ACCELERATED ALTERNATING MINIMIZATION FOR X-RAY TOMOGRAPHIC RECONSTRUCTION

PEIJIAN DING * AND ADVISOR: DR. JAMES G. NAGY †

Abstract. While Computerized Tomography (CT) images can help detect disease such as Covid-19, regular CT machines are large and expensive. Cheaper and more portable machines suffer from errors in geometry acquisition that downgrades CT image quality. The errors in geometry can be represented with parameters in the mathematical model for image reconstruction. To obtain a good image, we formulate a nonlinear least squares problem that simultaneously reconstructs the image and corrects for errors in the geometry parameters. We develop an accelerated alternating minimization scheme to reconstruct the image and geometry parameters.

Key words. inverse problems, ill-posed problems, X-ray tomography, least squares problem, medical imaging

1. Introduction. Tomography is a technique of displaying representations of a cross section through an object through the use of some penetrating waves such as X-ray or ultrasound. In simple words, it allows us to see the inside of an object without breaking it. Thus, tomography is widely used in medical imaging, seismic exploration, and material science. In medical imaging, a Computerized Tomography (CT) Scan creates a cross-sectional image of human body by combining X-ray images taken from different scanning positions θ.

During a CT scan, the patient lies on a bed that slowly moves through the gantry while the X-ray tube rotates around the patient and shoots X-ray beams through the human body, received by a detector. Then, an image of the cross section of the human body is reconstructed following a mathematical procedure. In this paper, we consider fan beam tomography where a single light source rotates around an object and emits in a fan-shaped distribution, as shown in Figure 1.

Although CT images can help doctors diagnose and monitor diseases such as Covid-19, regular CT machines are heavy and expensive and not widely available in less developed areas. The goal of this paper is to compensate for cheaper and more portable machines by solving for geometry parameters such as the source-to-object distance that may not be calibrated precisely during the imaging process. Related work in parallel-beam tomography can be found in [1] [5]

Source-to-object distance measures how far away the centroid of the object is from the X-ray source, as shown in Figure 2. Since the source-to-object distance may vary at different scanning positions, the reconstructed image will be corrupted if incorrect values are used, as illustrated in Figure 3. Similar conclusions can be drawn if incorrect scanning positions are used to reconstruct the image. Our algorithm will significantly improve image quality by taking into account the variation in geometry parameters.

For the purpose of this paper, we primarily focus on 2D computed tomography to simplify notations and to reduce computational costs required to perform a substantial number of numerical experiments. But the methods discussed in this paper can be extended to 3D problems. An X-ray imaging problem can be formulated as an inverse problem in linear algebra:

*This work was done while the author was an undergraduate student at Emory University. peijian.ding6@gmail.com

†Department of Mathematics, Emory University. jnagy@emory.edu

Copyright © SIAM
Unauthorized reproduction of this article is prohibited

This manuscript is for review purposes only.

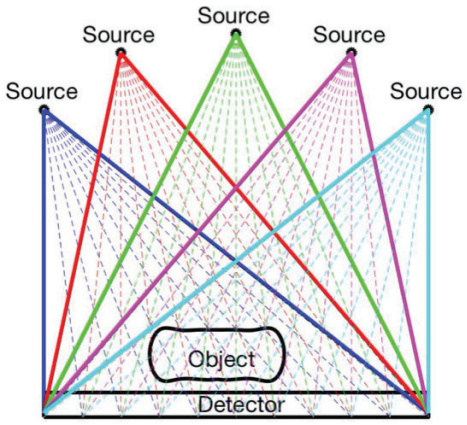

FIG. 1. *This is a simple illustration of our tomography problem.*

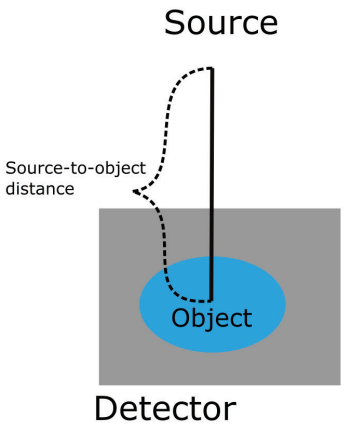

FIG. 2. *Illustration of source-to-centroid-of-object distance or source-to-object distance in short*

$$Ax = b.$$

The linear system is generated using the software package AIR Tools II [9] that simulates a 2D fan-beam CT with a curved detector. b is projection data, also referred to as a sinogram. Its entries are

$$b_i = \sum_{j \in S_i} L_{ij} x_j$$

S_i is the set of indices to those pixels that are penetrated by the ith ray, L_{ij} is the length of the ith ray through the jth pixel, and x_j is the attenuation coefficient in pixel j. A is a $m \times n$ forward operator, and its entries are defined as

$$a_{ij} = \begin{cases} L_{ij}, & j \in S_i \\ 0, & \text{otherwise.} \end{cases}$$

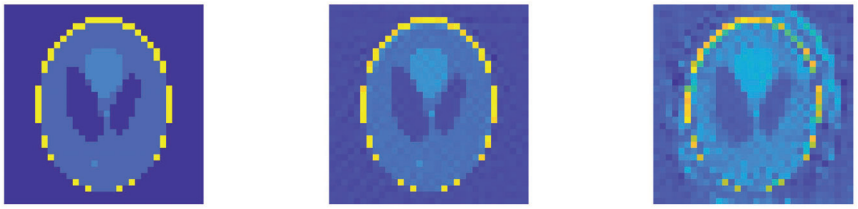

FIG. 3. 32×32 *true image of a Shepp–Logan phantom (left), image computed by taking into account the correct source-to-object distance (middle), image computed by using incorrect source-to-object distance (right)*

.

In AIR Tools II [9], it is assumed that the distance from the X-ray source to the centroid of the object at scanning position θ is $d_\theta n$, where n^2 is the number of pixels in the image and $d_\theta \in \mathbb{R}$. d is a vector of distance parameters d_θ at all scanning positions. Since n is fixed, we only need to compute the vector d to recover the source-to-object distance vector nd for all discrete scanning positions. The forward operator A is determined by geometry parameters r, a set of general geometry parameter vectors that can represent the vector d that determines the source-to-object distance and the vector of errors in scanning positions $\delta\theta$. A functional representation of the forward operator in terms of r is not available. We form the following nonlinear least squares problem:

$$\arg\min_{x,r} ||A(r)x - b||_2^2 \tag{1.1}$$

where x represents the image vector, and A is the forward operator that is a function of r and maps the true image x to the sinogram b. If we only consider the impact of source-to-object distance vector nd on the image quality, then $r = \{d\}$ and our nonlinear least squares problem becomes:

$$\arg\min_{x,d} ||A(d)x - b||_2^2.$$

If we want to model the impact of both source-to-object distance and errors in scanning positions $\delta\theta$ on reconstructed image, $r = \{d, \delta\theta\}$, then the least squares problem becomes:

$$\arg\min_{x,d,\delta\theta} ||A(d,\delta\theta)x - b||_2^2.$$

Thus, the conclusions we make about r also apply to both d and $\delta\theta$. This approach can also allow us to add more geometry parameter vectors that potentially have impact on the imaging quality. In this paper, we show that Equation 1 can be solved using an accelerated Block Coordinate Descent method. In addition to incorporating an acceleration scheme, we also exploit separability to further reduce the computational cost.

