Contributions to Divide-and-Conquer MCMC

Callum John Vyner

February 20, 2023

Declaration

This thesis has not been submitted, either in whole or in part, for a degree at this, or any other, university. I declare that the work presented in this thesis is, to the best of my knowledge and belief, original and my own work. Listed below is a description of the status of each paper forming the main corpus of this thesis.

The work in Chapter 3 is being prepared to be submitted for publication with Christopher Nemeth and Christopher Sherlock. My contributions include development of the methodology and undertaking the experiments.

Chapter 4 has been submitted to the journal Stat for publication and is currently under review. This is joint work with Christopher Nemeth and Christopher Sherlock. My contributions include partial formation of the idea, development of the methodology, and undertaking the experiments.

Acknowledgements

Firstly a massive thank you to both my supervisors, Professor Christopher Nemeth and Professor Christopher Sherlock. I could not have asked for better. They were both incredibly knowledgeable, supportive, and enthusiastic about the subject matter. I truly believe they were the best supervisors I could have had.

I would like to thank the EPSRC for funding my PhD, and Lancaster University for awarding it to me.

Finally thank you to Jessica Kendall for all the support throughout.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

When applying modern Bayesian methods you can run into issues evaluating expectations when fitting certain models to data sets due to computational constraints. These could be time restrictions, hardware restrictions, or constraints on data sharing that prevent the use of standard statistical techniques. One strategy to overcome these restrictions is to make use of parallelisation. Divide-and-conquer strategies focus on developing techniques to allow Monte Carlo to be performed in a parallel fashion by partitioning the data. Typically, once the data has been partitioned, standard sampling methods can be applied to each partition. The main focus of the divide-and-conquer literature has been to combine these samples generated from each partition to estimate expectations of interest, losing as little accuracy as possible from partitioning the data. In this Thesis we introduce two new methods in the 'Divide-and-Conquer' framework.

This thesis comprises of the following

Chapter 2 - Background. In this section we introduce standard Monte Carlo and give a full introduction to the divide-and-conquer framework. We cover expectations and how Monte Carlo methods can be used to approximate them. We then introduce the divide-and-conquer framework, present a discussion on when you might need these methods, and review some popular methods in the field.

Chapter 3 - Marginal Views (MarV). Marginal views is motivated by the idea that often in statistical problems interest lies in low-dimensional summaries of the parameter vector, such as individual components or one-dimensional functions of the full vector. MarV provides a methodology that allows the use of kernel density estimates for a low-dimensional summary of particular interest whilst taking dependence between the summary and the rest of the parameter vector into account.

Chapter 4 - Subposteriors with Inflation, Scaling, and Shifting (SwISS). SwISS was developed mainly as a competitor to a popular method in the divide-and-conquer framework: Consensus Monte Carlo. Consensus Monte Carlo is easy to implement and can be applied to a number of models. However, Consensus Monte Carlo can struggle when posterior distributions exhibit certain behaviours, such as non-Gaussian, or multi-modal posterior distributions. SwISS is easy to apply and often performs better or as well as Consensus, but we show it can be used successfully with a wider range of models.

Chapter 5 - Further Work. In this chapter, a brief overview of ideas for future work is presented. The first is an improvement to SwISS to correct for bias in the first moment. The second is a post combination correction step. Finally we outline some issues that arise when the data on the partitions are dependent, and a potential area to explore further.

Chapter 2

Background

2.1 Monte Carlo Methods

Many statistical problems require evaluating, or estimating, expectations. Let $\boldsymbol{m} : \chi \to \mathbb{R}^d$ be a measurable function, where χ is a measurable space and $d \in \mathbb{N}$. Let $\boldsymbol{\vartheta}$ be a random variable with probability density function π, also known as the target distribution. Often we are interested in

$$I := \mathbb{E}_\pi \left[\boldsymbol{m}(\boldsymbol{\vartheta}) \right] = \int \pi(\boldsymbol{\vartheta}) \boldsymbol{m}(\boldsymbol{\vartheta}) \mathrm{d}\boldsymbol{\vartheta}. \tag{2.1}$$

If the integral in Equation 2.1 is tractable it can be evaluated analytically. However, often in practice these expectations are not tractable. Monte Carlo methods provide a way to use samples drawn from the target distribution to estimate the integral in

Equation 2.1.

Assume we have simulated independent identically distributed (iid) samples, $\left\{\boldsymbol{\vartheta}^{(i)}\right\}_{i=1}^{N}$ from π. Define the following as an estimator of the expectation in Equation 2.1:

$$\hat{I} := \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{m}(\boldsymbol{\vartheta}^{(i)}). \tag{2.2}$$

Using the law of large numbers it is possible to show that $\hat{I}$ (Equation 2.2) is a consistent estimator for I (Robert and Casella, 2005). Due to the large number of sampling algorithms, many of them easily and generally applicable, Monte Carlo integration is a popular method. For the remainder of this section we will cover sampling schemes relevant to this thesis.

2.1.1 Markov Chain Monte Carlo - MCMC

Markov Chain Monte Carlo (MCMC) is a group of methods that can be used to approximate integrals even when you cannot sample from the target distribution directly. In this section we will describe how, when given a proposal distribution, a sampler can be used to form a Markov chain, which has the target distribution as the stationary distribution of the Markov Chain. We define a Markov chain as a group of samples, $\{\theta^{(i)}\}_{i=1}^{N}$, where the i-th sample is dependent only on the previous sample, *i.e.*,

$$\mathrm{P}(\vartheta^{(i)}|\vartheta^{(i-1)}) = \mathrm{P}(\vartheta^{(i)}|\vartheta^{(i-1)}, \ldots, \vartheta^{(1)}). \tag{2.3}$$

Markov chain samples are not independent, however they can still be used to produce consistent estimators of expectations as in Equation 2.2.

Metropolis-Hastings Algorithm

Metropolis et al. (1953) and Hastings (1970) developed the Metropolis-Hastings algorithm. This algorithm samples $\boldsymbol{\vartheta}^*$ from a proposal distribution, denoted $q(\cdot|\boldsymbol{\vartheta})$, then accepts or rejects the proposed sample according to the an accept-reject step. This accept-reject step is formulated in such a way that the samples generated are approximately from π. This is repeated until the desired number of samples is reached.

Assume $\boldsymbol{\vartheta}$ is the current state of the Markov chain. We sample $\boldsymbol{\vartheta}^* \sim q(\cdot|\boldsymbol{\vartheta})$, a proposed sample. We then calculate the acceptance probability:

$$\alpha\left(\boldsymbol{\vartheta}, \boldsymbol{\vartheta}^*\right) := \min\left\{1, \frac{\pi(\boldsymbol{\vartheta}^*)q(\boldsymbol{\vartheta}|\boldsymbol{\vartheta}^*)}{\pi(\boldsymbol{\vartheta})q(\boldsymbol{\vartheta}^*|\boldsymbol{\vartheta})}\right\}, \tag{2.4}$$

known as the Metropolis-Hastings acceptance rate. $\boldsymbol{\vartheta}^*$ is accepted with probability $\alpha\left(\boldsymbol{\vartheta}, \boldsymbol{\vartheta}^*\right)$. If $\boldsymbol{\vartheta}^*$ is rejected, the state of the Markov chain is unchanged. Algorithm 1

Algorithm 1 Metropolis-Hastings algorithm

```
procedure INPUT: $\boldsymbol{\vartheta}^{(1)}$
    $\boldsymbol{\vartheta}_c \leftarrow \boldsymbol{\vartheta}^{(1)}$
    for $i \in \{1, \ldots, N\}$ do
        $\boldsymbol{\vartheta}^* \sim q(\cdot|\boldsymbol{\vartheta}_c)$
        $\alpha = \alpha\left(\boldsymbol{\vartheta}_c, \boldsymbol{\vartheta}^*\right)$ (Equation 2.4)
        u $\sim$ Unif(0,1)
        if $\alpha > u$ then
            $\boldsymbol{\vartheta}^{(i)} = \boldsymbol{\vartheta}^*$
        else
            $\boldsymbol{\vartheta}^{(i)} = \boldsymbol{\vartheta}_c$
        end if
        $\boldsymbol{\vartheta}_c = \boldsymbol{\vartheta}^{(i)}$
    end for
    RETURN: $\left\{\boldsymbol{\vartheta}^{(i)}\right\}_{i=1}^{N}$
end procedure
```

gives pseudo-code for the general MCMC algorithm.

The choice of proposal distributions for MCMC methods is the focus of many research papers,*e.g.* MALA Roberts and Tweedie (1996), HMC Neal et al. (2011), and Gibbs Carter and Kohn (1994). Here we will only introduce the proposal distributions relevant for the thesis as needed.

2.1.2 Importance Sampling

Importance sampling is similar to MCMC methods, evaluating expectations with respect to a target distribution by simulating from a proposal distribution. However you do not have to form a Markov chain. Typically it is used for lower dimensional problems, and can struggle in higher dimensions.

We reintroduce a proposal distribution with density denoted q with the same support as π. Now we can produce a new estimator for I:

$$\begin{aligned} I &= \mathbb{E}_{\pi}\left[\boldsymbol{m}(\boldsymbol{\vartheta})\right], \\ &= \int_{\boldsymbol{\vartheta}} q(\boldsymbol{\vartheta}) \frac{\pi(\boldsymbol{\vartheta})}{q(\boldsymbol{\vartheta})} \boldsymbol{m}(\boldsymbol{\vartheta}) \mathrm{d}\boldsymbol{\vartheta}, \\ &= \mathbb{E}_{q}\left[\frac{\pi(\boldsymbol{\vartheta})}{q(\boldsymbol{\vartheta})} \boldsymbol{m}(\boldsymbol{\vartheta})\right], \\ &\approx \frac{1}{M} \sum_{j=1}^{M} \frac{\pi(\tilde{\boldsymbol{\vartheta}}^{(j)})}{q(\tilde{\boldsymbol{\vartheta}}^{(j)})} \boldsymbol{m}(\tilde{\boldsymbol{\vartheta}}^{(j)}), \end{aligned} \tag{2.5}$$

where $\left\{\tilde{\boldsymbol{\vartheta}}^{(j)}\right\}_{j=1}^{M}$ are independent samples from a random variable with density $q(.)$. We define $w(\boldsymbol{\vartheta}^{(j)}) = \pi(\boldsymbol{\vartheta}^{(j)})/q(\boldsymbol{\vartheta}^{(j)})$ to be our importance weight for sample j.

Importance sampling is unbiased for all choices of proposal q that have the same support as π. However the choice of q affects the variance of the estimator of I. We want to find the proposal that minimises the variance of the estimator, denoted q^*. Define Γ as the set of all distributions with support equal to that of π and define

$$q^*(\boldsymbol{\vartheta}) := \underset{q(\boldsymbol{\vartheta}) \in \Gamma(\boldsymbol{\vartheta})}{\arg\min} \operatorname{Var}\left[\frac{\pi(\boldsymbol{\vartheta}) \boldsymbol{m}(\boldsymbol{\vartheta})}{q(\boldsymbol{\vartheta})}\right]. \tag{2.6}$$

Robert and Casella (2005) (Theorem 3.12) showed that

$$q^*(\boldsymbol{\vartheta}) = \frac{|\boldsymbol{m}(\boldsymbol{\vartheta})| \pi(\boldsymbol{\vartheta})}{\int \pi(\boldsymbol{\vartheta}^*) \boldsymbol{m}(\boldsymbol{\vartheta}^*) \mathrm{d}\boldsymbol{\vartheta}^*}, \tag{2.7}$$

satisfies Equation 2.6. This is not usable directly since the denominator in Equation 2.7 is the exact integral we wish to evaluate. However, it gives an indication of what a good proposal should be, Robert and Casella (2005) suggested that we choose q such that $|\boldsymbol{m}|\pi/q$ is nearly constant. Hence a good choice for q is a Student-t approximation to π, with a low number of degrees of freedom *e.g.*, df $= 5$ with first two moments equal to those estimated for π. A Student-t is suggest as the distribution is heavy tailed, hence $|\boldsymbol{m}|\pi/q$ will be less likely to extremely large in the tails.

Importance sampling is useful for expectations with a small parameter space, but can struggle in higher dimensions (Au and Beck, 2003). It is usually faster than MCMC algorithms, and can be parallelised, but can produce undesirable high-variance estimators when a large number parameters are present.

Quasi-Importance sampling

Quasi-importance sampling can be used in place of standard importance sampling to increase the convergence rate of our estimator for I. Asmussen and Glynn (2007); Section 3 shows standard Monte Carlo integration has a convergence rate of $\mathcal{O}\left(1/\sqrt{M}\right)$ whereas, under regularity conditions, Quasi-Monte Carlo integration has a convergence rate of $\mathcal{O}\left(\log(M)^d/M\right)$.

Here we will outline how to implement Quasi-importance sampling when the

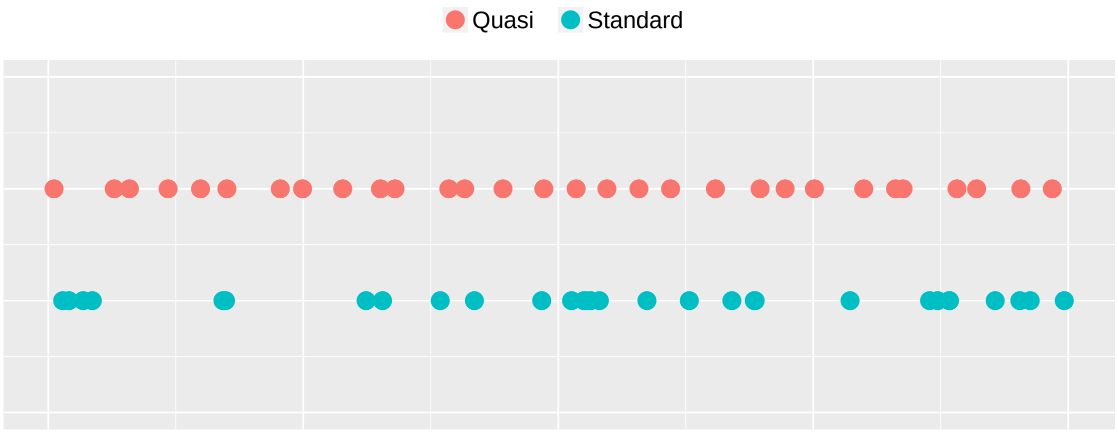


Figure 2.1: A comparison of Quasi sampling compared to standard sampling for 30 samples from a uniform distribution.

target is one-dimensional. Assume our distribution q has an invertible cumulative distribution function Q. Instead of sampling $\left\{\tilde{\boldsymbol{\vartheta}}^{(j)}\right\}_{j=1}^{M}$ independently, from Q we sample evenly between the quantiles of a random variable with density q, that is:

$$\tilde{\boldsymbol{\theta}}^{(j)} = Q^{-1}\left(\frac{j-u_j}{M}\right),$$

where $\{u_j\}_{j=1}^{M}$ are independently drawn from a (0,1) uniform distribution. Intuitively it would be expected that quasi-importance sampling should give a better spread of samples, which can lead to lower variance estimates when compared to standard importance sampling. The difference in spread is demonstrated in Figure 2.1 with a uniform distribution.

Randomised versions of Quasi Monte Carlo sampling, such as the one described above, still produces unbiased estimates of expectations, as shown in Asmussen and

Glynn (2007).

2.2 Kernel Density Estimation (KDE)

Kernel Density Estimation (KDE) methods are used to produce non-parametric estimators for target densities using samples drawn from that target density. As with a lot of non-parametric methods they are most useful in estimating low-dimensional distributions, when little is known or assumed about the parametric form of the target density.

Here we define a kernel to be a function $\mathcal{K}(.) : \mathbb{R}^d \to \mathbb{R}_0^+$ that satisfies:

$$\begin{aligned} \operatorname*{arg\,min}_{\boldsymbol{x} \in \mathbb{R}^d} \ \mathcal{K}(\boldsymbol{x}|\rho) &= \mathbf{0}, \\ \int_{\mathbb{R}} \mathcal{K}(\boldsymbol{x}|\rho) \mathrm{d}\mathbf{x} &= 1, \\ \mathcal{K}(\boldsymbol{x}|\rho) &= \mathcal{K}(-\boldsymbol{x}|\boldsymbol{\rho}), \end{aligned}$$

where ρ is the tuning parameter of the kernel, which typically dictates how fast the kernel tends to zero.

Let $\left\{\boldsymbol{\vartheta}^{(i)}\right\}_{i=1}^{N}$ be samples drawn from a random variable with target density π. We can approximate π using KDE methods:

$$\pi\left(\boldsymbol{\vartheta}\right) \approx \frac{1}{N} \sum_{i=1}^{N} \mathcal{K}\left(\boldsymbol{\vartheta} - \boldsymbol{\vartheta}^{(i)}|\rho\right). \tag{2.8}$$

This centres a kernel around each sample, then for a given value of $\boldsymbol{\vartheta}$ returns the mean of each of the kernels.

Choices of Kernels

There are many choices of kernel functions such as Gaussian, quadratic, uniform etc. The most popular choice of kernel is a Gaussian kernel which has the form:

$$\mathcal{K}^G(\boldsymbol{x}|\rho) = \mathcal{K}^G(\boldsymbol{x}) = \frac{1}{\rho^d\sqrt{2\pi}} \exp\left(-\frac{1}{2\rho^2}\boldsymbol{x}^\top\boldsymbol{x}\right),$$

for $\rho > 0$. Here we have shown ρ to be a scalar, although this can be extended to include a matrix ρ for a more generalisable approach, this is sufficient for this thesis.

The choice of bandwidth can largely affect the quality of the KDE; the larger ρ is, the more weight is given to close samples, a smaller ρ means the kernel function decays slower, giving a more even weight to samples. Figure 2.2 shows three kernel density estimates. The issue with having a bandwidth too small can lead to an un-smooth estimate, whereas a bandwidth too large can lead to a density that has heavier tails than the target density.

If using the Gaussian kernel a wide range of methods exist to choose the bandwidth. The Sheather-Jones bandwidth (Sheather and Jones, 1991) looks to minimise integrated squared error and is of $\mathcal{O}\left(N^2\right)$ in terms of computational cost. Sheather-Jones is generally the preferred bandwidth selector and can produce desirable results

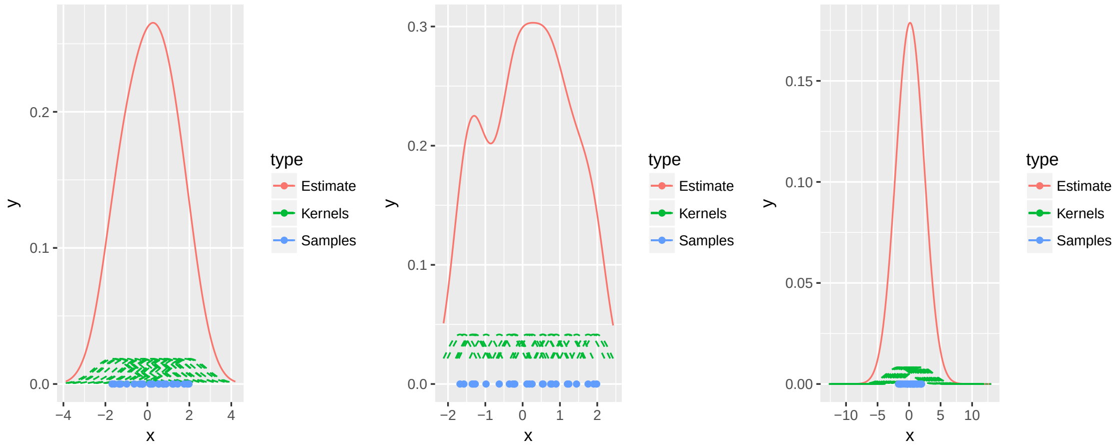


Figure 2.2: Figure shows what effect the choice of bandwidth can have on the KDE for a target distribution. From left to right the figures show: an appropriately tuned bandwidth; an un-smooth estimate of the target distribution, which can happen if the selected bandwidth is too small; the estimate of the target distribution is too heavy tailed, as a result of the selected bandwidth being too large. Note the scales on the x-axes are different. The y-axis is the estimated density.

in a multitude of examples. Silverman's rule of thumb, defined as:

$$\hat{\rho} = \left(\frac{4 \times \mathrm{sd}\left(\boldsymbol{\vartheta}\right)^5}{3N} \right)^{1/5},$$

has a computational cost of $\mathcal{O}\left(N\right)$. It is optimal in terms of mean integrated squared error if π is truly Gaussian, but can be undesirable in largely none-Gaussian cases (Silverman, 1986).

2.3 Divide-and-Conquer Methods

When dealing with very large data a lot of the algorithms introduced in the previous sections need to be altered due to computational limits. One way statisticians can adapt is to develop methods to run in parallel. An issue that presents itself when parallelising methods is how to pool inferences about a set of model parameters when those inferences have been drawn from distributed sources. Inference may need to be distributed due to the following reasons:

- **Computational** - Due to time constraints, it may not be feasible to wait for inference to be drawn. This could be due to a high number of observations making up the data set, and therefore data may need to be split to be analysed in time to meet deadlines.
- **Memory** - A memory bottle neck exists when the data and the processes needed to produce inferences require more random-access memory (RAM) than is available.
- **Disk** - A hardware limitation that arises when the size of data is larger than the total available storage.
- **Privacy** - There are several data centres and sharing the raw data might be restricted due to privacy concerns, for example with medical records. In these cases, centres may be restricted to sharing top level information, such as summary statistics, or Monte Carlo samples drawn from a target distribution.

The computational, memory, and disk bottle necks are often a result of '*big data*' which, similar to Scott et al. (2016), will be defined here to be data that cannot be comfortably processed on a single machine. As in Bardenet et al. (2017), we will define '*tall data*' to be data that are considered to be big because of the large number of observations, and will define '*wide data*' to be data that are considered to be big because of the large number of parameters. Data can be both tall and wide.

Monte Carlo methods (Section 2.1.1) can be applied to a wide variety of target distributions with ease, are often asymptotically exact, and there exists numerous software packages to aid in implementation (Stan Development Team (2016), Spiegelhalter et al. (2003)). For these reasons these methods are often used in Bayesian inference. However they can often be impracticable in big-data settings, due to the reasons listed above. When using MCMC methods the target distribution needs to be evaluated at the proposed value for every sample drawn. Without doing this, it is impossible to calculate the acceptance probability:

$$\alpha(\boldsymbol{\vartheta}, \boldsymbol{\vartheta}^*) = \min \left\{ 1, \frac{\pi(\boldsymbol{\vartheta}^*) q(\boldsymbol{\vartheta} | \boldsymbol{\vartheta}^*)}{\pi(\boldsymbol{\vartheta}) q(\boldsymbol{\vartheta}^* | \boldsymbol{\vartheta})} \right\}.$$

Coupled with big data this can introduce a computational bottle neck. Additionally, if the data cannot fit on RAM or disk, the target distribution cannot be (easily) calculated.

A natural approach from here would be to split the data across multiple servers and perform inference seperately on these batches. For this thesis we will assume the inference drawn is in the form of samples drawn from target distributions on each server. *Divide-and-conquer* (Neiswanger et al. (2013), Scott et al. (2016), Li et al. (2017), Nemeth and Sherlock (2018)), also known as *fork-and-merge*, methods describe how to unify these samples.

2.4 Divide-and-Conquer Framework

We will introduce the general divide-and-conquer framework. Some methods in this field deviate from this framework in places, but for ease of reading we will explain the most common structure here, and cover any discrepancies when reviewing the individual methods.

We will assume our target distribution is a posterior distribution of the following form:

$$\pi\left(\boldsymbol{\vartheta}|\mathbf{y}\right) \propto f\left(\mathbf{y}|\boldsymbol{\vartheta}\right)\pi(\boldsymbol{\vartheta}),$$

where f denotes the likelihood, $\mathbf{y}$ is our data, $\boldsymbol{\vartheta}$ are the parameters governing the behaviour of the model and $\pi(\boldsymbol{\vartheta})$ describes our prior belief in the parameters.

If we can split our data into disjoint, independent partitions then we can write $\mathbf{y} = (\mathbf{y}_1, \dots, \mathbf{y}_B)$, where $\mathbf{y}_b$ denotes the b-th independent batch up to B, our total number of batches. We can write the posterior distribution in the following way:

$$\pi(\boldsymbol{\vartheta}|\mathbf{y}) \propto \pi(\boldsymbol{\vartheta})f(\mathbf{y}|\boldsymbol{\vartheta}) = \prod_{b=1}^{B} \pi(\boldsymbol{\vartheta})^{1/B} f(\mathbf{y}_b|\boldsymbol{\vartheta}) \propto \prod_{b=1}^{B} \pi_b(\boldsymbol{\vartheta}|\mathbf{y}), \tag{2.9}$$

where we define

$$\pi_b(\boldsymbol{\vartheta}|\mathbf{y}) := \frac{\pi(\boldsymbol{\vartheta})^{1/B} f(\mathbf{y}_b|\boldsymbol{\vartheta})}{m(\mathbf{y}_b)}$$

to be our b-th *subposterior* distribution for each $b \in \{1, \ldots, B\}$.

Inference can be drawn in parallel in the form of N samples from each subposterior distribution. We define $\boldsymbol{\vartheta}_b := \left\{\boldsymbol{\vartheta}_b^{(i)}\right\}_{i=1}^{N}$ to be the N samples drawn from $\boldsymbol{\vartheta}|\mathbf{y}_b$ using a chosen Monte Carlo method. N could vary across batches, *i.e.*, we could draw a different number of samples from each subposterior distribution, but for ease of notation we assume we draw N samples from each batch.

The samples can be thought of as summaries from each of the subposterior distributions. However we are not interested in inference or expectations about the subposterior distributions but in the full-posterior distribution instead. So the challenge lies in finding a mapping, $\Upsilon : \chi^B \to \chi$, from the samples from each subposterior distribution to samples from an approximation to the full-posterior distribution:

$$\tilde{\boldsymbol{\vartheta}} := \Upsilon\left(\boldsymbol{\vartheta}_1, \ldots, \boldsymbol{\vartheta}_B\right).$$

At this point $\tilde{\boldsymbol{\vartheta}}$ can be used to draw inference about $\boldsymbol{\vartheta}|\mathbf{y}$, *i.e.*, we can now answer questions about the system using the full data, by using $\tilde{\boldsymbol{\vartheta}}$ and Monte Carlo integration:

$$\mathbb{E}_{\pi(\boldsymbol{\vartheta}|\mathbf{y})}\left[\boldsymbol{m}(\boldsymbol{\vartheta})\right] \approx \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{m}(\tilde{\boldsymbol{\vartheta}}).$$

2.4.1 Inflated-Subposterior Distributions

Instead of sampling from the subposterior distributions, some methods (Srivastava et al. (2018), Li et al. (2017), Entezari et al. (2018)) inflate the subposterior distributions before sampling from them. We define the inflated-subposterior distribution for a given batch b as:

$$\tilde{\pi}_b(\boldsymbol{\vartheta}|\mathbf{y}_b) := \pi_b(\boldsymbol{\vartheta}|\mathbf{y}_b)^B = \frac{\pi(\boldsymbol{\vartheta})f(\mathbf{y}_b|\boldsymbol{\vartheta})^B}{m(\mathbf{y}_b)^B}. \tag{2.10}$$

Note that the prior is not split in this case; this can be beneficial when sampling where the prior is important for regularisation.

Sometimes the inflated-subposterior distribution is called a stochastic approximation to the full-posterior distribution, $\pi(\boldsymbol{\vartheta}|\mathbf{y})$. Inflating the subposterior distribution is done so that the inflated posterior approximates the full-data posterior distribution, *i.e.*,

$$\tilde{\pi}(\boldsymbol{\vartheta}|\mathbf{y}_b) \approx \pi(\boldsymbol{\vartheta}|\mathbf{y}).$$

If each of the subposterior distributions are receiving the same amount of data then each of the inflated-subposterior distributions will have roughly equal variance to the full-posterior distribution, a visual representation of this is shown in Figure 2.3. Although these methods draw samples from the inflated-subposterior distributions, they still look to map the batch samples to a set of samples from an approximation to the full-posterior to perform Monte Carlo integration.

2.5 Existing Methods

In this section we will be outlining existing methods divide-and-conquer literature.

2.5.1 The Consensus Monte Carlo Algorithm

Scott et al. (2016) introduced the Consensus Monte Carlo algorithm, an algorithm that combines the samples from subposterior distributions through averaging.

The Consensus Monte Carlo algorithm was introduced with a general weighting, however, Scott et al. (2016) suggested, when averaging, to weight samples from each of the subposterior distributions according to the inverse variance of that subposterior distribution. This ensures that in the case that all the subposterior distributions are Gaussian, the inverse-variance weighting produces $\bar{\boldsymbol{\vartheta}}$ that are drawn exactly from the full-posterior distribution, up to some Monte Carlo error. This Monte Carlo error arises due to having to estimate the variance for each subposterior distribution from the subposterior samples.

