractable or computationally intractable, making it difficult to use likelihood-based methods.

An important situation in which the log-likelihood is computationally intractable is Big Data (Bardenet et al., 2015; Quiroz et al., 2015), where the log-likelihood function, under the independence assumption and without random effects, is a sum of a very large number of terms and thus too expensive to compute. Large panel data models (Fitzmaurice et al., 2011) are another example where the log-likelihood is both analytically and computationally intractable as it is a sum of many terms, each being the log of an integral over the random effects and cannot be computed analytically.

There are several methods in the literature that work with an intractable likelihood. A remarkable approach is the pseudo-marginal Metropolis-Hastings (PMMH) algorithm (Andrieu and Roberts, 2009), which replaces the intractable likelihood in the Metropolis-Hastings ratio by its non-negative unbiased estimator. An attractive

property of the PMMH approach is that it is exact in the sense that it is still able to generate samples from the posterior, as if the true likelihood was used. Similarly to standard Metropolis-Hastings algorithms, PMMH is extremely flexible. However, this method is highly sensitive to the variance of the likelihood estimator. The chain might get stuck and mix poorly if the likelihood estimates are highly variable (Flury and Shephard, 2011). This is because the asymptotic variance of PMMH estimators increases exponentially with the variance of the log of the estimator of the likelihood (Pitt et al., 2012), which in turn increases linearly with the sample size. Therefore the PMMH method can be computationally expensive making it unsuitable for Big Data applications.

VB is a computationally efficient alternative to MCMC (Attias, 1999; Bishop, 2006). However, most existing VB algorithms are restricted to cases where the likelihood is tractable, which precludes the use of VB in many interesting models. Tran et al. (2017) extend the scope of application of VB to cases where the likelihood is intractable but can be estimated unbiasedly, and name the method Variational Bayes with Intractable Likelihood (VBIL). Their method works with non-negative unbiased estimators of the *likelihood*, and is useful when it is convenient to obtain unbiased estimates of the likelihood such as in state space models. This paper presents a version of VBIL, called the Variational Bayes with Intractable Log-Likelihood (VBILL), that requires unbiased estimates of the gradient of the log-likelihood. It turns out that, in cases of Big Data both with and without random effects, it is easy and computationally efficient to obtain unbiased estimates of the gradient of the log-likelihood using data subsampling. Consider the case of Big Data without random effects, where the log-likelihood function is a sum of many terms, with each computed analytically. Then the gradient of the log-likelihood is a sum of tractable terms, and this sum can be estimated unbiasedly using data subsampling. In the case of Big Panel Data, each log-likelihood contribution is the log of an intractable integral and thus can no longer be computed analytically. However, we are still able to obtain an unbiased estimate of the gradient of the log-likelihood using subsampling and Fisher's identity (see Section 2.2). Although data subsampling has now been used in Big Data situations with a very large number of independent observations, we are not aware of any efficient estimation methods for Big Panel Data models. One of the main goals of our paper is to fill this gap.

This paper makes two important improvements to VBIL that greatly enhance its performance. The first is that we take into account the information of the gradient of the log-likelihood, which helps the stochastic optimization procedure more stable and converge faster. The second is that we now aim to minimize the same Kullback-Leibler divergence as that targeted when the likelihood is tractable. That is, VBILL is exact in the sense that, apart from an extra Monte Carlo error which can be controlled, it is able to produce estimators as if the true likelihood or full-data likelihood were used. The VBIL approach of Tran et al. (2017) is exact in this sense only under the condition that the variance of the likelihood estimator is constant.

Our paper also uses the MapReduce programming technique and develops a computationally efficient approach for analyzing massive datasets which do not fit into the memory of a single desktop PC. The implementation of MapReduce uses the divide and combine idea where the data is divided into small chunks, each chunk is processed separately and the chunk-based results are then combined to construct the final estimates. Under some regularity conditions, Battey et al. (2015) show that the information loss due to the divide and combine procedure is asymptotically negligible when the full sample size grows, as long as the number of chunks is not too large. In finite-sample settings, however, the resulting estimators are sensitive to how the data are divided. It is important to note that our final estimator is mathematically justified and independent of the data chunking, as we use the divide and combine procedure mainly to obtain an unbiased estimator of the gradient of log-likelihood for the VBILL algorithm.

The paper is organized as follows. Section 2 describes the VBILL approach and

its applications to Big Data problems using data subsampling. Section 3 presents empirical studies and Section 4 concludes. The appendix presents technical details.

2 Variational Bayes with Intractable Log-Likelihood

Let $p(\theta)$ be the prior, $L(\theta) := p(y|\theta)$ the likelihood and $\pi(\theta) \propto p(\theta)L(\theta)$ the posterior distribution of θ . Let $\ell(\theta) := \log L(\theta)$. Our paper, with a small abuse of notation, uses the same notation for the probability distribution and its density function. VB approximates the posterior distribution of θ by a probability distribution $q_{\lambda}(\theta)$ within some parametric class of distributions such as an exponential family, with parameter λ chosen to minimize the Kullback-Leibler divergence between $q_{\lambda}(\theta)$ and $\pi(\theta)$,

$$\mathrm{KL}(\lambda) = \mathrm{KL}(q_{\lambda} || \pi) := \int q_{\lambda}(\theta) \log \frac{q_{\lambda}(\theta)}{\pi(\theta)} d\theta.$$

Minimizing this divergence is equivalent to maximizing the lower bound

$$LB(\lambda) = \int q_{\lambda}(\theta) \log \frac{p(\theta)L(\theta)}{q_{\lambda}(\theta)} d\theta = A(\lambda) + \int q_{\lambda}(\theta)\ell(\theta)d\theta,$$

with $A(\lambda) = \int q_{\lambda}(\theta) \log \frac{p(\theta)}{q_{\lambda}(\theta)} d\theta$. Often $A(\lambda)$ can be computed analytically. Suppose that $\theta \sim q_{\lambda}(\theta)$ can be represented as a deterministic function of a random vector ϵ whose distribution is independent of λ . More precisely, let $g(\cdot, \cdot)$ be a function such that $\theta = g(\lambda, \epsilon) \sim q_{\lambda}(\theta)$, where $\epsilon \sim p_{\epsilon}(\epsilon)$ with $p_{\epsilon}(\cdot)$ not dependent on λ . For example, if $\theta \sim N(\mu, \Sigma)$, then θ can be written as $\theta = \mu + \Sigma^{1/2} \epsilon$ with $\epsilon \sim N(0, I)$. Hence,

$$LB(\lambda) = A(\lambda) + \mathbb{E}_{\epsilon \sim p_{\epsilon}}[\ell(g(\lambda, \epsilon))].$$

The gradient of the lower bound is

$$\nabla_{\lambda} LB(\lambda) = \nabla_{\lambda} A(\lambda) + \mathbb{E}_{\epsilon \sim p_{\epsilon}} \Big[\nabla_{\lambda} g(\lambda, \epsilon) \nabla_{\theta} \ell(g(\lambda, \epsilon)) \Big].$$

Note that if $A(\lambda)$ cannot be computed analytically or it is inconvenient to do so, we can always represent the entire $\nabla_{\lambda} LB(\lambda)$ as an expectation with respect to ϵ . By generating ϵ from $p_{\epsilon}(\cdot)$, we are able to obtain an unbiased estimator $\widehat{\nabla_{\lambda} LB}(\lambda)$ of the gradient $\nabla_{\lambda} LB(\lambda)$. Therefore, we can use stochastic optimization to optimize $LB(\lambda)$. The representation of the gradient $\nabla_{\lambda} LB(\lambda)$ in terms of an expectation with respect to ϵ rather than θ is the so-called reparameterization trick (Kingma and Welling, 2013). In general it works more efficient than the alternative methods that sample directly from $q_{\lambda}(\theta)$ (Ruiz et al., 2016; Tan and Nott, 2017). One of the reasons is that this reparameterization takes into account the information from the gradient of the log-likelihood.