2. Alternating Minimization Scheme. In order to solve the nonlinear least squares problem, an intuitive approach to consider is the Block Coordinate Descent (BCD) algorithm. In this section, we discuss the linear and nonlinear least squares

This manuscript is for review purposes only.

problem involved in BCD. We also briefly discuss the variable projection approach and argue the advantage of BCD over variable projection for our problem.

2.1. Block Coordinate Descent. Block Coordinate Descent is a simple approach to solve an optimization problem. Its idea is based on the general Coordinate Descent (CD) algorithm. Because of its lack of sophistication, most optimization researchers have not focused on this approach until recently when CD approaches were found to be computationally competitive to other reputable alternatives in various applications such as machine learning [17]. Since we are able to exploit separability in the geometry parameters, the BCD method, which is given in Algorithm 2.1, is worth investigating for tomographic reconstruction.

Algorithm 2.1 BCD to Reconstruct Geometry and Image Parameters

1: Input: $r_0 \in \mathbb{R}^{N_A}$, $x_0 \in \mathbb{R}^n$
2: **for** $k = 1, 2, \ldots$ until a stopping criterion holds **do**
3: $\quad r_k = \arg\min_r ||A(r)x_{k-1} - b||_2^2$
4: $\quad x_k = \arg\min_x ||A(r_k)x - b||_2^2$
5: **end for**

In Algorithm 2.1, n is the length of the image vector and N_A denotes the number of sets of scanning positions in which errors are assumed to be constant. For simplicity, we refer to N_A as "number of angles". More explanation about N_A will be given in the following discussion on partitioning matrix A into blocks. Note that in practice an initial estimate r_0 is given. With this information, we can easily obtain x_0 by solving the linear least squares problem,

$$x_0 = \arg\min_x ||A(r_0)x - b||_2^2.$$

We remark that for ill-posed problems, computing x_0 and Step 4 of Algorithm 2.1 typically requires incorporating regularization procedures to avoid amplifying noise when solving the linear least squares problems. This is discussed further in section 2.2.1. There is another property in our matrix that makes this alternating minimization scheme favorable. The matrix A and vector b can be partitioned into block of rows as

$$A = \begin{bmatrix} A_1(r^{(1)}) \\ \vdots \\ A_i(r^{(i)}) \\ \vdots \\ A_{N_A}(r^{(N_A)}) \end{bmatrix} \quad b = \begin{bmatrix} b_1 \\ \vdots \\ b_i \\ \vdots \\ b_{N_A} \end{bmatrix}$$

where errors in geometry parameters associated with rows in each $[A_i(r^{(i)}), b_i]$ block are assumed constant. For example, if we collect projections at $0, 1, 2, \ldots, 359$ degrees around the center of the object, and $N_A = 10$, then an error is introduced into the geometry parameters once every 36 degrees. In reality, $N_A = 360$, but if some of the errors are small enough to ignore, using a smaller N_A can reduce the computational cost. The choice of N_A is discussed further in Section 4.3.

This manuscript is for review purposes only.

The separability of the geometry parameters allows us to solve for a set of much smaller systems at Step 3 in Algorithm 2.1, namely,

$$r_k^{(i)} = \arg\min_r ||A_i(r)x_{k-1} - b_i||_2^2, \quad i = 1, 2, \ldots, N_A$$

and then

$$r_k = \begin{bmatrix} r_k^{(1)} \\ \vdots \\ r_k^{(N_A)} \end{bmatrix}.$$

Note that parameters in the vector r_k are completely independent, so they can be updated simultaneously on a parallel computing architecture, which could dramatically lower the computing time. In fact, if we only consider source-to-object distance as geometry parameters, the dimension of the parameter $r^{(i)}$ is one. Comparing to this alternating minimization scheme, Variable Projection [11][12] cannot utilize this matrix property because its matrix multiplication process would take away the separability property. Thus, it makes this alternating minimization scheme more worthwhile to investigate.

Next, we discuss linear least squares solvers and nonlinear least squares solvers respectively.

2.2. Linear Least Squares Problem. In this subsection, we consider the linear least squares problem in Step 4 of Algorithm 2.1, $\min_x ||Ax - b||_2^2$, where we assume matrix A is fixed, e.g. $A = A(r_k)$.

2.2.1. Regularization. Typically in inverse problems, the data we obtain is not the exact data. This is also the case in our X-ray tomography problem, even in the case when geometry parameters, and hence matrix A, are known exactly. Ideally, we want to solve $Ax_{true} = b_{true}$ but the linear system that we actually solve is:

$$Ax = b = b_{true} + \eta \tag{2.1}$$

where η is the noise in our measurement data, b_{true} is the noise-free data, $A \in \mathbb{R}^{m\times n}$, and $b \in \mathbb{R}^m$.

By using the full SVD of $A = U\Sigma V^T$ where U is $m \times m$, Σ is $m \times n$, and V is $n \times n$. σ_i is the i^{th} singular value and u_i, v_i are the i^{th} column of the left and right orthogonal matrices U and V. Let r be the rank of A [1]. We first recall that $Av_i = \sigma_i u_i$, and we note that we can find scalars α_i and η_i such that

$$x_{true} = \sum_{i=1}^{n} \alpha_i v_i \qquad \text{and} \qquad \eta = \sum_{i=1}^{m} \eta_i u_i$$

Using these relations, we obtain

$$b = \sum_{i=1}^{m} (\alpha_i \sigma_i + \eta_i) u_i, \tag{2.2}$$

[1] We admit an ambiguous use of notation here, using r to denote rank of matrix A and to denote the set of geometry parameter vectors. Outside Sections 2.2.1 and 2.2.2, we use r exclusively to denote the set of geometry parameters. We hope this ambiguity is not too confusing, and that the two usages are clear from the context.

This manuscript is for review purposes only.

where $\alpha_i \sigma_i = 0, \forall i > r$. Then, notice that

$$A^\dagger u_i = V \Sigma^\dagger U^T u_i = \begin{cases} \frac{1}{\sigma_i} v_i, & i \leq r \\ 0, & i > r \end{cases} \tag{2.3}$$

$A^\dagger$ is the pseudo-inverse of A, and $\Sigma^\dagger$ is $n \times m$ matrix whose first r entries on the main diagonal are respectively $\frac{1}{\sigma_1}, \ldots, \frac{1}{\sigma_r}$. The rest of the entries of $\Sigma^\dagger$ are zero.