Let $\hat{V}_b$ be the estimated variance matrix for the b-th subposterior distribution, which is estimated from the subposterior distribution samples. Then we define the Consensus Monte Carlo algorithm samples as follows:

$$\bar{\boldsymbol{\vartheta}} := \frac{1}{B}\left(\sum_{b=1}^{B}\hat{V}_b^{-1}\right)^{-1}\sum_{b=1}^{B}\hat{V}_b^{-1}\boldsymbol{\vartheta}_b. \tag{2.11}$$

Figure 2.4 shows a visual representation of the Consensus Monte Carlo algorithm. The dashed black and blue lines represent the subposteior densities and approximate posterior density. The points are averaged to obtain a point from the approximate posterior distribution.

Consensus Monte Carlo is popular since Equation 2.11 is easily applied. Scott et al. (2016) show the Consensus Monte Carlo algorithm can be applied in a variety of models: classification models such as Logistic regression, it can be used when subposterior distributions have differing variances, and in mixed effect models. However, the samples from the approximation to the full-posterior distribution can be unrepresentative of the full-posterior distribution. This can occur if the posterior or subposterior distributions are multi-modal, or if the the posterior is overly skewed.

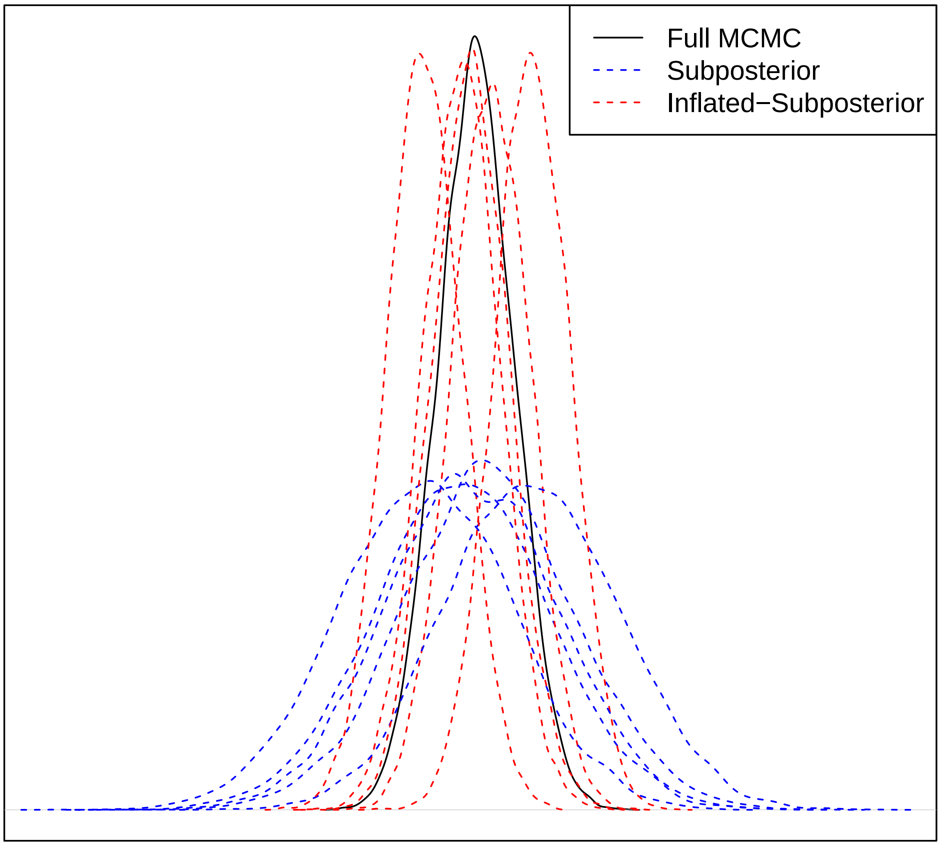


Figure 2.3: Example to show the difference in shape between a inflated-subposterior densities and subposterior densities. The density of the full-data posterior is given for reference.

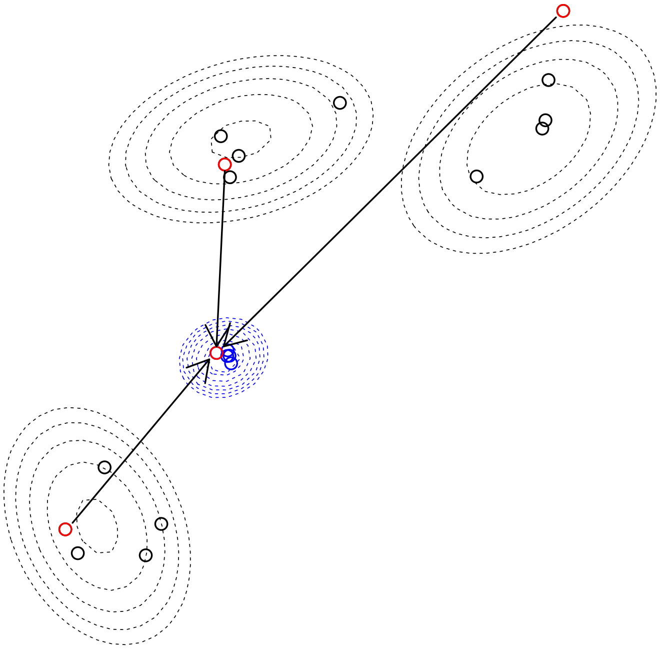


Figure 2.4: A 2-dimensional visual representation of the Consensus Monte Carlo algorithm. A sample from each of the subposterior distributions is chosen and then they are averaged to generate a consensus sample. This sample is from an approximation to the full-posterior distribution.

2.5.2 Asymptotically exact, embarrassingly parallel MCMC - Non-parametric and semi-parametric density estimates

Neiswanger et al. (2013) introduced an asymptotically exact, embarrassingly parallel MCMC algorithm. In this paper three merging methods were proposed:

- A full-Gaussian approximation to each subposterior distribution;
- A non-parametric kernel density estimator approach;
- A semi-parametric approach, which initially assumes a Gaussian approximation to the subposterior distributions, but then tends to a full kernel density approach as the number of subposterior distribution samples tends to infinity.

Gaussian Approximation

The Gaussian approximation was not the focus of the paper. The Gaussian approximation uses the samples from each subposterior distribution to estimate the mean and variance of the full-posterior distribution, assuming each subposterior distribution was Gaussian. Let $\hat{\mu}_b$ and $\hat{V}_b$ be Monte Carlo estimates from the subposterior distribution samples for the the mean and variance, respectively. We define the global variance and mean to be

$$\hat{V}^{-1} := \sum_{b=1}^{B} \hat{V}_b^{-1}, \text{ and } \hat{\mu} := \frac{1}{B}\sum_{b=1}^{B} \hat{V}_b^{-1}\hat{\mu}_b,$$

respectively, then:

$$\bar{\boldsymbol{\vartheta}} \sim \mathcal{N}_d(\hat{\mu}, \hat{V}).$$

Non-parametric density estimates

Assume N samples have been drawn from each d-dimensional subposterior distribution, denoted $\left\{\boldsymbol{\vartheta}_b^{(i_b)}\right\}_{i_b=1}^{N}$. The non-parametric approach uses a kernel density estimator to approximate each subposterior distribution. Given a kernel $\mathcal{K} : \mathbb{R}^d \to \mathbb{R}^+$ Neiswanger et al. (2013) defined the following estimator for the b-th subposterior distribution:

$$\hat{\pi}_b(\boldsymbol{\vartheta}|\mathbf{y}_b) = \frac{1}{N}\sum_{i=1}^{N} \mathcal{K}^G\left(\boldsymbol{\vartheta} - \boldsymbol{\vartheta}_b^{(i_b)}\right), \qquad (2.12)$$

where a Gaussian kernel is assumed as described in Section 2.2. Having this estimator for the subposterior distributions leads to the following estimator for the full-posterior distribution, which takes the form of a mixture of Gaussian distributions:

$$\hat{\pi}(\boldsymbol{\theta}|\mathbf{y}) = \frac{1}{N^B}\prod_{b=1}^{B}\sum_{i_b=1}^{N}\mathcal{N}_d\left(\boldsymbol{\vartheta}|\boldsymbol{\vartheta}_b^{(i_b)}, \rho^2 I_d\right),$$
$$= \sum_{i_1=1}^{N}\cdots\sum_{i_B=1}^{N} w_{i_*}\mathcal{N}_d\left(\boldsymbol{\vartheta}\,\middle|\,\tilde{\boldsymbol{\vartheta}}_{i_*}, \frac{\rho^2}{M}I_d\right), \tag{2.13}$$

where $\tilde{\boldsymbol{\vartheta}}_{i_*}$ is the mean of the i-th sample from each batch:

$$\tilde{\boldsymbol{\vartheta}}_{i_*} = \frac{1}{B}\sum_{b=1}^{B}\boldsymbol{\vartheta}_b^{(i_b)},$$

and w_{i_*} are unnormalised mixture weights defined as:

$$w_{i_*} = \prod_{b=1}^{B}\mathcal{N}_d\left(\boldsymbol{\vartheta}_b^{(i_b)}|\tilde{\boldsymbol{\vartheta}}_{i_*}, \rho^2 I_d\right).$$

There are N^B possible mixture components, so Neiswanger et al. (2013) proposed a Gibbs sampler to sample from the distribution in Equation 2.13 to form $\{\bar{\boldsymbol{\vartheta}}\}_{i=1}^{N}$.

Semi-parametric density estimates

The first method proposed by Neiswanger et al. (2013) had quick convergence, but had strict parametric assumptions. The second method was slower to converge, but could be applied to a wider range of models. The final method suggested was a semi-parametric approach. Let $\hat{f}_b(\boldsymbol{\vartheta})$ be a Gaussian approximation to the b-th subposterior distribution:

$$\hat{f}_b(\boldsymbol{\vartheta}) = \mathcal{N}_d\left(\boldsymbol{\vartheta}|\hat{\mu}_b, \hat{V}_b\right).$$

Define a non-parametric correction function $r_b(\boldsymbol{\vartheta}) = \pi_b(\boldsymbol{\vartheta}|\mathbf{y}_b)/f_b(\boldsymbol{\vartheta})$. KDE methods are used to estimate the correction function. When there are a low number of subposterior samples the estimate of the correction function, denoted $\hat{r}_b(\boldsymbol{\vartheta})$, is fairly flat for each batch, and so the approximate subposterior distributions are close to Gaussian. As the number of subposterior samples tend to infinity the subposterior approximations, $\hat{\pi}_b(\boldsymbol{\vartheta}|\mathbf{y}_b) = \hat{r}_b(\boldsymbol{\vartheta})\hat{f}_b(\boldsymbol{\vartheta})$, tend to full-kernel approximations. Again this model is sampled from using a Gibbs sampler, see Neiswanger et al. (2013) for details.

This semi-parametric approach allows the flexibility of KDE methods, but means the approach can still produce low variance estimates in higher dimensional problems.

2.5.3 Wasserstein Posteriors approximations

The following algorithms, Wasserstein posteriors (WASP) (Srivastava et al., 2018) and Posterior Interval Estimation (PIE) (Li et al. (2017)), use the ideas of Wasserstein distance and Wasserstein barycenters to form an approximation to the full-data posterior distribution.

Wasserstein distance and barycenters

Assume $\chi \subseteq \mathbb{R}^d$ and define $||\boldsymbol{\vartheta}_1 - \boldsymbol{\vartheta}_2||$ to be the Euclidean distance between $\boldsymbol{\vartheta}_1, \boldsymbol{\vartheta}_2 \in \chi$. For any two measures ν_1, ν_2 on χ let $\Gamma(\nu_1, \nu_2)$ be the set of all probability measures on $\chi \times \chi$ with marginals ν_1, ν_2. We define the Wasserstein-2 distance between ν_1, ν_2 as

$$W_2(\nu_1, \nu_2) = \left\{ \inf_{\gamma \in \Gamma(\nu_1, \nu_2)} \int_{\chi \times \chi} ||\boldsymbol{\vartheta}_1 - \boldsymbol{\vartheta}_2||^2 \mathrm{d}\gamma(\nu_1, \nu_2) \right\}^{1/2}.$$

We restrict the measures we consider so that $\nu_1, \nu_2 \in \mathcal{P}_2(\chi) = \left\{ \nu : \int_\chi ||\boldsymbol{\vartheta}||^2 \mathrm{d}\nu(\boldsymbol{\vartheta}) < \infty \right\}$, ensuring $W_2(\nu_1, \nu_2)$ is well defined. The Wasserstein-2 distance can be used to compare how *close* two distributions are. If the measures ν_1 and ν_2 are equal, then the Wasserstein distance will be zero. Another way to think of Wasserstein-2 distance is the square-root of the expected squared distance between the first and second coordinate of a sample from γ. γ is chosen to minimise that distance, whilst having specific marginals.

Given a collection of measures $\{\nu_b\}_{b=1}^B$ the Wasserstein barycenter for the measures is defined as

$$\bar{\nu} = \arg\min_{\mu \in \mathcal{P}_2(\chi)} \sum_{b=1}^{B} W_2^2(\mu, \nu_b). \tag{2.14}$$

Wasserstein Posteriors

Assume we have drawn N samples from each inflated-subposterior distribution, $\left\{\boldsymbol{\vartheta}_b^{(i)}\right\}_{i=1}^{N}$ for $b \in \{1, \ldots, B\}$. Let $\tilde{\Pi}_b(\boldsymbol{\vartheta}|\mathbf{y}_b)$ be the distribution function associated with the inflated-subposterior density $\tilde{\pi}_b(\boldsymbol{\vartheta}|\mathbf{y}_b)$ for each $b \in \{1, \ldots, B\}$. Empirical estimates of $\tilde{\Pi}_b(\boldsymbol{\vartheta}|\mathbf{y}_b)$ can be estimated from inflated-subposterior samples $\left\{\boldsymbol{\vartheta}_b^{(i)}\right\}_{i=1}^{N}$. We define the Wasserstein Posterior (WASP) to be the Wasserstein barycenter of $\left\{\tilde{\Pi}(\boldsymbol{\vartheta}|\mathbf{y}_b)\right\}_{b=1}^{B}$ as in Equation 2.14

$$\bar{\Pi}(\boldsymbol{\vartheta}|\mathbf{y}) := \underset{\mu \in \mathcal{P}_2(\chi)}{\arg\min} \sum_{b=1}^{B} W_2^2\left(\mu, \tilde{\Pi}_b(\boldsymbol{\vartheta}|\mathbf{y}_b)\right). \tag{2.15}$$

The Wasserstein Barycenter is the measure that minimises the sum of the Wasserstein distance between a set of measures and itself. Cuturi and Doucet (2014) shows how this can be estimated. WASP suggests using the Wasserstein Barycenter, using the empirical measures obtained from the inflated-subposterior distributions, as an approximation to full-posterior distribution.

Posterior interval estimation

In the one-dimensional case the estimation for WASP is tractable, and a linear program is not needed. Assume $\vartheta \in \mathbb{R}$. Let $F^{-1}(u) = \inf(\vartheta : F(\vartheta \geq u))$ be the quantile function of a generic uniform distribution function $F(\vartheta)$. For $F_1, F_2 \in \mathcal{P}_2(\chi)$, with quantile functions $F_1^{-1}(u)$ and $F^{-1}(u)$, for $u \in (0, 1)$. (Li et al., 2017) showed that the W_2 distance between F_1 and F_2 has the following form:

$$W_2(F_1, F_2) = \left\{ \int_0^1 \left(F_1^{-1}(u) - F_2^{-1}(u) \right)^2 \mathrm{d}u \right\}^{1/2}. \tag{2.16}$$

Therefore, in the one-dimensional case the inverse of the distribution function associated with the full-data posterior is exactly:

$$\bar{\Pi}^{-1}(u|\mathbf{y}) = \frac{1}{B} \sum_{b=1}^{B} \tilde{\Pi}^{-1}(u|\mathbf{y}_b) \tag{2.17}$$

As a corollary of Equation 2.17, given samples from each inflated-subposterior distribution it is simple to estimate the quantile function associated with the full-posterior distribution. Li et al. (2017) called this *posterior interval estimation* (PIE).

Chapter 3

Marginal Views (MarV) - Approximating Marginal Expectations within the Divide-and-Conquer Framework

3.1 Introduction

This chapter presents Marginal Views (MarV), a divide-and-conquer method for estimating marginal posterior expectations. What follows in this section is a brief justification for why this method is needed. Section 3.2 outlines the details of the method and the theoretical justification for it. In Section 3.3 it is shown that MarV performs well against other divide-and-conquer methods at estimating marginal-posterior expectations.

In Section 2.4 we highlighted why divide-and-conquer approaches are needed. Suppose we have a parameter vector $\boldsymbol{\vartheta}$ and data $\mathbf{y}$. When a posterior density $\pi(\boldsymbol{\vartheta}|\mathbf{y})$ is intractable, Monte Carlo methods are often used to produce samples that are from, or approximately from, the target π. These samples are useful in approximating posterior expectations via Monte Carlo integration. When $\mathbf{y}$ contains a large number of observations, N, repeated evaluations of π can be prohibitively slow, or $\mathbf{y}$ may not even fit on a single computer. In these cases, standard Markov Chain Monte Carlo (MCMC) algorithms are infeasible, and we can look to partition data into B batches, denoted $\mathbf{y}_1, \ldots, \mathbf{y}_B$, then sample from the respective subposteriors (Section 2.4), $\pi(\boldsymbol{\vartheta}|\mathbf{y}_1), \ldots \pi(\boldsymbol{\vartheta}|\mathbf{y}_B)$, to produce subposterior samples that will be denoted $\boldsymbol{\vartheta}^{(1)}, \ldots, \boldsymbol{\vartheta}^{(B)}$. A different motivation might be data privacy; different data owners may want to combine the inference from each data set, but they may be unwilling, or unable for legal reasons, to share their data. Using the divide-and-conquer framework, they would be able to share their subposterior samples without sharing data-level information. Regardless of the motivation, the aim is to collect summaries, usually in the from of subposterior samples, onto a central computer, or core, and then use the samples to estimate posterior expectations. In this chapter a method is presented that, instead of focusing on producing samples from an approximation to the full-posterior, focuses on developing estimators of low-dimensional posterior expectations.

Divide-and-conquer methods often focus on generating samples from an approximation to the full-posterior distribution; any posterior expectations of interest are evaluated using those samples. An exception to this is the Posterior Interval Estimation (PIE) method (Li et al. (2017), Section 2.5.3). PIE is a method that works with inflated subposteriors (Section 2.4.1) and by applying this method it can be shown that, when interested in one-dimensional, marginal expectations, taking the mean of the inflated subposterior quantiles gives the quantiles of the associated one-dimensional Wasserstein barycenter. Sampling from this barycenter gives samples from an approximation to a chosen one-dimensional marginal of the full-posterior distribution. Previously Srivastava et al. (2014) had described a method, WASP, that estimated the Wasserstein barycenter (see Section 2.5.3) for an arbitrary dimension. An issue facing this method was the computational complexity of having to solve a linear program. By only considering one-dimensional marginals of the posterior distribution, the computational complexity issues that faced WASP is significantly reduced.

This chapter presents *Marginal Views* (MarV), a method for producing an estimate of a chosen posterior expectation that takes an input of subposterior samples. MarV obtains weighted samples from an approximation to a given marginal-posterior distribution to produce estimates of posterior expectations. Although

$\boldsymbol{\vartheta}$ may be high dimensional, we are motivated by the same heuristic as the PIE method, that individual posterior expectations of interest are typically of a much lower dimension. When interested in low-dimensional expectations the MarV algorithm is robust, computationally feasible, and scalable in both dimension of the full-parameter space and number of observations. Once samples have been drawn from each subposterior, MarV can be repeatedly applied to estimate as many low-dimensional full-posterior expectations as needed.

When the target posterior is Gaussian or close to Gaussian, we find that MarV is competitive with methods that have stricter parametric assumptions, such as the Consensus Monte Carlo algorithm (Scott et al., 2016). However MarV differs, as we illustrate in Section 3.3, in that it can capture non-Gaussian behaviour in the target, for example, behaviours such as multi-modality.

3.2 Methodology

Assume our parameter vector $\boldsymbol{\vartheta} \in \boldsymbol{\chi}_{\boldsymbol{\vartheta}} \subseteq \mathbb{R}^{d_{\boldsymbol{\vartheta}}}$ for $d_{\boldsymbol{\vartheta}} \in \mathbb{N}$. We are interested in evaluating the posterior expectation of $\boldsymbol{\theta} := \boldsymbol{m}_{\boldsymbol{\theta}}(\boldsymbol{\vartheta}) \in \boldsymbol{\chi}_{\boldsymbol{\theta}}$, the output of a vector function, $m_{\boldsymbol{\theta}}$, of our parameter $\boldsymbol{\theta}$; *i.e.*, we would like to estimate I where

$$I := \mathbb{E}_{\boldsymbol{\vartheta}|\mathbf{y}}\left[\boldsymbol{\theta}\right], \tag{3.1}$$

Often in practice $\boldsymbol{\theta}$ will be a subset of or a low-dimensional summary of $\boldsymbol{\vartheta}$. If the dimension of $\boldsymbol{\vartheta}$, is greater than the dimension of $\boldsymbol{\theta}$, then there exists a non-unique formulation for the parameters to describe the dimensions of $\boldsymbol{\vartheta}$ not covered by $\boldsymbol{\theta}$. We define $\boldsymbol{\psi} = \boldsymbol{m}_{\boldsymbol{\psi}}(\boldsymbol{\vartheta}) \in \boldsymbol{\chi}_{\boldsymbol{\psi}}$ to be the completion of $\boldsymbol{\theta}$ in $\boldsymbol{\chi}_{\boldsymbol{\theta}}$, for some vector function $\boldsymbol{m}_{\boldsymbol{\psi}}()$, *i.e.*, there exists a bijection $\boldsymbol{\gamma}$ such that

$$\boldsymbol{\vartheta} = \boldsymbol{\gamma}(\boldsymbol{\theta}, \boldsymbol{\psi}).$$

For example:

- If $\theta = \vartheta_1$, *i.e.*, interest lies in a single parameter, then we could choose $\boldsymbol{\psi} = \boldsymbol{\vartheta}_{-1}$;

- If $\theta = \boldsymbol{x}^\top \boldsymbol{\vartheta}$ for some $1 \times d_{\boldsymbol{\vartheta}}$ column vector $\boldsymbol{x}$, *i.e.*, we are interested in some predictor, then we could choose $\boldsymbol{\psi} = \vartheta_{-d_{\boldsymbol{\vartheta}}}$, provided the final element of $\boldsymbol{x}$ is non-zero.

- If $\boldsymbol{\theta} = (\vartheta_1, \vartheta_3)$, *i.e.*, interest is in a small collection of transformations of the parameters, then we could choose $\boldsymbol{\psi} = (\vartheta_2, \boldsymbol{\vartheta}_{-1:3})$, or even $\boldsymbol{\psi} = (\vartheta_1\vartheta_2, \vartheta_3\vartheta_4, \boldsymbol{\vartheta}_{-1:4})$.

As long as there exits the one-to-one mapping, γ, such that $\gamma(\boldsymbol{\theta}, \boldsymbol{\psi}) = \boldsymbol{\vartheta}$, as above, we are flexible in our choice of $\boldsymbol{\psi}$. For the remainder of this chapter, unless otherwise stated, if $\boldsymbol{\theta} = \boldsymbol{\vartheta}_{\boldsymbol{i}}$ for some $\boldsymbol{i} \in \mathbb{N}^{d_{\boldsymbol{\theta}}}$ for $d_{\boldsymbol{\theta}} < d_{\vartheta}$, we will choose $\boldsymbol{\psi} := \boldsymbol{\vartheta}_{-i}$. In examples where $\boldsymbol{\theta}$ is not a subset of the components of $\boldsymbol{\vartheta}$ we will explicitly state $\boldsymbol{\psi}$.

An initial approach could be to simply ignore $\boldsymbol{\psi}$ and approximate each $\pi(\boldsymbol{\theta}|\mathbf{y}_b)$ with KDE methods using the samples drawn from each subposterior, then combine using the KDE methods described by Neiswanger et al. (2013) to obtain samples from an approximation to $\pi(\boldsymbol{\theta}|\mathbf{y})$. Since the dimension of $\boldsymbol{\theta}$ is low the KDE combination would not suffer from the curse of dimensionality even when the dimension of $\boldsymbol{\vartheta}$ is high. This approach would not be correct in almost all cases; it would be implicitly assuming independence between $\boldsymbol{\psi}|\mathbf{y}_b$ and $\boldsymbol{\theta}|\mathbf{y}_b$ for each batch. Consider the following:

$$\begin{aligned}\pi(\boldsymbol{\theta}|\mathbf{y}) &\propto \int_{\boldsymbol{\psi}} \prod_{b=1}^{B} \left\{\pi_b(\boldsymbol{\theta}|\mathbf{y}_b)\pi_b(\boldsymbol{\psi}|\mathbf{y}_b, \boldsymbol{\theta})\right\} \mathrm{d}\boldsymbol{\psi} \\ &= \prod_{b=1}^{B} \left\{\pi_b(\boldsymbol{\theta}|\mathbf{y}_b)\right\} \int_{\boldsymbol{\psi}} \prod_{b=1}^{B} \left\{\pi_b(\boldsymbol{\psi}|\mathbf{y}_b, \boldsymbol{\theta})\right\} \mathrm{d}\boldsymbol{\psi}, \\ &\not\propto \prod_{b=1}^{B} \left\{\pi_b(\boldsymbol{\theta}|\mathbf{y}_b)\right\}.\end{aligned} \tag{3.2}$$

That is to say, in general the integral of a product does not equal the product of integrals. This shows when combining the samples of $\boldsymbol{\theta}|\mathbf{y}_b$, the dependence structure between $\boldsymbol{\psi}$ and $\boldsymbol{\theta}$ within each batch needs to be accounted for. If $\boldsymbol{\psi}$ and $\boldsymbol{\theta}$ are independent then the dependence structure doesn't need to be accounted for:

$$\begin{aligned}\pi(\boldsymbol{\theta}|\mathbf{y}) &\propto \prod_{b=1}^{B} \left\{\pi_b(\boldsymbol{\theta}|\mathbf{y}_b)\right\} \int_{\boldsymbol{\psi}} \prod_{b=1}^{B} \left\{\pi_b(\boldsymbol{\psi}|\mathbf{y}_b)\right\} \mathrm{d}\boldsymbol{\psi}, \\ &\propto \prod_{b=1}^{B} \left\{\pi_b(\boldsymbol{\theta}|\mathbf{y}_b)\right\}.\end{aligned}$$

Additionally, if each of the $\boldsymbol{\psi}$ components are only relevant on one subposterior, as can be the case in random-effects models where each subject has all their information analysed by only one subposterior, then there is no need to account for dependency. However, in general we can not assume independence, and therefore we develop a method to account for this.

To marginalise out the product of conditionals in Equation 3.2 we need to approximate each batch's subposterior conditional density, $\pi_b(\boldsymbol{\psi}|\boldsymbol{\theta}, \mathbf{y}_b)$, with a density that, when there is a product of these densities, produces a tractable integral over $\boldsymbol{\psi}$ for each value of $\boldsymbol{\theta}$. We suggest making a Gaussian approximation of $\pi_b(\boldsymbol{\psi}|\boldsymbol{\theta}, \mathbf{y}_b)$, for each $b \in \{1, ..., B\}$ with the density denoted by $\hat{\mathrm{g}}_b(\boldsymbol{\psi}|\boldsymbol{\theta})$. A product of Gaussian densities is also a Gaussian density, hence the integral is tractable, as shown in Proposition 3.1. Let $V_b\left(\boldsymbol{\theta}\right)$ and $\mu_b\left(\boldsymbol{\theta}\right)$ be the conditional variance and mean of $\boldsymbol{\psi}|\boldsymbol{\theta}, \mathbf{y}_b$ respectively of subposterior b, evaluated at $\boldsymbol{\theta}$.

Proposition 3.1.

$$J(\boldsymbol{\theta}) := \int_{\boldsymbol{\psi}} \prod_{b=1}^{B} \hat{\mathrm{g}}_b(\boldsymbol{\psi}|\boldsymbol{\theta}) \mathrm{d}\boldsymbol{\psi} \propto \frac{\exp\left(-\frac{1}{2}\bar{\mu}^\top(\boldsymbol{\theta})\bar{\boldsymbol{V}}^{-1}(\boldsymbol{\theta})\bar{\mu}(\boldsymbol{\theta})\right)}{\exp\left(-\frac{1}{2}\sum\limits_{b=1}^{B} \mu_b^\top(\boldsymbol{\theta})\boldsymbol{V}_b^{-1}(\boldsymbol{\theta})\mu_b(\boldsymbol{\theta})\right)},$$

where

$$\bar{\boldsymbol{V}}(\boldsymbol{\theta}) = \left(\sum\limits_{b=1}^{B} \boldsymbol{V}_b^{-1}(\boldsymbol{\theta})\right)^{-1} \qquad \bar{\mu}(\boldsymbol{\theta}) = \bar{\boldsymbol{V}}(\boldsymbol{\theta}) \textstyle\sum_{b=1}^{B} \boldsymbol{V}_b^{-1}(\boldsymbol{\theta})\boldsymbol{\mu}_b(\boldsymbol{\theta})$$

Proof. see Appendix 3.C □

To estimate $\mu_b(\boldsymbol{\theta})$ and $V_b(\boldsymbol{\theta})$ for $b \in \{1, \ldots, B\}$ Monte Carlo samples drawn from each subposterior distribution can be used. Specific details are given in Section 3.2.3.

