ASYNC-RED: A PROVABLY CONVERGENT ASYNCHRONOUS BLOCK PARALLEL STOCHASTIC METHOD USING DEEP DENOISING PRIORS

Yu Sun¹, Jiaming Liu¹, Yiran Sun¹, Brendt Wohlberg², Ulugbek S. Kamilov¹
¹Washington University in St. Louis, ²Los Alamos National Laboratory
{sun.yu,jiaming.liu,yiran.s,kamilov}@wustl.edu, brendt@ieee.org

ABSTRACT

Regularization by denoising (RED) is a recently developed framework for solving inverse problems by integrating advanced denoisers as image priors. Recent work has shown its state-of-the-art performance when combined with pre-trained deep denoisers. However, current RED algorithms are inadequate for parallel processing on multicore systems. We address this issue by proposing a new asynchronous RED (ASYNC-RED) algorithm that enables asynchronous parallel processing of data, making it significantly faster than its serial counterparts for large-scale inverse problems. The computational complexity of ASYNC-RED is further reduced by using a random subset of measurements at every iteration. We present complete theoretical analysis of the algorithm by establishing its convergence under explicit assumptions on the data-fidelity and the denoiser. We validate ASYNC-RED on image recovery using pre-trained deep denoisers as priors.

1 INTRODUCTION

Imaging inverse problems seek to recover an unknown image $x \in \mathbb{R}^n$ from its noisy measurements $y \in \mathbb{R}^m$. Such problems arise in many fields, ranging from low-level computer vision to biomedical imaging. Since many imaging inverse problems are ill-posed, it is common to regularize the solution by using prior information on the unknown image. Widely-adopted image priors include total variation, low-rank penalties, and transform-domain sparsity (Rudin et al., 1992; Figueiredo & Nowak, 2001; 2003; Hu et al., 2012; Elad & Aharon, 2006).

There has been considerable recent interest in plug-and-play priors (PnP) (Venkatakrishnan et al., 2013; Sreehari et al., 2016) and regularization by denoising (RED) (Romano et al., 2017), as frameworks for exploiting image denoisers as priors for image recovery. The popularity of deep learning has led to a wide adoption of deep denoisers within PnP/RED, leading to their state-of-the-art performance in a variety of applications, including image restoration (Mataev et al., 2019), phase retrieval (Metzler et al., 2018), and tomographic imaging (Wu et al., 2020). Their empirical success has also prompted a follow-up theoretical work clarifying the existence of explicit regularizers (Reehorst & Schniter, 2019), providing new interpretations based on fixed-point projections (Cohen et al., 2020), and analyzing their coordinate/online variants (Sun et al., 2019a; Wu et al., 2020). Nonetheless, current PnP/RED algorithms are inherently serial. As illustrated in Fig. 1, this makes them suboptimal on multicore systems that are often required for processing large-scale datasets (Recht et al., 2011), such as those involving biomedical (Ong et al., 2020) and astronomical images (Akiyama et al., 2019).

We address this gap by proposing a novel asynchronous RED (ASYNC-RED) algorithm. The algorithm decomposes the inference problem into a sequence of partial (block-coordinate) updates on $x$ executed asynchronously in parallel over a multicore system. ASYNC-RED leads to a more efficient usage of available cores by avoiding synchronization of partial updates. ASYNC-RED is also scalable in terms of the number of measurements, since it processes only a small random subset of $y$ at every iteration. We present two new theoretical results on the convergence of ASYNC-RED based on a unified set of explicit assumptions on the data-fidelity and the denoiser. Specifically, we establish its fixed-point convergence in the batch setting and extend this analysis to the randomized minibatch scenario. Our results extend recent work on serial block-coordinate RED (BC-RED) (Sun et al.)
Figure 1: Visual illustration of serial and parallel image recovery on a multicore system. (a) Serial processing uses only one core of the system for every iteration. (b) Synchronous parallel processing has to wait for the slowest core to finish before starting the next iteration. (c) Asynchronous parallel processing can continuously iterate using all the cores without waiting. (d) Asynchronous parallel processing using the stochastic gradient leads to additional flexibility. (a), (b), and (c) use all the corresponding measurements at every iteration, while (d) uses only a small random subset at a time. ASYNC-RED adopts the schemes shown in (c) and (d).

and are fully consistent with the traditional asynchronous parallel optimization methods [Lian et al., 2015; Sun et al., 2017]. We numerically validate ASYNC-RED on image recovery from linear and noisy measurements using pre-trained deep denoisers as image priors.

