Bayesian Analysis (2018) **13**, Number 2, pp. 507–530

Merging MCMC Subposteriors through Gaussian-Process Approximations

Christopher Nemeth* and Chris Sherlock†

Abstract. Markov chain Monte Carlo (MCMC) algorithms have become powerful tools for Bayesian inference. However, they do not scale well to large-data problems. Divide-and-conquer strategies, which split the data into batches and, for each batch, run independent MCMC algorithms targeting the corresponding subposterior, can spread the computational burden across a number of separate computer cores. The challenge with such strategies is in recombining the subposteriors to approximate the full posterior. By creating a Gaussian-process approximation for each log-subposterior density we create a tractable approximation for the full posterior. This approximation is exploited through three methodologies: firstly a Hamiltonian Monte Carlo algorithm targeting the expectation of the posterior density provides a sample from an approximation to the posterior; secondly, evaluating the true posterior at the sampled points leads to an importance sampler that, asymptotically, targets the true posterior expectations; finally, an alternative importance sampler uses the full Gaussian-process distribution of the approximation to the log-posterior density to re-weight any initial sample and provide both an estimate of the posterior expectation and a measure of the uncertainty in it.

Keywords: big data, Markov chain Monte Carlo, Gaussian processes, distributed importance sampling.

1 Introduction

Markov chain Monte Carlo (MCMC) algorithms are popular tools for sampling from Bayesian posterior distributions in order to estimate posterior expectations. They benefit from theoretical guarantees of asymptotic convergence of the estimators as the number of MCMC samples grows. However, whilst asymptotically exact, they can be computationally expensive when applied to datasets with a large number of observations n. Indeed, the cost of generating one sample from the MCMC algorithm is at best $O(n)$ as the posterior distribution of the model parameters, conditional on the entire data set, must be evaluated at each MCMC iteration. For very large n, therefore, MCMC algorithms can become computationally impractical.

Research in the area of MCMC for big data can be broadly split into two streams: those which utilise one core of the central processing unit (CPU) and those that distribute the work load across multiple cores, or machines. For the single processor case, the computational cost of running MCMC on the full data set may be reduced by using

*Department of Mathematics and Statistics, Lancaster University, Lancaster LA1 4YF, U.K., c.nemeth@lancaster.ac.uk

†Department of Mathematics and Statistics, Lancaster University, Lancaster LA1 4YF, U.K., c.sherlock@lancaster.ac.uk

 https://doi.org/10.1214/17-BA1063

a random subsample of the data at each iteration (Quiroz et al., 2014; Maclaurin and Adams, 2014; Bardenet et al., 2014); however, the mixing of the MCMC chain can suffer as a result. Alternatively, the Metropolis-Hastings acceptance step can be avoided completely by using a stochastic gradient algorithm (Welling and Teh, 2011; Chen et al., 2014), where subsamples of the data are used to calculate unbiased estimates of the gradient of the log-posterior. Consistent estimates of posterior expectations are obtained as the gradient step-sizes decrease to zero (Whye Teh et al., 2016). While popular, subsampling methods do have the drawback that the data must be independent and the whole data set must be readily available at all times, and therefore data cannot be stored across multiple machines.

Modern computer architectures readily utilise multiple cores of the CPU for computation, but MCMC algorithms are inherently serial in implementation. Parallel MCMC, where multiple MCMC chains, each targeting the full posterior, are run on separate cores, or machines, can be easily executed (Wilkinson, 2005), however, this does not address the big-data problem as each machine still needs to store and evaluate the whole data set. In order to generate a significant computational speed-up, the data set must be partitioned into disjoint batches, where independent MCMC algorithms are executed on separate batches on independent processors (Huang and Gelman, 2005). Using only a subset of the entire data means that the MCMC algorithm is targeting a different posterior distribution conditional on only a subset of the data, herein referred to as a *subposterior*. This type of parallelisation is highly efficient as there is no communication between the parallel MCMC chains. The main challenge is to then reintegrate the samples from the separate MCMC chains to approximate the full posterior distribution. Scott et al. (2016) create a Gaussian approximation for the full posterior by taking weighted averages of the means and variances of the MCMC samples from each batch; this procedure is exact when each subposterior is Gaussian, and can work well approximately in non-Gaussian scenarios. Neiswanger et al. (2014) avoid the Gaussian assumption by approximating the subposteriors using kernel density estimation, however, kernel density approximations scale poorly in high dimensions (Liu et al., 2007). Also, the upper bounds on the mean squared error given in Neiswanger et al. (2014) grow exponentially with the number of batches, which is problematic in big data scenarios where the computational benefit of parallelisation is proportional to the number of available processors.

Previous approaches used to merge the product of subposterior densities have solely relied on the parameter samples outputted from each MCMC algorithm, but have neglected to utilise the subposterior densities which are calculated when evaluating the Metropolis-Hastings ratio. We place Gaussian-process (GP) priors on the log-density of each subposterior. The resulting approximation to the log of the full posterior density is a sum of Gaussian-processes, which is itself a Gaussian-process, assuming that each individual GP has finite variance. Using this formulation we can obtain point estimates of any expectation of interest. The uncertainty in these point estimates is captured by the covariance of the GP approximation to the posterior.

Starting from this Gaussian-process approximation to the full log-posterior density, we investigate three approaches to approximating the posterior. Firstly, an efficient

Hamiltonian Monte Carlo (HMC) algorithm (Neal, 2010) which targets the expectation of the posterior density (the exponential of the combined GP); samples from this provide our first means of estimating expectations of interest. Secondly, the HMC sample values may be sent to each of the cores, with each core returning the true log-subposterior at each of the sample points. Combining these *coincident* log-subposterior values provides the true posterior at the sampled points, which in turn provides importance weights for the HMC sample, leading to asymptotically consistent estimates of posterior expectations. Alternatively, one may wish to avoid the computational expense of running HMC on the expectation of the exponential of the GP, and of calculating the true subposteriors at a sample of points. We, therefore, also consider an importance proposal based upon any approximation to the true posterior and obtain repeated samples of importance weights by repeatedly sampling realisations of the GP approximation to the log-posterior. This provides both an estimate of any expectation of interest and a measure of its uncertainty.

This paper is structured as follows. Section 2 reviews the divide-and-conquer MCMC approach for sampling from the posterior, the HMC algorithm and importance sampling. Section 3 then outlines the creation of our Gaussian-process approximation for each of the individual subposteriors, and for combining these. In Section 4 we detail three methods for approximating posterior expectations, each utilising the combined Gaussian-process approximation. Section 5 provides a numerical comparison of our proposed method against four alternative algorithms. We consider five different statistical models leading to a range of posterior distributions. We compare our proposed method against competing algorithms from the literature, and show how our method is particularly successful at approximating non-Gaussian posteriors.

2 Bayesian inference and MCMC

Consider a data set $\mathcal{Y} = \{y_1, y_2, \ldots, y_n\}$ where we assume that the data are conditionally independent with a likelihood $\prod_{i=1}^{n} p(y_i|\vartheta)$, where $\vartheta \in \Theta \subseteq \mathbb{R}^d$ are model parameters. Assuming a prior $p(\vartheta)$ for the parameters, the posterior distribution for ϑ given $\mathcal{Y}$ is

$$\pi(\vartheta) = p(\vartheta|\mathcal{Y}) \propto p(\vartheta) \prod_{i=1}^{n} p(y_i|\vartheta). \tag{1}$$

Alternatively, the data set $\mathcal{Y}$ can be partitioned into C batches $\{\mathcal{Y}_1, \mathcal{Y}_2, \ldots, \mathcal{Y}_C\}$ where we define a *subposterior* operating on a subset of the data $\mathcal{Y}_c$ as

$$\pi_c(\vartheta) = p(\vartheta|\mathcal{Y}_c) \propto p(\mathcal{Y}_c|\vartheta)p(\vartheta)^{1/C}, \tag{2}$$

where $p(\vartheta)$ is chosen so that $p(\vartheta)^{1/C}$ is proper. The full posterior is given as the product of the subposteriors $\pi(\vartheta) \propto \prod_{c=1}^{C} \pi_c(\vartheta)$. In this setting we no longer require conditional independence of the data, but rather independence between the batches $\{\mathcal{Y}_c\}_{c=1}^{C}$, where now the data in each batch can exhibit an arbitrary dependence structure.

Creating an approximation to the posterior, $\pi(\vartheta)$, commences with sampling from each of the subposteriors $\pi_c(\vartheta)$ independently in parallel, where, given the independence

between data subsets, there is no communication exchange between the MCMC algorithms operating on the subposteriors. This type of parallelisation is often referred to as *embarrassingly parallel* (Neiswanger et al., 2014). The challenge then lies in *combining* the subposteriors, for which we propose using Gaussian-process approximations.

In this paper, we introduce the Hamiltonian Monte Carlo (HMC) algorithm as one possible MCMC algorithm that can be used to sample from $\pi_c(\vartheta)$. Moreover, we use HMC in Section 4 to sample from an approximation to the full posterior, $\pi(\vartheta)$. Other MCMC algorithms, including the random walk Metropolis (Roberts et al., 1997), Metropolis adjusted Langevin algorithm (Roberts and Rosenthal, 1998) and adaptive versions of these (e.g. Andrieu and Thoms, 2008) can also be used.