Whilst we may be willing to account for the influence of dependence between $\boldsymbol{\theta}$ and $\boldsymbol{\psi}$ using a Gaussian approximation to the distribution of $\boldsymbol{\psi}|\boldsymbol{\theta}$, we certainly wish to capture the key non-Gaussian behaviours in the parameter of interest, $\boldsymbol{\theta}$. This is similar to the aims of Rue et al. (2009), where the Integrated Nested Laplace Approximation (INLA) method was introduced. We approximate $\pi_b(\boldsymbol{\theta}|\mathbf{y}_b)$ by applying KDE methods to the subposterior samples; denote this approximation by $\hat{h}_b(\boldsymbol{\theta})$ such that:

$$\hat{h}_b(\boldsymbol{\theta}) \approx \pi_b(\boldsymbol{\theta}|\mathbf{y}_b) \tag{3.3}$$

KDE methods typically suffer from the curse of dimensionality (Bellman, 1966), however this will be less of a concern since the main focus of MarV is to estimate expectations where the dimension of $\boldsymbol{\theta}$ is low.

Given $\hat{h}_b(\boldsymbol{\theta})$ and $\hat{\mathrm{g}}_b(\boldsymbol{\psi}|\boldsymbol{\theta})$ for all $b \in \{1, \ldots, B\}$ we can evaluate an approximation to $\pi(\boldsymbol{\theta}|\mathbf{y})$, up to a constant of proportionality:

$$\hat{\pi}(\boldsymbol{\theta}|\mathbf{y}) \mathrel{\tilde{\propto}} \prod_{b=1}^{B} \left\{ \hat{h}_b(\boldsymbol{\theta}) \right\} \int_{\boldsymbol{\psi}} \prod_{b=1}^{B} \left\{ \hat{\mathrm{g}}_b(\boldsymbol{\psi}|\boldsymbol{\theta}) \right\} \mathrm{d}\boldsymbol{\psi}. \tag{3.4}$$

With an appropriate importance proposal (Section 3.2.1) we can then approximate the expectation I as defined in Equation 3.1.

To implement importance sampling using the approximation highlighted in Equation 3.4, we need to evaluate the integral in Equation 3.4. Evaluating this integral for each importance proposal sample results in a large storage cost on a single computer. For each importance proposal, of which there are M, the integral of a product of Gaussian distributions needs to be evaluated. As seen in Proposition 3.1, to evaluate the integral of $\boldsymbol{\psi}$ conditional on a given $\boldsymbol{\theta}$, for each batch the conditional mean and variance of $\boldsymbol{\psi}$ needs to be calculated and transferred to the central computer. The covariance contains $\mathcal{O}\left(d_{\boldsymbol{\psi}}^2\right)$ real numbers, and so the evaluation of the integral requires the central core to receive $\mathcal{O}\left(Bd_{\boldsymbol{\psi}}^2M\right)$ real numbers, which can induce a memory bottle neck. We do not pursue this approach, and instead make a further assumption similar to the Laplace Method (Barndorff-Nielsen, 1989) to circumvent the large memory cost.

Let $\tilde{\mathbf{g}}(\boldsymbol{\vartheta})$ denote the probability density function (pdf) of a Gaussian approximation to $\pi(\boldsymbol{\psi}|\boldsymbol{\theta},\mathbf{y})$, that is, an approximation to the full-data posterior density. Like in Scott et al. (2016), given samples from each subposterior distribution we can estimate mean and variance of $\tilde{\mathbf{g}}(\boldsymbol{\vartheta})$:

$$\bar{\boldsymbol{V}}_{\boldsymbol{\vartheta}} = \left(\sum_{b=1}^{B} \hat{\boldsymbol{V}}_{\boldsymbol{\vartheta},b}^{-1}\right)^{-1} \qquad \bar{\mu}_{\boldsymbol{\vartheta}} = \bar{\boldsymbol{V}}_{\boldsymbol{\vartheta}} \textstyle\sum_{b=1}^{B} \hat{\boldsymbol{V}}_{\boldsymbol{\vartheta},b}^{-1}\hat{\mu}_{\boldsymbol{\vartheta},b}\ , \tag{3.5}$$

where $\hat{\mu}_{\boldsymbol{\vartheta},b}$ and $\hat{V}_{\boldsymbol{\vartheta},b}$ are estimated using the Monte Carlo samples drawn from each subposterior distribution for $b \in \{1, \ldots, B\}$.

Let $\tilde{\mathrm{g}}(\boldsymbol{\psi}|\boldsymbol{\theta})$ denote the probability density function (pdf) of a conditional Gaussian approximation to $\pi(\boldsymbol{\psi}|\boldsymbol{\theta}, \mathbf{y})$ with mean and variance estimated from fitting a Gaussian approximation to the joint distribution of $\pi(\boldsymbol{\theta}, \boldsymbol{\psi}|\mathbf{y})$ (Appendix 3.A).

With the Gaussian approximation $\tilde{\mathrm{g}}(\boldsymbol{\psi}|\boldsymbol{\theta})$, and by applying Bayes Theorem, we can avoid integration entirely with the following approximation:

$$\pi(\boldsymbol{\theta}|\mathbf{y}) = \frac{\pi(\boldsymbol{\theta}, \boldsymbol{\psi}|\mathbf{y})}{\pi(\boldsymbol{\psi}|\boldsymbol{\theta}, \mathbf{y})} \mathrel{\tilde{\propto}} \frac{\prod_{b=1}^{B} \hat{h}_b(\boldsymbol{\theta}) \hat{\mathrm{g}}_b(\boldsymbol{\psi}|\boldsymbol{\theta})}{\tilde{\mathrm{g}}(\boldsymbol{\psi}|\boldsymbol{\theta})} =: \hat{\pi}(\boldsymbol{\theta}|\mathbf{y}), \tag{3.6}$$

up to a normalisation constant, for any $\boldsymbol{\psi} \in \boldsymbol{\chi}_{\boldsymbol{\psi}}$. With this approximation the central core only needs to receive MB scalars, the product of $\hat{h}_b$ and $\hat{\mathrm{g}}_b$ for each $b \in \{1, \ldots, B\}$, evaluated at each importance sample. To use Equation 3.4 instead, approximately $MBd_{\boldsymbol{\psi}}^2/2$ scalars would have to be transferred to the central core, since we would have to pass back a variance-covariance matrix from each batch, for each importance weight we would want to estimate.

Although the approximation in Equation 3.6 holds for all values $\boldsymbol{\psi}^*$ in the support of $\boldsymbol{\psi}$, Section 3.2.2 shows that some choices can result in lower variance approximations to $\hat{h}_b$ and $\hat{\mathrm{g}}_b$ and hence a lower overall variance estimator. We suggest, for a given $\boldsymbol{\theta}$, choosing $\boldsymbol{\psi}^*$ to be the expectation of $\boldsymbol{\psi}|\boldsymbol{\theta}$ under the approximate global model $\tilde{\mathsf{g}}$. That is, for a given importance sample $\tilde{\boldsymbol{\theta}}$:

$$\boldsymbol{\psi}^* = \mathbb{E}_{\tilde{\mathsf{g}}}\left[\boldsymbol{\psi}|\boldsymbol{\theta} = \tilde{\boldsymbol{\theta}}\right]. \tag{3.7}$$

Under the full Gaussian approximation with density $\tilde{\mathsf{g}}$, we would expect the region around $\boldsymbol{\psi}^*$ to contain the highest density of samples of ψ. In regions with a large number of samples the variance of our Monte Carlo estimators will be smaller. This is because KDE methods are able to locally estimate the parameters of $\hat{\mathsf{g}}_b$ for a given $\tilde{\boldsymbol{\theta}}$ and b, which produces a lower variance estimator of the density $\hat{\mathsf{g}}_b(\boldsymbol{\psi}|\boldsymbol{\theta})$ when the number of samples in the region increase. See Section 3.2.2 for further details. Choosing $\boldsymbol{\psi}^*$ as in Equation 3.7 means $\tilde{\mathsf{g}}$ is now independent of $\boldsymbol{\theta}$ and hence our approximation becomes:

$$\hat{\pi}(\boldsymbol{\theta}|\mathbf{y}) \propto \prod_{b=1}^{B} \hat{h}_b(\boldsymbol{\theta})\hat{\mathsf{g}}_b(\boldsymbol{\psi}^*|\boldsymbol{\theta}). \tag{3.8}$$

Algorithm 2 gives details of the full MarV algorithm, which can be viewed as an importance sampler (see Section 3.2.1 for details) targeting an approximation to the marginal $\pi(\boldsymbol{\theta}|\mathbf{y})$. In Algorithm 2 the algorithm is expressed in terms of subposterior densities, however, to avoid numerical underflow it is best to use the log-subposterior densities.

Algorithm 2 MarV

```
With approximations $\hat{\mathrm{g}}_b(\boldsymbol{\psi}|\boldsymbol{\theta}, \mathbf{y}_b)$ and $\hat{h}_b(\boldsymbol{\theta})$
procedure
    partition data into $B$ batches: $\{\mathbf{y}_1, \ldots, \mathbf{y}_B\}$
    parallel for $b \in \{1, \ldots, B\}$ do
        $\boldsymbol{\vartheta}^{(b)} \leftarrow N$ Sample from $\pi_b(\boldsymbol{\vartheta}|y_b)$
    end parallel for
    for $j \in \{1, \ldots, M\}$ do
        $\tilde{\boldsymbol{\theta}}_j \sim q(.)$ Equation 3.9
        $\boldsymbol{\psi}_j^* = \mathbb{E}_{\tilde{G}}\left[\boldsymbol{\psi}|\boldsymbol{\theta} = \tilde{\boldsymbol{\theta}}_j\right]$ Proposition 3.2
    end for
    parallel for $b \in \{1, \ldots, B\}$ do
        for $j \in \{1, \ldots, M\}$ do
            $\hat{\pi}_{b,j} = \hat{\mathrm{g}}_b(\boldsymbol{\psi}^*|\tilde{\boldsymbol{\theta}}_j)\hat{h}_b(\tilde{\boldsymbol{\theta}}_j)$ Equation 3.15 and Equation 3.16
        end for
    end parallel for
    for $j \in \{1, \ldots, M\}$ do
        $W_j = (\prod_{b=1}^{B} w_{b,j})/q(\tilde{\boldsymbol{\theta}}_j)$
    end for
    $W_j = W_j / \sum_{j=1}^{M} W_j$
    return $\sum\limits_{j=1}^{M} W_j \tilde{\boldsymbol{\theta}}_j$
end procedure
```

Section 3.2.3 shows how the parameters of $\hat{\mathrm{g}}_b$ and the value of $\hat{h}_b(\boldsymbol{\theta})$ for a given $\boldsymbol{\theta}$ can be estimated using kernel methods. Standard kernel methods have a computational cost that is $\mathcal{O}(MN)$, where M denotes the number of importance samples and N denotes the number of samples taken from each batch. Other kernel methods, such as the nearest neighbour algorithm (Keller et al., 1985), could be used but are not explored here. To reduce the computational cost Section 3.2.4 introduces a novel kernel, that in one dimension has a cost of $\mathcal{O}(M + N\log(N))$, reducing the overall cost of the MarV algorithm.

Estimation of our parameters and densities using kernel methods offers flexibility, but can lead to poor approximations in regions where samples are sparse. Section 3.2.5 advises that in these regions $\hat{h}_b$ should be replaced by a Gaussian approximation and the parameters of the $\hat{\mathsf{g}}_b$, for all $b \in \{1, \ldots, B\}$, be approximated using the closest reasonable estimate, or linear interpolation between high density regions.

3.2.1 Choice of Importance Proposal

In the previous section we let $\tilde{\mathsf{g}}$ be a Gaussian approximation to the full-posterior distribution, with mean and variance estimates described in Equation 3.5. Here we will use this approximation to inform our importance proposal. In Section 2.1.2 we looked at what to consider when choosing an importance proposal. A heavy tailed proposal is needed to ensure that the importance weights are finite (Asmussen and Glynn, 2007). For this reason, in one-dimensional cases, we choose a Student-t_5 proposal distribution with first two moments fixed to match those used in the Gaussian approximation $\tilde{\mathsf{g}}$.

Let $\mu_{b,\theta}$ and $\sigma_{b,\theta}$ be the mean and standard deviation respectively for a chosen θ parameter, for a given batch $b \in \{1, \ldots, B\}$. The following importance proposal density is suggested:

$$q(\theta) = t_5(\bar{\mu}_\theta, \bar{\sigma}_\theta), \tag{3.9}$$

where

$$\bar{\sigma}_\theta^{-2} = \sum_{b=1}^{B} \sigma_{b,\theta}^{-2}, \qquad \text{and,} \qquad \bar{\mu}_\theta = \bar{\sigma}_\theta^{2} \sum_{b=1}^{B} \sigma_{b,\theta}^{-2} \mu_{b,\theta}. \tag{3.10}$$

The batch means and standard deviations can be estimated from the Monte Carlo samples drawn from each subposterior distribution.

3.2.2 Choice of $\boldsymbol{\psi}^*$

To use MarV to approximate expectations, as described in Algorithm 2, for each $\tilde{\boldsymbol{\theta}}_j$ drawn from our importance proposal we need to choose $\boldsymbol{\psi}_j^* := \boldsymbol{\psi}_j^*(\boldsymbol{\theta})$ for each $j \in \{1, \ldots, M\}$, that is the $\boldsymbol{\psi}$ value at which to evaluate Equation (3.6). We recommend choosing

$$\boldsymbol{\psi}_j^* = \mathbb{E}_{\tilde{g}}\left[\boldsymbol{\psi} | \boldsymbol{\theta} = \tilde{\boldsymbol{\theta}}_j\right],$$

for each $j \in \{1, \ldots, M\}$. By evaluating our approximation at the conditional expectation we hope to be in the region with largest density of Monte Carlo samples. This is important as there are parameters that need to be estimated for other steps in the algorithm, and they will be approximated locally (Section 3.2.3). The higher the concentration of samples in these regions, the lower the variance will be on these parameter estimates. In the remainder of this section we will derive $\boldsymbol{\psi}_j^*$.

Recall $\tilde{\mathrm{g}}(\boldsymbol{\theta}, \boldsymbol{\psi})$ is our full Gaussian approximation to $\pi(\boldsymbol{\theta}, \boldsymbol{\psi}|\mathbf{y})$ based on each batch $b \in \{1, \ldots, B\}$. Under this model, let $\mu_{\boldsymbol{\theta}}$ and $\mu_{\boldsymbol{\psi}}$ denote the marginal expectations of $\boldsymbol{\theta}$ and $\boldsymbol{\psi}$, respectively. Additionally, let $V_{\boldsymbol{\theta}}$ and $V_{\boldsymbol{\psi}}$ denote the marginal variances of $\boldsymbol{\theta}$ and $\boldsymbol{\psi}$, and define $V_{\boldsymbol{\theta},\boldsymbol{\psi}} = V_{\boldsymbol{\psi},\boldsymbol{\theta}}^\top$ be the $d_{\boldsymbol{\theta}} \times d_{\boldsymbol{\psi}}$ covariance matrix between $\boldsymbol{\theta}$ and $\boldsymbol{\psi}$ so that

$$\mathbb{E}_{\tilde{\mathrm{g}}(\cdot)} \begin{bmatrix} \boldsymbol{\theta} \\ \boldsymbol{\psi} \end{bmatrix} = \begin{bmatrix} \mu_{\boldsymbol{\theta}} \\ \mu_{\boldsymbol{\psi}} \end{bmatrix}, \qquad \text{and} \qquad \mathrm{Var}_{\tilde{\mathrm{g}}(\cdot)} \begin{bmatrix} \boldsymbol{\theta} \\ \boldsymbol{\psi} \end{bmatrix} = \begin{bmatrix} V_{\boldsymbol{\theta}} & V_{\boldsymbol{\theta},\boldsymbol{\psi}} \\ V_{\boldsymbol{\psi},\boldsymbol{\theta}} & V_{\boldsymbol{\psi}} \end{bmatrix}.$$

With this set up a well known result gives the conditional distribution of $\boldsymbol{\psi}|\boldsymbol{\theta}$ under $\tilde{\mathrm{g}}(\cdot)$.

Proposition 3.2.

$$\boldsymbol{\psi}|\boldsymbol{\theta} \sim N\left(\mu_{\tilde{\mathrm{g}}(\boldsymbol{\psi}|\boldsymbol{\theta})}, V_{\tilde{\mathrm{g}}(\boldsymbol{\psi}|\boldsymbol{\theta})}\right),$$

where

$$V_{\tilde{\mathrm{g}}(\boldsymbol{\psi}|\boldsymbol{\theta})} := V_{\boldsymbol{\psi}} - V_{\boldsymbol{\psi},\boldsymbol{\theta}} V_{\boldsymbol{\theta}}^{-1} V_{\boldsymbol{\theta},\boldsymbol{\psi}}, \tag{3.11}$$

$$\mu_{\tilde{\mathrm{g}}(\boldsymbol{\psi}|\boldsymbol{\theta})} := \mu_{\boldsymbol{\psi}} + V_{\boldsymbol{\psi},\boldsymbol{\theta}} V_{\boldsymbol{\theta}}^{-1} \left(\boldsymbol{\theta} - \mu_{\boldsymbol{\theta}}\right), \tag{3.12}$$

under the assumption that $\boldsymbol{\psi}$ and $\boldsymbol{\theta}$ are jointly Gaussian distributed with joint density $\tilde{\mathsf{g}}(\boldsymbol{\theta}, \boldsymbol{\psi})$, for all $\boldsymbol{\theta}$ in its support.

Proof. This is a standard result - see Eaton (1983). Proof in Appendix 3.A.

□

Following Proposition 3.2 we set our estimator for $\boldsymbol{\psi}_j^*$ to be $\mu_{\tilde{\mathsf{g}}(\boldsymbol{\psi}|\boldsymbol{\theta})}$, as in Equation 3.12. In Section 3.2.3 we will show how to use kernel methods to develop the estimators for the other quantities needed to calculate this estimator.

3.2.3 Estimation of $\hat{h}_b(\tilde{\boldsymbol{\theta}})$ and $\hat{\mathsf{g}}_b(\tilde{\boldsymbol{\theta}})$

Both $\hat{h}_b$ and $\hat{\mathsf{g}}_b$ can be estimated in an embarrassingly parallel fashion, so focus is placed on how to estimate them in a single batch, b. Hence, for the remainder of this subsection we will drop the subscript b from the parameters for ease of notation. Let $\boldsymbol{\vartheta}^{(1:N)} := (\boldsymbol{\vartheta}_1, \ldots, \boldsymbol{\vartheta}_N)$ be the N Monte Carlo samples drawn from $\boldsymbol{\vartheta}|\mathbf{y}_b$ for our current batch of interest. We define $\boldsymbol{\psi}^{(1:N)} := (\boldsymbol{\psi}_1, \ldots, \boldsymbol{\psi}_N)$ and $\boldsymbol{\theta}^{(1:N)} := (\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_N)$ in a similar fashion.

For ease of notation we drop the bandwidth parameter from the kernel, that is $\mathcal{K}(x|\rho) = \mathcal{K}(x)$. Let $s : \mathbb{R}^{d_{\boldsymbol{\theta}}} \to \mathbb{R}^d$, for some $d \in \mathbb{N}$. For any kernel $\mathcal{K} : \mathbb{R}^{d_{\boldsymbol{\theta}}} \to$

$\mathbb{R}^+ \cup \{0\}$ with $\int_{\boldsymbol{\chi}_{\boldsymbol{\theta}}} \mathcal{K}(x)\mathrm{d}x = 1$, and values $\{\tilde{\boldsymbol{\theta}}_i\}_{i=I_1}^{I_2}$ for any $1 \leq I_1 \leq I_2 \leq N$, define

$$s_{\mathcal{K}}(\tilde{\boldsymbol{\theta}}; I_1 : I_2) = \sum_{i=I_1}^{I_2} \mathcal{K}(\tilde{\boldsymbol{\theta}} - \boldsymbol{\theta}_i) s(\boldsymbol{\theta}_i), \tag{3.13}$$

that is the sum between two indices I_1 and I_2 of the product of a function s and kernel $\mathcal{K}$, which is centred at $\tilde{\boldsymbol{\theta}}$, with both being evaluated at $\boldsymbol{\theta}_i$. We abbreviate $s_{\mathcal{K}}(\tilde{\boldsymbol{\theta}}; 1 : N)$ to $s_{\mathcal{K}}(\tilde{\boldsymbol{\theta}})$. Define $\mathbf{1}(\tilde{\boldsymbol{\theta}}) = 1$, so that the standard kernel estimate of the density at $\tilde{\theta}$ is:

$$\mathbf{1}_{\mathcal{K}}(\tilde{\boldsymbol{\theta}})/N = \frac{1}{N} \sum_{i=1}^{N} \mathcal{K}(\tilde{\boldsymbol{\theta}} - \boldsymbol{\theta}_i) \tag{3.14}$$

Similarly, the kernel estimate of $s(\tilde{\boldsymbol{\theta}})$ is $s_{\mathcal{K}}/\mathbf{1}_{\mathcal{K}}(\tilde{\boldsymbol{\theta}})$. This notation is introduced to make the following sections more concise. Let $\boldsymbol{\psi}(\theta_i) = \boldsymbol{\psi}_i$ and $\boldsymbol{\psi}\boldsymbol{\psi}^\top(\theta_i) = \boldsymbol{\psi}_i\boldsymbol{\psi}_i^\top$ for all $i \in \{1, \ldots, N\}$.

We set

$$\hat{h}(\tilde{\boldsymbol{\theta}}) = \mathbf{1}_{\mathcal{K}}(\tilde{\boldsymbol{\theta}})/N, \tag{3.15}$$

i.e., we use standard kernel estimation for $\hat{h}$. Recall that $\hat{\mathsf{g}}_b(\boldsymbol{\psi}|\tilde{\boldsymbol{\theta}})$ is the density of a conditional Gaussian approximation to $\pi(\boldsymbol{\psi}|\boldsymbol{\theta} = \tilde{\theta}, \mathbf{y})$. We want to estimate the mean and variance of $\boldsymbol{\psi}$ conditional on $\boldsymbol{\theta}$. Hence we set

$$\hat{\mu}_{\boldsymbol{\psi}|\tilde{\boldsymbol{\theta}}} = \frac{\boldsymbol{\psi}_{\mathcal{K}}(\tilde{\boldsymbol{\theta}})}{\mathbf{1}_{\mathcal{K}}(\tilde{\boldsymbol{\theta}})} \qquad \text{and} \qquad \hat{V}^b_{\boldsymbol{\psi}|\tilde{\boldsymbol{\theta}}}(\tilde{\boldsymbol{\theta}}) = T(\tilde{\boldsymbol{\theta}}) - \hat{\mu}\hat{\mu}^\top, \tag{3.16}$$

where

$$T(\tilde{\boldsymbol{\theta}}) := \frac{[\boldsymbol{\psi}\boldsymbol{\psi}^\top]_{\mathcal{K}}(\tilde{\boldsymbol{\theta}})}{\mathbf{1}_{\mathcal{K}}(\tilde{\boldsymbol{\theta}})},$$

and $\hat{\mu} = \hat{\mu}_{\boldsymbol{\psi}|\tilde{\boldsymbol{\theta}}}$.

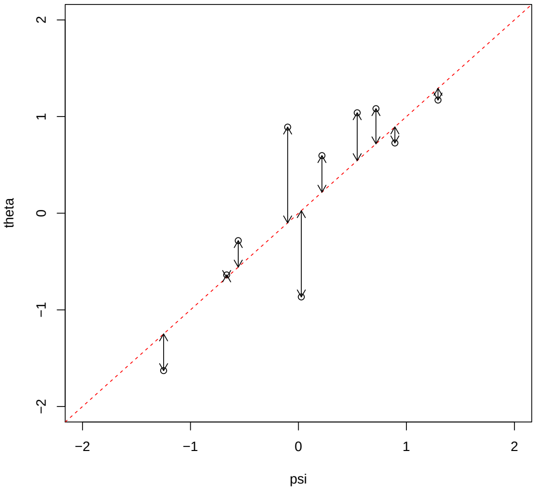


Figure 3.1: θ plotted against ψ for a simple linear model.

Recall from Section 2.2 the Gaussian kernel:

$$\mathcal{K}^G(\boldsymbol{x}) = \frac{1}{\rho^d\sqrt{2\pi}} \exp\left(-\frac{1}{2\rho^2}\boldsymbol{x}^\top\boldsymbol{x}\right).$$

Using this kernel with N samples from our subposterior and evaluating our M importance samples leads to a computational cost of $\mathcal{O}\left(MN\right)$, since for each importance sample $\tilde{\boldsymbol{\theta}}_j$ for $j \in \{1, \ldots, M\}$ we would have to evaluate the chosen kernel at each Monte Carlo sample:

$$\frac{1}{N}\sum_{i=1}^{N}\mathcal{K}^{G}(\boldsymbol{\theta}_i),$$

to obtain our estimate of $\hat{h}(\tilde{\boldsymbol{\theta}}_j)$. Due the variance estimate being a matrix of size $d_{\boldsymbol{\psi}} \times d_{\boldsymbol{\psi}}$, the total computational cost of evaluating $\hat{\mathsf{g}}(\tilde{\boldsymbol{\theta}}_j)$ and $\hat{h}(\tilde{\boldsymbol{\theta}}_j)$ would be of $\mathcal{O}\left(MNd_{\boldsymbol{\psi}}^2\right)$ using a Gaussian kernel. If this is not a computational issue, then a Gaussian kernel can be implemented for MarV. However, for situations where this is an issue, and $\boldsymbol{\theta}$ is one-dimensional, we propose a computationally cheaper kernel.

3.2.4 The Exponential-Uniform Window Kernel (EUWoK)

The exponential-uniform window kernel (EUWoK) is a kernel function that can use the information from previously evaluated points to reduce the overall computational cost of estimating the parameters of $\hat{\mathsf{g}}$ from $\mathcal{O}\left(MNd_{\boldsymbol{\psi}}^2\right)$ to $\mathcal{O}\left((M + N\log(N))d_{\boldsymbol{\psi}}^2\right)$. This is done by calculating the KDE at one point, which has an initial computational cost of $\mathcal{O}\left(N\right)$, then updating the original estimate for the remainder of $\{\tilde{\boldsymbol{\theta}}_j\}_{j=1}^{M}$, the computational cost of which is of $\mathcal{O}\left(M + N\right)$. To avoid confusion with the Gaussian bandwidth parameter ρ, we introduce $\lambda, \epsilon \in {}^{+}$ as bandwidth parameters for EUWoK.

The EUWoK kernel $\mathcal{K}^W : \mathbb{R} \to \mathbb{R}^+ \cup \{0\}$, for some $\epsilon, \lambda \in \mathbb{R}^+$ and $x \in \mathbb{R}$ we define

$$\mathcal{K}^W(x) := \frac{1}{2(\lambda + \epsilon)} \left(\lambda \cdot \mathcal{K}^-(x) + 2\epsilon \cdot \mathcal{K}^U(x) + \lambda \cdot \mathcal{K}^+(x) \right), \tag{3.17}$$

where

$$\mathcal{K}^-(x) := 1/\lambda \cdot \exp((x + \epsilon)/\lambda) \cdot \mathbb{I}(\{x < -\epsilon\}), \tag{3.18}$$

$$\mathcal{K}^U(x) := 1/2\epsilon \cdot \mathbb{I}(\{-\epsilon \leq x \leq \epsilon\}), \tag{3.19}$$

$$\mathcal{K}^+(x) := 1/\lambda \cdot \exp((-x - \epsilon)/\lambda) \cdot \mathbb{I}(\{x > \epsilon\}). \tag{3.20}$$

To tune our kernel we resort to similar methods to tuning a Gaussian kernel. For a given bandwidth ρ for the Gaussian kernel $\mathcal{K}^G$, we set

$$\lambda = 0.8362\rho \qquad \text{and} \qquad \epsilon = 0.3111\rho, \tag{3.21}$$

to minimise the KL-divergence from $\mathcal{K}^W$ to $\mathcal{K}^G$. Figure 3.2 compares EUWoK and a Gaussian kernel, with setting λ and β as in (3.21), showing the kernels have similar values over the kernel's support.

EUWoK is a piecewise mixture of a uniform window and an exponential kernel (sometimes referred to as a Laplace kernel). The idea behind combining kernels or '*window*' functions is not new. The *Tukey Window* (Bloomfield, 2004) and *Planc-Taper Window* (Tu, 2010) are both piece-wise defined functions. These are both similar to EUWoK in that they are both piece-wise functions with a a uniform part and then different tails. Tukey has cosine tails and Planc-Taper having tails similar to that of a Planc distribution (McKechan et al., 2010). Neither of these solved our computational issues, which led us to create EUWoK, a novel contribution developed for this work.

The advantage of EUWoK is that it reduces the computational cost relative to other kernels. If this was our only goal this could be achieved using a uniform kernel or a Laplace kernel. However, often a Gaussian kernel is the first choice when using KDE methods (Sheather and Jones, 1991), so we would also like a kernel that more closely resembles the Gaussian shape.