2 BACKGROUND

Inverse problems. Inverse problems are traditionally formulated as a composite optimization problem

$$\hat{x} = \arg \min_{x \in \mathbb{R}^n} g(x) + h(x), \quad (1)$$

where $g$ is the data-fidelity term that ensures consistency of $x$ with the measured data $y$ and $h$ is the regularizer that infuses the prior knowledge on $x$. For example, consider the smooth $\ell_2$-norm data-fidelity term $g(x) = \|y - Ax\|_2^2$, which assumes a linear observation model $y = Ax + e$, and the nonsmooth TV regularizer $h(x) = \tau \|Dx\|_1$, where $\tau > 0$ is the regularization parameter and $D$ is the image gradient [Rudin et al., 1992].

Regularization by denoising (RED). RED is a recent methodology for imaging inverse problems that seeks vectors $x^* \in \mathbb{R}^n$ satisfying

$$G(x^*) = \nabla g(x^*) + \tau (x^* - D_\sigma(x^*)) = 0 \iff x^* \in \text{zer}(G) := \{x \in \mathbb{R}^n : G(x) = 0\}, \quad (2)$$

where $\nabla g$ denotes the gradient of the data-fidelity term and $D_\sigma : \mathbb{R}^n \to \mathbb{R}^n$ is an image denoiser parameterized by $\sigma > 0$. Under additional technical assumptions, the solutions $x^* \in \text{zer}(G)$ can be associated with an explicit objective function of form (1). Specifically, when $D_\sigma$ is locally homogeneous and has a symmetric Jacobian satisfying strong passivity [Romano et al., 2017; Reehorst & Schniter, 2019], $H(x) := x - D_\sigma(x)$ corresponds to the gradient of a convex regularizer

$$h(x) = \frac{1}{2} x^T (x - D_\sigma(x)). \quad (3)$$

A simple strategy, known as GM-RED, for computing $x^* \in \text{zer}(G)$ is based on the first-order fixed-point iteration

$$x^t = x^{t-1} - \gamma G(x^{t-1}), \quad \text{with} \quad G : \mathbb{R}^n \to \mathbb{R}^n, \quad (4)$$

where $\gamma > 0$ denotes the stepsize. In this paper, we extend this first-order RED algorithm to design ASYNC-RED. Since many denoisers do not satisfy the assumptions necessary for having an explicit objective [Reehorst & Schniter, 2019], our theoretical analysis considers a broader setting where $D_\sigma$ does not necessarily correspond to any explicit regularizer. The benefit of our analysis is that it accommodates powerful deep denoisers (such as DnCNN [Zhang et al., 2017a]) that have been shown to achieve the state-of-the-art performance [Sun et al., 2019a; Wu et al., 2020; Cohen et al., 2020].
Plug-and-play priors (PnP) and other related work. There are other lines of works that combine the iterative methods with advanced denoisers. One closely-related framework is known as the deep mean-shift priors (Bigdeli et al., 2017). It develops an implicit regularizer whose gradient is specified by a denoising autoencoder. Another well-known framework is PnP, which generalizes proximal methods by replacing the proximal map with an image denoiser (Venkatakrishnan et al., 2013). Applications and theoretical analysis of PnP are widely studied in (Sreetharan et al., 2016; Zhang et al., 2017b; Liu et al., 2018; Zhou et al., 2018; Lian et al., 2018). Note that while we developed ASYNC-RED as a variant of RED, our framework and analysis can be also potentially applied to PnP/CE. The plug-in strategy can be also applied to another family of algorithms known as approximate message passing (AMP) (Metzler et al., 2016a; Fletcher et al., 2018). The AMP-based algorithms are known to be nearly-optimal for random measurement matrices, but are generally unstable for general A (Rangan et al., 2014; 2015).