2.1 Hamiltonian Monte Carlo

We now provide a brief overview of Hamiltonian Monte Carlo and its application in this paper; the interested reader is referred to Neal (2010) for a full and detailed review. The HMC algorithm considers the sampling problem as the exploration of a physical system with $-\log \pi(\vartheta)$ corresponding to the potential energy at the position ϑ. We then introduce artificial momentum variables $\varphi \in \mathbb{R}^D$, with $\varphi \sim \mathcal{N}(0, M)$ being independent of ϑ. Here M is a mass matrix that can be set to the identity matrix when there is no information about the target distribution. This scheme now augments our target distribution so that we are now sampling (ϑ, φ) from their joint distribution

$$\pi(\vartheta, \varphi) \propto \exp\left(\log \pi(\vartheta) - \frac{1}{2}\varphi^\top M^{-1}\varphi\right), \tag{3}$$

the logarithm of which equates to minus the total energy of the system. Samples from the marginal distribution of interest, $\pi(\vartheta)$, are obtained by discarding the φ samples.

We can sample from the target distribution by simulating ϑ and φ through fictitious time τ using Hamilton's equations (see Neal (2010) for details)

$$d\vartheta = M^{-1}\varphi d\tau, \qquad d\varphi = \nabla_\vartheta \log \pi(\vartheta) d\tau. \tag{4}$$

The differential equations in (4) are intractable and must be solved numerically. Several numerical integrators are available which preserve the volume and reversibility of the Hamiltonian system (Girolami and Calderhead, 2011), the most popular being the *leapfrog*, or Stormer-Verlet integrator. The leapfrog integrator takes L steps, each of size ϵ, on the Hamiltonian dynamics (4), with one step given as follows:

$$\begin{aligned}
\varphi_{\tau+\frac{\epsilon}{2}} &= \varphi_\tau + \frac{\epsilon}{2}\nabla_{\vartheta_\tau} \log \pi(\vartheta_\tau), \\
\vartheta_{\tau+\epsilon} &= \vartheta_\tau + \epsilon M^{-1}\varphi_{\tau+\frac{\epsilon}{2}}, \\
\varphi_{\tau+\epsilon} &= \varphi_{\tau+\frac{\epsilon}{2}} + \frac{\epsilon}{2}\nabla_{\vartheta_{\tau+\epsilon}} \log \pi(\vartheta_{\tau+\epsilon}).
\end{aligned}$$

Using a discretisation introduces a small loss or gain in the total energy, which is corrected through a Metropolis-Hastings accept/reject step. The full HMC algorithm is given in the Supplementary Material (Nemeth and Sherlock, 2017).

The HMC algorithm has a step-size parameter ϵ and number of leap frog steps L which need to be tuned. The performance of the algorithm is highly dependent on the tuning of the parameters. One way to tune the algorithm is to optimise the parameters such that the acceptance rate is approximately 65% (Beskos et al., 2013). Alternatively, the parameters could be adaptively tuned (Wang et al., 2013); for this paper, we use the popular NUTS sampler Hoffman and Gelman (2014), which tunes the trajectory length L to avoid the sampler doubling back on itself. The HMC algorithm can be efficiently implemented using the popular STAN (Carpenter et al., 2016) software package. The STAN modelling language automatically tunes the HMC algorithm, and by using efficient automatic differentiation, the user need only express their posterior model.

2.2 Importance sampling

A popular alternative to MCMC for estimating posterior expectations is the importance sampler (Robert and Casella, 1999). Given a proposal density, $q(\theta)$, and an unnormalised posterior density, $\pi(\theta)$, importance sampling (e.g. Geweke, 1989) aims to estimate expectations of some measurable function of interest, $h(\theta)$ by sampling from q. The starting point is

$$\mathbb{E}_{\pi/Z}[h(\theta)] = \frac{1}{Z}\int h(\theta)\frac{\pi(\theta)}{q(\theta)}q(\theta)d\theta = \frac{1}{Z}\mathbb{E}_q[h(\theta)\overline{w}(\theta)], \tag{5}$$

where $\overline{w}(\theta) := \pi(\theta)/q(\theta)$ and $Z := \int \pi(\theta)\mathrm{d}\theta$ is the normalisation constant.

Consider a sequence, $\{\theta_i\}_{i=1}^{\infty}$ with marginal density q. Provided that a strong law of large numbers (SLLN) applies, setting $h(\theta) = 1$ in the above equation implies that $\hat{Z}_N := \frac{1}{N}\sum_{i=1}^{N}\overline{w}(\theta_i) \to Z$, almost surely, and hence, almost surely,

$$\hat{E}_N(h) := \frac{1}{N}\sum_{i=1}^{N} w_N(\theta_i)h(\theta_i) \to \mathbb{E}_{\pi/Z}[h(\theta)], \tag{6}$$

where $w_N(\theta) := \overline{w}(\theta)/\hat{Z}_N$. In Section 4 we will use importance sampling to estimate expectations with respect to the combined posterior distribution.

3 A Gaussian-process approximation to the posterior

3.1 Gaussian-process approximations to the subposteriors

Parallelising the MCMC procedure over C computing nodes results in C subposteriors $\{\pi_c(\vartheta)\}_{c=1}^{C}$. The MCMC algorithm for each subposterior, c, has been iterated J times to give $\mathcal{D}_c = \{\vartheta_j, \ell_c(\vartheta_j)\}_{j=1}^{J}$, where $\ell_c(\vartheta_j) = \log \pi_c(\vartheta_j)$ and each pair consists of a sample from the Markov chain with its associated log-subposterior density, up to some fixed additive constant. We wish to convert this limited information on a finite set of points to information about $\log \pi_c$ over the whole support of ϑ. We start, *a priori*, by treating the whole log-subposterior (up to the same additive constant), $\mathcal{L}_c(\vartheta)$, as random with a Gaussian-process prior distribution:

$$\mathcal{L}_c(\vartheta) \sim \mathcal{GP}(m(\vartheta), K(\vartheta, \vartheta')),$$

where $m : \vartheta \to \mathbb{R}$ and $K : \vartheta \times \vartheta \to \mathbb{R}$ are, respectively, the mean and covariance functions. We model $\log \pi_c(\vartheta)$ rather than $\pi_c(\vartheta)$, so that our approximation to the overall log-posterior will be a sum of Gaussian-process (Section 3.3); modelling the log-posterior also avoids the need for non-negativity constraints when fitting the GP[1].

The mean function and covariance function are chosen by the user. A mean function of zero, $m(\vartheta) = 0$, would be inappropriate in this setting as our prior must be the logarithm of a probability density function up to a *finite* additive constant. We ensure that $\int \exp\{\mathcal{L}_c(\vartheta)\}\mathrm{d}\vartheta < \infty$ almost surely through a quadratic mean function of the form

$$m(\vartheta) = \beta_0 + \vartheta_1^\top \beta_1 + \mathrm{diag}(\vartheta^\top V^{-1}\vartheta)\beta_2, \quad \beta_2 < 0.$$

Here V is the empirical covariance the posterior for ϑ obtained from the MCMC sample and β_i, $(i = 0, 1, 2)$ are unknown constants. See Section 3.2 for a discussion on the choice of mean function.

The covariance function $K(\cdot, \cdot)$, determines the smoothness of the log-subposterior, which we shall assume is continuous with respect to ϑ. A popular choice is the squared-exponential function (e.g. Rasmussen and Williams, 2006)

$$K(\vartheta, \vartheta') = \omega^2 \exp\left(-\frac{1}{2}(\vartheta - \vartheta')^T \Lambda^{-1}(\vartheta - \vartheta')\right), \tag{7}$$

where Λ is a diagonal matrix and ω are hyperparameters. In this paper we analytically marginalise β_0 and β_1 (O'Hagan, 1978) and estimate β_2 and the kernel hyperparameters through maximum likelihood (details given in Chapter 5 of Rasmussen and Williams (2006)). Alternative functions can be applied and may be more appropriate depending on the characteristics of the log-subposterior density.

Given the choice of prior, $\mathcal{D}_c$ are observations of this Gaussian-process generated from an MCMC algorithm targeting the subposterior $\pi_c(\vartheta)$, giving up to a constant of proportionality the posterior distribution,

$$p(\ell_c(\vartheta)|\mathcal{D}_c) \propto p(\mathcal{D}_c|\ell_c(\vartheta))p(\ell_c(\vartheta)). \tag{8}$$

Define $\mathcal{L}_c(\vartheta_{1:J}) := \{\mathcal{L}_c(\vartheta_1), \ldots, \mathcal{L}_c(\vartheta_J)\}$ and, for some parameter, or parameter vector, $\theta := \theta_{1:N} := (\theta_1, \ldots, \theta_N)$, define $\mathcal{L}_c(\theta_{1:N}) := \{\mathcal{L}_c(\theta_1), \ldots, \mathcal{L}_c(\theta_N)\}$. We require the posterior distribution of $\mathcal{L}_c(\theta_{1:N})|\mathcal{D}_c$, that is the conditional distribution $\mathcal{L}_c(\theta_{1:N})|\{\mathcal{L}_c(\vartheta_{1:J}) = \ell_c(\vartheta_{1:J})\}$. Since the joint distribution between $\mathcal{L}_c(\vartheta_{1:J})$ and $\mathcal{L}_c(\theta_{1:N})$ is multivariate Gaussian, the conditional, $\mathcal{L}_c(\theta_{1:N})|\{\mathcal{L}_c(\vartheta_{1:J}) = \ell_c(\vartheta_{1:J}\})$ is also multivariate Gaussian,

$$\mathcal{L}_c(\theta_{1:N})|\mathcal{D}_c \sim \mathcal{N}(\mu_c(\theta_{1:N}), \Sigma_c(\theta_{1:N})) \tag{9}$$

with,

$$\begin{aligned} \mu_c(\theta_{1:N}) &= m_c(\theta_{1:N}) + K_*^\top \tilde{K}^{-1}(\mathcal{L}_c(\vartheta_{1:J}) - m_c(\vartheta_{1:J})) \\ \Sigma_c(\theta_{1:N}) &= K_{*,*} - K_*^\top \tilde{K}^{-1} K_*, \end{aligned} \tag{10}$$

and where $\tilde{K} = K(\vartheta_{1:J}, \vartheta_{1:J})$, $K_{*,*} = K(\theta_{1:N}, \theta_{1:N})$ and $K_* = K(\vartheta_{1:J}, \theta_{1:N})$.