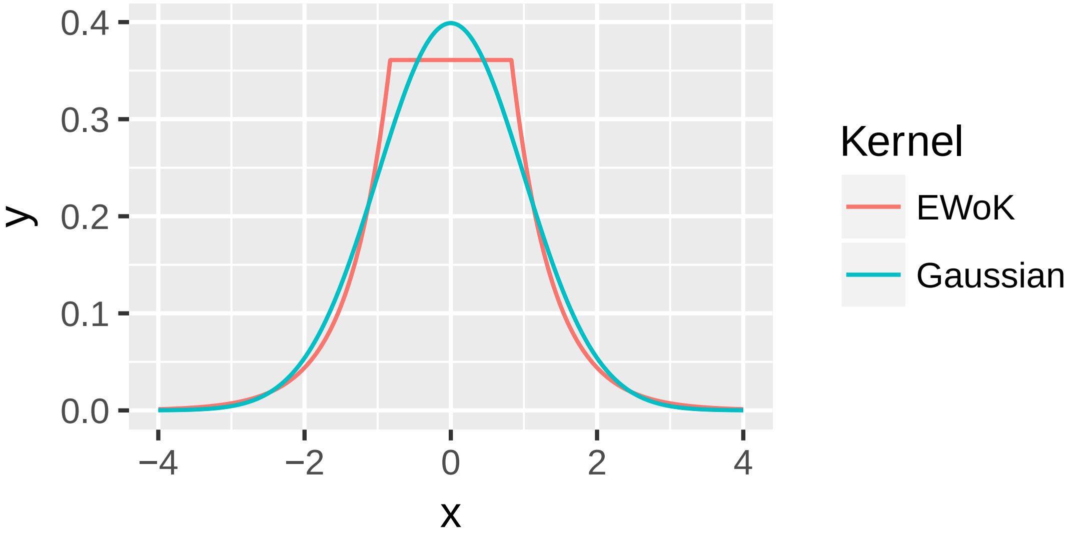


Figure 3.2: EUWoK compared to Gaussian Kernel with tuning parameters chosen to minimise Kullback-Leibler divergence from the Gaussian kernel to EUWoK.

Linear Sweep Algorithm

To gain the reduction in computational cost for one-dimensional θ, we need to consider $\mathcal{K}^-$, $\mathcal{K}^U$, and $\mathcal{K}^+$ separately. To calculate $s_{\mathcal{K}^W}$ for a given function s, it is sufficient to calculate $s_{\mathcal{K}^-}$, $s_{\mathcal{K}^U}$ and $s_{\mathcal{K}^+}$ separately since

$$s_{\mathcal{K}^W}(\tilde{\theta}) = \frac{1}{2(\lambda+\epsilon)}\left(\lambda \cdot s_{\mathcal{K}^-}(\tilde{\theta}) + 2\epsilon \cdot s_{\mathcal{K}^U}(\tilde{\theta}) + \lambda \cdot s_{\mathcal{K}^+}(\tilde{\theta})\right).$$

Our approach is to sort, in ascending order, the Monte Carlo samples drawn from the current subposterior of interest, denoted $\{\theta_i\}_{i=1}^N$ and the proposed importance samples $\{\tilde{\theta}_j\}_{j=1}^M$. This has, at worst, a cost of $\mathcal{O}(N \log N)$ using the merge sort algorithm (Knuth (1998)), we assume that Quasi-Monte Carlo was used to generate $\{\tilde{\theta}_j\}_{j=1}^M$, so these samples will have been sampled in ascending order. Given that both the Monte Carlo subposterior and importance samples are ordered, we will now show that $\{\mathcal{K}^-(\tilde{\theta}_j)\}_{j=1}^M$, $\{\mathcal{K}^U(\tilde{\theta}_j)\}_{j=1}^M$ and $\{\mathcal{K}^+(\tilde{\theta}_j)\}_{j=1}^M$ can be calculated with a computational cost of $\mathcal{O}(M+N)$. Let $n_j^- := \max\{n : \theta_n < \tilde{\theta}_j - \epsilon\}$ for $j \in \{1, \ldots, M\}$, that is the index of the largest MCMC sample that is less than the $j-th$ importance sample, when no such MCMC sample exists we define n_j to be 0. Similarly we define $n_j^+ := \min\{n : \theta_n > \tilde{\theta}_j + \epsilon\}$ for $j \in \{1, \ldots, M\}$, and define n_j^+

to be $N+1$ when $\theta_n > \tilde{\theta}_j + \epsilon$ for all $i \in \{1, \dots, N\}$. For $\tilde{\theta}_n < -\epsilon$:

$$\begin{aligned} s_{\mathcal{K}^-}(\tilde{\theta}_j; 1 : n_j^-) &= \sum_{i=1}^{n_j^-} \exp[(\tilde{\theta}_j - \theta_i)/\lambda] s(\theta_i) \\ &= \sum_{i=1}^{n_j^-} \exp[(\tilde{\theta}_j - \tilde{\theta}_{j+1} + \tilde{\theta}_{j+1} - \theta_i)/\lambda] s(\theta_i) \\ &= \exp[-(\tilde{\theta}_{j+1} - \tilde{\theta}_j)/\lambda] \sum_{i=1}^{n_j} \exp[(\tilde{\theta}_{j+1} - \theta_i)/\lambda] s(\theta_i) \\ &= \exp[-(\tilde{\theta}_{j+1} - \tilde{\theta}_j)/\lambda] s_{\mathcal{K}^-}(\tilde{\theta}_{j+1}; 1 : n_j^-) \end{aligned}$$

Then for all $n_j < n_{j+1}^-$:

$$\begin{aligned} s_{\mathcal{K}^-}(\tilde{\theta}_{j+1}) &= s_{\mathcal{K}^-}(\tilde{\theta}_{j+1}; 1 : n_{j+1}^-), \\ &= s_{\mathcal{K}^-}(\tilde{\theta}_{j+1}; 1 : n_j^-) + s_{\mathcal{K}^-}(\tilde{\theta}_{j+1}; (n_j^- + 1) : n_{j+1}^-), \\ &= \exp[(\tilde{\theta}_j - \tilde{\theta}_{j+1})/\lambda] s_{\mathcal{K}^-}(\tilde{\theta}_{j+1}; 1 : n_j^-) + s_{\mathcal{K}^-}(\tilde{\theta}_{j+1}; (n_j^- + 1) : n_{j+1}^-), \\ &= \exp[(\tilde{\theta}_j - \tilde{\theta}_{j+1})/\lambda] s_{\mathcal{K}^-}(\tilde{\theta}_{j+1}) + s_{\mathcal{K}^-}(\tilde{\theta}_{j+1}; (n_j^- + 1) : n_{j+1}^-). \end{aligned}$$

When using these relationships to calculate $\{s_{\mathcal{K}^-}(\tilde{\theta}_j)\}_{j=1}^M$ we can calculate $s_{\mathcal{K}^-}(\tilde{\theta}_1)$ with a computational cost of $\mathcal{O}\left(n_1^- + 1\right)$. Given $s_{\mathcal{K}^-}(\tilde{\theta}_j)$, we can calculate $s_{\mathcal{K}^-}(\tilde{\theta}_{j+1})$ with a computational cost of $\mathcal{O}\left(n_{j+1}^- - n_j^- + 1\right)$. Hence the entirety of $\{s_{\mathcal{K}^-}(\tilde{\theta}_j)\}_{j=1}^M$ can be calculated with a cost of $\mathcal{O}(N + M)$. Algorithm 3 gives pseudo code for the linear sweep algorithm. An analogous argument holds for the cost of calculating $\{s_{\mathcal{K}^+}(\tilde{\theta}_j)\}_{j=1}^M$.

Algorithm 3 EUWoK; linear implementation of lower tail

```
procedure
    $A_1 \leftarrow \{\boldsymbol{\theta}_i : \boldsymbol{\theta}_i < \tilde{\boldsymbol{\theta}}_1 - \epsilon\}$
    $s_{\mathcal{K}^+} \leftarrow \sum_{i \in A_1} 1/\lambda \cdot \exp(-((\tilde{\boldsymbol{\theta}}_1 - \epsilon - \boldsymbol{\theta}_i)\lambda)$
    for $j \in \{2, \dots, M\}$ do
        $A_j \leftarrow \{\theta_i : \tilde{\boldsymbol{\theta}}_{j-1} - \epsilon < \theta_i <= \tilde{\boldsymbol{\theta}}_j - \epsilon\}$
        $D_j \leftarrow \sum_{i \in A_j} 1/\lambda \cdot \exp(-((\tilde{\boldsymbol{\theta}}_1 - \epsilon - \boldsymbol{\theta}_i)\lambda)$
        $C_j \leftarrow \exp(-\lambda(\tilde{\boldsymbol{\theta}}_j - \tilde{\boldsymbol{\theta}}_{j-1})) \times S_{j-1}$
        $S_j \leftarrow C_j + D_j$
    end for
return $S = \{S_1, \dots, S_M\}$
end procedure
```

Showing $\{s_{\mathcal{K}^U}(\tilde{\theta}_j)\}_{j=1}^M$ can be calculated in $\mathcal{O}(M + N)$ is simpler. Again $\{\tilde{\theta}\}_{j=1}^M$ has to be sorted in ascending order, then for every $j \in \{1, \dots, M-1\}$:

$$s_{\mathcal{K}^U}(\tilde{\theta}_{j+1}) = s_{\mathcal{K}^U}(\tilde{\theta}_j) + \frac{(n_{j+1}^+ - n_j^+) - (n_{j+1}^- - n_j^-)}{2\epsilon},$$

which simply put, is to count how many samples have entered and left the uniform kernel, then adjust accordingly.

3.2.5 Tail Estimation

In the limit as the number of samples tends to infinity *i.e.*, $N \to \infty$ the bandwidth parameters (ρ for $\mathcal{K}^G$ or λ and ϵ for $\mathcal{K}^W$) tend to ∞ and the error in the KDE method tends to zero. In practice, N is finite and the methods can produce large-variance estimates in low density regions. This is demonstrated in Figure 3.3 and highlights an issue that can arise when evaluating KDE approximations in the tails of a distribution. Since we are weighting our parameter estimates for $\hat{\mathrm{g}}_b$ using kernel functions, this issue is not exclusive to $\hat{h}_b$.

Below we give a criterion for determining when KDE estimates can be poor, and propose alternative approaches that can be taken in these situations. We provide different solutions for approximating $\hat{h}_b$ and $\hat{\mathrm{g}}_b$ in low density regions.

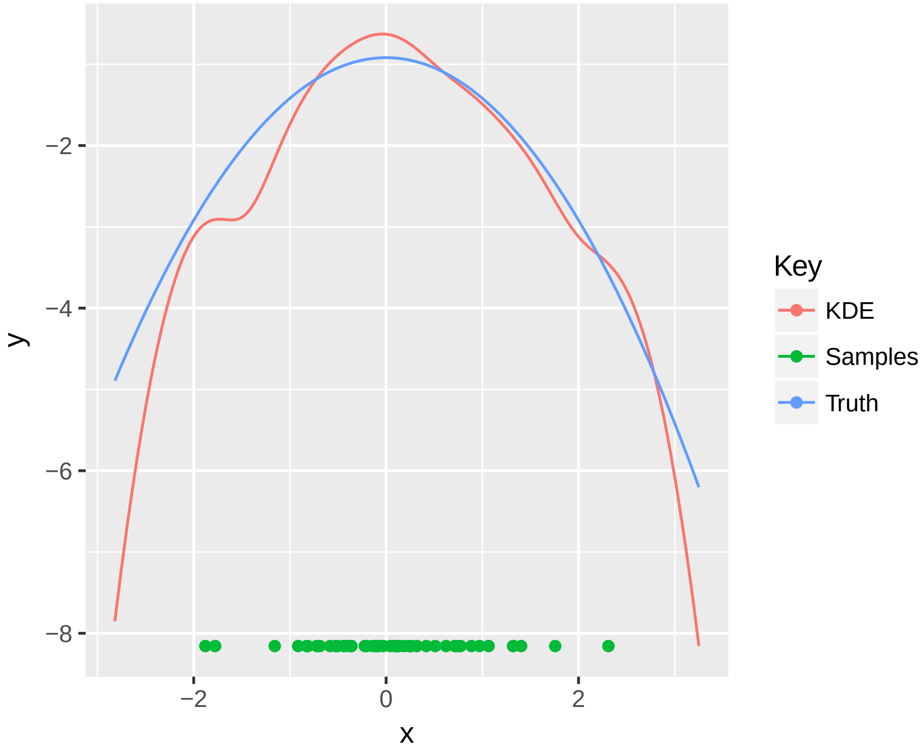


Figure 3.3: A KDE to a standard Gaussian distribution with 50 samples, using Sheather-Jones bandwidth selector, and where y is the log-density.

Suppose we wish to evaluate $\hat{h}_b(\boldsymbol{\theta}^*)$. We would like at least η of the N samples to lie within the $(\alpha/2, 1-\alpha/2)$ quantiles of the symmetric kernel function centred at $\boldsymbol{\theta}^*$, for some tuning parameters $\alpha \in [0, 1/2]$ and $\eta \in \mathbb{N}$, *i.e.*,

$$\left\{ \sum_{i=1}^{N} \mathbb{I} \left(q_{\mathcal{K}}(\alpha/2) < |\boldsymbol{\theta}_i - \boldsymbol{\theta}^*| < q_{\mathcal{K}}(1 - \alpha/2) \right) \right\} \geq \eta, \tag{3.22}$$

where $q_{\mathcal{K}}$ denotes the quantile function for a kernel $\mathcal{K}$ and $\mathbb{I}$ denotes the indicator function as defined in Section 2.2. If the condition in Equation 3.22 holds, we use the KDE approximation for $\pi_b(\boldsymbol{\theta}|\mathbf{y}_b)$, otherwise we use the density of a Gaussian of $\boldsymbol{\theta}$ estimated from the samples of the b-th subposterior distribution.

We choose $\alpha = 0.05$ and $\eta = 10$ *i.e.*, 10 samples contained within the 95% highest density region of the kernel. We would like to use KDE methods as much as possible, however, including estimates with next to no samples would lead to a large variance, making our result highly unreliable.

When estimating $\hat{\mathbf{g}}$, we have to estimate a variance-covariance matrix which has a number of components proportional to $d_{\boldsymbol{\psi}}^2$, hence we alter Equation 3.22:

$$\left\{ \sum_{i=1}^{N} \mathbb{I} \left(q_{\mathcal{K}}(\alpha/2) < \boldsymbol{\theta}_i < q_{\mathcal{K}}(1 - \alpha/2) \right) \right\} \geq \eta_1 d_{\boldsymbol{\psi}}^2. \tag{3.23}$$

If the condition in Equation 3.23 holds then we estimate $V(\boldsymbol{\theta}^*)$ and $\mu(\boldsymbol{\theta}^*)$ as in Equation 3.12. When it does not hold we do not have a sufficient number of samples in a nearby region to estimate the variance, and an attempt to do so might even produce a singular, or near singular matrix, causing the estimate of $\hat{\mathbf{g}}_b(\boldsymbol{\theta}^*)$ to have high variance or be impossible to evaluate.We suggest taking our nearest good estimate of the correlation and continue forward with that, whilst still estimating the marginal variances and mean until the following condition does not hold:

$$\left\{\sum_{i=1}^{N} \mathbb{I}\left(q_{\mathcal{K}}(\alpha/2) < \boldsymbol{\theta}_i < q_{\mathcal{K}}(1-\alpha/2)\right)\right\} \geq \eta_2 d_{\boldsymbol{\psi}}. \tag{3.24}$$

At this point we deem that there is insufficient information to estimate the mean and marginal variance, and instead the nearest estimate of $\hat{\mathsf{g}}_b(\tilde{\boldsymbol{\theta}})$ is carried forward. When the ‘low-density’ regions are between ‘high-density’ regions, instead of using nearest observation carried forward, linear interpolation can be used.

3.3 Experiments

We will be comparing MarV to the following methods. Each of them have been described in detail in Chapter 1, however here is a brief reminder of the ones we will be using in this chapter:

- The Consensus Algorithm (**Cons**) - an averaging method that is exact in the Gaussian case, but that can struggle with certain non-Gaussian behaviour in posteriors and subposteriors (Scott et al. (2016), Section 2.5.1 of the thesis).

- Semi-Parametric Kernel Density Estimation (**SKDE**) - Neiswanger et al. (2013) introduced this method that allows the user generate samples from an approximation to the full-posterior distribution, whilst making very few parametric assumptions. The method initially assumes a Gaussian distribution then tends to a full kernel approximation as the samples from each subposterior distribution tend to infinity. This method can capture all kinds of behaviour, but can struggle with high-dimensional posteriors (Section 2.5.2).

- Posterior Interval estimation (**PIE**) - similar to MarV, PIE does not focus on generating samples from a full-posterior approximation. PIE approximates one-dimensional quantiles based on calculating the Wasserstein barycenter, which can be thought of as an 'average distribution'. It is a simplification of Srivastava et al. (2014) and Srivastava et al. (2018)'s Wasserstein posteriors (WASP) method. This method is fast, but can struggle when subposterior and inflated-subposterior distributions have differing variances. (Li et al. (2017), Section 2.5.3).

- Gaussian Barycenter (**GB**) - a parametric version of WASP, this method assumes each subposterior distribution is Gaussian and finds the barycenter based on that assumption. In general this method is much easier to implement compared to the non-parametric WASP, but can only give samples from a Gaussian-posterior distribution.

- Average re-centering (**AR**) - introduced by Robert et al. (2019) this method re-centers the samples drawn from each inflated-subposterior distribution to the global mean, which is estimated from the inflated-subposterior distribution samples. This method is easy to apply, however it struggles when the higher moments differ within each inflated-subposterior distribution.

Within each experiment we will be using Monte Carlo algorithms to sample from the full-data posterior distribution, and both subposterior and inflated-subposterior distributions. This is so we can compare our results with all the methods listed above. Each sampling method will be described within each section. We will make use of the following measures to assess the accuracy of the divide-and-conquer approximations, (a), against the full-data posterior distribution, (f):

- Standardised mean (SM):

$$\frac{|\hat{\mu}_f - \hat{\mu}_a|}{\hat{\sigma}_f},$$

where $\hat{\mu}_f$ and $\hat{\sigma}_f$ are the mean and variances estimates. These are obtained using samples generated from an MCMC algorithm drawing from the full-data posterior distribution. Similarly, $\hat{\mu}_a$ denotes the mean estimated from samples generated from an approximation to the full-posterior distribution, for example those generated from MarV. This gives a measure of how different the locations of the divide-and-conquer and Monte Carlo approximations are, relative to the spread observed in these approximations. This is also known as one-dimensional Mahalanobis distance.

- Skewness (SK):

$$|\hat{\gamma}_f - \hat{\gamma}_a|.$$

where $\hat{\gamma}$ is an approximation to the standardised third moment(skewness):

$$\hat{\gamma}_f = \mathbb{E}\left[\left(\frac{(\theta_f - \hat{\mu}_f)}{\hat{\sigma}_f}\right)^3\right],$$

where $\hat{\mu}_f$ and $\hat{\sigma}_f$ are defined as above, and the expectation is estimated using the MCMC samples from the full-data posterior distribution. $\hat{\gamma}_a$ is defined is a similar way, substituting $\hat{\mu}_a$ and $\hat{\sigma}_a$ in place of $\hat{\mu}_f$ and $\hat{\sigma}_f$.

- Integrated absolute distance (IAD):

$$\int |\pi_a(\theta) - \pi_f(\theta)| \mathrm{d}\theta \in [0, 1],$$

where $\pi_f(\theta)$ and $\pi_a(\theta)$ denote the full-data posterior distribution and an approximation to the full-posterior distribution respectively. This metric is intractable, so we use kernel methods to estimate both $\pi_f(\theta)$ and $\pi_a(\theta)$. This is achieved using code from Chan et al. (2021). IAD is a measure of how little mass of two distributions overlap. An IAD value of one means the two distributions have no overlap, in contrast a value of zero means the two distributions are equal.

3.3.1 Tri-variate Gaussian Posterior Distribution

One of the points of focus when designing MarV was to create an algorithm that is competitive with the Consensus Monte Carlo algorithm in the Gaussian case. We would like to be able to capture non-Gaussian behaviour, however if MarV can not capture Gaussian behaviour its use would be limited. For this first example the posterior distribution is a tri-variate Gaussian. Asymptotically, as the number of Monte Carlo samples drawn from each subposterior or inflated-subposterior distribution tend to infinity, all the above methods are exact. However, typically non-parametric methods, or methods with weaker distributional assumptions, converge more slowly. This example exists to show that our method is competitive with the Consensus algorithm even in the Gaussian case.

Another aim of this experiment is to highlight the effects of having batches of data of different sizes, which can happen when dealing with privacy concerns or data centers with different data sizes. In these cases, the data are more likely to vary in terms of variance structure; different data owners may have different amounts of data leading to very different variances amongst the subposterior distributions. Some of the methods, namely AR, GB, and PIE, struggle with the differing variances, which will be reflected in the results.

Typically subposterior distributions are derived from a data-generating model. Here, for ease of simulation, we define each subposterior distribution directly, this is for ease of simulation. They are defined as follows:

$$\pi_b(\boldsymbol{\vartheta}|\boldsymbol{\mu}_b, \boldsymbol{\Sigma}_b) = \mathcal{N}_3(\boldsymbol{\mu}_b, \boldsymbol{\Sigma}_b),$$

where $\boldsymbol{\mu}_b$ and $\boldsymbol{\Sigma}_b$ have been sampled from:

$$\boldsymbol{\mu}_b \overset{iid}{\sim} \mathcal{N}(\mathbf{0}, I_3), \qquad\qquad \boldsymbol{\Sigma}_b^{-1} \sim \mathcal{W}\left(5, I_3/5\right),$$

for each $b \in \{1, \ldots, B\}$, and where $\mathcal{W}\left(\nu, S\right)$ denotes a Wishart distribution with degrees of freedom ν and shape matrix S.

We simulated 2,000 samples from each subposterior and inflated-subposterior distribution. We applied the approximation methods to the relevant samples, setting $\theta = \boldsymbol{\vartheta}_1$ and $\boldsymbol{\psi} = \boldsymbol{\vartheta}_{-1}$. This was repeated 5 times, with new $\boldsymbol{\mu}_b$ and $\boldsymbol{\Sigma}_b$ simulated for each repetition. Each subposterior and inflated-subposterior distribution was Gaus-

Table 3.1: Mean (standard deviation) for each Gaussian experiment and each metric, over 5 runs. The metrics are as follows: standardised mean (SM), Skewness (SK), and Integrated absolute distance (IAD). The methods tested against are: Marginal Views (MV), The Consensus Algorithm (Cons), Semi-Parametric Kernel Density Estimation (SKDE), Posterior Interval Estimation (PIE), Gaussian Barycenter (GB), and Average Re-centering (AR).

	SM	**SK**	**IAD**
MarV	0.06 (0.04)	**0.05 (0.04)**	0.05 (<0.01)
Cons	**0.02 (0.02)**	0.06 (0.04)	**0.03 (0.01)**
SKDE	0.26 (0.20)	0.18 (0.15)	0.16 (0.07)
PIE	0.32 (0.26)	0.08 (0.06)	0.74(0.02)
GB	0.33 (0.22)	0.07 (0.06)	0.70 (0.01)
AR	0.28 (0.15)	0.06 (0.05)	0.70 (0.01)

sian, with some variation between each subposterior distribution. For this reason, the subposterior and inflated-subposterior distribution plots have been omitted.

We recorded the mean result with the standard deviation for each of the metrics described in Section 3.3.1 displayed in Table 3.1 for all 5 runs. It can be seen that MarV is competitive with the Consensus Algorithm. MarV's mean discrepancy is within 2 standard deviations of the Consensus Algorithm's mean discrepancy for each of three measures. This is clearly not true for the other methods, showing MarV has close accuracy to the Consensus Algorithm in this Gaussian example.

Table 3.1 shows the PIE, GB, and AR algorithms generate samples that don't accurately approximate the full-posterior distribution when the subposterior distributions have different variances. Both PIE and AR assume the variance is roughly equal across batches, so the inaccuracy in the measures is not surprising. These methods should not be applied when the subposterior distributions have different variance structures. Whilst the GB algorithm does account for different variances it applies too much weight to samples associated with high-variance subposteriors distributions, and not enough to samples associated with low-variance subposterior distributions.

Table 3.1 shows the benefit of MarV over the SKDE approach when the true subposterior distributions are close, or are, Gaussian. Since MarV only applies the KDE approximation to a lower-dimensional parameter rather than the full parameter vector we are able to achieve a higher accuracy when the subposterior distributions associated with the θ parameter are Gaussian.

3.3.2 Logistic Regression

Simulated - Rare Data

In this example a simulated data set has been used as a proxy for online advertising data, similar to the example shown in Section 4.3 in Scott et al. (2016). Often in these models interest will lie in how parameters influence an action. For example, does the presence of an advert result in a visit to the advertised website? This example demonstrates how the methods perform when the different marginal subposterior distributions for a single parameter have quite drastically different shapes, variances and locations.

Table 3.2: Relative frequency of covariates being active with the values of the corresponding true generating parameters for a given subject $i \in \{1, \ldots, N\}$.

j	1	2	3	4	5
$\mathbb{P}(x_{i,j} = 1)$	1	0.2	0.3	0.5	0.001
$\vartheta_{T,j}$	-3	1.2	-0.5	0.8	3

The likelihood function for this model takes the form:

$$p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\vartheta}) \propto \prod_{i=1}^{N} p_i^{y_i}(1 - p_i)^{1-y_i}, \tag{3.25}$$

where $\boldsymbol{\vartheta}$ are the parameters of the model, d denotes the dimension of $\boldsymbol{\vartheta}$, $\boldsymbol{X}$ is a $N \times d$ matrix of known binary covariates, N denotes the number of observations, $\boldsymbol{y}$ is a length N vector of binary indicators for success (1) or failure (0) and

$$p_i = \text{logit}^{-1}\left(\boldsymbol{x}_i^\top \boldsymbol{\vartheta}\right),$$

where $\boldsymbol{x}_i$ is the i-th row of $\boldsymbol{X}$.

There are 5 binary covariates, meaning that any element in $\boldsymbol{X}$ can only take the value 0 or 1. Table 3.2 shows the probability, for each subject i, of each covariate taking the value 1 and the corresponding true parameter values. The most interesting parameter is ϑ_5; it is rare but highly predictive when present. Weakly-informative, independent Gaussian priors were chosen for $\boldsymbol{\vartheta}$ so that

$$\pi(\boldsymbol{\vartheta}) \propto \prod_{j=1}^{d} \mathcal{N}(\vartheta_j|0, 1000)$$

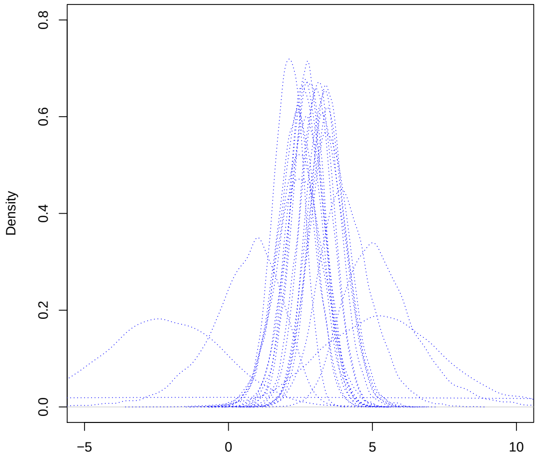


Figure 3.4: 20 subposteriors for the marginals of $\vartheta_5|y_b$ for each $b \in \{1 \ldots, B\}$, each receiving 4000 data points for the simulated rare-data Example.

When partitioning the data some batches will have very little information, or no information pertaining to ϑ_5, with only the partial prior information given to that batch. Even if the batches do have information on ϑ_5 there may still be issues if the responses are all equal. For example, suppose that if whenever $x_{i,5} = 1$ for a subject i assigned to a given batch b the corresponding response, y_i is equal to 1. The marginal of $\vartheta_5|\mathbf{y}_b$ would have a distribution with large variance and would be positively skewed. This means that even if our number of observation N is relatively large, due to the low probability of $\mathbb{P}(x_{i,5} = 1)$, the marginals of $\pi_b(\vartheta_5|\mathbf{y})$ can differ significantly. Figure 3.4 illustrates this, displaying subposterior distributions with widely differing means, standard deviations and skew. For this reason we set $\theta = \vartheta_5$ and $\boldsymbol{\psi} = \boldsymbol{\vartheta}_{-5}$.

In the simulation study we generated a total of $N = 100,000$ data points and split across $B = 25$ batches, giving each batch an equal portion of the data *i.e.*, $n_b = 5,000$. The priors were split evenly across the each subposterior. We drew $10,000$ MCMC samples from the full-data posterior distribution and from each subposterior and inflated-subposterior distribution, all using HMC implemented in STAN (Stan Development Team (2016)). For MarV and PIE $10,000$ samples were proposed and obtained. With these parameters the data was simulated and partitioned five times.

