Taylor & Francis
Taylor & Francis Group

Journal of Computational and Graphical Statistics

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/ucgs20

Asynchronous and Distributed Data Augmentation for Massive Data Settings

Jiayuan Zhou, Kshitij Khare & Sanvesh Srivastava

To cite this article: Jiayuan Zhou, Kshitij Khare & Sanvesh Srivastava (2023) Asynchronous and Distributed Data Augmentation for Massive Data Settings, Journal of Computational and Graphical Statistics, 32:3, 895-907, DOI: 10.1080/10618600.2022.2130928

To link to this article: https://doi.org/10.1080/10618600.2022.2130928

JOURNAL OF COMPUTATIONAL AND GRAPHICAL STATISTICS
2023, VOL. 32, NO. 3, 895–907
https://doi.org/10.1080/10618600.2022.2130928

Asynchronous and Distributed Data Augmentation for Massive Data Settings

Jiayuan Zhou[a], Kshitij Khare[a], and Sanvesh Srivastava[b]

[a]Department of Statistics, University of Florida, Gainesville, FL; [b]Department of Statistics and Actuarial Science, The University of Iowa, Iowa City, IA

ABSTRACT

Data augmentation (DA) algorithms are slow in massive data settings due to multiple passes through the entire data. We address this problem by developing a DA extension that exploits asynchronous and distributed computing. The extended DA algorithm is called Asynchronous and Distributed (AD) DA with the original DA as its parent. Any ADDA is indexed by a parameter $r \in (0, 1)$ and starts by dividing the entire data into k disjoint subsets and storing them on k processes. Every iteration of ADDA augments only an r-fraction of the k data subsets with some positive probability and leaves the remaining $(1 - r)$-fraction of the augmented data unchanged. The parameter draws are obtained using the r-fraction of new and $(1 - r)$-fraction of old augmented data. We show that the ADDA Markov chain is Harris ergodic with the desired stationary distribution under mild conditions on the parent DA algorithm. We demonstrate that ADDA is significantly faster than its parent for many (k, r) choices in three representative models. We also establish the geometric ergodicity of the ADDA Markov chain for all the three models, which yields asymptotically valid standard errors for estimates of desired posterior quantities. Supplementary materials for this article are available online.

ARTICLE HISTORY
Received March 2022
Accepted September 2022

KEYWORDS
Asynchronous computations;
Bayesian inference;
Divide-and-conquer;
Geometric ergodicity;
Markov chain Monte Carlo

1. Introduction

DA algorithms are a popular choice for Bayesian inference using Markov chain Monte Carlo due to their simplicity and numerical stability; however, DA algorithms are slow in massive data applications because they pass through the full data in every iteration. Taking advantage of asynchronous and distributed computations, we propose the ADDA class of algorithms as a scalable generalization of any DA-type algorithm. ADDA is useful for fitting flexible Bayesian models to arbitrarily large datasets, retains the convergence properties of its parent DA, and reduces to the parent DA under certain assumptions.

Consider the setup of DA algorithms. In a typical DA application, we augment "missing data" to the observed data and obtain the "augmented data" model, where the term missing data is interpreted broadly to include any additional parameters or latent variables. Under the augmented data model, the imputation (I) step draws the missing data from their conditional distribution given the observed data and the current parameter value. The I step is followed by the prediction (P) step that draws the "original" parameter from its conditional distribution given the augmented data. This completes one cycle of a DA algorithm. Starting from some initial value of the parameter, the I and P steps are repeated sequentially to obtain a Markov chain for the missing data and parameter. The marginal parameter samples from the DA algorithm form a Markov chain with the posterior distribution of parameters given the observed data as its stationary distribution (Hobert 2011; Robert and Casella 2013).

The convergence of the DA Markov chain to its stationary distribution can be extremely slow. The inefficiency of DA-type algorithms in massive data settings has received significant attention. This is addressed using *online* EM and subsampling based algorithms; see (Nemeth and Fearnhead 2021) for a recent overview. All these algorithms are scalable, have convergence guarantees, and are broadly applicable; however, the performance of these algorithms is sensitive to the specification of tuning parameters, including subsample size, gradient step-size, and the *learning rate*. Their optimal performance requires proposal tuning, which is a major limitation in their use as off-the-shelf algorithms for Bayesian inference.

Distributed Bayesian inference approaches based on the divide-and-conquer technique also rely on DA and exploit its ease of implementation. A typical application involves dividing the full data into smaller subsets, obtaining parameter draws in parallel on all the subsets using a DA-type algorithm, and combining parameter draws from all the subsets. The combined parameter draws are used for inference and predictions. A variety of algorithms have been developed at an increasing level of generality, but they have two major limitations (Scott et al. 2016; Li, Srivastava, and Dunson 2017; Srivastava, Li, and Dunson 2018; Xue and Liang 2019; Jordan, Lee, and Yang 2019; Wang and Srivastava 2021). First, the combined parameter draws do not provide a Markov chain with the target posterior based on the entire data as the stationary distribution, which implies that quantification of the Monte Carlo error is challenging. Second, the theoretical guarantees of these methods are based on a normal approximation of

CONTACT Sanvesh Srivastava ssrivastva@uiowa.edu Department of Statistics and Actuarial Science, The University of Iowa, Iowa City, IA.
Supplementary materials for this article are available online. Please go to www.tandfonline.com/r/JCGS.

the target posterior as the subset sample size tends to infinity. No guarantees are available about distance between the target posterior distribution and that of the combined parameter draws.

Addressing both these problems, the proposed ADDA offers an simple approach for bypassing problems due to the slow I step using *asynchronous* and *distributed* computations, but at the same time producing a *single Markov chain with the target posterior as a marginal of its stationary distribution*. The key observation which enables such a construction is that for many DA settings, the missing data can be partitioned into several sub-blocks such that two conditions are satisfied. First, all these sub-blocks are mutually independent given the original parameter and the observed data. Second, the conditional posterior distribution (given the original parameter) of each sub-block depends only on an exclusive subset of the observed data; see the representative examples in Sections 3.1–3.3 for greater details.

An ADDA algorithm starts by reserving $(k + l)$ computing units called *processes*, where $l \ll k$, k and l are the number of workers and managers, and processes could be cores in a CPU or machines in a cluster. The full data are divided into smaller k disjoint subsets and stored on the k worker processes. ADDA performs the I step in parallel on the k workers. They communicate the results of their I steps to the manager processes, which perform the P step. The managers also track progress of ADDA by maintaining the latest copies of I step results for every worker. For given $r \in (0, 1)$ and $\epsilon \in (0, 1)$, the managers wait for all workers to return their results with probability ϵ. With probability $1 - \epsilon$ the managers wait to receive the I step results from an r-fraction of the workers and perform the P step using an data augmented model that depends on the r-fraction of new I step results and the $(1 - r)$-fraction of old I step results. This sequence of I and P steps is repeated until convergence. ADDA reduces to a distributed generalization of the parent DA algorithm when $r = 1$.

The parameter ϵ is introduced as a theoretical device for ensuring that any ADDA algorithm produces a Markov chain. Any positive ϵ ensures that the missing data sub-block corresponding to each worker has a positive probability to be updated at every iteration. This is crucial for establishing theoretical results for the ADDA chain, including Harris ergodicity and geometric ergodicity. In practice, ϵ can be set small for faster computations.

It is important to compare and contrast the proposed ADDA algorithm with Asynchronous Gibbs sampling algorithms. This class of algorithms have been initially developed for topic modeling (Newman et al. 2009). The only similarity between this class of algorithms and ADDA is that the data are partitioned and stored on the workers, which run the sampling algorithm locally. The manager process is absent in asynchronous Gibbs sampling because every worker draws its local set of parameters and latent variables from the local full conditional without waiting for the full Gibbs cycle to finish. This also implies that the parameter and latent variable updates do not form a Markov chain. In simple Gaussian models, this strategy produces draws from a distribution that fails to converge to the target (Johnson, Saunderson, and Willsky 2013).

Terenin, Simpson, and Draper (2020) fix this issue by introducing a correction step that allows a worker to accept or reject a draw with a probability based on the corresponding acceptance ratio. This additional Metropolis step increases the computational burden and the amount of information that needs to be communicated among the workers, but the authors are able to establish convergence to the desired stationary distribution under additional regularity conditions. The sequence of iterates generated by this exact algorithm still do not form a Markov chain in general, and hence one does not have access to the standard Markov chain central limit theorem (CLT) based approaches to quantify the Monte Carlo standard error (see the discussion below). Developing a general theoretical understanding of these asynchronous Gibbs algorithms is still an active area of research (Atchadé and Wang 2021).

Any ADDA algorithm inherits several attractive properties of its parent DA. First, ADDA is numerically stable. Second, ADDA is the stochastic counterpart of the distributed EM (DEM) and is independent of the computer architecture (Srivastava, DePalma, and Liu 2019). The only similarity between the ADDA and DEM algorithms is that the I and P steps are the stochastic counterparts of the E and M steps. We focus on understanding the distributional convergence of the stochastic iterates produced by our ADDA. Our geometric ergodicity results help provide standard errors for these approximations, which are justified by the Markov chain central limit theorem. The analogous results are absent in Srivastava, DePalma, and Liu (2019). Finally, while the marginal sequence of original parameter values generated by the ADDA algorithm is not a Markov chain (unlike the DA algorithm), the joint sequence of original and augmented parameter values is a Markov chain whose stationary distribution has the targeted posterior as a marginal. This facilitates a simpler theoretical analysis of the Monte Carlo error based on the familiar tools for analyzing Markov chains in sampling-based posterior inference.

We now describe our key contributions. We develop the general ADDA framework and establish Harris ergodicity of the ADDA Markov chain in Theorem 2.1 (Section 2.1). We apply this framework on three representative DA algorithms for two kinds of massive data settings (Sections 3.1–3.3). The first is the "massive n setting," where the number of observations or samples n is extremely large, typically in the order of millions or larger. We consider two illustrative examples: the Polya-Gamma DA algorithm for Bayesian logistic regression (Polson, Scott, and Windle 2013) and the marginal DA algorithm for linear mixed-effects modeling (Van Dyk and Meng 2001). The number of augmented parameters in both DAs equals the number of observations, implying that their I steps are prohibitively slow. Algorithms 1 and 3 outline their ADDA extensions. The other setting arises in models with extremely large number of variables p (i.e., "massive p setting"). We develop an ADDA extension in Algorithm 2 for Bayesian lasso DA algorithm, which has been used for high-dimensional Bayesian variable selection using shrinkage priors (Rajaratnam et al. 2019).

The two hallmarks of ADDA are distributed and asynchronous posterior computations. Parallel processing by distributing the independent draws of the appropriate sub-blocks of the missing data to the k workers obviously leads to faster computations with no adverse impact on mixing of the Markov chain. The asynchronous updates, however, come with a tradeoff. Smaller values of the fraction r lead to faster

computations per iteration, but also slow down the mixing of the Markov chain; therefore, compared to the parent DA, the same number of iterations of the ADDA algorithm take much less time but provide less accurate estimates of desired posterior quantities. The idea, as demonstrated by our simulations, is that typically the computational gain is quite significant with a comparatively small loss of accuracy. For example, the real data analyses using logistic regression and linear mixed-effects models show that ADDA is anywhere between three to five times faster and only around 2% less accurate than its parent after 10,000 iterations; see Section 4.3 and Figure 4.