We now extend the method to the case where $\nabla_{\theta}\ell(\theta)$ is intractable but can be estimated unbiasedly. Let $\widehat{G}(\theta)$ be an unbiased estimator of $\nabla_{\theta}\ell(\theta)$, i.e. $\mathbb{E}(\widehat{G}(\theta)) = \nabla_{\theta}\ell(\theta)$, where the expectation is with respect to all random variables u needed for computing $\widehat{G}(\theta) = \widehat{G}(\theta, u)$. Typically, u is a set of uniform random numbers. Write $p_U(\cdot)$ for the distribution of u. Then,

$$\nabla_{\lambda} LB(\lambda) = \nabla_{\lambda} A(\lambda) + \mathbb{E}_{\epsilon \sim p_{\epsilon}, u \sim p_{U}} \Big[\nabla_{\lambda} g(\lambda, \epsilon) \widehat{G}(\theta, u) \Big], \quad \theta = g(\lambda, \epsilon),$$

which can be estimated unbiasedly by

$$\widehat{\nabla_{\lambda} \text{LB}(\lambda)} = \nabla_{\lambda} A(\lambda) + \frac{1}{S} \sum_{i=1}^{S} \nabla_{\lambda} g(\lambda, \epsilon_i) \widehat{G}(\theta_i, u_i)$$
 (1)

where $\epsilon_i \sim p_{\epsilon}(\cdot)$, $u_i \sim p_U(\cdot)$, $\theta_i = g(\lambda, \epsilon_i)$, i = 1,...,S. Therefore, we can use stochastic optimization (Robbins and Monro, 1951) to maximize LB(λ) as follows.

Algorithm 1. • Set the number of samples S, initialize $\lambda^{(0)}$ and stop the following iteration if the stopping criterion is met.

• For
$$t = 0, 1, ..., compute \lambda^{(t+1)} = \lambda^{(t)} + a_t I_F(\lambda^{(t)})^{-1} \widehat{\nabla_{\lambda} LB}(\lambda^{(t)}).$$

Here $I_F(\lambda) = \text{cov}_{q_{\lambda}}(\nabla_{\lambda} \log q_{\lambda}(\theta))$ is the Fisher information matrix of λ w.r.t. $q_{\lambda}(\theta)$.

The sequence $\{a_t, t \ge 0\}$ is the learning rate and should satisfy $a_t > 0$, $\sum_t a_t = \infty$ and $\sum_t a_t^2 < \infty$ (Robbins and Monro, 1951).

In Algorithm 1 we use the natural gradient of the lower bound, which is defined as $I_F(\lambda)^{-1}\nabla_{\lambda}\mathrm{LB}(\lambda)$. The natural gradient more adequately captures the geometry of the variational distribution q_{λ} ; see, e.g., Amari (1998). Using this gradient often makes the convergence faster (Tran et al., 2017; Hoffman et al., 2013). Here, we provide an informal explanation in the current context. It is easy to see that the Hessian matrix of the lower bound is

$$H(\lambda) = \nabla_{\lambda\lambda'}[\mathrm{LB}(\lambda)] = \int \nabla_{\lambda\lambda'}[\log q_{\lambda}(\theta)] q_{\lambda}(\theta) \log \frac{\pi(\theta)}{q_{\lambda}(\theta)} d\theta - I_F(\lambda),$$

which is approximately $-I_F(\lambda)$ when $q_{\lambda} \approx \pi$. That is, Algorithm 1 is a Newton-Raphson type algorithm as it uses the second-order information of the target function.

It is important to note that VBILL is exact in the sense that it minimizes the same Kullback-Leibler divergence $KL(q_{\lambda}||\pi)$ as the target that we would optimize if the exact likelihood is available. Therefore, apart from an extra Monte Carlo error, the VBILL approach is able to produce estimators as if the true likelihood, or full-data likelihood, was used. The VBIL approach of Tran et al. (2017) works on an augmented space and minimizes a Kullback-Leibler divergence that is equal to $KL(q_{\lambda}||\pi)$ only if the variance of the log of the estimated likelihood is constant.

Convergence properties of the stochastic optimization procedure in Algorithm 1 are well-known in the literature (see, e.g., Sacks, 1958). Similarly to Tran et al. (2017), it is possible to show that the variance of VBILL estimators increases only linearly with the variance of the estimated gradient of the log-likelihood, which suggests that VBILL is robust to variation in estimating this gradient.

Stopping rule

As in Tran et al. (2017), the updating algorithm is stopped if the change in the average value of the lower bounds over a window of K iterations,

$$\overline{LB}(\lambda^{(t)}) := \frac{1}{K} \sum_{k=1}^{K} \widehat{LB}(\lambda^{(t-k+1)}),$$

is less than some threshold ε , where $\widehat{LB}(\lambda)$ is an estimate of $LB(\lambda)$. Furthermore, we also use the scaled version of the lower bound $\widehat{LB}(\lambda)/n$, with n the size of the dataset. The scaled lower bound is roughly independent of the size of the dataset. Our paper sets K=5. Alternatively, we can stop the updating algorithm if the change in the average value $\overline{\lambda}^{(t)} = (1/K) \sum_{k=1}^K \lambda^{(t-k+1)}$, is less than some threshold ε . A byproduct of using the stopping rule based on the lower bound is that the lower bound estimates can be useful for model selection (Sato, 2001; Nott et al., 2012).

2.1 VBILL with Data Subsampling

This section presents the VBILL method for Big Data. Let $y = \{y_i, i=1,...,n\}$ be the data set. We assume that the y_i are independent so that the likelihood is $L(\theta) = \prod_{i=1}^{n} p(y_i|\theta)$ and we assume for now that each likelihood contribution $p(y_i|\theta)$ can be computed analytically. The log-likelihood is

$$\ell(\theta) := \sum_{i=1}^{n} \ell_i(\theta), \quad \text{where} \quad \ell_i(\theta) := \log p(y_i | \theta). \tag{2}$$

The gradient of the log-likelihood is

$$\nabla_{\theta} \ell(\theta) = \sum_{i} \nabla_{\theta} \ell_{i}(\theta),$$

where each $\nabla_{\theta} \ell_i(\theta)$ can be computed analytically or by using numerical differentiation. We are concerned with the case where this gradient is computationally intractable in the sense that n is so large that computing this sum is impractical.

The proposed VBILL approach to this problem is based on the key observation that it is convenient and computationally much cheaper to obtain an unbiased estimator $\widehat{G}(\theta)$ of the gradient of the log-likelihood $\nabla_{\theta} \ell(\theta)$.

Let $g_i(\theta) := \nabla_{\theta} \ell_i(\theta)$. Let $\overline{\theta}$ be some central value of θ obtained by using, for example, Maximum Likelihood or MCMC, based on a representative subset of the full data. By the first-order Taylor series expansion of the vector field $g_i(\theta)$ (Apostol, 1969, Chapter 8),

$$g_{i}(\theta) = g_{i}(\overline{\theta}) + \nabla_{\theta'} g_{i}(\overline{\theta})(\theta - \overline{\theta}) + o(\|\theta - \overline{\theta}\|)$$

$$= \nabla_{\theta} \ell_{i}(\overline{\theta}) + \nabla_{\theta\theta'}^{2} \ell_{i}(\overline{\theta})(\theta - \overline{\theta}) + o(\|\theta - \overline{\theta}\|) = w_{i}(\theta) + o(\|\theta - \overline{\theta}\|), \quad (3)$$

where

$$w_i(\theta) := \nabla_{\theta} \ell_i(\overline{\theta}) + \nabla^2_{\theta\theta'} \ell_i(\overline{\theta})(\theta - \overline{\theta})$$

and $o(\delta)$ denotes the small order of δ , meaning $o(\delta)/\delta \to 0$ as $\delta \to 0$. We can write

$$\nabla_{\theta} \ell(\theta) = \sum_{i} w_{i}(\theta) + \sum_{i} d_{i}(\theta) = w(\theta) + d(\theta),$$

with $d_i(\theta) = g_i(\theta) - w_i(\theta)$, $w(\theta) = \sum_i w_i(\theta)$ and $d(\theta) = \sum_i d_i(\theta)$. It is computationally cheap to compute the first term

$$w(\theta) = \sum_{i} \nabla_{\theta} \ell_{i}(\overline{\theta}) + \left(\sum_{i} \nabla_{\theta\theta'}^{2} \ell_{i}(\overline{\theta})\right) (\theta - \overline{\theta}) = A(\overline{\theta}) + B(\overline{\theta})(\theta - \overline{\theta})$$

because the sums $A(\overline{\theta})$ and $B(\overline{\theta})$ are computed just once. The second term $d(\theta)$ can be estimated unbiasedly by simple random sampling

$$\widehat{d}_m(\theta) = \frac{1}{m} \sum_{i=1}^m n d_{u_i}(\theta),$$

where $u = (u_1, ..., u_m)$, $u_i \in F = \{1, 2, ..., n\}$, is the $m \times 1$ vector of indices obtained by simple random sampling with replacement from the full index set F, $P(u_i = k) = 1/n$

for all $k \in F$. Here m < n is the size of subsamples. It is easy to show that $\mathbb{E}(\widehat{d}_m(\theta)) = d(\theta)$. Therefore,

$$\widehat{G}(\theta, u) := w(\theta) + \widehat{d}_m(\theta) \tag{4}$$

is an unbiased estimator of log-likelihood gradient $\nabla_{\theta} \ell(\theta)$.

