Quiroz, Matias; Villani, Mattias; Kohn, Robert

Working Paper

Scalable MCMC for large data problems using data subsampling and the difference estimator

Sveriges Riksbank Working Paper Series, No. 306

Provided in Cooperation with:
Central Bank of Sweden, Stockholm

SCALABLE MCMC FOR LARGE DATA PROBLEMS USING DATA SUBSAMPLING AND THE DIFFERENCE ESTIMATOR

MATIAS QUIROZ, MATTIAS VILLANI AND ROBERT KOHN

SCALABLE MCMC FOR LARGE DATA PROBLEMS USING DATA SUBSAMPLING AND THE DIFFERENCE ESTIMATOR

MATIAS QUIROZ, MATTIAS VILLANI AND ROBERT KOHN

ABSTRACT. We propose a generic Markov Chain Monte Carlo (MCMC) algorithm to speed up computations for datasets with many observations. A key feature of our approach is the use of the highly efficient difference estimator from the survey sampling literature to estimate the log-likelihood accurately using only a small fraction of the data. Our algorithm improves on the $O(n)$ complexity of regular MCMC by operating over local data clusters instead of the full sample when computing the likelihood. The likelihood estimate is used in a Pseudo-marginal framework to sample from a perturbed posterior which is within $O(m^{-1/2})$ of the true posterior, where m is the subsample size. The method is applied to a logistic regression model to predict firm bankruptcy for a large data set. We document a significant speed up in comparison to the standard MCMC on the full dataset.

KEYWORDS: Bayesian inference, Markov Chain Monte Carlo, Pseudo-marginal MCMC, estimated likelihood, GLM for large data.

JEL Classification: C11, C13, C15, C55, C83

1. INTRODUCTION

The popularity of Bayesian methods increased significantly in the early 90's due to advances in computer technology and the introduction of powerful simulation algorithms such

as Markov Chain Monte Carlo (MCMC) (Gelfand and Smith, 1990). However, posterior sampling with MCMC is time-consuming and there is an increasing awareness that new scalable algorithms are necessary for MCMC to remain an attractive choice for inference in large data sets. Perhaps the most useful advance in computing for statisticians is parallel computing which are now widely available in most statistical software. However, the inherent serial nature of MCMC algorithms precludes the use of efficient parallelization.

Current research on scalable MCMC algorithms belongs to two major groups. The first group employs parallelism through the typical MapReduce scheme (Dean and Ghemawat, 2008) by partitioning the data and computing posteriors in a parallel and distributed manner. The resulting draws are subsequently combined into a single posterior distribution. The main difference within this group is how weighting is performed and if the partitions communicate at runtime, see for example Scott et al., 2013; Neiswanger et al., 2013; Wang and Dunson, 2013; Minsker et al., 2014.

The second group of methods work with a small sample of the data in each MCMC iteration to speed up the algorithm. Korattikara et al. (2013) develop a M-H algorithm with an approximate accept/reject step, see also Bardenet et al. (2014). Maclaurin and Adams (2014) introduce binary auxiliary variables, one for each observation, that effectively determines which observations are used to compute the posterior. Banterle et al. (2014) use delayed acceptance in several steps so that a first rejection implies a rejection of the proposed value and no further computations are performed. Quiroz et al. (2015) propose subsampling the data using probability proportional-to-size (PPS) sampling to obtain an approximately unbiased estimate of the likelihood which is used in a M-H acceptance step. This approach is in the spirit of a Pseudo-marginal MCMC (PMCMC) algorithm. Andrieu and Roberts (2009) prove that PMCMC with an unbiased likelihood estimate generates samples from the true posterior. Quiroz et al. (2015) show that their algorithm generates samples from a distribution that is within $O(m^{-\frac{1}{2}})$ percent of the true posterior, where m is the subsample size.

Payne and Mallick (2014) combine the distributed and subsampling approaches where the consensus Monte Carlo algorithm (Scott et al., 2013) is used to distribute computations and a two stage M-H sampler is used within each data partition. The two stage M-H sampler uses simple random sampling to obtain a computationally cheap estimate of the likelihood to compute a first M-H ratio. If accepted, the second step uses the true M-H acceptance probability based on all the data, thus avoiding evaluating the full data likelihood for proposals that are unlikely to be accepted.

Our article extends the subsampling approach in Quiroz et al. (2015) in the following directions. First, the likelihood is estimated using the efficient and robust *difference estimator* from the survey sampling literature. We show that this estimator is in the class of estimators considered by Quiroz et al. (2015) and therefore that their theory applies directly. Second, the PPS sampling in Quiroz et al. (2015) requires an approximation of the log-likelihood contribution (log-density) for every observation. The approximation can be relatively costly and in such instances the algorithm is only likely to speed up computations for models with costly density evaluations. To speed up the MCMC sampling also for models with cheap density evaluations, we modify the estimator to operate only on a sparse set of the data and in addition we derive a computationally cheap approximation of the log-likelihood contribution for a large class of models. Third, we propose a sampling scheme that updates the subsample indicators infrequently and demonstrate that this gives a sampler where the inevitable efficiency loss from using an estimated likelihood instead of the full data likelihood is much smaller.

This paper is organized as follows. Section 2 outlines the methodology and discusses connections to previous research. Section 3 applies the method to a large micro-economic data set containing nearly 5 million observations. Section 4 concludes and discusses further research.

2. Methodology

2.1. MCMC with likelihood estimators from data subsampling. Consider a model parametrized by $p(y_k|\theta, x_k)$, where y_k is a potentially multivariate response vector and x_k is a vector of covariates for the kth observation. Let $l_k(\theta) = \log p(y_k|\theta, x_k)$ denote the kth observation's log-density. Given conditionally independent observations, the likelihood function can be written

$$p(y|\theta) = L(\theta) = \exp\left[l(\theta)\right],$$

where $l(\theta) = \sum_{k=1}^{n} l_k(\theta)$ is the log-likelihood function. We note that this setting includes any situation where the log-likelihood can be written as a sum of terms where each term depends on a unique piece of data information. The most obvious examples are longitudinal problems, where $l_k(\theta)$ is the log joint density of all measurements on the kth subject; in this case we would sample subjects rather than individual observations.

Let the set $F = \{1, 2, \ldots, n\}$ contain the indices for all observations in the full data set and let

$$u = (u_1, \ldots u_m), \quad u_i \in F$$

be the $m \times 1$ vector of indices obtained by sampling F with replacement. Sampling with replacement generally gives a slightly higher variance for any estimator compared to that of without replacement sampling. However, it allows us to use the theory developed in Quiroz et al. (2015). Suppose that we use u to construct a possibly biased estimator $\hat{p}_m(y|\theta, u)$ of the likelihood $p(y|\theta)$. Let $p(u)$ be the sampling density of u, and define the pseudo likelihood

$$p_m(y|\theta) := \int \hat{p}_m(y|\theta, u)p(u)du$$

and the corresponding pseudo marginal likelihood

$$p_m(y) = \int p_m(y|\theta)p(\theta)d\theta.$$

Define the target density on the augmented space (θ, u) as

$$(2.1) \qquad \tilde{\pi}_m(\theta, u) \quad := \quad \frac{\hat{p}_m(y|\theta, u)p(u)p(\theta)}{p_m(y)},$$

where $p(\theta)$ is the prior for θ. It is straightforward to show that $\tilde{\pi}_m(\theta, u)$ is a proper density with marginal

$$\pi_m(\theta) \quad = \quad \int \tilde{\pi}_m(\theta, u)du = \frac{p_m(y|\theta)p(\theta)}{p_m(y)}.$$

We now outline the MCMC scheme that targets (2.1). Suppose that the joint proposal for θ and u is given by

$$q(\theta, u|\theta_c, u_c) = p(u)q(\theta|\theta_c),$$

where c denotes the current state. The M-H acceptance probability becomes

$$\alpha \quad = \quad \min\left(1, \frac{\tilde{\pi}_m(\theta_p, u_p)/q(\theta_p, u_p|\theta_c, u_c)}{\tilde{\pi}_m(\theta_c, u_c)/q(\theta_c, u_c|\theta_p, u_p)}\right)$$

$$(2.2) \qquad = \quad \min\left(1, \frac{\hat{p}_m(y|\theta_p, u_p)p(\theta_p)/q(\theta_p|\theta_c)}{\hat{p}_m(y|\theta_c, u_c)p(\theta_c)/q(\theta_c|\theta_p)}\right).$$

It should be noted that this expression is similar to the standard M-H but with the true likelihood replaced by an estimate. By Andrieu and Roberts (2009), the MCMC iterates converge to draws from the target density, and in particular the iterates of θ converge to draws from $\pi_m(\theta)$. We note that $p(u)$ can also depend on θ, i.e. $p(u|\theta)$.