The iterates generated by the ADDA algorithm form a Markov chain whose stationary distribution has the target posterior as a marginal. This potentially allows us to directly leverage standard approaches to quantify the Monte Carlo standard error of the resulting estimators of posterior quantities. All asymptotically consistent estimators of the standard error, however, rely on the existence of a Markov chain CLT, which is typically established by showing geometric ergodicity; that is, the distribution of the Markov chain converges geometrically fast to the stationary distribution as the number of iterations increase. Establishing geometric ergodicity of statistical continuous state space Markov chains is known to be challenging (Jones and Hobert 2001). The drift and minorization analysis typically needed for this purpose can be quite involved and "a matter of art," requiring the analysis to be heavily adapted to the structure of individual Markov chains. The geometric ergodicity of the Markov chains corresponding to the three DA algorithms has been established, but adapting these analyses to establish similar results for the ADDA extensions is not feasible given the complications introduced by the asynchronous updates. Through a fresh analysis, we establish geometric ergodicity for the three ADDA extensions in Theorems 3.1–3.3.

2. Asynchronous and Distributed Data Augmentation

2.1. The General Framework

Consider the setup of a general ADDA algorithm. Assume that we have chosen an $r \in (0,1)$ and reserved k worker and 1 manager processes. Let θ be the model parameter, $\mathcal{D}_{\text{obs}}$ and $\mathcal{D}$ be the observed and missing data, $\mathcal{D}_{\text{aug}} = \{\mathcal{D}_{\text{obs}}, \mathcal{D}\}$ be the augmented data. Let $\mathbb{M}$ and Θ denote the missing data and the parameter spaces, respectively. We assume that both sets are equipped with countably generated σ-algebras, and respective σ-finite measures μ and ν. Let $f(\mathcal{D}, \theta \mid \mathcal{D}_{\text{obs}})$ denote the joint posterior density (with respect to $\mu \times \nu$) of $\mathcal{D}$ and θ, and $f(\mathcal{D} \mid \theta, \mathcal{D}_{\text{obs}})$ and $f(\theta \mid \mathcal{D}, \mathcal{D}_{\text{obs}})$ denote the conditional posterior densities of the missing data given the parameter, and the parameter given the augmented data, respectively. Let $\mathcal{D}_i$ denote the missing data subset distributed to worker i so that $\mathcal{D} = (\mathcal{D}_1, \ldots, \mathcal{D}_k)$. We assume that $\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_k$ are conditionally independent given $\mathcal{D}_{\text{obs}}, \theta$ so that $f(\mathcal{D} \mid \theta, \mathcal{D}_{\text{obs}}) = \prod_{i=1}^{k} f_i(\mathcal{D}_i \mid \theta, \mathcal{D}_{\text{obs}})$. If the parameter θ can also be divided into conditionally independent blocks, one could introduce more than one manager process, but for ease of exposition we restrict ourselves to a setting with one manager process.

For an $r \in (0,1)$ and an $\epsilon \in (0,1)$, any ADDA algorithm is implemented as follows:

1. The manager starts with some initial values $(\mathcal{D}^{(0)}, \theta^{(0)})$ at $t = 0$ and sends $\theta^{(0)}$ to the workers. For $t = 0, 1, \ldots, \infty$, the manager
   - (M-a) waits to receive only an r-fraction of updated $\mathcal{D}_i^{(t+1)}$s (see below) from the workers with probability $1 - \epsilon$, and with probability ϵ, waits to receive all the updated $\mathcal{D}_i^{(t+1)}$s from the workers;
   - (M-b) creates $\mathcal{D}^{(t+1)}$ by replacing the relevant $\mathcal{D}_i^{(t)}$s with the newly received $\mathcal{D}_i^{(t+1)}$;
   - (M-c) draws $\theta^{(t+1)}$ from $p(\theta \mid \mathcal{D}^{(t+1)}, \mathcal{D}_{\text{obs}})$; and
   - (M-d) sends $\theta^{(t+1)}$ to all the worker processes and resets $t = t + 1$.
2. For $t = 0, \ldots, \infty$, the worker i $(i = 1, \ldots, k)$
   - (W-a) waits to receive $\theta^{(t)}$ from the manager process;
   - (W-b) draws $\mathcal{D}_i^{(t+1)}$ from $p(\mathcal{D}_i \mid \mathcal{D}_{\text{obs}}, \theta^{(t)})$; and
   - (W-c) sends $\mathcal{D}_i^{(t+1)}$ to the manager process, resets $t = t+1$, and goes to (W-a) if $\theta^{(t+1)}$ is not received from the manager before the draw is complete; otherwise, it truncates the sampling process, resets $t = t + 1$, and goes to (W-b).

These steps are summarized into Asynchronous and Distributed (AD) I and P steps. For $t = 0, 1, \ldots, \infty$, the $(t+1)$th iteration of an ADDA algorithm has the following two steps:

AD-I step: Each worker draws $\mathcal{D}_i^{(t+1)}$ from $f(\mathcal{D}_i \mid \mathcal{D}_{\text{obs}}, \theta^{(t)})$ for $i = 1, \ldots, k$ in parallel and sends $\mathcal{D}_i^{(t+1)}$ to the manager (if the draw is finished before receiving $\theta^{(t+1)}$).

AD-P step: As soon as the manager receives the required fraction (r with probability $1 - \epsilon$ and 1 with probability ϵ) of updated $\mathcal{D}_i^{(t+1)}$ values from the workers, $\theta^{(t+1)}$ is drawn from $f(\theta \mid \mathcal{D}_1^{(t+1)}, \ldots, \mathcal{D}_k^{(t+1)}, \mathcal{D}_{\text{obs}})$, and sent to the workers.

Note that if the r-fraction update regime is chosen, then the manager sets $\mathcal{D}_i^{(t+1)} = \mathcal{D}_i^{(t)}$ for the remaining $(1 - r)$-fraction. As soon as the workers receive $\theta^{(t+1)}$, they stop any ongoing activity and proceed with the AD-I step for the $(t + 2)$th iteration.

One cycle of ADDA consists of AD-I step followed by AD-P step, and they are repeated sequentially to obtain the $\{(\mathcal{D}^{(t)}, \theta^{(t)}) : t = 0, 1, \ldots, \infty\}$ chain for the missing data and parameter. We emphasize again that at the end of iteration $t+1$, with probability $1-\epsilon$, only an r-fraction of $\mathcal{D}_1^{(t)}, \ldots, \mathcal{D}_k^{(t)}$ are replaced by the manager with new draws received from the workers, whereas the remaining $(1 - r)$-fraction of them haven't changed. The AD-I and AD-P in ADDA are distributed generalizations of the I and P steps in its parent DA. If $r = 1$, then AD-I and AD-P steps reduce to the I and P steps of their parent DA. The next section investigates crucial theoretical properties of the ADDA algorithm which guarantee that the ADDA Markov chain can be used to effectively approximate relevant posterior quantities.

2.2. Markov Property and Harris Ergodicity

The classical (parent) DA algorithm (the ADDA with $r = 1$) corresponds to a systematic scan two-block Gibbs sampler

from the joint density $f(\mathcal{D}, \theta \mid \mathcal{D}_{\text{obs}})$. Furthermore, in the classical DA setting, $\theta^{(t+1)}$ solely depends on $\mathcal{D}^{(t+1)}$ given $\mathcal{D}^{(t+1)}, \{(\mathcal{D}^{(j)}, \theta^{(j)})\}_{j=0}^{t}$ and $\mathcal{D}^{(t+1)}$ depends solely on $\theta^{(t)}$ given $\{(\mathcal{D}^{(j)}, \theta^{(j)})\}_{j=0}^{t}$. This implies that $\theta^{(t+1)}$ solely depends on $\theta^{(t)}$ given $\{\theta^{(j)}\}_{j=0}^{t}$ so that the marginal $\{\theta^{(t)}\}$ process is Markov. This fact is often useful when establishing theoretical properties such as geometric ergodicity of the DA Markov chain.

The ADDA can be interpreted as a *hybrid Gibbs sampler*. The AD-P step is a systematic scan step for sampling the parameter θ from its conditional posterior distribution given $\mathcal{D}$, but the AD-I step is a "random subset scan" step which updates a random subset of the missing data $\mathcal{D} = (\mathcal{D}_1, \ldots, \mathcal{D}_k)$. Other hybrid versions of systematic and random scan Gibbs steps have been considered in the literature. Backlund et al. (2021) consider a DA setup where the *parameter* θ is partitioned into two blocks. They construct and study a hybrid sampler that performs a systematic scan step for the entire missing data $\mathcal{D}$ at each iteration and updates exactly one of the two parameter blocks with fixed probabilities s and $1-s$, respectively. A general (non-DA) setting with multiple blocks is considered in Levine et al. (2005), where a sampler updates a single coordinate in each iteration. The choice of the block to update is made randomly based on a vector of fixed probabilities which depend on the block index that was updated in the previous iteration; therefore, this strategy subsumes the traditional systematic and random scan samplers as special cases. The ADDA is significantly different from the above strategies and updates a randomly chosen *subset* of the conditionally independent blocks of the missing data $\mathcal{D}$, while performing a systematic scan update of the entire parameter θ. Note also that the probabilities of choosing various relevant subsets at a given iteration can in general depend on the *current value* of the parameter θ.

A simple observation shows that $\left\{\mathcal{D}^{(t)}, \theta^{(t)}\right\}_{t=0}^{\infty}$ process in ADDA is Markov. For an ADDA algorithm with $r \in (0, 1)$, the knowledge of $(\mathcal{D}^{(t)}, \theta^{(t)})$ makes the entire past irrelevant for generating $(\mathcal{D}^{(t+1)}, \theta^{(t+1)})$. Specifically, $(\mathcal{D}^{(t+1)}, \theta^{(t+1)})$ is conditionally independent of $\left\{(\mathcal{D}^{(j)}, \theta^{(j)})\right\}_{j=0}^{t-1}$ given $(\mathcal{D}^{(t)}, \theta^{(t)})$; therefore, the process $\left\{\mathcal{D}^{(t)}, \theta^{(t)}\right\}_{t=0}^{\infty}$ is still Markov. It is also easily seen that both the AD-P step and the AD-I step leave $f(\mathcal{D}, \theta \mid \mathcal{D}_{\text{obs}})$ invariant. This is promising but to use the ADDA iterates to approximate relevant posterior expectations and quantiles, we need to establish Harris ergodicity. The following theorem shows that the ADDA Markov chain is Harris ergodic as long as ϵ is positive, the original DA chain is Harris ergodic, and its transition kernel satisfies a mild absolute continuity condition.

Theorem 2.1. Let K_{DA} be the transition kernel of the parent DA chain which is equivalent to the ADDA chain with $r = 1$, and $\Pi(\cdot \mid \mathcal{D}_{obs})$ be the joint posterior distribution of $\mathcal{D}$ and θ. Assume that $K_{DA}((\mathcal{D}, \theta), \cdot)$ is absolutely continuous with respect to Π for every $(\mathcal{D}, \theta)$. Then, if $\epsilon > 0$ and the parent DA chain is Harris ergodic, the ADDA chain is Harris ergodic.

