Distributed Block-Separable Ordered Subsets for Helical X-ray CT Image Reconstruction

Donghwan Kim and Jeffrey A. Fessler

***Abstract*—Statistical reconstruction for low-dose CT can provide desirable image quality, but the computational burden still remains a challenge, particularly for large 3D helical scans. Parallel computing can help reduce computation times, but simple parallelization methods for CT reconstruction can be hampered by relatively large data communication times between nodes that do not share memory. This paper describes a block-separable surrogate approach to developing algorithms that facilitate parallelization. These methods reduce communication between nodes and allow multiple independent updates on each node, while attempting to maintain convergence rates of recent accelerated algorithms. As a preliminary study, we investigated one version of the proposed algorithm in Matlab using a simulated 3D helical CT scan.**

I. Introduction

Statistical X-ray CT reconstruction [1] can reduce noise and artifacts from safer low-dose CT scans unlike conventional filtered back-projection (FBP) method, but reconstructing large helical CT scan data requires long computation times. Ordered subsets (OS) methods based on separable quadratic surrogates (SQS) [2], [3] have been used widely in CT research because they are very parallelizable and heuristically reduce the computational cost by using only a subset of data per iteration. However, these efficient OS methods are still slower than desired for ubiquitous clinical use, so further acceleration of OS methods has been studied [4]–[6].

Distributed computing resources have been central to accelerating many big data computational problems. CT researchers also have pursued speedups by distributing expensive computations, such as projection operations [7], [8] and unregularized SIRT methods [9], to several nodes (that do not share memory). For regularized statistical CT reconstruction, the OS-SQS method with momentum (OS-SQS-mom) [4] was implemented on a cloud service, showing promising reconstruction time [10]. (This algorithm was also implemented on GPU for accelerated interventional CT reconstruction [11].)

Although the distributed implementation of OS-SQS-mom reduced run time substantially, data communication between nodes became a bottleneck when using many nodes [10]. To reduce communication, while aiming to preserve the fast convergence speed of OS-SQS-mom, this paper adapts a block-separable surrogate (BSS) technique [12] to 3D helical CT scans.

This paper first reviews the OS-SQS-mom algorithm [4] and its distributed implementation [10]. For reduced communication time between nodes, we review and adapt BSS [12] with the OS-SQS-mom method. The result section reports a preliminary investigation of a Matlab version of the proposed parallelizable method applied to a 3D helical scan, comparing its convergence speed to that of standard OS-SQS-mom.

D. Kim and J. A. Fessler are with the Dept. of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109 USA (e-mail: kimdongh, fessler@umich.edu). Supported in part by NIH grant U01-EB-018753 and equipment donations from Intel Corporation.

II. Statistical image reconstruction using ordered subsets with momentum

We reconstruct a patient image $\hat{\boldsymbol{x}} \in \mathbb{R}_+^{N_p}$ by minimizing a (strictly convex and continuously differentiable) penalized weighted least squares (PWLS) cost function [1]:

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x} \succeq 0} \left\{ \Psi(\boldsymbol{x}) \triangleq \mathsf{L}(\boldsymbol{x}) + \mathsf{R}(\boldsymbol{x}) \right\}, \tag{1}$$

$$\mathsf{L}(\boldsymbol{x}) \triangleq \frac{1}{2} \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_{\boldsymbol{W}}^2,$$

where $\boldsymbol{x} \in \mathbb{R}_+^{N_p}$ is an (unknown) image, $\boldsymbol{y} \in \mathbb{R}^{N_d}$ is a noisy measured sinogram data, $\boldsymbol{A} \in \mathbb{R}_+^{N_d \times N_p}$ is a forward projection operator [13], $\boldsymbol{W} \in \mathbb{R}^{N_d \times N_d}$ is a diagonal statistical weighting matrix, and $\mathsf{R}(\boldsymbol{x})$ is an edge-preserving regularizer.