Quiroz et al. (2015) use estimators of the form

$$(2.3) \qquad \hat{p}_m(y|\theta, u) = \exp\left(\hat{l}_m - \hat{\sigma}_z^2/2\right),$$

where $\hat{l}_m$ is an unbiased estimator of the log-likelihood $l(\theta)$, $z = \hat{l} - l$ is the estimation error and $\hat{\sigma}_z^2$ is an unbiased estimator of $\sigma_z^2 = \mathrm{Var}(z)$. The motivation for this particular class of estimators is that $\exp\left(\hat{l}_m - \sigma_z^2/2\right)$ is unbiased for the likelihood function when $\hat{l}_m$ is normally distributed. Estimators of the form (2.3) redefine the estimation problem from estimating a product (the likelihood) into an estimation problem for a sum (the log-likelihood). This has the advantage that we can use established sampling schemes and efficient estimators of the

population total (a sum) from the survey literature. Moreover, focusing the estimation on the log-likelihood makes it possible to choose m optimally (Pitt et al., 2012; Doucet et al., 2015) and adaptively (Quiroz et al., 2015; Tran et al., 2015). Quiroz et al. (2015) consider log-likelihood estimators in the Hansen-Hurwitz class (Hansen and Hurwitz, 1943)

$$(2.4) \qquad \hat{l}_m = \frac{1}{m} \sum_{i=1}^{m} \zeta_i, \text{ where } \zeta_i = \frac{l_{u_i}}{p_{u_i}},$$

and p_k is the probability of sampling observation k. It is straightforward to show that $\hat{l}_m$ is an unbiased estimator of the log-likelihood l. The estimator $\exp\left(\hat{l}_m - \hat{\sigma}_z^2/2\right)$ is typically slightly biased for the likelihood, but Quiroz et al. (2015) prove that an MCMC sampling scheme in the joint space (θ, u) based on the likelihood estimator (2.3) will sample from a target distribution that is within $O(m^{-1/2})$ percent of the true posterior.

Quiroz et al. (2015) point out that the contribution to the log-likelihood varies substantially across the population (the individual data points) and that selecting the observations used for estimating the log-likelihood by simple random sampling is inefficient. They instead propose Probability Proportional-to-Size (PPS) sampling, where the inclusion probability p_k is proportional to an approximation of the log-likelihood contribution l_k. Since the approximate log-likelihood contributions need to be computed for every data observation in PPS sampling, the approximations are required to be fast compared to evaluating the log-likelihood contribution for subsampling to be effective. One of the main contributions of our article is the use of an alternative estimator, the so called difference estimator, which we describe in the following subsection. The difference estimator uses the approximate l_k in a way that allows us to sample a subset of the data and compute the likelihood estimate using far less computations than calculating the likelihood.

2.2. **The difference estimator.** The PPS scheme uses an approximation of l_k for each individual observation to construct an efficient sampling scheme. In contrast, the difference estimator uses an approximation of l_k directly in the estimator rather than in the sampling

scheme. Let $w_k(\theta)$ denote the approximation of $l_k(\theta)$ and let

$$
\begin{aligned}
l(\theta) &= \sum_{k \in F} w_k(\theta) + \sum_{k \in F} [l_k(\theta) - w_k(\theta)] \\
&= w + d
\end{aligned}
$$

with

$$
w = \sum_{k \in F} w_k(\theta), \quad d = \sum_{k \in F} d_k(\theta), \quad \text{and} \quad d_k(\theta) = l_k(\theta) - w_k(\theta).
$$

Here $w = \sum_{k \in F} w_k(\theta)$ is assumed to be known prior to sampling and the difference estimator is obtained by estimating d. Since $w_k(\theta)$ is an approximation of $l_k(\theta)$, we can expect that $l_k(\theta) - w_k(\theta)$ should have roughly the same size for all $k \in F$. We can therefore use simple random sampling with replacement (SIR) to estimate d:

$$
(2.5) \qquad \hat{d}_m = \frac{1}{m} \sum_{i=1}^{m} \zeta_i, \quad \text{with} \quad \zeta_i = n d_{u_i},
$$

where $p_{u_i} = 1/n$ for $u_i = 1, \dots n$. We can show that

$$
E[\zeta_i] = d, \quad E[\hat{d}_m] = d \quad \text{and} \quad \sigma_\zeta^2 = V[\zeta_i] = n \sum_{k \in F} (d_k - \bar{d}_F)^2, \quad V[\hat{d}_m] = \sigma_\zeta^2 / m,
$$

where $\bar{d}_A$ denotes the mean computed for the set A. Therefore, for the difference estimator

$$
(2.6) \qquad \hat{l}_m = w + \hat{d}_m,
$$

we obtain

$$
(2.7) \qquad E[\hat{l}_m] = l \quad \text{and} \quad V[\hat{l}_m] = \frac{n}{m} \sum_{k=1}^{n} (d_k - \bar{d}_F)^2.
$$

Moreover,

$$
(2.8) \qquad \hat{\sigma}_z^2 = \frac{n^2}{m} s^2, \quad s^2 = \frac{1}{m-1} \sum_{k \in S} (d_k - \bar{d}_S)^2
$$

is an unbiased estimator of $\sigma_z^2 = V[\hat{l}]$ computed on the set S of sampled observations. In Equation (2.5), $\hat{d}_m$ is of the form required in Quiroz et al. (2015) and we have verified their

Assumption 1. It is straightforward to show that Lemma 4 in Quiroz et al. (2015) holds for $\hat{l}_m = w + \hat{d}_m$. Consequently, we can apply their Theorem 1 to ensure that our algorithm samples from a perturbed posterior that is within $O(m^{-1/2})$ percent of the true posterior. We also note that

$$\sqrt{m}(\hat{l}_m - l) \to \mathcal{N}(0, \sigma_\zeta^2)$$

by the standard central limit theorem because the ζ_i's are iid.

2.3. Approximating l_k by Taylor series expansions at local data clusters.

We note from $V[\hat{l}_m]$ in (2.7) that the difference estimator is efficient when the d_k are close to $\bar{d}_F$, i.e. when $w_k \approx l_k$. Any of the methods developed in Quiroz et al. (2015) can be used to construct w_k. The w_k in Quiroz et al. (2015) are relatively cheap to compute but need to be computed for all points in the dataset. An advantage of the difference estimator is that it opens up the possibility of constructing w_k using only a sparse subset of the data observations, which we now describe. The idea is to cluster the data $z_k = (y_k, x_k)$ into N_C clusters, compute the exact log-likelihood contributions at all centroids and use a second order Taylor expansion at the centroid as a local approximation of l_k around each centroid. This allow us to compute $\sum_{k \in F} w_k(\theta)$ by simply scaling up quantities computed at the N_C centroids. Appendix A describes in detail the localization of data clusters. We now consider the computationally efficient local Taylor series approximations at the cluster centroids.

Consider a univariate response y for notional clarity. Define

$$l(z_k; \theta) = \log p(y_k | x_k, \theta) = l_k(\theta)$$

as a function of $z_k = (y_k, x_k)^T \in (p+1) \times 1$ for a given parameter $\theta \in p \times 1$. The change of notation emphasizes that $l(z_k; \theta)$ is a function of z_k, instead of θ. Let C denote the index set of observations within cluster c. For any $k \in C$, a second order Taylor approximation of $l(z_k; \theta)$ around the centroid z^c is

$$w(z_k; \theta) = l(z^c; \theta) + \nabla_z l(z^c; \theta)^T (z_k - z^c) + \frac{1}{2}(z_k - z^c)^T H(z^c; \theta)(z_k - z^c),$$

where $H(z^c; \theta) = \nabla_z^2 l(z^c; \theta)$ is the Hessian evaluated at z^c. Note that once $l(z^c; \theta)$ is computed, it is relatively cheap to evaluate $\nabla_z l(z^c; \theta)$ and $H(z^c; \theta)$ by using the chain rule. Appendix B provides formulas for computing $w = \sum_{k=1}^n w(z_k; \theta)$ at the centroids $\{z^{c_j}\}_{j=1}^{N_C}$, where typically $N_c << n$. Assuming that the density evaluations dominate the computational cost, our method performs $N_C + m$ log-density evaluations compared to n for regular MCMC.

The approximation error is given by the remainder term of the Taylor series. The remainder depends on the clustering algorithm through ϵ, which is the maximum distance between an observation in a cluster and its centroid. The choice of ϵ determines how local the approximation is. It is difficult to provide guidelines on how to choose ϵ (see Appendix A) and therefore also to determine a reasonable level for the error. However, as we demonstrate in Section 2.4, the user only needs to monitor the variance of the difference estimator for an optimal trade-off between computing time and efficiency. The variance is reduced (if needed) by increasing the size of the subsample. We can therefore only focus on the variance of the difference estimator when designing a PMCMC algorithm in our framework.

In Appendix C the approximation is derived for the class of Generalized Linear Models (GLM) (Nelder and Wedderburn, 1972). We emphasize that our method applies much more widely; the only requirement is that $l(z; \theta)$ is twice differentiable with respect to z. Even in models with vector valued θ it is typically straightforward to derive the approximation. We note that categorical variables, either response or covariates, are considered as continuous in the differentiation.