Table 3.2 shows the results from the simulation study. The differing variances affect AR and PIE's estimates as they have no weighting to account for the differing variances across subposteriors. Consensus, although only exact in the Gaussian case, somewhat surprisingly accounts for the skewness displayed in the subposterior distributions and captures the full-posterior distribution's behaviour well. MarV on average is slightly better than the Consensus Algorithm, but the difference is negligible as each consensus estimate is within a few standard errors. The SKDE method performs reasonably, but not quite as well as MarV or Consensus. The Gaussian Barycenter method struggles, similarly to AR and PIE, likely due to it struggling with subposterior distributions with differing variance structures.

Real Data - Hepmass

In the other experiments in this section MarV has been used to generate samples from a marginal of the posterior. However, when presenting MarV we stated it

Table 3.3: Mean (standard deviation) for the Rare-Data experiment, for each measure, over 5 runs. The metrics are as follows: standardised mean (SM), Skewness (SK), and Integrated absolute distance (IAD). The methods tested against are: Marginal Views (MV), The Consensus Algorithm (Cons), Semi-Parametric Kernel Density Estimation (SKDE), Posterior Interval Estimation (PIE), Gaussian Barycenter (GB), and Average Re-centering (AR).

	SM	**SK**	**IAD**
Marv	**0.38 (0.29)**	0.17 (0.18)	**0.18 (0.09)**
Cons	0.47 (0.35)	0.05 (0.03)	0.19 (0.11)
SKDE	1.24 (1.12)	3.59 (1.65)	0.50 (0.27)
PIE	5.45 (3.83)	0.233 (0.09)	0.88 (0.03)
GB	5.43 (4.10)	**0.02 (0.02)**	0.90 (0.03)
AR	5.44 (3.91)	0.51 (0.37)	0.89 (0.16)

was possible to estimate expectations from a transformation of parameters. In this example we will use MarV to generate samples from an approximation to the predictive distribution. This can be done without changing our sampling procedure for each subposterior distribution, *i.e.*, as long you have samples from each subposterior distribution it is possible to transform the samples afterwards, on a single core, to estimate expectations of transformations of the parameters.

This example uses the Hepmass (UCI Machine Learning repository, 2020) dataset, which contains a binary response $\mathbf{y}$ and $d_{\boldsymbol{\vartheta}} = 28$ covariates across $N = 1,000,000$ entries. This dataset was used by Baldi et al. (2016), the response is an indicator for whether a signal was indicative of an exotic particle being present as opposed to background noise. The data set contains 27 features, which we augmented with an intercept term for a total of $d_{\boldsymbol{\vartheta}} = 28$ parameters.

We used a logistic-regression model to classify whether a collision i is producing particles ($y_i = 1$) or not ($y_i = 0$), using the $d_{\boldsymbol{\vartheta}}$ covariate information. Hence, the same parametric form of the likelihood function as Section 3.3.2 was assumed, explicitly stated in (3.25). Independent priors were fitted to each of the parameters with the full prior being specified as follows:

$$\pi(\boldsymbol{\vartheta}) \propto \prod_{j=1}^{d_{\boldsymbol{\vartheta}}} \mathcal{N}(\vartheta_j | 0, 1000).$$

To estimate the predictive distribution we set $\theta = x_i^\top \vartheta$ for some given covariate information $\mathbf{x}_i$ *i.e.*, θ is the linear predictor for subject i. This demonstrates how MarV can be applied to transformations, not just marginals. We set $\boldsymbol{\psi} = \vartheta_{-1}$, we have to drop one of the parameters of $\boldsymbol{\vartheta}$ so that our parameters are identifiable. The specific form of $\boldsymbol{\psi}$ is chosen for ease, and by no means is shown to be optimal; we could have dropped any of the parameters, we were not limited to the intercept.

The data was uniformly spilt across $B = 20$ batches, meaning each batch received $n_b = 50,000$ data points. We drew $10,000$ samples from each subposterior and inflated-subposterior distribution. The data was partitioned randomly five times, and then for each of the partitions one of the covariate subjects were chosen to obtain our θ samples, for ease for run i the i-th subject was chosen. These were compared with the corresponding predictive distributions calculated from the samples generated from the full posterior.

Table 3.4 shows that all of the methods characterise each of the distributions

Table 3.4: Hepmass measure results for the five partitions of the data. Mean results for each measure with estimated standard error in brackets. The metrics are as follows: standardised mean (SM), Skewness (SK), and Integrated absolute distance (IAD). The methods tested against are: Marginal Views (MV), The Consensus Algorithm (Cons), Semi-Parametric Kernel Density Estimation (SKDE), Posterior Interval Estimation (PIE), Gaussian Barycenter (GB), and Average Re-centering (AR).

	SM	**SK**	**IAD**
Marv	**0.04** (0.02)	0.10 (0.12)	0.06 (<0.01)
Cons	**0.04** (0.02)	0.03 (0.02)	**0.03** (< 0.01)
SKDE	0.05 (0.02)	0.02 (0.02)	**0.03** (< 0.01)
PIE	0.07 (0.07)	0.02 (0.01)	**0.03** (0.02)
GB	0.07 (0.08)	**0.01** (0.01)	0.04 (0.02)
AR	0.07 (0.07)	0.02 (0.02)	**0.03** (0.02)

well. Figure 3.5 is a typical plot for each of the partitions. Each method is likely performing well due the large amount of data used to form each subposterior distribution. Each of the subposteriors are close to Gaussian with similar variance structures. However this example successfully demonstrates MarV's ability to be used to estimate expectations of transformed parameters.

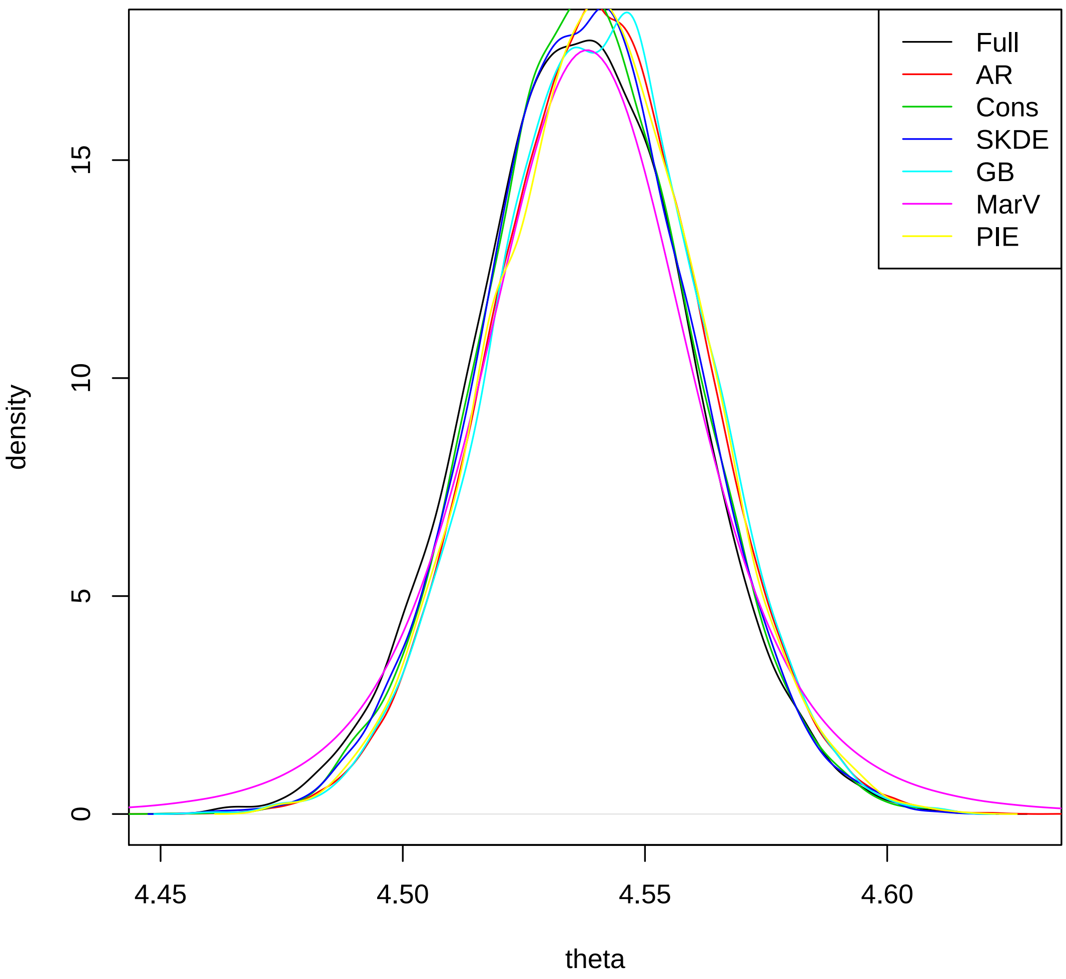


Figure 3.5: Estimated density plots of θ for a single run for the predictive Hepmass example. MarV has

3.3.3 Hierarchical Model - Mixed Effects Models

We previously looked at a simple linear model in Section 3.3.1 and a GLM extension in the form of a logistic-regression model in Section 3.3.2. Another natural extension is to introduce random effects alongside fixed effects. These models are of particular use when the data has a heirachical dependency structure, for example, to cluster student test scores based on classroom.

Let $y_{i,j} \in \mathcal{Y} \subseteq \mathbb{R}$ (for $i = 1, \ldots, n_j$, and $j = 1, \ldots, n$) be the response variable, where n_j is the number of observations for group j and n is the number of groups. The fixed and random effects are $\mathbf{x}_{i,j} \in \mathcal{X} \subseteq \mathbb{R}^{d_\beta}$ and $\mathbf{z}_j \in \mathcal{Z} \subseteq \mathbb{R}^{d_\alpha}$, respectively, and are related to the response variable by

$$y_{i,j}|\boldsymbol{\beta}, \boldsymbol{\alpha}_j, 1 \sim \text{Logistic }(\mathbf{x}_{i,j}\boldsymbol{\beta} + \mathbf{z}_j\boldsymbol{\alpha}_j, 1), \quad \boldsymbol{\alpha}_i \sim \mathcal{N}_r(0, \Sigma),$$

where $\boldsymbol{\beta} \in \mathbb{R}^p$ and $\boldsymbol{\alpha}_i \in \mathbb{R}^r$ are the fixed and random effect model coefficients and the $\mathsf{Logistic}(a, 1)$ distribution has a cumulative distribution function of $1/(1+\exp[-(x-a)])$. Our parameters of interest are then $\boldsymbol{\vartheta} = (\boldsymbol{\beta}, \boldsymbol{\alpha}, \Sigma)$, where Σ represents the variance of the random effects. *A priori* we assume an inverse-Wishart distribution for Σ, $\Sigma \sim \mathcal{W}^{-1}(\nu, S)$, with $\nu = 5$ and $S = 5I_r$, and *a priori* we assumed $\boldsymbol{\beta} \sim \mathcal{N}_p(0, 1000I_p)$.

We simulated a dataset that contains $200,000$ observations. We set the number of groups $n = 2,000$ and the number observations for each group $n_j = 100$, for

$j = 1, \ldots, n$. The number of parameters for the fixed effects were set to $d_\beta = 10$, with $\beta_{0,i} = (-1)^{(i-1)}$. The number of parameters for each random effect was set to be $d_\alpha = 2$, and we set

$$\Sigma_0 = \begin{bmatrix} 1 & 0.2 \\ 0.2 & 1 \end{bmatrix},$$

then α_i were simulated independently from a $\mathcal{N}_{d_\alpha}(0, \Sigma_0)$ distribution. We included an intercept term, that is $x_{i,j,1} = 1$ for all i, j, otherwise $\mathbf{x}_{i,j}$ and $\mathbf{z}_j$ were simulated from independent $\mathsf{Bernoulli}(0.5)$ distributions.

The data were randomly partitioned into $B = 10$ batches by group, so that each group only belonged to one batch. This chosen so that each batch had to only estimate the 200 $\boldsymbol{\alpha}$ parameters, rather than full 2,000. With a Gaussian observation model for each $y_{i,j}$, marginalisation over all of the random effects would be tractable. The logistic observation model necessitates the use of a sampling scheme such as MCMC.

We used the STAN software to sample from the full-posterior distribution and subposterior distributions generating $M = 5,000$ MCMC samples after an initial 1,000 sample burn in. Table 3.5 gives the discrepancy measures for each of the merging algorithms, averaged over 5 random partitions of the data. For ease we set $\theta = \beta_1$, the first non-intercept covariate coefficient and $\boldsymbol{\psi} = \boldsymbol{\vartheta}_{-1}$, where $\boldsymbol{\vartheta}$ does not include the $\boldsymbol{\alpha}$ parameters. This is possible since for each group, only one batch has information on that group. Therefore there is no need to pool information on the

Table 3.5: Discrepancy measures for the linear mixed effects model on the simulated dataset. These measures were averaged over 5 random partitions of the data. Estimated standard errors are given in brackets. The metrics are as follows: standardised mean (SM), Skewness (SK), and Integrated absolute distance (IAD). The methods tested against are: Marginal Views (MV), The Consensus Algorithm (Cons), Semi-Parametric Kernel Density Estimation (SKDE), Posterior Interval Estimation (PIE), Gaussian Barycenter (GB), and Average Re-centering (AR).

Algorithm	SM	Skew	IAD
MarV	0.29 (0.20)	0.28 (0.16)	0.17 (0.04)
Cons	0.08 (0.07)	0.08 (0.05)	0.04 (0.02)
SKDE	0.08 (0.06)	**0.05** (0.03)	0.04 (0.02)
PIE	0.02 (0.01))	**0.05** (0.03)	0.03 (0.01)
GB	0.02 (0.01)	**0.05** (0.03)	**0.02** (<0.01)
AR	**0.01** (0.01)	0.06 (0.01)	**0.02** (<0.01)

$\boldsymbol{\alpha}$ parameters.

The results in Table 3.5 show that MarV cannot characterise the posterior as well as the other algorithms. Figure 3.6 shows the merged posterior density plots for θ for a single partition. Although MarV is to the left of the others, it still shows MarV has been able to capture the main characteristics of the full-data posterior distribution.

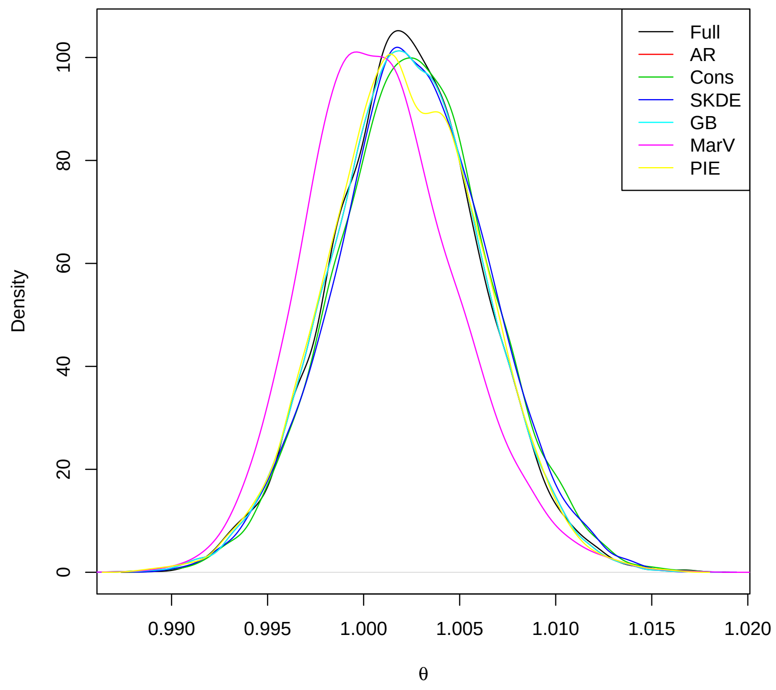


Figure 3.6: Marginal density plot for θ for a single run in the LME experiment.

3.3.4 Simulated data - Multi-modal distribution

In this Section we will show that MarV can be effective in a multi-modal setting. Multi-modal targets are often difficult to characterise, and sometimes it can be difficult to generate samples that represent the posterior. In previous sections we showed MarV can capture Gaussian behaviour as well as the Consensus algorithm, and here we will show it can capture multi-modal behaviour like methods that have less-strict parametric assumptions.

Before using any of the divide-and-conquer methods, it is important to ensure each sampler for each subposterior, or inflated-subposterior, distribution has fully converged, and explored each mode. None of the combination methods can account for the MC algorithms not having converged to the correct target distribution. We will explain how we sampled from the multi-modal distributions in this example, but we will not be covering how to sample from multi-modal distributions in general.

For a given $b \in \{1, \dots, B = 10\}$ and $\sigma = 1/10$ we define the following subposterior:

$$\pi_b(\theta|\mathbf{y}_b) := \frac{1}{2}\mathcal{N}_2(\mu_{1,b}, \sigma_\theta^2) + \frac{1}{2}\mathcal{N}_2(\mu_{2,b}, \sigma^2),$$

where $\mu_{1,b} \sim \mathcal{N}(0, 1/100)$ and $\mu_{2,b} \sim \mathcal{N}_2(1, 1/100)$. Each $\pi_b(\theta|\mathbf{y}_b)$ subposterior distribution is a mixture of two Gaussian distributions. We set

$$\pi_b(\boldsymbol{\psi}|\mathbf{y}_b) :\propto \prod_{i=2}^{d_{\boldsymbol{\vartheta}}} \mathcal{N}(\psi_i|\mu_{\boldsymbol{\psi},i}, \sigma),$$

where $\mu_{\boldsymbol{\psi},i} = (-1)^{(i-1)}$ for $i \in \{2, \dots, d_\vartheta = 10\}$. Since the focus of this example is multi-modal behaviour, for ease we assume independence between θ and $\boldsymbol{\psi}$, so that:

$$\pi(\boldsymbol{\vartheta}|\mathbf{y}) = \prod_{b=1}^{B} \pi_b(\theta|\mathbf{y}_b)\pi_b(\boldsymbol{\psi}|\mathbf{y}_b).$$

Since $\boldsymbol{\psi}$ and θ are independent, we can sample from the distributions for $\boldsymbol{\psi}$ and θ independently. Sampling from the $\pi_b(\boldsymbol{\psi}|\mathbf{y}_b)$ subposterior distributions are trivial since they are Gaussian.

Formulating a sampler for each marginal-subposterior distribution $\pi_b(\theta|\mathbf{y}_b)$ was done in two parts. For each sample draw u from a uniform(0,1) then:

- If $u < 1/2$ draw $\theta|\mathbf{y}_b$ from $\mathcal{N}(\mu_{1,b}, \sigma)$,
- If $u \geq 1/2$ draw $\theta|\mathbf{y}_b$ from $\mathcal{N}(\mu_{2,b}, \sigma)$.

We used an MCMC algorithm with an independent proposal for the inflated-subposterior distributions, $\pi_b(\theta|\mathbf{y}_b)^B$, and the full-posterior distribution, $\pi(\theta|\mathbf{y})$. Although both the inflated-subposterior distributions and full-posterior distributions are a mixture of Gaussian distributions, the inflated-subposteriors are each a mixture of 11 unique Gaussian distributions (assuming $\mu_{1,b} \neq \mu_{2,b}$), and the full-posterior distribution is a mixture of 2^{11} unique Gaussian distributions. Hence, we decided it was easier to formulate an MCMC sampler rather than calculating each of the Gaussian mixture components.

The proposal sampler used for the MCMC algorithm was a mixture of t-distributions with degrees of freedom 2 with proposal density, denoted $q_b(\theta|\mu_{1,b}, \mu_{2,b}, \sigma)$, used for each inflated-subposterior distribution:

$$q_b(\theta|\mu_{1,b}, \mu_{2,b}, \sigma_\theta) = \frac{1}{2}\mathrm{t}_2(\theta|\mu_{1,b}, \sigma/\sqrt(B)) + \frac{1}{2}\mathrm{t}_2(\theta|\mu_{2,b}, \sigma/\sqrt(B)).$$

Sampling from the mixture q_b was analogous to sampling from $\pi_b(\boldsymbol{\psi}|\mathbf{y}_b)$, but a t-distribution was sampled from instead of a Gaussian. Let $\bar{\mu}_i = \sum_{b=1}^{B} \mu_{i,b}/B$, for

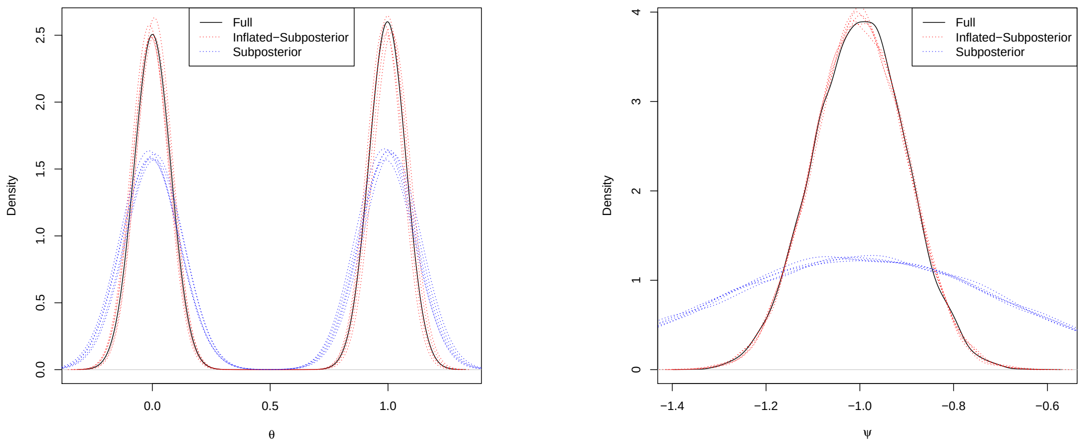
$i \in \{1, 2\}$, we set the proposal density for the full-posterior to be

$$q(\theta|\mu_{1,b}, \mu_{2,b}, \sigma_\theta) = \frac{1}{2}\mathrm{dt}(\theta|\bar{\mu}_1, \sigma_\theta) + \frac{1}{2}\mathrm{dt}(\theta|\bar{\mu}_2, \sigma_\theta/B).$$

For each inflated and non-inflated subposterior density, and the full-posterior density 10,000 samples were drawn. For the subposterior distributions these samples were exact. In contrast, we used an MCMC algorithm to sample from the inflated-subposterior distributions and full-data posterior distribution. Therefore, for a fair comparison when using the independence samplers 100,000 samples were drawn and thinned to 10,000. Once these samples were obtained each of the merging methods could be applied to obtain samples from an approximation to the full-data posterior distribution. This experiment was repeated 5 times.

Figure 3.7 (a) shows full-posterior, subposterior, and inflated-subposterior densities for θ plotted for comparison, for a single run. The θ densities look as we expected; two modes centred at $0, 1$ for each posterior, the variance of the inflated-subposterior distributions are similar to the full-posterior distribution, and the subposterior densities have a lower variance than the other posterior densities. Figure 3.7 (b) shows the full-posterior, subposterior, and inflated-subposterior densities plots for ψ_1 for a single run, again this looks as expected. Both these plots are typical of the plots for each of the runs.

The results for the 5 runs are recorded in Table 3.6, and a density plot showing



(a) Density plots for θ for a single run of the multi-modal experiment.

(b) Density plots for ψ_1 for a single run of the multi-modal experiment.

Figure 3.7

the density estimates for each of the methods for a single run is given in Figure 3.8. The results for SM and SKEW were omitted as they can be misleading; for each of these measures a method can score quite well, but the relevant estimated densities have almost no overlap with the estimated full-data posterior density.

Table 3.6 shows that MarV is middling. Figure 3.8 shows that MarV has correctly identified the modes, as have the PIE, GB, and AR methods. This is in contrast to the Consensus and SKDE methods, which incorrectly identity a single mode. Overall MarV is performing well, but is underestimating the variance relative to the PIE and AR methods. This is likely due to the PIE and AR methods not having to estimate the variance, coupled with the subposterior densities for this example all having the same variance. Overall we have shown that MarV can correctly characterise multi-modal densities in this example.

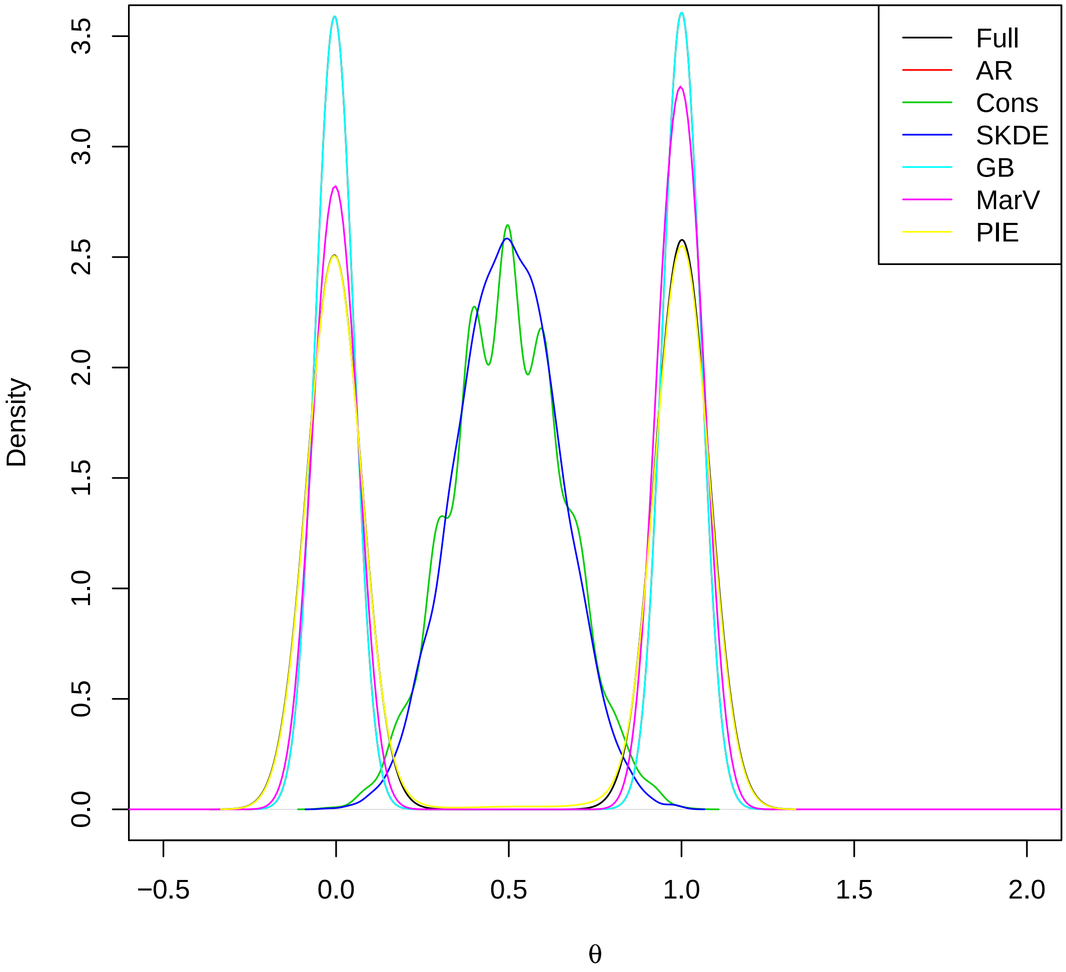


Figure 3.8: Estimated density plots of θ. PIE, MarV, GB, and AR capture the multi-modality of the posterior distribution, whilst the SKDE and Consensus methods estimate the density to have a single mode.

Table 3.6: IAD results for the multi-modal experiment. The Integrated absolute distance (IAD) was averaged across 5 simulations of the data. Estimated standard errors are given in brackets. The methods tested against are: Marginal Views (MV), The Consensus Algorithm (Cons), Semi-Parametric Kernel Density Estimation (SKDE), Posterior Interval Estimation (PIE), Gaussian Barycenter (GB), and Average Re-centering (AR).

Method	**IAD**
MarV	0.08 (0.01)
Cons	0.89 (0.01)
SKDE	0.91 (0.01)
PIE	**0.01 ($<$ 0.01)**
GB	0.66 ($<$ 0.01)
AR	**0.01 ($<$ 0.01)**

3.4 Conclusions

We have introduced MarV, an importance sampler with a strong theoretical backing within the divide-and-conquer framework, used to obtain approximations to low-dimensional posterior expectations.

We have shown that MarV can correctly characterise a wide array of distributions. In Section 3.3.1, the tri-variate Gaussian example, MarV was comparable with Consensus Monte Carlo, the current gold standard for Gaussian experiments. The rare-data example introduced in Section 3.3.2 demonstrated that MarV could produce accurate estimates of expectations even when the subposterior distributions have different mean, variance, and skew. The real-world Hepmass data example introduced in Section 3.3.2 showed that MarV could be used to accurately estimate posterior expectations of transformed parameters. Section 3.3.3 showed MarV, whilst weaker than the other methods, was still capturing the majority of the behaviour of the posterior distribution. Finally in Section 3.3.4 we demonstrated that MarV could capture bimodal posterior distributions.

In all of these experiments MarV was able to capture the main features of the chosen marginal distribution, demonstrating MarV's robustness. This was not shown by any of the other methods compared to.