Thus,

$$x = A^\dagger b = \sum_{i=1}^{r} \left(\alpha_i + \frac{\eta_i}{\sigma_i} \right) v_i \tag{2.4}$$

From this result we observe that the noise in the data is magnified by the small singular values. Since $\sigma_1 \geqslant \sigma_2 \geqslant ... \geqslant \sigma_r \geqslant 0$, larger indices correspond to smaller singular values. The computed solution is dominated by noise amplified by division of small singular values. Thus, we need regularization schemes to filter out this noise. In particular, a regularized solution can be written as:

$$x_{reg} = \sum_{i=1}^{r} \phi_i \left(\alpha_i + \frac{\eta_i}{\sigma_i} \right) v_i \tag{2.5}$$

where the scalar $0 \leqslant \phi_i \leqslant 1$ is called a filter factor. As σ_i decreases, the filter factors should approach zero so that the noise contributed by the small singular values are filtered out.

Truncated Singular Value Decomposition. Since the noise is magnified by small singular values, the most intuitive approach is to cut off the small singular values by setting them to zero. This is called Truncated SVD regularization. The TSVD solution to the inverse problem is given by

$$x_{reg} = \sum_{i=1}^{k} \frac{b^T u_i}{\sigma_i} v_i \tag{2.6}$$

where $k \leqslant n$. The critical part in the TSVD is identifying the threshold k. One approach is choosing k at a significant drop-off of singular values, as illustrated by Figure 4.

In this case, $k = 9$ can be easily identified as the threshold for the TSVD approach. However, typically in inverse problems, singular values decay smoothly, as illustrated by Figure 5. In cases like Figure 5, a reliable cut-off threshold for singular values is hard to find. Note that $k = 250$ is not a good choice for the cut-off because the singular values are very close to zero for smaller indices k (e.g. $\sigma_{200} \approx 10^{-8}$). Therefore, we need more advanced regularization techniques to deal with smoothly decaying singular values.

Tikhonov Regularization. Classical Tikhonov Regularization can be used to solve ill-posed problems. The regularized solution x_{reg} is the unique solution to the following:

$$\min_x ||Ax - b||_2^2 + \lambda^2 ||x||_2^2 \tag{2.7}$$

This manuscript is for review purposes only.

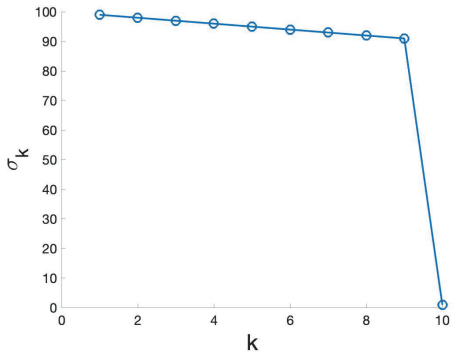

FIG. 4. *The singular values plot of a* 10×10 *diagonal matrix whose singular values are respectively* 99, 98 ... 91, *and* 1

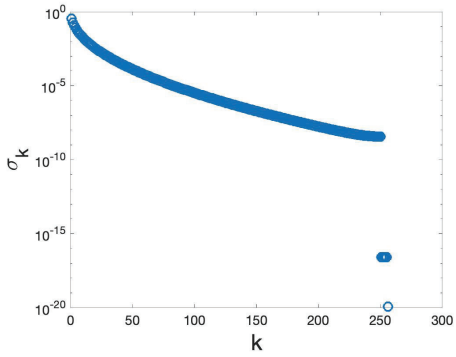

FIG. 5. *The singular values of the* 256×256 *test problem* heat *generated from Regularization Tools in MATLAB*

where λ is the regularization parameter that controls the smoothness of the regularized solution. The above equation is equivalent to the following:

$$\min_x \left\| \begin{bmatrix} A \\ \lambda I \end{bmatrix} x - \begin{bmatrix} b \\ 0 \end{bmatrix} \right\|_2^2 \tag{2.8}$$

Then the normal equations for this least squares problem can be written as:

$$(A^T A + \lambda^2 I) x_{reg} = A^T b. \tag{2.9}$$

From the normal equations, we obtain the following:

$$\begin{aligned} x_{reg} &= (A^T A + \lambda^2 I)^{-1} A^T b \\ &= \sum_{i=1}^{r} \phi_i \frac{b^T u_i}{\sigma_i} v_i \end{aligned} \tag{2.10}$$

where the filter factor is $\phi_i = \frac{\sigma_i^2}{\sigma_i^2 + \lambda^2}$. The modified Tikhonov regularization is not too much different from the classical Tikhonov except that it uses the 2-norm of Lx instead of x in Equation 2.7, where L is a $p \times n$ matrix with $p \leqslant n$.

$$\min_x ||Ax - b||_2^2 + \lambda^2 ||Lx||_2^2$$

This manuscript is for review purposes only.

In contrast to TSVD where singular values after σ_k are cut off, the Tikhonov regularization applies a smoother filter to all the singular values. Given a good regularization parameter, the large singular values would not be affected too much whereas the small values would be gradually filtered more as they approach zero. The quality of the filtering depends on how we choose the regularization parameter. If $\lambda = 0$, then all $\phi_i = 1$ and we are directly calculating the inverse (or pseudo-inverse) solution. If we select a very large $\lambda \gg \sigma_1$, all ϕ_i would approach zero and we would have over-smoothed the solution.

2.2.2. Parameter Choice Methods. The choice of regularization parameter is critical to the quality of the regularized solution. Parameter choice methods can usually be divided into two classes depending on their assumption about error norm $||\eta||_2^2 = ||b - b_{true}||_2^2$ where b is the measured data and b_{true} is the noise free data [8]. The first class contains methods based on a good estimate of $||\eta||_2^2$. The second class includes methods that are not based on a good estimate of $||\eta||_2^2$, but seek to extract this information from the given right hand side b. In this section, we introduce the Generalized Cross-Validation method (GCV) which is a popular method in the second class. The underlying idea of GCV is that a good regularization parameter should predict missing values. For example, if some data point b_i is missing in the right hand side, the regularized solution should predict this missing value well. GCV can be written as:

$$G(\lambda) = \frac{||Ax_{reg} - b||_2^2}{trace(I_m - AA_F^\dagger)^2} \tag{2.11}$$

where x_{reg} is the regularized solution, $A_F^\dagger = (A^TA + \lambda^2 I)^{-1}A^T$ is the pseudo-inverse of $\begin{bmatrix} A \\ \lambda I \end{bmatrix}$. The goal is to find λ such that $G(\lambda)$ is minimized. Now we simplify the numerator and the denominator of $G(\lambda)$ respectively:

$$\begin{aligned} ||Ax_{reg} - b||_2^2 &= ||\Sigma V^T x_{reg} - U^T b||_2^2 \\ &= ||(\Sigma\Sigma_F^\dagger - I)U^T b||_2^2 \end{aligned}$$

$$\begin{aligned} trace(I_m - AA_F^\dagger) &= trace((I_m - U\Sigma\Sigma_F^\dagger U^T)) \\ &= trace(I_m - \Sigma\Sigma_F^\dagger) \end{aligned}$$

Thus, $G(\lambda)$ becomes:

$$G(\lambda) = \frac{||(\Sigma\Sigma_F^\dagger - I)U^T b||_2^2}{trace(I_m - \Sigma\Sigma_F^\dagger)} \tag{2.12}$$

Then we can write $G(\lambda)$ in the case of Tikhonov regularization as:

$$G(\lambda) = \frac{\sum_{i=1}^{r}(\frac{\lambda^2\tilde{b}_i^2}{\lambda^2+\sigma_i^2} + \sum_{i=r+1}^{m}\tilde{b}_i^2)}{(m-r) + \sum_{i=1}^{r}\frac{\lambda^2}{\lambda^2+\sigma_i^2}} \tag{2.13}$$

where $\hat{b} = U^T b$. We can solve for λ by using the function **fminbnd** in MATLAB which is based on golden section search and parabolic interpolation. In practical applications, several studies have found that occasionally GCV would drastically undersmooth the solution by choosing the regularization parameter too small [4] [6] [15]. In

This manuscript is for review purposes only.