Among many optimization methods, we first focus on using massively *parallelizable* SQS methods [2], [3] that iteratively minimize the SQS surrogate at the nth iteration:

$$\Psi(\boldsymbol{x}) \le \phi_{\text{SQS}}(\boldsymbol{x}; \boldsymbol{x}^{(n)}) \tag{2}$$

$$\triangleq \Psi(\boldsymbol{x}^{(n)}) + (\boldsymbol{x} - \boldsymbol{x}^{(n)})^\top \nabla \Psi(\boldsymbol{x}^{(n)}) + \frac{1}{2} ||\boldsymbol{x} - \boldsymbol{x}^{(n)}||_{\boldsymbol{D}}^2$$

for a *diagonal* majorizing matrix $\boldsymbol{D} \in \mathbb{R}^{N_p \times N_p}$, as

$$\boldsymbol{x}^{(n+1)} = \arg\min_{\boldsymbol{x} \succeq 0} \phi_{\text{SQS}}(\boldsymbol{x}; \boldsymbol{x}^{(n)})$$

$$= \left[\boldsymbol{x}^{(n)} - \boldsymbol{D}^{-1} \nabla \Psi(\boldsymbol{x}^{(n)}) \right]_+, \tag{3}$$

where $[\cdot]_+$ enforces the nonnegativity constraint. After computing the gradient, SQS updates each voxel independently. The usual simple choice of $\boldsymbol{D}$ is [2]:

$$\boldsymbol{D} = \text{diag}\{ \boldsymbol{A}^\top \boldsymbol{W} \boldsymbol{A} \boldsymbol{1} \} + \boldsymbol{D}^{\mathsf{R}}, \tag{4}$$

where $\boldsymbol{1} = \{1\} \in \mathbb{R}^{N_p}$ and $\boldsymbol{D}^{\mathsf{R}}$ is a majorizer for the Hessian of $\mathsf{R}(\boldsymbol{x})$ [2], [3].

Computing the gradient $\nabla \Psi(\boldsymbol{x})$ in (3) is expensive for large helical scans, so OS methods [2] are used widely with SQS by dividing N_d projection data into (non-overlapping) M subsets and using the following approximation:

$$\nabla \Psi(\boldsymbol{x}) \approx M \nabla \Psi_m(\boldsymbol{x}) \tag{5}$$

$$\triangleq M \boldsymbol{A}_m^\top \boldsymbol{W}_m (\boldsymbol{A}_m \boldsymbol{x} - \boldsymbol{y}_m) + \nabla \mathsf{R}(\boldsymbol{x})$$

for $m = 0, \ldots, M-1$, leading (empirically) to M-times acceleration in run time (for early iterations). The matrices $\boldsymbol{A}_m$, $\boldsymbol{W}_m$ and $\boldsymbol{y}_m$ are submatrices of $\boldsymbol{A}$, $\boldsymbol{W}$ and $\boldsymbol{y}$ corresponding to the mth subset of N_d projection data. Note that the computation of $\nabla \mathsf{R}(\boldsymbol{x})$ is small compared to $\nabla \mathsf{L}(\boldsymbol{x})$.

Table I summarizes the momentum-accelerated [14] version of the OS-SQS method that shows significant (heuristic)

1: Initialize $\boldsymbol{x}^{(0)} = \boldsymbol{z}^{(0)}$ by FBP, $t^{(0)} = 1$, and compute $\boldsymbol{D}$.
2: **for** $n = 0, 1, \ldots$
3: **for** $m = 0, 1, \ldots, M-1$
4: $\quad k = nM + m$
5: $\quad t^{(k+1)} = \frac{1}{2}\left(1 + \sqrt{1 + 4\left[t^{(k)}\right]^2}\right)$
6: $\quad \boldsymbol{g}_m^{(k)} = M \nabla \Psi_m(\boldsymbol{z}^{(\frac{k}{M})})$
7: $\quad \boldsymbol{x}^{(\frac{k+1}{M})} = \left[\boldsymbol{z}^{(\frac{k}{M})} - \boldsymbol{D}^{-1}\boldsymbol{g}_m^{(k)}\right]_+$
8: $\quad \boldsymbol{z}^{(\frac{k+1}{M})} = \boldsymbol{x}^{(\frac{k+1}{M})} + \frac{t^{(k)}-1}{t^{(k+1)}}\left(\boldsymbol{x}^{(\frac{k+1}{M})} - \boldsymbol{x}^{(\frac{k}{M})}\right)$
9: **end for**
10: **end for**

TABLE I: OS-SQS-mom method

acceleration [4] and reduces to the ordinary OS-SQS when $t^{(k)} = 1$ for all k. This algorithm is (heuristically) expected to minimize the cost function with the fast rate $O(1/(nM)^2)$ in early iterations[1] when the approximation (5) is appropriate [4].

This OS-SQS-mom in Table I is well-suited for a parallelizable computing platforms that share memory. However, a typical implementation of OS-SQS-mom may be inefficient for distributed computing resources because the gradient computation $\nabla \Psi_m(\boldsymbol{x})$ requires expensive global communications across nodes [10]. The next section reviews the distributed version of OS-SQS-mom in [10] and notes some drawbacks.