The posterior distribution for the GP, $\mathcal{L}_c(\theta_{1:N})|\mathcal{D}_c$, up to a constant of proportionality, is a stochastic approximation of the log-subposterior surface.

[1] GaussianProcesses.jl (Fairbrother et al., 2017), is used for simulations.

3.2 Illustration

The Gaussian-process subposterior provides an estimate of the uncertainty in the log-subposterior at points, θ, where the log-subposterior has not been evaluated. This contrasts with current approaches (e.g. Scott et al., 2016; Neiswanger et al., 2014; Wang and Dunson, 2013) to approximate the subposterior which give no measure of uncertainty. This is illustrated below and used in Sections 4.3 and 5.4 to gauge the uncertainty in our estimates of posterior expectations.

When approximating a log-density, we assume that the mean function $m_c(\theta)$ of the Gaussian process approximation to each log-subposterior, $\mathcal{L}_c(\theta)$, has a quadratic form. This assumption ensures that $\mathcal{L}_c(\theta) \to -\infty$ as $\theta \to \pm\infty$. Alternatively, the quadratic form of the log-subposterior could be modelled via the kernel by using a zero mean function and taking $K(\theta, \theta')$ to be the product of two linear kernels (Duvenaud, 2014). We test the quality of the proposed GP prior specification on three models shown in Figure 1. Using our GP prior specification (7) we model the posterior of the Gaussian process approximation (10) on three models: standard Gaussian, skew Gaussian and mixture of Gaussians. Figure 1 gives the density of each model (left panel) along with a samples from the GP prior (centre panel), where the hyperparameters of the GP prior are estimated based on the observations, and the posterior (right panel) is fit to observations of the log-density of the respective model. The right panels of Figure 1 show that the GP provides a good approximation to the log-density of the respective models where the model has been evaluated. Outside of the range of evaluation points (i.e. as $\theta \to \pm\infty$) where there are no observations of the log-density, the GP reverts back to a quadratic form. We see that for these three models, the GP provides a good representation of the mass of the density, but can under-represent the tails of the distribution. Using our GP to model to approximate log-subposteriors can provide a good fit if the points at which we fit the GP are an appropriate representation of the true distribution. This will be the case if we fit the GP to MCMC samples $\{\vartheta_j\}_{j=1}^{J}$ drawn from the stationary distribution.

3.3 Merging the subposteriors

Our next goal is to approximate the full posterior $\pi(\theta) \propto \prod_{c=1}^{C} \pi_c(\theta)$ by merging the subposteriors together. The approximation of each of the C subposteriors as independent Gaussian-processes, $\mathcal{L}_c(\theta) \sim \mathcal{GP}(\cdot, \cdot)$ $(c = 1, \ldots, C)$ leads directly to the approximation of the full log-posterior (up to an additive constant) as the sum of C Gaussian-processes,

$$\mathcal{L}(\theta)|\mathcal{D} \propto \sum_{c=1}^{C} [\mathcal{L}_c(\theta)|\mathcal{D}_c] = \mathcal{GP}\left(\sum_{c=1}^{C} \mu_c(\theta), \sum_{c=1}^{C} \Sigma_c(\theta)\right). \tag{11}$$

Our assumption that the Gaussian-processes representing the log-subposteriors $\{\mathcal{L}_c\}_{c=1}^{C}$ are independent *a priori* follows by assuming that the subposteriors are independent (2). This may not be true in practice where deviations from the quadratic prior mean, $m_c(\vartheta)$, may be present across subposteriors. However, *a posteriori* these deviations should be accounted for through the posterior mean $\mu_c(\vartheta)$. Variability in the original partitioning of the data into batches, and variability in the sample points, $\vartheta_{1:J}$,

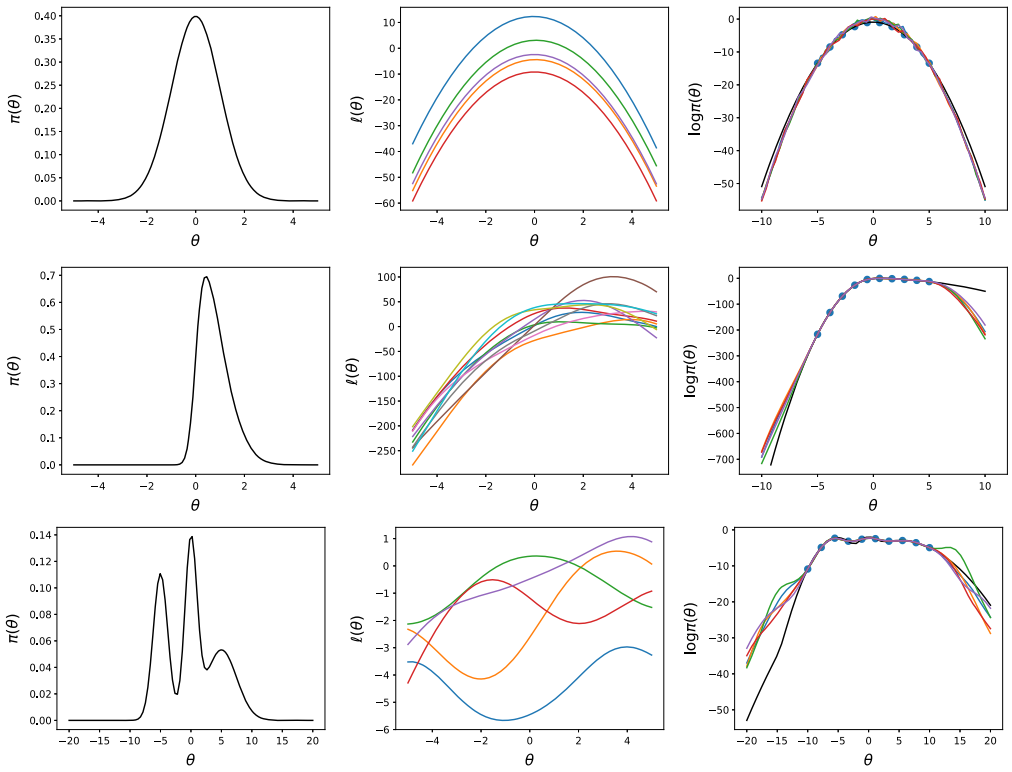

Figure 1: Left: Target density. Centre: Five random samples from the GP prior with varying hyperparameters. Right: Posterior samples from the GP conditional on samples and log-density estimates of the target.

across batches will both contribute to the more subtle variations of the GPs about their individual posterior means, so that the posterior correlation should be much smaller than the prior correlation.

4 Approximating the full posterior

We now detail three methods for approximating posterior expectations, all of which utilise our Gaussian-process approximation to the full posterior density.

4.1 The expected posterior density

Here we approximate the full posterior density (up to an unknown normalising constant) by its expectation under the Gaussian-process approximation:

$$\hat{\pi}_E(\theta) \propto \mathbb{E}[\exp\left(\mathcal{L}(\theta)|\mathcal{D}\right)] = \exp\left\{\sum_{c=1}^{C} \mu_c(\theta) + \frac{1}{2}\Sigma_c(\theta)\right\}, \tag{12}$$

using the properties of the log-Normal distribution. If the individual GPs provide a good approximation to the individual log-subposteriors, then $\mathbb{E}[\mathcal{L}(\theta)]$ will be a good approximation to the full log-posterior.

The HMC algorithm then provides an efficient mechanism for obtaining an approximate sample, $\{\theta_i\}_{i=1}^N$ from $\hat{\pi}_E$. Evaluating the GP approximation at each iteration of this MCMC algorithm is significantly faster than evaluating the true full posterior, $\pi(\theta)$, directly. As is apparent from the leapfrog dynamics, HMC requires the gradient of $\log \pi_E$, and here the tractability of our approximation is invaluable, since

$$\begin{aligned}\nabla \log \hat{\pi}_E &= \sum_{c=1}^{C} \frac{\partial}{\partial\theta}\mu_c(\theta) + \frac{1}{2}\frac{\partial}{\partial\theta}\Sigma_c(\theta) \\ &= \sum_{c=1}^{C} \frac{\partial}{\partial\theta} m(\theta) + \frac{\partial K_*^\top}{\partial\theta}\tilde{K}^{-1}(\ell_c(\vartheta_{1:J}) - m(\vartheta_{1:J})) + \frac{1}{2}\frac{\partial}{\partial\theta}K_{*,*} - \frac{\partial}{\partial\theta}K_*^\top\tilde{K}^{-1}.\end{aligned}$$

Given a sufficiently large sample from $\hat{\pi}_E$, approximations of posterior expectations can be highly accurate if the individual GPs provide a good approximation to the log-subposteriors. Moreover, the approximation $\hat{\pi}_E(\theta)$ to the full posterior can be further improved by using importance sampling on the true posterior.

4.2 Distributed importance sampling

Unlike the proposal, q, in Section 2.2, samples generated from the HMC algorithm represent an approximate, correlated sample from an approximation to the true posterior, instead of exact, independent samples from an approximation. Nonetheless, we may still correct for inaccuracies in $\hat{\pi}_E$ using importance sampling while spreading the computational burden across all C cores.

