A Data-Parallel Algorithm for Iterative Tomographic Image Reconstruction [*]

Calvin A. Johnson
Center for Information Technology
National Institutes of Health
Bethesda, MD 20892-5624
johnson@mail.nih.gov

Ariela Sofer
Operations Research and Engineering Department
George Mason University
Fairfax, VA 22030-4444
asofer@gmu.edu

Abstract

In the tomographic imaging problem, images are reconstructed from a set of measured projections. *Iterative reconstruction methods are computationally intensive alternatives to the more traditional Fourier-based methods. Despite their high cost, the popularity of these methods is increasing because of the advantages they pose. Although numerous iterative methods have been proposed over the years, all of these methods can be shown to have a similar computational structure. This paper presents a parallel algorithm that we originally developed for performing the expectation maximization algorithm in emission tomography. This algorithm is capable of exploiting the sparsity and symmetries of the model in a computationally efficient manner. Our parallelization scheme is based upon decomposition of the measurement-space vectors. We demonstrate that such a parallelization scheme is applicable to the vast majority of iterative reconstruction algorithms proposed to date.*

1. Introduction

Tomographic images are reconstructed from measured *projections* collected from the scanning instrument. These projections are essentially a finite set of blurred and noisy line integrals through the object of interest [11]. Applications of the tomographic image reconstruction problem are quite numerous and include x-ray computed tomography (CT), ultrasound computed tomography, emission tomography (positron emission tomography or PET and single photon emission computed tomography or SPECT), electron microscopy, geophysical imaging, diffraction-limited optics, and radio astronomy [18, 27]. Closely related applications exist in other fields, e.g. the inverse radiotherapy planning problem [29].

[*]Ariela Sofer is partly supported by National Science Foundation grant DMI-9414355.

Reconstruction from projections is an *inverse problem* where the goal is to invert the forward process $T\{\}$ from which the projections were generated in continuously defined space [7]. Defining the set of measurements to be the vector $y \in \Re^N$ with elements $y_j,\ j = 1, \ldots, N$ and the image solution estimate to be the vector $x \in \Re^n$ with elements $x_i, i = 1, \ldots, n$, the inverse problem attempts to solve $x = T^{-1}\{y\}$ in finite-parameter space. Due to imperfections in the data y or in the discrete-space model of the forward process $T\{\}$, the tomographic imaging problem is ill posed and generally requires regularity conditions on the solution estimate of the image in order to obtain useful images [7]. The size of the image and measurement vectors can be quite large (especially when working in 3-D), and consequently the reconstruction problem can be computationally intensive.

The classical approach to the tomographic image reconstruction problem is the *filtered backprojection* method, which is based on direct Fourier inversion. While filtered backprojection is relatively fast, it suffers from a number of limitations. It assumes that the measurements are samples of a Radon transform and thus ignores the (usually) stochastic nature of the projection data and the spatially-variant measurement uncertainty of the instrument. Scans collected over an incomplete range of projection angles do not satisfy necessary sampling requirements for Fourier-based methods and must be reconstructed using iterative techniques. Despite the limitations of Fourier-based reconstruction methods, they remain the predominant mode of image reconstruction in most tomographic applications, largely due to the high computational cost of the alternative *iterative* methods.

A major advantage of iterative reconstruction methods is that they can model the response function of the scanning instrument via a *system matrix* $C = [C_{i,j}] \in \Re^{n \times N}$. An accurate model of the spatially variant response function enables iterative reconstruction methods to recover resolution that had been lost in the projections due to measurement uncertainty. The forward process $T\{\}$ that we wish to invert is approximated in discrete space by the *forward-projection* operation $z = C^T\theta$, where θ is the current estimate of x. A certain class of the iterative methods attempt to solve the (generally inconsistent) system of equations $y = C^T x$, given measurements y and system matrix C. These methods are variants of the *algebraic reconstruction technique* or ART. Since a consistent solution generally does not exist and C is either non-square or non-invertible, these methods actually minimize a least-squares criterion. Other approaches are posed as solution methods for directly minimizing a least-squares functional such as $\left\|y - C^T\theta\right\|_2^2$. Modifications to these methods are often required to achieve a useful solution. These modifications include regularization of the least-squares functional and constraining the image variable estimates to be non-negative, i.e., $\theta \geq 0$.

Another class of iterative methods is based on a maximum likelihood (ML) criterion. The exact nature of these methods depends on the physical model of the imaging process and the statistical model used for the measurements. One of the tomographic imaging problems that has received considerable attention is the Poisson-likelihood reconstruction problem in emission and transmission tomography. The use of a Poisson model has been shown to improve image quality significantly in low-count scanning situations [8, 14]. The Poisson ML reconstruction problem is generally solved using a regularized ML functional and non-negativity constraints on the image variables.