III. Previous distributed OS approach

Table II summarizes our previous distributed OS-SQS-mom approach [10]. Because global communication for the gradient computation $\nabla \Psi_m(\boldsymbol{x})$ is expensive, we divided the problem (1) into S sub-problems (slabs of contiguous z-slices), and assigned each sub-problem to a cluster consisting of multiple nodes. For example in [10], a 320-slice helical CT scan was partitioned into 5 slabs of 64 slices, and each slab was reconstructed independently. However, because of the "long object" problem in helical CT, to reconstruct a slab of slices of interest one must include additional "padding" end slices. For example in [10], for each slab of 64 slices we also estimated an additional 32 slices on each end of each slab; these extra slices overlap with neighboring slabs and are eventually discarded, reducing efficiency and limiting how large S can be. (In [10], each cluster was assigned 10 nodes, summing up to $B = 50$ nodes total for $S = 5$ slabs.)

Mathematically, we divided an image $\boldsymbol{x}$ into S (overlapping) sub-images $\{\boldsymbol{x}_s\}$ and solved each sub-problem separately:

$$\hat{\boldsymbol{x}}_s = \underset{\boldsymbol{x}_s \succeq 0}{\arg\min} \left\{ \breve{\Psi}_{\cdot,s}(\boldsymbol{x}_s) \triangleq \frac{1}{2}||\boldsymbol{y} - \boldsymbol{A}_{\cdot,s}\boldsymbol{x}_s||^2_{\boldsymbol{W}} + \breve{\mathsf{R}}_s(\boldsymbol{x}_s) \right\}$$

for $s = 1, \ldots, S$, where $\boldsymbol{A}_{\cdot,s} \in \mathbb{R}^{N_d \times |\boldsymbol{x}_s|}$ is the system matrix and $\breve{\mathsf{R}}_s(\boldsymbol{x}_s)$ is the regularizer for each s. The solutions $\{\hat{\boldsymbol{x}}_s\}$ may differ from (1), particularly near the slab boundaries.

Even though slab partitioning reduces communication, the communication within each cluster s required after each iteration (k) remained a bottle-neck [10]. In this paper, we propose to adapt the BSS technique [12] that can reduce the global communications required for gradient computation.

[1]The convergence and stability of the OS-SQS-mom algorithm in Table I is discussed in detail in [4]. In addition, we recently developed another version of OS-SQS-mom that converges twice faster than Table I [15], [16].

1: Initialize $\boldsymbol{x}^{(0)} = \boldsymbol{z}^{(0)}$ by FBP, $t^{(0)} = 1$, and compute $\boldsymbol{D}$.
2: Distribute image $\boldsymbol{x}^{(0)}$ and data $\boldsymbol{y}$ into S clusters.
3: **for** $n = 0, 1, \ldots$
4: **for** $m = 0, 1, \ldots, M-1$
5: $\quad k = nM + m$
6: $\quad t^{(k+1)} = \frac{1}{2}\left(1 + \sqrt{1 + 4\left[t^{(k)}\right]^2}\right)$
7: $\quad$ **for** $s = 1, \ldots, S$ **simultaneously**
8: $\quad\quad \boldsymbol{g}_{m,s}^{(k)} = M \nabla \breve{\Psi}_{m,s}(\boldsymbol{z}_s^{(\frac{k}{M})})$, by communicating $\boldsymbol{W}_m(\boldsymbol{A}_{m,s}\boldsymbol{z}_s^{(\frac{k}{M})} - \boldsymbol{y}_m)$ within cluster s.
9: $\quad\quad \boldsymbol{x}_s^{(\frac{k+1}{M})} = \left[\boldsymbol{z}_s^{(\frac{k}{M})} - \boldsymbol{D}_s^{-1}\boldsymbol{g}_{m,s}^{(k)}\right]_+$
10: $\quad\quad \boldsymbol{z}_s^{(\frac{k+1}{M})} = \boldsymbol{x}_s^{(\frac{k+1}{M})} + \frac{t^{(k)}-1}{t^{(k+1)}}\left(\boldsymbol{x}_s^{(\frac{k+1}{M})} - \boldsymbol{x}_s^{(\frac{k}{M})}\right)$
11: $\quad\quad$ Communicate $\boldsymbol{z}_s^{(\frac{k+1}{M})}$ within cluster s.
12: $\quad$ **end for**
13: **end for**
14: **end for**