2.4. **Controlling the variance by adapting the sample size.** Although the result in Andrieu and Roberts (2009) is valid regardless of the variance of the estimator, controlling the variance is crucial for the efficiency. In general, an estimator with a lower variability gives a more efficient Markov Chain but is also more expensive to compute. Conversely, an estimator with higher variability gives a less efficient Markov Chain but is faster to compute. The trade-off between computing time and efficiency when using an unbiased estimator of the likelihood is investigated, under different assumptions, in Pitt et al. (2012), Doucet et al.

(2015), and Sherlock et al. (2015). The suggestions of the optimal σ_z^2 ranges from $[1, 3.3]$, with the larger values corresponding to a weaker proposal in the exact likelihood setting.

We follow Quiroz et al. (2015) and choose the sample size adaptively such that $\sigma_z^2 \approx 1$, which is a conservative choice that minimizes the risk of the Markov chain getting stuck. As in Quiroz et al. (2015) we adapt the sample size m at a given iteration so that the variance is never larger than a user specified maximum v_{max}. The adaptation strategy is to increase m whenever $\hat{\sigma}_z^2 = \hat{V}[\hat{l}(\theta_p)] > v_{max}$. A simple guess of m is achieved by Equation (2.8), i.e.

$$(2.9) \qquad m^* \;=\; \frac{1}{v_{max}(m-1)} \sum_{k \in S} (d_k - \bar{d}_S)^2.$$

In practice, we add $m^* - m$ observations (with SIR) and online algorithms can be used for computing the new variance, see e.g. Chan et al. (1983). Algorithm 2 in Appendix D describes the PMCMC algorithm with adaptive sample size.

2.5. **Infrequent updating of the data subset.** Since $\hat{l}_m$ is unbiased for any θ, we can consider updating u less frequently than θ. Infrequently updating u is not a good idea in the algorithm in Quiroz et al. (2015) because the PPS subsample is efficient only for the proposed θ. Infrequent updates of u would therefore lead to the estimator of the log-likelihood having a large variance. In contrast, the w_k in the difference estimator is constructed to be a good approximation for all θ, and $\sigma_z^2(\theta)$ will therefore be small for all proposed θ. We will demonstrate empirically that the strategy of updating u less frequently than θ gives a more efficient PMCMC chain compared to updating u in every iteration.

We note that the PMCMC is still valid because (i) The iterates form a Markov chain on the augmented space (θ, u); (ii) if the MCMC is ergodic, i.e. irreducible, aperiodic and Harris recurrent, then so is the PMCMC. (iii) The expected value of the estimated likelihood $\hat{p}_m(y|\theta, u)$ is $p_m(y|\theta)$ as we now demonstrate.

To achieve a sampling scheme that updates the subsample infrequently we propose to obtain a new subsample with probability ω and keep the current subsample otherwise. We will need the following lemma to derive the PMCMC with infrequent updates of u.

Lemma 1. *Suppose that the random variable U has density $p(u)$ with respect to the Lebesgue measure. Define the conditional probability measure of V given U as*

$$q_{V|U}(dv|u) := \omega p(v)dv + (1-\omega)\delta_u(dv), \quad 0 < \omega < 1,$$

where $\delta_u(dv)$ is 1 if $u \in dv$ and zero otherwise. Let $q_V(dv)$ be the marginal probability measure of V. Then,

    i. *$q_V(dv) = p(v)dv$, i.e. V has the density $p(v)$ with respect to the Lebesgue measure.*

    ii. *Define the conditional mixed measure,*

(2.10)
$$\lambda(dv|u) := dv \mathbb{1}(u \neq v) + \delta_u(dv)\mathbb{1}(u = v).$$

    Then, $q_{V|U}(dv|u)$ has density

$$q_{V|U}(v|u) := \omega p(v)\mathbb{1}(u \neq v) + (1-\omega)\mathbb{1}(u = v)$$

    with respect to $\lambda(dv|u)$.

    iii. *The ratio of conditional densities, each with respect to $\lambda(\cdot|\cdot)$, is*

$$\frac{q_{V|U}(v|u)}{q_{U|V}(u|v)} = \begin{cases} p(v)/p(u), & u \neq v \\ 1, & u = v. \end{cases}$$

Proof. Proof of (i): First, we note that

$$q_V(dv) := \int_u q_{V|U}(dv|u)p(u)du = \omega p(v)dv + (1-\omega)\int_u \delta_u(dv)p(u)du.$$

Let $h(v)$ be a bounded function of v. Then,

$$\begin{aligned} \int_v h(v)q_V(dv) &= \omega \int_v h(v)p(v)dv + (1-\omega)\int_u \int_v h(v)\delta_u(dv)p(u)du \\ &= \omega \int_v h(v)p(v)dv + (1-\omega)\int_u h(u)p(u)du \\ &= \int_v h(v)p(v)dv. \end{aligned}$$

If we take $h(v) = \mathbb{1}_A(v)$, i.e. the indicator function for the set A, then

$$q_V(A) = \int_A p(v)dv,$$

which shows (i).

To show (ii), we write

$$
\begin{aligned}
q_{V|U}(dv|u) &= \omega p(v)dv\mathbb{1}(u \neq v) + (1-\omega)\delta_u(dv)\mathbb{1}(u = v) \\
&= (\omega p(v)\mathbb{1}(u \neq v) + (1-\omega)\mathbb{1}(u = v))\,\lambda(dv|u).
\end{aligned}
$$

Part (iii) follows from (ii). $\qquad\square$

We consider the proposal of u conditional u_c,

$$(2.11) \qquad\qquad q(du|u_c) = \omega p(u) + (1-\omega)\delta_{u_c}(du)$$

with density $q(u|u_c)$ with respect to the measure $\lambda(du|u_c)$. The density of the marginal measure of (2.11) is $p(u)$ by Lemma 1(i), which has the property

$$p_m(y|\theta) = \int \hat{p}_m(y|\theta, u)p(u)du.$$

The M-H acceptance probability, when $u_p \sim q(u|u_c)$, is

$$\alpha = \min\left(1, \frac{\hat{p}_m(y|\theta_c, u_p)p(u_p)p(\theta_p)}{\hat{p}_m(y|\theta_c, u_c)p(u_c)p(\theta_c)} \times \frac{q(u_c|u_p)}{q(u_p|u_c)} \times \frac{q(\theta_c|\theta_p)}{q(\theta_p|\theta_c)}\right).$$

However, from Lemma 1(iii) it follows that we obtain the same expression as in (2.2).

Let PMCMC(ω) denote an algorithm which updates u with probability ω in each MCMC iteration. The algorithm clearly converges to the invariant distribution on the joint space (θ, u) for any $\omega > 0$ as the number of MCMC iterates $N \to \infty$. However, in practice we use a finite N and we need to ensure that ω is sufficiently large to guarantee convergence. This is explored in our application in Section 3 where we show that $\omega = 1$ and values as small as $\omega = 0.01$ result in the same inference. Algorithm 3 in Appendix D implements PMCMC(ω).

2.6. **Connection to Previous Research.** Our method is closely related to the FireFly Monte Carlo (FFMC) approach in Maclaurin and Adams (2014). FFMC augments each observation $\{y_k\}_{k=1}^n$ with an auxiliary variable $u_k = \{0, 1\}$. The distribution of u_k is determined so that the augmented posterior $p(\theta, u|y)$ only requires evaluating the likelihood contributions for the observations for which $u_k = 1$. The augmented likelihood has the form

$$(2.12) \qquad p(y|\theta, u) \;=\; \prod_{k=1}^{n} B_k(\theta) \prod_{\{k:u_k=1\}} \frac{L_k(\theta) - B_k(\theta)}{B_k(\theta)},$$

where $L_k(\theta) = p(y_k|\theta)$ and $B_k(\theta)$ is a positive lower bound of $L_k(\theta)$, i.e. $0 < B_k(\theta) \le L_k(\theta)$. The lower bound $B_k(\theta)$ is chosen so that $\prod_{k=1}^n B_k(\theta)$ can be computed using a sufficient statistic. It should be noted that it is a very difficult task to find a lower bound for most models. The posterior is sampled using Metropolis-within-Gibbs, updating from the full conditionals $p(\theta|u, y)$ and $p(u|\theta, y)$.

Note the analogy (in log-scale) of (2.12) to the difference estimator in (2.6). The first term, which operates over the full data set, corresponds to w. The second term is only evaluated for the subsample, which corresponds to $\hat{d}_m$. Our approach is closely related to a Pseudo-marginal version of FFMC, where θ and u are updated jointly. Indeed, Maclaurin and Adams (2014) conclude that FFMC with a joint update of θ and u, where each u_k is updated as a Bernoulli variable with probability 0.5, is a Pseudo-marginal MCMC. The likelihood in Equation (2.12) using this update is an unbiased estimate of the true likelihood. This strategy would, on average, only reduce the likelihood evaluations by a factor of 2, compared to our approach that reduces by a factor of 10 for our application in Section 3. Moreover, the efficiency is likely to be poor as it is a well-known fact that a random subsample gives a higher variance of an estimator (Särndal et al., 2003), and so there is a risk that the PMCMC chain may get stuck (Quiroz et al., 2015). We also note that our method does not require a lower bound for the data density.