The proof of the theorem is given in the supplementary materials. The assumption $\epsilon > 0$ is needed because it ensures that there is a positive probability that each worker returns an updated value at each iteration. This allows us to leverage the Harris ergodicity of the parent DA chain to establish relevant properties of the ADDA chain. If $\epsilon = 0$, then assumptions regarding the computer architecture of the workers are required to guarantee Harris ergodicity; for example, see Terenin, Simpson, and Draper (2020). This can get quite messy and tricky to ensure in real-life computing. Using $\epsilon > 0$ is an elegant and clean way of side-stepping these issues. Of course, smaller values of epsilon are preferred for faster computations. Theorem 2.1 is a fundamental step toward understanding theoretical properties of the $\{(\mathcal{D}^{(t)}, \theta^{(t)})\}$ process in ADDA but is still far from the end. We need to establish a CLT for the $\{(\mathcal{D}^{(t)}, \theta^{(t)})\}$ process by proving geometric ergodicity, which is quite challenging in general. For Markov chains where a proof is available, it is typically based on drift and minorization conditions, which are specifically tailored for the particular chain at hand.

In the next section, we illustrate the application of ADDA on logistic regression, high-dimensional variable selection, and linear mixed-effects modeling. For the three examples, the $\{(\mathcal{D}^{(t)}, \theta^{(t)})\}$ process in the parent DA is known to be geometrically ergodic. Extension of each of these proofs to ADDA is difficult for at least one of two main reasons. First, each cycle of ADDA updates (with $1 - \epsilon$ probability) a random r-fraction of the $\mathcal{D}_1, \ldots, \mathcal{D}_k$. This random subset update significantly complicates the transition density in terms of establishing a minorization condition involved in the proof of geometric ergodicity. Second, some proofs for the geometric ergodicity of the parent DA focus on the marginal process $\{\theta^{(t)}\}$, which is a Markov chain in the parent DA setting, and extend the result to the full process $\{(\mathcal{D}^{(t)}, \theta^{(t)})\}$. This strategy fails in for ADDA because the $\{\theta^{(t)}\}$ process is not Markov in general. We provide results establishing geometric ergodicity for the ADDA chains in the three models in the next section. A key idea in the proofs is to identify and establish geometric drift conditions, where the drift functions are unbounded off compact sets. Some of these proofs are established under weaker conditions than those required for the parent DA chains.

3. Applications

3.1. ADDA for Bayesian Logistic Regression

Consider the implementation of Pólya-Gamma DA for Bayesian logistic regression (Polson, Scott, and Windle 2013). Let n be the sample size, $\beta = (\beta_1, \ldots, \beta_p)^{\mathrm{T}} \in \mathbb{R}^p$ be the regression coefficients, and $y_i, s_i, x_i = (x_{i1}, \ldots, x_{ip})^{\mathrm{T}}$ be the response, number of trials, covariates for sample i, respectively, where $y_i \in \{0, 1, \ldots, s_i\}$, $s_i \in \mathbb{N}$, $x_i \in \mathbb{R}^p$. The hierarchical model for logistic regression is

$$y_i \mid \beta \overset{\text{ind.}}{\sim} \text{Binomial}\{s_i, 1/(1 + e^{-\psi_i})\},$$
$$\psi_i = x_i^{\mathrm{T}}\beta, \quad i = 1, \ldots, n, \quad \beta \sim N(\mu_\beta, \Sigma_\beta). \tag{1}$$

The DA algorithm augments the model in (1) with n Pólya-Gamma random variables $\omega_1, \ldots, \omega_n$ specific to the n samples, and cycles between the I and P steps for a large number of iterations starting from a given value of β:

PG-1 (I step) Draw ω_i given β from PG(s_i, $|x_i^{\mathrm{T}}\beta|$) for $i = 1, \ldots, n$, where PG is the Pólya-Gamma distribution.

PG-2 (P step) Draw β given $\omega_1, \ldots, \omega_n$ and $y_1, \ldots, y_n$ from $N(m_\omega, V_\omega)$, where $V_\omega = (X^{\mathrm{T}}\Omega X + \Sigma_\beta^{-1})^{-1}$, $m_\omega =$

$V_\omega(X^{\mathrm{T}}\kappa + \Sigma_\beta^{-1}\mu_\beta)$, $\kappa = (y_1 - s_1/2, \ldots, y_n - s_n/2)^{\mathrm{T}}$, and Ω is the diagonal matrix of ω_is.

The marginal Markov chain $\{\beta^{(t)}\}$ of the β draws collected in step PG-2 has the posterior distribution of β in (1) as its invariant distribution (Choi and Hobert 2013; Wang and Roy 2018b, 2018a).

The ADDA algorithm with Pólya-Gamma DA algorithm as its parent DA modifies step PG-1. Let $s = (s_1, \ldots, s_n)$ and $\omega = \{\omega_1, \ldots, \omega_n\}$. Then, following the notation of Section 2.1, the Pólya-Gamma DA has

$$\theta = \{\beta\}, \quad \mathcal{D}_{\text{obs}} = \{y, s, X\}, \quad \mathcal{D} = \{\omega\},$$
$$\mathcal{D}_{\text{aug}} = \{(y_i, s_i, x_i, \omega_i) : i = 1, \ldots, n\}.$$

For many datasets, the sample size n can be really large. For example, the MovieLens data analyzed in Section 4.3 has 10 million samples. Instead of updating n ω_is in every iteration, ADDA updates only an r fraction of them (with probability $1 - \epsilon$) and reduces to its parent DA when $r = 1$. The ADDA algorithm runs after we randomly split the n samples into k disjoint subsets and store them on k worker processes. Let n_i be the number of samples assigned to worker i, $(y_{j(i)}, s_{j(i)}, x_{j(i)}, \omega_{j(i)})$ be the jth augmented sample on worker i ($j = 1, \ldots, n_i$) and $\mathcal{D}_i = \{\omega_{1(i)}, \ldots, \omega_{n_i(i)}\}$. Note that $\omega_{j(i)}$s in $\mathcal{D}_i$ are rearranged to match the ordering of samples in $\mathcal{D}_{\text{obs}}$ so that $\mathcal{D} = \omega = \{\omega_1, \ldots, \omega_n\}$.

The ADDA extension of Pólya-Gamma DA for a given (k, r) initializes (ω, β) at $(\omega^{(0)}, \beta^{(0)})$ and the AD-I and AD-P steps in the $(t+1)$th iteration are described in Algorithm 1. ADDA generates the Markov chain $\Phi_{\text{ADPG}} = \{\omega^{(t)}, \beta^{(t)}\}_{t=0}^{\infty}$. The following Theorem establishes the geometric ergodicity of Φ_{ADPG} when $\epsilon > 0$ and its proof is in the supplementary material.

Algorithm 1 AD Pólya-Gamma DA

AD-I step

1. Draw $\omega_{j(i)}^{(t+1)}$ given $\beta^{(t)}$ from PG($s_{j(i)}$, $|x_{j(i)}^T \beta^{(t)}|$) on worker i for $j = 1, \ldots, n_i$.
2. Send $\mathcal{D}_i^{(t+1)} = \left\{\omega_{1(i)}^{(t+1)}, \ldots, \omega_{n_i(i)}^{(t+1)}\right\}$ to the manager process if these draws are finished before receiving $\beta^{(t+1)}$ ($i = 1, \ldots, k$).

AD-P step

1. With probability $1 - \epsilon$,
   - wait to receive updated $\mathcal{D}_i^{(t+1)}$ values from an r-fraction of workers, set $\mathcal{D}_i^{(t+1)} = \mathcal{D}_i^{(t)}$ for the remaining $(1 - r)$-fraction of workers, and define $\Omega^{(t+1)} = \text{diag}(\omega^{(t+1)})$, $V_{\omega^{(t+1)}} = (X^\top \Omega^{(t+1)} X + \Sigma_\beta^{-1})^{-1}$, $m_{\omega^{(t+1)}} = V_{\omega^{(t+1)}}(X^T\kappa + \Sigma_\beta^{-1}\mu_\beta)$; and
   - draw $\beta^{(t+1)}$ given $\omega^{(t+1)}, \mathcal{D}_{\text{obs}}$ from $N(m_{\omega^{(t+1)}}, V_{\omega^{(t+1)}})$.
2. With probability ϵ, wait to receive updated $\mathcal{D}_i^{(t+1)}$ values from all the workers, and then proceed to drawing $\beta^{(t+1)}$ given $\omega^{(t+1)}$ and $\mathcal{D}_{\text{obs}}$.

Theorem 3.1. If $\epsilon > 0$, the Markov chain Φ_{ADPG} in Algorithm 1 is geometrically ergodic.

The convergence analysis of the parent PG DA chain has been performed in the literature. In particular, Choi and Hobert (2013) show that the parent PG DA chain is uniformly ergodic (using the marginal $\beta^{(t)}$ chain) if proper priors are used. Following this work, Wang and Roy (2018b) show the geometric ergodicity of the PG DA chain with flat priors. The uniform ergodicity proof for the parent DA chain in Choi and Hobert (2013) for the binary and proper prior setting is based on a minorization argument on the marginal $\{\beta^{(t)}\}$ chain; however, this proof strategy fails for ADDA because the $\{\beta^{(t)}\}$ process for Φ_{ADPG} is not Markov. As a result, we take a different approach where we establish a geometric drift condition using a function of (β, ω), which is unbounded off compact sets. Wang and Roy (2018b) follow a similar strategy for the proving geometric ergodicity of the parent DA chain with improper priors, but they focus on the marginal ω chain instead of the (β, ω) chain. Furthermore, the previous two works consider the parent DA chain when the response is binary, whereas Theorem 3.1 deals with the ADDA chain in a more general binomial setting.

3.2. ADDA for Bayesian High Dimensional Variable Selection

Consider high dimensional variable selection using a Bayesian lasso shrinkage prior Rajaratnam et al. (2019). Let n be the sample size, $\beta = (\beta_1, \ldots, \beta_p)$ be the regression coefficients, $y = (y_1, \ldots, y_n) \in \mathbb{R}^n$ be the response vector, and $X \in \mathbb{R}^{n \times p}$ be the design matrix. The Bayesian lasso DA algorithm augments dimension j with local shrinkage parameter τ_j ($j = 1, \ldots, p$), and the hierarchical model for variable selection is

$$\begin{aligned} &y \mid \beta, \sigma^2 \sim N(X\beta, \sigma^2 I_n), \quad \beta \mid \sigma^2, \tau \sim N(0, \sigma^2 D_\tau), \\ &\tau = (\tau_1, \ldots, \tau_p), \; D_\tau = \text{diag}(\tau), \\ &\sigma^2 \sim \text{Inverse-Gamma}(\alpha, b), \quad \tau_j \overset{\text{ind.}}{\sim} \text{Exponential}(\lambda^2/2), \\ &j = 1, \ldots, p, \end{aligned} \tag{2}$$

where I_n is an $n \times n$ identity matrix, $\sigma^2 > 0$ is the variance of idiosyncratic error term, $\tau = (\tau_1, \ldots, \tau_p)$, and α, b, λ are hyperparameters. The DA algorithm starts with some initial values of β and σ^2 and cycles between the I and P steps for a large number of iterations:

BL-1 (I step) Draw τ_j given β, σ^2 from Inverse-Gaussian $(\frac{|\lambda||\sigma|}{|\beta_j|}, \lambda^2)$ for $j = 1, \ldots, p$.

