



## Global Consensus Monte Carlo

Lewis J. Rendell , Adam M. Johansen , Anthony Lee & Nick Whiteley


To cite this article: Lewis J. Rendell , Adam M. Johansen , Anthony Lee & Nick Whiteley (2020): Global Consensus Monte Carlo, Journal of Computational and Graphical Statistics, DOI: [10.1080/10618600.2020.1811105](https://doi.org/10.1080/10618600.2020.1811105)

To link to this article: <https://doi.org/10.1080/10618600.2020.1811105>



© 2020 The Author(s). Published with license by Taylor & Francis Group, LLC.




[View supplementary material](#) 



Published online: 16 Oct 2020.



[Submit your article to this journal](#) 



Article views: 286



[View related articles](#) 



[View Crossmark data](#) 

# Global Consensus Monte Carlo

Lewis J. Rendell<sup>a</sup> , Adam M. Johansen<sup>a</sup> , Anthony Lee<sup>b</sup> , and Nick Whiteley<sup>b</sup> 

<sup>a</sup>Department of Statistics, University of Warwick, Coventry, UK; <sup>b</sup>School of Mathematics, University of Bristol, Bristol, UK

## ABSTRACT

To conduct Bayesian inference with large datasets, it is often convenient or necessary to distribute the data across multiple machines. We consider a likelihood function expressed as a product of terms, each associated with a subset of the data. Inspired by global variable consensus optimization, we introduce an instrumental hierarchical model associating auxiliary statistical parameters with each term, which are conditionally independent given the top-level parameters. One of these top-level parameters controls the unconditional strength of association between the auxiliary parameters. This model leads to a distributed MCMC algorithm on an extended state space yielding approximations of posterior expectations. A trade-off between computational tractability and fidelity to the original model can be controlled by changing the association strength in the instrumental model. We further propose the use of an SMC sampler with a sequence of association strengths, allowing both the automatic determination of appropriate strengths and for a bias correction technique to be applied. In contrast to similar distributed Monte Carlo algorithms, this approach requires few distributional assumptions. The performance of the algorithms is illustrated with a number of simulated examples. Supplementary materials for this article are available online.

## ARTICLE HISTORY

Received August 2019  
Revised April 2020

## KEYWORDS

Bayesian inference;  
Distributed inference;  
Markov chain Monte Carlo;  
Sequential Monte Carlo

## 1. Introduction

Large datasets arising in modern statistical applications present serious challenges for standard computational techniques for Bayesian inference, such as Markov chain Monte Carlo (MCMC) and other approaches requiring repeated evaluations of the likelihood function. We consider here the situation where the data are distributed across multiple computing nodes. This could be because the likelihood function cannot be computed on a single computing node in a reasonable amount of time, for example, the data might not fit into main memory.

We assume that the likelihood function can be expressed as a product of terms so that the posterior density for the statistical parameter  $Z$  satisfies



$$\pi(z) \propto \mu(z) \prod_{j=1}^b f_j(z), \quad (1)$$

where  $Z$  takes values  $z \in E \subseteq \mathbb{R}^d$ , and  $\mu$  is a prior density. We assume that  $f_j$  is computable on computing node  $j$  and involves consideration of  $\mathbf{y}_j$ , the  $j$ th subset or “block” of the full dataset, which comprises  $b$  such blocks.

Many authors have considered “embarrassingly parallel” MCMC algorithms approximating expectations with respect to (1), following the consensus Monte Carlo (CMC) approach of Scott et al. (2016). Such procedures require separate MCMC chains to be run on each computing node, each targeting a distribution dependent only on the associated likelihood contribution  $f_j$ . The samples from each of these chains are then combined

in a final post-processing step to generate an approximation of the true posterior  $\pi$ . Such algorithms require communication between the nodes only at the very beginning and end of the procedure, falling into the MapReduce framework (Dean and Ghemawat 2008); their use is therefore more advantageous when inter-node communication is costly, for example, due to high latency. However, the effectiveness of such approaches at approximating the true posterior  $\pi$  depends heavily on the final combination step. Many proposed approaches are based on assumptions on the likelihood contributions  $f_j$ , or employ techniques that may be infeasible in high-dimensional settings. We later review some of these approaches, and some issues surrounding their use, in Section 2.4.

Instead of aiming to minimize entirely communication between nodes, we propose an approach that avoids employing a final aggregation step, thereby avoiding distributional assumptions on  $\pi$ . This approach is motivated by global variable consensus optimization (see, e.g., Boyd et al. 2011, sec. 7; we give a summary in Section 2.1). Specifically we introduce an instrumental hierarchical model, associating an auxiliary parameter with each likelihood contribution (and therefore each computing node), which are conditionally independent given  $Z$ . An additional top-level parameter controlling their unconditional strength of association is also introduced. This allows the construction of an MCMC algorithm on an extended state space, yielding estimates of expectations with respect to  $\pi$ . By tuning the association strength through the top-level parameter, a trade-off between computational tractability and fidelity to the original model can be controlled.

**CONTACT** Lewis J. Rendell  [lrendell@gmail.com](mailto:lrendell@gmail.com)  Department of Statistics, University of Warwick, Coventry CV4 7AL, UK.

 Supplementary materials for this article are available online. Please go to [www.tandfonline.com/r/JCGS](http://www.tandfonline.com/r/JCGS).

© 2020 The Author(s). Published with license by Taylor and Francis Group, LLC

This is an Open Access article distributed under the terms of the Creative Commons Attribution License (<http://creativecommons.org/licenses/by/4.0>), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. The moral rights of the named author(s) have been asserted.

As well as avoiding issues associated with embarrassingly parallel algorithms, our framework presents benefits compared to the simple approach of constructing an MCMC sampler to directly target (1). In settings where communication latency is nonnegligible but the practitioner’s time budget is limited, our approach allows a greater proportion of this time to be spent on computation rather than communication, allowing for faster exploration of the state space.

Our approach was initially presented in Rendell et al. (2018). A proposal to use essentially the same framework in a serial context has been independently and contemporaneously published by Vono, Dobigeon, and Chainais (2019), who construct a Gibbs sampler via a “variable splitting” approach. Rather than distributing the computation, the authors focus on the setting where  $b = 1$  to obtain a relaxation of the original simulation problem. An implementation of this approach for problems in binary logistic regression has been proposed in Vono, Dobigeon, and Chainais (2018), with a number of nonasymptotic and convergence results presented more recently in Vono, Paulin, and Doucet (2019). Our work focuses on distributed settings, providing a sequential Monte Carlo (SMC) implementation of the framework that may be used to generate bias-corrected estimates.

We introduce the proposed framework and the resulting algorithmic structure in Section 2, including some discussion of issues in its implementation, and comparisons with related approaches in the literature. We then introduce a SMC implementation of the framework in Section 3. Various simulation examples are presented in Section 4, before conclusions are provided in Section 5.

## 2. The Instrumental Model and MCMC

For simplicity, we shall occasionally abuse notation by using the same symbol for a probability measure on  $E$ , and for its density with respect to some dominating measure. For the numerical examples presented herein,  $E \subseteq \mathbb{R}^d$  and all densities are defined with respect to a suitable version of the Lebesgue measure. We use the notation  $x_{m:n} := (x_m, \dots, x_n)$  for arbitrary  $x_m, \dots, x_n$ . For a probability density function  $\nu$  and function  $\varphi$ , we denote by  $\nu(\varphi)$  the expectation of  $\varphi(Z)$  when  $Z \sim \nu$ , that is,

$$\nu(\varphi) := \int \varphi(z)\nu(z) dz.$$

### 2.1. The Instrumental Model

The goal of the present article is to approximate (1). We take an approach that has also been developed in contemporaneous work by Vono, Dobigeon, and Chainais (2019), although their objectives were somewhat different. Alongside the variable of interest  $Z$ , we introduce a collection of  $b$  instrumental variables each also defined on  $E$ , denoted by  $X_{1:b}$ . On the extended state space  $E \times E^b$ , we define the probability density function  $\tilde{\pi}_\lambda$  by

$$\tilde{\pi}_\lambda(z, x_{1:b}) \propto \mu(z) \prod_{j=1}^b K_j^{(\lambda)}(z, x_j) f_j(x_j), \quad (2)$$

where for each  $j \in \{1, \dots, b\}$ ,  $\{K_j^{(\lambda)} : \lambda \in \mathbb{R}_+\}$  is a family of Markov transition densities on  $E$ . Defining

$$f_j^{(\lambda)}(z) := \int_E K_j^{(\lambda)}(z, x) f_j(x) dx, \quad (3)$$

the density of the  $Z$ -marginal of  $\tilde{\pi}_\lambda$  may be written as

$$\pi_\lambda(z) := \int_{E^b} \tilde{\pi}_\lambda(z, x_{1:b}) dx_{1:b} \propto \mu(z) \prod_{j=1}^b f_j^{(\lambda)}(z). \quad (4)$$

Here, we may view each  $f_j^{(\lambda)}$  as a smoothed form of  $f_j$ , with  $\pi_\lambda$  being the corresponding smoothed form of the target density (1).