3. APPLICATION

3.1. **Data and model.** Our data set contains annual observations for Swedish firms for the time period 1991-2008. We model the probability of bankruptcy conditional on a set of covariates. The firm-specific financial variables are all scaled with respect to total assets. These variables are earnings before interest and taxes, total liabilities, cash and liquid assets, and tangible assets. We also use the logarithm of deflated total sales and the logarithm of firm age in years as control variables. Finally, we include the macroeconomic variables GDP-growth rate (yearly) and the interest rate set by the Swedish central bank. We have in total $534,717$ firms and $n = 4,748,089$ firm-year observations. The data set contains $41,566$ cases of bankruptcy.

We consider the logistic regression model

$$p(y_k|x_k,\beta) = \left(\frac{1}{1+\exp(x_k^T\beta)}\right)^{y_k}\left(\frac{1}{1+\exp(-x_k^T\beta)}\right)^{1-y_k},$$

where x_k includes the variables above plus an intercept term. We set $p(\beta) \sim N(0,10I)$ for simplicity.

3.2. **Performance evaluation.** The Inefficiency Factor (IF), or the integrated autocorrelation time, is defined as

$$(3.1) \qquad IF = 1 + 2\sum_{l=1}^{\infty}\rho_l,$$

where ρ_l is the autocorrelation at the lth lag of the chain. We estimate IF using the CODA package in R (Plummer et al., 2006). IF measures the number of draws required to obtain the equivalent of a single independent draw. Because PMCMC uses an estimated likelihood it has in general a higher IF than MCMC. The Relative Inefficiency Factor (RIF) is defined as

$$RIF = \frac{IF^{PMCMC}}{IF^{MCMC}}.$$

RIF increases with the efficiency of the proposal for θ in the exact likelihood setting (Doucet et al., 2015). Quiroz et al. (2015) studies this in the context of subsampling data for estimating the likelihood and find that RIF can be very large when a very efficient proposal for θ is used. Section 3.4 shows that updating u only at randomly chosen iterations as described in Section 2.5 avoids this problem.

We evaluate the performance of the algorithm using the Effective Draws (ED)

$$(3.2) \qquad ED \; := \; \frac{N}{IF \times DE},$$

where DE is the number of log-density evaluations per iteration and N is the number of MCMC iterates. For PMCMC this includes the subsample (after possibly adapting, in which case we compute the average DE) and the number of clusters (N_c), and for MCMC we have $DE = n$. This measure is independent of the implementation and is reasonable under the assumption that the computational bottleneck of MCMC is the log-density evaluations. The Relative Effective Draws (RED) is defined as

$$RED \; := \; \frac{ED^{PMCMC}}{ED^{MCMC}}.$$

3.3. **Implementation details.** Since the bankruptcy observations ($y_k = 1$) are sparse in this application, we only estimate the contribution from the $y_k = 0$ observations, i.e. the second term in

$$l(\theta) \;\; = \sum_{\{k; y_k=1\}} l_k(\theta) + \sum_{\{k; y_k=0\}} l_k(\theta),$$

and the first term is always evaluated (and included in DE). Thus we only cluster the $4,664,957$ remaining observations for which $y = 0$ and this results in $N_C = 173,135$ clusters which corresponds to 3.71% (of the remaining observations). Together with the $41,566$ default observations, PMCMC starts with $4.6\% \cdot n$ evaluations prior to evaluating the subsample.

Two different proposals for $\theta = \beta$ are considered; the Random walk M-H (RWM) and the Independence M-H (IMH). The RWM uses the Hessian $H(\theta^*)$ of $p(\theta|y)$ evaluated at the posterior mode θ^* obtained from numerical optimization and sets $q(\theta|\theta_c) = N(\theta_c, c_\lambda H^{-1}(\theta^*))$. The IMH uses $q(\theta|\theta_c) = q(\theta) = t_\nu(\theta^*, H^{-1}(\theta^*))$, where t_ν is the multivariate Student-t distribution with $\nu = 10$ degrees of freedom. For all algorithms we sample $55,000$ draws and discard the first $5,000$ draws as burn-in.

We explore the mixing of the chain for different values of ω, the probability that u is updated in a given iteration. During the burn-in phase of the algorithm we start with $\omega = 1$ and subsequently set some $0 < \omega < 1$ after the burn-in period. To determine if a Markov chain with an $\omega < 1$ mixes well, we compare its mean with that of a chain generated with ω set to 1. This is a standard statistical test which takes the sampling variability of the chains into account by computing standard errors assuming both chains are weakly stationary. The test is conducted after the burn-in period. Figure 1 shows, for a range of ω values, the confidence intervals (95%) to conduct the test for difference in means for our application. For none of the ω values can we reject that the chains have the same means, so we can safely use small values of ω in our application.

3.4. **Results.** Figure 2 and 3 show the loss in efficiency and the number of effective draws for PMCMC(1) with a RWM and IMH proposal, respectively. The figure also illustrates that adapting m does not make much difference in this particular application because the approximations are accurate throughout the region of the parameter space explored by the proposal. The relative inefficiency factor in Figure 2, where PMCMC is implemented with the inefficient RWM proposal, is small (Doucet et al., 2015). In contrast, for the very efficient IMH proposal, RIF increases significantly as shown by Figure 3. In turn this adversely affects the relative effective draws

Figure 4 illustrates that by implementing PMCMC with a random update of u, the sizeable increase in RIF for the IMH sampler can be prevented. We note that the same applies for the RWM proposal, but is much less pronounced because the loss in efficiency is already small for $\omega = 1$ (see Figure 2). Table 1 gives additional results of the algorithms. Furthermore,
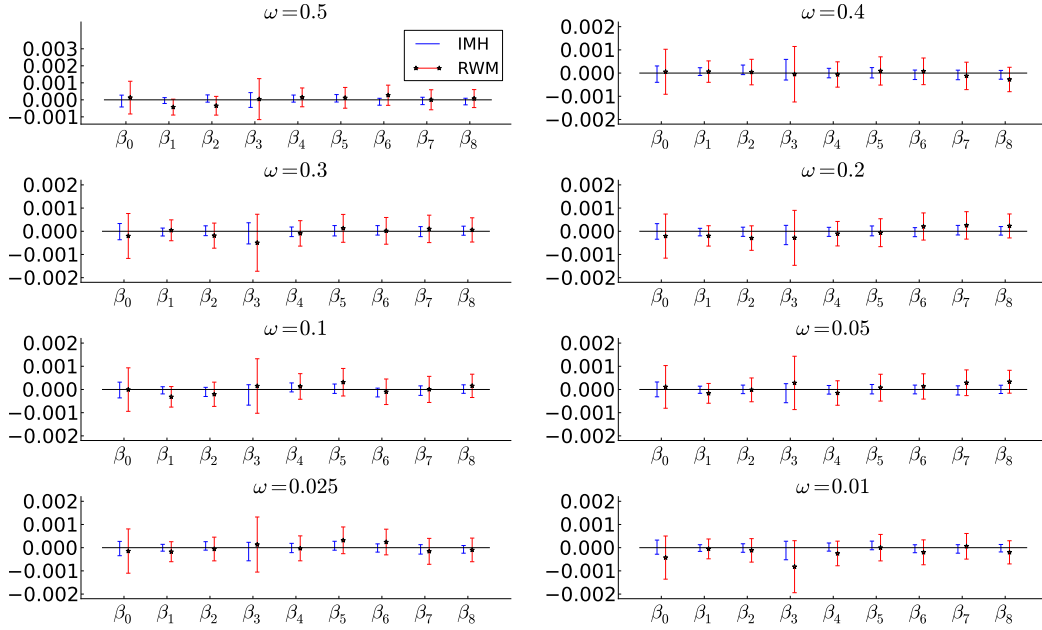
FIGURE 1. Confidence intervals (95%) for difference in posterior means obtained using different algorithms. The mean of the first chain is obtained using PMCMC(1) (same for all figures) and the second mean is obtained using PMCMC(ω) (corresponding to each subplot). These confidence intervals are constructed for each parameter in the application in Section 3, using two proposals; Independent Metropolis Hastings (IMH, blue vertical line) Random Walk Metropolis (RWM, red vertical line with star). The black horizontal line marks the value 0 of the difference in means under H_0.
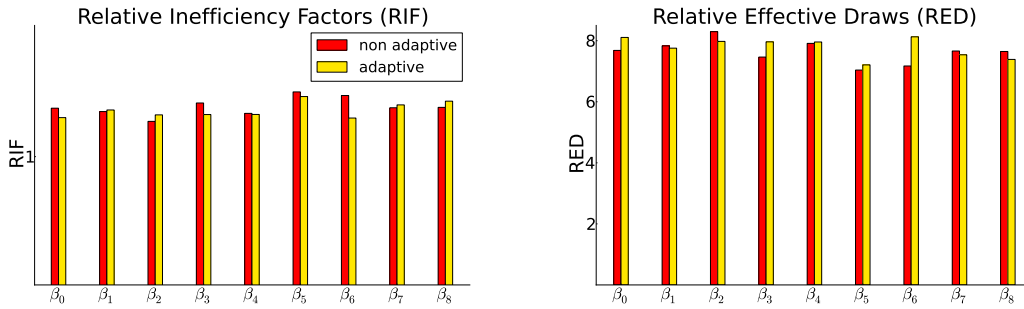
FIGURE 2. The left panel shows the Relative Inefficiency Factors (RIF) for PMCMC(1) adaptive (yellow bar) and PMCMC(1) non adaptive (red bar) for each parameter obtained with a random walk Metropolis proposal. The right panel shows the corresponding Relative Effective Draws (RED).