The full sample from the HMC algorithm targeting $\hat{\pi}_E$, $\{\theta_i\}_{i=1}^N$, is sent to each of the C cores. Each of the C cores then evaluates the true subposterior at each θ_i. A single core then combines the subposterior densities for each θ_i to provide the full true posterior density: $\pi(\theta_i) = \prod_{c=1}^{C} \pi_c(\theta_i)$. To be clear, each sub-posterior is evaluated at the *same* set of θ values, allowing them to be combined exactly. In contrast, the original HMC runs, performed on each individual subposterior, created a different set of θ values for each subposterior so that a straightforward combination was not possible.

Each value from the sample, θ_i, is then associated with an unnormalised weight, $\overline{w}(\theta_i) = \pi(\theta_i)/\hat{\pi}_E(\theta_i)$. Defining $\hat{Z}_N$ and $w_N(\theta)$ as in Section 2.2 provides an approximation $\hat{E}_N(h)$ to $\mathbb{E}_\pi[h(\theta)]$ as defined in (6).

Since the unknown normalising constants for both π and $\hat{\pi}_E$ appear in both the numerator and the denominator of this expression, they are not needed. Almost sure convergence of $\hat{E}_N(h)$ to $\mathbb{E}_\pi[h(\theta)]$ as the HMC sample size, $N \to \infty$ relies on the strong law of large numbers (SLLN) for Markov chains (e.g. Tierney, 1996, Theorem 4.3). In addition, if desired, an unweighted approximate sample from π may be obtained by resampling θ_i with a probability proportional w_i.

Algorithm 1 Distributed Importance Sampler

Input: Proposal distribution $q(\theta_{1:N})$, which has marginal $q_1(\theta_i)$ identical for all $\theta_i,\ i = 1, \ldots, n$.
- Sample $\theta_{1:N} \sim q(\cdot)$.
- For $c = 1, \ldots, C$ evaluate each subposterior $\pi_c(\theta_{1:N})$.
- Set $\pi(\theta_i) = \prod_{c=1}^{C} \pi_c(\theta_i)$.
- Weight the samples, $w_i = \frac{\pi(\theta_i)}{q_1(\theta_i)}$.

Output: Weighted sample $\{w_i, \theta_i\}_{i=1}^{N}$ approximating $\pi(\theta)$.

We expect our HMC importance proposal to be especially efficient since it mimics the true posterior. However, other proposal distributions based on competing algorithms for merging subposteriors (e.g. Scott et al., 2016; Neiswanger et al., 2014; Wang and Dunson, 2013) can be used instead; these are compared in Section 5. Algorithm 1 describes this general distributed importance sampler.

4.3 Gaussian-process importance sampler (GP-IS)

Finally, we present an importance sampler that uses the posterior distribution of $\mathcal{L}$, the GP approximation to the unnormalised log-posterior conditional on $\{\vartheta_{c,j}, \pi_c(\vartheta_{c,j})\}_{c=1,j=1}^{C,J}$. Compared with the importance sampler in Section 4.2, the set of points $\{\theta_i\}_{i=1}^{N}$ is generated from a simple proposal distribution, rather than the HMC algorithm applied to $\hat{\pi}_E$. Moreover, given the set of points $\{\theta_i\}_{i=1}^{N}$ the computationally-expensive evaluation of each subposterior at this set of values is replaced with repeated, but relatively cheap sampling of realisations of $\mathcal{L}$ at these points. For a fixed number of GP training points, J, estimates of posterior expectations are no-longer asymptotically exact in N, however estimates of the uncertainty in these estimates are also supplied.

As in Sections 4.2 and 2.2 we are interested in $\mathsf{I}_h := \mathbb{E}_\pi[h(\theta)] = \frac{1}{Z}\int \pi(\theta)h(\theta)\mathrm{d}\theta$. Here we consider approximating this with

$$\mathsf{I}_h(\ell) := \frac{1}{Z(\ell)} \int \exp\{\ell(\theta)\}\, h(\theta)\mathrm{d}\theta,$$

where ℓ is a realisation of $\mathcal{L}$ from the distribution in (11) and $Z(\ell) := \int \exp\{\ell(\theta)\}\mathrm{d}\theta$ is the associated normalisation constant.

First, consider the hypothetical scenario where it is possible to store ℓ completely and evaluate $\mathsf{I}_h(\ell)$. A set of M realisations of $\mathcal{L}$, $\{\ell_m\}_{m=1}^{M}$ would lead to M associated estimates of I_h, $\{\mathsf{I}_h(\ell_m)\}_{m=1}^{M}$, which would approximate the posterior distribution of I_h under (11). The mean of these would then target, the posterior expectation,

$$\mathsf{I}_h^{\mathbb{E}} := \mathbb{E}\left[\frac{1}{Z(\mathcal{L})}\int h(\theta)\exp\left(\mathcal{L}(\theta)\right)\mathrm{d}\theta\right].$$

As an alternative, robust, point estimate, the median of $\{\mathsf{I}_h(\ell_m)\}_{m=1}^{M}$ would target the posterior median. Other posterior summaries for I_h, such as a 95% credible interval, could also be estimated from the sample.

Algorithm 2 GP Importance Sampler

Input: GP approximation $\mathcal{L}(\theta)$ and proposal distribution $\mathsf{q}(\theta)$.
- Sample $\theta_{1:N} \sim \mathsf{q}(\cdot)$ iid.
- Sample $m = 1, \ldots, M$ realisations of the GP approximation to the log-posterior $\mathcal{L}_m(\theta_{1:N})$ in (11).
- Weight the samples according to (13).

Output: Weighted sample $\{w_i, \theta_i\}_{i=1}^N$, approximately from the marginal of $\{\mathcal{L}, \pi(\theta|\mathcal{L})\}$.

Unfortunately, it is not possible to store the infinite-dimensional object, ℓ; and even if it were, for moderate dimensions, numerical evaluation of $\mathsf{I}_h(\ell)$ would be computationally infeasible. Instead, we use importance sampling. Consider a proposal distribution $q(\theta)$ that approximately mimics the true posterior distribution, $\pi(\theta)$ and sample N independent points from it: $\theta_{1:N} := (\theta_1, \ldots, \theta_N)$. For each $m \in \{1, \ldots, M\}$ we then sample the *finite-dimensional* object $(\ell_m(\theta_1), \ldots, \ell_m(\theta_N))$ from the joint distribution of the GP in (11). For each such realisation we then construct an approximation to the normalisation constant and to $\mathsf{I}_h(\ell)$:

$$\hat{Z}(\ell_m) := \frac{1}{N}\sum_{i=1}^{N} \overline{w}(\theta_i; \ell_m) \quad \text{and} \quad \hat{\mathsf{I}}_h(\ell_m) := \frac{1}{N\hat{Z}(\ell_m)}\sum_{i=1}^{N} \overline{w}(\theta_i; \ell_m) h(\theta_i),$$

where $\overline{w}(\theta; \ell) := \exp\{\ell(\theta)\}/q(\theta)$. The set $\{\hat{\mathsf{I}}_h(\ell_m)\}_{m=1}^M$ is then used in place of $\{\mathsf{I}_h(\ell_m)\}_{m=1}^M$ for posterior inference on I_h.

For the specific case of $\mathsf{I}_h^{\mathbb{E}}$ a simplified expression for the approximation may be derived:

$$\hat{\mathsf{I}}_h^{\mathbb{E}} = \frac{1}{N}\sum_{i=1}^{N} w_i h(\theta_i), \quad \text{where } w_i := \frac{1}{Mq(\theta_i)}\sum_{m=1}^{M} \frac{\exp\{\ell_m(\theta_i)\}}{\hat{Z}(\ell_m)}. \tag{13}$$

Algorithm 2 creates point estimates based upon this.

The proposal density $q(\theta_i)$ should be a good approximation to the posterior density. To create a computationally cheap proposal, and with a similar motivation to the consensus Monte Carlo approximation (Scott et al., 2016), we make $q(\theta_i)$ a multivariate Student-t distribution on 5 degrees of freedom with mean and variance matching those of the Gaussian posterior that would arise given the mean and variance of each sub-posterior and if each sub-posterior were Gaussian. Alternatively, it would be possible to use the output from the HMC algorithm of Section 4.1 in an analogous manner to the way it is used in Section 4.2.

Many aspects of our importance sampler can, if necessary, be parallelised: in particular, calculating $\mu_c(\theta_{1:N})$ and $\Sigma_c(\theta_{1:N})$, and then sampling $\ell_1, \ldots, \ell_m$ and obtaining the sample $\{\hat{\mathsf{I}}_h(\ell_m)\}_{m=1}^m$.

4.4 Computational cost

We briefly review some of the notation in the paper as a point of reference for this section.

- $n :=$ Number of data points, y.
- $C :=$ Number of processing cores (i.e. number of batches).
- $J :=$ Number of MCMC samples drawn from $\pi_c(\theta),\ c = 1, \ldots, C$; for simplicity, we assume J samples are drawn from each subposterior.
- $N :=$ Number of samples drawn from the approximation to the merged posterior $\hat{\pi}_E(\theta)$, or, for GP-IS, from the Student-t proposal.

The overall computational cost of applying the methods in Sections 4.1 and 4.2 to create an approximate (weighted) sample from the full posterior can be summarised in three (four) steps:

– **Run MCMC on each subposterior** (see Section 2). This step is common to all divide-and-conquer MCMC algorithms (e.g. Scott et al., 2016; Neiswanger et al., 2014; Wang et al., 2015) and has a cost of $\mathcal{O}(Jn/C)$.