Asynchronous parallel optimization. There are two main lines of work in asynchronous parallel optimization, the one involving the asynchrony in coordinate updates (Iutzeler et al., 2013; Liu et al., 2015), and the other focusing on the study of various asynchronous stochastic gradient methods (Recht et al., 2011; Lian et al., 2015; Liu et al., 2018; Zhou et al., 2018; Lian et al., 2018).

Our work contributes to the area by developing a novel deep-regularized asynchronous parallel method with provable convergence guarantees.

3 ASYNCHRONOUS RED

ASYNC-RED allows efficient processing of data by simultaneously considering the asynchronous partial updates of solution x and the use of randomized subset of measurements y. In this section, we introduce the algorithmic details of our method. We start with the basic batch formulation of ASYNC-RED (ASYNC-RED-BG) followed by its minibatch variant (ASYNC-RED-SG).

3.1 ASYNC-RED USING BATCH GRADIENT

When the gradient uses all the measurements y ∈ ℝᵐ, ASYNC-RED-BG is the asynchronous extension of the recent block-coordinate RED (BC-RED) algorithm (Sun et al., 2019a). Consider the decomposition of the variable space ℝⁿ into b ≥ 1 blocks

\[ x = (x_1, \ldots, x_b) \in \mathbb{R}^{n_1} \times \cdots \times \mathbb{R}^{n_b} = \mathbb{R}^n \quad \text{with} \quad n = n_1 + n_2 + \cdots + n_b. \]

For each \( i \in \{1, \ldots, b\} \), we introduce the operator \( U_i : \mathbb{R}^{n_i} \rightarrow \mathbb{R}^n \) that injects a vector in \( \mathbb{R}^{n_i} \) into \( \mathbb{R}^n \) and its transpose \( U_i^T \) that extracts the \( i \)th block from a vector in \( \mathbb{R}^n \). This directly implies that

\[ I = U_1U_1^T + \cdots + U_bU_b^T \quad \text{and} \quad \|x\|_2^2 = \|x_1\|_2^2 + \cdots + \|x_b\|_2^2 \quad \text{with} \quad x = U_i^T x_i. \]

In analogy to the RED operator \( G \) in (2), we define the block-coordinate operator \( G_i \) as

\[ G_i(x) := U_iU_i^T G(x), \quad \text{with} \quad x \in \mathbb{R}^n \quad \text{and} \quad G_i : \mathbb{R}^n \rightarrow \mathbb{R}^n. \]

Due to the asynchrony in the block updates, the iterate might be updated several times by different cores during a single update cycle of a core, which means that the evaluation of \( x^{k+1} \) relies on a stale iterate \( \bar{x}^k \)

\[ x^{k+1} \leftarrow x^k - \gamma G_i(\bar{x}^k), \quad \text{with} \quad \bar{x}^k = x^k + \sum_{s=k-\Delta_k}^{k-1} (x^s - x^{s+1}), \quad \Delta_k \leq \lambda. \]

Here, we assume that the stale iterate \( \bar{x}^k \) exits as a state of \( x \) in the shared memory, and the delay between them is bounded by a finite number \( \lambda \in \mathbb{Z}_+ \). These two assumptions are often referred to as the consistent read (Recht et al., 2011) and the bounded delay (Liu & Wright, 2015) in the traditional asynchronous block coordinate optimization. Although we implement the consistent read in ASYNC-RED, the algorithm never imposes a global lock on \( x^k \). We refer to Supplement A for the related discussion.
We clarify the difference between \( \mathbf{A} \) specifically consider the \( \mathbf{B} \) with the computation resources at hand. A where each \( \mathbf{C} \) is evaluated on the subset \( \mathbf{D} \) is proportional to the total number \( \mathbf{E} \) do...\( \mathbf{F} \) for global \( \mathbf{G} \), \( \mathbf{H} \) \( \mathbf{I} \) \( \mathbf{J} \) \( \mathbf{K} \) \( \mathbf{L} \) \( \mathbf{M} \) \( \mathbf{N} \) \( \mathbf{O} \) \( \mathbf{P} \) \( \mathbf{Q} \) \( \mathbf{R} \) \( \mathbf{S} \) \( \mathbf{T} \) \( \mathbf{U} \) \( \mathbf{V} \) \( \mathbf{W} \) \( \mathbf{X} \) \( \mathbf{Y} \) \( \mathbf{Z} \)

The first variant, \( \text{ASYNC-RED-BG} \), is summarized in Algorithm 1, where \( \text{read}(\cdot) \) reads a block from the shared memory to the local memory. When the algorithm is run on a single core system without parallelization (that is to say \( \bar{x}^k = x^k \)), it reduces to the normal BC-RED algorithm. Hence, our analysis is also applicable to BC-RED.