Since $w_i(\theta)$ is an approximation of $g_i(\theta)$, the differences $d_i(\theta)$ should have roughly the same size across i. Thus, $\widehat{d}_m(\theta)$ is expected to be an efficient estimator of $d(\theta)$ (Quiroz et al., 2015). If $\overline{\theta}$ is an MLE of θ , then by the Bernstein von Mises theorem (see, e.g. Chen, 1985), $\|\theta - \overline{\theta}\| = O_P(n^{-1/2})$, with O_P the stochastically big order with respect to the posterior distribution of θ . Then, from (3), the $d_i(\theta)$ are very small, and thus $\widehat{G}(\theta,u)$ has a small variance. See also Bardenet et al. (2015) and Quiroz et al. (2015) who demonstrate the efficiency of data subsampling estimators. This guarantees that the variance of the gradient (1) is small, which makes the VBILL procedure highly efficient.

2.2 VBILL with Data Subsampling for Big Panel Data

This section describes the VBILL method for estimating Big Panel Data models. For panel data models with n panels $\{y_1,...,y_n\}$, also called longitudinal models or generalized linear mixed models, the log-likelihood is still in the form of (2), but each likelihood contribution $p(y_i|\theta)$ is an integral over random effects α_i ,

$$p(y_i|\theta) = \int p(y_i|\theta,\alpha_i)p(\alpha_i|\theta)d\alpha_i.$$
 (5)

In many cases the integral (5) is analytically intractable and hence also its first and second derivatives. Therefore, it is intractable to compute the gradients $\nabla_{\theta_i} \ell(\theta)$ and $\nabla_{\theta} \ell(\theta)$, even when n is small. However, it is still possible to estimate these gradients unbiasedly.

We first describe how to obtain an unbiased estimator of the gradient of the

log-likelihood contributions $\nabla_{\theta} l_i(\theta)$. The gradient $\nabla_{\theta} l_i(\theta)$ can be written as

$$\nabla_{\theta} l_{i}(\theta) = \frac{1}{p(y_{i}|\theta)} \nabla_{\theta} \int p(y_{i}, \alpha_{i}|\theta) d\alpha_{i}$$

$$= \frac{1}{p(y_{i}|\theta)} \int \nabla_{\theta} \log p(y_{i}, \alpha_{i}|\theta) \times p(y_{i}, \alpha_{i}|\theta) d\alpha_{i}$$

$$= \int \nabla_{\theta} \log p(y_{i}, \alpha_{i}|\theta) \times p(\alpha_{i}|y_{i}, \theta) d\alpha_{i}.$$
(6)

The representation in (6) is known in the literature as Fisher's identity (Cappe et al., 2005). Therefore, we can obtain an unbiased estimator of $\nabla_{\theta} l_i(\theta)$ by using importance sampling. When n is large, it is clear that we can use the data subsampling approach described in the previous section, together with Fisher's identity in (6), to obtain an unbiased estimator of the gradient $\nabla_{\theta} \ell(\theta)$.

2.3 Gaussian variational distribution with factor decomposition

Our paper uses a d-variate Gaussian distribution $\mathcal{N}(\theta;\mu,\Sigma)$, with d the number of parameters, for the VB distribution $q_{\lambda}(\theta)$. If necessary, all the parameters can be transformed so that it is appropriate to approximate the posterior of the transformed parameters by a normal distribution. This simplifies the stochastic VB procedure and avoids the factorization assumption in conventional VB that ignores the posterior dependence between the parameter blocks. Using a Gaussian approximation is motivated by the Bernstein von Mises theorem (Chen, 1985), which states that the posterior of θ is approximately Gaussian when n is large. Therefore, using a Gaussian variational distribution results in a highly accurate approximation of the posterior distribution.

We also use a factor decomposition for Σ ,

$$\Sigma = BB' + c^2 I_d,\tag{7}$$

with B a $d \times 1$ -vector, c a scalar and I_d the $d \times d$ identity matrix. This factor decomposition helps to reduce the total number of VB parameters from d+d(d+1)/2 in the conventional parameterization $\lambda = (\mu, \Sigma)$ to 2d+1 in the parameterization $\lambda = (\mu, B, c)$. The factorization also helps overcome the problem of obtaining a negative-definite Σ if it is updated directly. It is possible to use the factor decomposition (7) where B is a $d \times k$ matrix and k is the number of factors, with k chosen by some model selection criterion (Tan and Nott, 2017). We use k=1 in this paper. Because B is a vector, we are able to find a closed-form expression for the inverse of the Fisher information matrix $I_F(\lambda)$ used in Algorithm 1. This closed-form expression leads to a significant gain in computational efficiency in Big Data settings. The closed-form expression for $I_F(\lambda)^{-1}$ is in the Appendix.

Initializing λ

Let $\widehat{\theta}_{n_{\mathrm{sub}}}$ be an estimate of θ based on a subsample of size n_{sub} from the full data of size n, and let $I_{F,n_{\mathrm{sub}}}$ be the observed Fisher information matrix based on this subsample. We estimate the observed Fisher information matrix based on the full data by $I_{F,n}:=(n/n_{\mathrm{sub}})I_{F,n_{\mathrm{sub}}}$. Let $\widehat{\Sigma}:=I_{F,n}^{-1}$. Motivated by the Bernstein von Mises theorem (see, e.g. Chen, 1985), which states that the posterior of θ is approximately Gaussian when n is large, we initialize the VB distribution $q_{\lambda}(\theta)$ by the Gaussian distribution with mean $\widehat{\theta}_{n_{\mathrm{sub}}}$ and covariance matrix $\widehat{\Sigma}$. Let (ν_i, ν_i) , $\nu_1 \geq \nu_2 \geq \ldots$, be the pairs of eigenvalues ν_i with the corresponding eigenvectors v_i of $\widehat{\Sigma}$. Because $\widehat{\Sigma} = \sum_i \nu_i v_i v_i'$, we initialize B by $B = \sqrt{\nu_1} v_1$ and c by $c = [\sum (\mathrm{diag}(\widehat{\Sigma} - BB'))/d]^{1/2}$. Then $BB' + c^2 I_d \approx \widehat{\Sigma}$. The mean μ is initialized by $\widehat{\theta}_{n_{\mathrm{sub}}}$.

2.4 Randomised Quasi Monte Carlo

Numerical integration using quasi-Monte Carlo (QMC) methods has proved in many cases to be more efficient than standard Monte Carlo methods (Niederreiter, 1992; Dick and Pillichshammer, 2010; Glasserman, 2004). Standard Monte Carlo estimates

the integral of interest based on i.i.d points from the uniform distribution $u_s \sim U[0,1]^d$. QMC chooses these points deterministically and more evenly in the sense that they minimize the so-called star discrepancy of the point set. Randomized QMC (RQMC) then adds randomness to these points such that the resulting points preserve the low-discrepancy property and, at the same time, they have a uniform distribution marginally. Introducing randomness into QMC points is important in order to obtain statistical properties such as unbiasedness and central limit theorems. Our paper generates RQMC numbers using the scrambled net method of Matousek (1998). We use RQMC to sample $\epsilon \sim p_{\epsilon}(\cdot)$. For the panel data example, we also use RQMC to obtain unbiased estimates of the likelihood and the gradient of the log-likelihood.