– **Fit GP to each subposterior** (see Section 3). Fitting a Gaussian-process to each subposterior has a cost of $\mathcal{O}(J^3)$ due to the inversion of the $J \times J$ matrix $\tilde{K}$. One of the drawbacks of Gaussian-processes is the computational cost. Faster, approximate Gaussian-processes, referred to as *sparse GPs* (e.g Csató and Opper, 2002; Seeger et al., 2003; Quiñonero-Candela et al., 2005; Snelson and Ghahramani, 2006) can be used to reduce the computational cost (see Section 5.4).[2] In this paper we apply the simpler speed-up technique of first thinning the subposterior Markov chain; for example, using only every twentieth sample. The thinned Markov chain has the same stationary distribution as the full chain, but the autocorrelation is reduced and, more importantly for us, the sample contains fewer points. Secondly, we remove duplicate samples from the subposterior; because we have the log-density of the subposterior, these duplicate samples provide no additional information when fitting the GP, and can cause the kernel matrix $\tilde{K}$ to become singular. Fitting C independent GPs to each of the subposteriors is embarrassingly parallel as the MCMC output from each subposterior is stored on a separate core.

– **Perform HMC on** $\hat{\pi}_E$ (see Section 4.1). Each iteration of the HMC algorithm requires an evaluation of μ_c and Σ_c from (10) with $N = 1$, and multiple evaluations of the gradient terms given in Section 4.1. Since $\tilde{K}^{-1}$ has already been calculated, the total cost over all N iterations of the HMC algorithm is $\mathcal{O}(NJ^2)$. The cost of this step is equivalent to competing algorithms including (Neiswanger et al., 2014; Wang and Dunson, 2013), which also use an MCMC-type step to sample from the approximation to the posterior.

[2] Generally speaking, sparse GPs introduce p inducing points as training point locations $\vartheta_{P_i},\ i = 1, \ldots, p$, to fit the GP. The computational complexity of such an approach is reduced if $P << J$ to give an overall cost of $O(P^2 J)$.

– **Re-weight GP samples for DIS** (see Section 4.2). Our distributed importance sampler weights the approximate samples from $\hat{\pi}_E$ according to the true posterior and requires the subposteriors to be re-evaluated at each point in the GP-HMC sample. More generally, a sample from any sensible proposal distribution could be used. This has a cost of $\mathcal{O}(Nn/C)$.

– **GP-IS** (see Section 4.3). Taking the proposal, q, to be the Student-t distribution described at the end of Section 4.3, creating a sample of size N has a cost of $\mathcal{O}(N)$. For the sample, $\theta_{1:N}$, creation of each $\Sigma_c(\theta_{1:N})$ $(c = 1, \ldots, C)$ in parallel is $\mathcal{O}(N^2J + NJ^2)$. Cholesky decomposition of their sum, Σ, is $\mathcal{O}(N^3)$; however a spectral decomposition truncated to the largest T eigenvalues is $\mathcal{O}(N^2T)$. The M multiplications $\Sigma^{1/2}z$ (where z is a vector of independent standard Gaussians) that generate realisations of $\mathcal{L}$ can be spread between processors, leading to a cost of $\mathcal{O}(MN^2/C)$ (or $\mathcal{O}(MNT/C)$ for a truncated spectral decomposition).

GP-IS may, therefore, be preferable to DIS when $MN << n$ (or $MT << n$).

5 Experiments

In this section we compare our Gaussian-process algorithms for merging the subposteriors against several competing algorithms:

- **Consensus Monte Carlo** (Scott et al., 2016), where samples are weighted and aggregated.
- **Nonparametric density product** Neiswanger et al. (2014), where each subposterior is approximated using kernel density estimation.
- **Semiparametric density product**[3] Neiswanger et al. (2014), similar to the nonparametric method, but where subposteriors are approximated semiparametrically as in Hjort and Glad (1995).
- **Weierstrass rejection sampler**[4] (Wang and Dunson, 2013), where the nonparametric density estimates are passed through a Weierstrass transform to give the merged posterior.

We consider five examples (one in the Supplementary Material) which capture common distributional features and popular statistical models: a warped Gaussian target, a mixture of bivariate Gaussians, a Bernoulli model with rare events which leads to a skewed posterior and two logistic regression models for large data sets. Additionally, in the Supplementary Material, we consider a mixture of Laplace distributions which only becomes identifiable with a large amount of data. These examples highlight some of the challenges faced by merging non-Gaussian subposteriors and the computational efficiency of large-scale Bayesian inference.

[3]Implemented using the *parallelMCMCcombine* R package.
[4]Implemented using the authors R package https://github.com/wwrechard/weierstrass.

Our Gaussian-process approximation method is implemented using $J = 100$ samples from the thinned chain for each subposterior to fit the GPs for the Bernoulli and multimodal examples; for the logistic regression examples $J = 500$. Both for our methods and for competitor methods, $N = 5000$ samples from each merged posterior are created. To ensure a fair comparison, the sample from each $\hat{\pi}_E$ that is used both directly and in our DIS algorithm is the *unthinned* output from the HMC run. The Student-t proposals for the Gaussian-process importance sampler are iid.

Weighted samples from DIS and GP-IS are converted to unweighted samples by resampling with replacement, where the probability of choosing a given θ is proportional to its weight.

For each of the models studied in this section we denote the true parameter values by θ^* (when known). We obtain an accurate estimate of the true posterior from a long MCMC run, thinned to a size of N, with samples denoted by θ^f and the true posterior mean and variance m_f and V_f, respectively. Samples from the approximation are denoted by θ^a, and their mean and variance are m_a and V_a. We use the following metrics to compare the competing methods:

- Mahalanobis distance, $D_{Mah.} = \sqrt{(\mathsf{m_a} - \mathsf{m_f})^\top \mathsf{V_f}^{-1} (\mathsf{m_a} - \mathsf{m_f})}$.
- Kullback–Leibler divergence for the Bernoulli and mixture example is calculated using a nearest neighbour search[5] and for the logistic regression example, approximate multivariate Gaussian Kullback–Leibler divergence (see Wang and Dunson (2013) for details) between the true posterior π and merged posterior $\hat{\pi}$ is calculated as

$$D_{KL}(\hat{\pi}(\theta)||\pi(\theta)) = \frac{1}{2}(\mathrm{tr}(\mathsf{V_f}^{-1}\mathsf{V_a}) + (\mathsf{m_f} - \mathsf{m_a})^\top \mathsf{V_f}^{-1}(\mathsf{m_f} - \mathsf{m_a}) - d - \log(|\mathsf{V_a}|/|\mathsf{V_f}|))).$$

- Posterior concentration ratio, $\rho = \sqrt{\sum_{i=1}^N ||\theta_i^{\mathsf{a}} - \theta^*||_2^2 / \sum_{i=1}^N ||\theta_i^{\mathsf{f}} - \theta^*||_2^2}$ (Wang et al., 2015), which gives a measure for the posterior spread around the true value θ^* (ρ=1 being ideal).
- Mean absolute skew deviation, $\eta = \frac{1}{d}\sum_{i=1}^d |\gamma_i^{\mathsf{a}} - \gamma_i^{\mathsf{f}}|$, where i is the component, $\gamma_i = \mathbb{E}[\{(\theta_i - \mathsf{m}_i)/\mathsf{V}_{ii}^{1/2}\}^3]$ is the third standardised moment, and the superscripts f and a denote empirical approximations obtained from the samples obtained using the true posterior and the approximation, respectively.

5.1 Warped Gaussian model

We start by considering the warped Gaussian distribution, where the posterior, and subposteriors, exhibit a complex banana shape as a result of the lack of identifiability of

[5]Implemented using the *FNN* R package.

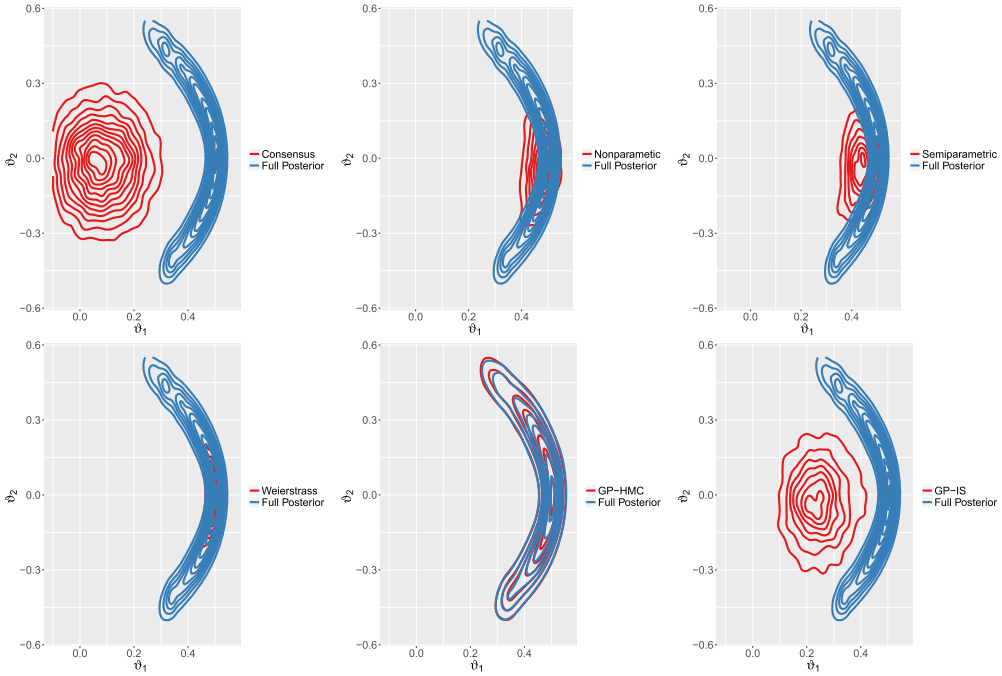

Figure 2: Posterior contour plots for the full posterior of the warped Gaussian target, overlayed with each of the competing algorithms.