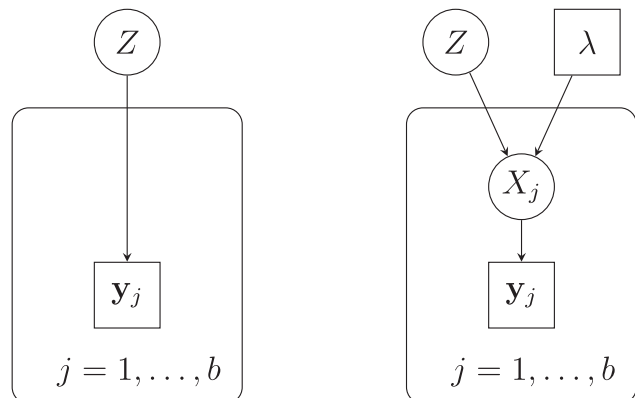
The role of  $\lambda$  is to control the fidelity of  $f_j^{(\lambda)}$  to  $f_j$ , and so we assume the following in the sequel.

**Assumption 1.** For all  $\lambda > 0$ ,  $f_j^{(\lambda)}$  is bounded for each  $j \in \{1, \dots, b\}$ ; and  $f_j^{(\lambda)} \rightarrow f_j$  pointwise as  $\lambda \rightarrow 0$  for each  $j \in \{1, \dots, b\}$ .

For example, Assumption 1 implies that  $\pi_\lambda$  converges in total variation to  $\pi$  by Scheffé’s lemma (Scheffé 1947), and therefore  $\pi_\lambda(\varphi) \rightarrow \pi(\varphi)$  for bounded  $\varphi : E \rightarrow \mathbb{R}$ . Assumption 1 is satisfied for essentially any kernel that may be used for kernel density estimation; here,  $\lambda$  takes a similar role to that of the smoothing bandwidth.

On a first reading one may assume that the  $K_j^{(\lambda)}$  are chosen to be independent of  $j$ ; for example, with  $E = \mathbb{R}$  one could take  $K_j^{(\lambda)}(z, x) = \mathcal{N}(x; z, \lambda)$ . We describe some considerations in choosing these transition kernels in Section S1.2 of the supplementary materials, and describe settings in which choosing these to differ with  $j$  may be beneficial.

The instrumental hierarchical model is presented diagrammatically in Figure 1. The variables  $X_{1:b}$  may be seen as “proxies” for  $Z$  associated with each of the data subsets, which are conditionally independent given  $Z$  and  $\lambda$ . Loosely speaking,  $\lambda$  represents the extent to which we allow the local variables  $X_{1:b}$  to differ from the global variable  $Z$ . In terms of computation, it is the separation of  $Z$  from the subsets of the data  $y_{1:b}$ , given  $X_{1:b}$  introduced by the instrumental model, that can be exploited by distributed algorithms.



**Figure 1.** Directed acyclic graphs, representing the original statistical model (left) and the instrumental model we construct (right).

This approach to constructing an artificial joint target density is easily extended to accommodate random effects models, in which the original statistical model itself contains local variables associated with each data subset. These variables may be retained in the resulting instrumental model, alongside the local proxies  $X_{1:b}$  for  $Z$ . A full description of the resulting model is presented in Rendell (2020).

The framework we describe is motivated by concepts in distributed optimization, a connection that is also explored in the contemporaneous work of Vono, Dobigeon, and Chainais (2019). The global consensus optimization problem (see, e.g., Boyd et al. 2011, sec. 7) is that of minimizing a sum of functions on a common domain, under the constraint that their arguments are all equal to some global common value. If for each  $j \in \{1, \dots, b\}$  one uses the Gaussian kernel density  $K_j^{(\lambda)}(z, x) = \mathcal{N}(x; z, \lambda)$ , then taking the negative logarithm of (2) gives

$$-\log \tilde{\pi}_\lambda(z, x_{1:b}) = C - \log \mu(z) - \sum_{j=1}^b \log f_j(x_j) + \frac{1}{2\lambda} \sum_{j=1}^b (z - x_j)^2, \quad (5)$$

where  $C$  is a normalizing constant. Maximizing  $\pi(z)$  is equivalent to minimizing this function under the constraint that  $z = x_j$  for  $j \in \{1, \dots, b\}$ , which may be achieved using the alternating direction method of multipliers (Bertsekas and Tsitsiklis 1989). Specifically, (5) corresponds to the use of  $1/\lambda$  as the penalty parameter in this procedure.

There are some similarities between this framework and approximate Bayesian computation (ABC; see Marin et al. 2012, for a review of such methods). In both cases, one introduces a kernel that can be viewed as acting to smooth the likelihood. In the case of (2), the role of  $\lambda$  is to control the scale of smoothing that occurs in the parameter space; the tolerance parameter used in ABC, in contrast, controls the extent of a comparable form of smoothing in the observation (or summary statistic) space.

## 2.2. Distributed Metropolis-Within-Gibbs

The instrumental model described forms the basis of our proposed global consensus framework; “global consensus Monte Carlo” (GCMC) is correspondingly the application of Monte Carlo methods to form an approximation of  $\pi_\lambda$ . We focus here on the construction of a Metropolis-within-Gibbs Markov kernel that leaves  $\tilde{\pi}_\lambda$  invariant. If  $\lambda$  is chosen to be sufficiently small, then the  $Z$ -marginal  $\pi_\lambda$  provides an approximation of  $\pi$ . Therefore given a chain with values denoted  $(Z^i, X_{1:b}^i)$  for  $i \in \{1, \dots, N\}$ , an empirical approximation of  $\pi$  is given by

$$\pi_\lambda^N := \frac{1}{N} \sum_{i=1}^N \delta_{Z^i}, \quad (6)$$

where  $\delta_z$  denotes the Dirac measure at  $z$ .

The Metropolis-within-Gibbs kernel we consider uses the full conditional densities

$$\tilde{\pi}_\lambda(x_j | z) \propto K_j^{(\lambda)}(z, x_j) f_j(x_j) \quad (7)$$

for  $j \in \{1, \dots, b\}$ , and

$$\tilde{\pi}_\lambda(z | x_{1:b}) \propto \mu(z) \prod_{j=1}^b K_j^{(\lambda)}(z, x_j), \quad (8)$$

where (7) follows from the mutual conditional independence of  $X_{1:b}$  given  $Z$ . Here, we observe that  $K_j^{(\lambda)}(z, x_j)$  simultaneously provides a pseudo-prior for  $X_j$  and a pseudo-likelihood for  $Z$ .

We define  $M_1^{(\lambda)}$  to be a  $\tilde{\pi}_\lambda$ -invariant Markov kernel that fixes  $z$ ; we may write

$$M_1^{(\lambda)}((z, x_{1:b}); d(z', x'_{1:b})) = \delta_z(dz') \prod_{j=1}^b P_{j,z}^{(\lambda)}(x_j, dx'_j), \quad (9)$$

where for each  $j$ ,  $P_{j,z}^{(\lambda)}(x_j, \cdot)$  is a Markov kernel leaving (7) invariant. We similarly define  $M_2^{(\lambda)}$  to be a  $\tilde{\pi}_\lambda$ -invariant Markov kernel that fixes  $x_{1:b}$ ,

$$M_2^{(\lambda)}((z, x_{1:b}); d(z', x'_{1:b})) = \left[ \prod_{j=1}^b \delta_{x_j}(dx'_j) \right] P_{x_{1:b}}^{(\lambda)}(z, dz'), \quad (10)$$

where  $P_{x_{1:b}}^{(\lambda)}(z, \cdot)$  is a Markov kernel leaving (8) invariant.

Using these Markov kernels, we construct an MCMC kernel that leaves  $\tilde{\pi}_\lambda$  invariant; we present the resulting sampling procedure as Algorithm 1. In the special case in which one may sample exactly from the conditional distributions (7) and (8), this algorithm takes the form of a Gibbs sampler.