3 Experimental studies

3.1 The US airlines data

We use the airline on-time performance data from the 2009 ASA Data Expo to demonstrate the VBILL methodology. This is a massive dataset that exceeds the memory (RAM) of a single desktop computer. The dataset, used previously by Wang et al. (2015) and Kane et al. (2013) among others, consists of the flight arrival and departure details for all commercial flights within the USA, from October 1987 to April 2008. The full dataset, ignoring the missing values, has 22,347,358 observations.

The response variable of the logistic regression model is late arrival, which is set to 1 if a flight is late by more than 15 minutes and 0 otherwise. There are three covariates. The two binary covariates are: night (1 if the departure occurred at night and 0 otherwise) and weekend (1 if the departure occurred on weekend and 0 otherwise). The third covariate is distance, which is the distance from origin to destination (in 1000 miles).

We first compare the performance of VBILL with data subsampling to MCMC for a subset of 1113638 observations from the full dataset. This "moderate" data

example allows us to run the "gold standard" MCMC, so that we are able to assess the accuracy of VBILL. The MCMC chain, based on the adaptive random walk Metropolis-Hastings algorithm in Haario et al. (2001), consists of 30000 iterates with another 10000 iterates used as burn-in. We also compare the use of RQMC and MC. We use a diffuse normal prior $N(0,50I_4)$ for the coefficient vector β .

The central value $\bar{\theta}$ is set as the MLE of β based on 30% of the full dataset (the one with 1113638 observations). For the number of samples S in (1), we use $S = 2^8$ for both MC and RQMC. Choosing S as a power of 2 (with 2 the base of the Sobol' sequence in the scrambled net method of Matousek (1998)) is convenient for the RQMC method. We run this example on a single desktop with 4 local processors. We set the threshold $\varepsilon = 10^{-7}$.

Table 1 summarizes the estimation results and, in particular, reports the posterior means and posterior standard deviations of the parameters for the VBILL and MCMC methods. The results for VBILL are obtained by averaging over 100 replications. For VBILL, we use a subsample size m=10000 and 20000, i.e. 1% and 2% of the full dataset. The table shows that the VBILL estimates are very close to the "gold standard" MCMC estimates, but that VBILL is around 20 times faster than MCMC in this moderate data example. The more processors we have, the faster the VBILL method will be. In general, VBILL with RQMC converges with fewer iterations than VBILL with MC. Figures 1 and 2 plot the MCMC and VBILL estimates of the marginal posterior densities of $p(\beta|y)$. The MCMC density estimates are obtained using the Matlab kernel density function ksdensity. The figures show that the VBILL estimates are very close to the MCMC estimates.

To study the stability of VBILL, Table 2 reports the standard errors, estimated over 100 replications, of the VBILL estimates of the posterior means and posterior standard deviations. Clearly, the standard errors decrease when the subsample size m increases. These standard errors suggest that the VBILL approach in this example is stable in the sense that the VBILL estimates across different runs stay the same

up to at least the second decimal place. We can also reduce these standard errors by using a smaller threshold ε .

Table 1: Logistic regression model (n=1113638). The table summarizes the estimates of the posterior means and the posterior standard deviations (in brackets). The VBILL results are obtained by averaging over 100 replications. The CPU time (in minutes) and the number of iterations for VBILL are averaged over the replications.

		MCMC		VB	ILL	
Parameter	$\overline{ heta}$		M	[C	RQ	MC
β_0	-1.598	-1.613 (0.004)	-1.609 (0.004)	-1.609 (0.004)	-1.609 (0.004)	$\frac{-1.609}{^{(0.004)}}$
eta_1	-0.175	-0.155 (0.006)	-0.158 (0.004)	-0.159 (0.004)	-0.159 (0.004)	-0.159 (0.004)
eta_2	0.051	0.089 (0.004)	0.086 (0.004)	0.085 (0.004)	0.085 (0.004)	0.085 (0.004)
eta_3	0.805	0.764 (0.007)	$\underset{(0.007)}{0.766}$	$\underset{(0.007)}{0.766}$	$0.766 \atop \scriptscriptstyle (0.007)$	$\underset{(0.007)}{0.767}$
\overline{m}			1%	2%	1%	2%
Iter.		40000	52	29	35	29
CPU time		20.23	1.52	1.03	1.07	0.99

Table 2: Logistic regression model (n = 1113638). Monte Carlo standard errors of the estimates over 100 replications. The results show that VBILL is stable in the sense that the VBILL estimates across different runs stay the same up to at least the second decimal place.

Parameter	VBIL	L-MC	VBILL-RQMC		
$\mathbb{E}(eta_0 y)$	0.0037	0.0027	0.0035	0.0029	
$\mathbb{E}(eta_1 y)$	0.0036	0.0027	0.0034	0.0028	
$\mathbb{E}(eta_2 y)$	0.0036	0.0025	0.0033	0.0027	
$\mathbb{E}(\beta_3 y)$	0.0031	0.0021	0.0028	0.0023	
$\mathbb{V}(eta_0 y)$	0.2808×10^{-5}	0.1552×10^{-5}	0.2103×10^{-5}	0.1639×10^{-5}	
$\mathbb{V}(\beta_1 y)$	0.2436×10^{-5}	0.1359×10^{-5}	0.1816×10^{-5}	0.1411×10^{-5}	
$\mathbb{V}(eta_2 y)$	0.2356×10^{-5}	0.1321×10^{-5}	0.1771×10^{-5}	0.1392×10^{-5}	
$\mathbb{V}(eta_3 y)$	0.6523×10^{-5}	0.3481×10^{-5}	0.4756×10^{-5}	0.3653×10^{-5}	
$\overline{}$	1%	2%	1%	2%	

Figure 1: Logistic regression model (n = 1113638, m = 10000): Plots of the MCMC and VBILL estimates of the marginal posteriors $p(\beta_j|y)$.

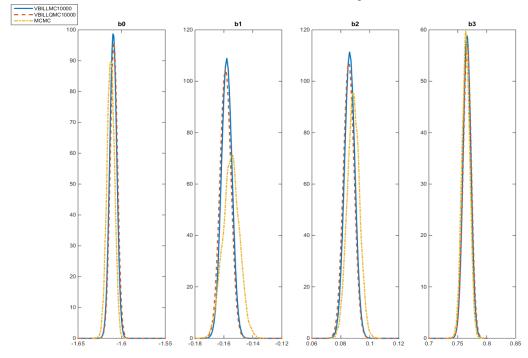
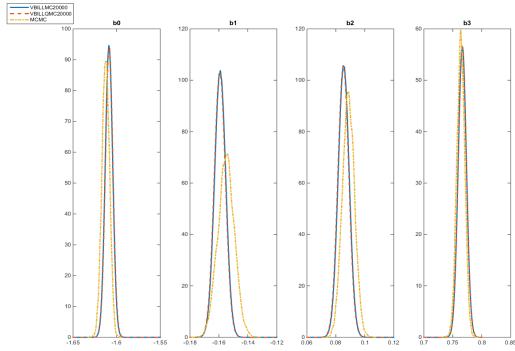


Figure 2: Logistic regression model (n = 1113638, m = 20000): Plots of the MCMC and VBILL estimates of the marginal posteriors $p(\beta_j|y)$.



We now run VBILL for the full dataset, which exceeds the memory of a single desktop computer. Given our computational facilities, it is computationally infeasible to run MCMC to fit the model using such a large dataset. We use the MapReduce programming technique in Matlab to overcome the computer's memory issue. MapReduce is available in the R2014b release of Matlab. The MapReduce programming model has three components

- A datastore function that reads and organizes the dataset into small chunks for the "map" function.
- A map function calculates the quantities of interest for each individual chunk.
 MapReduce calls the map function once for each data chunk organized by datastore.
- A reduce function aggregates outputs from the map function and produces the final results.