TABLE II: Distributed OS-SQS-mom method

IV. Block-separable surrogate

To reduce communication, we would like each node to perform multiple iterations before synchronizing, instead of communicating for every (subset) gradient computation. We also want the algorithm to be guaranteed to converge in its one-subset ($M = 1$) version. The BSS approach in [12], previously applied to image restoration, provides block-separable surrogates where each node can minimize its (non-overlapping) block $\boldsymbol{x}_b$ independently. The BSS for B blocks (nodes) has the form:

$$\begin{aligned} \Psi(\boldsymbol{x}) &\le \phi_{\mathrm{BSS}}(\boldsymbol{x}; \boldsymbol{x}^{(n)}) \\ &\triangleq \sum_{b=1}^{B} \left\{ \phi_{\mathrm{BSS}}^{\mathsf{L},\mathrm{b}}(\boldsymbol{x}_b; \boldsymbol{x}^{(n)}) + \phi_{\mathrm{BSS}}^{\mathsf{R},\mathrm{b}}(\boldsymbol{x}_b; \boldsymbol{x}^{(n)}) \right\}, \end{aligned} \tag{6}$$

where the BSS surrogate of the quadratic $\mathsf{L}(\boldsymbol{x})$ is defined as

$$\begin{aligned} \phi_{\mathrm{BSS}}^{\mathsf{L},\mathrm{b}}(\boldsymbol{x}_b; \boldsymbol{x}^{(n)}) \triangleq C_b^{(n)} + (\boldsymbol{x}_b - \boldsymbol{x}_b^{(n)})^\top \nabla_b \mathsf{L}(\boldsymbol{x}^{(n)}) \\ + \frac{1}{2}||\boldsymbol{x}_b - \boldsymbol{x}_b^{(n)}||^2_{\boldsymbol{H}_b^{\mathsf{L}}} \end{aligned} \tag{7}$$

for $b = 1, \ldots, B$, where $\mathsf{L}(\boldsymbol{x}^{(n)}) = \sum_{b=1}^{B} C_b^{(n)}$. A natural choice for the majorizer Hessian $\boldsymbol{H}_b^{\mathsf{L}}$ [12] is

$$\boldsymbol{H}_b^{\mathsf{L}} \triangleq \boldsymbol{A}_{\cdot,b}^\top \boldsymbol{W} \boldsymbol{\Lambda}^b \boldsymbol{A}_{\cdot,b}, \tag{8}$$

where $\boldsymbol{A}_{\cdot,b} \in \mathbb{R}^{N_d \times |\boldsymbol{x}_b|}$ is the columns of the system matrix corresponding to the slices in the bth block, and

$$\boldsymbol{\Lambda}^b \triangleq \mathsf{diag}\{\boldsymbol{A}\boldsymbol{1}\}\,\mathsf{diag}^{-1}\{\boldsymbol{A}_{\cdot,b}\boldsymbol{1}_b\} \tag{9}$$

for $b = 1, \ldots, B$. The definition of the BSS surrogate $\phi_{\mathrm{BSS}}^{\mathsf{R},\mathrm{b}}(\boldsymbol{x}_b; \boldsymbol{x}^{(n)})$ for $\mathsf{R}(\boldsymbol{x})$ is omitted for brevity.

Since $\nabla^2 \Psi(\boldsymbol{x}) = \boldsymbol{A}^\top \boldsymbol{W} \boldsymbol{A} + \nabla^2 \mathsf{R}(\boldsymbol{x})$ will be somewhat block band-diagonal for helical scans, we expect a block diagonal majorizer $\nabla^2 \phi_{\mathrm{BSS}}(\boldsymbol{x}; \boldsymbol{x}^{(n)})$ of BSS to be closer to $\nabla^2 \Psi(\boldsymbol{x})$ than a diagonal majorizer $\boldsymbol{D}$, *i.e.*,

$$\nabla^2 \Psi(\boldsymbol{x}) \precsim \nabla^2 \phi_{\mathrm{BSS}}(\boldsymbol{x}; \boldsymbol{x}^{(n)}) \precnsim \boldsymbol{D}. \tag{10}$$

However, for large B providing massively distributed computation, the difference between $\nabla^2 \Psi(\boldsymbol{x})$ and $\nabla^2 \phi_{\mathrm{BSS}}(\boldsymbol{x}; \boldsymbol{x}^{(n)})$

may become undesirably large. Therefore, we study the trade-off regarding the choice of B in the result section.