---

### Algorithm 1 Global consensus Monte Carlo: MCMC algorithm

---

Fix  $\lambda > 0$ . Set initial state  $(Z^0, X_{1:b}^0)$ ; choose chain length  $N$ .

For  $i = 1, \dots, N$ :

- For  $j \in \{1, \dots, b\}$ , sample  $X_j^i \sim P_{j,Z^{i-1}}^{(\lambda)}(X_j^{i-1}, \cdot)$ .
- Sample  $Z^i \sim P_{X_{1:b}^i}^{(\lambda)}(Z^{i-1}, \cdot)$ .

Return  $(Z^i, X_{1:b}^i)_{i=1}^N$ .

---

The interest from a distributed perspective is that the full conditional density (7) of each  $X_j$ , for given values  $x_j$  and  $z$ , depends only on the  $j$ th block of data (through the partial likelihood  $f_j$ ) and may be computed on the  $j$ th machine. Within Algorithm 1, the sampling of each  $X_j^i$  from  $P_{j,Z^{i-1}}^{(\lambda)}(X_j^{i-1}, \cdot)$  may therefore occur on the  $j$ th machine; these  $X_{1:b}^i$  may then be communicated to a central machine that draws  $Z^i$ .

Our approach has particular benefits when sampling exactly from (7) is not possible, in which case Algorithm 1 takes a Metropolis-within-Gibbs form. One may choose the Markov kernels  $P_{j,z}^{(\lambda)}$  to comprise multiple iterations of an MCMC kernel leaving (7) invariant; indeed multiple computations of each  $f_j$  (and therefore multiple accept/reject steps) may be conducted on each of the  $b$  nodes, without requiring communication between machines. This stands in contrast to more straightforward MCMC approaches directly targeting  $\pi$ , in which such communication is required for each evaluation of (1), and therefore for every accept/reject step. Similar to such a “direct” MCMC approach, each iteration of Algorithm 1 requires communication to and from each of the  $b$  machines on which the data are stored; but in cases where the communication latency is high, the resulting sampler will spend a greater proportion of time exploring the state space. This may in turn result in

faster mixing (e.g., with respect to wall-clock time). We further discuss this comparison, and the role of communication latency, in Section S1.1 of the supplementary materials.

The setting in which [Algorithm 1](#) describes a Gibbs sampler, with all variables drawn exactly from their full conditional distributions, is particularly amenable to analysis. We provide such a study in Section S2 of the supplementary materials. This analysis may also be informative about the more general Metropolis-within-Gibbs setting, when  $P_{j,z}^{(\lambda)}$  comprises enough MCMC iterations to exhibit good mixing.

### 2.3. Choosing the Regularization Parameter

For  $\varphi : \mathbf{E} \rightarrow \mathbb{R}$ , we may estimate  $\pi(\varphi)$  using (6) as  $\pi_\lambda^N(\varphi)$ . The regularization parameter  $\lambda$  here takes the role of a tuning parameter; we can view its effect on the mean squared error of such estimates using the bias–variance decomposition,

$$\mathbb{E}\left[\left(\pi_\lambda^N(\varphi) - \pi(\varphi)\right)^2\right] = [\pi_\lambda(\varphi) - \pi(\varphi)]^2 + \text{var}[\pi_\lambda^N(\varphi)], \quad (11)$$

which holds exactly when  $\mathbb{E}[\pi_\lambda^N(\varphi)] = \pi_\lambda(\varphi)$ . In many practical cases, this decomposition will provide a very accurate approximation for large  $N$ , as the squared bias of  $\pi_\lambda^N(\varphi)$  is typically asymptotically negligible in comparison to its variance.

The decomposition (11) separates the contributions to the error from the bias introduced by the instrumental model and the variance associated with the MCMC approximation. If  $\lambda$  is too large, the squared bias term in (11) can dominate while if  $\lambda$  is too small, the Markov chain may exhibit poor mixing due to strong conditional dependencies between  $X_{1:b}$  and  $Z$ , and so the variance term in (11) can dominate.

It follows that  $\lambda$  should ideally be chosen to balance these two considerations. We provide a theoretical analysis of the role of  $\lambda$  in Section S2 of the supplementary materials, by considering a simple Gaussian setting. In particular, we show that for a fixed number of data, one should scale  $\lambda$  with the number of samples  $N$  as  $\mathcal{O}(N^{-1/3})$  to minimize the mean squared error. We also consider the consistency of the approximate posterior  $\pi_\lambda$  as the number of data  $n$  tends to infinity. Specifically, suppose these are split equally into  $b$  blocks; considering  $\lambda$  and  $b$  as functions of  $n$ , we find that  $\pi_\lambda$  exhibits posterior consistency if  $\lambda/b$  decreases to 0 as  $n \rightarrow \infty$ , and that credible intervals have asymptotically correct coverage if  $\lambda/b$  decreases at a rate strictly faster than  $n^{-1}$ .

As an alternative to selecting a single value of  $\lambda$ , we propose in [Section 3](#) a SMC sampler employing Markov kernels formed via [Algorithm 1](#). In this manner, a decreasing sequence of  $\lambda$  values may be considered, which may result in lower-variance estimates for small  $\lambda$  values; we also describe a possible bias correction technique.

### 2.4. Related Approaches

As previously mentioned, a Gibbs sampler construction essentially corresponding to [Algorithm 1](#) has independently been proposed by Vono, Dobigeon, and Chainais (2019). Their main objective is to improve algorithmic performance when computation of the full posterior density is intensive by constructing full conditional distributions that are more computationally

tractable, for which they primarily consider a setting in which  $b = 1$ . In contrast, we focus on the exploitation of this framework in distributed settings (i.e., with  $b > 1$ ), in the manner described in [Section 2.2](#).

The objectives of our algorithm are similar, but not identical, to those of the previously introduced “embarrassingly parallel” approaches proposed by many authors. These reduce the costs of communication latency to a near-minimum by simulating a separate MCMC chain on each machine; typically, the chain on the  $j$ th machine is invariant with respect to a “subposterior” distribution with density proportional to  $\mu(z)^{1/b}f_j(z)$ . Communication is necessary only for the final aggregation step, in which an approximation of the full posterior is obtained using the samples from all  $b$  chains.

A well-studied approach within this framework is CMC (Scott et al. 2016), in which the post-processing step amounts to forming a “consensus chain” by weighted averaging of the separate chains. In the case that each subposterior density is Gaussian this approach can be used to produce samples asymptotically distributed according to  $\pi$ , by weighting each chain using the precision matrices of the subposterior distributions. Motivated by Bayesian asymptotics, the authors suggest using this approach more generally. In cases where the subposterior distributions exhibit near-Gaussianity this performs well, with the final “consensus chain” providing a good approximation of posterior expectations. However, there are no theoretical guarantees associated with this approach in settings in which the subposterior densities are poorly approximated by Gaussians. In such cases, CMC sometimes performs poorly in forming an approximation of the posterior  $\pi$  (as in examples of Wang et al. 2015; Srivastava et al. 2015; Dai, Pollock, and Roberts 2019), and so the resulting estimates of integrals  $\pi(\varphi)$  exhibit high bias.

Various authors (e.g., Minsker et al. 2014; Rabinovich, Angelino, and Jordan 2015; Srivastava et al. 2015; Wang et al. 2015) have therefore proposed alternative techniques for utilizing the values from each of these chains to approximate posterior expectations, each of which presents benefits and drawbacks. For example, Neiswanger, Wang, and Xing (2014) proposed a strategy based on kernel density estimation; based on this approach, Scott (2017) suggested a strategy based on finite mixture models, though notes that both methods may be impractical in high-dimensional settings.

An aggregation procedure proposed by Wang and Dunson (2013) bears some relation to our proposed framework, being based on the application of Weierstrass transforms to each subposterior density. The resulting smoothed densities are analogous to (3), which represents a smoothed form of the partial likelihood  $f_j$ . As well as proposing an aggregation method based on rejection sampling, the authors propose a technique for “refining” an initial posterior approximation, which may be expressed in terms of a Gibbs kernel on an extended state space. Comparing with our framework, this is analogous to applying [Algorithm 1](#) for one iteration with  $N$  different initial values.