The datastore function splits the full dataset into K chunks randomly, each fits into the memory of a single desktop computer. The log-likelihood, its gradient and Hessian can be decomposed as

$$\ell(\theta) = \sum_{k=1}^{K} l^{(k)}(\theta), \ \nabla_{\theta} \ell(\theta) = \sum_{k=1}^{K} \nabla_{\theta} l^{(k)}(\theta), \ \nabla_{\theta\theta'}^{2} \ell(\theta) = \sum_{k=1}^{K} \nabla_{\theta\theta'}^{2} l^{(k)}(\theta),$$

where $l^{(k)}(\theta)$, $\nabla_{\theta}l^{(k)}(\theta)$, and $\nabla^2_{\theta\theta'}l^{(k)}(\theta)$ are the log-likelihood contribution and its gradient and Hessian based on data chunk k. Similarly, we denote by $\hat{l}_{m_k}(\theta)$ and $\widehat{\nabla_{\theta}l_{m_k}(\theta)}$ the unbiased estimator of $l^{(k)}(\theta)$ and $\nabla_{\theta}l^{(k)}(\theta)$, based on a random subset of size m_k from data chunk k. The map function is used to calculate the chunk based estimate $\hat{l}_{m_k}(\theta)$ and $\widehat{\nabla_{\theta}l_{m_k}(\theta)}$ for each chunk k. Then, the reduce function aggregates all the chunk-based unbiased estimates into the full data based estimate

of the gradient of the log-likelihood

$$\widehat{G}_{m}(\theta) = \sum_{k=1}^{K} \widehat{\nabla_{\theta} l_{m_{k}}(\theta)}.$$
(8)

Then,

$$\mathbb{E}\left(\widehat{G}_{m}(\theta)\right) = \sum_{k=1}^{K} \mathbb{E}\left(\widehat{\nabla_{\theta} l_{m_{k}}(\theta)}\right) = \sum_{k=1}^{K} \nabla_{\theta} l^{(k)}(\theta) = \nabla_{\theta} l(\theta).$$

We note that this method is computer-memory efficient in the sense that the full dataset does not need to remain on-hold and can be stored in different places. It is important to note that our VBILL estimator is mathematically justified and independent of data chunking, as the estimator $\widehat{G}_m(\theta)$ in (8) is guaranteed to be unbiased.

The central value $\bar{\theta}$ is the MLE based on 1 million observations of the full dataset and is given in Table 3. We use approximately 5% of the data in each subset. We estimate the gradient of the lower bound using RQMC and set $S=2^8$. The VBILL method stopped after 24 iterations and the running time was 77.55 minutes. Table 3 shows the results and Figure 3 shows the marginal posterior density estimates of the parameters, which are bell-shaped with a very small variance as expected with a large dataset. This example demonstrates that the VBILL methodology is useful for Bayesian inference for Big Data.

Table 3: Logistic regression model (n=22,347,358). Estimates of the posterior means and the posterior standard deviations (in brackets). The CPU time is in minutes.

Param.	$\overline{ heta}$	VBILL-RQMC
$\overline{\beta_0}$	-1.613	-1.608
0	0.155	(0.0008)
eta_1	-0.155	-0.154 (0.0008)
eta_2	0.088	0.084
, -		(0.0008)
eta_3	0.764	$\underset{(0.0015)}{0.770}$
Iter.		24
CPU time		77.55

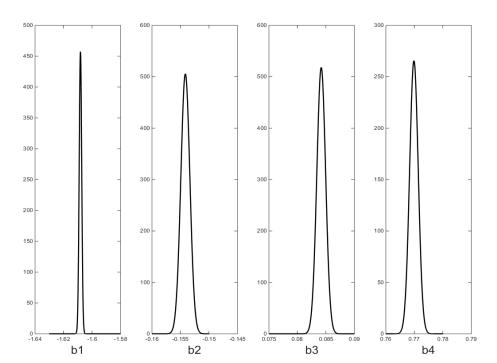


Figure 3: Logistic regression model (n=22,347,358).

3.2 Simulation Study: Panel Data Model

This section studies the performance of the VBILL method for the panel data model.

Data are generated from the following logistic model with a random intercept

$$p(y_{it}|\beta, \alpha_i) = \text{Binomial}(1, p_{it}),$$

 $\text{logit}(p_{it}) = x'_{it}\beta + \alpha_i, \ \alpha_i \sim N(0, \tau^2),$

for i=1,...,n and t=1,...,5. We generate two datasets of n=1000 and n=10000 with $x_{1,it},...,x_{10,it} \sim U(0,1)$. Let $\gamma = \log(\tau^2)$, so the model parameters are $\theta = (\beta,\gamma)$. We use a diffuse normal prior $N(0,50I_d)$ for θ ; d=12 in this example.

The performance of VBILL is compared to the pseudo-marginal MCMC simulation method (Andrieu and Roberts, 2009), which is still able to generate samples from the posterior when the likelihood in the Metropolis-Hastings algorithm is replaced by its unbiased estimator. The likelihood in the panel data context is a product

of n integrals over the random effects. Each integral is estimated unbiasedly using importance sampling, with the number of importance samples chosen such that the variance of unbiased likelihood estimator is approximately 1 (Pitt et al., 2012). Each MCMC chain consists of 30000 iterates with another 10000 used as burn-in iterates.

For VBILL, the central value $\overline{\theta}$ is a simulated maximum likelihood estimate of θ based on a 30% randomly selected subset of the full dataset. We set $S=2^8$ and use both MC and RQMC in VBILL. The number of importance samples used to estimate integrals in (6) is $N=2^8$, and $\varepsilon=10^{-5}$. Tables 4 summarizes the performance results for the three methodologies: pseudo-marginal MCMC, VBILL-MC and VBILL-RQMC for various values of subsample size m. For VBILL, the results are obtained by averaging over 100 replications. The VBILL estimates are close to the MCMC estimates and they are all close to the true values. However, VBILL is much faster than MCMC. Figures 4, 5, and 6 plot the VBILL and MCMC estimates of the marginal posterior of the parameters. The three figures show that the VBILL marginal posterior estimates are very close to the MCMC estimates for both MC and RQMC and for all subsample sizes. We note that VBILL with RQMC takes longer to run compared to VBILL with MC, with not much difference in the number of iterations and also the resulting marginal posterior estimates in this example. This is because generating RQMC numbers takes a longer time than generating plain pseudo random numbers.

Table 4: Panel data example (n=1000): Posterior mean estimates (with posterior standard deviation in brackets). The VBILL results are obtained by averaging over 100 replications. The CPU time (in minutes) and the number of iterations for VBILL are averaged over 100 replications.