BL-2 (P step) Draw (β, σ^2) given τ and y as

$$\begin{aligned} \sigma^2 \mid \tau, y &\sim \text{Inverse-Gamma}(n/2 + \alpha, y^\top \\ &\quad (I - XA_\tau^{-1}X^\top)y/2 + b), \quad A_\tau = X^\top X + D_\tau^{-1}, \\ \beta \mid \sigma^2, \tau, y &\sim N\left(A_\tau^{-1}X^\top y, \sigma^2 A_\tau^{-1}\right). \end{aligned} \tag{3}$$

The Markov chain $\{\beta^{(t)}, \sigma^{2(t)}\}$ of the (β, σ^2) draws collected in step BL-2 has the posterior distribution of (β, σ^2) in (2) as its invariant distribution (Rajaratnam et al. 2019).

This DA is different from Pólya-Gamma DA in that the I step in BL-1 draws p latent variables instead of n. In high-dimensional problems, $n \ll p$ and updating $\tau_1, \ldots, \tau_p$ in every

iteration is time consuming even when n is small. Following Section 2.1,

$$\theta = \{\beta, \sigma^2\}, \quad \mathcal{D}_{\text{obs}} = \{y, X\}, \quad \mathcal{D} = \{\tau\},$$
$$\mathcal{D}_{\text{aug}} = \{(y_i, x_{ij}, \tau_j) : i = 1, \ldots, n; j = 1, \ldots p\}$$

in the Bayesian lasso DA. Unlike Pólya-Gamma DA, the computational bottleneck in the I step happens when p is large. The ADDA algorithm starts by partitioning $\{1, \ldots, p\}$ into k disjoint subsets denoted as $\mathcal{P}_1, \ldots, \mathcal{P}_k$. The elements of τ and β are stored on k worker processes. Let $p_i = |\mathcal{P}_i|$, $((\beta_{j(i)}, \tau_{j(i)}) : j \in \mathcal{P}_i)$ be the β and τ elements on worker i, and $\mathcal{D}_i = \{\tau_{1(i)}, \ldots, \tau_{p_i(i)}\}$ $(i = 1, \ldots, k)$. The disjoint partitioning ensures that $p_1 + \cdots + p_k = p$. Note that the $\tau_{j(i)}$'s are rearranged so that $\mathcal{D} = \tau = \{\tau_1, \ldots, \tau_p\}$

The ADDA extension of Bayesian lasso DA, at any iteration, updates (with probability $1-\epsilon$) only an r fraction of $\mathcal{D}_1, \ldots, \mathcal{D}_k$, and with probability ϵ updates all of $\mathcal{D}_1, \ldots, \mathcal{D}_k$. Clearly, the ADDA extension reduces to the parent DA when $r = 1$. For a given k and r, the ADDA extension of the Bayesian Lasso DA initializes (τ, β, σ^2) at $(\tau^{(0)}, \beta^{(0)}, \sigma^{2(0)})$ and the AD-I and AD-P steps in the $(t+1)$th iteration are given in Algorithm 2.

Similar to the Pólya-Gamma DA extension, this ADDA cycles generate the Markov chain $\Phi_{\text{ADBL}} = \{\tau^{(t)}, \beta^{(t)}, \sigma^{2(t)}\}_{t=0}^{\infty}$. The following theorem establishes the geometric ergodicity of Φ_{ADBL} when $\epsilon > 0$, and the proof is given in the supplementary material.

Theorem 3.2. If $\epsilon > 0$ and $n \geq 3$, the Markov chain Φ_{ADBL} in Algorithm 2 is geometrically ergodic.

Geometric ergodicity of a three block version of the parent DA chain was established in Khare and Hobert (2013), and geometric ergodicity for the parent DA chain described in BL.1 and BL.2 above was established in Rajaratnam et al. (2019). Both these proofs make use of drift *and* minorization. In particular, Rajaratnam et al. (2019) prove results on the marginal (β, σ^2) chain, and leverage them to establish geometric ergodicity of the parent DA chain. We are unable to follow this route because the marginal (β, σ^2) chain is not Markov, and establishing minorization with the asynchronous updates is nontrivial. As such, our proof of Theorem 3.2 is based on establishing a geometric drift condition using a drift function that is unbounded off compact sets.

3.3. ADDA for Linear Mixed-Effects Modeling

Consider the setup for linear mixed-effects models (Van Dyk and Meng 2001). Let n be the number of observations, m be the number of subjects, (y_i, X_i, Z_i) be the response, fixed effects covariates, and random effects covariates, respectively, for subject i, p, and q be the number of fixed and random effects, $\beta \in \mathbb{R}^p$ be the fixed effect, and Σ be a $q \times q$ positive definite covariance matrix. Then, the linear mixed-effects model assumes that

$$y_i = X_i\beta + Z_i b_i + e_i, \quad b_i \sim N(0, \Sigma),$$
$$e_i \sim N(0, \sigma^2 I), \quad b_i \perp e_i, \quad i = 1, \ldots, m, \tag{4}$$

where $b_i \in \mathbb{R}^q$ and $e_i \in \mathbb{R}^{n_i}$ are the random effect and idiosyncratic error for subject i, b_i and e_i are mutually independent, Σ is the covariance matrix of random effects, σ^2 is the error variance, I is an identity matrix, $y_i \in \mathbb{R}^{n_i}$, $n = \sum_{i=1}^m n_i$, and the dimensions of X_i, Z_i, e_i, I are chosen appropriately. The b_is are unobserved and the interest lies in inference on $\theta = \{\beta, \Sigma, \sigma^2\}$.

** The DA algorithm for posterior inference on θ is based on modified random effects. Let Γ be a non-singular matrix, $\gamma = \text{vec}(\Gamma)$ stacks the columns of Γ into a q^2-dimensional vector, $\tilde{p} = p + q^2$, $y^{\text{T}} = (y_1^{\text{T}}, \ldots, y_m^{\text{T}}) \in \mathbb{R}^n$, $\tilde{\Sigma} = \Gamma^{-1}\Sigma\Gamma^{-\top}$, $\alpha = (\beta, \gamma)$, and $\tilde{X} \in \mathbb{R}^{n \times \tilde{p}}$ be the matrix with $[X_i \ (d_i^{\text{T}} \otimes Z_i)]$ as its ith row block, where $d_i = \Gamma^{-1} b_i$ for $i = 1, \ldots, m$. The prior

Algorithm 2 AD Bayesian Lasso DA

AD-I step

1. Draw $\tau_{j(i)}^{(t+1)}$ given $\beta^{(t)}, \sigma^{2(t)}$ from Inverse-Gaussian$(|\lambda||\sigma^{(t)}|/|\beta_{j(i)}^{(t)}|, \lambda^2)$ on worker i for $j = 1, \ldots, p_i$.
2. Send $\mathcal{D}_i^{(t+1)} = \left\{\tau_{1(i)}^{(t+1)}, \ldots, \tau_{p_i(i)}^{(t+1)}\right\}$ to the manager process if these draws are finished before receiving $\beta^{(t+1)}, \sigma^{2(t+1)}$ $(i = 1, \ldots, k)$.

AD-P step

1. With probability $1 - \epsilon$,
   - wait to receive updated $\mathcal{D}_i^{(t+1)}$ values from an r-fraction of workers, set $\mathcal{D}_i^{(t+1)} = \mathcal{D}_i^{(t)}$ for the remaining $(1-r)$-fraction of workers, and define $D_{\tau^{(t+1)}} = \text{diag}(\tau^{(t+1)})$, $A_{\tau^{(t+1)}} = X^\top X + D_{\tau^{(t+1)}}^{-1}$; and
   - draw $(\beta^{(t+1)}, \sigma^{2(t+1)})$ given $\mathcal{D}_1^{(t+1)}, \ldots, \mathcal{D}_k^{(t+1)}$ and $\mathcal{D}_{\text{obs}}$ as

$$\sigma^{2(t+1)} \mid \tau^{(t+1)}, y \sim \text{Inverse-Gamma}(n/2 + \alpha, y^\top (I - XA_{\tau^{(t+1)}}^{-1} X^\top) y/2 + b),$$
$$\beta^{(t+1)} \mid \sigma^{2(t+1)}, \tau^{(t+1)}, y \sim N(A_{\tau^{(t+1)}}^{-1} X^\top y, \sigma^{2(t+1)} A_{\tau^{(t+1)}}^{-1}).$$

2. With probability ϵ, wait to receive updated $\mathcal{D}_i^{(t+1)}$ values from all the workers, and then proceed to drawing $\beta^{(t+1)}, \sigma^{2(t+1)}$ given $\mathcal{D}_1^{(t+1)}, \ldots, \mathcal{D}_k^{(t+1)}$ and $\mathcal{D}_{\text{obs}}$.

distributions on (σ^2, α) and $\tilde{\Sigma}$ are assigned as

$$\sigma^2 \sim \frac{M}{\chi^2_a}, \quad \alpha \mid \sigma^2 \sim N(0, \sigma^2 V_\alpha^{-1}),$$
$$\tilde{\Sigma} \sim \text{IW}(W, s), \quad M > 0,\ a > 0,\ s > q+1, \tag{5}$$

where $V_\alpha \succ 0, W \succ 0, \succ 0$ denotes a symmetric positive definite matrix, and IW stands for inverse Wishart distribution. Then, the DA algorithm for posterior inference in linear mixed-effects modeling starts with some given value of θ and repeats the following I and P steps for a large number of iterations:

LM-1 (I step) Draw d_i given θ and y_i from Normal(m_{d_i}, V_{d_i}) for $i = 1, \ldots, m$, where

$$m_{d_i} = \tilde{\Sigma}\Gamma^{\text{T}} Z_i^{\text{T}} \left(Z_i \Gamma \tilde{\Sigma} \Gamma^{\text{T}} Z_i^{\text{T}} + \sigma^2 I\right)^{-1} (y_i - X_i \beta),$$
$$V_{d_i} = \tilde{\Sigma} - \tilde{\Sigma}\Gamma^{\text{T}} Z_i^{\text{T}} \left(Z_i \Gamma \tilde{\Sigma} \Gamma^{\text{T}} Z_i^{\text{T}} + \sigma^2 I\right)^{-1} Z_i \Gamma \tilde{\Sigma}. \tag{6}$$

LM-2 (P step) Draw Σ, σ^2 and $\alpha = (\beta, \gamma)$ given $d_1, \ldots, d_m$, $y_1, \ldots, y_m$ in three steps:

$$\tilde{\Sigma}^{-1} \sim \text{Wishart}_{m+s-q} \left\{ \left(\sum_{i=1}^m d_i d_i^{\text{T}} + W \right)^{-1} \right\},$$
$$\sigma^2 \sim \frac{\|y - \hat{y}\|_2^2 + M}{\chi^2_{n+a-p-q^2}},$$
$$\alpha \mid \sigma^2 \sim N\{\hat{\alpha}, \sigma^2 (\tilde{X}^{\text{T}} \tilde{X} + V_\alpha)^{-1}\}, \tag{7}$$

where $\hat{\alpha} = (\tilde{X}^{\text{T}} \tilde{X} + V_\alpha)^{-1} \tilde{X}^{\text{T}} y$, $\hat{y} = \tilde{X}\hat{\alpha}$, $\Gamma = \text{unvec}(\gamma)$, and $\Sigma = \Gamma \tilde{\Sigma}^{-1} \Gamma^{\text{T}}$.