Next we must select an algorithm for minimizing the BSS majorizer (6). For simplicity we consider OS-SQS[2]-mom in this paper, but we plan to investigate other choices [5], [6] in the future. The next section describes the proposed algorithm that iteratively minimizes the BSS surrogate function node-independently using OS-SQS-mom.

V. Proposed distributed block-separable surrogate algorithm

Table III summarizes the proposed distributed BSS algorithm that guarantees monotonic descent update and convergence for the one-subset ($M=1$) version. The key difference between the two algorithms in Tables II and III is that the latter does not require communication when computing the gradient and image transfer after every update, unlike the former.

Except for the communication time (and memory issue), both Tables II and III require fairly similar computational cost per iteration, because the (mth subset) gradient in Table III is computed efficiently as

$$\begin{aligned} \boldsymbol{g}_{m,b}^{(l)} =& M \boldsymbol{A}_{m,b}^{\top} \boldsymbol{W}_m \\ & \times \left\{ \boldsymbol{A}_m \boldsymbol{z}^{(0)} - \boldsymbol{y}_m + \boldsymbol{\Lambda}_m^{\mathrm{b}} \boldsymbol{A}_{m,b} (\boldsymbol{z}_b^{(\frac{l}{M})} - \boldsymbol{z}_b^{(0)}) \right\} \\ & + \nabla \phi_{\mathrm{BSS}}^{\mathrm{R,b}} (\boldsymbol{z}_b^{(\frac{l}{M})}; \boldsymbol{z}_b^{(0)}), \end{aligned} \tag{11}$$

requiring one (mth subset) forward and back projection similar to the computation of $\boldsymbol{g}_{m,s}^{(k)}$ in line 8 of Table II. Here, we use two-input forward projector that simultaneously computes $\boldsymbol{A}_{m,b}\boldsymbol{z}^{(0)}$ and $\boldsymbol{A}_{m,b}(\boldsymbol{z}_b^{(\frac{l}{M})} - \boldsymbol{z}_b^{(0)})$ in (11) with small overhead [3]. For efficiency, we precompute the matrix $\boldsymbol{\Lambda}^b$ in (9) simultaneously with $\boldsymbol{D}$ in (4).

VI. Results

We simulated the performance of the proposed BSS algorithm in Table III, and compared it with OS-SQS-mom in Table I (or Table II for a single-slab), using a $256 \times 256 \times 160$ XCAT phantom image [17] scanned by one turn of a simulated helical trajectory (pitch 1.0) with total 492 projection views and 444×32 detector size. (The 160-slice image includes 32 extra slices padded on each end to deal with the "long object" problem.) Fig. 1 illustrates the initial FBP image $\boldsymbol{x}^{(0)}$ and the converged image $\hat{\boldsymbol{x}}$. Although our goal is to evaluate the efficiency of the algorithm on a multi-node platform, this paper reports only a preliminary convergence result versus "iteration" and predicted "cluster time" based on the component run-time statistics in [10].

Fig. 2 shows convergence plots of the proposed BSS method with the choices of $B = 2, 5, 10$ and $L/M = 2, 5, 10, 20$ for $M = 12$ subsets, compared to OS-SQS-mom in Table I for $M = 12$. Convergence speed is measured by the root mean square difference (RMSD) between $\boldsymbol{x}^{(n)}$ and the converged image $\hat{\boldsymbol{x}}$ in Hounsfield Units (HU), *i.e.*, $\mathrm{RMSD}^{(n)} = (\boldsymbol{x}^{(n)} - \hat{\boldsymbol{x}})/\sqrt{N_p}$ [HU], versus iteration (that counts one iteration after M subset iterations). Fig. 2(a) shows

[2]A standard choice of a diagonal majorizing matrix $\boldsymbol{D}_b \in \mathbb{R}^{|\boldsymbol{x}_b| \times |\boldsymbol{x}_b|}$ of SQS for minimizing the BSS surrogate (6) has values that are the same as $\boldsymbol{D}$ in (4) for each block b.