A potential issue common to these approaches is the treatment of the prior density  $\mu$ . Each subposterior density is typically assigned an equal share of the prior information in the form of a fractionated prior density  $\mu(z)^{1/b}$ , but it is not clear when this approach is satisfactory. For example, suppose  $\mu$  belongs to an exponential family; any property that is not invariant to

multiplying the canonical parameters by a constant will not be preserved in the fractionated prior. As such if  $\mu(z)^{1/b}$  is proportional to a valid probability density function of  $z$ , then the corresponding distribution may be qualitatively very different to the full prior. Although Scott et al. (2016) noted that fractionated priors perform poorly on some examples (for which tailored solutions are provided), no other general way of assigning prior information to each block naturally presents itself. In contrast our approach avoids this problem entirely, with  $\mu$  providing prior information for  $Z$  at the “global” level.

Finally, we believe this work is complementary to other distributed algorithms lying outside of the “embarrassingly parallel” framework (including Xu et al. 2014; Jordan, Lee, and Yang 2019), and to approaches that aim to reduce the amount of computation associated with each likelihood calculation on a single node, for example, by using only a subsample or batch of the data (as in Korattikara, Chen, and Welling 2014; Bardenet, Doucet, and Holmes 2014; Huggins, Campbell, and Broderick 2016; Maclaurin and Adams 2014).

### 3. SMC Approach

As discussed in Section 2.3, as  $\lambda$  approaches 0 estimators  $\pi_\lambda^N(\varphi)$  formed using (6) exhibit lower bias but higher variance, due to poorer mixing of the resulting Markov chain. To obtain lower-variance estimators for  $\lambda$  values close to 0, we consider the use of SMC methodology to generate suitable estimates for a sequence of  $\lambda$  values.

SMC methodology employs sequential importance sampling and resampling; recent surveys include Doucet and Johansen (2011) and Doucet and Lee (2018). We consider here approximations of a sequence of distributions with densities  $\tilde{\pi}_{\lambda_0}, \tilde{\pi}_{\lambda_1}, \dots$ , where  $\lambda_0, \dots, \lambda_n$  is a decreasing sequence. The procedure we propose, specified in Algorithm 2, is an SMC sampler within the framework of Del Moral, Doucet, and Jasra (2006). This algorithmic form was first proposed by Gilks and Berzuini (2001) and Chopin (2002) in different settings, building upon ideas in Crooks (1998) and Neal (2001).

The algorithm presented involves simulating particles using  $\tilde{\pi}_\lambda$ -invariant Markov kernels, and has a genealogical structure imposed by the ancestor indices  $A_p^i$  for  $p \in \{0, \dots, n-1\}$  and  $i \in \{1, \dots, N\}$ . The specific scheme for simulating the ancestor indices here is known as multinomial resampling; other schemes can be used (see Douc, Cappé, and Moulines 2005; Gerber, Chopin, and Whiteley 2019, for a summary of some schemes and their properties). We use this simple scheme here as it validates the use of the variance estimators used in Section 3.1. This optional resampling step is used to prevent the degeneracy of the particle set; a common approach is to carry out this step whenever the particles’ effective sample size (Liu and Chen 1995) falls below a predetermined threshold.

Under weak conditions  $\tilde{\pi}_{\lambda_p}^N(\varphi)$  converges almost surely to  $\tilde{\pi}_{\lambda_p}(\varphi)$  as  $N \rightarrow \infty$ . One can also define the particle approximations of  $\pi_{\lambda_p}$  via

$$\pi_{\lambda_p}^N := \frac{\sum_{i=1}^N W_p^i \delta_{Z_p^i}}{\sum_{i=1}^N W_p^i}, \quad (12)$$

where  $Z_p^i$  is the first component of the particle  $\zeta_p^i$ .

---

#### Algorithm 2 Global consensus Monte Carlo: SMC algorithm

---

Fix a decreasing sequence  $(\lambda_0, \lambda_1, \dots, \lambda_n)$ . Set number of particles  $N$ .

Initialize:

- For  $i \in \{1, \dots, N\}$ , sample  $\zeta_0^i = (Z_0^i, X_{0,1:b}^i) \sim \tilde{\pi}_{\lambda_0}$  and set  $W_0^i \leftarrow 1$ .

For  $p = 1, \dots, n$ :

1. For  $i \in \{1, \dots, N\}$ , set  $\tilde{W}_p^i \leftarrow W_{p-1}^i w_p(\zeta_{p-1}^i)$ , where

$$w_p(z, x_{1:b}) := \frac{\tilde{\pi}_{\lambda_p}(z, x_{1:b})}{\tilde{\pi}_{\lambda_{p-1}}(z, x_{1:b})} = \prod_{j=1}^b \frac{K_j^{(\lambda_p)}(z, x_j)}{K_j^{(\lambda_{p-1})}(z, x_j)}.$$

2. Optionally, carry out a resampling step: for  $i \in \{1, \dots, N\}$ ,

- Independently sample  $A_{p-1}^i$  from the categorical distribution with probabilities proportional to  $(\tilde{W}_p^1, \dots, \tilde{W}_p^N)$ .
- Set  $W_p^i \leftarrow 1$ .

Otherwise: for  $i \in \{1, \dots, N\}$  set  $A_{p-1}^i \leftarrow i$ ,  $W_p^i \leftarrow \tilde{W}_p^i$ .

3. For  $i \in \{1, \dots, N\}$ , sample  $\zeta_p^i = (Z_p^i, X_{p,1:b}^i) \sim M_p(\zeta_{p-1}^{A_{p-1}^i}, \cdot)$ , where  $M_p$  is a  $\tilde{\pi}_{\lambda_p}$ -invariant MCMC kernel constructed in the manner of Algorithm 1.
4. Optionally, store the particle approximation of  $\tilde{\pi}_{\lambda_p}$ ,

$$\tilde{\pi}_{\lambda_p}^N := \frac{\sum_{i=1}^N W_p^i \delta_{\zeta_p^i}}{\sum_{i=1}^N W_p^i}.$$


---

Although the algorithm is specified for simplicity in terms of a fixed sequence  $\lambda_0, \dots, \lambda_n$ , a primary motivation for the SMC approach is that the sequence used can be determined adaptively while running the algorithm. A number of such procedures have been proposed in the literature in the context of tempering, allowing each value  $\lambda_p$  to be selected based on the particle approximation of  $\tilde{\pi}_{\lambda_{p-1}}$ . For example, Jasra et al. (2011) proposed a procedure that controls the decay of the particles’ effective sample size. Within the examples in Section 4 we employ a procedure proposed by Zhou, Johansen, and Aston (2016), which generalizes this approach to settings in which resampling is not conducted in every iteration, aiming to control directly the dissimilarity between successive distributions. A possible approach to determining when to terminate the procedure, based on minimizing the mean squared error (11), is detailed in Section S3.2 of the supplementary materials.

With regard to initialization, if it is not possible to sample from  $\tilde{\pi}_{\lambda_0}$  one could instead use samples obtained by importance sampling, or one could initialize an SMC sampler with some tractable distribution and use tempering or similar techniques to reach  $\tilde{\pi}_{\lambda_0}$ . At the expense of the introduction of an additional approximation, an alternative would be to run a  $\tilde{\pi}_{\lambda_0}$ -invariant Markov chain, and obtain an initial collection of particles by thinning the output (an approach that may be validated using results of Finke, Doucet, and Johansen 2020). Specifically, one

could use [Algorithm 1](#) to generate such samples for some large  $\lambda_0$ , benefiting from its good mixing and low autocorrelation when  $\lambda$  is sufficiently large. The effect of [Algorithm 2](#) may then be seen as refining or improving the resulting estimators, by bringing the parameter  $\lambda$  closer to zero.

Other points in favor of this approach are that many of the particle approximations (12) can be used to form a final estimate of  $\pi(\varphi)$  as explored in [Section 3.1](#), and that SMC methods can be more robust to multimodality of  $\pi$  than simple Markov chain schemes. We finally note that in such an SMC sampler, a careful implementation of the MCMC kernels used may allow the inter-node communication to be interleaved with likelihood computations associated with the particles, thereby reducing the costs associated with communication latency.

### 3.1. Bias Correction Using Local Linear Regression

We present an approach to use many of the particle approximations produced by [Algorithm 2](#). A natural idea is to regress the values of  $\pi_{\lambda}^N(\varphi)$  on  $\lambda$ , extrapolating to  $\lambda = 0$  to obtain an estimate of  $\pi(\varphi)$ . A similar idea has been used for bias correction in the context of ABC, albeit not in an SMC setting, regressing on the discrepancy between the observed data and simulated pseudo-observations (Beaumont, Zhang, and Balding 2002; Blum and François 2010).