Appendix

3.A Conditional Gaussian Parameter Estimation

Define I_d to be the identity matrix with dimension $d \times d$ and $\mathbf{0}_{d_1,d_2}$ to be the $d_1 \times d_2$ dimensional matrix with all elements equal to zero.

Lemma 3.3. *Woodbury Matrix Identity - Higham (2002). Let A and B denote invertible matrices with dimensions $d_A \times d_A$ and $d_B \times d_B$ respectively. Let C be a $d_A \times d_B$ dimensional matrix and D to be a $d_B \times d_A$ matrix then:*

$$(A + CBD)^{-1} = A^{-1} - A^{-1}C\left(B^{-1} + DA^{-1}C\right)^{-1} DA^{-1},$$

provided that

$$\left(B^{-1} + DA^{-1}C\right)^{-1}$$

is non singular.

Proof.

$$
\begin{aligned}
&(A+CBD)\left(A^{-1}-A^{-1}C\left(B^{-1}+DA^{-1}C\right)^{-1}DA^{-1}\right)\\
=&I_{d_A}+CBDA^{-1}-\left(C+CBDA^{-1}C\right)\left(B^{-1}+DA^{-1}C\right)^{-1}DA^{-1},\\
=&I_{d_A}+CBDA^{-1}-CB\left(B^{-1}+DA^{-1}C\right)\left(B^{-1}+DA^{-1}C\right)^{-1}DA^{-1},\\
=&I_{d_A}+CBDA^{-1}-CBDA^{-1}\\
=&I_{d_A}
\end{aligned}
$$

□

Define V be a $d \times d$ symmetric positive definite matrix with inverse τ. Let $d > d_1, d_2 > 0$. We can partition V in the following way:

$$
V = \begin{pmatrix} V_{1,1} & V_{1,2} \\ V_{2,1} & V_{2,2} \end{pmatrix},
$$

where $V_{1,1}$ is a $d_1 \times d_1$ dimensional matrix with elements equal to $V_{[1:d_1,1:d_1]}$, $V_{2,2}$ is a $d_2 \times d_2$ dimensional matrix with elements equal to $V_{[(d_1+1):(d_1+d_2),(d_1+1):(d_1+d_2)]}$, $V_{2,1}$ is a $d_2 \times d_1$ dimensional matrix with elements equal to $V_{[(d_1+1):(d_2+d_1),1:d_1]}$ and $V_{1,2} = V_{2,1}^\top$. Partitioning τ we get:

$$
\tau = \begin{pmatrix} \tau_{1,1} & \tau_{1,2} \\ \tau_{2,1} & \tau_{2,2} \end{pmatrix},
$$

where $\tau_{1,1}, \tau_{1,2}, \tau_{2,1}$ and $\tau_{2,2}$ are defined in a similar way to $V_{1,1}$, $V_{1,2}, V_{2,1}$ and $V_{2,2}$ respectively.

Lemma 3.4. *Matrix block inversion - Bernstein (2009)*

$$\begin{aligned}
\tau_{1,1} &= \left(V_{1,1} - V_{1,2}V_{2,2}^{-1}V_{2,1}\right)^{-1} \\
\tau_{2,2} &= \left(V_{2,2} - V_{2,1}V_{1,1}^{-1}V_{1,2}\right)^{-1} \\
\tau_{1,2} &= -V_{1,1}^{-1}V_{1,2}\left(V_{2,2} - V_{2,1}V_{1,1}^{-1}V_{1,2}\right)^{-1} \\
\tau_{2,1} &= -V_{2,2}^{-1}V_{2,1}\left(V_{1,1} - V_{1,2}V_{2,2}^{-1}V_{2,1}\right)^{-1} = \tau_{1,2}^{\top}
\end{aligned}$$

Proof. Since V is the inverse of τ we can write:

$$\begin{pmatrix} I_{d_1} & \mathbf{0}_{d_1,d_2} \\ \mathbf{0}_{d_2,d_1} & I_{d_2} \end{pmatrix} = I_d = V\tau = \begin{pmatrix} V_{1,1}\tau_{1,1} + V_{1,2}\tau_{2,1} & V_{1,1}\tau_{1,2} + V_{1,2}\tau_{2,2} \\ V_{2,1}\tau_{1,1} + V_{2,2}\tau_{2,1} & V_{2,1}\tau_{1,2} + V_{2,2}\tau_{2,2} \end{pmatrix}. \tag{3.26}$$

This gives the following four equalities:

$$\begin{aligned}
I_{d_1} &= V_{1,1}\tau_{1,1} + V_{1,2}\tau_{2,1}, \\
\mathbf{0}_{d_2,d_1} &= V_{2,2}\tau_{2,1} + V_{2,1}\tau_{1,1}, \\
\mathbf{0}_{d_1,d_2} &= V_{1,1}\tau_{1,2} + V_{1,2}\tau_{2,2}, \\
I_{d_2} &= V_{2,2}\tau_{2,2} + V_{2,1}\tau_{1,2}.
\end{aligned}$$

Rearranging, and remembering that an inverse of a symmetric matrix is symmetric, gives the equalities:

$$\tau_{1,1} = V_{1,1}^{-1} - V_{1,1}^{-1} V_{1,2} \tau_{1,2}^{\top}, \tag{3.27}$$

$$\tau_{1,2}^{\top} = -V_{2,2}^{-1} V_{2,1} \tau_{1,1}, \tag{3.28}$$

$$\tau_{1,2} = -V_{1,1}^{-1} V_{1,2} \tau_{2,2}, \tag{3.29}$$

$$\tau_{2,2} = V_{2,2}^{-1} - V_{2,2}^{-1} V_{2,1} \tau_{1,2}. \tag{3.30}$$

Substituting (3.29) into (3.30) gives:

$$\begin{aligned} \tau_{2,2} &= V_{2,2}^{-1} + V_{2,2}^{-1} V_{2,1} V_{1,1}^{-1} V_{1,2} \tau_{2,2}, \\ \implies \tau_{2,2} &= \left(I_{d_2} - V_{2,2}^{-1} V_{2,1} V_{1,1}^{-1} V_{1,2} \right)^{-1} V_{2,2}^{-1}, \end{aligned}$$

applying the result of Lemma 3.3 we get

$$\begin{aligned} \tau_{2,2} &= \left(I_{d_2} + I_{d_2} V_{2,2}^{-1} V_{2,1} \left(V_{1,1} + V_{1,2} I_{d_2} V_{2,2} V_{2,1} \right)^{-1} V_{1,2}^{-1} I_{d_2} \right) V_{2,2}^{-1}, \\ &= V_{2,2}^{-1} - V_{2,2}^{-1} V_{2,1} \left(V_{1,1} + V_{1,2} V_{2,2}^{-1} V_{2,1} \right)^{-1} V_{1,2}^{-1} V_{2,2}^{-1}. \end{aligned}$$

Applying the result of Lemma 3.3 again gives the required result for $\tau_{2,2}$:

$$\tau_{2,2} = \left(V_{2,2} - V_{2,1} V_{1,1}^{-1} V_{1,2} \right)^{-1}. \tag{3.31}$$

Due to the symmetry between the $\tau_{1,1}$ and $\tau_{2,2}$, it is trivial to see substituting

(3.28) into (3.27) and following similar steps as above gives the required result for $\tau_{1,1}$:

$$\tau_{1,1} \left(V_{1,1} - V_{1,2}V_{2,2}^{-1}V_{2,1}\right)^{-1}. \qquad (3.32)$$

To see the required result for $\tau_{1,2}$ we substitute (3.31) into (3.29):

$$\tau_{1,2} = -V_{1,1}^{-1}V_{1,2} \left(V_{2,2} - V_{1,2}V_{2,2}^{-1}V_{2,1}\right)^{-1},$$

and the required result for $\tau_{2,1}$ is seen by substituting (3.32) into (3.28):

$$\tau_{2,1} = -V_{2,2}^{-1}V_{2,1} \left(V_{1,1} - V_{2,1}V_{1,1}^{-1}V_{1,2}\right)^{-1}.$$

□

For a given value of $\boldsymbol{\theta}$, denoted $\boldsymbol{\theta}^*$, and by using Bayes Theorem we can write:

$$\begin{aligned}\tilde{\mathrm{g}}(\boldsymbol{\psi}|\boldsymbol{\theta} = \boldsymbol{\theta}^*) &\propto G(\boldsymbol{\theta}^*, \boldsymbol{\psi}) \\ &\propto \exp\left\{-\frac{1}{2}\begin{pmatrix}\boldsymbol{\theta}^* - \mu_{\boldsymbol{\theta}} \\ \boldsymbol{\psi} - \mu_{\boldsymbol{\psi}}\end{pmatrix}^\top \begin{pmatrix}\tau_{\boldsymbol{\theta}} & \tau_{\boldsymbol{\theta},\boldsymbol{\psi}} \\ \tau_{\boldsymbol{\psi},\boldsymbol{\theta}} & \tau_{\boldsymbol{\psi}}\end{pmatrix}\begin{pmatrix}\boldsymbol{\theta}^* - \mu_{\boldsymbol{\theta}} \\ \boldsymbol{\psi} - \mu_{\boldsymbol{\psi}}\end{pmatrix}\right\}.\end{aligned}$$

Let

$$Q(\boldsymbol{\psi}) := \begin{pmatrix} \boldsymbol{\theta}^* - \mu_{\boldsymbol{\theta}} \\ \\ \boldsymbol{\psi} - \mu_{\boldsymbol{\psi}} \end{pmatrix}^\top \begin{pmatrix} \tau_{\boldsymbol{\theta}} & \tau_{\boldsymbol{\theta},\boldsymbol{\psi}} \\ \\ \tau_{\boldsymbol{\psi},\boldsymbol{\theta}} & \tau_{\boldsymbol{\psi}} \end{pmatrix} \begin{pmatrix} \boldsymbol{\theta}^* - \mu_{\boldsymbol{\theta}} \\ \\ \boldsymbol{\psi} - \mu_{\boldsymbol{\psi}} \end{pmatrix},$$

so that

$$\tilde{\mathrm{g}}(\boldsymbol{\psi}|\boldsymbol{\theta} = \boldsymbol{\theta}^*) \propto \exp\left\{-\frac{1}{2}Q(\boldsymbol{\psi})\right\}.$$

Expanding $Q(\boldsymbol{\psi})$ gives:

$$\begin{aligned} Q(\boldsymbol{\psi}) = & \left(\boldsymbol{\psi} - \mu_{\boldsymbol{\psi}}\right)^\top \tau_{\boldsymbol{\psi}} \left(\boldsymbol{\psi} - \mu_{\boldsymbol{\psi}}\right) \\ & + \left(\boldsymbol{\psi} - \mu_{\boldsymbol{\psi}}\right)^\top \tau_{\boldsymbol{\psi},\boldsymbol{\theta}} \left(\boldsymbol{\theta} - \mu_{\boldsymbol{\theta}}\right) \\ & + \left(\boldsymbol{\theta} - \mu_{\boldsymbol{\theta}}\right)^\top \tau_{\boldsymbol{\theta},\boldsymbol{\psi}} \left(\boldsymbol{\psi} - \mu_{\boldsymbol{\psi}}\right) \\ & + \left(\boldsymbol{\theta} - \mu_{\boldsymbol{\theta}}\right)^\top \tau_{\boldsymbol{\theta}} \left(\boldsymbol{\theta} - \mu_{\boldsymbol{\theta}}\right), \\ = & \boldsymbol{\psi}^\top \tau_{\boldsymbol{\psi}} \boldsymbol{\psi} \\ & - \boldsymbol{\psi}^\top \left(\tau_{\boldsymbol{\psi}} \mu_{\boldsymbol{\psi}} - \tau_{\boldsymbol{\theta},\boldsymbol{\psi}}^\top \left(\boldsymbol{\theta} - \mu_{\boldsymbol{\theta}}\right)\right) \\ & - \left(\tau_{\boldsymbol{\psi}} \mu_{\boldsymbol{\psi}} - \tau_{\boldsymbol{\theta},\boldsymbol{\psi}}^\top \left(\boldsymbol{\theta} - \mu_{\boldsymbol{\theta}}\right)\right)^\top \boldsymbol{\psi} \\ & + C, \end{aligned}$$

Using Lemma 3.5 we know

$$\begin{aligned}\tau_{\boldsymbol{\psi}} &= \left(V_{\boldsymbol{\psi}} - V_{\boldsymbol{\psi},\boldsymbol{\theta}} V_{\boldsymbol{\theta}}^{-1} V_{\boldsymbol{\theta},\boldsymbol{\psi}}\right)^{-1}, \\ \tau_{\boldsymbol{\theta},\boldsymbol{\psi}} &= -V_{\boldsymbol{\theta}}^{-1} V_{\boldsymbol{\theta},\boldsymbol{\psi}} \left(V_{\boldsymbol{\psi}} - V_{\boldsymbol{\psi},\boldsymbol{\theta}} V_{\boldsymbol{\theta}}^{-1} V_{\boldsymbol{\theta},\boldsymbol{\psi}}\right)^{-1} = V_{\boldsymbol{\psi}}^{-1} V_{\boldsymbol{\psi},\boldsymbol{\theta}} \tau_{\boldsymbol{\psi}},\end{aligned}$$

hence

$$\tau_{\boldsymbol{\theta},\boldsymbol{\psi}}^{\top} = \tau_{\boldsymbol{\psi}} V_{\boldsymbol{\theta},\boldsymbol{\psi}} V_{\boldsymbol{\psi}}^{-1}.$$

Let

$$\mu_{\boldsymbol{\psi}|\boldsymbol{\theta}} := \mu_{\boldsymbol{\psi}} + V_{\boldsymbol{\theta},\boldsymbol{\psi}} V_{\boldsymbol{\psi}}^{-1} \left(\boldsymbol{\theta} - \mu_{\boldsymbol{\theta}}\right)$$

Now we can factorise $Q(\boldsymbol{\psi})$, only taking note of terms including $\boldsymbol{\psi}$.

$$\begin{aligned}Q(\boldsymbol{\psi}) =& \boldsymbol{\psi}^{\top} \tau_{\boldsymbol{\psi}} \boldsymbol{\psi} \\ & - \boldsymbol{\psi}^{\top} \tau_{\boldsymbol{\psi}} \left(\mu_{\boldsymbol{\psi}} + V_{\boldsymbol{\theta},\boldsymbol{\psi}} V_{\boldsymbol{\psi}}^{-1} \left(\boldsymbol{\theta} - \mu_{\boldsymbol{\theta}}\right)\right) \\ & - \left(\mu_{\boldsymbol{\psi}}^{\top} + \left(\boldsymbol{\theta} - \mu_{\boldsymbol{\theta}}\right) V_{\boldsymbol{\psi}}^{-1} V_{\boldsymbol{\psi},\boldsymbol{\theta}}\right) \tau_{\boldsymbol{\psi}} \boldsymbol{\psi} \\ & + C, \\ =& \left(\boldsymbol{\psi} - \mu_{\boldsymbol{\psi}|\boldsymbol{\theta}}\right)^{\top} \tau_{\boldsymbol{\psi}} \left(\boldsymbol{\psi} - \mu_{\boldsymbol{\psi}|\boldsymbol{\theta}}\right) - \mu_{\boldsymbol{\psi}|\boldsymbol{\theta}}^{\top} \tau_{\boldsymbol{\psi}} \mu_{\boldsymbol{\psi}|\boldsymbol{\theta}} + C.\end{aligned}$$

We substitute this form for $Q(\boldsymbol{\psi})$ back into $\tilde{\mathrm{g}}(\boldsymbol{\psi}|\boldsymbol{\theta}=\boldsymbol{\theta}^*)$:

$$\begin{aligned}
\tilde{\mathrm{g}}(\boldsymbol{\psi}|\boldsymbol{\theta}=\boldsymbol{\theta}^*) &\propto \exp\left\{-\frac{1}{2}Q(\boldsymbol{\psi})\right\}, \\
&= \exp\left\{-\frac{1}{2}\left(\left(\boldsymbol{\psi}-\mu_{\boldsymbol{\psi}|\boldsymbol{\theta}}\right)^\top \tau_{\boldsymbol{\psi}}\left(\boldsymbol{\psi}-\mu_{\boldsymbol{\psi}|\boldsymbol{\theta}}\right) - \mu_{\boldsymbol{\psi}|\boldsymbol{\theta}}^\top \tau_{\boldsymbol{\psi}}\mu_{\boldsymbol{\psi}|\boldsymbol{\theta}} + C\right)\right\}, \\
&\propto \exp\left\{-\frac{1}{2}\left(\boldsymbol{\psi}-\mu_{\boldsymbol{\psi}|\boldsymbol{\theta}}\right)^\top \tau_{\boldsymbol{\psi}}\left(\boldsymbol{\psi}-\mu_{\boldsymbol{\psi}|\boldsymbol{\theta}}\right)\right\}.
\end{aligned}$$

This is recognisable as a kernel of a Gaussian distribution with mean

$$\mu_{\boldsymbol{\psi}|\boldsymbol{\theta}} = \mu_{\boldsymbol{\psi}} + V_{\boldsymbol{\theta},\boldsymbol{\psi}}V_{\boldsymbol{\psi}}^{-1}\left(\boldsymbol{\theta}-\mu_{\boldsymbol{\theta}}\right),$$

and variance

$$\tau_{\boldsymbol{\psi}}^{-1} = V_{\boldsymbol{\psi}} - V_{\boldsymbol{\psi},\boldsymbol{\theta}}V_{\boldsymbol{\theta}}^{-1}V_{\boldsymbol{\theta},\boldsymbol{\psi}},$$

showing the result. Eaton (1983).

Define V to be a $d \times d$ symmetric positive-definite matrix. Let $d > d_1, d_2 > 0$. We can partition V in the following way:

$$V = \begin{pmatrix} V_{1,1} & V_{1,2} \\ V_{2,1} & V_{2,2} \end{pmatrix},$$

where $V_{1,1}$ is a $d_1 \times d_1$ dimensional matrix with elements equal to $V_{[1:d_1,1:d_1]}$, $V_{2,2}$ is a $d_2 \times d_2$ dimensional matrix with elements equal to $V_{[(d_1+1):(d_1+d_2),(d_1+1):(d_1+d_2)]}$, $V_{2,1}$ is a $d_2 \times d_1$ dimensional matrix with elements equal to $V_{[(d_1+1):(d_2+d_1),1:d_1]}$ and

$V_{1,2} = V_{2,1}^{\top}$. With the same partitioning as for V, we define:

$$V^{-1} := \tau = \begin{pmatrix} \tau_{1,1} & \tau_{1,2} \\ \\ \tau_{2,1} & \tau_{2,2} \end{pmatrix},$$

where $\tau_{1,1}, \tau_{1,2}, \tau_{2,1}$ and $\tau_{2,2}$ are defined in a similar way to $V_{1,1}$, $V_{1,2}, V_{2,1}$ and $V_{2,2}$ respectively.

Lemma 3.5.

$$\begin{aligned}
\tau_{1,1} &= \left(V_{1,1} - V_{1,2}V_{2,2}^{-1}V_{2,1}\right)^{-1} \\
\tau_{1,2} &= -V_{1,1}^{-1}V_{1,2}\left(V_{2,2} - V_{2,1}V_{1,1}^{-1}V_{1,2}\right)^{-1} \\
\tau_{2,1} &= -V_{2,2}^{-1}V_{2,1}\left(V_{1,1} - V_{1,2}V_{2,2}^{-1}V_{2,1}\right)^{-1} = \tau_{1,2}^{\top}
\end{aligned}$$

$$\mu_{\boldsymbol{\psi}} := \mathbb{E}_G\left[\boldsymbol{\psi}\right] \quad \text{and} \quad V_{\boldsymbol{\psi}} := \mathrm{Var}_G\left[\boldsymbol{\psi}\right],$$

and define $\mu_{\boldsymbol{\theta}}$ and $V_{\boldsymbol{\theta}}$ in a similar fashion. Define

$$V_G := \mathrm{Var}_G\left[\boldsymbol{\theta}, \boldsymbol{\psi}\right].$$

Let $V_{\boldsymbol{\psi},\boldsymbol{\theta}}$ be the the $d_{\boldsymbol{\psi}} \times d_{\boldsymbol{\theta}}$ matrix that satisfies

$$V_{G(\cdot)} = \begin{pmatrix} V_{\boldsymbol{\theta}} & V_{\boldsymbol{\psi},\boldsymbol{\theta}}^{\top} \\ \\ V_{\boldsymbol{\psi},\boldsymbol{\theta}} & V_{\boldsymbol{\psi}} \end{pmatrix},$$

i.e., the covariance between $\boldsymbol{\psi}$ and $\boldsymbol{\theta}$ under the Gaussian approximation characterised by distribution g. Let $V_{\boldsymbol{\theta},\boldsymbol{\psi}} := V_{\boldsymbol{\psi},\boldsymbol{\theta}}^{\top}$. Define $\tau_G := V_G^{-1}$. Now we partition τ_G:

$$\tau_G = \begin{pmatrix} \tau_{\boldsymbol{\theta}} & \tau_{\boldsymbol{\theta},\boldsymbol{\psi}} \\ \\ \tau_{\boldsymbol{\psi},\boldsymbol{\theta}} & \tau_{\boldsymbol{\psi}} \end{pmatrix},$$

with the definitions for $\tau_{\boldsymbol{\theta}}, \tau_{\boldsymbol{\theta},\boldsymbol{\psi}}, \tau_{\boldsymbol{\psi},\boldsymbol{\theta}}$ and $\tau_{\boldsymbol{\psi}}$ mirroring those for $V_{\boldsymbol{\theta}}, V_{\boldsymbol{\theta},\boldsymbol{\psi}}, V_{\boldsymbol{\psi},\boldsymbol{\theta}}$ and $V_{\boldsymbol{\psi}}$ respectively.

3.B Partitioning a Wishart Prior

Let $\boldsymbol{\theta} \sim \mathcal{W}_d(S, \nu)$.

$$\begin{aligned}
\log\left(\pi(\boldsymbol{\theta})^{1/B}\right) &\propto \frac{(\nu - d - 1)\log|\boldsymbol{\theta}| - \operatorname{trace}\left(S^{-1}\boldsymbol{\theta}\right)}{2B} \\
&= \frac{(\nu/B - d/B - 1/B)\log|\boldsymbol{\theta}| - \operatorname{trace}\left(\frac{S^{-1}}{B}\boldsymbol{\theta}\right)}{2} \\
&= \frac{\left(\frac{\nu+(d+1)(B-1)}{B}\right) - d - 1)\log|\boldsymbol{\theta}| - \operatorname{trace}\left((BS)^{-1}\boldsymbol{\theta}\right)}{2}
\end{aligned}$$

hence if $\pi(\boldsymbol{\theta})$ is the pdf of a Wishart distribution with scale matrix S and degrees of freedom ν then $\pi(\boldsymbol{\theta})^{1/B}$ is proportional to the pdf of a Wishart distribution with scale matrix BS and degrees of freedom $\frac{\nu+(d+1)(B-1)}{B}$.

Restrictions: S is positive definite, no problems since BS will be positive definite. $\nu > d-1$. Ensure $\frac{\nu+(d+1)(B-1)}{B} > d-1$, which is always true when $\nu > d-1$

$$\begin{aligned}\frac{\nu+(d+1)(B-1)}{B} &> \frac{d-1+(d+1)(B-1)}{B},\\ &= d - \frac{B-2}{B},\\ &> d-1,\end{aligned}$$

for $B > 1/2$. If specifying subposterior priors, ensure $\nu \geq d+1$

3.C Product of Gaussian Distributions

Assume the density for a parameter $\boldsymbol{\psi} \in \mathbb{R}^{d_\psi}$ can be wrote as a product of $B \in \mathbb{N}$ Gaussian distributions. Let the b-th density have mean and variance denoted μ_b and V_b respectively for $b \in \{1, \ldots, B\}$.

$$
\begin{aligned}
\pi(\boldsymbol{\psi}) &= \prod_{b=1}^{B} \mathcal{N}_d(\boldsymbol{\psi}|\mu_b, V_b),\\
&\propto \prod_{b=1}^{B} \exp\left\{-\frac{1}{2}(\boldsymbol{\psi}-\mu_b)^\top V_b^{-1}(\boldsymbol{\psi}-\mu_b)\right\},\\
&= \exp\left\{-\frac{1}{2}\sum_{b=1}^{B}\left\{\boldsymbol{\psi}^T V_b^{-1}\boldsymbol{\psi} - \mu_b^\top V_b^{-1}\boldsymbol{\psi} - \boldsymbol{\psi}^\top V_b^{-1}\mu_b + \mu_b^T V_b^{-1}\mu_b\right\}\right\}.
\end{aligned}
$$

Let $\bar{V}^{-1} = \sum_{b=1}^{B} V_b^{-1}$, then:

$$
\begin{aligned}
\pi(\boldsymbol{\psi}) &\propto \exp\left\{-\frac{1}{2}\left(\boldsymbol{\psi}^\top\left(\bar{V}^{-1}\right)\boldsymbol{\psi} - \left(\sum_{b=1}^{B}\mu_b^\top V_b^{-1}\right)\boldsymbol{\psi} - \boldsymbol{\psi}^\top\left(\sum_{b=1}^{B} V_b^{-1}\boldsymbol{\mu}_b\right)\right)\right\},\\
&\propto \exp\left\{-\frac{1}{2}(\boldsymbol{\psi}-\bar{\boldsymbol{\mu}})^\top \bar{V}^{-1}(\boldsymbol{\psi}-\bar{\boldsymbol{\mu}})\right\},
\end{aligned}
$$

where $\bar{\boldsymbol{\mu}} = \bar{V}\sum_{b=1}^{B} V_b^{-1}\boldsymbol{\mu}_b$. Therefore: $\boldsymbol{\psi} \sim \mathrm{N}(\bar{\mu}, \bar{V})$. Eaton (1983).

Chapter 4

Subposteriors with Inflation, Shifting, and Scaling - SwISS

4.1 Introduction

Markov chain Monte Carlo (MCMC) algorithms are widely used within Bayesian modelling to sample from the often intractable posterior distribution. These techniques are widely applicable and only require point-wise evaluation of the posterior density. One of the potential drawbacks of MCMC algorithms is their lack of scalability. The computational cost of MCMC is typically, at best, linear in the amount of data and can be prohibitive for large data sets, both in computational cost and storage.

In settings with large data sets, or where the model is computationally expensive, evaluating the posterior at every iteration of the MCMC algorithm may be infeasible. Strategies to overcome this include data subsampling Welling and Teh (2011); Baker et al. (2019a); Nemeth and Fearnhead (2021); Baker et al. (2019b), where only a subset of the data is used at each MCMC iteration, or delayed acceptance (Sherlock et al., 2017; Quiroz et al., 2018), where the Metropolis–Hastings accept-reject probability is replaced with a cheaper approximation to the true posterior and the full data posterior is evaluated less frequently.

In situations where it is possible to easily parallelise computation in a MapReduce framework, or through cloud-computing infrastructure such as Amazon Web Services, then statistical modelling becomes easily scalable to large data sets. However, applying this approach in practice using algorithms such as MCMC, which are designed to work in serial rather than parallel, is challenging. In this chapter, we consider the divide-and-conquer strategy to circumvent the computational bottleneck of MCMC, where the data are partitioned into batches, and each batch is stored on a separate computer core. MCMC is then applied independently on each data batch and posterior samples from each computer are combined to form an accurate approximation of the full posterior, *i.e.*, the posterior that would have been obtained using the full data set.

The main challenge with divide-and-conquer approaches for MCMC lies in the merge step. A range of approaches has been considered in the literature such as:

the use of weighted averages of the batch samples (Scott et al., 2016); kernel density estimation (Neiswanger et al., 2013); Gaussian process approximations (Nemeth and Sherlock, 2018); finding the Wasserstein barycenter of different measures Srivastava et al. (2014), the geometric median of batch samples (Minsker et al., 2014), as well as using a post-MCMC importance sample Entezari et al. (2018), to name a few.

One of the most popular algorithms in the literature is the consensus Monte Carlo algorithm (Scott et al., 2016), which approximates the full posterior using a weighted average of sub-posterior samples. The consensus approach is computationally cheap to apply, does not require tuning and scales well to high-dimensional parameter spaces. It is also analytically exact in the case of Gaussian sub-posteriors, but can produce poor approximations when the sub-posteriors are non-Gaussian (see Section 4.4.1).

In this chapter we propose SwISS, an algorithm that is as fast as the consensus algorithm, is exact in the Gaussian case, does not require tuning, and which scales well to high-dimensional posterior distributions. However, in the case of non-Gaussian sub-posteriors, it can produce more accurate posterior approximations than the consensus algorithm. Unlike the consensus approach, SwISS does not merge samples but instead applies a transformation to the posterior samples that are generated from a stochastic approximation of the full posterior. As in Entezari et al. (2018), we refer to this stochastic approximation as the *inflated sub-posterior*, which is the posterior density, conditional on a subset of the data, raised to a positive power. Inflating the sub-posterior in this manner has the effect of approximately preserving the shape of the posterior density conditional on the full data. Affine transformations (shift and re-scale) are applied to each batch of sub-posterior samples to form an approximate sample from the full posterior. This is a generalisation of the algorithm of Robert et al. (2019) which simply shifts each sub-posterior, with no further correction and hence performs poorly when the sub-posterior variances differ substantially. There are many different affine transformations that produce a sample from the true posterior when the sub-posteriors are Gaussian; we provide theoretical support for our particular choice, showing that, in a natural sense, it is optimal amongst the set of transformations that are exact in the Gaussian case.

