Fast GPU-Driven Model-Based X-Ray CT Image Reconstruction via Alternating Dual Updates

Madison G. McGaffin Jeffrey A. Fessler

***Abstract*—Model-based image reconstruction (MBIR) methods for X-ray CT reconstruction can improve image quality and reduce patient X-ray dose. These methods produce images by solving high-dimensional, statistically motivated numerical optimization problems, but unfortunately the high computational costs of solving these problems have kept MBIR algorithms from reaching ubiquity in the clinic. In this paper, we present an X-ray CT image reconstruction algorithm that uses duality and group coordinate ascent to alternately perform efficient tomography and denoising updates. The algorithm can handle non-smooth regularizers like anisotropic total variation (TV) and stores only two image-sized vectors on the GPU. Preliminary experiments show the algorithm converges very quickly in time.**

I. Introduction

Consider the following model-based X-ray image reconstruction (MBIR) problem:[1]

$$\widehat{\mathbf{x}} = \operatorname*{argmin}_{\mathbf{x}} \ \mathsf{L}(\mathbf{A}\mathbf{x}) + \mathsf{R}(\mathbf{C}\mathbf{x}) \tag{1}$$

with X-ray CT system matrix $\mathbf{A}$, 3D finite differencing matrix $\mathbf{C}$, log-likelihood data-fit term L and edge-preserving regularizer R. Both L and R are separable sums of convex functions:

$$\mathsf{L}(\mathbf{A}\mathbf{x}) = \sum_{i=1}^{M} l_i([\mathbf{A}\mathbf{x}]_i) \qquad \mathsf{R}(\mathbf{C}\mathbf{x}) = \sum_{k=1}^{N_d} r_k([\mathbf{C}\mathbf{x}]_k). \tag{2}$$

A common choice [14] for the data-fit term L is the Gaussian-inspired quadratic term $\mathsf{L}(\mathbf{A}\mathbf{x}) = \frac{1}{2}||\mathbf{A}\mathbf{x} - \mathbf{y}||^2_{\mathbf{W}}$ with noisy measurements $\mathbf{y}$ and diagonal matrix of positive statistical weights $\mathbf{W}$. The edge-preserving regularizer is conventionally a weighted sum of nonquadratically penalized finite differences,

$$r_k = \beta_k \psi([\mathbf{C}\mathbf{x}]_k) = \beta_k \psi(x_{k1} - x_{k2}), \tag{3}$$

with convex ψ and $\beta_k > 0$. This includes anisotropic total variation (TV) and many other noise-reducing regularizers.

The image $\widehat{\mathbf{x}}$ produced by solving (1) can be higher-quality than images produced with conventional filtered-backprojection methods, and MBIR algorithms can produce diagnostically-useful images at lower doses than conventional CT reconstruction algorithms. Unfortunately, solving MBIR's optimization problem is computationally challenging and time-consuming, especially compared with conventional methods. New algorithms, distributed computing, and highly parallel hardware may provide a solution to making MBIR's advantages ubiquitous in the clinic. In this paper we focus on the first and last of these approaches, and propose a technique that combines the high parallelism available in modern graphics processing units (GPUs) with structure-exploiting algorithms to rapidly solve the MBIR optimization problem (1).

The next section presents the mathematical framework for the proposed algorithm, and Section III describes its implementation on the GPU. Section IV contains some experimental results, and Section V has some concluding remarks.

M. G. McGaffin and J. A. Fessler are with the Dept. of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109 USA. (email: `mcgaffin@umich.edu`, `fessler@umich.edu`)

Supported in part by NIH grant R01-HL-098686 and equipment donations from Intel Corporation.

[1]This paper describes this simpler problem without a nonnegativity constraint [14], but the method presented be easily extended to accommodate one.

II. Algorithm framework

Let $\mathbf{x}^{(n)} \in \mathbb{R}^N$ be an estimate of $\widehat{\mathbf{x}}$. Customarily, $\mathbf{x}^{(0)}$, the initial image, is generated using a fast filtered backprojection method. The algorithm presented in this paper has two steps that it alternates between until convergence:

- identify an update $\mathbf{d}^{(n)}$ to $\mathbf{x}^{(n)}$; and then
- apply the update, generating $\mathbf{x}^{(n+1)} = \mathbf{x}^{(n)} + \mathbf{d}^{(n)}$.