We specifically consider the random block selection strategy in \( \text{ASYNC-RED-BG} \), namely that every block index \( i_k \) is selected as an i.i.d random variable uniformly distributed over \( \{1, \ldots, b\} \). Such a strategy is commonly adopted for simplifying the convergence analysis. Nevertheless, our method and analysis can be generalized to the scenario where \( i_k \) follows some arbitrary probability \( \mathcal{P}(i_k = i) = p_i \) specified by the user.

Compared with serial RED algorithms, \( \text{ASYNC-RED-BG} \) enjoys considerable scalability by dividing the computation of the full operator \( G \) into \( b \) parallel evaluation of \( G_i \) distributed across all cores. Thus, without any modification to the algorithmic design, one can easily improve the performance of the algorithm by simply integrating more cores into the system. In Section 5, we experimentally demonstrate the significant speed-up and scale-up in solving the context of image recovery.

### 3.2 Async-RED Using Stochastic Gradient

The scale of measurements is another important factor influencing the computational complexity in the large-scale inference tasks. \( \text{ASYNC-RED-SG} \) improves the applicability of \( \text{ASYNC-RED} \) to these cases by further considering the decomposition of the measurement space \( \mathbb{R}^m \) into \( \ell \geq 1 \) blocks

\[
\mathbf{y} = (\mathbf{y}_1, \cdots, \mathbf{y}_\ell) \in \mathbb{R}^{m_1} \times \cdots \times \mathbb{R}^{m_\ell} = \mathbb{R}^m \quad \text{with} \quad m = m_1 + m_2 + \cdots + m_\ell.
\]

Hence, \( \text{ASYNC-RED-SG} \) considers the following data-fidelity \( g \) and its gradient \( \nabla g \)

\[
g(\mathbf{x}) = \frac{1}{\ell} \sum_{j=1}^\ell g_j(\mathbf{x}) \quad \Rightarrow \quad \nabla g(\mathbf{x}) = \frac{1}{\ell} \sum_{j=1}^\ell \nabla g_j(\mathbf{x}),
\]

where each \( g_j \) is evaluated on the subset \( \mathbf{y}_j \in \mathbb{R}^{m_j} \) of the full \( \mathbf{y} \). From (8), we know that the computation of \( \nabla g(\mathbf{x}) \) is proportional to the total number \( \ell \). To reduce the per-iteration cost, we follow the idea of stochastic optimization to approximate the batch gradient by using the stochastic gradient that relies on a minibatch of \( w \ll \ell \) measurements

\[
\hat{\nabla} g(\mathbf{x}) = \frac{1}{w} \sum_{s=1}^w \nabla g_{j_s}(\mathbf{x}),
\]

where \( j_s \) is picked from the set \( \{1, \ldots, \ell\} \) as i.i.d uniform random variable. Based on the minibatch gradient, we define the block stochastic operator \( \check{G}_i : \mathbb{R}^n \to \mathbb{R}^n \) as

\[
\check{G}_i := U_i U_i^\top \hat{G}(\mathbf{x}), \quad \text{with} \quad \hat{G} := \hat{\nabla} g(\mathbf{x}) + \tau (\mathbf{x} - D_x(\mathbf{x})), \quad \check{G} : \mathbb{R}^n \to \mathbb{R}^n.
\]

Note that the computation of \( \check{G}_i \) is now dependent on the minibatch size \( w \) that is adjustable to cope with the computation resources at hand. \( \text{ASYNC-RED-SG} \) is summarized in Algorithm 2.

