A versatile distributed MCMC algorithm for large scale inverse problems

Pierre-Antoine Thouvenin*, Audrey Repetti†‡, and Pierre Chainais*
*Univ. Lille, CNRS, Centrale Lille, UMR 9189 CRIStAL, F-59000 Lille, France
†Department of Actuarial Mathematics & Statistics, Heriot-Watt University, Edinburgh EH14 4AS, UK
‡Institute of Sensors, Signals and Systems, Heriot-Watt University, Edinburgh EH14 4AS, UK

***Abstract*—For large scale inverse problems, inference can be tackled with distributed algorithms, dividing the task over multiple computing nodes or cores referred to as *workers*. Since random sampling methods yield not only estimates but also credibility intervals, we leverage data augmentations and MCMC algorithms to design a distributed sampler. In contrast with usual approaches relying on a client-server architecture, we propose a flexible distributed sampler relying on a Single Program Multiple Data implementation, in which all workers have a similar task. This distributed strategy allows the computing time and volume of communications to be reduced by separately handling blocks of data and parameters on different workers. Experiments on a large synthetic image inpainting problem illustrate the performance of the proposed approach to produce high quality estimates in a small amount of time.**

***Index Terms*—Markov chain Monte-Carlo methods, distributed algorithm, inverse problems, Single Program Multiple Data architecture.**

I. Introduction

An inverse problem consists in inferring unknown latent variables $\boldsymbol{x} \in \mathbb{R}^N$ from a collection of degraded and noisy observations $\boldsymbol{y} \in \mathbb{R}^M$. Observations and parameters are related through an observation model of the form

$$\boldsymbol{y} = \mathcal{D}(\boldsymbol{A}\boldsymbol{x}), \tag{1}$$

where $\boldsymbol{A} \in \mathbb{R}^{M \times N}$ is the measurement operator and $\mathcal{D}\colon \mathbb{R}^M \to \mathbb{R}^M$ models random perturbations affecting the measurements. The first step in Bayesian inference consists in forming the posterior distribution of the problem, combining the information from the likelihood and the prior:

$$\pi(\boldsymbol{x} \mid \boldsymbol{y}) \propto \exp\left(-f_{\boldsymbol{y}}(\boldsymbol{A}\boldsymbol{x}) - g(\boldsymbol{B}\boldsymbol{x})\right), \tag{2}$$

where $f_{\boldsymbol{y}}\colon \mathbb{R}^M \to]-\infty, +\infty]$ is the data-fidelity term related to the statistical model of the observations, and $\boldsymbol{B} \in \mathbb{R}^{P \times N}$ and $g\colon \mathbb{R}^P \to]-\infty, +\infty]$ forms the prior on $\boldsymbol{x}$ (e.g., total variation (TV) norm). The difficulty of solving such problems increases with the dimension of both the unknown variables and the observations. In addition, these methods need to scale to large number of parameters and observations to efficiently handle high dimensional problems. To this end, a usual strategy consists in distributing both parameters and observations among several *workers*.

This work was partly supported by the ANR project "Chaire IA Sherlock" ANR-20-CHIA-0031-01 hold by P. Chainais, as well as by the national support within the *programme d'investissements d'avenir* ANR-16-IDEX-0004 ULNE and Région HDF.

Splitting optimization methods provide a large class of versatile parallelizable and distributed methods for solving (1), by minimizing the negative log-posterior. Among them, one can cite the celebrated *alternating direction methods of multipliers* (ADMM) [1], and more generally primal-dual algorithms [2]–[5]. These approaches allow a *maximum a posteriori* (MAP) estimate to be formed while efficiently exploiting the computing resources available. However, in situations where the ground truth is not available (*e.g.*, in astronomical imaging [6]), these approaches do not enable the uncertainty about the provided estimates to be directly quantified.

Markov chain Monte Carlo (MCMC) are methods of choice to obtain both estimates and associated credibility intervals [7]. MCMC algorithms consists in drawing samples from the posterior distribution (2). Nevertheless, they are often considered computationally too expensive to handle high dimensional problems. Over the last decade, multiple authors have proposed more versatile and scalable optimization-inspired MCMC methods [8]–[11].