[2], GCV has been found to over-smooth the solution in the Lanczos-hybrid methods, discussed in the next subsection. To alleviate this difficulty, it was proposed to use the weighted GCV method:

$$G(w,\lambda) = \frac{||Ax_{reg} - b||_2^2}{trace(I_m - wAA_F^{\dagger})^2} \tag{2.14}$$

where w is the weight parameter that determines the function $G(w,\lambda)$ along with λ. When $w = 1$, we have the non-weighted version of GCV like in Equation 2.12. When $w > 1$, the solution is smoother. When $w < 1$, the solution is less smooth. We use a weighted-GCV approach to choose the weight adaptively. More details can be found in [2]

2.2.3. Hybrid LSQR. We have so far discussed regularization and parameter choice methods using the SVD. For large-scale problems, such as in image reconstruction, directly applying SVD based methods is not computationally feasible. In this subsection we describe a hybrid LSQR scheme that combines an efficient iterative method with SVD based approaches that enforce regularization on small projected sub-problems. LSQR is an iterative conjugate gradient method to solve least squares problems [13]. A hybrid approach, called hybrid LSQR in [2] and implemented as IRhybrid_lsqr in IR Tools [7] solves the Tikhonov regularized least squares problem, and is able to adaptively estimate regularization parameters at each iteration using a weighted generalized cross validation (GCV) method. For implementation details, see [7].

The standard LSQR algorithm projects the linear least squares problem onto a sequence of Krylov subspace of small and increasing dimensions. The best approximation x_k to the least squares problem in the Krylov subspace K_k is given by $x_k = V_k y_k$, where V_k is a $n \times k$ matrix with orthonormal columns at the k^{th} step of the Golub-Kahan bidiagonalization process [13]. y_k can be solved by

$$y_k = \arg\min_{y_k} ||B_k y_k - \beta_1 e_1||_2$$

where B_k is the bidiagonal matrix at the k^{th} step of the Golub-Kahan bidiagonalization process, β is a scalar, and e_1 is the standard basis vector. When being applied to ill-posed problems, LSQR exhibits a semi-convergence behavior which means that early iterations construct information related to the solution while later iterations construct information related to noise [2].

This can be compensated by applying a direct regularization method such as Tikhonov or TSVD, which can be solved cheaply on a small scale problem of the reduced linear least squares in the Krylov subspace. So, we can write the Hybrid LSQR using Tikhonov regularization as:

$$\min_{y_k} ||B_k y_k - \beta_1 e_1||_2^2 + \lambda_k^2 ||y_k||_2^2$$

where λ_k is a regularization parameter chosen at the k^{th} iteration using the weighted GCV method discussed in Section 2.2.2. A method like GCV can be used to choose a stopping iteration so that k will not be too large. details can be found in [2].

Comparing to LSQR, this hybrid method can effectively stabilize the iterations [2]. Although at each iteration a new regularization parameter must be chosen, it is not computationally expensive for the projected problem.

To summarize the method, the hybrid LSQR method projects the large scale linear least squares problem onto a low-dimensional Krylov subspace where we can inexpensively apply a direct regularization method like the adaptive weighted-GCV.

2.3. Nonlinear Least Squares. In our alternating minimization scheme, we iteratively solve the image and the geometry parameters. While we have discussed methods to solve the linear least squares problem in the previous section, we need other tools to solve the nonlinear least squares problem:

$$\min_r ||A(r)x - b||_2 \tag{2.15}$$

where x is approximated by the linear least squares solution we obtained by using hybrid LSQR in Section 2.3.

We utilize the implicit filtering method which solves the bound-constraint optimization problem for which the derivative information is not available [10]. Since we do not have the derivative information of our objective function and a reasonable bound can be established for the geometry parameters in our tomographic reconstruction problem, implicit filtering serves as a good tool to solve our problem. Implicit filtering builds the local model of the objective function using a quasi-Newton method.

In our numerical experiments, we compare implicit filtering to the MATLAB function **fminbnd**.

3. Acceleration Algorithms. In the previous section, we have introduced the alternating minimization scheme and the methods we use to solve least squares problems. In this section, we introduce methods that will accelerate the convergence of our minimization scheme.

3.1. Accelerated Block Coordinate Descent. Since we can divide variables in our least squares problem into two blocks – geometry parameters r and image x, it makes sense for us to directly investigate methods that accelerate the BCD algorithm. We implemented **Accelerated Block Coordinate Descent** (ABCD). This method can be applied to a four-block problem by dividing it into two larger blocks, but in our problem we do not have to do so. Theoretically, the proposed acceleration method in [3] has a complexity of $O(\frac{1}{k^2})$. In our implementation, we simplify the algorithm as Algorithm 3.1.

Algorithm 3.1 Accelerated Block Coordinate Descent

1: Inputs: $t_0 = 1$, $r_0 \in \mathbb{R}^{N_A}$ and $x_0 \in \mathbb{R}^n$
2: **for** $k = 1, 2, \ldots$ until a stopping criterion holds **do**
3: $\tilde{r}_k = \arg\min_r ||A(r)x_{k-1} - b||_2^2$
$\tilde{x}_k = \arg\min_x ||A(\tilde{r}_k)x - b||_2^2 + \lambda^2||x||_2^2$
4: $\tilde{w}_k = (\tilde{x}_k, \tilde{r}_k)$
$t_k = \frac{1}{2}(1 + \sqrt{1 + 4t_{k-1}^2})$
$w_k = \tilde{w}_{k-1} + \frac{t_{k-1}}{t_{k+1}}(\tilde{w}_k - \tilde{w}_{k-1})$, where $w_k = (x_k, r_k)$
5: **end for**

N_A denotes the number of angles, and n is the length of the image vector. We remark that we can exploit separability when solving for $\tilde{r}_k$, as discussed in Section 2.1. As

This manuscript is for review purposes only.

mentioned in Algorithm 2.1, since r_0 is given,

$$x_0 = \arg\min_x ||A(r_0)x - b||_2^2 + \lambda^2 ||x||_2^2.$$

The Tikhonov regularized least squares problems for x_0 and $\tilde{x}_k$ are solved using the hybrid scheme **IRhybrid_lsqr** provided in **IRTools**[7] in MATLAB. The nonlinear least squares problem for $\tilde{r_k}$ is solved using a MATLAB package called **imfil** [10]. We have found that accelerating the solution vector x alone has yielded stabler results with slightly better accuracy than performing acceleration on both x and r. We show the numerical experiments in section 4.