This DA is different from the previous two DA algorithms in Sections 3.1 and 3.2 due to the presence of random effects. Following Section 2.1, we have

$$\theta = \{\beta, \Gamma, \Sigma, \sigma^2\}, \quad \mathcal{D}_{\text{obs}} = \{y, \tilde{X}\}, \quad \mathcal{D} = \{d_1, \ldots, d_m\},$$
$$\mathcal{D}_{\text{aug}} = \{(y_i, X_i, Z_i, d_i) : i = 1, \ldots, m\},$$

The main computational bottleneck here is the repeated sampling of $d_1, \ldots, d_m$ in the I step LM-1 when m is large. For example, the MovieLens data analyzed in Section 4.3 has $m \approx$ 72,000. The ADDA-based extension of this algorithm starts by partitioning $\{1, \ldots, m\}$ into k disjoint subsets denoted as $\mathcal{P}_1, \ldots, \mathcal{P}_k$. Let $m_i = |\mathcal{P}_i|$, $\{(y_{j(i)}, X_{j(i)}, Z_{j(i)}, d_{j(i)}) : j \in \mathcal{P}_i\}$ be the augmented sample on worker i, and $\tilde{X}_{j(i)} = [X_{j(i)}\ (d_{j(i)}^{\text{T}} \otimes Z_{j(i)})]$ $(j = 1, \ldots, m_i)$. Note that $m_1 + \cdots + m_k = m$ due to the disjoint partitioning. Define the statistics $S_{dd(i)}, S_{xx(i)}, S_{xy(i)}$ for worker i as

$$S_{dd(i)} = \sum_{j=1}^{m_i} d_{j(i)} d_{j(i)}^{\text{T}}, \quad S_{xx(i)} = \sum_{j=1}^{m_i} \tilde{X}_{j(i)}^{\text{T}} \tilde{X}_{j(i)},$$
$$S_{xy(i)} = \sum_{j=1}^{m_i} \tilde{X}_{j(i)}^{\text{T}} y_{j(i)}. \tag{8}$$

Then, the augmented data sufficient statistics $\{\tilde{X}_{1(i)}, \ldots, \tilde{X}_{m_i(i)}, S_{dd(i)}, S_{xx(i)}, S_{xy(i)}\}$ for worker i play an important role in both the DA and ADDA algorithms. In particular, $\hat{\alpha}$ and $\hat{y}$ in P step of the parent DA in (7) become

$$\hat{\alpha} = (S_{XX} + V_\alpha)^{-1} S_{XY}, \quad \hat{y} = \tilde{X}\hat{\alpha}, \quad S_{XX} = \sum_{i=1}^k S_{xx(i)},$$
$$S_{XY} = \sum_{i=1}^k S_{xy(i)}, \tag{9}$$

and $S_{DD}^{-1} = \left(\sum_{i=1}^k S_{dd(i)} + W\right)^{-1}$ is the scale parameter of the Wishart density in (7). Note that $\tilde{X}$ is obtained by arranging $\tilde{X}_{1(i)}, \ldots, \tilde{X}_{m_i(i)}$ $(i = 1, \ldots, k)$ in the order of samples in $\tilde{X}$ and binding them along the rows.

The ADDA extension of the linear mixed effects DA, at any iteration, updates (with probability $1 - \epsilon$) only an r fraction of $\mathcal{D}_1, \ldots, \mathcal{D}_k$, and with probability ϵ updates all of $\mathcal{D}_1, \ldots, \mathcal{D}_k$. Clearly, the ADDA extension reduces to the parent DA in steps LM-1–LM-2 when $r = 1$. For a given k and r, the ADDA extension of the marginal DA initializes $(d_1, d_2, \cdots, d_m, \alpha, \Sigma, \sigma^2)$ at $(d_1^{(0)}, \ldots, d_m^{(0)}, \alpha^{(0)}, \Sigma^{(0)}, \sigma^{2(0)})$ and the AD-I and AD-P steps in the $(t+1)$th iteration are presented in Algorithm 3.

Denote the Markov chain obtained from this ADDA algorithm as $\Phi_{\text{ADLME}} = \{d_1^{(t)}, \ldots, d_m^{(t)}, \alpha^{(t)}, \Sigma^{(t)}, \sigma^{2(t)}\}_{t=0}^\infty$. The convergence analysis of the linear mixed-effect model parent DA chain is extremely challenging. In fact, the literature on geometric ergodicity of the parent DA chain described in LM-1 and LM-2, to the best of our knowledge, focuses only on the special case when the $\Sigma^{(t)}$ is diagonal and Γ is fixed to be the identity matrix; for example, see Román and Hobert (2015) and Abrahamsen and Hobert (2019). Following Román and Hobert (2015) and Abrahamsen and Hobert (2019), we assume that $\Gamma = I_q$ for the convergence analysis; however, unlike these analyses of the parent DA chain, we do *not* restrict Σ to be a diagonal matrix. Algorithm 3, without the Γ updates (i.e., Γ is fixed at I_p), provides a Markov chain $\{d_1^{(t)}, \ldots, d_m^{(t)}, \beta^{(t)}, \Sigma^{(t)}, \sigma^{2(t)}\}_{t=0}^\infty$. In this setting, from the priors used in (5), only the prior for α given σ^2 is modified to the prior for β given σ^2 as $N(0, \sigma^2 V_\beta^{-1})$ for some $V_\beta \succ 0$. We need the following assumptions for our geometric ergodicity result.

Assumption 3.1.

(a) $Z_i^{\text{T}} Z_i \succ 0$ $(i = 1, \ldots, m)$ and $X^{\text{T}} X \succ 0$, where $X = \left[X_1^{\text{T}}\ X_2^{\text{T}}\ \cdots\ X_m^{\text{T}}\right]^{\text{T}}$.

(b) The prior shape parameter s for Σ satisfies $s - q - 1 > \frac{(1-\epsilon)m}{\epsilon}$.

(c) The matrix V_β from the prior distribution on β, and the degrees of freedom a from the prior distribution on σ^2 satisfy $\frac{(p+mq)}{n+a-p-2}(I_p - H) \prec \epsilon I_p - H$, where $H = X(V_\beta + X^{\text{T}} X)^{-1} X^{\text{T}}$ and $A_1 \prec A_2$ denotes that $A_2 - A_1 \succ 0$.

Assumption (a) is quite reasonable, especially since the ADDA algorithm is geared toward applications with large m, n and typically small values of p and q (such as the MovieLens data analyzed in Section 4.3.3). Assumptions (b) and (c) on the prior hyper-parameters s, a, V_β might seem restrictive, but it is important to keep in mind that we are in a challenging setting with a general non-diagonal Σ for which convergence results

Algorithm 3 AD linear mixed-effects model DA

AD-I step

1. Draw $d_{j(i)}^{(t+1)}$ given $\theta^{(t)} = (\alpha^{(t)}, \Sigma^{(t)}, \sigma^{2(t)})$ and $y_{j(i)}$ from $N(m_{d_{j(i)}}, V_{d_{j(i)}})$ on worker i for $j = 1, \ldots, m_i$, where $m_{d_{j(i)}}, V_{d_{j(i)}}$ are obtained by replacing (y_i, X_i, Z_i) in (6) with $(y_{j(i)}, X_{j(i)}, Z_{j(i)})$.
2. Compute $\mathcal{D}_i^{(t+1)}$ from the sampled $d_{j(i)}^{(t+1)}$ values and send $\mathcal{D}_i^{(t+1)}$ to the manager process if these draws are finished before receiving $\theta^{(t+1)} = (\alpha^{(t+1)}, \Sigma^{(t+1)}, \sigma^{2(t+1)})$ $(i = 1, \ldots, k)$.

AD-P step

1. With probability $1 - \epsilon$,
   - wait to receive updated $\mathcal{D}_i^{(t+1)}$ values from an r-fraction of workers, set $\mathcal{D}_i^{(t+1)} = \mathcal{D}_i^{(t)}$ for the remaining $(1-r)$-fraction of workers, and define $\hat{\alpha}^{(t+1)}, \hat{y}^{(t+1)}, S_{DD}^{(t+1)}, S_{XX}^{(t+1)}$ using (9); and
   - draw $(\alpha^{(t+1)}, \Sigma^{(t+1)}, \sigma^{2(t+1)})$ given $\mathcal{D}_1^{(t+1)}, \ldots, \mathcal{D}_k^{(t+1)}$ and $\mathcal{D}_{\text{obs}}$ as follows:

$$\left(\tilde{\Sigma}^{(t+1)}\right)^{-1} \sim \text{Wishart}_{m-q}\left\{\left(S_{DD}^{(t+1)}\right)^{-1}\right\}, \quad \sigma^{2(t+1)} \sim \frac{\|y - \hat{y}^{(t+1)}\|_2^2 + M}{\chi^2_{n+a-p-q^2}},$$

$$\alpha^{(t+1)} \mid \sigma^{2(t+1)} \sim N(\hat{\alpha}^{(t+1)}, \sigma^{2(t+1)}(S_{XX}^{(t+1)} + V_\alpha)^{-1}), \ \Sigma^{(t+1)} = \Gamma^{(t+1)}\tilde{\Sigma}^{(t+1)}\Gamma^{(t+1)\top}.$$

2. With probability ϵ, wait to receive updated $\mathcal{D}_i^{(t+1)}$ values from all the workers, and then proceed to drawing $\alpha^{(t+1)}, \Sigma^{(t+1)}, \sigma^{2(t+1)}$ given $\mathcal{D}_1^{(t+1)}, \ldots, \mathcal{D}_k^{(t+1)}$ and $\mathcal{D}_{\text{obs}}$.

are unavailable even in the parent DA setting. As discussed previously, the asynchronous updates in the ADDA setting present additional challenges in the analysis; however, we are still able to establish geometric ergodicity in this more general ADDA setting, as described in the result below. The proof is provided in the supplementary materials.

Theorem 3.3. If $\epsilon > 0$ and Assumption 3.1 holds, the Markov chain Φ_{ADLME} in Algorithm 3 (adjusted for $\Gamma = I_q$) is geometrically ergodic.

4. Experiments

4.1. Setup

We evaluate the numerical accuracy and computational efficiency of ADDA algorithms for $k \in \{10, 25, 50\}$ and $r \in \{0.20, 0.40, 0.60, 0.80\}$ using their parents as the benchmarks. Every experiment with a given choice of (n, k, r) is replicated 10 times. Before running an ADDA algorithm, the samples or the parameter dimensions are split into k disjoint subsets that are stored on k worker machines, whereas the parent DA uses the full data stored on a single machine. We set $\epsilon = 0$ in our simulations and $\epsilon = 0, 0.01, 0.10$ in the real data analyses. The $\epsilon = 0$ setting is best for fast computations that are required for multiple simulation replications, but it is not clear if the ADDA chain is Harris ergodic because Theorem 2.1 requires that $\epsilon > 0$. In our simulations for ADDA algorithms with $\epsilon = 0$ that run for 20,000 iterations, we have rarely encountered a situation where a worker never sends its I step results to the manager. Our observations from the real data analyses suggests that one could set ϵ to be 10^{-4} or 10^{-5} to maintain theoretical convergence guarantees while compromising very little on the computational gains.