Figure 5 shows the improvement in relative effective draws as a consequence of the reduced
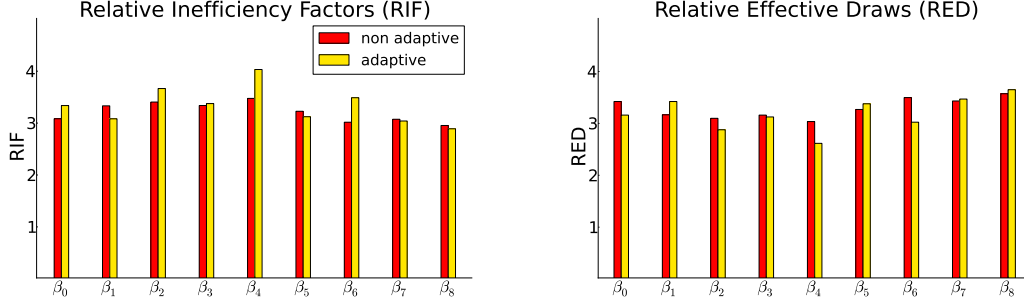
FIGURE 3. The left panel shows the Relative Inefficiency Factors (RIF) for PMCMC(1) adaptive (yellow bar) and PMCMC(1) non adaptive (red bar) for each parameter obtained with an independent Metropolis-Hastings proposal. The right panel shows the corresponding Relative Effective Draws (RED).
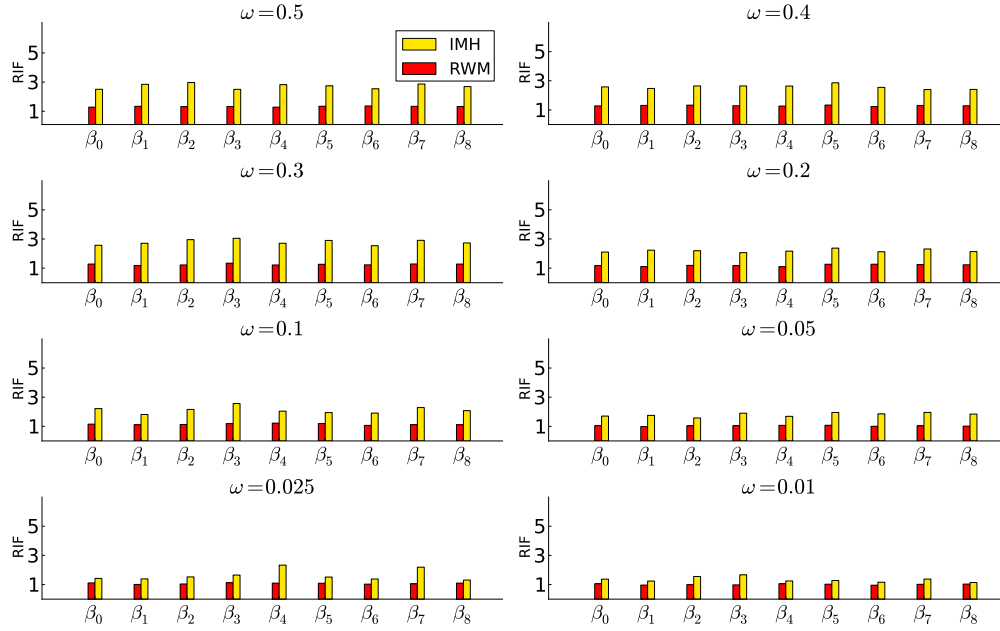
FIGURE 4. Relative Inefficiency Factors (RIF) for different PMCMC(ω). The figure shows the RIF for all parameters using two different proposals; Independence Metropolis Hastings (IMH, yellow bar) and Random Walk Metropolis (RWM, red bar).

inefficiency. We conclude that randomly updating u is very beneficial for the efficiency of the algorithm.

Figure 6 shows that the marginal posterior obtained with some different values of ω are very close to the true posterior obtained with MCMC. This accuracy is further confirmed in

TABLE 1. *Additional results of the algorithms.* The table shows the mean acceptance probability $\Pr(Acc)$ and also the mean of the estimated standard deviation $\bar{\sigma}_z(\theta_p)$ at the proposed θ values. The quantities are shown for MCMC and $\text{PMCMC}(\omega)$ for different ω in the example and for the proposals Random Walk Metropolis (RWM) and Independence Metropolis Hastings (IMH). All PMCMC algorithms use 9.48% of the data in total (of which 5% is the subsample).

Algorithm	$\Pr(Acc)$		$\bar{\sigma}_z(\theta_p)$	
	RWM	IMH	RWM	IMH
MCMC	0.251	0.753	n/a	n/a
$\text{PMCMC}(\omega)$				
$\omega = 1$ adaptive	0.185	0.457	0.946	0.940
$\omega = 1$ non adaptive	0.184	0.453	0.947	0.940
$\omega = 0.5$	0.192	0.532	0.946	0.940
$\omega = 0.4$	0.196	0.554	0.947	0.940
$\omega = 0.3$	0.205	0.570	0.946	0.940
$\omega = 0.2$	0.212	0.619	0.946	0.940
$\omega = 0.1$	0.218	0.657	0.947	0.941
$\omega = 0.05$	0.234	0.690	0.947	0.940
$\omega = 0.025$	0.241	0.709	0.947	0.940
$\omega = 0.01$	0.245	0.720	0.944	0.939

Figure 7, which shows the upper bound of the fractional error in the likelihood approximation as derived in Quiroz et al. (2015).

4. CONCLUSIONS AND FUTURE RESEARCH

We propose an algorithm for speeding up MCMC for large data problems. The method uses a small subset of the data together with the difference estimator to estimate the log-likelihood efficiently. The estimator uses an approximation of the (log) data density via its Taylor series approximation around a centroid in a local data cluster. We propose an algorithm to obtain the local data clusters and furthermore we derive the approximation for a large class of models. Our estimator uses the sum of the individual approximations to obtain an accurate estimate of the full data log-likelihood. This operation has computational complexity $O(n)$, where n is the number of data observations. To overcome this issue we
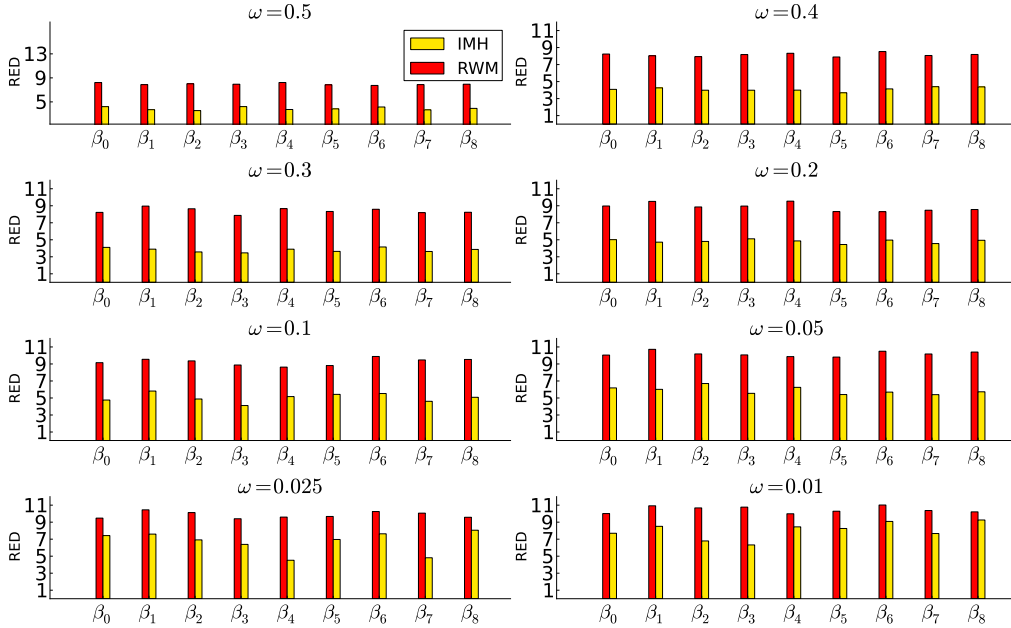
FIGURE 5. Relative Effective Draws (RIF) for different PMCMC(ω). The figure shows the RED for all parameters using two different proposals; Independence Metropolis Hastings (IMH, yellow bar) and Random Walk Metropolis (RWM, red bar).

derive expressions for the sum which only requires evaluating quantities at the centroids, thereby reducing the complexity to $O(N_C)$, where $N_C << n$ is the number of local data clusters.