3.2. Anderson Acceleration. Anderson Acceleration, also called Anderson mixing, is a method used to accelerate the convergence of fixed point iteration. Note that we can write out alternating minimization scheme as a fixed point iteration, where

Algorithm 3.2 Fixed point iteration of the image vector

1: Input: $x_k \in \mathbb{R}^n$, Output: $x_{k+1} = g(x_k)$
2: $r_{k+1} = \arg\min_r ||A(r)x_k - b||_2^2$
3: $x_{k+1} = \arg\min_x ||A(r_{k+1})x - b||_2^2 + \lambda^2 ||x||_2^2$

$g : \mathbb{R}^n \to \mathbb{R}^n$ is the fixed point iteration of image vector x, as shown in Algorithm 3.2.

For this fixed point iteration, the general form of Anderson Acceleration is formed as the following:

Algorithm 3.3 Anderson Acceleration

1: Inputs: x_0 and $m \geqslant 1$
2: Set $x_1 = g(x_0)$, using Algorithm 3.2
3: **for** $k = 1, 2, \ldots$ until a stopping criterion holds **do**
4: $m_k = min(m, k)$
5: Set $F_k = (f_{k-m_k}, ..., f_k)$, where $f_i = g_i(x_i) - x_i$ and $g_i(x_i)$ comes from Algorithm 3.2
6: Determine $\alpha^{(k)} = (\alpha_0^{(k)}, ..., \alpha_{m_k}^{(k)})^T$ that solves $\min_\alpha ||F_k\alpha||_2$ s.t $\sum_{i=0}^{m_k} \alpha_i = 1$
7: Set $x_{k+1} = \sum_{i=0}^{m_k} \alpha_i^{(k)} g(x_{k-m_k+i})$, where $g(x_{k-m_k+i})$ is from Algorithm 3.2
8: **end for**

We can cast the linear constrained optimization problem in Step 7 of Algorithm 3.3 into an unconstrained form which is straightforward to solve and convenient for efficient implementation [16].

We define $\nabla f_i = f_{i+1} - f_i$ for each i and set $\nabla F = (\nabla f_{k-m_k}, ..., \nabla f_{k-1})$. Then the least squares problem is equivalent to

$$\min_{\gamma=(\gamma_0,\ldots\gamma_{m_k-1})^T} ||f_k - F_k\gamma||_2,$$

where $\alpha_0 = \gamma_0$ and $\alpha_i = \gamma_i - \gamma_{i-1}$, for $1 \leqslant i \leqslant m_k - 1$ and $\alpha_{m_k} = 1 - \gamma_{m_k-1}$.

This unconstrained least squares problem leads to a modified version of Anderson Acceleration in Algorithm 3.4,

This manuscript is for review purposes only.

Algorithm 3.4 Modified Anderson Acceleration

1: Given x_0 and $m \geqslant 1$
2: Set $x_1 = g(x_0)$, using Algorithm 3.2
3: **for** k=1,2,... **do**
4: $m_k = min(m, k)$
5: Determine $\gamma^{(k)} = (\gamma_0^{(k)}, ..., \gamma_{m_k-1}^{(k)})^T$ that solves $\underset{\gamma=(\gamma_0,...\gamma_{m_k-1})^T}{min} ||f_k - F_k\gamma||_2$
6: Set $x_{k+1} = g(x_k) - G_k\gamma^{(k)}$, where $g(x_k)$ comes from Algorithm 3.2
7: **end for**

where

$$x_{k+1} = g(x_k) - \sum_{i=1}^{m_k-1} \gamma_i^{(k)}[g(x_{k-m_k+i+1}) - g(x_{k-m_k+i})] = g(x_k) - G_k\gamma^{(k)}$$

with $G = (\nabla g_{k-m_k}, ..., \nabla g_{k-1})$, $\nabla g_i = g(x_{i+1}) - g(x_i)$.

Homer Walker proposed implementation that efficiently updates the QR factors in the decomposition $F_k = Q_k R_k$ [16]. The basic logic is the following: every F_k is obtained from F_{k-1} with a column added on the right. If the resulting matrix has more columns than m, then delete one from the left. The column addition can be achieved by a modified Gram–Schmidt process. The deletion process is a little more complicated. We delete the first column on the left when $m_{k-1} = m$. If $F_{k-1} = QR$, then $F_{k-1}(:, 2:m) = QR(:, 2:m)$, where $R(:, 2:m)$ is upper-Hessenberg. Then, we can determine m Givens rotations to cancel out the entries in the sub-diagonal.

4. Numerical Experiments. In this section, we make a few comparisons of different methods to solve the X-ray tomography problem. Firstly, we compare the speed of BCD exploiting the separability of geometry (BCDS) and the speed of regular BCD. Secondly, we compare results produced from function evaluation budget in implicit filtering. Thirdly, we test how BCDS works when different "numer of angles" are used. Fourthly, we compare acceleration schemes. Fifthly, different regularization parameters in the linear least squares solvers are compared. Lastly, we show our algorithm works for both geometry parameters d and $\delta\theta$. For experiments 1-5, we solve problems with only unknown source-to-object distance *nd*. Since image size n is fixed, we only need to solve for geometry parameters vector $r = \{d\}$. For the last experiment, $r = \{d, \delta\theta\}$. Comparisons in all experiments are made about geometry errors and reconstruction errors. Geometry errors are defined as the relative errors of geometry parameters, $\frac{||r-r_{true}||_2}{||r_{true}||_2}$. For experiment 1-5, where $r = \{d\}$, geometry errors are $\frac{||d-d_{true}||_2}{||d_{true}||_2}$. For experiment 6, where $r = \{d, \delta\theta\}$ geometry errors are represented by both $\frac{||d-d_{true}||_2}{||d_{true}||_2}$ and $\frac{||\delta\theta-\delta\theta_{true}||_2}{||\delta\theta_{true}||_2}$. The reconstruction errors are defined as the relative errors of the image, $\frac{||x-x_{true}||_2}{||x_{true}||_2}$.

We use fan-beam projection for all our tomography problems for the sake of consistency. Note that we can also easily adapt our code to solve parallel beam projection problems by using the **IRtools** [7] and **AIR Tools** [9] MATLAB packages.

In practice, good initial guesses of geometry parameters r are available and prior knowledge can help us set the bounds for them. For the source-to-object distance d, we generate a test problem where true d values, d_{true}, are random numbers (chosen from a uniform distribution) between 1.5 and 2.5. We use a constant initial guess of

This manuscript is for review purposes only.

$d = 2$ for all scanning positions and set the bounds for d from 1.5 to 2.5. For the errors in scanning positions $\delta\theta$, we assume for each set of scanning positions the error bound is from 0 to 5. The test image we use is the Shepp-Logan Phantom [14]. The image size is 32×32 in the first two experiments – efficient BCD algorithm that exploits separability and function evaluation budget in the nonlinear least squares solver. In the first experiment, we want to demonstrate the advantage of BCDS over BCD in terms of computing time. In the second experiment, we want to empirically show what a good parameter is so that we can use it for larger tests later. For the rest of the experiments, we use image size 128×128. The noise level is $\frac{||\eta||_2}{||b||_2} = 0.01$, where η is a vector with random entries chosen from a normal distribution. Budget is a hyper-parameter in **imfil** that stands for the maximum number of function evaluations in the nonlinear least squares solver. Moreover, 0^{th} iteration is included in the relative errors figures below. This represents the relative error of the initial guesses with regard to the true solution.