The operation minibatch\( \hat{G}(\cdot) \) computes the estimate of \( G \) based on \( w \) randomly selected measurements. We clarify the difference between \( \text{ASYNC-RED-BG} \) and \( \text{ASYNC-RED-SG} \) via a specific example.
Algorithm 2 Async-RED-SG

1: input: $x^0 \in \mathbb{R}^n$, $\gamma > 0$, $\tau > 0$.
2: setup: A multicore system with one shared memory storing $x$ and global iteration $k$.
3: for global $k = 1, 2, 3, \ldots$ do
4:     $\bar{x}^k \leftarrow \text{read}(x)$
5:     $G(\bar{x}^k) \leftarrow \text{minibatch}G(\bar{x}^k, w)$ with random $j_w \in \{1, \ldots, \ell\}$ \triangleright Minibatch Gradient
6:     $\tilde{G}_{ik}(\bar{x}^k) \leftarrow U_k U_k^T G(\bar{x}^k)$ with random $i_k \in \{1, \ldots, b\}$ \triangleright Block Operation
7:     $x^{k+1} \leftarrow x^k - \gamma \tilde{G}_{ik}(\bar{x}^k)$
8: update $x$ in the shared memory using $x^{k+1}$
9: end for

Consider the least-squares function $g$ with a block-friendly operator $A$ and a block-efficient denoiser $D_g$. We can write the update of Async-RED-BG regarding a single iteration as

$$G_i(\bar{x}) = A_i^T (A_i \bar{x} - y_i) + \tau (\bar{x}_i - D(\bar{x}_i)),$$

where $\bar{x}$ is the delayed iterate for $x$, and $A_i \in \mathbb{R}^{m \times n_i}$ is a submatrix of $A$ consisting of columns corresponding to the $i$th blocks. Although the per-iteration complexity is reduced by roughly $b = n_i/n$ times by working with $A_i$ instead of $A$, Async-RED-BG still needs to work with all the measurements $y_i$ related to the $i$th block at every iteration. Consider the corresponding update of Async-RED-SG with one measurement used at a time

$$\tilde{G}_i(\bar{x}) = A_i^T (A_i \bar{x} - y_{ji}) + \tau (\bar{x}_i - D(\bar{x}_i)),$$

where $y_{ji}$ denotes the $j$th measurement of $x_i$, and $A_{ji} \in \mathbb{R}^{m_j \times n_j}$ is the submatrix crossed by the rows and columns corresponding to the $j$th measurement and the $i$th blocks. This indicates that the reduction of the per-iteration complexity from Async-RED-BG to Async-RED-SG can be up to $\ell = m_i/m$ times. In the practice, it is common to use $w > 1$ measurements at a time to optimize the total runtime. Note that if $U = U^T = I$, Async-RED-BG becomes the asynchronous stochastic RED algorithm. In the next section, we will present a complete analysis of Async-RED and theoretically discuss its connection to the related algorithms.

4 Convergence Analysis of Async-RED

The proposed analysis is based on the following explicit assumptions. Note that these assumptions serve as sufficient conditions for the convergence.

Assumption 1. We assume bounded maximal delay $\lambda < \infty$. Hence, during any update cycle of an agent, the estimate $x$ in the shared memory is updated at most $\lambda \in \mathbb{Z}_+$ times by other cores.

The value of $\lambda$ is often dependent on the number of cores involved in the computation (Wright [2015]). If every core takes a similar amount of time to compute its update, $\lambda$ is expected to be a multiple of the number of cores. Related work has investigated the convergence with unbounded maximal delays in the context of traditional optimization (Hannah & Yin [2013]; Peng et al. [2019]; Zhou et al. [2018]).

Assumption 2. The operator $G$ is such that $\text{zer}(G) \neq \emptyset$, and the distance of the initial $x^0 \in \mathbb{R}^n$ to any element in $\text{zer}(G)$ is bounded, that is $\|x^0 - x^*\| \leq R_0$ for all $x^* \in \text{zer}(G)$ with $R_0 < \infty$.

This assumption ensures the existence of a solution for the RED problem and is related to the existence of minimizers in traditional coordinate minimization (Nesterov [2012]; Beck & Teboulle [2013]).