The emphasis of this paper is on the computational structure of the various methods and their parallel implementations. We believe that having an efficient parallel implementation of a tomographic reconstruction method is critical, since the computational expense of an iterative method can be substantial if not prohibitive. Many important issues are omitted from the present discussion, including the theoretical limit properties of the algorithms and the suitability of the various methods for particular applications. Such discussions can be found in the cited references. Here we shall demonstrate that a parallelization scheme that we originally developed for the expectation maximization (EM) algorithm in emission tomography can be applied to almost any tomographic reconstruction method. For this reason, we begin our discussion with the expectation maximization (EM) algorithm for ML reconstruction in emission tomography.

2. ML-EM Algorithm for Emission Tomography

Since the radioactive decay process is known to obey Poisson counting statistics, in emission tomography the measurement vector y can be considered a realization of a Poisson-distributed random vector $\mathbf{y}$ with mean value

$$E\left\{\mathbf{y}\right\} = C^T\hat{x},$$

where $\hat{x}$ is the vector of expected emission counts, i.e., $\hat{x} = E\left\{\mathbf{x}\right\}$. A more realistic model of the measurement vector would include sources of degradation that are beyond the scope of the present discussion. The goal of the ML reconstruction problem is to maximize the log-likelihood objective function

$$f_{ML}\left(\theta\right) = \sum_{j=1}^{N}\left(-\hat{y}_j + y_j \log \hat{y}_j\right), \tag{1}$$

where

$$\hat{y} = C^T\theta \tag{2}$$

is a *forward-projection* vector and θ is our current estimate of $\hat{x}$, subject to non-negativity constraints on the image variables. Thus the constrained optimization problem is

$$\begin{array}{ll} \max & f_{ML}\left(\theta\right) \\ \text{s.t.} & \theta \geq 0. \end{array} \tag{3}$$

The ML-EM algorithm for emission tomography was first presented by Shepp and Vardi [28]. It requires a strictly positive initial solution $\theta^0 > 0$ and consists of the following update equation:

$$\theta_i^{k+1} = \frac{\theta_i^k}{q_i}\sum_{j=1}^{N}\frac{C_{i,j}y_j}{\hat{y}_j^k}, \quad i = 1, \ldots, n, \tag{4}$$

where θ_i^k is the current (k-th iteration) image estimate, $\hat{y}^k$ is the current forward-projection vector, and $q = Ce_N$ is a constant *image-space* vector (we shall refer to vectors with n elements as "image-space" vectors) where $e_N \in \Re^N$ is a vector of ones in *projection space* (we shall refer to vectors with N elements as "projection-space" vectors.) Successive application of the EM update equation (4) produce a sequences of iterates that converges in the limit to the optimal solution of (3). The implementation of a single EM iteration (4) requires first a forward projection $\hat{y}^k = C^T\theta^k$ followed by a *back projection*

$$\nu^k = C\hat{Y}_k^{-1}y, \tag{5}$$

where $\hat{Y}_k = \operatorname{diag}\left\{y_j^k,\ j = 1, \ldots, N\right\}$. The cost of an EM iteration is in fact dominated by the cost of forward and back projecting, since (4) can be rewritten as

$$\theta_i^{k+1} = \frac{\theta_i^k\nu_i^k}{q_i}, \quad i = 1, \ldots, n \tag{6}$$

and the cost of the element-wise vector update (6) is comparatively insignificant. A similar Poisson-likelihood problem exists in transmission tomography, and ML reconstruction methods for that problem (including an EM algorithm [20]) roughly parallel the methods presented here for the emission case.

2.1 Exploiting Sparsity in the Measurement Data

ML reconstructions have been shown to yield improved image quality (over filtered backprojection) when the counting statistics are poor, i.e., when the total number of detection events is low [8, 14]. In low-count scanning situations, the measurement vector y may be 50% sparse or higher (and sometimes much higher), especially in 3-D when the detector sampling is fine. It is interesting to discover that we can exploit the sparsity of the measurement data to substantially reduce the computational cost of performing ML reconstructions.