To the best of the authors' knowledge, only a few distributed samplers have been proposed in the literature [12]–[14]. The so-called *asymptotically exact data augmentation* (AXDA) approach [14] provides a framework to scale an MCMC algorithm to larger dimensions *via* a variable splitting strategy reminiscent of the ADMM (*divide to conquer strategy*). So far, AXDA has exclusively been used to design distributed algorithms relying on a client-server architecture, observing that splitting variables are conditionally independent. A client-server setting [12] is however limited in terms of distribution flexibility, and may induce communication bottlenecks as all workers (*clients*) need to communicate with the server.

In this work, we consider a class of inverse problems for which couplings between latent parameters are *localized*, *i.e.*, $\boldsymbol{A}$ and $\boldsymbol{B}$ in (2) are block-sparse, see Fig. 1 in Section II. This is the case in applications such as image deconvolution or inpainting. Such a specific structure is amenable to flexible distributed strategies. Specifically, we introduce a versatile MCMC algorithm leveraging the AXDA strategy. We propose a Single Program Multiple Data (SPMD) [15] implementation of the resulting distributed sampling method. Compared to a client-server architecture, each worker involved in an SPMD architecture is assigned a similar task, and communications occur between a small number of neighbouring workers. For the considered problem, a larger number of workers can be

involved in the sampling task, while reducing the volume of the communications.

The rest of the paper is organized as follows. The model considered in this work is introduced in Section II. To simplify the presentation, the proposed application of the AXDA framework is described in the case of an inpainting problem in Sections III-A and III-B, leading to an original distributed MCMC algorithm detailed in Sections III-C and III-D. Section IV illustrates the performance of the proposed approach on an image inpainting problem under a total variation prior, in comparison with the sampler proposed in [13]. Conclusions and perspectives are summarized in Section V.

II. Problem description

A. Block-sparse model description

In this work, we consider an inverse problem with corresponding posterior distribution (2), where the matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ have a block-sparse structure (*i.e.*, induce localized couplings between parameters). We further assume that both the number of observations M and parameters N are of the same order. We thus aim to distribute both the data and the parameters over multiple workers. Let $K \in \mathbb{N}^*$ be the number of selected workers, with $K \leq \min\{M, N, P\}$. Let the columns of $\boldsymbol{A}$ and $\boldsymbol{B}$ be split over the K workers. We consider a compact overlap between workers. Precisely, we assume that the number of rows with nonzero elements affecting multiple parameter blocks is small compared to $\min\{\lfloor M/K \rfloor, \lfloor P/K \rfloor\}$. In addition, these rows are assumed to have a small number of nonzero overlapping elements compared to $\lfloor N/K \rfloor$. An example of such matrix structure and splitting is illustrated in Fig. 1.

Most of the classical operators in image processing satisfy these conditions, *e.g.*, convolution, compressive sensing and inpainting operators [16, Sections 3.4 and 13.3].

In the remainder, we consider inverse problems with this structure. The noise is further assumed not to induce additional correlations. This ensures that statistically independent blocks of contiguous observations can be formed.

B. Observation model

According to the model structure introduced in Section II-A, we assume that the observations $\boldsymbol{y}$ can be partitioned into K statistically independent blocks $\boldsymbol{y} = (\boldsymbol{y}_k)_{1\leq k\leq K}$, where for every $k \in \{1, \dots, K\}$, $\boldsymbol{y}_k \in \mathbb{R}^{M_k}$, and $M = \sum_k M_k$. Thus, the operator $\mathcal{D}$ is separable as $\boldsymbol{z} = (\boldsymbol{z}_k)_{1\leq k\leq K} \in \mathbb{R}^M \mapsto \mathcal{D}(\boldsymbol{z}) = (\mathcal{D}_k(\boldsymbol{z}_k))_{1\leq k\leq K}$. Then, for every $k \in \{1, \dots, K\}$,

$$\boldsymbol{y}_k = \mathcal{D}_k(\boldsymbol{A}_k \boldsymbol{C}_k \boldsymbol{x}), \tag{3}$$

where $\boldsymbol{C}_k \in \mathbb{R}^{\widetilde{N}_k \times N}$ selects the $\widetilde{N}_k$ contiguous entries from $\boldsymbol{x}$ required to form $\boldsymbol{y}_k$, and $\boldsymbol{A}_k \in \mathbb{R}^{M_k \times \widetilde{N}_k}$ with $N \leq \sum_k \widetilde{N}_k$[1]. The matrices $\boldsymbol{A}_k$ and $\boldsymbol{C}_k$ are illustrated in Fig. 1.