Param. True θ MC RQMC $β_0$ -1.5 -1.72 -1.64 -1.67 <th></th> <th></th> <th></th> <th>MCMC</th> <th></th> <th></th> <th>VB</th> <th>VBILL</th> <th></th> <th></th>				MCMC			VB	VBILL		
$\begin{array}{cccccccccccccccccccccccccccccccccccc$	Param.	True	θ			MC			RQMC	
$\begin{array}{cccccccccccccccccccccccccccccccccccc$	β_0	-1.5	-1.72	-1.64	-1.67	-1.67	-1.67	-1.670	-1.67	-1.67
$\begin{array}{cccccccccccccccccccccccccccccccccccc$	3	<u>-</u>	1 73	1.67	1.70	1.70	1 70	1.69	1.69	1.69
$\begin{array}{cccccccccccccccccccccccccccccccccccc$	\leq 1	0:1	0	(0.17)	(0.13)	(0.14)	(0.13)	(0.13)	(0.13)	(0.13)
$\begin{array}{cccccccccccccccccccccccccccccccccccc$	eta_2	0.5	0.22	0.16	0.18	0.18	0.18	0.17	0.17	0.17
$\begin{array}{cccccccccccccccccccccccccccccccccccc$				(0.15)	(0.12)	(0.12)	(0.12)	(0.12)	(0.12)	(0.12)
$\begin{array}{cccccccccccccccccccccccccccccccccccc$	eta_3	0.25	0.22	$0.33_{(0.16)}$	0.29	$0.29 \\ (0.12)$	0.29	0.32 (0.13)	$0.32 \\ (0.12)$	0.32 (0.12)
$\begin{array}{cccccccccccccccccccccccccccccccccccc$	β_{λ}	0.3	0.52	0.42	0.48	0.48	0.48	0.46	0.46	0.46
$\begin{array}{cccccccccccccccccccccccccccccccccccc$	#			(0.16)	(0.13)	(0.13)	(0.13)	(0.13)	(0.12)	(0.12)
$\begin{array}{cccccccccccccccccccccccccccccccccccc$	eta_5	8.0	0.69	0.78	0.75	0.75	0.75	0.77	0.77	0.77
$\begin{array}{cccccccccccccccccccccccccccccccccccc$				(0.16)	(0.13)	(0.13)	(0.13)	(0.13)	(0.13)	(0.13)
$\begin{array}{cccccccccccccccccccccccccccccccccccc$	eta_6	0.45	0.46	$\underset{(0.15)}{0.61}$	$0.55 \\ (0.13)$	$0.55 \\ (0.13)$	$0.55 \\ (0.13)$	$0.59 \\ (0.13)$	0.59 (0.13)	$0.59 \\ (0.12)$
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$	β_7	0.85	1.03	0.75	0.89	0.89	0.89	0.83	0.82	0.82
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$				(0.16)	(0.13)	(0.13)	(0.13)	(0.13)	(0.12)	(0.12)
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$	β_8	0.75	0.46	0.58	0.53	0.53	0.53	0.56	0.56	$0.5\overline{6}$
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$				(0.16)	(0.13)	(0.13)	(0.13)	(0.13)	(0.13)	(0.13)
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$	β_9	0.67	1.24	0.95	1.09	1.09	1.09	1.02	1.02	1.01
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$				(0.15)	(0.13)	(0.13)	(0.13)	(0.13)	(0.13)	(0.13)
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$	eta_{10}	1.5	1.15	$1.36_{(0.17)}$	$\frac{1.27}{0.13}$	$\frac{1.27}{0.13}$	$\frac{1.27}{0.13}$	$1.33_{\tiny (0.13)}$	$\begin{array}{c} 1.33 \\ 0.13 \end{array}$	$\frac{1.33}{0.13}$
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$		7		(0.1.0)	(0:10)	(2:52)	(0.15)	(0.15)	(0.10)	(0:10)
. 40000 24 22 25 25 26 26 26 286 1.88 3.12	~	0.41	09:0	0.41 (0.13)	$0.47 \\ (0.12)$	$0.47 \\ (0.12)$	$0.47 \\ (0.12)$	0.47 (0.12)	$0.46 \\ (0.12)$	$0.46 \\ (0.12)$
. 40000 24 22 25 25 26 ime 80 1.09 1.62 2.86 1.88 3.12	m				50	100	200	50	100	200
2.86 1.88	Iter.			40000	24	22	22	25	26	26
	CPU time			80	1.09	1.62	2.86	1.88	3.12	5.40

Figure 4: Panel data example ($n\!=\!1000$): comparing MCMC and VBILL estimates with $m\!=\!50.$

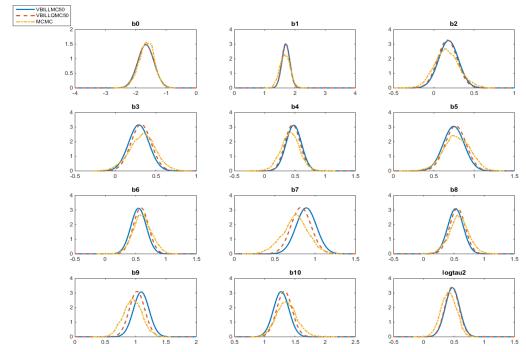


Figure 5: Panel data example ($n\!=\!1000$): comparing MCMC and VBILL estimates with $m\!=\!100$.

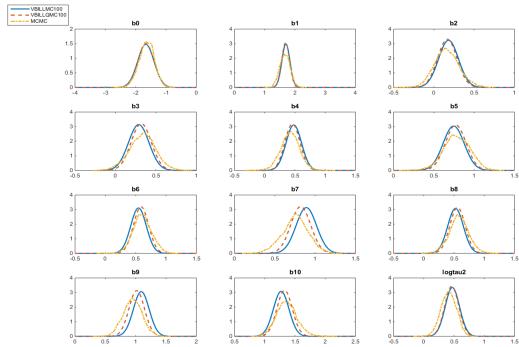
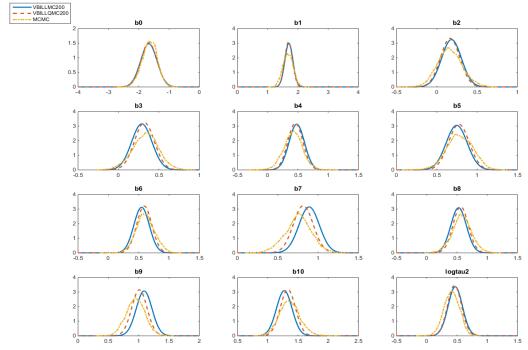


Figure 6: Panel data example (n = 1000): comparing the MCMC and VBILL estimates with m = 200.



Larger Panel Data Example

This section describes a scenario where it is difficult to use the pseudo-marginal MCMC method. We consider a large data case with the number of panels n=10000. The MCMC method is extremely expensive since the variance of the unbiased estimator of the likelihood is large and it requires approximately N=8000 importance samples in order to target the optimal variance of 1 (Pitt et al., 2012). Hence, if an optimal MCMC procedure is run on our computer to generate 40000 iterations, it would take 4053.30 minutes. We run the VBILL-MC and VBILL-RQMC with the subsample size m=500 and $\varepsilon=10^{-6}$. Table 5 summarizes the results as averages over 100 replications. Both methods converge on average after 28 and 25 iterations respectively, and result in similar estimates of the posterior mean and standard deviation. The times taken are 8.65 and 13.14 minutes for VBILL-MC and VBILL-RQMC, respectively. The MC errors of the estimates, based on 100 replications, suggest that using RQMC to estimate integrals in (6) helps to stablize the VBILL

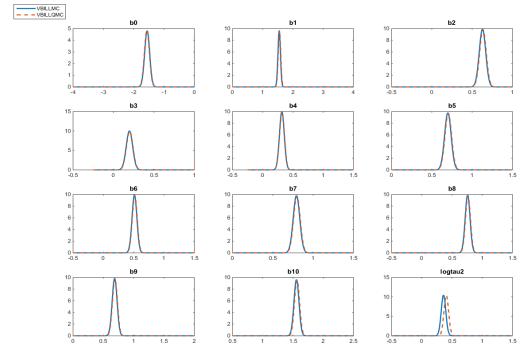
estimates.

Figure 7 plots the variational approximations of the marginal posteriors of the parameters, which are bell-shaped with small posterior variance as expected with a very large dataset. The two methods produce very similar results. This example demonstrates that VBILL offers a computationally efficient approach for Big Panel Data problems.

Table 5: Panel data example (n=10000): The table shows the estimates of posterior mean (first line) and posterior standard deviation (second line). The numbers in brackets are MC standard errors over 100 replications, which suggests that VBILL-RQMC is more stable than VBILL-MC. The CPU time and number of iterations are averaged over the replications.