Under very mild assumptions on the transition densities  $K_j^{(\lambda)}$ ,  $\pi_{\lambda}(\varphi)$  is smooth as a function of  $\lambda$ . Considering a first-order Taylor expansion of this function, a simple approach is to model the dependence of  $\pi_{\lambda}(\varphi)$  on  $\lambda$  as linear, for  $\lambda$  sufficiently close to 0. Having determined a subset of the values of  $\lambda$  used for which a linear approximation is appropriate (we propose a heuristic approach in [Section S3.1](#) of the supplementary materials) one can use linear least squares to carry out the regression. To account for the SMC estimates  $\pi_{\lambda_p}^N(\varphi)$  having different variances, we propose the use of weighted least squares, with the ‘‘observations’’  $\pi_{\lambda_p}^N(\varphi)$  assigned weights inversely proportional to their estimated variances; we describe methods for computing such variance estimates in [Section 3.2](#). A bias-corrected estimate of  $\pi(\varphi)$  is then obtained by extrapolating the resulting fit to  $\lambda = 0$ , which corresponds to taking the estimated intercept term.

To make this explicit, first consider the case in which  $\varphi : \mathbb{E} \rightarrow \mathbb{R}$ , so that the estimates  $\pi_{\lambda}^N(\varphi)$  are univariate. For each value  $\lambda_p$  denote the corresponding SMC estimate by  $\eta_p := \pi_{\lambda_p}^N(\varphi)$ , and let  $v_p$  denote some proxy for the variance of this estimate. Then for some set of indices  $S := \{p^*, \dots, n\}$  chosen such that the relationship between  $\eta_p$  and  $\lambda_p$  is approximately linear for  $p \in S$ , a bias-corrected estimate for  $\pi(\varphi)$  may be computed as

$$\pi_S^{\text{BC}}(\varphi) := \tilde{\eta}_S - \tilde{\lambda}_S \frac{\sum_{p \in S} (\lambda_p - \tilde{\lambda}_S)(\eta_p - \tilde{\eta}_S)/v_p}{\sum_{p \in S} (\lambda_p - \tilde{\lambda}_S)^2/v_p}, \quad (13)$$

where  $\tilde{\lambda}_S$  and  $\tilde{\eta}_S$  denote weighted means given by

$$\tilde{\lambda}_S := \frac{\sum_{p \in S} \lambda_p/v_p}{\sum_{p \in S} 1/v_p}, \quad \tilde{\eta}_S := \frac{\sum_{p \in S} \eta_p/v_p}{\sum_{p \in S} 1/v_p}. \quad (14)$$

The formal justification of this estimate assumes that the observations are uncorrelated, which does not hold here. We

demonstrate in [Section 4](#) and in [Section S4.1](#) of the supplementary materials examples on which this simple approach is nevertheless effective, but in principle one could use generalized least squares combined with some approximation of the full covariance matrix of the SMC estimates.

In the more general case where  $\varphi : \mathbb{E} \rightarrow \mathbb{R}^d$  for  $d > 1$ , we propose simply evaluating (13) for each component of this quantity separately, which corresponds to fitting an independent weighted least squares regression to each component. This facilitates the use of the variance estimators described in the following section, though in principle one could use multivariate weighted least squares or other approaches.

### 3.2. Variance Estimation for Weighted Least Squares

We propose the weighted form of least squares here since, as the values of  $\lambda$  used in the SMC procedure approach zero, the estimators generated may increase in variance: partly due to poorer mixing of the MCMC kernels as previously described, but also due to the gradual degeneracy of the particle set. To estimate the variances of estimates generated using SMC, several recent approaches have been proposed that allow this estimation to be carried out online by considering the genealogy of the particles. Using any such procedure, one may estimate the variance of  $\pi_{\lambda}^N(\varphi)$  for each  $\lambda$  value considered by [Algorithm 2](#), with these values used for each  $v_p$  in (13).

Within our examples, we use the estimator proposed by Lee and Whiteley (2018), which for fixed  $N$  coincides with an earlier proposal of Chan and Lai (2013) (up to a multiplicative constant). Specifically, after each step of the SMC sampler we compute an estimate of the *asymptotic* variance of each estimate  $\pi_{\lambda_p}^N(\varphi)$ ; that is, the limit of  $N \text{var}[\pi_{\lambda_p}^N(\varphi)]$  as  $N \rightarrow \infty$ . While this is not equivalent to computing the true variance of each estimate, for fixed large  $N$  the relative sizes of these asymptotic variance estimates should provide a useful indicator of the relative variances of each estimate  $\pi_{\lambda}^N(\varphi)$ . In [Section S4.1](#) of the supplementary materials, we show empirically that inversely weighting the SMC estimates according to these estimated variances can result in more stable bias-corrected estimates as the particle set degenerates. We also explain in [Section S3.2](#) how these estimated variances can be used within a rule to determine when to terminate the algorithm.

The asymptotic variance estimator described is consistent in  $N$ . However, if in practice resampling at the  $p$ th time step causes the particle set to degenerate to having a single common ancestor, then the estimator evaluates to zero, and so it is impossible to use this value as the inverse weight  $v_p$  in (13). Such an outcome may be interpreted as a warning that too few particles have been used for the resulting SMC estimates to be reliable, and that a greater number should be used when rerunning the procedure.

## 4. Examples

### 4.1. Log-Normal Toy Example

To compare the posterior approximations formed by our global consensus framework with those formed by some embarrassingly parallel approaches discussed in [Section 2.4](#), we conduct a simulation study based on a simple model. Let

$\mathcal{LN}(x; \mu, \sigma^2)$  denote the density of a log-normal distribution with parameters  $(\mu, \sigma^2)$ ; that is,

$$\mathcal{LN}(x; \mu, \sigma^2) = \frac{1}{x\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(\log(x) - \mu)^2}{2\sigma^2}\right).$$

One may consider a model with prior density  $\mu(z) = \mathcal{LN}(z; \mu_0, \sigma_0^2)$  and likelihood contributions  $f_j(z) = \mathcal{LN}(\log(\mu_j); \log(z), \sigma_j^2)$  for  $j \in \{1, \dots, b\}$ . This may be seen as a reparameterization of the Gaussian model in Section S2 of the supplementary materials, in which each likelihood contribution is that of a data subset with a Gaussian likelihood. This convenient setting allows for the target distribution  $\pi$  to be expressed analytically. For the implementation of the global consensus algorithm, we choose Markov transition kernels given by  $K_j^{(\lambda)}(z, x) = \mathcal{LN}(x; \log(z), \lambda)$  for each  $j \in \{1, \dots, b\}$ , which satisfy Assumption 1; this allows for exact sampling from all the full conditional distributions.

As a toy example to illustrate the effects of non-Gaussian partial likelihoods we consider a case in which  $f_j(z) = \mathcal{LN}(\log(\mu_j); \log(z), 1)$  for each  $j$ , and  $\mu(z) = \mathcal{LN}(z; 0, 25)$ . Here we took  $b = 32$ , and selected the location parameters  $\mu_j$  as iid samples from a standard normal distribution. We ran GCMC using the Gibbs sampler form of Algorithm 1, for values of  $\lambda$  between  $10^{-5}$  and 10. For comparison, we also drew samples from each subposterior distribution as defined in Section 2.4, combining the samples using various approaches. These are the CMC averaging of Scott et al. (2016); the nonparametric density product estimation (NDPE) approach of Neiswanger, Wang, and Xing (2014); and the Weierstrass rejection sampling (WRS) combination technique of Wang and Dunson (2013), using their R implementation (<https://github.com/wwrehard/weierstrass>). In each case, we ran the algorithm 25 times, drawing  $N = 10^5$  samples.

To demonstrate the role of  $\lambda$  in the bias–variance decomposition (11), Table 1 presents the means and standard deviations of estimates of  $\pi(\varphi)$ , for various test functions  $\varphi$ . In estimating the first moment of  $\pi$ , GCMC generates a low-bias estimator when  $\lambda$  is chosen to be sufficiently small; however, as expected, the variance of such estimators increases when very small values

**Table 1.** True values and estimates of  $\pi(\varphi)$ , for various test functions  $\varphi$ , for the log-normal toy model.