We perform the first step by approximately solving the following optimization problem:

$$\mathbf{d}^{(n)} = \operatorname*{argmin}_{\mathbf{d}} \ \mathsf{L}\Big(\mathbf{A}\mathbf{x}^{(n)} + \mathbf{A}\mathbf{d}\Big) + \mathsf{R}\Big(\mathbf{C}\mathbf{x}^{(n)} + \mathbf{C}\mathbf{d}\Big) + \frac{\epsilon}{2}||\mathbf{d}||_2^2, \tag{4}$$

with $\epsilon > 0$. Solving (4) is nearly as challenging as solving the original optimization problem (1), but the additional $\frac{\epsilon}{2}||\cdot||_2^2$ term allows us to use the following duality-based approach.

A. Duality approach

Let L^* and R^* be the convex conjugates [2, pg. 91] of L and R, respectively:

$$\mathsf{L}^*(\mathbf{u}) = \sum_{i=1}^{M} l_i^*(u_i), \qquad \mathsf{R}^*(\mathbf{v}) = \sum_{k=1}^{N_d} r_k^*(v_k), \tag{5}$$

where l_i^* are r_k^* are the convex conjugates of l_i and r_k, respectively. Rewrite (4) using the biconjugate property of the convex functions L and R:

$$\mathbf{d}^{(n)} = \operatorname*{argmin}_{\mathbf{d}} \ \max_{\mathbf{u},\mathbf{v}} \mathbf{u}^\mathsf{T}\mathbf{A}\Big(\mathbf{x}^{(n)} + \mathbf{d}\Big) + \mathbf{v}^\mathsf{T}\mathbf{C}\Big(\mathbf{x}^{(n)} + \mathbf{d}\Big) - \mathsf{L}^*(\mathbf{u}) - \mathsf{R}^*(\mathbf{v}) + \frac{\epsilon}{2}||\mathbf{d}||_2^2. \tag{6}$$

After swapping the order of the "min" and the "max" in (6), we can easily solve for the update $\mathbf{d}^{(n)}$ in terms of $\mathbf{u}$ and $\mathbf{v}$:

$$\mathbf{d}^{(n)}(\mathbf{u},\mathbf{v}) = -\frac{1}{\epsilon}(\mathbf{A}^\mathsf{T}\mathbf{u} + \mathbf{C}^\mathsf{T}\mathbf{v}). \tag{7}$$

Plug (7) back into (6) to produce the dual problem:

$$\mathbf{u}^{(n)}, \mathbf{v}^{(n)} = \operatorname*{argmax}_{\mathbf{u},\mathbf{v}} \ \mathsf{D}^{(n)}(\mathbf{u},\mathbf{v}), \qquad \text{where} \tag{8}$$

$$\begin{aligned}\mathsf{D}^{(n)}(\mathbf{u},\mathbf{v}) \triangleq & -\mathsf{L}^*(\mathbf{u}) - \mathsf{R}^*(\mathbf{v}) + \mathbf{u}^\mathsf{T}\mathbf{A}\mathbf{x}^{(n)} \\ & + \mathbf{v}^\mathsf{T}\mathbf{C}\mathbf{x}^{(n)} - \frac{1}{2\epsilon}||\mathbf{A}^\mathsf{T}\mathbf{u} + \mathbf{C}^\mathsf{T}\mathbf{v}||_2^2.\end{aligned} \tag{9}$$

We approximately solve (8) with a coordinate ascent method that is similar to stochastic dual coordinate ascent (SDCA) [12]. In each of N_{inner} inner iterations, we

- randomly choose a group of elements of either $\mathbf{u}$ or $\mathbf{v}$; and
- holding all other variables constant, update the selected group of variables to increase the dual function $\mathsf{D}^{(n)}(\mathbf{u},\mathbf{v})$.

This inner procedure is a simple, convergent group coordinate ascent algorithm. We call optimizations over elements of $\mathbf{u}$ "tomography updates" and over the elements of $\mathbf{v}$ "denoising updates." The next two subsections describe how to perform these updates.

B. Tomography updates

In [9], the authors found that a duality-based proximal technique can efficiently solve quadratic problems involving the CT system matrix $\mathbf{A}$. In 3D CT these problems are challenging to solve in the primal domain without ordered subsets (OS) approximations [1], and unlike OS, the duality-based approach is convergent. We use this approach here.