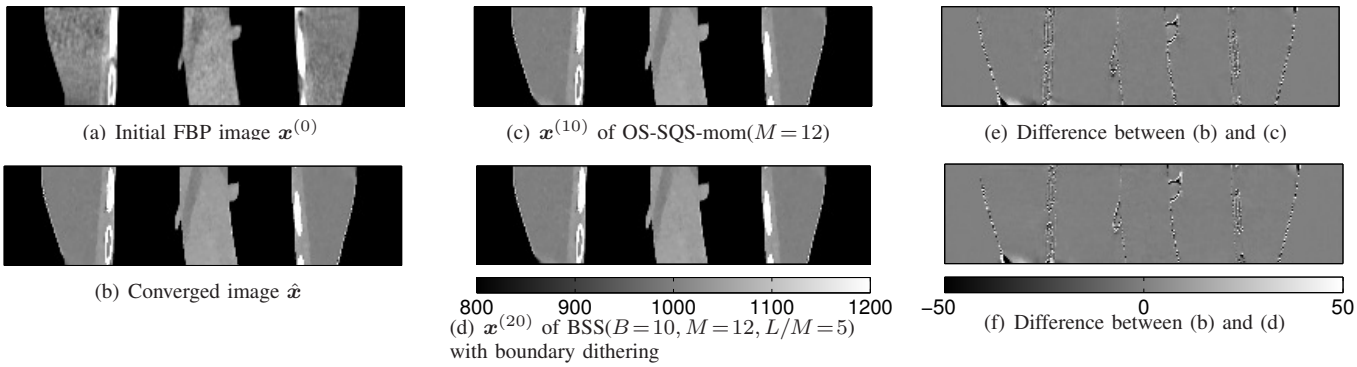
1: Initialize $\tilde{\boldsymbol{x}}^{(0)}$ by FBP, and compute $\boldsymbol{D}$.
2: Distribute image $\tilde{\boldsymbol{x}}^{(0)}$ and data $\boldsymbol{y}$ into B nodes.
3: **for** $n = 0, 1, \ldots$
4: Minimize $\phi_{\mathrm{BSS}}(\boldsymbol{x}; \tilde{\boldsymbol{x}}^{(n)})$ in (6) using L sub-iterations of OS-SQS-mom.
1) Initialize $\boldsymbol{x}^{(0)} = \boldsymbol{z}^{(0)}$ by $\tilde{\boldsymbol{x}}^{(n)}$, and $t^{(0)} = 1$.
2) **for** $l = 0, 1, \ldots, L-1$
3) $m = l \bmod M$
4) $t^{(l+1)} = \frac{1}{2}\left(1 + \sqrt{1 + 4\left[t^{(l)}\right]^2}\right)$
5) **for** $b = 1, \ldots, B$ **simultaneously**
6) $\boldsymbol{g}_{m,b}^{(l)} = M \nabla_b \phi_{\mathrm{BSS},m}(\boldsymbol{z}^{(\frac{l}{M})}; \boldsymbol{z}^{(0)})$
7) $\boldsymbol{x}_b^{(\frac{l+1}{M})} = \left[\boldsymbol{z}_b^{(\frac{l}{M})} - \boldsymbol{D}_b^{-1} \boldsymbol{g}_{m,b}^{(l)}\right]_+$
8) $\boldsymbol{z}_b^{(\frac{l+1}{M})} = \boldsymbol{x}_b^{(\frac{l+1}{M})} + \frac{t^{(l)}-1}{t^{(l+1)}}\left(\boldsymbol{x}_b^{(\frac{l+1}{M})} - \boldsymbol{x}_b^{(\frac{l}{M})}\right)$
9) **end for**
10) **end for**
11) $\tilde{\boldsymbol{x}}^{(n+1)} = \boldsymbol{x}^{(\frac{L}{M})}$
5: Communicate $\tilde{\boldsymbol{x}}^{(n+1)}$.
6: **end for**

TABLE III: Proposed distributed BSS method with OS-SQS-mom

that using many blocks (B) for massive parallelization leads to loose BSS surrogates, exhibiting slow convergence speed (vs. iteration). The results also illustrate that letting sub-iterations L to be large initially may be preferable to fully benefit from the fast initial speed of OS-SQS-mom, while the algorithm might reach near the optimum of BSS after only a few sub-iterations and have minimal updates for the rest of L sub-iterations. For example in Fig. 2(b), the RMSD plot of BSS($B=10, L/M=20$) algorithm does not decrease after 10 (outer) iterations.