4.1. BCD Exploiting Separability vs BCD. To distinguish from the regular BCD method, we refer to our BCD algorithm that exploits the separability of the geometry parameters as BCDS. The "S" here stands for separability. In this section, we compare the running time of BCD and that of BCDS. In this numerical experiment, $budget = 1000$ (used in **imfil** to put a limit on the number of function evaluations), and $N_A = 10$. As we see in Figure 6, BCDS dramatically speeds up the convergence because the separability allows us to solve a much smaller problem independently for one $r(i)$ at a time. The average running time of image reconstruction using BCD is $12.8s$, around the same as BCDS's $12.6s$. The time of geometry reconstruction using BCD is $1857.5s$, more then ten times longer than BCDS's $157.1s$. This is also the reason BCDS is discussed first in this section. In the remaining experiments, we always use the BCDS to reduce the running time. Moreover, the geometry errors and reconstruction errors of BCDS are both better than that of BCD. In Figure 7, the phantom reconstructed by BCDS is much closer to the true image than the one from BCD.

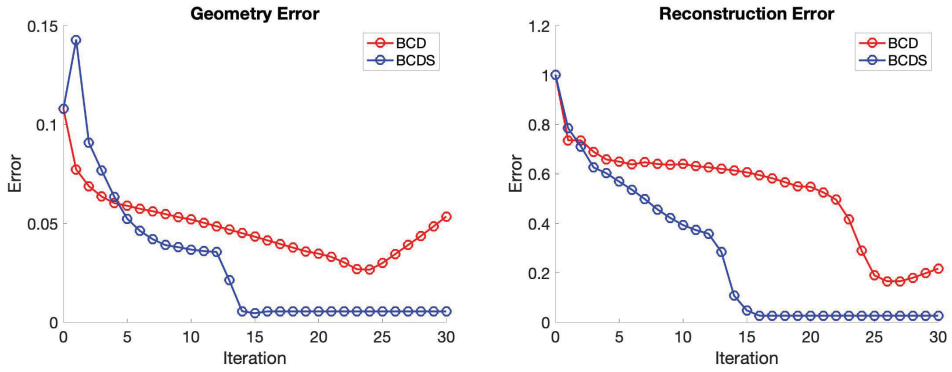

FIG. 6. *Comparison between BCD and BCDS.*

A typical CT image using fan beam x-ray sources collects data at scanning positions of one degree increment, from 0 to 359 degrees. In a perfect machine, the geometry parameters are precisely known each time the source is rotated to a new scanning position. In our experimental scenario, we assume the geometry parameters

FIG. 7. *Comparison of* 32×32 *Shepp-Logan phantom: true image (left), BCDS image (middle), BCD image (right).*

are only known approximately. To experiment with various scenarios, we assume that errors are introduced into the geometry parameters once every $\frac{360}{N_A}$ degrees. That is, with $N_A = 10$, errors occur once every 36 degrees.

Partially, N_A depends on the precision of machine calibration. For a good machine, N_A may be small. Also, N_A may depend on how precise we measure the data. For example, two geometry parameters that are different in terms of double precision may be rounded to the same number in single precision. In this scenario, the difference could be small enough that we can treat the two geometry parameters as being equal.

If the number of angles is larger than the true number of angles, we may end up solving a larger problem than we need. For example, if for every 36 degrees only one geometry error is introduced, then $N_{A_{true}} = 10$. If we assume $N_A = 20$, the average of the two geometry errors per 36 degrees would approximate the one true geometry error introduced in that set of scanning positions. If the number of angles is smaller than the true number of angles, we end up solving a low dimension approximation. We do not seek to solve for the image exactly but aim to compute good approximations that yield much better results than not considering geometry parameters at all. In practice, we can consider the number of angles as a hyper-parameter that practitioners can set based on their expertise and knowledge of the machine calibration. In our problems, we assume to know the number of angles N_A.

4.2. Imfil Budget. We explore the effect of evaluation budgets in the nonlinear least squares solver on the geometry and reconstruction errors. We set $N_A = 10$, and use $budget = 10, 100, 1000, 10000$. Since the budget size may greatly affect the nonlinear least squares solutions, we explore its effects on BCDS without any acceleration.

As we can see in Figure 8, 100, 1000, and 10000 are equally good. Both geometry and reconstruction errors are very small when 100, 1000, 10000 are used. Thus, 100 is the best budget out of the four because any more evaluation beyond 100 does not make the solution better. When budget is 1000 and 10000 respectively, we wasted many evaluations without making any progress. When 10 is used, even though we get convergent results earlier, the small budget causes the algorithm to terminate before it finds a better solution.

4.3. Number of Angles. We test our algorithm when the number of angles N_A is 5, 10, and 20 respectively. $budget = 100$. We want to explore the differences in relative error of r, relative error of x, the convergence, and the image quality.

Both geometry and image parameters converge for all three different number of angles, as shown in Figure 9. Although the errors when $N_A = 10$ are not as low as when $N_A = 5$, the reconstruction errors when $N_A = 10$ still have a 40% reduction

This manuscript is for review purposes only.

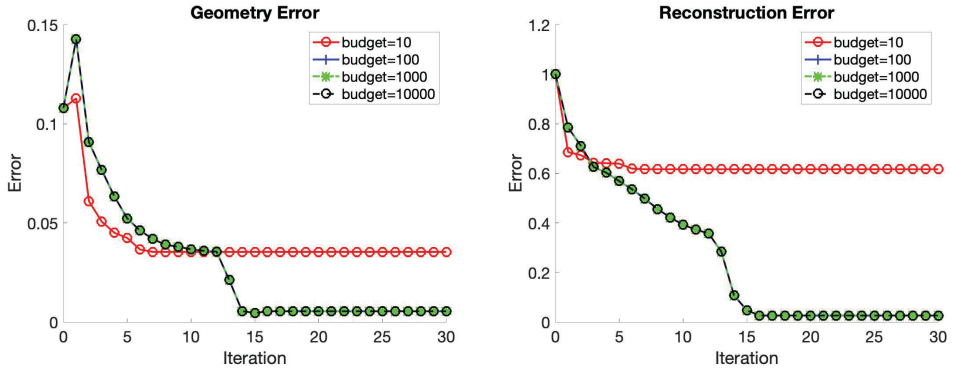

FIG. 8. *Comparison of budgets: geometry errors (left), reconstruction errors (right)*