[1]When $N = \sum_k \widetilde{N}_k$, the matrix $\boldsymbol{A}$ is block diagonal (up to a row permutation), and there is no coupling between consecutive parameter blocks.

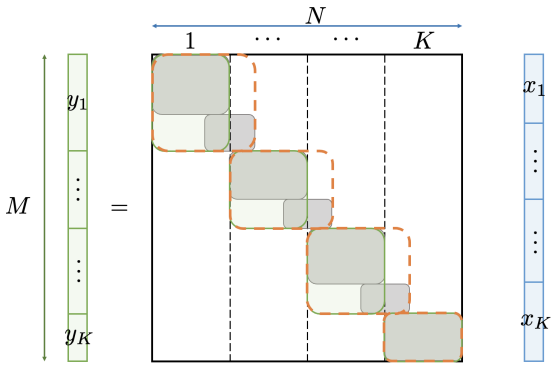

Fig. 1. Example of the block-sparse structure of the matrix $\boldsymbol{A}$ involved in (3). The columns of $\boldsymbol{A}$ are split into contiguous blocks (dashed lines), with a small overlap compared to $\lfloor N/K \rfloor$. Each matrix $\boldsymbol{A}_k$ from (3) is in dashed orange lines, and $\boldsymbol{C}_k$ selects the corresponding entries in $\boldsymbol{x}$ to form $\boldsymbol{y}_k$.

Assuming that $\mathcal{D}$ is block separable is equivalent to assuming that the function $f_{\boldsymbol{y}}$ in (2) is additively separable as

$$(\forall \boldsymbol{x} \in \mathbb{R}^N)(\forall \boldsymbol{y} \in \mathbb{R}^M) \quad f_{\boldsymbol{y}}(\boldsymbol{A}\boldsymbol{x}) = \sum_{k=1}^{K} f_{\boldsymbol{y}_k}(\boldsymbol{A}_k \boldsymbol{C}_k \boldsymbol{x}), \tag{4}$$

where, for every $k \in \{1, \dots, K\}$, $f_k \colon \mathbb{R}^{M_k} \to]-\infty, +\infty]$.

In this setting, each worker k can handle both a data block $\boldsymbol{y}_k$ and the corresponding parameter block $\boldsymbol{x}_k$, the latter gathering most of the information contained in $\boldsymbol{C}_k \boldsymbol{x}$.

The assumption on the compact overlap, described in Section II-A, illustrated in Fig. 1, ensures that for a worker $k \in \{1, \dots, K\}$, the elements of $\boldsymbol{C}_k \boldsymbol{x}$ not stored on the worker k can be collected by exchanging a small number of parameters with the neighbouring workers $\{k-1, k+1\}$.

Note that (3) covers multiple noise models, including additive Gaussian noise with a (block) diagonal covariance matrix, Poisson noise or multiplicative noise.

C. Prior model

Multiple classical priors involve linear operators with structure described in Section II-A. In our setting, we further assume that g is block additively separable, similarly to f:

$$(\forall \boldsymbol{x} \in \mathbb{R}^N) \quad g(\boldsymbol{B}\boldsymbol{x}) = \sum_{k=1}^{K} g_k(\boldsymbol{B}_k \boldsymbol{D}_k \boldsymbol{x}), \tag{5}$$

where, for every $k \in \{1, \dots, K\}$, $g_k \colon \mathbb{R}^{P_k} \to]-\infty, +\infty]$, $\boldsymbol{B}_k \in \mathbb{R}^{P_k \times \overline{N}_k}$, and $\boldsymbol{D}_k \in \mathbb{R}^{\overline{N}_k \times N}$ is a selection operator. Classical examples of this form include priors promoting spatial correlations such as the TV [5], [17], sparsity in a wavelet dictionary domain [18], or separable constraints such as the non-negativity prior. Note that $\boldsymbol{D}_k$ is analogous to $\boldsymbol{C}_k$ in (3), but can select a different neighbourhood of $\boldsymbol{x}_k$, dictated by the structure of $\boldsymbol{B}$.