the sign of one parameter. We simulate $n = 50{,}000$ observations from a warped Gaussian distribution with density,

$$p(y_i|\vartheta) = \mathcal{N}(y_i|\vartheta_1 + \vartheta_2^2, \sigma^2),$$

where $\vartheta = (0.5, 0.0)$. We assume the variance σ^2 is known and the prior for ϑ is $\mathcal{N}(0, 0.5)$. The data is split across $C = 20$ processors with independent Hamiltonian MCMC algorithms (Carpenter et al., 2016) applied to each subset of the data targeting independent subposteriors. The subposteriors are re-merged using one of the competing methods and the posterior contour plots for each competing method are given in Figure 2. Out of the various re-merging algorithms, only the GP-HMC is able to accurately approximate the full posterior. The kernel density based methods are to reasonable capture the posterior mode and approximate posterior shape, but significantly underestimate the variance of ϑ_2. The GP-IS sampler also struggles to adequately approximate the posterior. This is due to the importance proposal $q(\vartheta)$, which is a multivariate t-distribution approximation of the consensus Monte Carlo posterior. This example illustrates that the accuracy of the GP-IS sampler is highly dependent on the choice of importance proposal, and while the GP approximation can correct for some of the discrepancy in the proposal, generating a good approximation to the full posterior from the GP-IS sampler requires an importance proposal that sufficiently captures the posterior covariance structure.

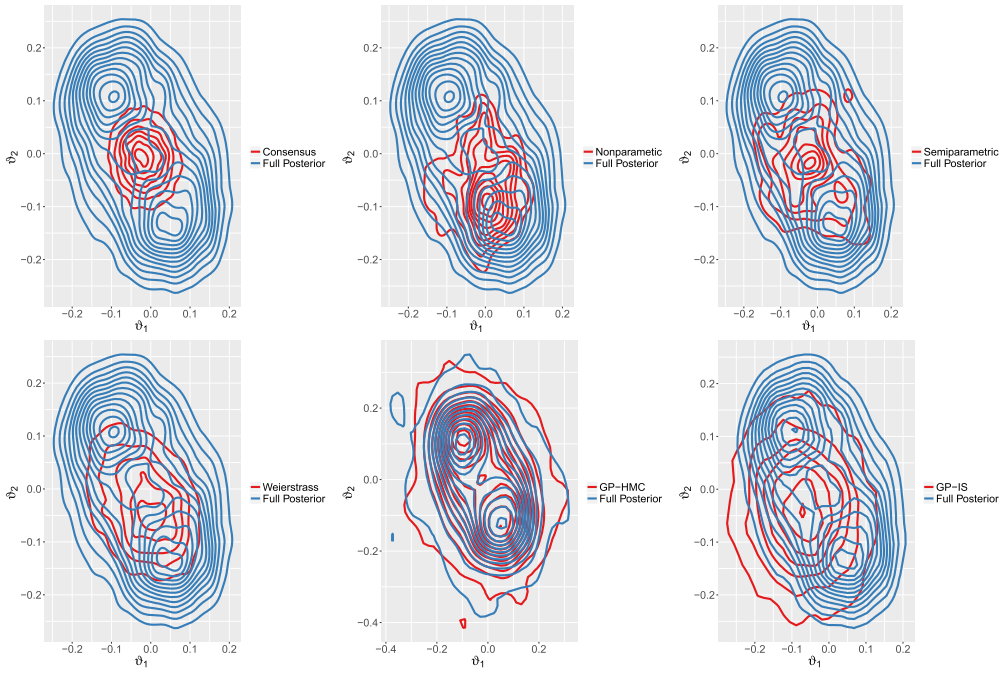

Figure 3: Posterior contour plots for the full posterior of the mixture of Gaussians target, overlayed with each of the competing algorithms.

5.2 Mixture of Gaussians

Mixture models are popular in the divide-and-conquer MCMC literature (Wang and Dunson, 2013; Neiswanger et al., 2014). We sample $n = 50{,}000$ observations from a mixture of two bivariate Gaussian distributions with density,

$$p(y_i|\vartheta) = \frac{1}{2}\mathcal{N}(y|\vartheta_1, \mathbf{I}_2) + \frac{1}{2}\mathcal{N}(y|\vartheta_2, \mathbf{I}_2),$$

where $\vartheta_1 = (0.1, 0.1)$ and $\vartheta_2 = (-0.1, -0.1)$. We assume independent priors on $\vartheta \sim \mathcal{N}(0, 100)$ and split the data across $C = 20$ processors and run independent Hamiltonian MCMC (Carpenter et al., 2016) on each subposterior. This model has been constructed so that the posterior density exhibits bimodality. This is a result of placing the modes of the mixture components close together causing the MCMC algorithm to jump between the modes. Applying each of the merging algorithms to the subposteriors, we can see from the approximation to the full posterior, shown in Figure 3, that only the GP-HMC algorithm is able to sufficiently capture both the posterior mode and covariance structure. The kernel density methods are able to reasonably capture the mass of the posterior, but underestimate the covariance of the recombined full posterior. Unsurprisingly, the consensus Monte Carlo approximation is unable to capture the bimodality of the posterior, however, using the consensus approximation as a proposal

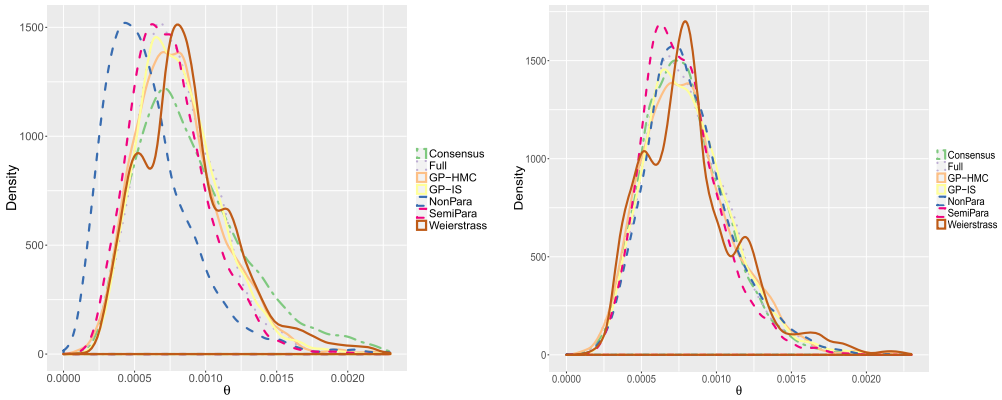

Figure 4: Left: Standard implementation of competing methods to approximate of the full posterior on a Bernoulli model (left). Right: Samples from each merging algorithm are used as a proposal in the distributed importance sampler.

within the GP-IS sampler, we are able to partially recover the shape of the posterior, but not sufficiently well to extract the posterior modes.

5.3 Rare Bernoulli events

In the examples considered above, the subposteriors had approximately the same shape as the full posterior. This is not always the case and is largely dependent on how the data is split. It is possible that the data could be split in such a way some subposteriors are significantly more informative than others. We sample $n = 10{,}000$ Bernoulli random variables, $y_i \sim \text{Bern}(\vartheta)$, and assume a $\text{Beta}(2, 2)$ prior distribution for ϑ. The data is split across $C = 10$ processors. We set $\vartheta = C/n$ so that the probability of observing an event is rare. In fact, each subset only contains one success on average. Furthermore, we repeat this simulation study 100 times, each time randomly re-splitting the original data. By doing this we capture the uncertainty in our discrepancy metrics that result from the data splitting process.

Figure 4 shows the posterior approximation resulting from each of the merging algorithms. Both GP-HMC and GP-IS samplers produce good approximations to the posterior. All of the competing algorithms can reasonably identify the mode of the posterior, but do not adequately fit the tail of the density. This example illustrates the advantage of the GP approximation, which utilises estimates of the log-subposterior density, over simply shifting and re-weighting subposterior samples using only the covariance of the subposteriors, as in the case of the consensus algorithm.

We can generate samples from the full posterior using the distributed importance sampler (Algorithm 1), where samples from each of the merging algorithms can be used as a proposal. Figure 4 (right panel) shows that using the DIS improves the accuracy of all of the competing methods. This improvement is most noticeable for the consensus and

Algorithm	$D_{Mah.}$	$D_{KL}(\pi\|\|\hat{\pi})$	$D_{KL}(\hat{\pi}\|\|\pi)$	ρ	η	Time
Consensus	1.69 (0.59)	0.33 (0.33)	0.33 (0.40)	1.51 (0.55)	0.55 (0.18)	0.07
Nonparametric	1.42 (0.16)	0.40 (0.15)	0.46 (0.25)	1.20 (0.27)	1.00 (0.90)	2.03
Semiparametric	1.12 (0.25)	0.27 (0.42)	0.28 (0.68)	**1.03** (0.28)	0.24 (0.43)	2.80
Weierstrass	1.27 (0.25)	0.20 (0.12)	0.14 (0.14)	1.26 (0.25)	0.14 (0.11)	1.8
GP-HMC sampler	**1.03** (0.06)	**0.09** (0.02)	0.09 (0.02)	1.04 (0.07)	**0.10** (0.08)	13.08
GP-IS sampler	**1.03** (0.05)	**0.09** (0.02)	**0.08** (0.02)	1.04 (0.06)	**0.10** (0.07)	14.22

Table 1: Mean discrepancy of various merge algorithms over 100 simulated rare event Bernoulli models. Kullback–Leibler results are reported as ($\times 10^1$) and $D_{Mah.}$ as ($\times 10^5$). Average execution time is given in seconds. Results in brackets represent standard deviation of metrics over 100 datasets.

nonparametric approximations. Ultimately, the overall accuracy of the approximation to the full posterior will dependent on the quality of the proposal distribution.