Algorithm	$\varphi(z) = z$ ( $\pi(\varphi) = 1.141$ )	$\varphi(z) = z^5$ ( $\pi(\varphi) = 2.644$ )	$\varphi(z) = \log(z)$ ( $\pi(\varphi) = 0.1164$ )	
GCMC	$\lambda = 10^1$	1.329 ± 0.003	121.154 ± 10.487	0.1151 ± 0.0019
	$\lambda = 10^0$	1.159 ± 0.002	3.901 ± 0.037	<b>0.1165 ± 0.0014</b>
	$\lambda = 10^{-1}$	<b>1.144 ± 0.003</b>	<b>2.763 ± 0.044</b>	0.1173 ± 0.0030
	$\lambda = 10^{-2}$	1.140 ± 0.011	2.648 ± 0.143	0.1150 ± 0.0090
	$\lambda = 10^{-3}$	1.142 ± 0.022	2.661 ± 0.295	0.1191 ± 0.0199
	$\lambda = 10^{-4}$	1.120 ± 0.077	3.505 ± 1.136	0.1630 ± 0.0651
	$\lambda = 10^{-5}$	1.400 ± 0.110	6.195 ± 2.217	0.3283 ± 0.0810
CMC	1.073 ± 0.010	16.092 ± 5.675	0.0135 ± 0.0095	
NDPE	1.148 ± 0.029	2.800 ± 0.385	0.1231 ± 0.0246	
WRS	1.111 ± 0.007	2.444 ± 0.086	0.0862 ± 0.0063	

NOTE: Estimates obtained using global consensus Monte Carlo with various values of  $\lambda$ , and three embarrassingly parallel methods (see main text for abbreviations). For each method the mean estimate ± Monte Carlo standard error is presented, as computed over 25 replicates; the estimator corresponding to the lowest mean squared error is printed in bold.

of  $\lambda$  are chosen. While the other methods produce estimators of reasonably low variance, these exhibit somewhat higher bias. For CMC, the bias is especially pronounced when estimating higher moments of the posterior distribution, as exemplified by the estimates of the fifth moment. Note however that high biases are also introduced when using GCMC with large values of  $\lambda$  (as seen here with  $\lambda = 10$ ), for which  $\pi_\lambda$  is a poor approximation of  $\pi$ .

Also of note are estimates of  $\int \log(z)\pi(z) dz$ , corresponding to the mean of the Gaussian model of which this a reparameterization. While GCMC performs well across a range of  $\lambda$  values, the other methods perform less favorably; CMC produces an estimate that is incorrect by an order of magnitude. While this could be solved by a simple reparameterization of the problem in this case, in more general settings no such straightforward solution may exist.

In Section S4.2 of the supplementary materials, we present second example based on a log-normal model, demonstrating the robustness of these methods to permutation and repartitioning of the data.

## 4.2. Logistic Regression

Binary logistic regression models are commonly used in settings related to marketing. In web design for example, A/B testing may be used to determine which content choices lead to maximized user interaction, such as the user clicking on a product for sale.

We assume that we have a dataset of size  $n$  formed of responses  $\eta_l \in \{-1, 1\}$ , and vectors  $\xi_l \in \{0, 1\}^d$  of binary covariates, where  $l \in \{1, \dots, n\}$ . The likelihood contribution of each block of data then takes the form  $f_j(z) = \prod_l S(\eta_l z^T \xi_l)$ ,  $z \in \mathbb{R}^d$ , where the product is taken over those indices  $l$  included in the  $j$ th block of data, and  $S$  denotes the logistic function,  $S(x) = (1 + \exp(x))^{-1}$ .

For the prior  $\mu$ , we use a product of independent zero-mean Gaussians, with standard deviation 20 for the parameter corresponding to the intercept term, and 5 for all other parameters. For the Markov transition densities in GCMC, we use multivariate Gaussian densities:  $K_j^{(\lambda)}(z, x) = \mathcal{N}(x; z, \lambda I)$  for each  $j \in \{1, \dots, b\}$ .

We investigated several such simulated datasets and the efficacy of various approaches in approximating the true posterior  $\pi$ . To illustrate the bias–variance trade-off described in Section 2.3, in the presentation of these results we focus on the estimation of the posterior first moment; denoting the identity function on  $\mathbb{R}^d$  by  $\text{Id}$ , we may write this as  $\pi(\text{Id})$ . While our global consensus approach was consistently successful in forming estimators with low mean squared error in each component, in low-dimensional settings the application of CMC often resulted in marginal improvements. However, in many higher-dimensional settings, the estimators resulting from CMC exhibited relatively large biases.

We present here an example in which the  $d$  predictors correspond to  $p$  binary input variables, their pairwise products, and an intercept term, so that  $d = 1 + p + \binom{p}{2}$ . In settings where the interaction effects corresponding to these pairwise products are of interest, the dimensionality  $d$  of the space can be very large compared to  $p$ . We used a simulated dataset with  $p = 20$  input



variables, resulting in a parameter space of dimension  $d = 211$ . The data comprise  $n = 80,000$  observations, split into  $b = 8$  equally sized blocks. Each observation of the 20 binary variables was generated from a Bernoulli distribution with parameter 0.1, and for each vector of covariates, the response was generated from the correct model, for a fixed underlying parameter vector  $z^*$ .

#### 4.2.1. Metropolis-Within-Gibbs

We applied GCMC for values of  $\lambda$  between  $10^{-2}$  and 1. We used a Metropolis-within-Gibbs formulation of Algorithm 1, sampling directly from the Gaussian conditional distribution of  $Z$  given  $X_{1:b}$ . To sample approximately from the conditional distributions of each  $X_j$  given  $Z$  we used Markov kernels  $P_{j,z}^{(\lambda)}$  comprising  $k = 20$  iterations of a random walk Metropolis kernel.

As mentioned in Section 2.2, in settings of high communication latency our approach allows a greater proportion of wall-clock time to be spent on likelihood contributions, compared to an MCMC chain directly targeting the full posterior  $\pi$ . To compare across settings, we therefore consider an abstracted distributed setting as discussed in Section S1.1 of the supplementary materials, here assuming that the latency is 10 times the time taken to compute each partial likelihood  $f_j$  (in the notation of Section S1.1,  $C = 10\ell$ ).

We also compare with the same embarrassingly parallel approaches as in Section 4.1 (CMC, NDPE, WRS), which are comparatively unaffected by communication latency. For these methods, we again used random walk Metropolis to draw samples from each subposterior distribution. To ease computation, we thinned these chains before applying the combination step; in practice, the estimators obtained using these thinned chains behaved very similarly to those obtained using all subposterior samples.

To provide a “ground truth” against which to compare the results we ran a random walk Metropolis chain of length 500,000 targeting  $\pi$ . For all our random walk Metropolis samplers we used Gaussian proposal kernels. To determine the covariance matrices of these, we formed a Laplace approximation of the target density following the approach of Chopin and Ridgway (2017), scaling the resulting covariance matrix optimally according to results of Roberts and Rosenthal (2001).

For each algorithmic setting, we ran the corresponding sampler 25 times. To compare the resulting estimators of the posterior mean we computed the mean squared error of each of the  $d$  components of the posterior mean, summing these to obtain a “mean sum of squared errors.”

Table 2 compares the values obtained by each algorithm after an approximate wall-clock time equal to 200,000 times the time taken to compute a single partial likelihood  $f_j$ . Accounting for latency in the abstracted distributed setting described above, the GCMC approach is able to generate 5000 approximate posterior samples during this time, spending 50% of time on likelihood computations. In contrast, a direct MCMC approach generates 9523 samples, but would only spend 4.8% of the time on likelihood computations, with the remainder lost due to latency.

The result is that the estimators generated by GCMC for appropriately chosen  $\lambda$  exhibit lower mean sums of squared

**Table 2.** Mean sum of squared errors over all  $d$  components of estimates of the posterior mean for the logistic regression model, formed using various algorithmic approaches as described in the main text, during an approximate wall-clock time equal to 200,000 times that required to compute a single partial likelihood  $f_j$ .

Algorithm		Mean sum of squared errors
GCMC	$\lambda = 10^0$	0.1835
	$\lambda = 10^{-0.5}$	0.1379
	$\lambda = 10^{-1}$	0.0770
	$\lambda = 10^{-1.5}$	<b>0.0478</b>
	$\lambda = 10^{-2}$	0.0662
CMC		0.3710
NDPE		0.8476
WRS		0.6402
Direct MCMC		0.0884

NOTE: All values computed over 25 replicates, with the lowest value printed in bold.