III. Proposed distributed method

Considering the model structure (3)–(5), the proposed approach leverages the AXDA framework [12], [14]. The underlying splitting strategy allows standard transition kernels to be more easily applied (such as the Gibbs sampler [7]). It also paves the way to distributed algorithms [12].

A. Inpainting problem

For the sake of simplicity, we introduce the proposed distributed algorithm for the particular inpainting inverse problem considered for the experiments in Section IV. We also focus on the particular case when the noise in (3) is an additive white Gaussian noise. In this case, the matrix $\boldsymbol{A}$ is a selection operator and the splitting strategy is only applied to the prior term[2]. Hence, (3) and (4) respectively reduce to

$$(\forall k \in \{1,\dots,K\}) \quad \boldsymbol{y}_k = \boldsymbol{A}_k \boldsymbol{x}_k + \boldsymbol{w}_k, \tag{6}$$

and

$$(\forall \boldsymbol{x} \in \mathbb{R}^N)(\forall \boldsymbol{y} \in \mathbb{R}^M)\; f_{\boldsymbol{y}}(\boldsymbol{A}\boldsymbol{x}) = \frac{1}{2\sigma^2} \sum_{k=1}^{K} \|\boldsymbol{A}_k \boldsymbol{x}_k - \boldsymbol{y}_k\|_2^2, \tag{7}$$

where $\boldsymbol{A}_k \in \mathbb{R}^{M_k \times N_k}$ is a selection operator only acting on block $\boldsymbol{x}_k$, and $\boldsymbol{w}_k$ is a realization of an additive white Gaussian noise with variance σ^2.

B. AXDA-based splitting structure

Applying AXDA with a Gaussian kernel to (2) leads to an approximate model with posterior distribution

$$\widetilde{\pi}\big(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{u} \mid \boldsymbol{y}\big) \propto \exp\big(-h(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{u})\big) \tag{8}$$

with

$$\begin{aligned} h(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{u}) = f_{\boldsymbol{y}}(\boldsymbol{A}\boldsymbol{x}) + g(\boldsymbol{z}) \\ + \frac{1}{2\alpha}\|\boldsymbol{B}\boldsymbol{x} - \boldsymbol{z} + \boldsymbol{u}\|_2^2 - \frac{1}{2\beta}\|\boldsymbol{u}\|_2^2, \end{aligned} \tag{9}$$

where $(\alpha, \beta) \in]0, +\infty[^2$, and $(\boldsymbol{z}, \boldsymbol{u}) \in (\mathbb{R}^P)^2$ are auxiliary variables associated with the prior. Considering the model structure introduced in Section II, h can be rewritten as

$$h(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{u}) = \sum_{k=1}^{K} h_k(\boldsymbol{x}, \boldsymbol{z}_k, \boldsymbol{u}_k), \tag{10}$$

with, for every $k \in \{1,\dots,K\}$,

$$\begin{aligned} h_k(\boldsymbol{x}, \boldsymbol{z}_k, \boldsymbol{u}_k) = \frac{1}{2\sigma^2}\|\boldsymbol{y}_k - \boldsymbol{A}_k\boldsymbol{x}_k\|_2^2 + g_k(\boldsymbol{z}_k) \\ + \frac{1}{2\alpha}\|\boldsymbol{B}_k\boldsymbol{D}_k\boldsymbol{x} - \boldsymbol{z}_k + \boldsymbol{u}_k\|_2^2 + \frac{1}{2\beta}\|\boldsymbol{u}_k\|_2^2. \end{aligned} \tag{11}$$

The contribution of each function h_k will be handled by a single worker $k \in \{1,\dots,K\}$. Then, the blocks $\boldsymbol{x}_k$, $\boldsymbol{z}_k$ and $\boldsymbol{u}_k$ can directly be sampled from their corresponding conditional distribution. This is possible provided the (small number of) neighbouring elements have been exchanged with a suitable communication strategy detailed in Section III-C and III-D.

[2]In the case of a non-Gaussian noise, one may need to introduce an additional splitting variable in the likelihood term, following considerations reported in Section III-D.