Fig. 2(c) plots RMSD versus predicted cluster time that considers the communication time analysis for 10-nodes in [10]. Fig. 5 in [10] illustrates that using 10 nodes (for a single 128-slice slab and 12 subsets in Table II) reduces the computation time of the forward and back projection by about $6\times$ (not an ideal $10\times$) per iteration, while undesirably adding the communication time that is equivalent to the computation time of the projection per iteration, leading to overall about only $3\times$ acceleration from using 10 nodes. Hypothetically, Fig. 2(c) illustrates that our proposed method converges faster than OS-SQS-mom in (predicted) run time initially. Fig. 2(c) suggests we must further improve the BSS method to maintain the initial acceleration of BSS (in run time) in later iterations so that the BSS method stays faster than the standard OS-SQS-mom overall.

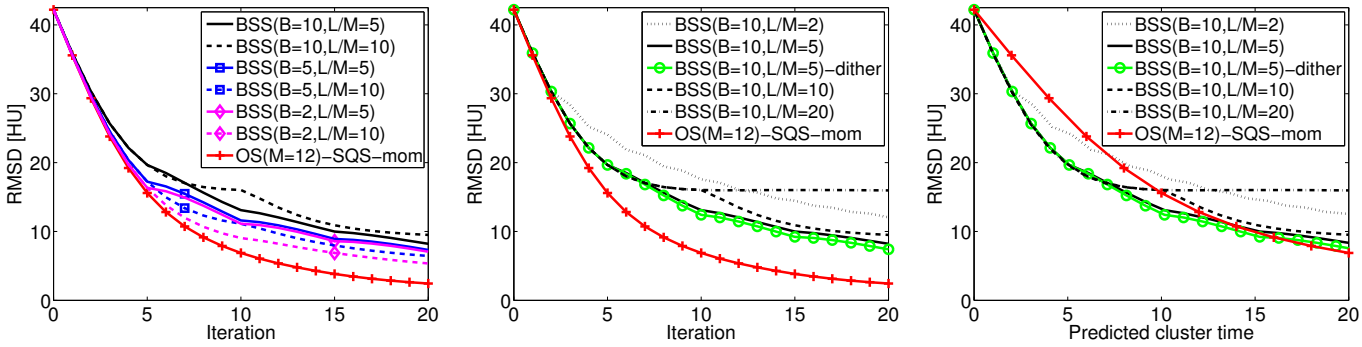
Figs. 1(c) and 1(d) show reconstructed images from 10 iterations of OS-SQS-mom and 20 iterations of BSS($B = 10, L/M = 5$) that are expected to have similar computation time for 10-nodes platform, and their difference images from the converged image are given for visibility in Fig. 1. The resulting image after a finite number of iterations of the BSS showed incomplete convergence near block boundaries. (Results not shown.) This is because image information of neighboring blocks provided to each node that is mostly used for voxel updates near block boundaries becomes outdated due to reduced communication in the BSS method. Thus, we modified the method by dithering the block boundaries for



(a) Initial FBP image $\boldsymbol{x}^{(0)}$ (c) $\boldsymbol{x}^{(10)}$ of OS-SQS-mom($M=12$) (e) Difference between (b) and (c)

(b) Converged image $\hat{\boldsymbol{x}}$ (d) $\boldsymbol{x}^{(20)}$ of BSS($B=10$, $M=12$, $L/M=5$) with boundary dithering (f) Difference between (b) and (d)

Fig. 1: 3D XCAT simulation: Center coronal plane of (a) initial FBP image $\boldsymbol{x}^{(0)}$, (b) converged image $\hat{\boldsymbol{x}}$, and reconstructed images by (c) 10 iterations of OS-SQS-mom($M=12$) and (d) 20 iterations of BSS($B=10$, $M=12$, $L/M=5$) with boundary dithering. Difference images (e): between (b) and (c), and (f): between (b) and (d) are shown for visibility. (Padded end slices are cropped.)



(a) vs. iteration (b) vs. iteration (c) vs. predicted cluster time

Fig. 2: RMSD plots of the OS-SQS-mom and the proposed BSS method (a-b) vs. iteration, and (c) vs. predicted cluster time.

each (n-th) BSS surrogate to improve convergence speed near block boundaries, leading to the image illustrated in Fig. 1(d). (Details omitted.) Fig. 1(d) is close to Fig. 1(c) (having similar RMSD values after comparable predicted cluster time) and does not have any block discontinuity unlike the plain BSS. Dithering significantly improves convergence speed for slices near the block boundaries, but because the boundaries are a small fraction of the image volume the RMSD values in Fig. 2 change only slightly.

VII. Conclusion

We introduced a block-separable surrogate technique to reduce communication time in a potentially more efficient distributed OS-SQS-mom approach. Preliminary predicted timing results suggest that the early convergence speed of the proposed BSS algorithm can be faster than that of the conventional OS-SQS-mom method, but further refinements are needed for overall acceleration. In addition, orchestrating the memory communication and computation on a real distributed platform is needed as future work.