h

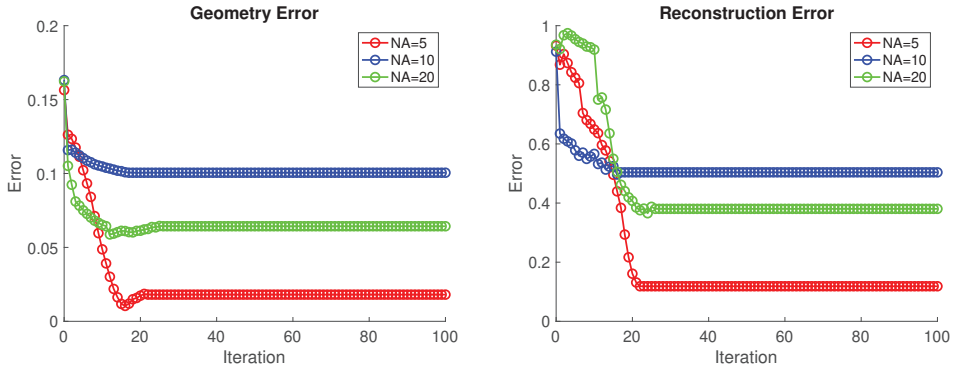

FIG. 9. *Comparison of number of angles N_A: geometry errors (left), reconstruction errors (right)*

comparing to its initial error level. Although the initial reconstruction error was close to 1, our algorithm converges and has error reduction for the CT image for at least 40% when $N_A = 5, 10, 20$ respectively.

4.4. Acceleration. In this section, we compare BCDS with no acceleration, ABCDS, and Anderson acceleration. ABCDS refers to the Accelerated Block Coordinate Descent algorithm that exploits separability of geometry parameters. The number of columns of the matrix F_k , also referred to as memory size in this paper, in Algorithm 3.4 is $m = 5$. Since the linear least squares problem at Step 5 of Algorithm 3.4 is relatively small, we directly use the MATLAB backslash operator to solve it.

In Figure 10, geometry errors and reconstruction errors converge for all three methods, however the error reduction in Anderson acceleration is very small. While Anderson acceleration does perform well in some other numerical experiments, we found it to be less consistent in achieving good error reduction as ABCDS as shown in Figure 10. ABCDS converges much faster than BCDS at around 30th iteration, while BCDS slowly improves and ends up achieving 10% better error reduction in this experiment

4.5. Regularization. In this section, we compare BCDS with different regularization parameters: no regularization, GCV, and weighted-GCV, as shown in Figure

This manuscript is for review purposes only.

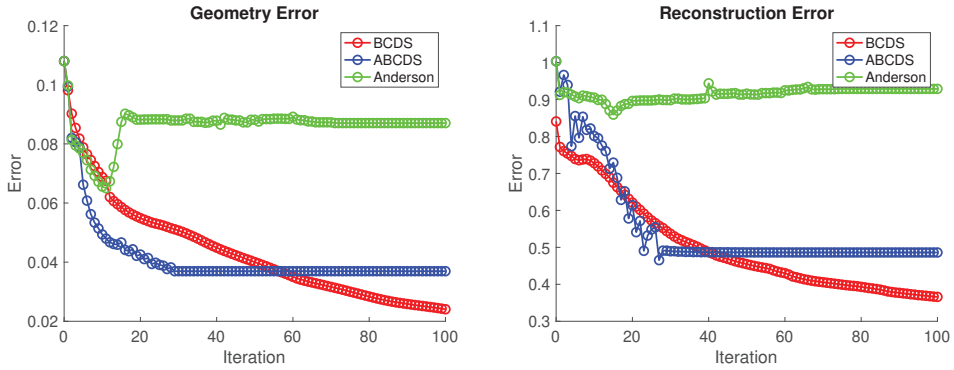

FIG. 10. *Comparison of BCDS, ABCDS, and Anderson: geometry errors (left), reconstruction errors (right)*

11. The reason we use BCDS without the acceleration methods is that we want to see the direct impact of regularization on the alternating minimization scheme. In this experiment, $N_A = 10$ and $budget = 100$.

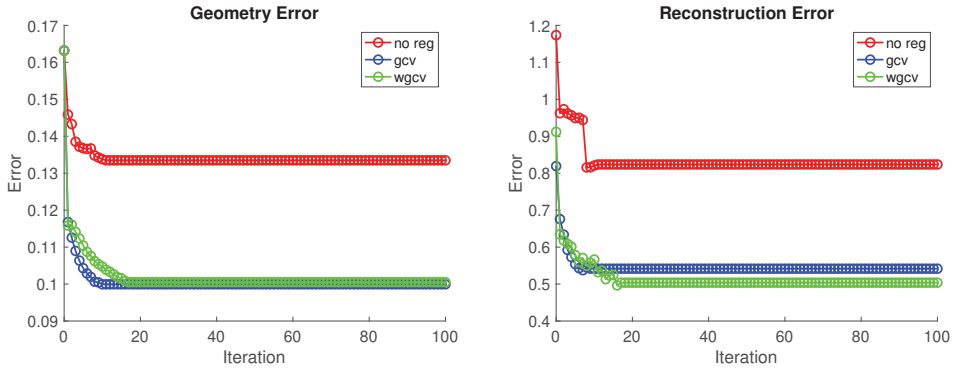

FIG. 11. *Comparison of regularization: geometry errors (left), reconstruction errors (right)*

When there is no regularization, **IRhybrid_lsqr** is essentially an LSQR algorithm. Since the geometry error converges at 18th iteration, the linear system arising from the geometry parameters stays the same after that, and results in a constant reconstruction error. GCV and weighted-GCV result in much better error reduction than when no regularization is used, while weighted-GCV produces slightly better error reduction than GCV.

4.6. Geometry Parameters: d and $\delta\theta$. We show that our alternating scheme as well as the accelerated version can also be applied to the case when geometry parameters $r = (d, \delta\theta)$.The nonlinear least squares problem becomes:

$$\arg\min_{x,\delta\theta,d} ||A(\delta\theta, d)x - b||.$$

In all previous experiments we assume there was no error introduced in scanning positions. If the scanning positions contain errors as well, the image quality could

also be affected, as illustrated in Figure 12. Since there is noise in the data vector b, we cannot obtain the true image even if we use the true geometry parameters. The images in the top middle and top right are obtained from ABCDS and BCDS respectively. We can see that there is virtually no difference between them except the image from ABCDS has a shade of blue that is closer to the best possible image. The bottom left image is obtained from BCDS when assuming there is no error in the scanning position. The color of this image is not as close to the best possible image as the previous two. Also, the image is more blurry. In Figure 13, we provide numerical evidence for the image comparison in Figure 12. In Figure 13, ABCDS produces the best results for all three parameter; BCDS that models only the source-to-object distance and ignores the errors in scanning position has the worst image reconstruction errors; ABCDS also converges faster and reaches better approximation for errors in "angles" or scanning positions.