C. Client-server versus SPMD architecture

The client-server architecture usually adopted with the AXDA approach [12], [14], exploiting conditional independence between variables, has several limitations. First, the number of processes that can be used is restricted to the number of independent splitting variables considered (typically less than 5). Second, the resulting algorithm does not allow data and parameters to be simultaneously distributed at a low communication cost. Finally, this architecture is not adapted to configurations where blocks are unbalanced (*i.e.*, with different update costs and memory requirements per block). In this context, load balancing issues and communication bottlenecks may occur.

To address these issues, the separability assumptions (3)–(5) can be exploited to design a distributed sampler relying on an SPMD architecture [15]. This strategy overcomes the limitations of the client-server setting, and offers multiple advantages. First, each worker can simultaneously be assigned a data block and the corresponding parameter block to maintain data locality. Second, the amount of redundancy between different blocks can be controlled when distributing parts of the inference task among multiple workers. Third, only a limited number of parameters requires to be communicated between neighbouring workers during the inference process. This results in lower communication costs compared to a client-server architecture.

D. Proposed algorithm

This section describes the structure of the proposed distributed sampler. The structure of $\widetilde{\pi}$, given in equations (8)-(10)-(11), ensures that each worker k can successively sample the variables $\boldsymbol{x}_k$, $\boldsymbol{z}_k$ and $\boldsymbol{u}_k$ once a few parameter values have been communicated between neighbouring workers. This configuration can be efficiently addressed with an SPMD architecture (see Section III-C).

We propose to combine a Gibbs sampler with the *proximal stochastic gradient Langevin algorithm* (PSGLA) [10], [11]. Precisely, we use the Gibbs sampler as global structure to draw samples of the different variables involved in (10). For the variables whose full conditional distribution is not standard (e.g., involving non-smooth potentials), we use a PSGLA transition kernel. A detailed pseudo-code of the proposed approach is given in Algorithm 1.

For every $k \in \{1,\dots,K\}$, the PSGLA step in Algorithm 1 line 8 requires the gradient of h_k with respect to $\boldsymbol{x}_k$ to be computed (see (11)). Communications are necessary to form these gradients, due to the couplings induced by $\boldsymbol{B}$ between blocks of parameters. Similar communications are necessary to sample $\boldsymbol{z}_k$ and $\boldsymbol{u}_k$ in steps 10 and 11, respectively.

Note that the algorithm presented in this section can be generalised to the full model described in Section II, when the matrices $\boldsymbol{C}_k$ are not reduced to identity. In this context, the coefficients of $\boldsymbol{x}$ selected by both $\boldsymbol{C}_k$ and $\boldsymbol{D}_k$ not contained in $\boldsymbol{x}_k$ can be communicated during a unique communication step. This allows the number of communications to be reduced.

Algorithm 1: Proposed SPMD distributed sampler.

Input: $(\alpha, \beta, \tau) \in]0, +\infty[^3$, $N_{\text{MC}} \in \mathbb{N}^*$

$\boldsymbol{x}^{(0)} \in \mathbb{R}^N$, $(\boldsymbol{z}^{(0)}, \boldsymbol{u}^{(0)}) \in (\mathbb{R}^P)^2$

$\nu = \alpha\beta(\alpha+\beta)^{-1}$; $\eta = 0.99\alpha$;

$\gamma = 0.99(1/\sigma^2 + \|\boldsymbol{B}^{\mathsf{T}}\boldsymbol{B}\|_2/\alpha)^{-1}$

for $t = 0$ **to** $N_{MC} - 1$ **do**

  for *each worker* $k \in \{1, \ldots, K\}$ **do** in parallel

    `// Update` $\boldsymbol{x}$ `with PSGLA kernel [11]`

    Communications to compute $\nabla_{\boldsymbol{x}_k} h_k$;

    $\boldsymbol{x}_k^{(t+1)} = \boldsymbol{x}_k^{(t)} - \gamma \nabla_{\boldsymbol{x}_k} h_k\big(\boldsymbol{x}^{(t)}, (\boldsymbol{z}_k^{(t)}, \boldsymbol{u}_k^{(t)})\big) + \sqrt{2\gamma}\boldsymbol{\xi}_k$,

    with $\boldsymbol{\xi}_k \sim \mathcal{N}(\boldsymbol{0}, \mathbf{I}_{N_k \times N_k})$;