Param.	True	$\overline{ heta}$	VBILL-MC	VBILL-RQMC
	-1.5	-1.71	-1.56 (0.0261)	-1.57 (0.0178)
			$0.083 \ (0.82 \times 10^{-3})$	$0.083 \ (0.82 \times 10^{-3})$
eta_1	1.5	1.60	$1.54 \ (0.0107)$	1.55 (0.0064)
			$0.041 \ (0.30 \times 10^{-3})$	$0.042 \ (0.25 \times 10^{-3})$
eta_2	0.5	0.71	0.63(0.0171)	$0.63 \ (0.0133)$
			$0.040 \ (0.29 \times 10^{-3})$	$0.041 \ (0.25 \times 10^{-3})$
eta_3	0.25	0.24	$0.20 \ (0.0069)$	$0.20 \ (0.0057)$
			$0.040 \ (0.28 \times 10^{-3})$	$0.040 \ (0.24 \times 10^{-3})$
eta_4	0.3	0.34	$0.32\ (0.0039)$	$0.32\ (0.0030)$
			$0.040 \ (0.29 \times 10^{-3})$	$0.040 \ (0.24 \times 10^{-3})$
eta_5	0.8	0.69	$0.70 \ (0.0041)$	$0.70 \ (0.0033)$
			$0.041 \ (0.29 \times 10^{-3})$	$0.041 \ (0.25 \times 10^{-3})$
eta_6	0.45	0.56	$0.51 \ (0.0087)$	$0.51 \ (0.0066)$
			$0.040 \ (0.29 \times 10^{-3})$	$0.041 \ (0.25 \times 10^{-3})$
eta_7	0.85	0.80	$0.79 \ (0.0022)$	$0.80 \ (0.0008)$
			$0.041 \ (0.29 \times 10^{-3})$	$0.041 \ (0.25 \times 10^{-3})$
eta_8	0.75	0.79	$0.76 \ (0.0059)$	$0.76 \ (0.0037)$
			$0.040 \ (0.29 \times 10^{-3})$	$0.041 \ (0.25 \times 10^{-3})$
eta_9	0.67	0.70	$0.69 \ (0.0021)$	$0.69 \ (0.0006)$
			$0.040 \ (0.29 \times 10^{-3})$	$0.041 \ (0.25 \times 10^{-3})$
eta_{10}	1.5	1.56	$1.56 \ (0.0028)$	$1.57 \ (0.0034)$
			$0.041 \ (0.30 \times 10^{-3})$	$0.042 \ (0.25 \times 10^{-3})$
γ	0.41	0.35	$0.36 \ (0.0117)$	$0.41 \ (0.0070)$
			$0.038 \ (0.27 \times 10^{-3})$	$0.038 \ (0.23 \times 10^{-3})$
\overline{m}			500	500
Iteration			28	25
CPU time (min)			8.65	13.14

Figure 7: Panel data example (n = 10000): comparing the MCMC and VBILL estimates with m = 500



4 Conclusions

We propose the VBILL approach for Bayesian inference when the likelihood function is intractable but the gradient of the log-likelihood can be estimated unbiasedly. The method is useful for Big Data situations where it is convenient to obtain unbiased estimates of the gradient of the log-likelihood.

Unlike MCMC approaches that can in principle sample from the exact posterior, VBILL, as a variant of VB, is an approximate method for estimating the posterior distribution of the parameters. The main advantage of VBILL is that it is much more computationally efficient than MCMC, but produces very similar results to MCMC as shown in the simulated and real examples. To the best of our knowledge, in Big Data situations, there is no MCMC approach in the current literature that is both computationally efficient and able to sample from the exact posterior. Conventional MCMC is exact, but is well-known to be extremely expensive in Big Data, while

data subsampling MCMC can be computationally efficient but is not exact (Bardenet et al., 2015; Quiroz et al., 2015). For these reasons, we believe that VBILL can be the method of choice for Big Data applications.

Acknowledgement

The work of David Gunawan and Robert Kohn was partially supported by an ARC Center of Excellence Grant CE140100049 and an ARC Discovery grant DP150104630. We would like to thank two anonymous referees and an annonymous associate editor for greatly helping to improve the content and presentation of the paper.

Appendix

Closed-form expression for $I_F(\lambda)^{-1}$

The Fisher information matrix $I_F(\lambda) = \text{cov}_{q_{\lambda}}(\nabla_{\lambda} \log q_{\lambda}(\theta))$, where

$$\log q_{\lambda}(\theta) \propto -\frac{1}{2} \log |\Sigma| - \frac{1}{2} (\theta - \mu)' \Sigma^{-1} (\theta - \mu),$$

with $\Sigma = BB' + c^2 I_d$. Let U = U(x) be a matrix-valued function of a scalar x, and g(U) a real-valued function of U. Then, by the chain rule (Petersen and Pedersen, 2012) $\nabla_x (g(U(x))) = \text{tr} \left[\nabla_U (g(U))' \nabla_x (U(x)) \right]$. Noting that, for vectors a and b, $\nabla_U (a'U^{-1}b) = -U^{-1}ab'U^{-1}$ (Petersen and Pedersen, 2012), we have

$$\nabla_B \Big((\theta - \mu)' \Sigma^{-1} (\theta - \mu) \Big) = -2\Sigma^{-1} (\theta - \mu) (\theta - \mu)' \Sigma^{-1} B.$$

$$\begin{split} &\nabla_x(\log|U|) = \operatorname{tr}(U^{-1}\nabla_x U) \text{ because } \nabla_U(\log|U|) = U^{-1}. \text{ Hence, } \nabla_B \Big(\log|\Sigma|\Big) = 2\Sigma^{-1}B. \\ &\operatorname{Similarly, } \nabla_c \Big(\log|\Sigma|\Big) = 2c \times \operatorname{tr}(\Sigma^{-1}) \text{ and } \nabla_c \Big((\theta-\mu)'\Sigma^{-1}(\theta-\mu)\Big) = -2c(\theta-\mu)'\Sigma^{-2}(\theta-\mu)' = 0. \end{split}$$

 μ). Hence,

$$\nabla_{\lambda} \log q_{\lambda}(\theta) = \begin{pmatrix} \nabla_{\mu} \log q_{\lambda}(\theta) \\ \nabla_{B} \log q_{\lambda}(\theta) \\ \nabla_{c} \log q_{\lambda}(\theta) \end{pmatrix} = \begin{pmatrix} \Sigma^{-1}(\theta - \mu) \\ -\Sigma^{-1}B + \Sigma^{-1}(\theta - \mu)(\theta - \mu)'\Sigma^{-1}B \\ -c \times \operatorname{tr}(\Sigma^{-1}) + c(\theta - \mu)'\Sigma^{-2}(\theta - \mu) \end{pmatrix}.$$
(9)

Let $X = \Sigma^{-1}(\theta - \mu) \sim \mathcal{N}(0, \Sigma^{-1})$. Using the results on cubic and quadratic forms of a Gaussian random vector (Petersen and Pedersen, 2012), it can be shown that

$$I_{F}(\lambda) = \begin{pmatrix} \Sigma^{-1} & O_{d \times d} & O_{d \times 1} \\ O_{d \times d} & \Sigma^{-1} B B' \Sigma^{-1} + (B' \Sigma^{-1} B) \Sigma^{-1} & 2c \Sigma^{-2} B \\ O_{1 \times d} & 2c B' \Sigma^{-2} & 2c^{2} \operatorname{tr}(\Sigma^{-2}) \end{pmatrix}.$$
(10)

To compute the inverse matrix $I_F(\lambda)^{-1}$, we first need some preliminary results. Let A be a $d \times d$ matrix, b a $d \times 1$ vector and ω a scalar. Then,

$$(A+bb')^{-1} = A^{-1} - \frac{1}{1+b'A^{-1}b}A^{-1}bb'A^{-1}$$

and

$$\begin{pmatrix} A & b \\ b' & \omega \end{pmatrix}^{-1} = \begin{pmatrix} A^{-1} + \frac{1}{c_2} A^{-1} b b' A^{-1} & -\frac{1}{c_2} A^{-1} b \\ -\frac{1}{c_2} b' A^{-1} & \frac{1}{c_2} \end{pmatrix}, \text{ with } c_2 = \omega - b' A^{-1} b.$$

Then,

$$\Sigma^{-1} = (BB' + c^2I)^{-1} = \frac{1}{c^2} \left(I - \frac{1}{c^2 + B'B}BB' \right),$$

and

$$\Sigma^{-1}B = \alpha B$$
, with $\alpha = 1/(c^2 + B'B)$.