Table 1 provides metrics to assess the accuracy of each of the merge algorithms. We report the mean and standard deviation (in brackets) of each metric taken over 100 simulations, where, for each simulation, we re-split the data. On average the GP-HMC and GP-IS samplers display the best performance across all metrics, most notably with regards to Kullback–Leibler divergence. The GP-HMC and GP-IS samplers also have the lowest standard deviation compared to the alternative subposterior merge algorithms. This improvement, however, comes at a higher computational cost than the competing methods. In the Supplementary Material we further investigate the variability of the discrepancy metrics.

5.4 Logistic regression

Synthetic data set We use a synthetic data set based on internet click rate behaviour, where one of the covariates is highly predictive, but rarely observed. The dataset has $n = 10{,}000$, with 5 covariates and is generated according to Section 4.3 of Scott et al. (2016). The data are partitioned across 10 machines. We repeat this experiment 100 times, randomly re-partitioning the original dataset for each experiment.

Additionally to the algorithms discussed at the start of Section 5, we introduce sparse Gaussian process versions of the GP-HMC and GP-IS samplers. We apply the sparse GP presented by Titsias (2009), which uses a variational approach to infer the inducing inputs (see Quiñonero-Candela et al. (2005) for a review of sparse GP approximations).

The posterior distribution for this model is approximately Gaussian and all algorithms perform equally well in this setting (see Table 2). The standard deviation of the discrepancy metrics is generally lower than that of the Bernoulli model (Table 1). For the logistic regression example, there is less variation in the distribution of the data batches over repeated simulations, compared to the Bernoulli example, where there is greater variability from splitting the data. The distributed importance sampler is applied to the posterior approximations with the results given in the Supplementary Material. We also provide additional simulations where the posterior is approximated

Algorithm	$D_{Mah.}$	$D_{KL}(\pi\|\|\hat{\pi})$	$D_{KL}(\hat{\pi}\|\|\pi)$	ρ	η	Time
Consensus	2.36 (0.07)	**0.03** (0.01)	**0.03** (0.01)	0.16 (0.00)	0.09 (0.03)	0.01
Nonparametric	3.67 (0.25)	0.75 (0.14)	2.08 (0.54)	0.16 (0.00)	0.16 (0.06)	0.72
Semiparametric	2.28 (0.16)	0.23 (0.09)	0.17 (0.06)	0.16 (0.00)	0.17 (0.08)	4.23
Weierstrass	2.37 (0.13)	0.11 (0.06)	0.10 (0.04)	0.16 (0.00)	0.17 (0.06)	0.17
GP-HMC	**2.10** (0.71)	0.62 (0.23)	0.67 (0.12)	**0.75** (0.02)	0.16 (0.18)	261
GP-HMC-Sparse	2.69 (0.79)	0.69 (0.26)	0.70 (0.13)	0.71 (0.02)	0.18 (0.17)	94
GP-IS	3.42 (0.08)	1.86 (0.14)	2.85 (0.23)	0.74 (0.00)	**0.05** (0.01)	16.43
GP-IS-Sparse	3.42 (0.08)	1.85 (0.14)	2.86 (0.24)	0.72 (0.00)	**0.05** (0.01)	14.6

Table 2: Mean discrepancy of various merge algorithms over 100 data splits of the logistic regression model with simulated data. Average execution time is given in seconds. Results in brackets represent standard deviation of metrics over 100 data splits.

Algorithm	$D_{Mah.}$	$D_{KL}(\pi\|\|\hat{\pi})$	$D_{KL}(\hat{\pi}\|\|\pi)$	η	Time
Consensus	5.21 (0.05)	**11.55** (0.05)	11.46 (0.03)	0.10 (0.01)	0.03
Nonparametric	9.83 (0.23)	16.57 (0.32)	30.04 (2.00)	0.11 (0.02)	4.15
Semiparametric	5.22 (0.11)	13.67 (0.31)	12.66 (0.14)	0.11 (0.02)	13.36
Weierstrass	5.39 (0.27)	15.83 (0.41)	13.48 (0.15)	0.13 (0.01)	0.74
GP-HMC	5.35 (0.59)	21.78 (0.63)	11.08 (2.24)	0.10 (0.01)	283.56
GP-HMC-Sparse	**5.06** (0.36)	21.79 (0.71)	**10.34** (1.30)	0.10 (0.01)	137.94
GP-IS	5.96 (0.06)	16.76 (0.28)	15.61 (0.18)	**0.09** (0.01)	15.95
GP-IS-Sparse	5.97 (0.08)	16.78 (0.29)	15.69 (0.29)	**0.09** (0.01)	14.25

Table 3: Mean discrepancy of various merge algorithms over 100 data splits of the logistic regression model with the Hepmass dataset. Average execution time is given in seconds. Results in brackets represent standard deviation of metrics over 100 data splits.

with varying sample sizes. We show that it is possible to apply our GP algorithms with fewer samples, giving a reduced computational cost, while maintaining a high level of accuracy.

Real data set We conduct divide-and-conquer MCMC experiments on the Hepmass[6] data set. The challenge is to accurately classify the collisions of exotic particles by separating the particle-producing collisions from the background source. The full data set contains 10.5 millions instances with 28 attributes representing particle features. In our experiments, we use the first million instances and partition the data equally across $C = 20$ machines.

Table 3 gives the mean and standard deviation of the discrepancy metrics for each algorithm taken over 100 simulations (additional plots given in the Supplementary Material). For this example, the subposteriors and full posterior distributions are approximately Gaussian and so all methods approximate the full posterior with more or less the same level of accuracy. As discussed in Neiswanger et al. (2014), nonparametric methods scale poorly with dimension (i.e. number of covariates) with the Weierstrass and semiparametric algorithms performing better than the simple nonparametric method.

[6] https://archive.ics.uci.edu/ml/datasets/HEPMASS.

	$\pi(\vartheta)$	Mean	Median	Quantiles (2.5%, 97.5%)
$\mathbb{E}_{\hat{\pi}}[\vartheta_1]$	0.45	0.45	0.45	$(0.44, 0.46)$
$\mathrm{Var}_{\hat{\pi}}[\vartheta_1]$ $(\times 10^5)$	1.26	1.25	1.24	$(1.21, 1.28)$
$\mathbb{E}_{\hat{\pi}}[\vartheta_{17}]$ $(\times 10^2)$	0.22	0.22	0.21	$(0.17, 0.28)$
$\mathrm{Var}_{\hat{\pi}}[\vartheta_{17}]$ $(\times 10^5)$	7.79	7.75	7.61	$(7.60, 7.89)$

Table 4: Expectation and variance of ϑ_1 and ϑ_{17} from the logistic regression model with the HEPMASS dataset. Mean, median and quantile estimates of the quantities are calculated from 500 samples from the GP-IS sampler (i.e. M = 500).

As a result, applying the DIS step does not lead to a significant improvement in the approximation.

The major difference in the results from Table 3 is the computational time. We see that the GP-IS sampler has comparable computational cost to the nonparametric algorithms, but the GP-HMC samplers have the highest cost overall. It is important to note that, while more expensive than some cheaper competitors, the goal is to produce highly accurate posterior approximations that circumvent applying MCMC to the full dataset. For this example, running an HMC algorithm on the full data takes 19.4 hours. Therefore, relative to this computational cost, applying the GP-HMC sampler accounts for only 0.4% of the total time.

Finally, in Section 4.3, we note that the GP-IS sampler draws multiple realisations from the posterior distribution of the GP approximation to the posterior. Each of these realisations provides an estimate of the expectation of interest, their centre (mean or median) provides a point estimate and their spread (2.5% and 97.5% quantiles) provide a measure of the uncertainty. In Table 4 we estimate the posterior mean and variance of two randomly selected parameters (for ease of presentation) and compare these estimates against those calculated from an MCMC run on the full posterior. Sampling $M = 500$ realisations from the GP, we report the mean, median and 95% interval for estimates of the mean and find that these results are consistent with those of the full applying MCMC on the full data.

6 Discussion

Merging subposteriors generated through parallel, independent MCMC simulations, to form the full posterior distribution is challenging. Currently, available methods either produce a Gaussian approximation to the posterior, or utilise nonparametric estimators which are difficult to tune and do not scale well to high-dimensional settings. In this paper, we have presented an alternative approach to this problem by directly modelling the log-density of the subposteriors. Using Gaussian-process priors, we were able to employ a fully Bayesian strategy towards approximating the full posterior, and unlike competing methods, we were able to account for the uncertainty in the approximation.

Compared to the nonparametric methods, fitting the Gaussian-processes is straightforward using a mixture of marginalisation and maximum likelihood techniques for the hyperparameters. The main drawback of using Gaussian-process approximations is the

computational cost. We have reduced the computational cost by, for each subposterior sample, thinning the Markov chain and removing duplicate MCMC samples prior to fitting the GP. We have shown that using only a small number of samples from the subposterior, we can accurately approximate the full posterior. Furthermore, the computationally intensive step of fitting the individual GPs to the subposteriors is automatically parallelised, as the subposteriors are independent by definition and the GPs are independent by design. While more computationally costly than some competing methods, it is important to note that the cost of fitting, and then sampling from the GP, is significantly cheaper than running an MCMC algorithm on the full data.