The (empirical) accuracy of an ADDA algorithm relative to its parent DA algorithm is defined using the total variation distance. Let $\eta = h(\theta) = (\eta_1, \ldots, \eta_c)$ be a function of the underlying parameter θ with c components and $q_{rk}^t(\eta_j)$ and $p^t(\eta_j)$ be the posterior density estimates of η_j after t iterations of ADDA and its parent, respectively. Then, the accuracy metric based on η_j for the ADDA algorithm at the end of t iterations is defined as

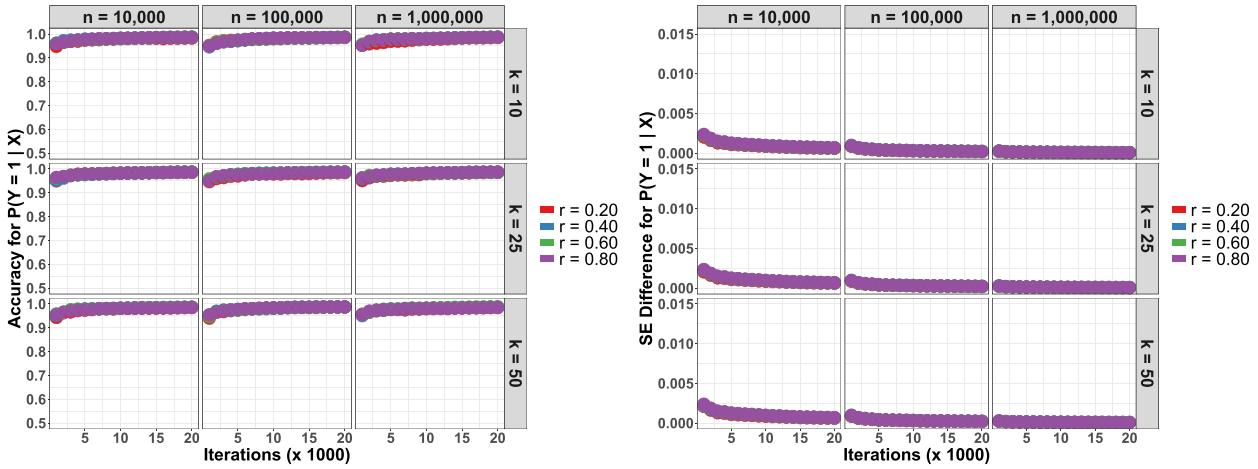
$$\text{Acc}_{rk,j}(t) = 1 - \text{TV}(p^t, q_{rk}^t) = 1 - \tfrac{1}{2}\int_{\mathbb{R}} \left|p^t(\eta_j) - q_{rk}^t(\eta_j)\right| \, d\eta_j, \tag{10}$$

where r and k are given, $\text{TV}(p^t, q_{rk}^t)$ is the total variation distance between p^t and q_{rk}^t, and $\text{Acc}_{rk,j}(t) \in [0, 1]$ because $\text{TV}(p^t, q_{rk}^t) \in [0, 1]$ for every r, k, t. We estimate $\text{Acc}_{rk,j}(t)$ using kernel density estimation in two steps. First, we select the first t draws of η_j from the ADDA algorithm and its parent DA, respectively, to estimate q_{rk}^t and p^t using the *bkde* function in the KernSmooth R package. Second, these estimates are used to numerically approximate the integral in (10) and obtain the $\text{Acc}_{rk,j}(t)$ $(j = 1, \ldots, c)$ estimates. The overall accuracy metric estimate based on η for the ADDA algorithm is defined as

$$\text{Acc}_{rk}(t) = c^{-1}\sum_{j=1}^{c} \text{Acc}_{rk,j}(t). \tag{11}$$

The larger the $\text{Acc}_{rk}(t)$ value, the better an ADDA is at approximating its parent DA.

We use the *mcse* function in the R package mcmcse to compute the Monte Carlo standard errors for ADDA's and its parent's Markov chains based on the overlapping batch means method (Jones et al. 2006; Vats, Flegal, and Jones 2019). Following the notation from the previous paragraph, let $\text{SE}_{rk,j}^{\text{A}}(t)$ and $\text{SE}_j^{\text{P}}(t)$, respectively be outputs of the mcse function using the first t



(a) $\mathrm{Acc}_{rk}(t)$ (b) $\mathrm{SE}_{rk}(t)$

Figure 1. $\mathrm{Acc}_{rk}(t)$ and $\mathrm{SE}_{rk}(t)$ metrics for inference on $\mathrm{P}(Y = 1 \mid x_1 = 1, \ldots, x_p = 1)$ using the ADDA Algorithm 1 for logistic regression with $r = 0.20, 0.40, 0.60, 0.80$ and $k = 10, 25, 50$.

draws of η_j from the ADDA algorithm for a given (r, k) and its parent DA algorithm ($j = 1, \ldots, c$). Then, a metric for comparing their overall Monte Carlo standard errors is defined as

$$\mathrm{SE}_{rk}(t) = c^{-1} \sum_{j=1}^{c} \Delta\mathrm{SE}_{rk,j}(t),$$
$$\Delta\mathrm{SE}_{rk,j}(t) = |\mathrm{SE}^{\mathrm{A}}_{rk,j}(t) - \mathrm{SE}^{\mathrm{P}}_{j}(t)|. \quad (12)$$

The smaller the $\mathrm{SE}_{rk}(t)$ value, the closer are the standard errors of ADDA and its parent.

All ADDA algorithms are implemented in R using the parallel package. The ease of implementation and reproducibility drive this choice. Our ADDA implementation in real data analyses shows three to five times gains in run-times (Figure 4).

4.2. Simulated Data Analyses

The computational bottlenecks in DA for logistic regression and linear mixed-effects models arise due to massive sample sizes (Sections 4.2.1 and 4.2.3). In these models, we partition the samples into k subsets and store them on the worker machines. On the other hand, the augmentation of local shrinkage parameters specific to the regression coefficients leads to the inefficiency of the DA for Bayesian lasso (Section 4.2.2). The augmented parameters here are partitioned into k subsets and stored on the worker machines. The sample or missing data sizes are relatively small in this section to facilitate multiple replications for various choices of (k, r), so we defer run-time comparisons to the real data analyses in Section 4.3. The ADDA algorithm and its parent DA run for 20,000 iterations in this section.

4.2.1. Logistic Regression

We simulate the data using the model in (1). We set $p = 10$, $\beta^{\mathrm{T}} = (-2, 2, \ldots, -2, 2)$, $s_i = 10$ for every i, and generate the entries of X as independent standard normal random variables. Varying $n \in \{10^4, 10^5, 10^6\}$, we simulate $y_1, \ldots, y_n$ using the hierarchical model in (1). The rows of X and y are randomly split into k subsets that are stored on the worker machines.

The $\mathrm{Acc}_{rk}(t)$ and $\mathrm{SE}_{rk}(t)$ metrics are evaluated for $\eta = \mathrm{P}(Y = 1 \mid x_1 = 1, \ldots, x_p = 1)$ (Figure 1). The $\mathrm{Acc}_{rk}(t)$ and $\mathrm{SE}_{rk}(t)$ values are insensitive to the choices of n, r, k. The $\mathrm{Acc}_{rk}(t)$ and $\mathrm{SE}_{rk}(t)$ values quickly increase to 1 and decrease to 0, respectively, as t increases. Our simulations demonstrate that after a sufficiently large number of iterations, the ADDA Algorithm 1 has similar accuracy and Monte Carlo standard errors as its parent DA irrespective of the choice of n, r, and k. In the supplementary material, we provide additional results for $\eta = \beta$ and show that a similar conclusion holds.

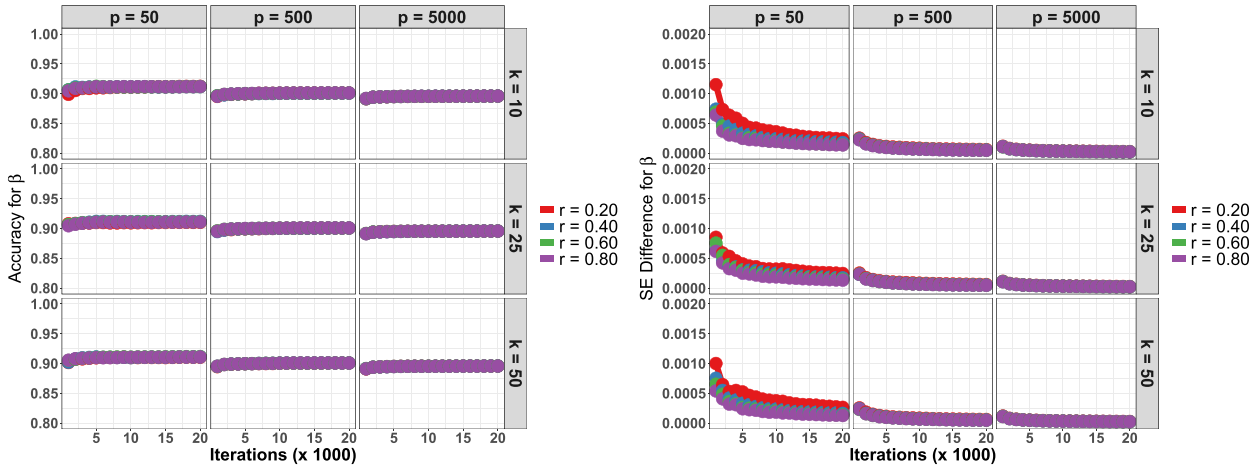
4.2.2. High-Dimensional Variable Selection

We simulate the data using the model in (2). We set the first $0.90p$ entries of β to 0 and the next $0.10p$ entries to -2 and 2 alternatively starting from -2. The entries of X are independent standard normal random variables. Varying n and p such that $n = p$ and $p \in \{50, 500, 5000\}$, we simulate $y_1, \ldots, y_n$ following (2) with $\sigma^2 = 0.01$. The columns of X are randomly split into k subsets that are stored on the worker machines.

For $\eta = \beta$ and every p, r, k, t, the $\mathrm{Acc}_{rk}(t)$ value is high and $\mathrm{SE}_{rk}(t)$ value is low (Figure 2). The $\mathrm{Acc}_{rk}(t)$ values are fairly high even for small t, increase slightly with t, and are insensitive to the choice of k and r. Agreeing with this observation, $\mathrm{SE}_{rk}(t)$ values also decay to 0 as t increases and the decay is insensitive to the choices of p, k, and r. We conclude that ADDA Algorithm 2 is an accurate asynchronous and distributed generalization its parent DA.