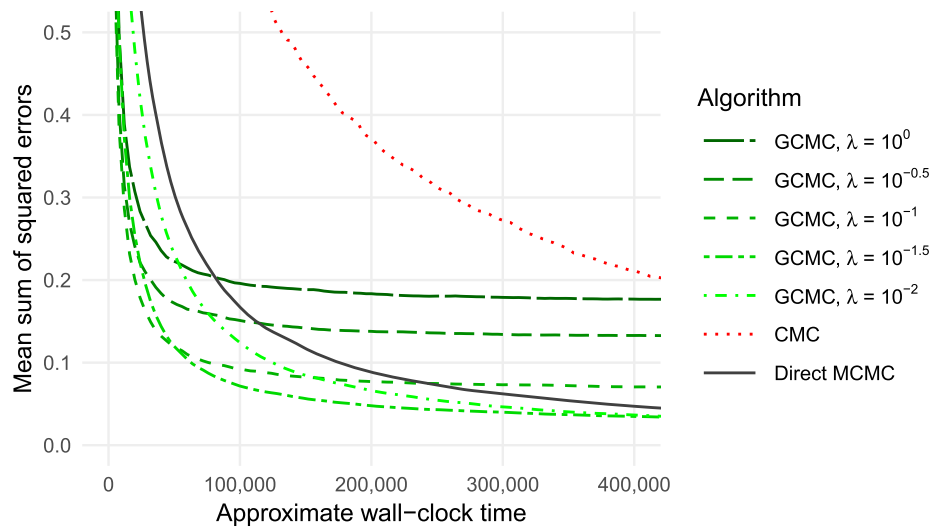
errors: we conduct many more accept/reject steps in each round of inter-node communication than if we were to directly target  $\pi$ , and so it becomes possible to achieve faster mixing of the  $Z$ -chain (and a better estimator) compared to such a direct approach. This may be seen when comparing the effective sample size (ESS) of each chain, where we estimate this via the “batch means” approach of Vats, Flegal, and Jones (2019): we find that the average ESS of the direct MCMC chains is only 1111, while depending on the choice of  $\lambda$ , the shorter GCMC chains have average ESS values between 1327 and 4577.

Despite being unaffected by latency and therefore allowing many more samples to be drawn, the embarrassingly parallel approaches (CMC, NDPE, WRS) perform poorly compared to GCMC. This is particularly true of the NDPE method of Neiswanger, Wang, and Xing (2014): while asymptotically exact even in non-Gaussian settings, the resulting estimator is based on kernel density estimators and is not effective in this high-dimensional setting.

Figure 2 shows the mean sums of squared errors as a function of the approximate wall-clock time (for simplicity we include only the best-performing of the three embarrassingly parallel methods, omitting the results for NDPE and WRS). We see that for large enough  $\lambda$ , the GCMC estimators  $\pi_\lambda^N(\text{Id})$  exhibit rather lower values than the corresponding CMC and “direct” MCMC estimators. As the number of samples used grows, the squared bias of these estimators begins to dominate, and so smaller  $\lambda$  values result in lower mean squared errors. As  $\lambda$  becomes smaller the autocorrelation of the resulting  $Z$ -chain increases; indeed we found that for  $\lambda$  too small, the GCMC estimator  $\pi_\lambda^N(\text{Id})$  will always have a greater mean squared error than the “direct” MCMC estimator, no matter how much time is used. Of course, since an MCMC estimator formed by directly targeting  $\pi$  is consistent in  $N$ , given sufficient time such an estimator will always outperform estimators formed using GCMC, which are biased for any  $\lambda$ . However, in many practical big data settings it may be infeasible to draw large numbers of samples using the available time budget.

#### 4.2.2. Sequential Monte Carlo

We also applied the SMC procedure to this logistic regression model. While we found that the SMC approach was most effective in lower-dimensional settings (see Section S4.1 of the supplementary materials for a simple example) in which it is less computationally expensive, the SMC procedure can be more



**Figure 2.** Mean sum of squared errors over all  $d$  components of estimates of the posterior mean for the logistic regression model, formed using various algorithmic approaches as described in the main text. Values plotted against the approximate wall-clock time, relative to the time taken to compute a single partial likelihood term. All values computed over 25 replicates.

widely useful as a means of “refining” the estimator formed using a single  $\lambda$  value, as discussed in Section 3.

We used  $N = 1250$  particles, initializing the particle set by thinning the chain generated by the Metropolis-within-Gibbs procedure with  $\lambda = 10^{-1}$ . To generate a sequence of subsequent  $\lambda$  values we used the adaptive procedure of Zhou, Johansen, and Aston (2016), using tuning parameter  $\text{CESS}^* = 0.98$ . For the Markov kernels  $M_p$ , we used Metropolis-within-Gibbs kernels as previously, with each update of  $X_j$  given  $Z$  comprising  $k = 50$  iterations of a random walk Metropolis kernel.

The estimator  $\pi_{\lambda_0}^N(\text{Id})$  formed using the initial particle set was found to have a mean sum of squared errors of 0.0692. After a fixed number of iterations ( $n = 100$ ) the resulting SMC estimate exhibited a mean sum of squared errors of 0.0418; this represents a decrease of 40%, and has the benefit of avoiding the need to carefully specify a single value for  $\lambda$ .

Used alone, the bias correction procedure of Section 3.1 was found to perform best in lower-dimensional settings (as in Section S4.1 of the supplementary materials); here, it resulted in a mean sum of squared errors of 0.0682 after 100 iterations. However, improved results were obtained using the stopping rule we propose in Section S3.2 of the supplementary materials (with stopping parameter  $\kappa = 15$ ), which is based on our proposed bias correction procedure. The estimator selected by this stopping rule, which automatically determines when to terminate the algorithm, obtained a mean sum of squared errors of 0.0367, a decrease of 47% from the estimator generated using the initial particle set.

## 5. Conclusion

We have presented a new framework for sampling in distributed settings, demonstrating its application on some illustrative examples in comparison to embarrassingly parallel approaches. Given that our proposed approach makes no additional assumptions on the form of the likelihood beyond its factorized form as in (1), we expect that our algorithm will be most effective in those big data settings for which approximate Gaussianity of

the likelihood contributions may not hold. These may include high-dimensional settings, for which some subsets of the data may be relatively uninformative about the parameter. In such cases, the likelihood contributions may be highly non-Gaussian, so that the CMC approach of averaging across chains results in estimates of high bias; simultaneously, the high dimensionality may preclude the use of alternative combination techniques (e.g., the use of kernel density estimates).

This framework may be of use in serial settings. As previously noted, the contemporaneous work of Vono, Dobigeon, and Chainais (2019) presents an example in which the use of an analogous framework (with  $b = 1$ ) results in more efficient simulation than approaches that directly target the true posterior. Our proposed SMC sampler implementation and the associated bias correction technique may equally be applied to such settings, reducing the need to specify a single value of the regularization parameter  $\lambda$ .

There is potential for further improvements to be made to the procedures we present here. In the SMC case for example, while our proposed use of weighted least squares as a bias correction technique is simple, nonlinear procedures (such those proposed in an ABC context by Blum and François 2010) may provide a more robust alternative, with some theoretical guarantees. We also stress that the MCMC and SMC procedures presented here constitute only two possible approaches to inference within the instrumental hierarchical model that we propose, and there is considerable scope for alternative sampling algorithms to be employed within this global consensus framework.

## Supplementary Materials

### Supplement to “Global consensus Monte Carlo”:

“GCMCsupplement.pdf” includes supplementary materials covering implementation considerations, theoretical analysis for a simple model, and heuristic procedures for the proposed SMC sampler (with numerical demonstrations).

**R code for examples:** “GCMCexamples.zip” contains R scripts used to generate the numerical results and figures presented here and in the supplement.

## Funding

The authors gratefully acknowledge the support of The Alan Turing Institute under the EPSRC grant EP/N510129/1, the Lloyd's Register Foundation–Alan Turing Institute Programme on Data-Centric Engineering, and the EPSRC under grants EP/M508184/1, EP/R034710/1 and EP/T004134/1.