The estimated (biased) likelihood is used within the M-H algorithm to sample from an approximate posterior distribution. We demonstrate that the proposed estimator belongs to the class of estimators in Quiroz et al. (2015). Therefore our method samples from a posterior that is within $O(m^{-1/2})$ percent of the true posterior, where m is the sample size. Moreover, we get explicit upper bounds for the approximation error in the likelihood and illustrate that these are small in our application.

We propose to update the subsample used for estimation randomly in each iteration of the algorithm. We demonstrate empirically that this is an effective strategy because it reduces the inevitable efficiency loss from using an estimated likelihood in the M-H ratio, especially in cases where the proposal is very efficient.
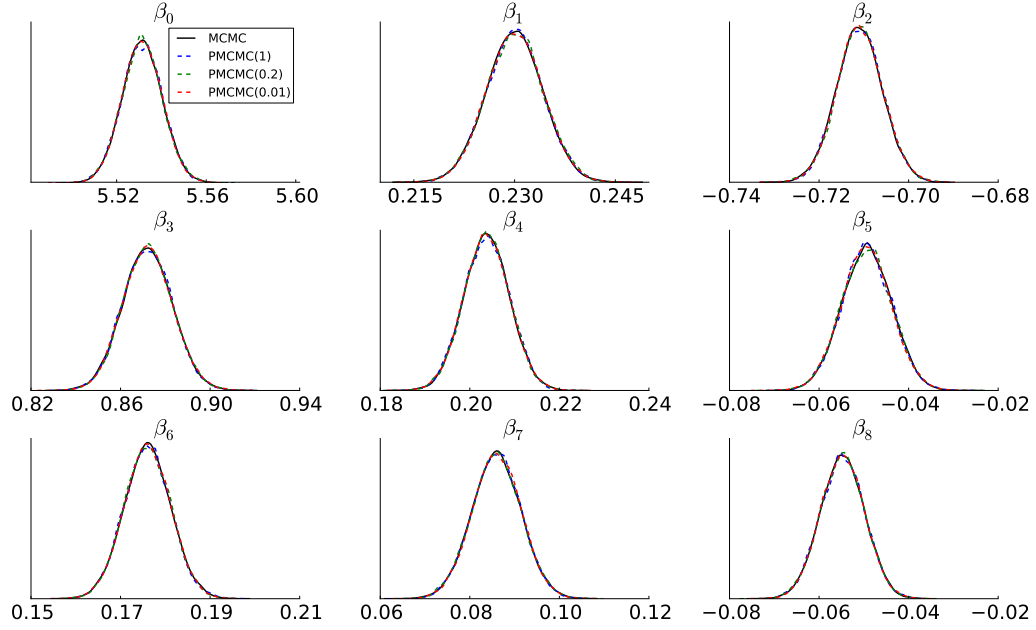
FIGURE 6. Kernel density estimations of marginal posteriors. The figure shows the marginal posteriors obtained using PMCMC(ω) with $\omega = 1, 0.2, 0.01$ (dashed blue, green and red, respectively) and regular MCMC (solid black line).
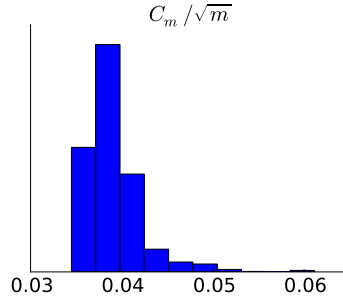
FIGURE 7. The bound for part (i) of Theorem 1 in Quiroz et al. (2015) for our application. The figure shows the upper bound for the fractional error in the likelihood approximation computed over 1000 draws from the posterior. The subsample size m is chosen so that $\sigma_z \approx 1$ resulting in $m = 235, 326$ (5% of the full sample size).

The method is applied to a dataset with nearly 5 million observations. We consider a logistic model for predicting firm bankruptcy conditional on standard covariates used in the literature. Using a measure that balances the number of density evaluations and the

efficiency of the resulting chains, we document a very favorable outcome for our algorithm compared to regular MCMC.

Future research concerns improved methods to obtain the local data clusters in the presence of a huge number of covariates. This is especially important when many of the covariates are categorical.

References

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

Banterle, M., Grazian, C., and Robert, C. P. (2014). Accelerating Metropolis-Hastings algorithms: Delayed acceptance with prefetching. *arXiv preprint arXiv:1406.2660*.

Bardenet, R., Doucet, A., and Holmes, C. (2014). Towards scaling up Markov chain Monte Carlo: an adaptive subsampling approach. In *Proceedings of The 31st International Conference on Machine Learning*, pages 405–413.

Chan, T. F., Golub, G. H., and LeVeque, R. J. (1983). Algorithms for computing the sample variance: Analysis and recommendations. *The American Statistician*, 37(3):242–247.

Dean, J. and Ghemawat, S. (2008). MapReduce: simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113.

Doucet, A., Pitt, M., Deligiannidis, G., and Kohn, R. (2015). Efficient implementation of Markov Chain Monte Carlo when using an unbiased likelihood estimator. *To appear in Biometrika*.

Gelfand, A. E. and Smith, A. F. (1990). Sampling-based approaches to calculating marginal densities. *Journal of the American statistical association*, 85(410):398–409.

Hansen, M. H. and Hurwitz, W. N. (1943). On the theory of sampling from finite populations. *The Annals of Mathematical Statistics*, 14(4):333–362.

Korattikara, A., Chen, Y., and Welling, M. (2013). Austerity in MCMC land: Cutting the Metropolis- Hastings budget. *arXiv preprint arXiv:1304.5299*.

Maclaurin, D. and Adams, R. P. (2014). Firefly Monte Carlo: Exact MCMC with subsets of data. *arXiv preprint arXiv:1403.5693*.

Minsker, S., Srivastava, S., Lin, L., and Dunson, D. (2014). Scalable and robust Bayesian inference via the median posterior. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1656–1664.

Neiswanger, W., Wang, C., and Xing, E. (2013). Asymptotically exact, embarrassingly parallel MCMC. *arXiv preprint arXiv:1311.4780*.

Nelder, J. A. and Wedderburn, R. W. M. (1972). Generalized linear models. *Journal of the Royal Statistical Society. Series A (General)*, 135(3):pp. 370–384.

Payne, R. D. and Mallick, B. K. (2014). Bayesian big data classification: A review with complements. *arXiv preprint arXiv:1411.5653*.

Pitt, M. K., Silva, R. d. 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.

Plummer, M., Best, N., Cowles, K., and Vines, K. (2006). Coda: Convergence diagnosis and output analysis for mcmc. *R News*, 6(1):7–11.

Quiroz, M., Villani, M., and Kohn, R. (2015). Speeding up mcmc by efficient data subsampling. *arXiv preprint arXiv:1404.4178v2*.

Särndal, C.-E., Swensson, B., and Wretman, J. (2003). *Model assisted survey sampling*. Springer.

Scott, S. L., Blocker, A. W., Bonassi, F. V., Chipman, H., George, E., and McCulloch, R. (2013). Bayes and big data: the consensus Monte Carlo algorithm. In *EFaBBayes 250" conference*, volume 16.

Sherlock, C., Thiery, A. H., Roberts, G. O., and Rosenthal, J. S. (2015). On the efficiency of pseudo-marginal random walk Metropolis algorithms. *To appear in Annals of Statistics*.

Tran, M.-N., Scharth, M., Pitt, M. K., and Kohn, R. (2015). Importance sampling squared for bayesian inference in latent variable models. *arXiv preprint arXiv:1309.3339v3*.

Wang, X. and Dunson, D. B. (2013). Parallel MCMC via Weierstrass sampler. *arXiv preprint arXiv:1312.4605.*

Appendix A. Local Data Clusters

Let z^c and n_c denote the centroid and the number of observations in cluster c, respectively. Note that $\sum_{c=1}^{N_C} n_c = n$ and typically $N_C << n$. Algorithm 1 provides an easily implemented clustering algorithm. The maximum distance ϵ between an observation and the cluster is a user defined input. The clustering is a one time cost that can be stored for future use, and is easily sequentially updated as new data arrives.