We update one view of $\mathbf{u}$ at a time, written $\mathbf{u}_\beta$. The corresponding CT projection operator, noisy data, and statistical weights are $\mathbf{A}_\beta$, $\mathbf{y}_\beta$ and $\mathbf{W}_\beta$, respectively. Tomography terms unrelated to $\mathbf{u}_\beta$ are indicated using the subscript "$\backslash\beta$", *e.g.*, $\mathbf{u}_{\backslash\beta}$.

Holding all variables other than $\mathbf{u}_\beta$ constant, we want to find $\mathbf{u}_\beta^+$ such that

$$\mathsf{D}^{(n)}\left(\mathbf{u}_\beta^+, \mathbf{u}_{\backslash\beta}, \mathbf{v}\right) \geq \mathsf{D}^{(n)}(\mathbf{u}_\beta, \mathbf{u}_{\backslash\beta}, \mathbf{v}), \tag{10}$$

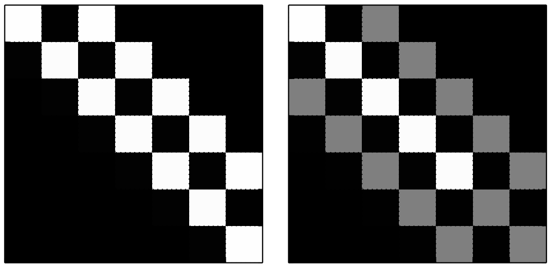
then update $\mathbf{u}_\beta \leftarrow \mathbf{u}_\beta^+$ in place. We find $\mathbf{u}_\beta^+$ as the maximizer of the minorizing surrogate function S_β for $\mathsf{D}^{(n)}$:

$$\mathbf{u}_\beta^+ = \operatorname*{argmax}_{\mathbf{u}_\beta} \ \mathsf{S}_\beta\left(\mathbf{u}_\beta; \mathbf{u}, \mathbf{v}, \mathbf{x}^{(n)}\right), \qquad \text{where} \tag{11}$$

$$\begin{aligned}\mathsf{S}_\beta\left(\mathbf{u}_\beta^+; \mathbf{u}, \mathbf{v}, \mathbf{x}^{(n)}\right) = & -\mathsf{L}_\beta^*\left(\mathbf{u}_\beta^+\right) - \frac{1}{2}\left|\left|\mathbf{u}_\beta^+ - \mathbf{u}_\beta\right|\right|_{\mathbf{M}_\beta}^2 \\ & + \left(\mathbf{u}_\beta^+\right)^\mathsf{T}\mathbf{A}_\beta\left(\mathbf{x}^{(n)} + \mathbf{d}^{(n)}(\mathbf{u},\mathbf{v})\right)\end{aligned} \tag{12}$$

Updating $\mathbf{u}_\beta^+$ in this way will increase the dual function $\mathsf{D}^{(n)}$ if $\mathbf{M}_\beta \succeq \mathbf{A}_\beta\mathbf{A}_\beta^\mathsf{T}$, *i.e.*, if all the eigenvalues of $\mathbf{M}_\beta - \mathbf{A}_\beta\mathbf{A}_\beta^\mathsf{T}$ are nonnegative. Finding the "tightest" so-called majorizer for $\mathbf{A}_\beta\mathbf{A}_\beta^\mathsf{T}$ is challenging, but the following diagonal matrix is relatively efficient to compute and useful in practice [1]:

$$\mathbf{M}_\beta = \operatorname*{diag}_i\left\{\left[\mathbf{A}_\beta\mathbf{A}_\beta^\mathsf{T}\mathbf{1}\right]_i\right\}. \tag{13}$$



(a) $|\mathbf{C}_d|$ (b) $|\mathbf{C}_d\mathbf{C}_d^\mathsf{T}|$

Fig. 1: Absolute values of entries of $\mathbf{C}_d$ and $\mathbf{C}_d\mathbf{C}_d^\mathsf{T}$ matrices. Only adjacent elements are coupled by $\mathbf{C}_d\mathbf{C}_d^\mathsf{T}$. Black entries are zero.