## ORCID

Lewis J. Rendell  <http://orcid.org/0000-0002-9489-3094>  
 Adam M. Johansen  <http://orcid.org/0000-0002-3531-7628>  
 Anthony Lee  <http://orcid.org/0000-0001-7765-0616>  
 Nick Whiteley  <http://orcid.org/0000-0002-8337-626X>

## References

- Bardenet, R., Doucet, A., and Holmes, C. (2014), “Towards Scaling Up Markov Chain Monte Carlo: An Adaptive Subsampling Approach,” in *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pp. 405–413. [5]
- Beaumont, M. A., Zhang, W., and Balding, D. J. (2002), “Approximate Bayesian Computation in Population Genetics,” *Genetics*, 162, 2025–2035. [6]
- Bertsekas, D. P., and Tsitsiklis, J. N. (1989), *Parallel and Distributed Computation: Numerical Methods*, Englewood Cliffs, NJ: Prentice-Hall. [3]
- Blum, M. G., and François, O. (2010), “Non-Linear Regression Models for Approximate Bayesian Computation,” *Statistics and Computing*, 20, 63–73. [6,9]
- Boyd, S., Parikh, N., Chu, E., Peleato, B., and Eckstein, J. (2011), “Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers,” *Foundations and Trends in Machine Learning*, 3, 1–122. [1,3]
- Chan, H. P., and Lai, T. L. (2013), “A General Theory of Particle Filters in Hidden Markov Models and Some Applications,” *The Annals of Statistics*, 41, 2877–2904. [6]
- Chopin, N. (2002), “A Sequential Particle Filter Method for Static Models,” *Biometrika*, 89, 539–552. [5]
- Chopin, N., and Ridgway, J. (2017), “Leave Pima Indians Alone: Binary Regression as a Benchmark for Bayesian Computation,” *Statistical Science*, 32, 64–87. [8]
- Crooks, G. E. (1998), “Nonequilibrium Measurements of Free Energy Differences for Microscopically Reversible Markovian Systems,” *Journal of Statistical Physics*, 90, 1481–1487. [5]
- Dai, H., Pollock, M., and Roberts, G. (2019), “Monte Carlo Fusion,” *Journal of Applied Probability*, 56, 174–191. [4]
- Dean, J., and Ghemawat, S. (2008), “MapReduce: Simplified Data Processing on Large Clusters,” *Communications of the ACM*, 51, 107–113. [1]
- Del Moral, P., Doucet, A., and Jasra, A. (2006), “Sequential Monte Carlo Samplers,” *Journal of the Royal Statistical Society, Series B*, 68, 411–436. [5]
- Douc, R., Cappé, O., and Moulines, E. (2005), “Comparison of Resampling Schemes for Particle Filtering” in *Proceedings of the 4th International Symposium on Image and Signal Processing and Analysis*, IEEE, pp. 64–69. [5]
- Doucet, A., and Johansen, A. M. (2011), “A Tutorial on Particle Filtering and Smoothing: Fifteen Years Later,” in *Handbook of Nonlinear Filtering*, eds. D. Crisan and B. Rozovskii, Oxford: Oxford University Press, pp. 656–704. [5]
- Doucet, A., and Lee, A. (2018), “Sequential Monte Carlo methods,” in *Handbook of Graphical Models*, eds. M. Maathuis, M. Drton, S. L. Lauritzen, and M. Wainwright, Boca Raton, FL: CRC Press, pp. 165–189. [5]
- Finke, A., Doucet, A., and Johansen, A. M. (2020), “Limit Theorems for Sequential MCMC Methods,” *Advances in Applied Probability*, 52, 377–403. [5]
- Gerber, M., Chopin, N., and Whiteley, N. (2019), “Negative Association, Ordering and Convergence of Resampling Methods,” *The Annals of Statistics*, 47, 2236–2260. [5]
- Gilks, W. R., and Berzuini, C. (2001), “Following a Moving Target—Monte Carlo Inference for Dynamic Bayesian Models,” *Journal of the Royal Statistical Society, Series B*, 63, 127–146. [5]
- Huggins, J., Campbell, T., and Broderick, T. (2016), “Coresets for Scalable Bayesian Logistic Regression,” in *Advances in Neural Information Processing Systems*, pp. 4080–4088. [5]
- Jasra, A., Stephens, D. A., Doucet, A., and Tsagaris, T. (2011), “Inference for Lévy-Driven Stochastic Volatility Models via Adaptive Sequential Monte Carlo,” *Scandinavian Journal of Statistics*, 38, 1–22. [5]
- Jordan, M. I., Lee, J. D., and Yang, Y. (2019), “Communication-Efficient Distributed Statistical Inference,” *Journal of the American Statistical Association*, 114, 668–681. [5]
- Korattikara, A., Chen, Y., and Welling, M. (2014), “Austerity in MCMC Land: Cutting the Metropolis-Hastings Budget,” in *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*. [5]
- Lee, A., and Whiteley, N. (2018), “Variance Estimation in the Particle Filter,” *Biometrika*, 105, 609–625. [6]
- Liu, J. S., and Chen, R. (1995), “Blind Deconvolution via Sequential Imputations,” *Journal of the American Statistical Association*, 90, 567–576. [5]
- Maclaurin, D., and Adams, R. P. (2014), “Firefly Monte Carlo: Exact MCMC With Subsets of Data,” in *Proceedings of the 30th International Conference on Uncertainty in Artificial Intelligence (UAI-14)*, pp. 543–552. [5]
- Marin, J. M., Pudlo, P., Robert, C. P., and Ryder, R. J. (2012), “Approximate Bayesian Computational Methods,” *Statistics and Computing*, 22, 1167–1180. [3]
- Minsker, S., Srivastava, S., Lin, L., and Dunson, D. (2014), “Scalable and Robust Bayesian Inference via the Median Posterior,” in *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pp. 1656–1664. [4]
- Neal, R. M. (2001), “Annealed Importance Sampling,” *Statistics and Computing*, 11, 125–139. [5]
- Neiswanger, W., Wang, C., and Xing, E. (2014), “Asymptotically Exact, Embarrassingly Parallel MCMC,” in *Proceedings of the 30th International Conference on Uncertainty in Artificial Intelligence (UAI-14)*, pp. 623–632. [4,7,8]
- Rabinovich, M., Angelino, E., and Jordan, M. I. (2015), “Variational Consensus Monte Carlo,” in *Advances in Neural Information Processing Systems*, pp. 1207–1215. [4]
- Rendell, L. J. (2020), “Sequential Monte Carlo Variance Estimators and Global Consensus,” Ph.D. thesis, University of Warwick. [3]
- Rendell, L. J., Johansen, A. M., Lee, A., and Whiteley, N. (2018), “Global Consensus Monte Carlo,” arXiv no. 1807.09288. [2]
- Roberts, G. O., and Rosenthal, J. S. (2001), “Optimal Scaling for Various Metropolis–Hastings Algorithms,” *Statistical Science*, 16, 351–367. [8]
- Scheffé, H. (1947), “A Useful Convergence Theorem for Probability Distributions,” *The Annals of Mathematical Statistics*, 18, 434–438. [2]
- Scott, S. L. (2017), “Comparing Consensus Monte Carlo Strategies for Distributed Bayesian Computation,” *Brazilian Journal of Probability and Statistics*, 31, 668–685. [4]
- 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. [1,4,5,7]
- Srivastava, S., Cevher, V., Dinh, Q., and Dunson, D. (2015), “WASP: Scalable Bayes via Barycenters of Subset Posteriors,” in *Artificial Intelligence and Statistics*, pp. 912–920. [4]
- Vats, D., Flegal, J. M., and Jones, G. L. (2019), “Multivariate Output Analysis for Markov Chain Monte Carlo,” *Biometrika*, 106, 321–337. [8]
- Vono, M., Dobigeon, N., and Chainais, P. (2018), “Sparse Bayesian Binary Logistic Regression Using the Split-and-Augmented Gibbs Sampler,” in *Proceedings of the IEEE International Workshop on Machine Learning for Signal Processing*. [2]
- (2019), “Split-and-Augmented Gibbs Sampler—Application to Large-Scale Inference Problems,” *IEEE Transactions on Signal Processing*, 67, 1648–1661. [2,3,4,9]

- Vono, M., Paulin, D., and Doucet, A. (2019), “Efficient MCMC Sampling With Dimension-Free Convergence Rate Using ADMM-Type Splitting,” arXiv no. 1905.11937. [2]
- Wang, X., and Dunson, D. B. (2013), “Parallelizing MCMC via Weierstrass Sampler,” arXiv no. 1312.4605. [4,7]
- Wang, X., Guo, F., Heller, K. A., and Dunson, D. B. (2015), “Parallelizing MCMC With Random Partition Trees,” in *Advances in Neural Information Processing Systems*, pp. 451–459. [4]
- Xu, M., Lakshminarayanan, B., Teh, Y. W., Zhu, J., and Zhang, B. (2014), “Distributed Bayesian Posterior Sampling via Moment Sharing,” in *Advances in Neural Information Processing Systems*, pp. 3356–3364. [5]
- Zhou, Y., Johansen, A. M., and Aston, J. A. (2016), “Towards Automatic Model Comparison: An Adaptive Sequential Monte Carlo Approach,” *Journal of Computational and Graphical Statistics*, 25, 701–726. [5,9]