The chapter is organised as follows: Section 4.2 provides an introduction to divide-and-conquer MCMC, covering the notation for posterior and sub-posterior densities. In Section 4.3 we introduce our proposed algorithm, SwISS, and provide supporting theoretical results and pseudo-code for implementation. Section 4.4 covers the numerical performance of SwISS and is compared against other popular divide-and-conquer algorithms from the literature. Finally, Section 4.5 gives a summary of the contributions from the chapter.

4.2 Preliminaries

Let $f(\mathbf{y}|\boldsymbol{\vartheta})$ be the likelihood for a statistical model, parameterised by $\boldsymbol{\vartheta} \in \mathbb{R}^d$, for a data set $\mathbf{y} = \{y_1, y_2, \ldots, y_n\}$ of length n. Let $\pi_0(\boldsymbol{\vartheta})$ denote the prior density for the parameter vector $\boldsymbol{\vartheta}$, then our posterior density is, up to a constant of proportionality,

$$\pi(\boldsymbol{\vartheta}|\mathbf{y}) \propto \pi_0(\boldsymbol{\vartheta}) f(\mathbf{y}|\boldsymbol{\vartheta}). \tag{4.1}$$

We assume that $\mathbf{y}$ can be partitioned into B batches, $\mathbf{y}_1, \ldots, \mathbf{y}_B$, such that the likelihood for the full data is the product of the likelihoods for the individual batches, *i.e.*, $f(\mathbf{y}|\boldsymbol{\vartheta}) = \prod_{b=1}^{B} f_b(\mathbf{y}_b|\boldsymbol{\vartheta})$. This is the case, for example, when the individual data points are independent. The posterior density for $\boldsymbol{\vartheta}$ given $\mathbf{y}$ is, up to a constant of proportionality,

$$\pi(\boldsymbol{\vartheta}|\mathbf{y}) \propto \pi_0(\boldsymbol{\vartheta}) \prod_{b=1}^{B} f_b(\mathbf{y}_b|\boldsymbol{\vartheta}). \tag{4.2}$$

In the literature, there are generally two approaches to applying MCMC on batches of data. In the first approach, MCMC is applied to target a *sub-posterior* density for each batch b, of the form

$$\pi_b(\boldsymbol{\vartheta}|\mathbf{y}_b) \propto \pi_0(\boldsymbol{\vartheta})^{\frac{1}{B}} f(\mathbf{y}_b|\boldsymbol{\vartheta}), \tag{4.3}$$

where $b = 1, \ldots, B$, such that $\prod_{b=1}^{B} \pi_b(\boldsymbol{\vartheta}|\mathbf{y}_b) = \pi(\boldsymbol{\vartheta}|\mathbf{y})$ as defined in (4.2).

If we assume that there are N sub-posterior samples from each of the B batches, which we define as $(\boldsymbol{\vartheta}_b^{(j)}; j \in \{1, \dots, N\}, b \in \{1, \dots, B\})$, then the consensus Monte Carlo algorithm (Scott et al., 2016) gives a simple strategy for approximating the full posterior (4.2) through a weighted average of the sub-posterior samples,

$$\boldsymbol{\vartheta}^{(j)} = \left(\sum_{b=1}^{B} \mathbf{w}_b \right)^{-1} \sum_{b=1}^{B} \mathbf{w}_b \boldsymbol{\vartheta}_b^{(j)},$$

where the weights are typically chosen to be $\mathbf{w}_b = \mathrm{Var}\left[\boldsymbol{\vartheta} | \mathbf{y}_b\right]^{-1}$. If each $\pi_b(\boldsymbol{\vartheta}|\mathbf{y}_b)$ is Gaussian, then the consensus algorithm produces exact samples from the full posterior.

A second approach applies MCMC to each *inflated sub-posterior*, where the target density for batch $b = 1, \dots, B$ is

$$\pi_b^B(\boldsymbol{\vartheta}|\mathbf{y}_b) \propto \pi_0(\boldsymbol{\vartheta}) f(\mathbf{y}_b|\boldsymbol{\vartheta})^B. \tag{4.4}$$

This is a stochastic (across partitions of the data) approximation to the full posterior $\pi(\boldsymbol{\vartheta}|\mathbf{y}) \approx \pi_b^B(\boldsymbol{\vartheta}|\mathbf{y}_b)$, and hence individual samples from it, in a sense, are already on the same scale as samples from the full posterior.

If we assume that the data are partitioned equally across batches, then in the limit, as the amount of data in each batch $n_* = n/B$ approaches infinity, the likelihood will typically dominate the prior, so that by the Bernstein von Mises the-

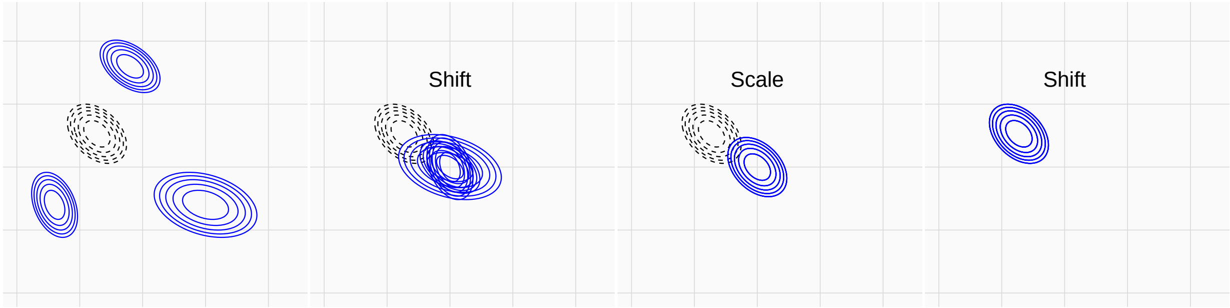


Figure 4.3.1: A visual representation of the SwISS algorithm applied to a two-dimensional Gaussian with three sub-posteriors (blue) approximating the full posterior (black). First, the batch samples are shifted by their respective means ($\boldsymbol{\vartheta}_b - \mu_b$), then scaled by the matrix $\mathbf{A}_b$ before finally being shifted by the global mean μ.

orem, the b^{th} inflated sub-posterior is $\boldsymbol{\vartheta}_b \sim \mathcal{N}(\hat{\boldsymbol{\vartheta}}_b, I_{O,b}^{-1}(\hat{\boldsymbol{\vartheta}}_b))$, where approximately, $\hat{\boldsymbol{\vartheta}}_b \sim \mathcal{N}(\boldsymbol{\vartheta}_0, BI_E^{-1}(\boldsymbol{\vartheta}_0))$ and where I_E and $I_{O,b}$ are the full-data expected information and the observed information from the inflated likelihood for batch b, respectively, and $\boldsymbol{\vartheta}_0$ is the true parameter value. Hence, the difference between the expectations of the inflated sub-posteriors are $\mathcal{O}(n_*^{-1/2})$ and, since $\lim_{n\to\infty} ||I_E^{-1} I_{O,b}|| = 1 + \mathcal{O}(n_*^{-1/2})$, the ratio of the variances of the sub-posteriors is $1 + \mathcal{O}(n_*^{-1/2})$. However, in practice, both the location and scale of the inflated sub-posteriors can vary considerably if the partitioned data sets are imbalanced (see examples in Section 4.4). Our proposed algorithm, SwISS, provides a correction for the discrepancy in the variance and location of the sub-posterior approximations.

4.3 SwISS Algorithm

Suppose that we have applied independent Monte Carlo algorithms, such as MCMC, in parallel to sample from the inflated sub-posteriors (4.4), and denote the j^{th} (of N) sample from the b^{th} (of B) batch by $\boldsymbol{\vartheta}_b^{(j)}$. The SwISS algorithm transforms each sample from the inflated sub-posterior into a sample from an approximation to the full posterior (4.2) using a batch-specific affine transformation. In the case of Gaussian sub-posteriors, as we show below, these affine transformations produce a set of samples from the correct Gaussian full posterior, and SwISS is exact in this setting. In general, sub-posteriors are non-Gaussian; however under standard regularity conditions and the Bernstein-von Mises theorem (Le Cam and Yang, 2000), as $n^* = n/B$ approaches infinity the sub-posteriors will be approximately Gaussian and SwISS can be expected to produce samples from an approximation to the full posterior.

Firstly, let us suppose that each inflated sub-posterior is Gaussian with expectation μ_b and invertible variance matrix $\mathbf{V}_b$, so that the full posterior is $\boldsymbol{\vartheta}|\mathbf{y} \sim \mathcal{N}_d(\mu, \mathbf{V})$ where,

$$\mathbf{V} = \left(\frac{1}{B} \sum_{b=1}^{B} \mathbf{V}_b^{-1} \right)^{-1} \text{ and } \mu = \mathbf{V} \frac{1}{B} \sum_{b=1}^{B} \mathbf{V}_b^{-1} \mu_b \tag{4.5}$$

are the variance and mean of the full posterior.

Since it is invertible, $\mathbf{V}_b$ is a positive-definite matrix and therefore it has a $d \times d$, invertible square root, $\mathbf{M}_b$; *i.e.* $\mathbf{V}_b = \mathbf{M}_b\mathbf{M}_b^\top$. Similarly, the full posterior variance $\mathbf{V}$ has a square root, $\mathbf{M}$, so that $\mathbf{V} = \mathbf{M}\mathbf{M}^\top$. Let samples from the b^{th} inflated sub-posterior, $\boldsymbol{\vartheta}_b^{(1:N)}$, be (marginally) realisations from the random variable $\boldsymbol{\vartheta}_b \sim \pi_b^B$ and define the transformed random variable:

$$\boldsymbol{\theta}_b := \mathbf{A}_b \left(\boldsymbol{\vartheta}_b - \mu_b\right) + \mu. \tag{4.6}$$

where $\mathbf{A}_b$ is any matrix satisfying $\mathbf{A}_b\mathbf{V}_b\mathbf{A}_b^\top = \mathbf{V}$; for example, $\mathbf{A}_b = \mathbf{M}\mathbf{M}_b^{-1}$.

Clearly, $\mathbb{E}\left[\boldsymbol{\theta}_b\right] = \mu$ and $\mathrm{Var}\left[\boldsymbol{\theta}_b\right] = \mathbf{V}$. Furthermore, an affine transformation of a Gaussian random variable is Gaussian, and hence $\boldsymbol{\theta} \sim \mathcal{N}_d(\mu, \mathbf{V})$. Applying the same transformation to individual samples from the b^{th} batch, therefore provides a sample from the full posterior. As discussed above, even when the sub-posteriors are not Gaussian, we can still apply the same scaling and shifting to any sub-posterior samples and produce samples from an approximation to the full posterior.

4.3.1 Choice of Matrix Square Roots

Matrix square roots are not unique; *e.g.* for a diagonal matrix, each element of the diagonal square root could be negated; methods for finding a square root of a positive-definite matrix include the Cholesky decomposition, or the simple symmetric square root arising from the spectral decomposition. Moreover, for square roots

$\mathbf{M}$ and $\mathbf{M}_b$, $\mathbf{A}_b$ need not be simply $\mathbf{M}\mathbf{M}_b^{-1}$, and indeed, this is not always the most sensible choice.

For now, let $\mathbf{M}$ be any $d \times d$ square root of $\mathbf{V}$ and let

$$\widetilde{\mathbf{V}}_b := \mathbf{M}^{-1}\mathbf{V}_b \left(\mathbf{M}^{-1}\right)^\top \quad \text{and} \quad \mathbf{A}_b = \mathbf{M}\widetilde{\mathbf{M}}_b^{-1}\mathbf{M}^{-1},$$

where $\widetilde{\mathbf{M}}_b$ is any $d \times d$ square root of $\widetilde{\mathbf{V}}_b$. Then, $\mathrm{Var}\left[\mathbf{M}^{-1}\boldsymbol{\vartheta}_b\right] = \widetilde{\mathbf{V}}_b$, so $\mathrm{Var}\left[\widetilde{\mathbf{M}}_b^{-1}\mathbf{M}^{-1}\boldsymbol{\vartheta}_b\right] = \mathbf{I}$ and hence $\mathrm{Var}\left[\mathbf{A}_b\boldsymbol{\vartheta}_b\right] = \mathbf{M}\mathbf{M}^\top = \mathbf{V}$.

If $\mathbf{V}_b = \mathbf{V}$ for all $b = 1, \ldots, B$, then $\widetilde{\mathbf{V}}_b = \mathbf{I}$, and provided $\widetilde{\mathbf{M}}_b$ is chosen to be $\mathbf{I}$, $\mathbf{A}_b$ becomes the identity transformation. To be clear, though, if some diagonal elements of $\widetilde{\mathbf{M}}_b$ had been chosen to be -1 rather than 1 then $\mathbf{A}_b$ would not be the identity and, unless the initial distribution of points $\boldsymbol{\vartheta}_b^{(1:N)}$ was elliptically symmetric, the transformation in (4.6) would not then lead to a set of points that represented the true posterior at all.

Applying the same logic as above, the transformation $\widetilde{\mathbf{M}}_b$ should be the square root of $\widetilde{\mathbf{V}}_b$ that moves the individual points $\boldsymbol{\vartheta}_b^{(j)}$ as little as possible. With this in mind, we define a natural measure of the distance moved by N points, $\boldsymbol{\vartheta}^{(1:N)}$, to which a linear transformation $\mathbf{A}$ is applied, as:

$$D\left(\mathbf{A}; \boldsymbol{\vartheta}^{(1:N)}\right) := \frac{1}{N}\sum_{j=1}^{N} \left\|\boldsymbol{\vartheta}^{(j)} - \mathbf{A}\boldsymbol{\vartheta}^{(j)}\right\|^2, \tag{4.7}$$

where $\|.\|^2$ denotes Euclidean distance. We wish to find the linear transformation $\widetilde{\mathbf{M}}_b$ that minimises $D(\widetilde{\mathbf{M}}_b^{-1}; \boldsymbol{\vartheta}^{(1:N)})$ subject to the constraint that $\widetilde{\mathbf{M}}_b\widetilde{\mathbf{M}}_b^\top = \widetilde{\mathbf{V}}_b$. In Section 4.3.2 we show that, provided the points have expectation zero, as $N \uparrow \infty$, the best choice of $\widetilde{\mathbf{M}}_b$ is the positive-definite, symmetric square root of $\widetilde{\mathbf{V}}_b$; this is the square root used by SwISS. The choice of square root, $\mathbf{M}$, of $\mathbf{V}$ is less important, since within the linear transformation $\mathbf{A}_b$, the initial transformation by $\mathbf{M}^{-1}$ is later inverted; however, with the general motivation of preventing excess movement, SwISS sets $\mathbf{M}$ to be the positive-definite, symmetric square root of $\mathbf{V}$. Finally, the averaged re-centring algorithm of Robert et al. (2019) can be viewed as a special case of SwISS where $\mathbf{A}_b = \mathbf{I}$.

4.3.2 The Positive-Definite, Symmetric Square Root and its Optimality

We first define the positive-definite symmetric square root of a positive-definite matrix and detail the sense in which it is optimal with respect to the distance measure D (4.7).

Let $\mathbf{V}$ be a positive-definite matrix and let its spectral decomposition be

$$\mathbf{V} = \mathbf{U}\boldsymbol{\Lambda}\boldsymbol{\Lambda}\mathbf{U}^\top. \tag{4.8}$$

where $\boldsymbol{\Lambda}$ is a diagonal matrix with entries equal to the positive square roots of the eigenvalues of $\mathbf{V}$, and $\mathbf{U}$ is a unitary matrix (*i.e.* the columns of $\mathbf{U}$ are the orthonormal right eigenvectors of $\mathbf{V}$, so $\mathbf{U}\mathbf{U}^\top = \mathbf{I} = \mathbf{U}^\top\mathbf{U}$) and so

$$\mathbf{V}^{1/2} := \mathbf{M} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^\top. \tag{4.9}$$

The natural interpretation of $\mathbf{M}$ is as a simple scaling transformation with different scalings applied along each of the eigenvectors of $\mathbf{V}$.

As explained previously, we require a matrix $\mathbf{A}_b$ such that the transformation (4.6) leads to a sample with a variance of $\mathbf{V}$; however, when inflated sub-posteriors are non-Gaussian, we need a transformation that preserves the shape and orientation of the inflated sub-posterior as much as possible. Theorem 4.1 shows that for large N, $\mathbf{M}^{-1}$ is not likely to cause more than the minimum discrepancy, given the constraints. This is a novel result for this chapter.

Theorem 4.1. *Let $\boldsymbol{\vartheta}^{(j)} \in \mathbb{R}^d$ $(j = 1, \dots, N)$ be a set of independent and identically distributed realisations of a random variable $\boldsymbol{\vartheta}$ with $\mathbb{E}[\boldsymbol{\vartheta}] = 0$ and Var$[\boldsymbol{\vartheta}] = \mathbf{V}$. Let $\mathbf{V}$ have a spectral decomposition as in* (4.8) *and let $\mathbf{M} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^\top$. Let $\mathbf{A}$ be any other $d \times d$ matrix such that Var$[\mathbf{A}\boldsymbol{\vartheta}] = \mathbf{I}_d$. Then*

$$\mathbb{P}\left(\lim_{N\to\infty}\left[D\left(\mathbf{M}^{-1};\boldsymbol{\vartheta}^{(1:N)}\right) - D\left(\mathbf{A};\boldsymbol{\vartheta}^{(1:N)}\right)\right] > 0\right) = 0,$$

where $D(\cdot;\cdot)$ is as defined in (4.7).

Theorem 4.1 relies upon the following two results.

Proposition 4.2. *Let $Z_i \in \mathbb{R}^d$ $(i = 1, \ldots, n)$ be an independent and identically distributed sequence of random variables with $\mathbb{E}[Z] = 0$ and Var$[Z] = I_d$, and let B be any $d \times d$ matrix. Then, as $n \to \infty$, $D(B; Z_{1:n}) \to d + \mathsf{trace}\left(BB^T\right) - 2\mathsf{trace}(B)$, almost surely.*

Proof.

$$D(B; Z_{1:n}) = \frac{1}{n}\sum_{i=1}^{n} ||(B-I)Z||^2 \to \mathbb{E}\left[||(B-I)Z||^2\right] = \mathbb{E}\left[\sum_{i,j,k=1}^{d} Z_i (B-I)^T_{i,j}(B-I)_{j,k}Z_k\right]$$

$$= \sum_{i,j=1}^{d} (B-I)^T_{i,j}(B-I)_{j,i} = \mathsf{trace}\left((B-I)^T(B-I)\right)$$

$$= \mathsf{trace}\left(B^T B\right) + \mathsf{trace}\left(I\right) - 2\mathsf{trace}\left(B\right),$$

where the convergence is almost sure. But $\mathsf{trace}\left(B^T B\right) = \mathsf{trace}\left(BB^T\right)$, giving the required result. □

Lemma 4.3. *Let V, Λ, and U be as defined in Theorem 4.1, and let and $\lambda_i = \Lambda_{i,i}$ $(i = 1, \ldots, d)$. Then*

$$\sup_{M: MM^T = V} \mathsf{trace}\left(M\right) = \sum_{i=1}^{d} \lambda_i.$$

The supremum is achieved when $M = U\Lambda U^T$.

Proof. The rows of U form an orthonormal basis $e_1, \ldots, e_d$, with $e_i^T V e_i = \lambda_i^2$ $(i =$

$1, \dots, d)$. For any matrix M with $MM^T = V$,

$$\lambda_i^2 = e_i^T M M^T e_i = ||M^T e_i||^2 = ||f_i||^2,$$

where $f_i = M^T e_i$. Next, recall that for any unitary matrix, U, and square matrix M, $\mathsf{trace}\left(UMU^T\right) = \mathsf{trace}\,(M)$. Thus, using the Cauchy-Schwarz inequality, and since U^T is also unitary,

$$\begin{aligned}\mathsf{trace}\,(M) = \mathsf{trace}\left(U^T M U\right) &= \sum_{i,j,k=1}^{d} (e_i)_j (e_i)_k M_{j,k} \\ &= \sum_{i=1}^{d} e_i^T M e_i = \sum_{i=1}^{d} f_i^T e_i \leq \sum_{i=1}^{d} ||f_i||\ ||e_i|| = \sum_{i=1}^{d} \lambda_i.\end{aligned}$$

The final part of the Lemma follows as $\mathsf{trace}\left(U \Lambda U^T\right) = \mathsf{trace}\,(\Lambda) = \sum_{i=1}^{d} \lambda_i$. □

To prove Theorem 4.1, let $Z_i = AX_i$, so $\mathbb{E}\left[Z_i\right] = 0$ and $\mathrm{Var}\left[Z_i\right] = I_d$, and let $B = A^{-1}$ so $BB^T = V$. Since $D(A; X_{1:n}) = D(B; Z_{1:n})$, by Proposition 4.2, and then from Lemma 4.3, we have almost surely,

$$\begin{aligned}D(A; X_{1:n}) - D(M^{-1}; X_{1:n}) &= D(B; Z_{1:n}) - D(M; Z_{1:n}) \\ &\to 2\mathsf{trace}\,(M) - 2\mathsf{trace}\,(B) \geq 0. \quad \square\end{aligned}$$

The affine transformation (4.6) of SwISS is easy to apply to each batch of inflated sub-posterior samples, making the algorithm as fast and as simple to use as the

Algorithm 4 SwISS Algorithm; here $\mathsf{SPSQ}(\mathbf{V})$ denotes the symmetric positive-definite square root of the matrix $\mathbf{V}$ as described through (4.8) and (4.9).

```
Require: $\{\boldsymbol{\vartheta}_b^j\}_{j=1}^N$ - $N$ Monte Carlo samples from each of the $B$ inflated posteriors

  Calculate the mean and variance for each of the inflated posteriors
  for $b \in \{1, \dots, B\}$ do
      $\mu_b \leftarrow \mathsf{mean}[\boldsymbol{\vartheta}_b^{(1:N)}]$   and   $\mathbf{V}_b \leftarrow \mathsf{var}[\boldsymbol{\vartheta}_b^{(1:N)}]$
  end for

  Set the global mean $\mu$ and variance $\mathbf{V}$ and calculate the matrix square root,
  $\mathbf{V} = \left(\frac{1}{B}\sum_{b=1}^{B}\mathbf{V}_b^{-1}\right)^{-1}$,   $\mu = \mathbf{V}\frac{1}{B}\sum_{b=1}^{B}\mathbf{V}_b^{-1}\mu_b$,   and   $\mathbf{M} \leftarrow \mathsf{SPSQ}(\mathbf{V})$

  Apply the affine transformation to the inflated posterior samples
  for $b \in \{1, \dots, B\}$ do
      $\widetilde{\mathbf{V}}_b \leftarrow \mathbf{M}^{-1}\mathbf{V}_b\mathbf{M}^{-1}$
      $\widetilde{\mathbf{M}}_b \leftarrow \mathsf{SPSQ}(\widetilde{\mathbf{V}}_b)$
      $\mathbf{A}_b \leftarrow \mathbf{M}\widetilde{\mathbf{M}}_b^{-1}\mathbf{M}^{-1}$
      Set $\boldsymbol{\theta}_b^{1:N} \leftarrow \mathbf{A}_b\left(\boldsymbol{\vartheta}_b - \mu_b\right) + \mu$
  end for

  Concatenate the transformed samples $\boldsymbol{\theta}_b^{1:N}$ to give a Monte Carlo approximation
  of the full posterior distribution $\pi(\boldsymbol{\vartheta}|\mathbf{y})$
  return $\left\{\boldsymbol{\theta}_1^{(1:N)}, \dots, \boldsymbol{\theta}_B^{(1:N)}\right\}$
```

consensus algorithm, with the guarantee of exactness in the Gaussian case. A visual representation of SwISS is given in Figure 4.3.1 and pseudo-code for implementing the algorithm is given in Algorithm 4.

4.4 Experiments

In this section we test the accuracy of the SwISS algorithm to merge batch posterior samples drawn from a variety of posterior distributions. We consider various complex posterior geometries to highlight the difference between affine transformations of posterior samples (*i.e.* SwISS) and averaging posterior samples (*i.e.* Consensus Monte Carlo). We also investigate the efficiency of alternative merging algorithms on popular statistical models with simulated and real data. We compare the SwISS algorithm against the following popular competing algorithms from the literature:

- **Consensus Monte Carlo** (Cons) algorithm (Scott et al., 2016), as described in Section 4.2.
- **Semiparametric density estimation** (SKDE)[1] from Neiswanger et al. (2013), where sub-posteriors are approximated semi-parametrically as described in Hjort and Glad (1995).
- **Average re-centring** (AR) algorithm from Robert et al. (2019), which is a special case of SwISS where $\mathbf{A}_b = \mathbf{I}$.
- **Gaussian Barycenter** (GB) algorithm (Srivastava et al., 2018), assuming a Gaussian approximation for each inflated sub-posterior, the barycenter is the geometric center of the inflated sub-posterior distributions.

We assess the accuracy of the above algorithms to combine batch posterior samples to form an approximation of the full posterior, comparing the merged approximations against the full posterior, which is generated by sampling (in serial) from the posterior conditional on the full data set. Accuracy of estimation of the poste-

[1] Implemented using the *parallelMCMCcombine* R package

rior of the d-dimensional parameter, $\boldsymbol{\vartheta}$, is assessed with the following discrepancy measures:

- **Mahalanobis distance** (Mah):

$$D_{\text{Mah}} := \sqrt{(\mu_a - \mu_f)^\top \, \mathbf{V}_f^{-1} \, (\mu_a - \mu_f)},$$

  where $\mathbf{V}_f$ and μ_f are the variance and mean estimates of posterior samples taken from the full data posterior using an MCMC algorithm. For a given posterior approximation algorithm, *e.g.* SwISS, μ_a denotes the estimated mean.

- **Mean absolute skew deviation** (Skew):

$$\eta := \frac{1}{d} \sum_{i=1}^{d} |\hat{\gamma}_i^a - \hat{\gamma}_i^f|,$$

  where $\gamma_i = \mathbb{E}[\{(\boldsymbol{\vartheta}_i - \mu_i)/\mathbf{V}_{ii}^{1/2}\}^3]$; *i.e.* η is the sum over components of the third standardised moments.

- **Integrated absolute distance** (IAD):

$$D_{\text{IAD}} := \frac{1}{2d} \sum_{j=1}^{d} \int |\hat{\pi}_j^a(\theta_j) - \hat{\pi}_j^f(\theta_j)| \mathrm{d}\theta_j \in [0,1],$$

  the average of the integrated absolute differences between two kernel density estimates of the marginal posteriors for each component, j, of ϑ: $\hat{\pi}_j^f$, obtained

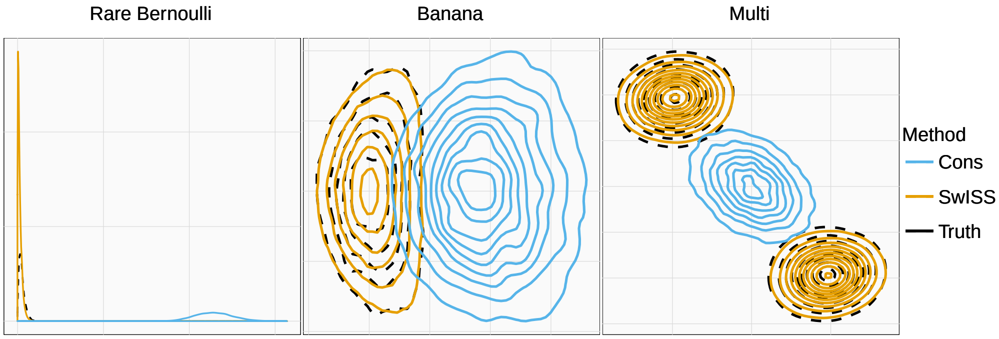


Figure 4.4.1: Density plots showing the posterior reconstructions using the SwISS algorithm against the Consensus algorithm on a rare Bernoulli target (left), warped Gaussian (also known as the banana-shaped target) (middle) and bi-modal target (right). In all cases the x-axis is ϑ_1; for the left plot the y-axis is density, and for the other two plots it is ϑ_2.

from samples from the true posterior, and $\hat{\pi}_j^a$ using one of the approximate merging algorithms (Chan et al., 2021).

4.4.1 Complex Posterior Geometries

One of the main motivations for using MCMC to sample from a posterior distribution, rather than using deterministic approximations (*e.g.* Laplace), is that the posteriors are often non-Gaussian. We consider three artificially generated posterior distributions of dimension one or two (see Figure 4.4.1) which reflect a range of potential posterior shapes and we compare SwISS against the consensus Monte Carlo algorithm in these settings. Here $\phi(\mu)$ denotes the probability density function of a standard Gaussian $\mathcal{N}(\mu, 1)$, for some $\mu \in \mathbb{R}$

- **Rare Bernoulli density**

  $$\pi_b(\vartheta_1|\mathbf{y}_b) \propto \vartheta_1(1-\vartheta_1)^{999}, \quad \text{for each } b \in \{1,\ldots,B\},$$

  This corresponds to a posterior with 1000 Bernoulli observations with a single positive response and a uniform prior on the success probability, ϑ, which gives a skewed posterior density.