These $\mathbf{M}_\beta$ depend only on $\mathbf{A}$ (*i.e.*, they are independent of any patient data) so they can be precomputed. Storing $\mathbf{M}_\beta$ for all β takes the same amount of memory as the projection data $\mathbf{y}$.

When $\mathsf{L}(\mathbf{A}\mathbf{x}) = \frac{1}{2}||\mathbf{A}\mathbf{x} - \mathbf{y}||_\mathbf{W}^2$, the update (11) is

$$\begin{aligned}\mathbf{u}_\beta^+ = & \ \mathbf{u}_\beta + \epsilon\mathbf{W}_\beta(\epsilon\mathbf{I} + \mathbf{W}_\beta\mathbf{M}_\beta)^{-1} \\ & \left(\mathbf{A}_\beta\left(\mathbf{x}^{(n)} + \mathbf{d}^{(n)}(\mathbf{u},\mathbf{v})\right) - \mathbf{y}_\beta - \mathbf{u}_\beta\right).\end{aligned} \tag{14}$$

The major computational costs of implementing (14) are the one-view forward projection ($\mathbf{A}_\beta$), a series of diagonal operations, and a one-view backprojection to update $\mathbf{d}^{(n)}(\mathbf{u},\mathbf{v})$.

C. Denoising updates

The regularizer R couples neighboring pixels together through the finite differencing matrix $\mathbf{C}$. We previously exploited this property to update groups of pixels from different neighborhoods simultaneously, resulting in a fast denoising algorithm [10]. The dual function $\mathsf{D}^{(n)}$ (9) has a similar structure as a function of $\mathbf{v}$. Let D be the number of directions along which R penalizes voxel differences (*e.g.*, horizontal, vertical, axial, *etc.*); we write $\mathbf{v} = [\mathbf{v}_1^\mathsf{T} \cdots \mathbf{v}_D^\mathsf{T}]^\mathsf{T}$. Because R^* is separable (5), the only coupling between different elements of $\mathbf{v}$ comes from the quadratic term:

$$\begin{aligned}||\mathbf{A}^\mathsf{T}\mathbf{u} + \mathbf{C}^\mathsf{T}\mathbf{v}||^2 = & \begin{bmatrix}\mathbf{v}_1 \\ \vdots \\ \mathbf{v}_D\end{bmatrix}^\mathsf{T} \begin{bmatrix}\mathbf{C}_1\mathbf{C}_1^\mathsf{T} & \dots & \mathbf{C}_1\mathbf{C}_D^\mathsf{T} \\ \vdots & \ddots & \vdots \\ \mathbf{C}_D\mathbf{C}_1^\mathsf{T} & \dots & \mathbf{C}_D\mathbf{C}_D^\mathsf{T}\end{bmatrix} \begin{bmatrix}\mathbf{v}_1 \\ \vdots \\ \mathbf{v}_D\end{bmatrix} \\ & + 2\mathbf{v}^\mathsf{T}\mathbf{C}\mathbf{A}^\mathsf{T}\mathbf{u} + c(\mathbf{u}),\end{aligned} \tag{15}$$

where each $\mathbf{C}_d$ is a banded upper triangular finite differencing matrix in a single direction with 1s along the diagonal and -1s along an upper band.

Consider optimizing over one "direction" of differences at a time; *i.e.*, hold all variables except one $\mathbf{v}_d$ constant. The entries of $\mathbf{v}_d$ are coupled in the dual function by $\mathbf{C}_d\mathbf{C}_d^\mathsf{T}$. Figure 1 illustrates an example of $\mathbf{C}_d\mathbf{C}_d^\mathsf{T}$: only differences that are adjacent to one another along the dth direction are coupled by $\mathbf{C}_d\mathbf{C}_d^\mathsf{T}$. For example, if d corresponds to the horizontal direction, then the only elements of $\mathbf{v}_d$ coupled by $\mathbf{C}_d\mathbf{C}_d^\mathsf{T}$ are those horizontally adjacent to one another. If we first optimize over the "even" entries of $\mathbf{v}_d$, and then optimize over the "odd" entries of $\mathbf{v}_d$, each of these half-$\mathbf{v}_d$ optimizations will be over decoupled variables. Consequently, they can be performed independently and in parallel.