Assumption 3. (a) Every component function $g_i$ is convex differentiable and has a Lipschitz continuous gradient of constant $L_i > 0$. (b) At every update, the stochastic gradient is unbiased estimator of $\nabla g$ that has a bounded variance:

$$E \left[ \nabla g(x) \right] = g(x), \quad E \left[ || \nabla g(x) - \nabla g(x) ||^2 \right] \leq \frac{\nu^2}{w}, \quad x \in \mathbb{R}^n, \quad \nu > 0.$$

The first part of the assumption implies that $g$ is also convex and has Lipschitz continuous gradient with constant $L = \max\{L_1, \ldots, L_\ell\}$. The second part is a standard assumption on the unbiasedness.
We can now state the theorems on $A$.

**Assumption 4.** The denoiser $D_\sigma$ is a nonexpansive operator $\|D_\sigma(x) - D_\sigma(y)\| \leq \|x - y\|$.

Compared with the conditions stated in Section 2 (namely, that it is locally homogeneous with a symmetric Jacobian), our requirement on the denoiser is milder. One can train a nonexpansive $D_\sigma$ by constraining the Lipschitz constant of $D_\sigma$ via the spectral normalization, which is an active area of research in deep learning [Miyato et al., 2018; Sedghi et al., 2019; Ami et al., 2019; Jerris et al., 2020].

We can now state the theorems on $ASYNC-RED$.

**Theorem 1.** Let Assumptions 4 hold true. Run $ASYNC-RED-BG$ for $t > 0$ iterations with uniform i.i.d block selection using a fixed step-size $\gamma \in (0, 1/(1 + 2\lambda)(L + 2\tau))$. Then, the iterates of the algorithm satisfy

$$
\min_{0 \leq k \leq t-1} \mathbb{E} \left[ \|G(x^k)\|^2 \right] \leq \left[ \frac{D}{b} + 2 \right] \left( \frac{L + 2\tau}{\gamma t} \right) R_0^2 + \left[ \frac{2D}{b} + 2 \right] \frac{C}{\sqrt{w}}
$$

where $D = 2\lambda^2/(1 + \lambda)^2$ is a constant.

**Theorem 2.** Let Assumptions 4 hold true. Run $ASYNC-RED-SG$ for $t > 0$ iterations with uniform i.i.d selections of blocks and measurements using a fixed step-size $\gamma \in (0, 1/(1 + 2\lambda)(L + 2\tau))$. Then, the iterates of the algorithm satisfy

$$
\min_{0 \leq k \leq t-1} \mathbb{E} \left[ \|G(x^k)\|^2 \right] \leq \left[ \frac{D}{b} + 2 \right] \left( \frac{L + 2\tau}{\gamma t} \right) R_0^2 + \left[ \frac{2D}{b} + 2 \right] \frac{C}{\sqrt{w}}
$$

where $C = (L + 2\tau)(1 + \lambda)\nu^2$ and $D = 2\lambda^2/(1 + \lambda)^2$ are constants.

**Remark 1.** Set the stepsize to be $\gamma = 1/\sqrt{wt}$. If the maximal delay satisfies $\lambda \leq (1/2)[\sqrt{wt}/(L + 2\tau) - 1]$, then after $t > 0$ iterations we have

$$
\min_{0 \leq k \leq t-1} \mathbb{E} \left[ \|G(x^k)\|^2 \right] \leq \left[ \frac{D}{b} + 2 \right] \left( \frac{L + 2\tau}{\sqrt{wt}} \right) R_0^2 + \left[ \frac{2D}{b} + 2 \right] \frac{C}{\sqrt{wt}}.
$$
We validate our theorems on the CS task with 6 test images selected from the Set 12 dataset [Zhang et al. 2017a]. Each test image is rescaled to the size of 240 × 240 pixels (see Fig. 6 in the supplement for the visualization). The block-diagonal matrix \( A \) is set to consist of 9 submatrices, corresponding to a 3 × 3 grid of blocks with the size of 80 × 80 pixels in every image. The elements in \( A \) are i.i.d zero-mean Gaussian random variables of variance of 1/m, and the compression ratio is set to be \( \frac{m}{n} = 0.7 \), which indicates that the total number of measurements is 4800 for each block. We obtain the measurements by multiplying \( A \) with each vectorized image and adding additional noise corresponding to the input SNR of 30 dB. Finally, we use the normalized distance \( \frac{\|G(x^k)\|_2^2}{\|G(x^0)\|_2^2} \) to quantify the fixed-point convergence, with \( b \) block updates grouped as one iteration. The distance is expected to approach zero as the algorithm converges to a fixed point. The average performance of all methods is obtained by running a single trial for each image.