The Fisher information matrix in (10) can written as

$$I_F(\lambda) = \begin{pmatrix} \Sigma^{-1} & O_{d \times d} & O_{d \times 1} \\ O_{d \times d} & A & b \\ O_{1 \times d} & b' & \omega \end{pmatrix}$$

with

$$\begin{split} A &= \Sigma^{-1}BB^{T}\Sigma^{-1} + (B^{T}\Sigma^{-1}B)\Sigma^{-1} = \alpha^{2}BB' + \alpha(B'B)\Sigma^{-1}, \\ b &= 2c\Sigma^{-2}B = 2c\alpha^{2}B = \frac{2c}{(c^{2} + B'B)^{2}}B, \\ \omega &= 2c^{2}\mathrm{tr}(\Sigma^{-2}) = \frac{2}{c^{2}}\left[d - 1 + \left(\frac{c^{2}}{c^{2} + B'B}\right)^{2}\right]. \end{split}$$

We have

$$A^{-1} = \left[\left(1 + \frac{c^2}{B'B} \right) - \frac{1}{2} \left(1 + \frac{c^2}{B'B} \right)^2 \right] BB' + c^2 \left(1 + \frac{c^2}{B'B} \right) I_d, \text{ and } A^{-1}b = \kappa B,$$

with

$$\kappa = \left[\left(1 + \frac{c^2}{B'B} \right) - \frac{1}{2} \left(1 + \frac{c^2}{B'B} \right)^2 \right] \frac{2c(B'B)}{(c^2 + B'B)^2} + \frac{2c^3}{B'B(c^2 + B'B)}.$$

Finally,

$$I_F(\lambda)^{-1} = \begin{pmatrix} BB' + c^2 I_d & O_{d \times d} & O_{d \times 1} \\ O_{d \times d} & A^{-1} + \frac{\kappa^2}{c_2} BB' & -\frac{\kappa}{c_2} B \\ O_{1 \times d} & -\frac{\kappa}{c_2} B' & 1/c_2 \end{pmatrix},$$

with

$$c_2 = \frac{2}{c^2} \left[d - 1 + \left(\frac{c^2}{c^2 + B'B} \right)^2 \right] - \frac{2c\kappa B'B}{(c^2 + B'B)^2}.$$

Computing $\nabla_{\lambda}A(\lambda)$

If we use a normal prior $p(\theta) = \mathcal{N}(0, \sigma_0^2 I_d)$, then

$$A(\lambda) = \mathbb{E}_{q_{\lambda}} \left(\log \frac{p(\theta)}{q_{\lambda}(\theta)} \right)$$

$$= -\frac{1}{2\sigma_{0}^{2}} \left(\mu' \mu + \text{tr}(BB' + c^{2}I) \right) - \frac{1}{2} \log |BB' + c^{2}I| + \text{constants}$$

$$= -\frac{1}{2\sigma_{0}^{2}} (\mu' \mu + B'B + dc^{2}) - \frac{1}{2} \log \left(c^{2d} \left(1 + B'B/c^{2} \right) \right) + \text{constants}.$$

Hence,

$$\nabla_{\lambda} A(\lambda) = \begin{pmatrix} \nabla_{\mu} A(\lambda) \\ \nabla_{B} A(\lambda) \\ \nabla_{c} A(\lambda) \end{pmatrix} = \begin{pmatrix} -\frac{1}{\sigma_{0}^{2}} \mu \\ -\left(\frac{1}{\sigma_{0}^{2}} + \frac{1}{c^{2} + B'B}\right) B \\ -\frac{dc}{\sigma_{0}^{2}} - \frac{1}{c} \left(d - \frac{B'B}{c^{2} + B'B}\right) \end{pmatrix}.$$

References

Amari, S. (1998). Natural gradient works efficiently in learning. *Neural computation*, 10(2):251–276.

Andrieu, C. and Roberts, G. (2009). The pseudo-marginal approach for efficient Monte Carlo computations. *The Annals of Statistics*, 37:697–725.

Apostol, T. (1969). Calculus, Vol. 2: Multi-Variable Calculus and Linear Algebra with Applications to Differential Equations and Probability. John Wiley & Sons, 2nd edition.

Attias, H. (1999). Inferring parameters and structure of latent variable models by variational Bayes. In *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence*, pages 21–30.

Bardenet, R., Doucet, A., and Holmes, C. (2015). On Markov chain Monte Carlo methods for tall data. arXiv:1505.02827v1.

- Battey, H., Fan, J., Liu, H., Lu, J., and Zhu, Z. (2015). Distributed estimation and inference with statistical guarantees. Technical report. arXiv:1509.05457v1.
- Bishop, C. M. (2006). Pattern Recognition and Machine Learning. New York: Springer.
- Cappe, O., Moulines, E., and Ryden, T. (2005). Inference in Hidden Markov Models.
 New York: Springer.
- Chen, C.-F. (1985). On asymptotic normality of limiting density functions with Bayesian implications. Journal of the Royal Statistical Society. Series B (Methodological), 47(3):540–546.
- Dick, J. and Pillichshammer, F. (2010). Digital nets and sequence. Discrepancy theory and quasi-Monte Carlo integration. Cambridge University Press, Cambridge.
- Fitzmaurice, G. M., Laird, N. M., and Ware, J. H. (2011). *Applied Longitudinal Analysis*. John Wiley & Sons, Ltd, New Jersey, 2nd edition.
- Flury, T. and Shephard, N. (2011). Bayesian inference based only on simulated likelihood: Particle filter analysis of dynamic economic models. *Econometric Theory*, 1:1–24.
- Glasserman, P. (2004). Monte Carlo Methods in Financial Engineering. Springer-Verlag, New York.
- Haario, H., Saksman, E., and Tamminen, J. (2001). An adaptive Metropolis algorithm. *Bernoulli*, 7:223–242.
- Hoffman, M. D., Blei, D. M., Wang, C., and Paisley, J. (2013). Stochastic variational inference. *Journal of Machine Learning Research*, 14:1303–1347.
- Kane, M. J., Emerson, J., and Weston, S. (2013). Scalable strategies for computing with massive data. *Journal of Statistical Software*, 55 (14):1–19.

- Kingma, D. P. and Welling, M. (2013). Auto-encoding variational bayes. Technical report. https://arxiv.org/abs/1312.6114.
- Matousek, J. (1998). On the L2-discrepancy for anchored boxes. *Journal of Complexity*, 14:527–556.
- Niederreiter, H. (1992). Random Number Generation and Quasi-Monte Carlo Methods. Society for Industrial and Applied Mathematics, Philadelphia.
- Nott, D. J., Tan, S., Villani, M., and Kohn, R. (2012). Regression density estimation with variational methods and stochastic approximation. *Journal of Computational and Graphical Statistics*, 21:797–820.
- Petersen, K. B. and Pedersen, M. S. (2012). The matrix cookbook. http://www2.imm.dtu.dk/pubdb/views/edoc_download.php/3274/pdf/imm3274.pdf.
- Pitt, M. K., Silva, R. S., Giordani, P., and Kohn, R. (2012). On some properties of Markov chain Monte Carlo simulation methods based on the particle filter. *Journal* of *Econometrics*, 171(2):134–151.
- Quiroz, M., Villani, M., Kohn, R., and Tran, M.-N. (2015). Speeding up MCMC by efficient data subsampling. arXiv preprint arXiv:1404.4178v2.
- Robbins, H. and Monro, S. (1951). A stochastic approximation method. *Annals of Mathematical Statistics*, 22:400–407.
- Ruiz, F. J. R., Titsias, M. K., and Blei, D. M. (2016). The generalized reparameterization gradient. Technical report. arXiv:1610.02287v3.
- Sacks, J. (1958). Asymptotic distribution of stochastic approximation procedures.

 The Annals of Mathematical Statistics, 29(2):373–405.
- Sato, M. (2001). Online model selection based on the variational Bayes. *Neural Computation*, 13(7):1649–1681.

- Tan, L. S. L. and Nott, D. (2017). Gaussian variational approximation with sparse precision matrices. *Statistics and Computing*. To appear.
- Tran, M.-N., Nott, D., and Kohn, R. (2017). Variational Bayes with intractable likelihood. *Journal of Computational and Graphical Statistics*. To appear.
- Wang, C., Chen, M. H., Schifano, E., Wu, J., and Yan, J. (2015). A survey of statistical methods and computing for big data. Technical report. arXiv:1502.07989v1.