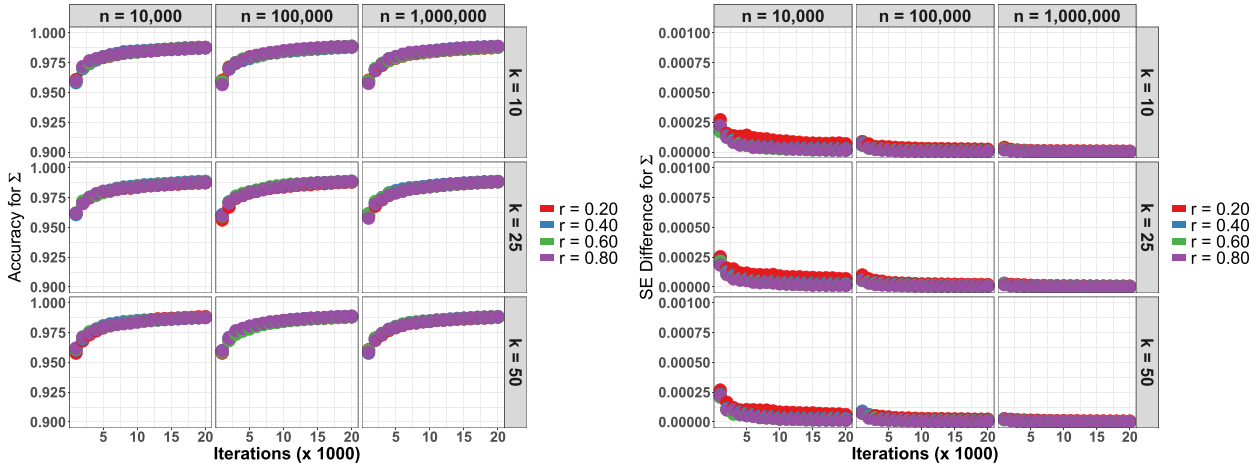
4.2.3. Linear Mixed-Effects Modeling

We evaluate the empirical performance of the ADDA Algorithm 3 for linear mixed-effects modeling with $m \in \{100, 1000, 10,000\}$, $p = 4$, $q = 3$, and $n_i = n/m$ for every i. Following Li, Srivastava, and Dunson (2017), the entries of $\beta = (-2, 2, -2, 2)^{\mathrm{T}}$, the entries of X, Z are set to 1 or -1 with equal probability satisfying Assumption 3.1, $\sigma^2 = 1$, and $\Sigma_{ii} = i$ $(i = 1, 2, 3)$, $\Sigma_{12} = -0.56$, $\Sigma_{31} = 0.52$, $\Sigma_{23} = 0.0025$. Finally, $y_1, \ldots, y_m$ are simulated using the model in (4). The rows of



(a) $\mathrm{Acc}_{rk}(t)$ (b) $\mathrm{SE}_{rk}(t)$

Figure 2. $\mathrm{Acc}_{rk}(t)$ and $\mathrm{SE}_{rk}(t)$ metrics for inference on β in (2) using the ADDA Algorithm 2 for high-dimensional variable selection with $r = 0.20, 0.40, 0.60, 0.80$ and $k = 10, 25, 50$.



(a) $\mathrm{Acc}_{rk}(t)$ (b) $\mathrm{SE}_{rk}(t)$

Figure 3. $\mathrm{Acc}_{rk}(t)$ and $\mathrm{SE}_{rk}(t)$ metrics for inference on Σ in (4) using the ADDA Algorithm 3 for linear mixed-effects modeling with $r = 0.20, 0.40, 0.60, 0.80$ and $k = 10, 25, 50$.

X, Z, and y are split into k disjoint subsets that are stored on the worker machines.

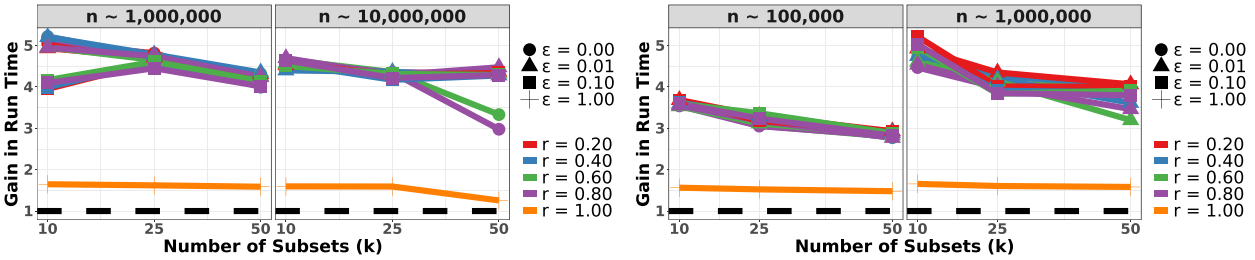
We evaluate the $\mathrm{Acc}_{rk}(t)$ and $\mathrm{SE}_{rk}(t)$ metrics for $\eta = \Sigma$ (Figure 3). The $\mathrm{Acc}_{rk}(t)$ and $\mathrm{SE}_{rk}(t)$ values insensitive to the choices of n, r, k. If t is sufficiently large, then $\mathrm{Acc}_{rk}(t)$ and $\mathrm{SE}_{rk}(t)$ values are close to 1 and 0, respectively. We conclude that the ADDA Algorithm 3 and its parent have similar accuracy and Monte Carlo standard errors in estimating the posterior distribution of Σ. We provide additional results in the supplementary material showing that a similar conclusion holds for $\eta = \beta$ and $\eta = \sigma^2$, respectively.

4.3. Real Data Analysis

4.3.1. MovieLens Data

We use the MovieLens ratings data (*http://grouplens.org*) that contain more than 10 million movie ratings from about 72 thousand users, where each rating ranges from 0.5 to 5 in increments of 0.5. Starting with Perry (2017), a modified form of this data has been used for illustrating scalable inference in linear mixed-effects models (Srivastava, DePalma, and Liu 2019; Srivastava and Xu 2021). Specifically, a movie's category is defined using its genre: *action* category includes action, adventure, fantasy, horror, sci-fi, or thriller genres; *children* category includes animation or children; *drama* category includes crime, documentary, drama, film-noir, musical, mystery, romance, war, or western; and *comedy* category includes comedy genre only.

Three predictors are defined using this data. The first predictor encodes a movie's category using three dummy variables with *action* category as the baseline. If a movie belongs to C categories, then its category predictor is $1/C$ for each of the C categories. The second and third predictors are numeric variables that capture a movie's popularity and a user's mood. The popularity predictor of a movie equals $\mathrm{logit}\{(l + 0.5)/(r + 1)\}$, where r users rated the movie and l of them gave a rating of 4 or higher. The user's mood predictor equals 1 if the user gave the previously 30 rated movies a rating of 4 or higher and 0 otherwise.



(a) Logistic regression (b) Linear mixed-effects model

Figure 4. Gain in run times for the logistic regression and linear mixed-effects models used in MovieLens data analysis for $r = 0.20, 0.40, 0.60, 0.80, 1.00$ and $k = 10, 25, 50$. The distributed (or parallel) implementation of the parent DA corresponds to $r = 1.00$ (or $\epsilon = 1.00$). The gain is defined as the ratio of run times of the parent DA and its ADDA-based extension. The dashed black line indicates equal run times for an ADDA algorithm and its parent.

We model this data using logistic regression and linear mixed-effects model in Sections 4.3.2 and 4.3.3. Before fitting the logistic regression model, the response in MovieLens data is modified for defining the 0/1-valued response. If the user's rating in a given sample is greater than 3, then the response is 1; otherwise, the response is 0. No modification is required for the application of linear mixed-effects model. The original PG DA and marginal DA algorithms for logistic regression and linear mixed-effects modeling, respectively, are slow if we use with the full MovieLens data; therefore, we analyze randomly selected subsets of the MovieLens data to facilitate posterior computations. The conclusions from $\text{Acc}_{rk}(t)$ and $\text{SE}_{rk}(t)$ metrics agree with those in the simulation studies of Sections 4.2.1 and 4.2.3, so we have presented them in the supplementary material. In this section, we only provide run-time comparisons between the ADDA algorithms and their parents (Figure 4).

4.3.2. Logistic Regression

We randomly select two subsets of users in the MovieLens data such that the total number of ratings in them are about 10^6 and 10^7, respectively. The MovieLens data with modified 0/1-valued responses are analyzed using logistic regression model in (1) with six predictors, including an intercept. There are three dummy variables for the movie categories and one each for a movie's popularity and a user's mood. The β vector is six-dimensional. The responses and predictors in the modified data are collected into the response vector y and design matrix X. We apply the ADDA Algorithm 1 used in Section 4.2.1 with two modifications: the number of iterations is 10,000 instead of 20,000 and $\epsilon = 0, 0.01, 0.10$. This setup is replicated 10 times by randomly choosing the users in each replication.

$\text{Acc}_{rk}(t)$ and $\text{SE}_{rk}(t)$ comparisons. The $\text{Acc}_{rk}(t)$ and $\text{SE}_{rk}(t)$ metrics are evaluated for $\eta = \beta$, $\text{P}(Y = 1 \mid x)$, where $x = (0, 0, 0, 0, 1, 0)$ is the predictor for a popular movie. When $\epsilon \neq 0$, the $\text{Acc}_{rk}(t)$ and $\text{SE}_{rk}(t)$ values are insensitive to the choices of n, r, k and converge to 1 and 0, respectively, after $t = 2000$. If $\epsilon = 0$, then the setup of Theorem 3.1 is violated. This results in slower convergence of $\text{Acc}_{rk}(t)$ and $\text{SE}_{rk}(t)$ values when k and r are relatively large and small, respectively ($k = 50$; $r = 0.20, 0.40$); see supplementary material for figures.

Run-time comparisons. The ADDA Algorithm 1 is three to five times faster than its parent DA (Figure 4(a)). The gain in run-times are insensitive to the choices of n, r, k, or ϵ. The minor variations in run-time gains are present due to the varying loads on the cluster. A distributed (or parallel) implementation of the parent DA corresponds to an ADDA with $r = 1.00$ (or $\epsilon = 1.00$). For every choice of n, k, and $\epsilon \in \{0.00, 0.01, 0.10\}$, the asynchronous implementations of ADDA with $r \in \{0.20, 0.40, 0.60, 0.80\}$ are two to three times faster than the distributed implementation of parent DA. An interesting observation is that the gain in runtime going from $r = 1$ to $r = 0.8$ is more significant than going from $r = 0.8$ to $r = 0.2$. One possible reason is that there is a small minority of workers in each iteration who take significantly more time than others. Asynchronous computations allow us to bypass these workers for that particular iteration (with probability $1 - \epsilon$) and speed up the computation.

Conclusions. For a sufficiently large t and every n, r, k, the $\text{Acc}_{rk}(t)$ and $\text{SE}_{rk}(t)$ values for the ADDA chain corresponding to Algorithm 1 are close to 1 and 0, respectively, as long as ϵ is positive. The ADDA chain is also three to five times faster than its parent DA depending on (k, r). The asynchronous computations are advantageous in that ADDA with $r < 1$ are two to three times faster than the distributed implementation of parent DA.

4.3.3. Mixed-Effects Modeling

We randomly select two subsets of users such that the total number of ratings in them are approximately 10^5 and 10^6, respectively, and fit the model in (4). There are six fixed and random effects predictors, including an intercept, three dummy variables for the movie categories, and one each for a movie's popularity and a user's mood. The dimensions of β and Σ are 6×1 and 6×6. We run Algorithm 3 with $\epsilon = 0, 0.01, 0.10$ and its parent for 10,000 iterations. This setup is replicated 10 times by randomly choosing the users.