Theorem 1 establishes the convergence of Async-RED-BG to the fixed point set \( \text{zer}(G) \). This is illustrated in Fig. 2 for four different numbers of accessible cores \( n_c \in \{2, 4, 6, 8\} \). In the left figure, the average normalized distance is plotted against the iteration number, while the middle and
Asynchronous parallel methods have gained increasing importance in optimization for solving large-scale imaging inverse problems. We have introduced \texttt{ASYNC-RED} as an extension of the recent RED framework and theoretically analyze its convergence in batch and stochastic settings. We have experimentally demonstrated the effectiveness of \texttt{ASYNC-RED} by reconstructing a CT image from its 180 projections. For block parallel updates, the image is decomposed into 16 blocks, each having the size of 200 \times 200 pixels. The Radon matrix used in the experiment corresponds to 180 angles with 1131 detectors, and the noise level is set to 70 dB. We refer to Supplement D.2 for additional technical details. Fig. \ref{fig:recon_results} shows the visual illustration of the reconstructed images by \texttt{ASYNC-RED-BG/SG} and \texttt{GM-RED}. Each algorithm starts from the filtered back-projection (FBP) of the measurements and runs for 1 hour. Here, \texttt{ASYNC-RED-SG} randomly uses one-third of the total measurements at every iteration. Given the same amount of time, \texttt{ASYNC-RED-BG/SG} successfully mitigates the noise-artifacts, while the result of \texttt{GM-RED} is still noisy. In particular, the per-iteration time cost of \texttt{ASYNC-RED-BG/SG} and \texttt{GM-RED} is 5.23, 3.21, and 19.19 seconds, respectively. This experiment clearly illustrates the fast processing speed of the asynchronous procedure.

5.2 Effectiveness for Computational Imaging

We additionally demonstrate the effectiveness of our algorithm by reconstructing a 800 \times 800 CT image from its 180 projections. For block parallel updates, the image is decomposed into 16 blocks, each having the size of 200 \times 200 pixels. The Radon matrix used in the experiment corresponds to 180 angles with 1131 detectors, and the noise level is set to 70 dB. We refer to Supplement D.2 for additional technical details. Fig. \ref{fig:recon_results} shows the visual illustration of the reconstructed images by \texttt{ASYNC-RED-BG/SG} and \texttt{GM-RED}. Each algorithm starts from the filtered back-projection (FBP) of the measurements and runs for 1 hour. Here, \texttt{ASYNC-RED-SG} randomly uses one-third of the total measurements at every iteration. Given the same amount of time, \texttt{ASYNC-RED-BG/SG} successfully mitigates the noise-artifacts, while the result of \texttt{GM-RED} is still noisy. In particular, the per-iteration time cost of \texttt{ASYNC-RED-BG/SG} and \texttt{GM-RED} is 5.23, 3.21, and 19.19 seconds, respectively. This experiment clearly illustrates the fast processing speed of the asynchronous procedure.

6 Conclusion

Asynchronous parallel methods have gained increasing importance in optimization for solving large-scale imaging inverse problems. We have introduced \texttt{ASYNC-RED} as an extension of the recent RED framework and theoretically analyze its convergence in batch and stochastic settings. We have...
validated its convergence guarantees and demonstrated its effectiveness in CT image reconstruction. We note that this work is complementary to traditional acceleration strategies, such as Nesterov acceleration and variance-reduction, commonly used in optimization. Future work will investigate
ASYNC-RED with Nesterov acceleration (as was done in \cite{Hannah2019} for traditional asynchronous block-coordinate algorithms) and variance-reduction (as was done in \cite{Johnson2013} for traditional stochastic gradient method) to better understand the tradeoffs between acceleration and scalability in multicore systems. We will additionally investigate theoretical limits of ASYNC-RED in the unbounded maximal delay setting.

ACKNOWLEDGEMENT

Research presented in this article was supported by NSF award CCF-1813910 and the Laboratory Directed Research and Development program of Los Alamos National Laboratory under project number 20200061DR.

REFERENCES