    Communications to compute $\boldsymbol{B}_k \boldsymbol{D}_k \boldsymbol{x}^{(t+1)}$;

    `// Update` $\boldsymbol{z}$ `with PSGLA [11]`

    $\boldsymbol{z}_k^{(t+1)} = \text{prox}_{\eta g_k}\big(\boldsymbol{z}_k^{(t)} - \frac{\eta}{\alpha}(\boldsymbol{z}_k^{(t)} - \boldsymbol{B}_k\boldsymbol{D}_k\boldsymbol{x}^{(t+1)} - \boldsymbol{u}_k^{(t)}) + \sqrt{2\eta}\boldsymbol{\zeta}_k\big)$, with $\boldsymbol{\zeta}_k \sim \mathcal{N}(\boldsymbol{0}, \mathbf{I}_{P_k \times P_k})$;

    `// Sample` $\boldsymbol{u}$ `from its full conditional`

    $\boldsymbol{u}_k^{(t+1)} \sim \mathcal{N}\big(\frac{\nu}{\alpha}(\boldsymbol{z}_k^{(t+1)} - \boldsymbol{B}_k\boldsymbol{D}_k\boldsymbol{x}^{(t+1)}), \nu\mathbf{I}_{P_k \times P_k}\big)$;

Output: $(\boldsymbol{x}^{(t)}, \boldsymbol{z}^{(t)}, \boldsymbol{u}^{(t)})_{1 \le t \le N_{\text{MC}}}$

IV. Application to image inpainting

This section illustrates the proposed approach on the inpainting problem described in the previous section. Performance assessment is conducted on synthetic data, in comparison with the AXDA-based sampler [13, Section V-B] relying on another splitting choice and MYULA [9] transition kernels.

A. Model and experimental setting

Model. The inpainting problem (6) is addressed with a TV prior. It can be expressed as in (5) by taking $\boldsymbol{B}$ as the discrete gradient operator (i.e., $P = 2N$) and $g = \tau\|\cdot\|_{2,1}$, with $\tau > 0$ (see [17, Section 3.A]). The resulting posterior distribution admits a factorization of the form (3)-(5).

Compared methods. The proposed approach is compared with [13]. Note that it requires evaluations of the proximal operator of the TV, and cannot leverage a client-server architecture, as the parameters are conditionally dependent.

Performance is evaluated in terms of runtime and quality of the *minimum mean square error* (MMSE) estimator, denoted by $\widehat{\boldsymbol{x}}$. The latter is quantified in dB with the reconstruction signal-to-noise ratio (SNR):

$$\text{SNR}_{\boldsymbol{x}}(\widehat{\boldsymbol{x}}) = 20\log_{10}\left(\frac{\|\boldsymbol{x}\|_2}{\|\widehat{\boldsymbol{x}} - \boldsymbol{x}\|_2}\right). \tag{12}$$

Experimental setting. The sampler from [13] is systematically applied with the parameters provided by the authors, i.e., $(\alpha, \beta, \tau) = (9, 1, 0.2)$, using $\text{N}_{\text{MC}} = 5 \times 10^3$ and using $\text{N}_{\text{bi}} = 200$ burn-in samples to form $\widehat{\boldsymbol{x}}$. The proposed approach uses $(\alpha, \beta, \tau) = (9, 1, 0.2)$, with $\text{N}_{\text{MC}} = 10^4$ and $\text{N}_{\text{bi}} = 5 \times 10^3$.

All the experiments have been conducted on the high performance computing grid of the University of Lille[3]. A single compute node has been used, equipped with two 2.1 GHz, 18-core, Intel Xeon E5-2695 v4 series processors (36 CPU cores in total). Each worker thus corresponds to a single process running on one CPU core. The proposed approach has been implemented in Python using the `mpi4py` library [19].

[3] `https://hpc.univ-lille.fr/`

Fig. 2. Strong-scaling experiment results: ground truth and observations (first row), MMSE estimates (second row) and 95% credibility intervals (third row). For the rows 2 and 3, the results of the serial sampler [13] are reported on the left, those of the proposed approach on the right. Note that the credibility maps do not have the same scale to better highlight the distribution of uncertainties.