$\text{Acc}_{rk}(t)$ and $\text{SE}_{rk}(t)$ comparisons. When $\epsilon \neq 0$ and $r \neq 0.2$, the $\text{Acc}_{rk}(t)$ and $\text{SE}_{rk}(t)$ values for β and Σ converge to 1 and 0, respectively, for every n, k after $t = 5000$. For smaller t values, $\text{Acc}_{rk}(t)$ is relatively low for $n = 10^6$ and $r = 0.2$; however, the differences between $\text{Acc}_{rk}(t)$ and $\text{SE}_{rk}(t)$ values, respectively, disappear for a sufficiently large t given $\epsilon \neq 0$. If $\epsilon = 0$, then an assumption of Theorem 3.3 is violated that results in slower convergence of $\text{Acc}_{rk}(t)$ and $\text{SE}_{rk}(t)$ compared to the previous cases, especially when $r \in \{0.20, 0.40\}$; see supplementary material for figures. Therefore, we recommend choosing a small but positive ϵ in practice.

Run-time comparisons. The ADDA Algorithm 3 is three to five times faster than its parent DA for all the choices of n, r, k, and ϵ (Figure 4(b)). For a given k, the gain in run-time increases with n for every ϵ and r. On the other hand, for a given n, the gain in run-times decreases with increasing k for every ϵ and r due to the increased communication among manager and worker processes. Similar to our observations in logistic regression, a distributed (or parallel) implementation of the parent DA is two to three times slower than the asynchronous implementations of ADDA with $r < 1$ and $\epsilon < 1$ for every choice of n, k, and ϵ. Varying loads on the cluster have a minimal impact on the run-times.

Conclusions. If ϵ is positive and t is sufficiently large, then the $\text{Acc}_{rk}(t)$ and $\text{SE}_{rk}(t)$ values of Algorithm 3 are close to 1 and 0, respectively, for every n, r, k. The ADDA chain corresponding to Algorithm 3 is also three to five times faster than its parent DA depending on (k, r). The asynchronous computations are more efficient than parallel computations in that ADDA with $r < 1$ is faster than the distributed implementation of parent DA.

5. Discussion

Our schemes and theoretical results can be extended to other models. For example, assigning a multivariate t distribution to the random effects of the linear mixed-effects model in Section 3.3 motivates robust extensions of Algorithm 3 (Pinheiro, Liu, and Wu 2001). On a related note, Pólya-Gamma data augmentation algorithm converges extremely slowly to its stationary distribution in massive data settings where the number of 1s and 0s are unbalanced (Johndrow et al. 2019). Our simulated and real data analyses have focused on the balanced case, so we have not encountered this problem. It is interesting to explore if our scheme with nonrandom partitions is able to bypass this problem.

Second, our real data analysis illustrates the dependence of accuracy and Monte Carlo standard errors on the choice of (k, r, ϵ). Due to varying loads on worker processes in a cluster, some workers return their I step results to the manager more often than others. Our empirical results suggest this variability decreases with n. For example, if we define the empirical estimate of r for a worker as the fraction of the number of times the manager accepts its I step results over the total number of iterations, then empirical estimates of r are closer to the true r as n increases for every worker and k choices; see supplementary material for the figure. Additionally, the Bernstein-von Mises theorem implies that the posterior distribution is accurately approximated as a Gaussian for a large n. We are exploring techniques for balancing the Monte Carlo and statistical errors that depend on (t, k, r, ϵ) and n, respectively, for choosing the total number of Monte Carlo iterations to achieve a desired accuracy.

Our ADDA extensions exploit two features of the augmented data model. First, the k augmented data subsets are mutually independent given the original parameter and the observed data. Second, the conditional posterior distribution of the missing data on any subset depends only on the local copy of the observed data. Both assumptions are violated in models for time series data (e.g., hidden Markov models). Data augmentation has been widely used for posterior inference and predictions in these models, but it suffers from similar computational bottlenecks in massive data settings that have been highlighted in this article. Recently, this problem has been addressed partly using stochastic gradient descent (Cappé 2011; Le Corff and Fort 2013). As part of future research, we will explore extensions of our scheme to hidden Markov models and Gaussian state-space models.

Supplementary Materials

The supplementary material contains the proofs of Theorems 2.1, and 3.1–3.3 and additional numerical results for the simulated and real data analyses presented in Section 4.

Acknowledgments

Luke Tierney helped the author in implementing asynchronous computations using R.

Funding

The author gratefully acknowledges grants from the Office of Naval Research (ONR-BAA N000141812741) and the National Science Foundation (DMS-1854667/1854662).

ORCID

Sanvesh Srivastava https://orcid.org/0000-0002-5483-9579

References

Abrahamsen, T., and Hobert, J. P. (2019), "Fast Monte Carlo Markov chains for Bayesian shrinkage models with Random Effects," *Journal of Multivariate Analysis*, 169, 61–80. [901]

Atchadé, Y., and Wang, L. (2021), "A Fast Asynchronous MCMC Sampler for Sparse Bayesian Inference," arXiv preprint arXiv:2108.06446. [896]

Backlund, G., Hobert, J. P., Jung, Y. J., and Khare, K. (2021), "A Hybrid Scan Gibbs Sampler for Bayesian Models with Latent Variables," *Statistical Science*, 36, 379–399. [898]

Cappé, O. (2011), "Online EM Algorithm for Hidden Markov Models," *Journal of Computational and Graphical Statistics*, 20, 728–749. [906]

Choi, H. M., and Hobert, J. P. (2013), "The Pólya-Gamma Gibbs Sampler for Bayesian Logistic Regression is Uniformly Ergodic," *Electronic Journal of Statistics*, 7, 2054–2064. [899]

Hobert, J. P. (2011), "The Data Augmentation Algorithm: Theory and Methodology," in *Handbook of Markov Chain Monte Carlo*, pp. 279–320, Boca Raton, FL: Chapman and Hall/CRC. [895]

Johndrow, J. E., Smith, A., Pillai, N., and Dunson, D. B. (2019), "MCMC for Imbalanced Categorical Data," *Journal of the American Statistical Association*, 114, 1394–1403. [906]

Johnson, M. J., Saunderson, J., and Willsky, A. (2013), "Analyzing Hogwild Parallel Gaussian Gibbs Sampling," in *Advances in Neural Information Processing Systems (NeurIPS)* (Vol. 26), eds. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger, Red Hook, NY: Curran. [896]

Jones, G. L., Haran, M., Caffo, B. S., and Neath, R. (2006), "Fixed-Width Output Analysis for MCMC," *Journal of the American Statistical Association*, 101, 1537–1547. [902]

Jones, G. L., and Hobert, J. P. (2001), "Honest Exploration of Intractable Probability Distributions via Markov Chain Monte Carlo," *Statistical Science*, 16, 312–334. [897]

Jordan, M. I., Lee, J. D., and Yang, Y. (2019), "Communication-Efficient Distributed Statistical Inference," *Journal of the American Statistical Association*, 114, 668–681. [895]

Khare, K., and Hobert, J. P. (2013), "Geometric Ergodicity of the Bayesian Lasso," *Electronic Journal of Statistics*, 7, 2150–2163. [900]

Le Corff, S., and Fort, G. (2013), "Online Expectation Maximization based Algorithms for Inference in Hidden Markov Models," *Electronic Journal of Statistics*, 7, 763–792. [906]

Levine, R. A., Yu, Z., Hanley, W. G., and Nitao, J. J. (2005), "Implementing Random Scan Gibbs Samplers," *Computational Statistics*, 20, 177–196. [898]

Li, C., Srivastava, S., and Dunson, D. B. (2017), "Simple, Scalable and Accurate Posterior Interval Estimation," *Biometrika*, 104, 665–680. [895,903]

Nemeth, C., and Fearnhead, P. (2021), "Stochastic Gradient Markov Chain Monte Carlo," *Journal of the American Statistical Association*, 116, 433–450. [895]

Newman, D., Asuncion, A., Smyth, P., and Welling, M. (2009), "Distributed Algorithms for Topic Models," *Journal of Machine Learning Research*, 10, 1801–1828. [896]

Perry, P. O. (2017), "Fast Moment-based Estimation for Hierarchical Models," *Journal of the Royal Statistical Society*, Series B, 79, 267–291. [904]

Pinheiro, J. C., Liu, C., and Wu, Y. N. (2001), "Efficient Algorithms for Robust Estimation in Linear Mixed-Effects Models using the Multivariate t Distribution," *Journal of Computational and Graphical Statistics*, 10, 249–276. [906]

Polson, N. G., Scott, J. G., and Windle, J. (2013), "Bayesian Inference for Logistic Models using Pólya–Gamma Latent Variables," *Journal of the American statistical Association*, 108, 1339–1349. [896,898]

Rajaratnam, B., Sparks, D., Khare, K., and Zhang, L. (2019), "Uncertainty Quantification for Modern High-Dimensional Regression via Scalable Bayesian Methods," *Journal of Computational and Graphical Statistics*, 28, 174–184. [896,899,900]

Robert, C., and Casella, G. (2013), *Monte Carlo Statistical Methods*, New York: Springer. [895]

Román, J. C., and Hobert, J. P. (2015), "Geometric Ergodicity of Gibbs Samplers for Bayesian General Linear Mixed Models with Proper Priors," *Linear Algebra and its Applications*, 473, 54–77. [901]

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. [895]

Srivastava, S., DePalma, G., and Liu, C. (2019), "An Asynchronous Distributed Expectation Maximization Algorithm for Massive Data: The DEM Algorithm," *Journal of Computational and Graphical Statistics*, 28, 233–243. [896,904]

Srivastava, S., Li, C., and Dunson, D. B. (2018), "Scalable Bayes via Barycenter in Wasserstein Space," *Journal of Machine Learning Research*, 19, 312–346. [895]

Srivastava, S., and Xu, Y. (2021), "Distributed Bayesian Inference in Linear Mixed-Effects Models," *Journal of Computational and Graphical Statistics*, 30, 594–611. [904]

Terenin, A., Simpson, D., and Draper, D. (2020), "Asynchronous Gibbs Sampling," in *Proceedings of AISTATS*, Volume 108 of Proceedings of Machine Learning Research, eds. S. Chiappa and R. Calandra, pp. 144–154. [896,898]

Van Dyk, D. A., and Meng, X.-L. (2001), "The Art of Data Augmentation," *Journal of Computational and Graphical Statistics*, 10, 1–50. [896,900]

Vats, D., Flegal, J. M., and Jones, G. L. (2019), "Multivariate Output Analysis for Markov Chain Monte Carlo," *Biometrika*, 106, 321–337. [902]

Wang, C., and Srivastava, S. (2021), "Divide-and-Conquer Bayesian inference in hidden Markov models, arXiv preprint arXiv:2105.14395. [895]

Wang, X., and Roy, V. (2018a), "Analysis of the Pólya-Gamma block Gibbs Sampler for Bayesian Logistic Linear Mixed Models," *Statistics & Probability Letters*, 137, 251–256. [899]

——— (2018b), Geometric Ergodicity of Pólya-Gamma Gibbs Sampler for Bayesian Logistic Regression with a Flat Prior," *Electronic Journal of Statistics*, 12, 3295–3311. [899]

Xue, J., and Liang, F. (2019), "Double-Parallel Monte Carlo for Bayesian Analysis of Big Data," *Statistics and Computing*, 29, 23–32. [895]