A *full* forward- or back-projection operation traverses through the entirety of projection space, i.e., through all measurements. For most computational operations, however, it is not necessary to "visit" every measurement. Consider the computation of ν^k, as specified by (5). If $y_j = 0$, the value of $\hat{y}_j^k$ is irrelevant to the computation, and thus the term $C_{ij} y_j / \hat{y}_j^k$ can be completely ignored. Thus, (5) can be rewritten

$$\nu_i^k = \sum_{j: y_j \neq 0} \frac{C_{ij} y_j}{\hat{y}_j^k}. \tag{7}$$

The presence of y_j in the numerator obviates the need for visits to the "unoccupied" coincidence lines. In the function evaluation, the zero-valued subspace of y can also be ignored:

$$f_{ML}\left(\theta^k\right) = -q^T \theta^k + \sum_{j: y_j \neq 0} y_j \log \hat{y}_j^k.$$

We assert that this principle is true for *all* operations involved in ML reconstruction methods. As such, the values of $\hat{y}_j^k$ in the zero-valued subspace of y need never be computed. The subsequent computational savings should be similar to the proportion of density of y. This principle does not extend, however, to reconstruction methods based on least-squares criteria.

3. Computing the Projection Operations

In this section we generalize to consider the computation of the forward- and back-projection operations that were introduced in Section 2. The ML-EM algorithm is known as a *simultaneous update* algorithm since in (2), all elements of $\hat{y}$ are simultaneously updated as are all elements of θ in (6). All simultaneous update algorithms in tomographic imaging (ML or otherwise) are dominated by forward- and back-projection operations in the manner of (2) and (5). Non-simultaneous algorithms update either only a smaller subset of the image variables per iteration (these are known as *coordinate-ascent* methods for a maximization problem) or update only a smaller subset of the forward-projection vector per iteration (these are known as *block-iterative* methods). In non-simultaneous algorithms, the forward- and back-projection operations also effectively dominate the computation, but they are not implemented in the simultaneous-update manner of (2) and (5). In this section we introduce a parallel implementation of the forward- and back-projection operations for simultaneous update methods. In Sections 4.3, 4.4, and 5.1 we shall show how our parallelization scheme can be applied to non-simultaneous methods.

Table 1. Properties affecting the size of the problem.

	thick-slice	thin-slice
image size in voxels	$128^2 \times 23$	$128^2 \times 85$
variables, n	376,882	1.40×10^6
measurements, N	5.36×10^6	6.30×10^7
elements in C	2.02×10^{12}	8.82×10^{13}
density of C	.93%	.35%
nonzeros in C	1.87×10^{10}	3.11×10^{11}
symmetry reduction factor	96	436
base-symmetry chords, p	55,800	144,150
storage required for C	390 MB	1.42 GB

3.1 System-Matrix Sparsity and Symmetries

Iterative tomographic reconstructions are characterized by huge vector spaces and sparsity in the system matrix, especially when working in 3-D. These concepts are illustrated in Table 1 for two reasonably representative 3-D problem sizes that we have worked with in PET. The larger "thin-slice" reconstructions cover the same axial (i.e., vertical) extent as the smaller "thin-slice" reconstructions, but the voxels are nearly cubic in the larger problem. As such, the thin-slice images may be visualized from multiple views, as displayed in Figure 1. The thin-slice problem not only has a greater number of image variables (i.e., voxels) n than the thick-slice problem, but is also more severely overdetermined.

The full size of the system matrix C, being determined by the product nN, is so large that raw storage of the matrix is simply not possible. Fortunately, C is quite sparse and its sparsity pattern is well structured. The jth column of C is essentially a discretization of the scanning instrument's response function for the jth measurement. Each measurement is a blurred projection line and thus the response function for the jth projection line is indicative of line blurring. Beyond a certain truncation point from the center of the line (5% of maximum in our implementation), the value of the response function can be set to zero without

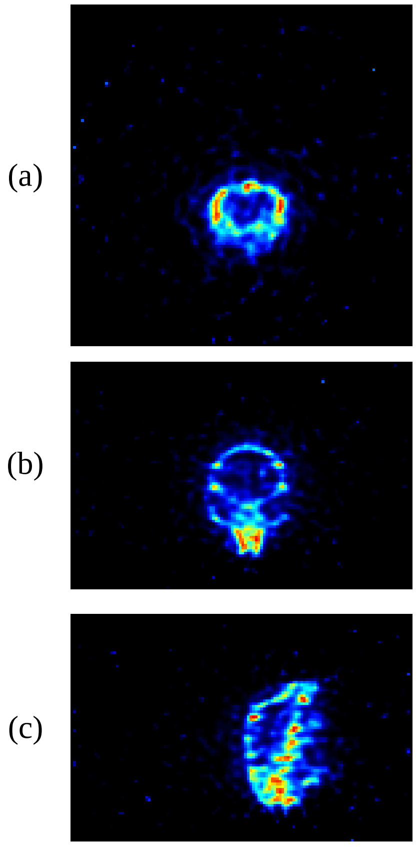

Figure 1. Three views of a “thin-slice” PET reconstruction of a mouse skull: (a) transverse, (b) coronal, (c) sagittal. This image is the result of the OSEM method taken to 10 iterations.