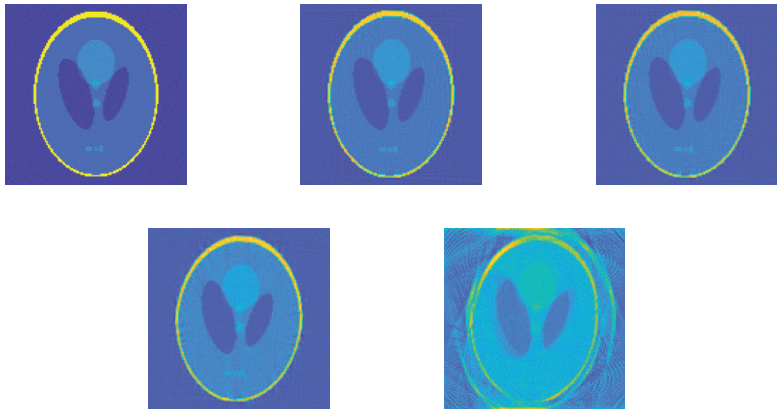

FIG. 12. *Best image possible (top left), image obtained by considering both $\delta\theta$ and d using ABCDS (top middle) , image obtained by considering both $\delta\theta$ and d using BCDS (top right), image obtained by using correct d but incorrect $\delta\theta$ (bottom left), image obtained by using incorrect geometry parameters d and $\delta\theta$ (bottom right).*

FIG. 13. *Comparison of ABCDS, BCDS, and BCDS that assumes geometry errors only include errors in source-to-object distance: geometry errors (left), reconstruction errors (middle), errors in scanning positions (right)*

5. Conclusion. In this paper, we propose an accelerated alternating minimization scheme to solve X-ray tomography problems. The linear least squares problem is solved by a weighted hybrid LSQR algorithm with Tikhonov regularization. The

This manuscript is for review purposes only.

nonlinear least squares problem is solved by implicit filtering. We also investigated ABCDS and Anderson mixing to accelerate convergence. We have the following findings:

1. BCDS runs faster than the normal BCD because the separability of parameters allow us to solve each entry of r independently. Also, since solving for each parameter separately makes the nonlinear least squares problem much easier than solving all parameters at once, BCDS is faster at finding a good solution and converge faster than BCD.
2. Choosing an appropriate budget for implicit filtering is important. When the budget is chosen too small, better solutions are not explored. When budget is chosen too big, many function evaluations are wasted without making progress. A suggested number in the **imfil** documentation is $10N^2$, where N is the number of geometry parameters. We found this formula does not always give the appropriate budget. Since we solve for each geometry parameter using separability, the dimension of each small problem is one. But, we have found 100 to be the best budget for 32×32 test problem with $N_A = 10$.
3. ABCDS is much less computationally costly than BCDS with Anderson acceleration because Anderson acceleration requires solving a linear least squares problem in each iteration when choosing the weight. Although both methods converge, ABCDS have better acceleration effects than Anderson acceleration.
4. The weighted-GCV approach seems to be the best among the parameter choice methods we compared. It achieves the best geometry error and reconstruction error based on our numerical experiments.
5. When we introduce errors in both scanning positions and source-to-object distance, ABCDS produces the closest approximation of the best possible image obtained by using the true geometry parameters. In practice, we could account for other geometry errors by adding nonlinear parameters in the least squares problem.

The ABCDS method has shown its success in our tomographic reconstruction problems. We believe this algorithm can be used to effectively solve X-ray tomography problems that have variations in the geometry parameter. A future direction towards more improvement would be adapting, applying, and advancing this algorithm on X-ray images produced in clinical trials.

Acknowledgments. This work was done as part of an undergraduate honors thesis project at Emory University, and was partially supported by the U.S. National Science Foundation under Grant DMS-1819042. The author would like to thank Chang Meng and James Nagy for their guidance and mentorship.

REFERENCES

[1] A. P. Austin, Z. W. Di, S. Leyffer, and S. M. Wild, *Simultaneous sensing error recovery and tomographic inversion using an optimization-based approach*, 2019, https://arxiv.org/abs/1902.02027.

[2] J. Chung, J. G. Nagy, and D. P. O'Leary, *A weighted GCV method for Lanczos hybrid regularization*, Electronic Transactions on Numerical Analysis, 28 (2008).

[3] Y. Cui, D. Sun, and K. Toh, *Computing the best approximation over the intersection of a polyhedral set and the doubly nonnegative cone*, SIAM Journal on Optimization, 29 (2019), pp. 2785–2813, https://doi.org/10.1137/18M1175562.

[4] D. Cummins, T. Filloon, and D. Nychka, *Confidence intervals for nonparametric curve estimates: Toward more uniform pointwise coverage*, Journal of the American Statistical

Association, 96 (2001), pp. 233–246, https://doi.org/10.2307/2670362.
[5] A. Dogandzic, R. Gu, and K. Qiu, *Mask iterative hard thresholding algorithms for sparse image reconstruction of objects with known contour*, Circuits, Systems and Computers, 1977. Conference Record. 1977 11th Asilomar Conference on, (2011), https://doi.org/10.1109/ACSSC.2011.6190402.
[6] J. H. Friedman and B. W. Silverman, *Flexible parsimonious smoothing and additive modeling*, Technometrics, 31 (1989).
[7] S. Gazzola, P. C. Hansen, and J. G. Nagy, *IR tools: A MATLAB package of iterative regularization methods and large-scale test problems*, 2018, https://arxiv.org/abs/1712.05602.
[8] P. C. Hansen, *Rank-Deficient and Discrete Ill-Posed Problems: Numerical Aspects of Linear Inversion*, Society for Industrial and Applied Mathematics, Philadelphia, 1998.
[9] P. C. Hansen and J. S. Jørgensen, *AIR tools ii: algebraic iterative reconstruction methods, improved implementation*, Numerical Algorithms, 79 (2018), pp. 107–137, https://doi.org/10.1007/s11075-017-0430-x.
[10] C. T. Kelley, *Implicit filtering*, Society for Industrial and Applied Mathematics, (2011).
[11] D. P. O'Leary and B. W. Rust, *Variable projection for nonlinear least squares problems*, Computational Optimization and Applications, 54 (2013), pp. 579–593.
[12] V. Pathuri-Bhuvana, S. Schuster, and A. Och, *Joint calibration and tomography based on separable least squares approach with constraints on linear and non-linear parameters*, in 2020 28th European Signal Processing Conference (EUSIPCO), IEEE, 2021, pp. 1931–1935.
[13] Å. Björck, *Numerical Methods for Least Squares Problems*, Society for Industrial and Applied Mathematics, 1996, https://doi.org/10.1137/1.9781611971484.
[14] L. Shepp and B. F. Logan, *The Fourier reconstruction of a head section*, IEEE Transactions on Nuclear Science., (1974).
[15] R. Vio, P. Ma, W. Zhong, J. G. Nagy, L. Tenorio, and W. Wamsteker, *Estimation of regularization parameters in multiple-image deblurring*, A&A, 423 (2004), pp. 1179–1186, https://doi.org/10.1051/0004-6361:20047113.
[16] H. F. Walker, *Anderson acceleration: Algorithms and implementations*, WPI Math. Sciences Dept. Report MS-6-15-50, (2011).
[17] S. J. Wright, *Coordinate descent algorithms*, Mathematical Programming, 151 (2015), pp. 3–34, https://doi.org/10.1007/s10107-015-0892-3.