Let $\mathbf{v}_g$ be one such half-$\mathbf{v}_d$ group and $\mathbf{C}_g$ be the matrix of corresponding rows from $\mathbf{C}_d$. We find the updated $\mathbf{v}_g^+$ by maximizing the following separable surrogate for $\mathsf{D}^{(n)}$:

$$\mathbf{v}_g^+ = \underset{\mathbf{v}_g}{\operatorname{argmax}} \; \mathsf{S}_g\left(\mathbf{v}_g; \mathbf{u}, \mathbf{v}, \mathbf{x}^{(n)}\right), \tag{16}$$

$$\mathsf{S}_g\left(\mathbf{v}_g^+; \mathbf{u}, \mathbf{v}, \mathbf{x}^{(n)}\right) = \sum_{v_k^+ \in \mathbf{v}_g} \mathrm{s}_k\left(v_k^+; \mathbf{u}, \mathbf{v}, \mathbf{x}^{(n)}\right), \tag{17}$$

$$\begin{aligned} \mathrm{s}_k\left(v_k^+; \mathbf{u}, \mathbf{v}, \mathbf{x}^{(n)}\right) &= v_k^+ \left[\mathbf{C}_g\left(\mathbf{x}^{(n)} + \mathbf{d}^{(n)}(\mathbf{u}, \mathbf{v})\right)\right]_k \\ &\quad - r_k^*(v_k^+) - \frac{1}{2\epsilon} \|\mathbf{C}_g \mathbf{e}_k\|^2 \left(v_k^+ - v_k\right)^2, \end{aligned} \tag{18}$$

where $\mathbf{e}_k$ is the kth elementary basis vector.

If the dual function r_k^* has a convenient closed form, then the elements of $\mathbf{v}_g^+$ can be computed from (18). While this is true for the absolute value and quadratic potential functions this is often not the case. Instead, if the potential function ψ has a closed form shrinkage function, we maximize (18) by again invoking duality:

$$v_k^+ = v_k + \frac{\epsilon}{2}(q_k - z_k), \tag{19}$$

$$q_k = \left[\mathbf{C}_g\left(\mathbf{x}^{(n)} + \mathbf{d}^{(n)}(\mathbf{u}, \mathbf{v})\right)\right]_k, \tag{20}$$

$$z_k = \underset{z}{\operatorname{argmin}} \; \frac{\epsilon}{4}\left(z - \left(q_k + \frac{2}{\epsilon} v_k\right)\right)^2 + r_k(z). \tag{21}$$

This approach is more convenient for the many potential functions with closed-form shrinkage operators but complicated dual functions, such as the Fair potential [7].

III. GPU implementation

Memory is a scarce resource on the GPU, and transfers between the GPU's on-board memory and the host computer's memory are relatively expensive. For reasonably-sized problems, some data will need to be transferred between the host and the GPU. It is up to the algorithm designer to decide what data should remain on the GPU, what data can be transferred between the GPU and the host, and how to hide the latency of these transfers (normally by simultaneously performing computations on the GPU). A naïve design (*e.g.*, the split-Bregman algorithm [4] in [10]) can be unacceptably slow due to data transfer delays.

We implemented the algorithm in this paper by storing two image-sized vectors on the GPU:

- a vector $\mathbf{g}$ reflecting the current value of $\mathbf{x}^{(n+1)}$ if the inner iterations were terminated (*cf.*, (7))

$$\mathbf{g} = \mathbf{x}^{(n)} + \mathbf{d}^{(n)}(\mathbf{u}, \mathbf{v}) = \mathbf{x}^{(n)} - \frac{1}{\epsilon}(\mathbf{A}^\mathsf{T}\mathbf{u} + \mathbf{C}^\mathsf{T}\mathbf{v}),$$

- and a vector to store $\mathbf{v}_d$ when updating a group of the regularizer dual variables.

We hide the latency of transferring $\mathbf{v}_d$ to and from the GPU by performing tomography updates (14) between starting the transfer of $\mathbf{v}_d$ to the GPU, updating $\mathbf{v}_d$, and transferring $\mathbf{v}_d$ back to the host. See Figure 2.

```
Loop $N_{\text{inner}}$ times:
  1) Choose regularizer direction $d$ at random, and enqueue transfer of $\mathbf{v}_d$ to GPU.
  2) Perform $N_{\text{tomo}}$ tomography updates with randomly selected views.
  3) Update $\mathbf{v}_d$, then begin transfer of $\mathbf{v}_d$ back to the host computer.
  4) Perform $N_{\text{tomo}}$ tomography updates with randomly selected views.
```

Fig. 2: Implementation of algorithm inner loop that hides the cost of transferring $\mathbf{v}_d$ by interleaving transfers with tomography updates.