For models with a categorical response, we cluster separately for each category (i.e. $z_k = x_k$). In the presence of many categories. we suggest transforming y_k to a (finite) interval of real values and, once the centroids are obtained, applying the inverse transformation that maps to the closest category. E.g. in the Poisson regression model $y_k \in \{0, 1, 2, \ldots, \}$ and we define the one-to-one mapping $\phi : \{y_{min}, \ldots, y_{max}\} \mapsto [K_1(\epsilon), K_2(\epsilon)]$, where K_1 and K_2 control the scale. The algorithm clusters the data $\tilde{z} = (\tilde{y}_k, x_k)$ with $\tilde{y}_k = \phi(y_k)$, and returns centroids $\{\tilde{z}^{c_j} = (\tilde{y}^{c_j}, x^{c_j})\}_{j=1}^{N_c}$. For each centroid, we choose the point in $a^{c_j} \in [K_1(\epsilon), K_2(\epsilon)]$ which is closest to $\tilde{y}^{c_j}$. The final centroids are given by $z_j^c = (\phi^{-1}(a^{c_j}), x^{c_j})$.

Note that if an observation does not have any neighbors within an ϵ neighbourhood, it forms a singleton cluster and is the centroid of that cluster. When the dimension of the data space increases the number of such singleton clusters is likely to increase. By increasing the radius of the ϵ-ball this can be prevented up to a certain point where the local approximation becomes poor within the cluster. However, in practice the data are seldom uniformly distributed on a hyper-cube; in high dimensions data tend to cluster on a subspace. In this setting, an ϵ ball will always reduce the size of the data set, but it is difficult to provide guidelines on how to choose ϵ when the dimension increases, as it is depends on the geometry of the data. In practice, we run the algorithm for a given ϵ and sequentially monitor the fraction N_C/n. It is usually rapidly discovered after a few iterations if this fraction is too large and then the algorithm is restarted with a larger value of ϵ. In problems

---

Algorithm 1 Clustering data points within an ϵ-radius ball

1: **procedure** CLUSTERDATA(y, x, ϵ)
2:     $z_k \leftarrow (y_k, x_k)^T$
3:     $z \leftarrow (z_1^T, \ldots, z_n^T)^T.$ $\triangleright$ Standardized data.
4:     $Z \leftarrow (0, \ldots, 0)^T$ $\triangleright$ 0 - observation is not clustered.
5:     $(i, j) \leftarrow (0, 0)$ $\triangleright$ Initialize counters.
6:     **while** $\sum Z \neq n$ **do**
7:         **if** $Z_i = 0$ **then** $\triangleright$ If not clustered yet.
8:             $C_j \leftarrow \{k; \|z_i - z_k\| \leq \epsilon\}$ $\triangleright$ Form cluster within an ϵ-ball.
9:             $N_j \leftarrow |C_j|$
10:            $z^{c_j} \leftarrow \frac{1}{N_j} \sum_{k \in C_j} z_k$ $\triangleright$ Create centroid with N_j observations.
11:            $Z_{C_j} \leftarrow 1$ $\triangleright$ Mark clustered observations.
12:            $j \leftarrow j + 1$
13:        **end if**
14:        $i \leftarrow i + 1$
15:    **end while**
16:    $N_C \leftarrow j$
17:    **return** $\{z^{c_j}\}_{j=1}^{N_C}, \{C_j\}_{j=1}^{N_C}$
18: **end procedure**

---

where the covariate space is huge we suggest reducing the dimension of the covariate space using principal components.

APPENDIX B. COMPACT MATRIX COMPUTATIONS

Let z^{c_j} denote the centroid in cluster c_j, $j = 1, \ldots, N_c$. Let C_j denote the index set of observations within c_j with $N_j = |C_j|$. The second order Taylor approximation $l(z_k; \theta)$ in cluster j, for $k \in C_j$, is

$$w(z_k; \theta) = l(z^{c_j}; \theta) + \nabla_z l(z^{c_j}; \theta)^T (z_k - z^{c_j}) + \frac{1}{2}(z_k - z^{c_j})^T H(z^{c_j}; \theta)(z_k - z^{c_j}).$$

We now derive a compact expression for

$$w = \sum_{k=1}^{n} w(z_k; \theta) = \sum_{k=1}^{n} l(z^{c_j}; \theta) + \sum_{k=1}^{n} \nabla_z l(z^{c_j}; \theta)^T (z_k - z^{c_j}) + \frac{1}{2} \sum_{k=1}^{n} (z_k - z^{c_j})^T H(z^{c_j}; \theta)(z_k - z^{c_j}).$$

Note that, within a centroid j, $l(z^{c_j}; \theta), \nabla_z l(z^{c_j}; \theta)$ and $H(z^{c_j}; \theta)$ are constant. Therefore the first term is

$$\sum_{k=1}^{n} l(z^{c_j}; \theta) = \sum_{k \in C_1} l(z^{c_1}; \theta) + \cdots + \sum_{k \in C_{N_c}} l(z^{c_{N_c}}; \theta) = l(z^{c_1}; \theta) \sum_{k \in C_1} 1 + \cdots + l(z^{c_{N_c}}; \theta) \sum_{k \in C_{N_c}} 1$$

$$= l(z^{c_1}; \theta) N_1 + \cdots + l(z^{c_{N_c}}; \theta) N_{N_c}.$$

For the middle term, we have

$$\sum_{k=1}^{n} \nabla_z l(z^{c_j}; \theta)^T (z_k - z^{c_j}) = \sum_{k \in C_1} \nabla_z l(z^{c_1}; \theta)^T (z_k - z^{c_1}) + \cdots + \sum_{k \in C_{N_c}} \nabla_z l(z^{c_{N_c}}; \theta)^T (z_k - z^{c_{N_c}})$$

$$= \nabla_z l(z^{c_1}; \theta)^T \sum_{k \in C_1} (z_k - z^{c_1}) + \cdots + \nabla_z l(z^{c_{N_c}}; \theta)^T \sum_{k \in C_{N_c}} (z_k - z^{c_{N_c}}),$$

where $\sum_{k \in C_j} (z_k - z^{c_j}) \in (p+1 \times 1)$ is obtained as a the vector sum of the indices in C_j for the jth centroid. It is independent of θ so it only needs to be computed once before the MCMC.

For the last term, $\sum_{k=1}^{n} (z_k - z^{c_j})^T H(z^{c_j}; \theta)(z_k - z^{c_j})$, by the definition of the quadratic form

$$b_k^T H b_k = \sum_{i,j} H_{ij} b_{ki} b_{kj},$$

with $b_k = (z_k - z^{c_j})^T \in p \times 1$ and $H = H(z^{c_j}; \theta)$ we obtain

$$\sum_{k=1}^{n} b_k{}^T H b_k = \sum_{k=1}^{n} \sum_{i,j} H_{ij} b_{ki} b_{kj}$$

$$= \sum_{i,j} \sum_{k=1}^{n} H_{ij} b_{ki} b_{kj}$$

$$= \sum_{i,j} \left(H_{ij}^{c_1} \sum_{k \in C_1} b_{ik} b_{jk} + \cdots + H_{ij}^{c_{N_c}} \sum_{k \in C_{N_c}} b_{ik} b_{jk} \right).$$

Let B^j be a $p+1 \times p+1$ matrix with elements $\{\sum_{k \in C_j} b_{ki} b_{kj}\}_{ij}$. Then

$$\sum_{k=1}^{n} b_k{}^T H b_k = \sum vec \left(\sum_{j=1}^{N_c} H^{c_j} \circ B^j \right),$$

where $\circ$ denotes the Hadamard product (element wise multiplication) and the sum without indices is over all elements after vectorization. B^j does not depend on θ so we can compute it before the MCMC.

We assume that the dominating cost of the MCMC is the density evaluations. In data sets with a reasonable amount of covariates, the term $\sum_{k=1}^{n} b_k{}^T H b_k$ might be costly as it involves

$N_c \times (p+1)^2$ summations, which can be reduced (H and B are symmetric) to $N_C \times \frac{(p+1)(p+2)}{2}$. In models where the density is log-concave (or convex) we have found that evaluating the second order term in the Taylor approximation for a fixed θ, e.g. the posterior mode, provides a good approximation. Finally, we note that once $l(z^{c_j}; \theta)$ is computed, it is relatively cheap to evaluate the gradient and the Hessian.