The results from Section 5 (and the Supplementary Material) show that in scenarios where both the subposterior and full posterior are approximately Gaussian, the consensus algorithm works well, and is computationally efficient to apply. In settings where either the subposteriors (mixture of Laplace distributions (see the Supplementary Material)) or full posterior (warped Gaussian (Section 5.1) and mixture model (Section 5.2)) are non-Gaussian, our proposed Gaussian process approach is significantly superior to competing methods. This improved performance follows from using the log-subposterior densities to approximate the density of the full posterior, which other algorithms neglect to utilise.

The algorithms we propose scale well with the number of data points n, but fitting a GP when the dimension, d, of ϑ is high, can be computationally expensive as the number of input points required to produce an accurate approximation grows exponentially with d. We have explored the use of sparse GP approximations to reduce the computational burden and have shown that such approximations can be applied in this setting to produce faster algorithms with a similar level of accuracy as the standard GP. This is an ongoing area of research in the Gaussian process community and many alternative sparse GP approximations could be applied, potentially yielding improved results.

Finally, while not the focus of this work, we have numerically explored the effect of randomly partitioning the data. In scenarios where the dataset is heavily imbalanced (e.g. Bernoulli model from Section 5.3), randomly partitioning the data can lead to non-overlapping subposteriors. This issue has not yet been addressed in the literature, and further work investigating ways to efficiently partition the data to ensure a more even distribution of the data across batches is ongoing.

Supplementary Material

Supplement for "Merging MCMC Subposteriors through Gaussian-Process Approximations" (DOI: 10.1214/17-BA1063SUPP; .pdf).

References

Andrieu, C. and Thoms, J. (2008). "A tutorial on adaptive MCMC." *Statistics and Computing*, 18(4): 343–373. MR2461882. doi: https://doi.org/10.1007/s11222-008-9110-y. 510

Bardenet, R., Doucet, A., and Holmes, C. (2014). "Towards scaling up Markov chain Monte Carlo: an adaptive subsampling approach." *Proceedings of The 31st International Conference on Machine Learning*, (4): 405–413. 508

Beskos, A., Pillai, N., Roberts, G., Sanz-Serna, J.-M., and Stuart, A. (2013). "Optimal tuning of the hybrid Monte Carlo algorithm." *Bernoulli*, 19(5A): 1501–1534. MR3129023. doi: https://doi.org/10.3150/12-BEJ414. 511

Carpenter, B., Gelman, A., Hoffman, M., Lee, D., Goodrich, B., Betancourt, M., Brubaker, M. A., Guo, J., Li, P., and Riddell, A. (2016). "Stan: A probabilistic programming language." *Journal of Statistical Software*, 20: 1–37. 511, 521, 522

Chen, T., Fox, E. B., and Guestrin, C. (2014). "Stochastic Gradient Hamiltonian Monte Carlo." In *Proceedings of the 31st International Conference on Machine Learning*, volume 32(2), 1683–1691. 508

Csató, L. and Opper, M. (2002). "Sparse Online Gaussian Processes." *Neural Computation*, 14(2): 641–669. 518

Duvenaud, D. (2014). "Automatic model construction with Gaussian processes." Ph.D. thesis, University of Cambridge. 513

Fairbrother, J., Nemeth, C., and Rischard, M. (2017). "GaussianProcesses.jl: A Nonparametric Bayes package for the Julia Language (preprint)." 512

Geweke, J. (1989). "Bayesian inference in econometric models using Monte Carlo integration." *Econometrica: Journal of the Econometric Society*, 57(6): 1317–1339. MR1035115. doi: https://doi.org/10.2307/1913710. 511

Girolami, M. and Calderhead, B. (2011). "Riemann manifold Langevin and Hamiltonian Monte Carlo methods." *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73(2): 123–214. MR2814492. doi: https://doi.org/10.1111/j.1467-9868.2010.00765.x. 510

Hjort, N. L. and Glad, I. K. (1995). "Nonparametric Density Estimation with a Parametric Start." *The Annals of Statistics*, 23(3): 882–904. MR1345205. doi: https://doi.org/10.1214/aos/1176324627. 519

Hoffman, M. and Gelman, A. (2014). "The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo." *Journal of Machine Learning Research*, 15(2008): 30. MR3214779. 511

Huang, Z. and Gelman, A. (2005). "Sampling for Bayesian Computation with Large Datasets." 508

Liu, H., Lafferty, J., and Wasserman, L. (2007). "Sparse Nonparametric Density Estimation in High Dimensions Using the Rodeo." *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics (AISTATS-07)*, 2: 283–290. MR2387963. doi: https://doi.org/10.1214/009053607000000811. 508

Maclaurin, D. and Adams, R. P. (2014). "Firefly Monte Carlo: Exact MCMC with Subsets of Data." In *Proceedings of the 30th Conference on Uncertainty in Artificial Intelligence*, 543–552. 508

Neal, R. M. (2010). "MCMC Using Hamiltonian Dynamics." In *Handbook of Markov Chain Monte Carlo (Chapman & Hall/CRC Handbooks of Modern Statistical Methods)*, 113–162. MR2858447. 509, 510

Neiswanger, W., Wang, C., and Xing, E. (2014). "Asymptotically Exact, Embarrassingly Parallel MCMC." In *Proceedings of the 30th Conference on Uncertainty in Artificial Intelligence*, 623–632. 508, 510, 513, 516, 518, 519, 522, 525

Nemeth, C. J. and Sherlock, C. (2017). "Supplement for "Merging MCMC Subposteriors through Gaussian-Process Approximations"." *Bayesian Analysis*. doi: https://doi.org/10.1214/17-BA1063SUPP. 510

O'Hagan, A. (1978). "Curve Fitting and Optimal Design for Prediction." *Journal of the Royal Statistical Society, Series B*, 40(1): 1–42. MR0512140. 512

Quiñonero-Candela, J., Rasmussen, C. E., and Herbrich, R. (2005). "A unifying view of sparse approximate Gaussian process regression." *Journal of Machine Learning Research*, 6: 1935–1959. MR2249877. 518, 524

Quiroz, M., Villani, M., and Kohn, R. (2014). "Speeding up MCMC by efficient data subsampling." *arXiv preprint arXiv:1404:4178v1*, (Mcmc): 1–37. 508

Rasmussen, C. and Williams, C. (2006). *Gaussian Processes for Machine Learning*. MIT Press. MR2514435. 512

Robert, C. and Casella, G. (1999). *Monte Carlo Statistical Methods*. Springer-Verlag, New York, Inc. MR1707311. doi: https://doi.org/10.1007/978-1-4757-3071-5. 511

Roberts, G. O., Gelman, A., and Gilks, W. (1997). "Weak Convergence and Optimal Scaling of the Random Walk Metropolis Algorithms." *The Annals of Applied Probability*, 7(1): 110–120. MR1428751. doi: https://doi.org/10.1214/aoap/1034625254. 510

Roberts, G. O. and Rosenthal, J. S. (1998). "Optimal scaling of discrete approximations to Langevin diffusions." *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 60(1): 255–268. MR1625691. doi: https://doi.org/10.1111/1467-9868.00123. 510

Scott, S. L., Blocker, A. W., Bonassi, F. V., Chipman, H. A., George, E. I., and McCulloch, R. E. (2016). "Bayes and big data: The consensus Monte Carlo algorithm." *International Journal of Management Science and Engineering Management*, 11(2): 78–88. 508, 513, 516, 517, 518, 519, 524

Seeger, M., Williams, C., and Lawrence, N. (2003). "Fast forward selection to speed up sparse Gaussian process regression." In *9th International Workshop on Artificial Intelligence and Statistics.*, 2003. 518

Snelson, E. and Ghahramani, Z. (2006). "Sparse Gaussian processes using pseudo-inputs." *Neural Information Processing Systems 18*. 518

Tierney, L. (1996). "Introduction to general state-space Markov chain theory." In Gilks, W., Richardson, S., and Spiegelhalter, D. (eds.), *Markov Chain Monte Carlo in Practice*, 59–74. New York: Chapman and Hall. MR1397968. 515

Titsias, M. K. (2009). "Variational learning of inducing variables in sparse Gaussian processes." In *International Conference on Artificial Intelligence and Statistics*, 567–574. 524

Wang, X. and Dunson, D. B. (2013). "Parallelizing MCMC via Weierstrass Sampler." *Arxiv preprint arXiv:1312.4605*. 513, 516, 518, 519, 520, 522

Wang, X., Guo, F., Heller, K. A., and Dunson, D. B. (2015). "Parallelizing MCMC with random partition trees." In *Advances in Neural Information Processing Systems*, 451–459. 518, 520

Wang, Z., Mohamed, S., and Freitas, N. (2013). "Adaptive Hamiltonian and Riemann manifold Monte Carlo samplers." In *International Conference on Machine Learning*, 1462–1470. 511

Welling, M. and Teh, Y. W. (2011). "Bayesian learning via stochastic gradient Langevin dynamics." In *Proceedings of the 28th International Conference on Machine Learning (ICML)*, 681–688. MR3157685. doi: https://doi.org/10.4310/CIS.2012.v12.n3.a3. 508

Whye Teh, Y., Thiéry, A., and Vollmer, S. (2016). "Consistency and fluctuations for stochastic gradient Langevin dynamics." *Journal of Machine Learning Research*, 17(7): 1–33. MR3482927. 508

Wilkinson, D. J. (2005). "Parallel Bayesian Computation." *Handbook of Parallel Computing and Statistics*, 477–509. MR2282989. doi: https://doi.org/10.1201/9781420028683.ch16. 508

Acknowledgments

We would like to thank Prof. Paul Fearnhead for helpful discussions. The first author gratefully acknowledges the support of the EPSRC funded EP/H023151/1 STOR-i centre for doctoral training.