After N_{inner} iterations, we update $\mathbf{x}^{(n)}$. A simple way to do this is by simply setting the buffers storing $\mathbf{v}$ and $\mathbf{u}$ to zero! This restarts the update search problem (4) from $\mathbf{d} = \mathbf{0}$, and the GPU vector $\mathbf{g}$ is equal to $\mathbf{x}^{(n+1)}$. However, we find it useful to "warm-start" the next update search with the current values of $\mathbf{u}^{(n)}$ and $\mathbf{v}^{(n)}$. Consequently, we must also update $\mathbf{g}$:

$$\mathbf{g} \leftarrow \mathbf{x}^{(n+1)} - \frac{1}{\epsilon}\left(\mathbf{A}^\mathsf{T}\mathbf{u}^{(n)} + \mathbf{C}^\mathsf{T}\mathbf{v}^{(n)}\right) = \mathbf{g} + \left(\mathbf{g} - \mathbf{x}^{(n)}\right). \tag{22}$$

IV. Experiments

We reconstructed a simulated XCAT axial phantom [11] (Figure 3) and a helical shoulder scan with real patient data (Figure 4). In addition to the algorithm proposed in this paper, we reconstructed each image with

- OS-SQS: ordered subsets with separable quadratic surrogates [1],
- OS-SQS-FGM: OS-SQS with Nesterov's acceleration [6], and
- OS-SQS-OGM: OS-SQS with optimized gradient steps [5].

All OS algorithms used 12 subsets.

All algorithms were implemented in OpenCL and C and run on an aging NVIDIA GTX 480 with 2.5 GB of memory. For the OS algorithms, the data $\mathbf{y}$ and statistical weights $\mathbf{W}$ were transferred view-by-view to the GPU as required, but all other variables were stored on the GPU.

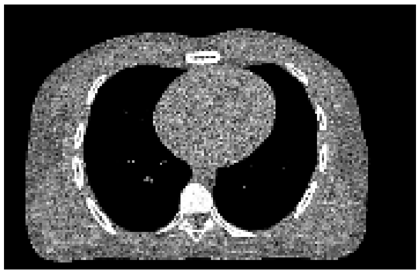
We used the Fair potential function

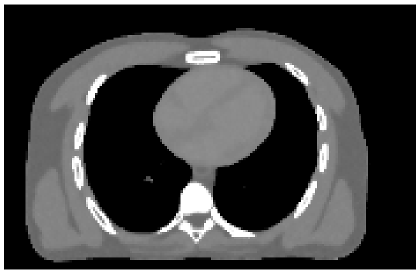
$$\psi_{\text{Fair}}(t) = \delta^2(|t/\delta| - \log(1 + |t/\delta|)), \tag{23}$$

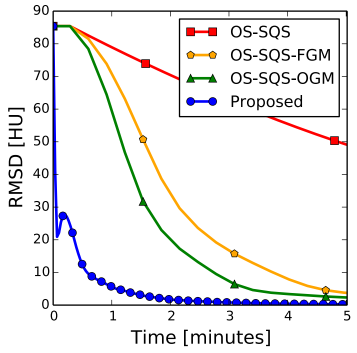
with $\delta = 10$ HU, which has a closed-form shrinkage operator, to penalize all 26 neighboring voxel differences. We used the separable footprints CT system model [8].

We chose the parameter ϵ to be the mean of the diagonal entries of $\mathbf{MW}$, where $\mathbf{M}$ is the diagonal majorizer used in the tomography update (13). We set $N_{\text{inner}} = N_{\text{view}}/120$ and $N_{\text{tomo}} = 5$ for both experiments.

Figures 3c and 4c show the root mean squared difference (RMSD) over a region of interest between the current iterate of each algorithm for the axial phantom and shoulder cases, respectively. The proposed algorithm converges very quickly in time. Markers are placed every 5 iterations.