Appendix C. Approximation for The GLM Class

The Generalized Linear Model (Nelder and Wedderburn, 1972) is given by

$$
\begin{aligned}
p(y|x, \theta) &\sim h(y)g(\theta)\exp\left(b(\theta)T(y)\right) \\
E[y|x] &= \theta \\
k(\theta) &= x^T\beta.
\end{aligned}
$$

The log-density as a function of data $z = (y, x)^T \in p+1 \times 1$ is

$$
\begin{aligned}
l(z; \theta) &= \log(h(y)) + \log(g(\theta)) + b(\theta)T(y) \\
\theta &= k^{-1}(x^T\beta).
\end{aligned}
$$

To save space, define

$$
\begin{aligned}
k^{-1'} &= \left.\frac{d}{da}k^{-1}(a)\right|_{a=x^T\beta} \\
k^{-1''} &= \left.\frac{d^2}{da^2}k^{-1}(a)\right|_{a=x^T\beta}
\end{aligned}
$$

The gradient $\nabla_z l(z; \theta)$ is the $p+1 \times 1$ vector

$$
\begin{bmatrix} \frac{\partial f}{\partial y} \\ \frac{\partial f}{\partial x} \end{bmatrix} = \begin{bmatrix} \frac{h'(y)}{h(y)} + b(\theta)T'(y) \\ \left(\frac{g'(\theta)}{g(\theta)}k^{-1'} + b'(\theta)T(y)\right)\beta \end{bmatrix}
$$

evaluated at $\theta = k^{-1}(x^T\beta)$, $\beta \in p \times 1$. The hessian $\nabla_z^2 p(z;\theta)$ is the $p+1 \times p+1$ matrix with elements

$$\begin{bmatrix} \frac{\partial^2 l}{\partial y^2} & \frac{\partial^2 l}{\partial y \partial x^T} \\ \frac{\partial^2 l}{\partial y \partial x} & \frac{\partial^2 l}{\partial x \partial x^T} \end{bmatrix}$$

where

$$\frac{\partial^2 l}{\partial y^2} = \frac{1}{h(y)}\left(h''(y) - \frac{h'(y)}{h(y)}\right) + b(\theta)T''(y)$$

$$\frac{\partial^2 l}{\partial y \partial x} = \left(b'(\theta)k^{-1'}T'(y)\right)\beta$$

$$\frac{\partial^2 l}{\partial x \partial x^T} = \left(\left(k^{-1'}\right)^2 \frac{1}{g(\theta)}\left(g''(\theta) - \frac{g'(\theta)}{g(\theta)}\right) + \frac{g'(\theta)}{g(\theta)}k^{-1''} + b''(\theta)k^{-1'}T(y)\right)\beta\beta^T.$$

APPENDIX D. THE PMCMC ALGORITHM

For clarity, we present the adaptive sample size (Algorithm 2) and the infrequent updates (Algorithm 3) separately. It is clear how to implement both features in the same algorithm. However, in this case we note that since the proposal for u depends on the current value u_c, which in turn might have been adapted in the previous iteration, the sample size might be unnecessarily large. This sample size is then kept until the update of u with m observations occurs. If ω is small this might be inefficient as a possibly too large sample size is used for many iterations.

We recommend using adaptive sample size when the approximation is poor for some parts of the parameter space explored by the proposal. It is then crucial to reduce the variance (by increasing the sample size). If the approximation is fairly accurate for large regions of the parameter space, it is usually sufficient with only randomly updating the proposed subsample. As documented in Section 3.4 this increases the efficiency dramatically in the case when an efficient proposal for θ is used.

Algorithm 2 PMCMC(1) with adaptive sample size m such that $\hat{\sigma}_z^2 < v_{max}$, where v_{max} is the maximum variance tolerated in the log-likelihood estimate.

1: **procedure** PMCMC	▷ Pr(update u) = 1.
2: $\quad (\theta_c, u_c) \leftarrow (\theta_0, \text{RandomSample}(F))$	▷ Initialize subsample of size m with SIR.
3: $\quad (\text{LogL}_c, \hat{\sigma}_z^2(\theta_c)) \leftarrow \text{LogLikelihood}(\theta_c, u_c)$	
4: $\quad \text{LogL}_c \leftarrow \text{LogL}_c - \hat{\sigma}_z^2(\theta_c)/2$	▷ "Bias-correction".
5: $\quad$ **for** $i \leftarrow 1, \dots, N$ **do**	
6: $\qquad u_p \sim p(u)$	▷ $p(u)$ SIR with m observations.
7: $\qquad \theta_p \sim q(\theta \| \theta_c)$	▷ Propose parameter.
8: $\qquad (\text{LogL}_p, \hat{\sigma}_z^2(\theta_p)) \leftarrow \text{LogLikelihood}(\theta_p, u_p)$	
9: $\qquad$ **while** $\hat{\sigma}_z^2(\theta_p) > v_{max}$ **do**	
10: $\qquad\quad u^* \sim \text{RandomSample}(F)$	▷ Draw a subsample of size $m^* - m$.
11: $\qquad\quad u_p \leftarrow \text{Append}(S_{u^*}, S_{u_p})$	▷ Augment the subsample.
12: $\qquad\quad m \leftarrow |S_{u_p}|$	
13: $\qquad\quad (\text{LogL}_p, \hat{\sigma}_z^2(\theta_p)) \leftarrow \text{LogLikelihood}(\theta_p, u_p)$	
14: $\qquad$ **end while**	
15: $\qquad \text{LogL}_p \leftarrow \text{LogL}_p - \hat{\sigma}_z^2(\theta_p)/2$	▷ "Bias-correction".
16: $\qquad \alpha \leftarrow 1 \wedge \exp\left(\text{LogL}_p + \log p(\theta_p) - \text{LogL}_c + \log p(\theta_c)\right) \times \frac{q(\theta_c\|\theta_p)}{q(\theta_p\|\theta_c)}$	
17: $\qquad v \sim \text{Uniform}(0,1)$	
18: $\qquad$ **if** $v \leq \alpha$ **then**	▷ Draw is accepted.
19: $\qquad\quad \text{LogL}_c \leftarrow \text{LogL}_p$	
20: $\qquad\quad (\theta_c, u_c) \leftarrow (\theta_p, u_p)$	
21: $\qquad\quad \theta_i \leftarrow \theta_p$	
22: $\qquad$ **else**	
23: $\qquad\quad \theta_i = \theta_c$	
24: $\qquad$ **end if**	
25: $\qquad$ **end for**	
26: $\qquad$ **return** $\{\theta_i\}_{i=1}^N$	
27: **end procedure**	
28:	
29: **procedure** LogLikelihood(θ, u)	▷ Unbiased estimation of log-likelihood.
30: $\quad \hat{l}_m \leftarrow w + \hat{d}_m$	▷ The difference estimator.
31: $\quad \hat{\sigma}_z^2 \leftarrow \frac{n^2}{m(m-1)} \sum_{k \in S} \left(d_k - \bar{d}_S\right)^2$	
32: $\quad$ **return** $\hat{l}_m, \hat{\sigma}_z^2$	
33: **end procedure**	

Algorithm 3 PMCMC(ω) with random update of u.

1: **procedure** PMCMC(ω) $\triangleright$ Pr(update u) = ω.
2:    $(\theta_c, u_c) \leftarrow (\theta_0, \text{RandomSample}(F))$ $\triangleright$ Initialize subsample of size m with SIR.
3:    $(\text{LogL}_c, \hat{\sigma}_z^2(\theta_c)) \leftarrow \text{LOGLIKELIHOOD}(\theta_c, u_c)$ $\triangleright$ LOGLIKELIHOOD() as in Algorithm 2.
4:    $\text{LogL}_c \leftarrow \text{LogL}_c - \hat{\sigma}_z^2(\theta_c)/2$ $\triangleright$ "Bias-correction".
5:    **for** $i \leftarrow 1, \ldots, N$ **do**
6:        $v_1 \sim \text{Uniform}(0, 1)$
7:        **if** $v_1 \leq \omega$ **then** $\triangleright$ Propose a new subset.
8:            $u_p \sim p(u)$ $\triangleright$ $p(u)$ SIR with m observations.
9:        **else**
10:            $u_p = u_c$ $\triangleright$ Keep the subset.
11:        **end if**
12:        $\theta_p \sim q(\theta|\theta_c)$ $\triangleright$ Propose parameter.
13:        $(\text{LogL}_p, \hat{\sigma}_z^2(\theta_p)) \leftarrow \text{LOGLIKELIHOOD}(\theta_p, u_p)$
14:        $\text{LogL}_p \leftarrow \text{LogL}_p - \hat{\sigma}_z^2(\theta_p)/2$ $\triangleright$ "Bias-correction".
15:        $\alpha \leftarrow 1 \wedge \exp\left(\text{LogL}_p + \log p(\theta_p) - \text{LogL}_c + \log p(\theta_c)\right) \times \frac{q(\theta_c|\theta_p)}{q(\theta_p|\theta_c)}$
16:        $v_2 \sim \text{Uniform}(0, 1)$
17:        **if** $v_2 \leq \alpha$ **then** $\triangleright$ Draw is accepted.
18:            $\text{LogL}_c \leftarrow \text{LogL}_p$
19:            $(\theta_c, u_c) \leftarrow (\theta_p, u_p)$
20:            $\theta_i \leftarrow \theta_p$
21:        **else**
22:            $\theta_i = \theta_c$
23:        **end if**
24:    **end for**
25:    **return** $\{\theta_i\}_{i=1}^N$
26: **end procedure**

Earlier Working Papers:

For a complete list of Working Papers published by Sveriges Riksbank, see www.riksbank.se