- **Warped bivariate Gaussian density**

  $$\pi_b(\boldsymbol{\vartheta}|\mathbf{y}_b) \propto \phi(\vartheta_1)\phi(\vartheta_2+\vartheta_1^2), \quad \text{for each } b \in \{1,\ldots,B\},$$

  where $\boldsymbol{\vartheta} = (\vartheta_1, \vartheta_2)$.

- **Mixture of bivariate Gaussian densities**

  $$\pi_b(\boldsymbol{\vartheta}|\mathbf{y}_b) \propto \phi(\boldsymbol{\vartheta}-\mu_1)+\phi(\boldsymbol{\vartheta}-\mu_2), \quad \text{for each } b \in \{1,\ldots,B\},$$

  where $\boldsymbol{\vartheta} = (\mu_1, \mu_2)$.

Both the SwISS and the consensus algorithm are guaranteed to be exact in the case of merging Gaussian posterior samples, but it can be shown that both algorithms still work well for a variety of non-Gaussian posteriors. However, one of the drawbacks of the consensus algorithm is that averaging across batches of sub-posterior samples can remove posterior features such as skewness and multi-modality, as illustrated in Figure 4.4.1.

Figure 4.4.1 shows posterior density plots for each of the three models, where full MCMC has been utilised to provide a *ground truth* approximation for the full data posterior. The consensus Monte Carlo and SwISS approximations are based on combing samples from $B = 10$ sub-posterior and inflated sub-posterior approximations, respectively. The results from these three test cases show that the consensus algorithm struggles to approximate the full data posterior when the target density exhibits non-Gaussian behaviours. The SwISS algorithm, which utilises affine transformations of the inflated sub-posterior samples, rather than averaging, can produce reliable approximations when the posterior is significantly non-Gaussian.

4.4.2 Scalability with parameter dimension

Typically, divide-and-conquer methods are advertised for use with tall data, *i.e.* a large number of observations and up to a moderate number of parameters. Here, we test the accuracy and computational speed of the merging algorithms as the number of parameters grows.

Let $\boldsymbol{\vartheta}|\mathbf{y}_b \sim \mathcal{N}_d(\mu_b, \boldsymbol{V}_b)$ for $b \in \{1, \dots, B = 10\}$, where d is the dimension of the parameter space and let $\mu_b \sim \mathcal{N}_d(0, \mathrm{I}_d)$ and $\boldsymbol{V}_b \sim \mathcal{W}^{-1}(5d, \mathrm{I}_d)$. Each sub-posterior is Gaussian, with expectation and variance drawn respectively from Gaussian and inverse-Wishart distributions. For each experiment $N = 5,000$ samples were drawn from each sub-posterior and inflated sub-posterior. Using this model, the full data posterior is tractable:

$$\boldsymbol{\vartheta}|\mathbf{y} \sim \mathcal{N}_d\left(V \sum_{b=1}^{B} V_b^{-1}\mu_b, V\right),$$

where $V^{-1} = \sum_{b=1}^{B} V_b^{-1}$. The following set of dimensions were used: $d \in \{5, 10, 20, 40, 80\}$.

Figure 4.4.2 shows that both the consensus Monte Carlo algorithm and SwISS perform well with increasing dimension (as measured by integrated absolute distance) and are both computationally efficient. The semi-parametric KDE approach, Gaussian barycenter and average re-centering approaches display reduced accuracy (as measured by integrated absolute distance). Only SwISS and consensus are robust to increasing the dimension of the parameter space. In terms of the computational cost required to merge the posterior samples, all approaches are generally fast, with the exception of the semi-parametric KDE approach.

4.4.3 Linear Mixed Effects Model

A natural way to extend the simple linear model is to introduce both fixed and random effects. This extension can be particularly useful when data exhibit a hierarchical dependency structure, for example, to cluster student test scores based on classroom. Let $y_{i,j} \in \mathcal{Y} \subseteq \mathbb{R}$ (for $i, j = 1, \dots, n_j$, and $j = 1, \dots, n$) be the response variable, where n_j is the number of observations for group j. The fixed and random

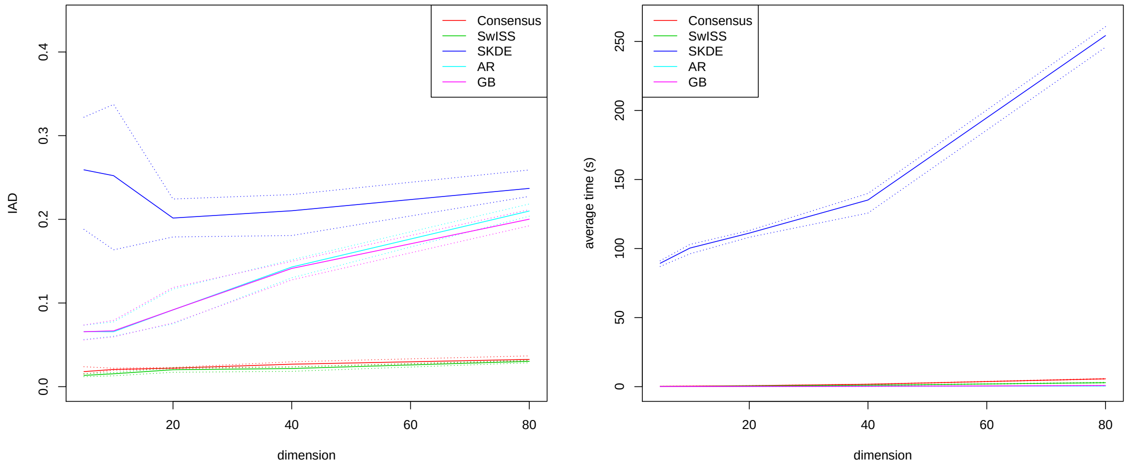


Figure 4.4.2: The left plot shows integrated absolute distance for each method for a different number of parameters. The right plot shows the mean time each combination method took to obtain the samples from an approximate posterior.

effects are $\mathbf{x}_{i,j} \in \mathcal{X} \subseteq \mathbb{R}^p$ and $\mathbf{z}_j \in \mathcal{Z} \subseteq \mathbb{R}^r$, respectively, and are related to the response variable by

$$y_{i,j}|\boldsymbol{\beta}, \boldsymbol{\alpha}_j, 1 \sim \text{Logistic } (\mathbf{x}_{i,j}\boldsymbol{\beta} + \mathbf{z}_j\boldsymbol{\alpha}_j, 1), \quad \boldsymbol{\alpha}_i \sim \mathcal{N}_r(0, \Sigma),$$

where $\boldsymbol{\beta} \in \mathbb{R}^p$ and $\boldsymbol{\alpha}_i \in \mathbb{R}^r$ are the fixed and random effect model coefficients and the $\mathsf{Logistic}(a, 1)$ distribution has a cumulative distribution function of $1/(1+\exp[-(x-a)])$. Our parameters of interest are then $\boldsymbol{\vartheta} = (\boldsymbol{\beta}, \boldsymbol{\alpha}, \Sigma)$, where Σ represents the variance of the random effects. We assume an inverse-Wishart distribution for the prior of Σ, $\Sigma \sim \mathcal{W}^{-1}(\nu, S)$, with $\nu = 5$ and $S = 5I_r$, and we assumed $\boldsymbol{\beta} \sim \mathcal{N}_p(0, 1000I_p)$.

We simulated a dataset that contains $200,000$ observations. We set the number of groups $n = 2000$ and the number observations for each group $n_j = 100$, for

$j = 1, \ldots, n$. The number of parameters for the fixed effects were set to $p = 10$, with $\beta_{0,i} = (-1)^{(i-1)}$ set for the simulation. The number of parameters for each random effect was set to be $r = 2$, and we set

$$\Sigma_0 = \begin{bmatrix} 1 & 0.2 \\ 0.2 & 1 \end{bmatrix},$$

then α_i were simulated independently from a $\mathcal{N}_r(0, \Sigma_0)$ distribution. We included an intercept term, that is $x_{i,j,1} = 1$ for all i, j, otherwise $\mathbf{x}_{i,j}$ and z_j were simulated from independent $\mathsf{Bernoulli}(0.5)$ distributions.

The data were randomly partitioned into $B = 10$ batches by group, so that each group only belonged to one batch. This was necessary since divide and conquer methods assume independence between the batches. With a Gaussian observation model for the $y_{i,j}$, marginalisation over all of the random effects would be tractable. The logistic observation model necessitates the use of a sampling scheme such as MCMC.

We used the STAN software to sample from the full posterior and sub-posteriors generating $N = 5,000$ MCMC samples after an initial 1,000 sample burn in. Table 4.4.1 gives the discrepancy measures for each of the merging algorithms, averaged over 10 random partitions of the data. The results show that all algorithms perform well, with the exception of AR and the Gaussian barycenter, both on the Mahalanobis metric. The SwISS and Consensus algorithms are robust across the range

Table 4.4.1: Discrepancy measures for the linear mixed effects model on the simulated dataset. These measures were averaged over 5 random partitions of the data. Estimated standard errors are given in brackets.

Algorithm	Mah	Skew	IAD
SwISS	0.69 (0.14)	**0.02** (<0.01)	0.06 (0.01)
Consensus	**0.39** (0.13)	0.03 (0.01)	**0.04** (0.01)
Average Re-centring	2.20 (0.31)	**0.02** (<0.01)	0.13 (0.01)
Semi-parametric KDE	**0.39** (0.08)	0.04 (<0.01)	**0.04** (0.01)
Gaussian Barycenter	2.19 (0.31)	0.05 (0.01)	0.13 (0.01)

of metrics.

4.4.4 Logistic Regression Model

Logistic regression is a popular technique for modelling binary data, *i.e.* $y_i \in \{0, 1\}$. Features $\mathbf{x}_i \in \mathbb{R}^d$, also known as covariates, that can indicate the classification outcome are mapped onto the binary observations using a logit transformation, where the outcome probability $\mathbb{P}(y_i = 1) = \exp(\mathbf{x}_i^\top \boldsymbol{\vartheta}) / \left(1 + \exp(\mathbf{x}_i^\top \boldsymbol{\vartheta})\right)$, is the success probability of a Bernoulli random variable. Our parameter of interest $\boldsymbol{\vartheta} \in \mathbb{R}^d$ is the vector of coefficients.

We consider two data sets, the first is a synthetic data set which is designed to simulate a scenario with rare but highly informative features. This data set is similar to the one given in Scott et al. (2016). We simulate $N = 100,000$ data points with $d = 5$ binary features with relative frequencies of $\mathbf{x}_i = 1$ being $(1, 0.02, 0.03, 0.05, 0.001)$ and the corresponding true parameter values are $\boldsymbol{\vartheta}_0 = (-3, 1.2, -0.5, 0.8, 3)$. Due to the rarely occurring final feature, this can lead to largely differing variances across the sub-posteriors. For our experiments, we spilt the data equally across $B = 25$ batches.

We also consider a real-world data set; the Hepmass data set[2] from high-energy particle physics where the response is an indicator for whether a signal was indicative of an exotic particle being present as opposed to background noise. The data set contains 27 real features which we augmented with an intercept term to give $d = 28$ parameters. The full data set contains 10.5 million responses, in this experiment we considered the first $N = 100,000$ and split the data across $B = 20$ batches. This subset was considered as we needed the full-data MCMC samples to compare against.

In each of the our experiments, the data were repeatedly partitioned $n_{\text{runs}} = 5$ times with a Monte Carlo average of the discrepancy metrics given in Table 4.4.2. The STAN (Stan Development Team, 2016) software, which implements an automatically-tuned version of Hamiltonian Monte Carlo sampling, was used as the MCMC sampler and applied to the full posterior and sub-posteriors for each experiment. Each sampler drew $N = 10,000$ samples after a burn in of $1,000$ iterations.

The results in Table 4.4.2 show that SwISS and the Consensus algorithm outperform all of the others on the simulated data, whereas for the real example all of the methods work well. The AR algorithm performs especially poorly on the synthetic data example as the variance of the sub-posteriors varies across subposteriors,

[2] http://archive.ics.uci.edu/ml/datasets/HEPMASS

Table 4.4.2: Discrepancy measures for the logistic regression model with simulated and Hepmass data sets. Each metric was averaged over 5 runs. Estimated standard error are given in brackets.

Dataset	Algorithm	Mah	Skew	IAD
Simulated data	SwISS	**0.46** (0.30)	**0.04** (0.01)	**0.05** (0.03)
	Consensus	0.48 (0.35)	0.05 (0.01)	0.06 (0.03)
	Average Re-centring	5.46 (3.92)	0.13 (0.07)	0.20 (0.03)
	Semi-parametric KDE	1.25 (1.11)	0.76 (0.33)	0.12 (0.06)
	Gaussian Barycenter	5.42 (3.75)	**0.04** (0.01)	0.20 (0.01)
Hepmass	SwISS	0.56 (0.05)	**0.03** (<0.01)	0.03 (<0.01)
	Consensus	**0.35** (0.05)	**0.03** (<0.01)	0.03 (<0.01)
	Average Re-centring	0.47 (0.04)	**0.03**(<0.01)	**0.02** (<0.01)
	Semi-parametric KDE	0.36 (0.05)	**0.03**(<0.01)	0.03 (<0.01)
	Gaussian Barycenter	0.46 (0.05)	**0.03**(<0.01)	**0.02** (<0.01)

and the AR algorithm does not correct for this when the sub-posterior samples are merged. The similarity of eprformance on the Hepmass data could be due to the sub-posteriors all being close to Gaussian. We would expect both Consensus and SwISS to work well in this setting as they are exact for Gaussian sub-posteriors, and much faster to apply than nonparametric methods such as SKDE.

4.5 Conclusions

We have introduced a new method to merge posterior samples generated in parallel on independent batches of data. Our algorithm, SwISS, is fast, scalable to high-dimensional settings, and accurate on a variety of test cases. The SwISS algorithm, like the consensus Monte Carlo algorithm, is simple to apply and competitive against popular alternative divide-and-conquer algorithms. SwISS also has the advantage that it does not require hyper-parameter tuning and is faster to apply than many of the alternative divide-and-conquer algorithms given in the literature. We have provided theoretical support for our choice of affine transformations and shown that SwISS is exact in the case of merging inflated Gaussian sub-posteriors. Code to recreate this work is available through the Github link: https://github.com/CJohnVyner/SwISS

Chapter 5

Further Work

In this Chapter we introduce three potential ideas for further work.

5.1 SwISS Bias Correction

Small sample bias might not usually be considered in big-data settings, however, it can cause problems when parallelisation methods are used. With standard non-parallelised Monte Carlo the bias in the estimator tends to zero as the number of observations tends to infinity. This is not necessarily the case in the divide-and-conquer framework. If we assume the data in each batch tends to infinity then the bias will tend to zero. However, this assumption seems at odds with the point of the divide-and-conquer methods; if we could analyse endless amounts of data on a single computer core we would not have a need for these methods. A more natural assumption, would be to assume the maximum amount of data able to be processed on a single batch is fixed. Then, to account for the number of observations, N, we increase the number of batches, B, that is as $N \to \infty$, $B \to \infty$, and $N_b \leq K$, where N_b is the number of observations per batch $b \in \{1, \ldots, B\}$ for some known fixed constant $K \in \mathbb{N}$. Under this assumption the bias for the estimators for the first and second moments of each subposterior distribution will always be present in some models no matter how many observations there are. Here we will outline a potential jackknife correction for SwISS to account for this bias (Shao and Tu, 2012).

SwISS averages the estimates of the first two moments of each inflated subposterior distribution to obtain an estimate of the first two moments of the full-data posterior distribution. Let $\hat{\boldsymbol{\mu}}_b$ and $\hat{\boldsymbol{V}}_b$ be the inflated-subposterior distribution estimates for the mean and variance respectively for $b \in \{1, \ldots, B\}$. Recall the estimators for the full-data posterior distribution:

$$\hat{\boldsymbol{V}}^{-1} = \frac{1}{B} \sum_{b=1}^{B} \hat{\boldsymbol{V}}_b^{-1}, \qquad \hat{\boldsymbol{\mu}} = \frac{1}{B} \hat{\boldsymbol{V}} \sum_{b=1}^{B} \hat{\boldsymbol{V}}_b^{-1} \hat{\boldsymbol{\mu}}_b.$$

Assume $\boldsymbol{V}_j = \boldsymbol{V}_i$ for all $i, j \in \{1, \ldots, B\}$ so that $\hat{\boldsymbol{\mu}} = 1/B \sum_{b=1}^{B} \hat{\boldsymbol{\mu}}_b$. This assumption would be restrictive, it is only made to make presenting the bias correction easier.

We assume

$$\begin{aligned}\hat{\mu_b} &\approx \mu_{T,b} + \frac{\beta_1(\vartheta)}{N_b} + \frac{\beta_2(\vartheta)}{N_b^2} + \dots, \\ &\approx \mu_{T,b} + \frac{\beta_1(\vartheta)}{N_b},\end{aligned}$$

where $\beta_k(\vartheta)$ is the k-th term in the Taylor expansion of the bias, for $k \in \mathbb{N}$ and $\mu_{T,b}$ is the true mean under the data-generating model for batch b. If we run a Monte Carlo algorithm on each batch with a subset of the data you could estimate $\beta_1(\vartheta)$. Figure 5.1.1 shows a visual representation of the bias assuming the inverse relationship with N_b. Bias in the plot is referring to $\beta_1(\vartheta)/N_b$. The point estimates could be obtained from each batch then once the red curve is estimated, you could correct the bias to what it would have been had the all the observations been used.

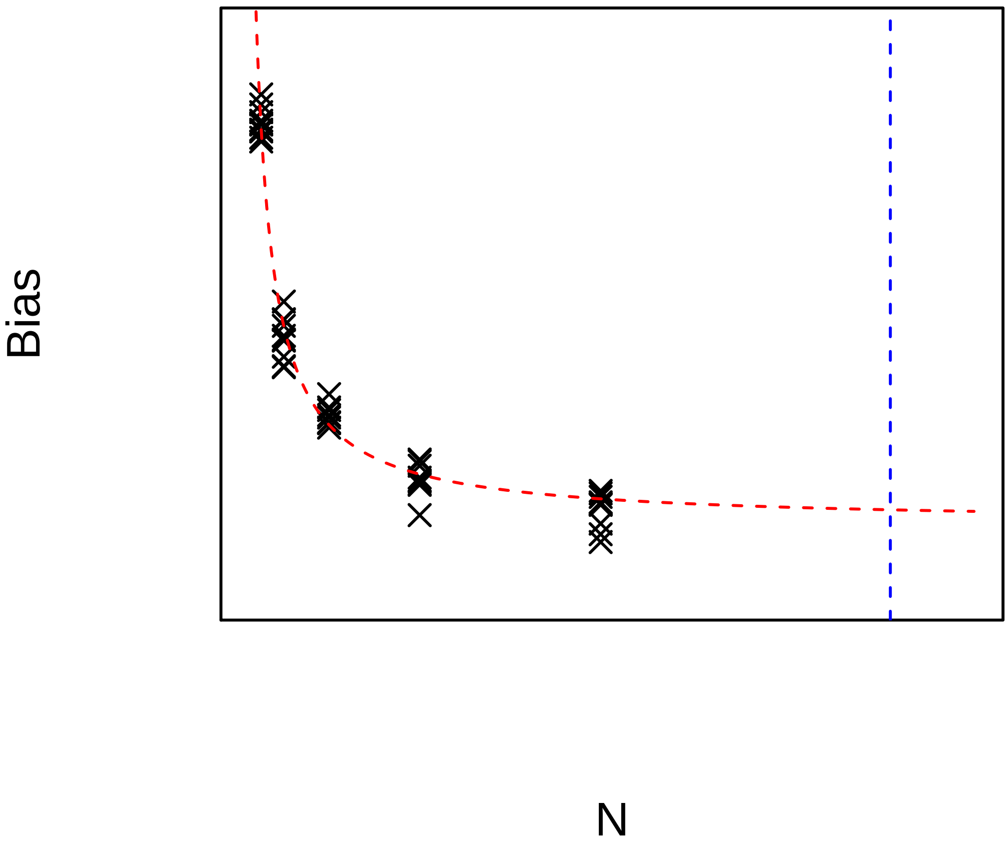


Figure 5.1.1: An example of bias decreasing as number of observations increase. Assuming bias is inversely proportional to N, the number of observations, it is possible to estimate the curve in red. Given the red curve you could estimate the bias at any point, such as the blue line.

5.2 Post Sampling Correction

Many of the divide-and-conquer methods typically partition the data, generate samples from the subposterior (or inflated-subposterior) distributions, collect the samples and then use those samples to obtain samples from an approximation to the full-data posterior distribution. A considerable benefit of this approach is that there is very little communication between the cores. If the cores have to communicate often there can be latency issues, which can vastly increase the computational cost. However, there could be some benefit to a single pass back after the samples have been combined.

One idea could be to use importance sampling to re-weight your combined samples. To use this approach you would need to know your proposal density, so it would not be appropriate for most of the methods, but could be applied to methods in which the posterior and proposal densities could be calculated.

5.3 Data Dependency Issues

Standard divide-and-conquer methods assume conditional independence between each partition of data. However, there are a wide variety of models that do not assume independence in the data, for example for example time series and spatial models. Recall that typically we write our posterior density as

$$\pi(\boldsymbol{\vartheta}|\mathbf{y}) = \prod_{b=1}^{B} \pi(\boldsymbol{\vartheta}|\mathbf{y}_b),$$

where the posterior is a product of the subposterior densities. This allows us to partition data and then run the MCMC in an embarrassingly parallel fashion. If we were to include models where the likelihood model had the following form:

$$f(y_j|\boldsymbol{\vartheta}, y_{j-1}, \ldots, y_{j-k}),$$

where $k \in \mathbb{N}$ and $j \in \{k+1, \ldots, n\}$, then when partitioning the data you would also have to include data from the other batches to account for the dependency structure in the data. Generalising the divide-and-conquer framework to include these types of models could be an interesting area of future research.

Bibliography

Asmussen, S. and Glynn, P. (2007). *Stochastic Simulation: Algorithms and Analysis*, volume 57. Springer.

Au, S. and Beck, J. (2003). Important sampling in high dimensions. *Structural Safety*, 25(2):139 – 163.

Baker, J., Fearnhead, P., Fox, E. B., and Nemeth, C. (2019a). Control variates for stochastic gradient MCMC. *Statistics and Computing*, 29(3):599–615.

Baker, J., Fearnhead, P., Fox, E. B., and Nemeth, C. (2019b). sgmcmc: An R package for stochastic gradient Markov chain Monte Carlo. *Journal of Statistical Software*, 91(1):1–27.

Baldi, P., Cranmer, K., Faucett, T., Sadowski, P., and Whiteson, D. (2016). Parameterized neural networks for high-energy physics. *The European Physical Journal C*, 76.

Bardenet, R., Doucet, A., and Holmes, C. C. (2017). On Markov chain Monte Carlo methods for tall data. *Journal of Machine Learning Research*, 18(47).

Barndorff-Nielsen, O. E. O. E. (1989). *Asymptotic techniques for use in statistics*. Monographs on statistics and applied probability. Chapman and Hall, London ; New York.

Bellman, R. (1966). Dynamic programming. *Science*, 153(3731):34–37.

Bernstein, D. S. (2009). Matrix Mathematics. In *Matrix Mathematics*. Princeton university press.

Bloomfield, P. (2004). *Fourier Analysis of Time Series: An Introduction*. Wiley Series in Probability and Statistics. Wiley.

Carter, C. K. and Kohn, R. (1994). On Gibbs sampling for state space models. *Biometrika*, 81(3):541–553.

Chan, R. S., Pollock, M., Johansen, A. M., and Roberts, G. O. (2021). Divide-and-Conquer Monte Carlo Fusion. *arXiv preprint arXiv:2110.07265*.

Cuturi, M. and Doucet, A. (2014). Fast omputation of Wasserstein barycenters. In *International conference on machine learning*, pages 685–693. PMLR.

Eaton, M. L. (1983). Multivariate statistics: a vector space approach. *JOHN WILEY & SONS*.

Entezari, R., Craiu, R. V., and Rosenthal, J. S. (2018). Likelihood Inflating Sampling Algorithm. *Canadian Journal of Statistics*, 46(1):147–175.

Hastings, W. K. (1970). Monte Carlo Sampling Methods Using Markov Chains and Their Applications. *Biometrika*, 57(1):97–109.

Higham, N. J. (2002). *Accuracy and stability of numerical algorithms*. SIAM.

Hjort, N. L. and Glad, I. K. (1995). Nonparametric Density Estimation with a Parametric Start. *The Annals of Statistics*, 23(3):882–904.

Keller, J. M., Gray, M. R., and Givens, J. A. (1985). A fuzzy k-nearest neighbor algorithm. *IEEE transactions on systems, man, and cybernetics*, pages 580–585.

Knuth, D. E. (1998). *The Art of Computer Programming, Volume 3: (2nd Ed.) Sorting and Searching*. Addison Wesley Longman Publishing Co., Inc., USA.

Le Cam, L. and Yang, G. L. (2000). Asymptotics in statistics: some basic concepts. *Berlin, German: Springer*.

Li, C., Srivastava, S., and Dunson, D. B. (2017). Simple, scalable and accurate posterior interval estimation. *Biometrika*, 104(3):665–680.

McKechan, D., Robinson, C., and Sathyaprakash, B. (2010). A tapering window for time-domain templates and simulated signals in the detection of gravitational waves from coalescing compact binaries. *Classical and Quantum Gravity*, 27.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 21(6):1087–1092.

Minsker, S., Srivastava, S., Lin, L., and Dunson, D. B. (2014). Scalable and Robust Bayesian Inference via the Median Posterior. *Proceedings of the 31st International Conference on Machine Learning (ICML)*, pages 1656–1664.

Neal, R. M. et al. (2011). MCMC using Hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2(11):2.

Neiswanger, W., Wang, C., and Xing, E. (2013). Asymptotically Exact, Embarrassingly Parallel MCMC. *Uncertainty in Artificial Intelligence - Proceedings of the 30th Conference, UAI 2014*.

Nemeth, C. and Fearnhead, P. (2021). Stochastic Gradient Markov Chain Monte Carlo. *Journal of the American Statistical Association*, 116(533):433–450.

Nemeth, C. and Sherlock, C. (2018). Merging MCMC Subposteriors through Gaussian-Process Approximations. *Bayesian Analysis*, 13(2):507 – 530.

Quiroz, M., Tran, M.-N., Villani, M., and Kohn, R. (2018). Speeding up MCMC by delayed acceptance and data subsampling. *Journal of Computational and Graphical Statistics*, 27(1):12–22.

Robert, C., Wu, C., and Robert, C. P. (2019). Average of Recentered Parallel MCMC for Big Data. working paper or preprint.

Robert, C. P. and Casella, G. (2005). *Monte Carlo Statistical Methods (Springer Texts in Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Roberts, G. O. and Tweedie, R. L. (1996). Exponential convergence of langevin distributions and their discrete approximations. *Bernoulli*, pages 341–363.

Rue, H., Martino, S., and Chopin, N. (2009). Approximate bayesian inference for latent gaussian models by using integrated nested laplace approximations. *Journal of the royal statistical society: Series b (statistical methodology)*, 71(2):319–392.

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*, 11:78–88.

Shao, J. and Tu, D. (2012). *The jackknife and bootstrap*. Springer Science & Business Media.

Sheather, S. J. and Jones, M. C. (1991). A reliable data-based bandwidth selection method for kernel density estimation. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 683–690.

Sherlock, C., Golightly, A., and Henderson, D. A. (2017). Adaptive, delayed-acceptance MCMC for targets with expensive likelihoods. *Journal of Computational and Graphical Statistics*, 26(2):434–444.

Silverman, B. W. (1986). *Density estimation for statistics and data analysis*, volume 26. CRC press.

Spiegelhalter, D., Thomas, A., Best, N., and Lunn, D. (2003). Winbugs user manual v1.4. *University of Cambridge - MRC Biostatistics Unit, https://www.mrc-bsu.cam.ac.uk/wp-content/uploads/manual14.pdf.*

Srivastava, S., Cevher, V., Tran-Dinh, Q., and Dunson, D. B. (2014). Wasp: Scalable bayes via barycenters of subset posteriors. *Duke Discussion Paper-2014-05.*

Srivastava, S., Li, C., and Dunson, D. B. (2018). Scalable Bayes via barycenter in Wasserstein space. *The Journal of Machine Learning Research*, 19(1):312–346.

Stan Development Team (2016). Stan modeling language users guide and reference manual, version 2.14.0. *http://mc-stan.org.*

Tu, L. (2010). *An Introduction to Manifolds.* Universitext. Springer New York.

UCI Machine Learning repository (2020). https://archive.ics.uci.edu/ml/datasets/hepmass. *ML datasets.*

Welling, M. and Teh, Y. W. (2011). Bayesian Learning via Stochastic Gradient

Langevin Dynamics. In *Proceedings of the 28th International Conference on Machine Learning*.