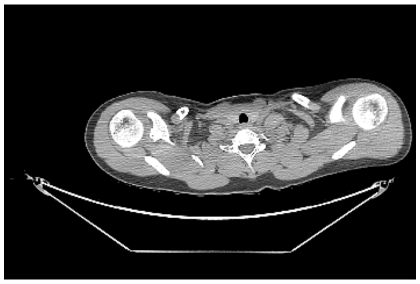

(a) Initial image

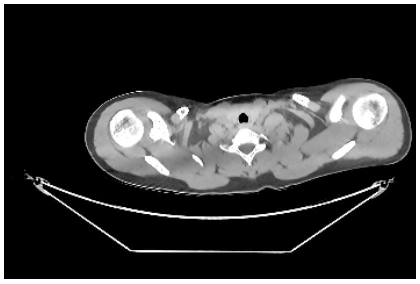

(b) Reference

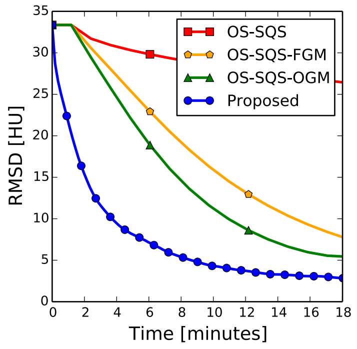


(c) RMSD *v.s.* Time

Fig. 3: Center slice (Fig. 3a) and converged reference (Fig. 3b) from axial phantom experiment, clipped to a 800 - 1200 modified HU window (with water at 0 HU). The proposed method converges quickly in time.


(a) Initial image


(b) Reference



(c) RMSD *v.s.* Time

Fig. 4: Center slice (Fig. 3a) and converged reference (Fig. 3b) from axial phantom experiment, clipped to a 800 - 1200 modified HU window (with water at 0 HU). The proposed method is within 5 HU RMSD of the reference in under 10 minutes.

V. Conclusions and Future Work

The duality-based algorithm in this paper appears to converge quickly, can handle a wide variety of edge-preserving regularizers, and can be efficiently implemented on the GPU. Similar to other algorithms like ASD-POCS [13] and variable splitting methods, the proposed algorithm alternates between tomography and denoising updates. If the update direction $\mathbf{d}^{(n)}$ is solved for exactly (4), then the proposed algorithm is convergent, but much slower than the experimental results shown. We suspect that the algorithm with inexact updates as implemented in this paper is convergent under ADMM-like conditions [3], but leave that result for future work.

Future work will also explore multiple-GPU implementations of the proposed algorithm.

References

[1] H. Erdoğan and J. A. Fessler. Ordered subsets algorithms for transmission tomography. *Phys. Med. Biol.*, 44(11):2835–51, November 1999.

[2] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge, UK, 2004.

[3] J. Eckstein and D. P. Bertsekas. On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators. *Mathematical Programming*, 55(1-3):293–318, April 1992.

[4] T. Goldstein and S. Osher. The split Bregman method for L1-regularized problems. *SIAM J. Imaging Sci.*, 2(2):323–43, 2009.

[5] 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*, pages 103–6, 2014.

[6] 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.*, 32(11):1965–78, November 2013.

[7] K. Lange. Convergence of EM image reconstruction algorithms with Gibbs smoothing. *IEEE Trans. Med. Imag.*, 9(4):439–46, December 1990. Corrections, T-MI, 10:2(288), June 1991.

[8] 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.*, 29(11):1839–50, November 2010.

[9] 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*, pages 359–62, 2014.

[10] M. G. McGaffin and J. A. Fessler. Fast edge-preserving image denoising via group coordinate descent on the GPU. In *Proc. SPIE 9020 Computational Imaging XII*, page 90200P, 2014.

[11] 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.*, 35(8):3800–8, August 2008.

[12] S. Shalev-Shwartz and T. Zhang. Stochastic dual coordinate ascent methods for regularized loss minimization. *J. Mach. Learning Res.*, 14:567–99, February 2013.

[13] E. Y. Sidky and X. Pan. Image reconstruction in circular cone-beam computed tomography by constrained, total-variation minimization. *Phys. Med. Biol.*, 53(17):4777–808, September 2008.

[14] 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.*, 34(11):4526–44, November 2007.