Two experimental settings have been considered. Both are composed of $M = \lfloor 0.6N \rfloor$ observations, corrupted with a white Gaussian noise whose variance results in a 40 dB SNR.

a) *Strong scaling experiment:* For a fixed-size problem $N = 256 \times 256$, the proposed algorithm is applied using an increasing number of workers $K \in \{1, 2, 4, 8, 16\}$.

b) *Large inpainting experiment:* The reconstruction of an $N = 1024 \times 1024$ image using 16 cores is investigated with the proposed sampler. Since the problem cannot be addressed in a reasonable time with the serial sampler from [13] and the proposed approach with $K = 1$, the comparison is exclusively conducted in terms of per-iteration runtime, computed over 10 iterations.

B. Results

a) Strong scaling experiment: The results of this experiments are reported in Table I and Fig. 2. We can observe that the proposed approach can provide estimators and confidence intervals similar to those obtained with the serial sampler [13] using a significantly smaller amount of time. In particular, the proposed approach is 5 to 60 times faster than the one

TABLE I

STRONG SCALING EXPERIMENT RESULTS. THE RUNTIME PER ITERATION IS REPORTED ON AVERAGE OVER THE ITERATIONS, WITH THE ASSOCIATED STANDARD DEVIATION. THE ACCELERATION FACTOR IS GIVEN WITH RESPECT TO THE PROPOSED SAMPLER WITH $K = 1$.

Method (number of cores)	[13, Section V-B] (1)	Proposed (1)	Proposed (2)	Proposed (4)	Proposed (8)	Proposed (16)
Runtime per sample (10^{-3} s)	65.56 ($\pm$ 2.08)	12.21 ($\pm$ 0.63)	6.07 ($\pm$ 0.42)	3.50 ($\pm$ 0.21)	1.93 ($\pm$ 0.77)	**1.08** ($\pm$ 2.35)
Per-iteration acceleration factor	0.19	1	2.01	3.49	6.33	**11.30**
Total runtime (s)	262.20	61.04	30.37	17.50	9.63	**5.38**
$\mathrm{SNR}_{\boldsymbol{x}}(\widehat{\boldsymbol{x}})$ (dB)	23.33	23.45	23.46	**23.48**	23.44	**23.48**

Fig. 3. Large scale inpainting experiment results. From left to right: ground truth, observations and MMSE estimate obtained with the proposed approach.

proposed in [13], at the price of a slightly lower SNR value. The 95% credibility intervals maps are displayed in the bottom row of Fig. 2. Both approaches show larger uncertainties on the contours, with lower uncertainty levels for the proposed approach. The overall estimation quality and lower per-iteration computing cost makes the proposed approach a highly compelling alternative to [13] for high dimensional problems.

A close to ideal acceleration is observed from 2 to 8 cores, decreasing around 16 cores (see Table I). As the number of workers increases, the communication costs is expected to represent a larger load compared to the amount of computations conducted by each worker. Overall, the proposed approach shows a competitive strong scaling performance on a single node from $K = 2$ to 16.

b) Large inpainting experiment: Dealing with the larger problem, the serial sampler [13] would require almost 4 hours to generate $\mathrm{N_{MC}} = 10^4$ samples, i.e., 1.35s per sample on average. In comparison, the proposed approach with $K = 1$ would need less than 1 hour, with 0.23s per sample. Using $K = 16$, only 90 seconds are needed (17.93 ms per sample on average) to produce the MMSE estimator displayed in Fig. 3, with $\mathrm{SNR}_{\boldsymbol{x}} = 26.60$ dB. That is, a factor 75 acceleration compared to [13], and 13 compared to $K = 1$. This illustrates the ability of the proposed approach to form large image estimates in a reasonable amount of time.

V. CONCLUSION AND PERSPECTIVES

We have considered an original application of the AXDA framework to address large scale inverse problems with a block-sparse structure. A versatile distributed sampler, leveraging a combination of PSGLA and Gibbs transition kernels, has been introduced to efficiently form estimators with quantified uncertainty. The proposed sampler has been implemented using an SPMD architecture, offering more flexibility in the distribution of data and parameter blocks compared to alternatives based on a client-server architecture. Experiments on a synthetic inpainting problem illustrated the remarkable performance of the proposed approach, with accelerations nearly proportional to the number of workers. Perspectives include the application of the proposed approach to more general problems and extensions to an asynchronous implementation.