References

[1] J-B. Thibault, K. Sauer, C. Bouman, and J. Hsieh, "A three-dimensional statistical approach to improved image quality for multi-slice helical CT," *Med. Phys.*, vol. 34, no. 11, pp. 4526–44, Nov. 2007.

[2] H. Erdoğan and J. A. Fessler, "Ordered subsets algorithms for transmission tomography," *Phys. Med. Biol.*, vol. 44, no. 11, pp. 2835–51, Nov. 1999.

[3] D. Kim, D. Pal, J-B. Thibault, and J. A. Fessler, "Accelerating ordered subsets image reconstruction for X-ray CT using spatially non-uniform optimization transfer," *IEEE Trans. Med. Imag.*, vol. 32, no. 11, pp. 1965–78, Nov. 2013.

[4] D. Kim, S. Ramani, and J. A. Fessler, "Combining ordered subsets and momentum for accelerated X-ray CT image reconstruction," *IEEE Trans. Med. Imag.*, vol. 34, no. 1, pp. 167–78, Jan. 2015.

[5] H. Nien and J. A. Fessler, "Fast X-ray CT image reconstruction using a linearized augmented Lagrangian method with ordered subsets," *IEEE Trans. Med. Imag.*, vol. 34, no. 2, pp. 388–99, Feb. 2015.

[6] M. G. McGaffin and J. A. Fessler, "Duality-based projection-domain tomography solver for splitting-based X-ray CT reconstruction," in *Proc. 3rd Intl. Mtg. on image formation in X-ray CT*, 2014, pp. 359–62.

[7] B. Meng, G. Pratx, and L. Xing, "Ultrafast and scalable cone-beam CT reconstruction using MapReduce in a cloud computing environment," *Med. Phys.*, vol. 38, no. 12, pp. 6603–9, Dec. 2011.

[8] S. Srivastava, A. R. Rao, and V. Sheinin, "Accelerating statistical image reconstruction algorithms for fan-beam x-ray CT using cloud computing," in *Proc. SPIE 7961 Medical Imaging 2011: Phys. Med. Im.*, 2011, p. 796134.

[9] J. Gregor, "Distributed multi-core implementation of SIRT with vectorized matrix kernel for micro-CT," in *Proc. Intl. Mtg. on Fully 3D Image Recon. in Rad. and Nuc. Med*, 2011, pp. 64–7.

[10] J. M. Rosen, J. Wu, T. F. Wenisch, and J. A. Fessler, "Iterative helical CT reconstruction in the cloud for ten dollars in five minutes," in *Proc. Intl. Mtg. on Fully 3D Image Recon. in Rad. and Nuc. Med*, 2013, pp. 241–4.

[11] A. S. Wang, J. W. Stayman, . Y. Otake, G. Kleinszig, S. Vogt, and J. H. Siewerdsen, "Nesterov's method for accelerated penalized-likelihood statistical reconstruction for C-arm cone-beam CT," in *Proc. 3rd Intl. Mtg. on image formation in X-ray CT*, 2014, pp. 409–13.

[12] S. Sotthivirat and J. A. Fessler, "Image recovery using partitioned-separable paraboloidal surrogate coordinate ascent algorithms," *IEEE Trans. Im. Proc.*, vol. 11, no. 3, pp. 306–17, Mar. 2002.

[13] Y. Long, J. A. Fessler, and J. M. Balter, "3D forward and back-projection for X-ray CT using separable footprints," *IEEE Trans. Med. Imag.*, vol. 29, no. 11, pp. 1839–50, Nov. 2010.

[14] Y. Nesterov, *Introductory lectures on convex optimization: A basic course*, Kluwer, 2004.

[15] D. Kim and J. A. Fessler, "Optimized first-order methods for smooth convex minimization," 2014, arxiv 1406.5468.

[16] D. Kim and J. A. Fessler, "Optimized momentum steps for accelerating X-ray CT ordered subsets image reconstruction," in *Proc. 3rd Intl. Mtg. on image formation in X-ray CT*, 2014, pp. 103–6.

[17] W. P. Segars, M. Mahesh, T. J. Beck, E. C. Frey, and B. M. W. Tsui, "Realistic CT simulation using the 4D XCAT phantom," *Med. Phys.*, vol. 35, no. 8, pp. 3800–8, Aug. 2008.