REFERENCES

[1] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[2] L. Condat, "A primal-dual splitting method for convex optimization involving Lipschitzian, proximable and linear composite terms," *J. Opt. Soc. America*, vol. 158, pp. 460–479, 2013.

[3] B. C. Vũ, "A splitting algorithm for dual monotone inclusions involving cocoercive operators," *Advances in Computational Mathematics*, vol. 38, no. 3, pp. 667–681, Apr. 2013.

[4] J.-C. Pesquet and A. Repetti, "A class of randomized primal-dual algorithms for distributed optimization," *Journal of nonlinear and convex analysis*, vol. 16, no. 12, pp. 2453–2490, Nov. 2015.

[5] A. Chambolle and T. Pock, "A First-Order Primal-Dual Algorithm for Convex Problems with Applications to Imaging," *J. Math. Imag. Vision*, vol. 40, no. 1, pp. 120–145, May 2011.

[6] X. Cai, M. Pereyra, and J. D. McEwen, "Uncertainty quantification for radio interferometric imaging – I. Proximal MCMC methods," *Monthly Notices of the Royal Astronomical Society*, vol. 480, no. 3, pp. 4154–4169, 07 2018.

[7] C. P. Robert and G. Casella, *Monte Carlo Statistical Methods*, 2nd ed., ser. Springer Texts in Statistics. New York, NY: Springer, 2010.

[8] M. Pereyra, "Proximal Markov chain Monte Carlo algorithms," *Stat. Comput.*, vol. 26, no. 4, pp. 745–760, Jul. 2016.

[9] A. Durmus, É. Moulines, and M. Pereyra, "Efficient Bayesian Computation by Proximal Markov Chain Monte Carlo: When Langevin Meets Moreau," *SIAM J. Imaging Sci.*, vol. 11, no. 1, pp. 473–506, 2018.

[10] A. Wibisono, "Sampling as optimization in the space of measures: The Langevin dynamics as a composite optimization problem," in *Conference On Learning Theory*. PMLR, Jul. 2018, pp. 2093–3027.

[11] A. Salim and P. Richtàrik, "Primal dual interpretation of the proximal stochastic gradient langevin algorithm," in *Adv. in Neural Information Processing Systems*, vol. 33, 2020, pp. 3786–3796.

[12] L. J. Rendell, A. M. Johansen, A. Lee, and N. Whiteley, "Global consensus Monte Carlo," *J. Comput. and Graph. Stat.*, vol. 30, no. 2, pp. 249–259, 2021.

[13] M. Vono, N. Dobigeon, and P. Chainais, "Split-and-augmented Gibbs sampler - application to large-scale inference problems," *IEEE Trans. Signal Process.*, vol. 67, no. 6, pp. 1648–1661, March 2019.

[14] ——, "Asymptotically exact data augmentation: Models, properties, and algorithms," *J. Comput. and Graph. Stat.*, vol. 30, no. 2, pp. 335–348, 2021.

[15] F. Darema, "The SPMD model: Past, present and future," in *Recent Advances in Parallel Virtual Machine and Message Passing Interface*, Y. Cotronis and J. Dongarra, Eds., Berlin, Heidelberg, 2001, pp. 1–1.

[16] S. Mallat, *A Wavelet Tour of Signal Processing*, 3rd ed. Boston: Academic Press, 2009.

[17] L. Condat, "A Generic Proximal Algorithm for Convex Optimization—Application to Total Variation Minimization," *IEEE Signal Process. Lett.*, vol. 21, no. 8, pp. 985–989, Aug. 2014.

[18] P.-A. Thouvenin, A. Abdulaziz, M. Jiang, A. Repetti, A. Dabbech, and Y. Wiaux, "A faceted prior for scalable wideband imaging: Application to radio astronomy," in *Proc. IEEE Int. Workshop Comput. Adv. Multi-Sensor Adaptive Process. (CAMSAP)*, Dec. 2019.

[19] L. Dalcin and Y.-L. L. Fang, "mpi4py: Status update after 12 years of development," *IEEE Comput. Sci. Eng.*, vol. 23, no. 4, pp. 47–54, 2021.