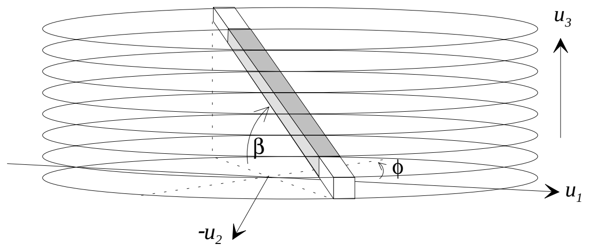

Figure 2. One column of the system matrix is stored as a 3-D *chord* which represents the response function corresponding to a particular measurement. Each measurement location (i.e., column of C) is characterized by a number of physical parameters including azimuthal projection angle ϕ, polar projection angle β, and displacement $\vec{r}$ between chord center and center of the scanner’s field of view ($\vec{r} = 0$ in the figure.)

affecting the outcome of the reconstruction. The jth column can thus be stored as a narrow “chord” around the projection line, as illustrated in Figure 2 for a 3-D tomographic imaging system.

Manageable storage of the system matrix requires exploitation of its symmetries as well as its sparsity. If the projections in y are measured at regularly spaced intervals (of ϕ, β, and $\vec{r}$ - see Figure 2 for definitions of these symbols), as is the case in most commercial PET, SPECT, and CT scanners, then certain symmetry-related columns of C contain redundant information. The information in a symmetry-related chord can be extracted from a base-symmetry chord according to a prescribed set of rules. Much of the computational expense of performing the forward- and back-projection operations in large-scale 3-D reconstructions is due to the costs of reading the base-symmetry chords from disk and of reshuffling the chord data to generate symmetry-related chords from the base-symmetry chords. These “overhead” costs would seriously compromise the accuracy of a complexity analysis based only on multiplications and additions in (2) and (5).

The three types of symmetries in the system matrix are illustrated in Figure 3. *In-plane symmetries* [17] involve reflection and rotation about ϕ and provide an 8-fold storage reduction. The *axial symmetry* [4] involves a reflection about β and yields a 2-fold symmetry (unless $\beta = 0$, in which case there is no gain). *Axial parallel chord redundancies* are the simplest to implement and yield the largest reduction in a 3-D system. The corresponding reduction is $N_R - s$ per symmetry-related group, where N_R is the number of axial sample bins in the 3-D scanner and s is the axial bin separation of the base-symmetry chord. After accounting for the symmetries, only the p base-symmetry chords need to be stored to disk (these quantities are listed in Table 1).

3.2 Parallel Implementation

This section presents a parallelization strategy for computing the dominant operations in tomographic image reconstruction methods. Adopting the notation of Section 2, our goal is to compute a forward projection (2) and a back projection (5) efficiently in parallel. The parallel algorithm permits the forward-projection operation for computing $\hat{y}^k$ to be performed in the same iteration as the back-projection operation for computing ν^k. Similar algorithms for computing forward-only projections and back-only projections are also presented.

Since in most large-scale 3-D reconstructions $N \gg n$, a reasonable data decomposition strategy would involve a partition of projection-space data across the processors. In-

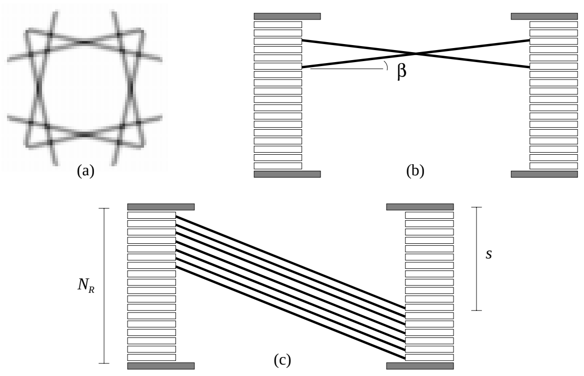

Figure 3. The three symmetries in the system matrix: (a) in-plane symmmetries, (b) axial symmetry, (c) axial parallel chord redundancies.

deed, from the data in Table 1, the cost to memory of storing an 8-bit y vector and a single-precision $\hat{y}$ vector is 315 MB for a thin-slice reconstruction. Due to memory considerations, it may in fact be necessary to partition in projection space and decompose the problem accordingly. In our implementation, the symmetries are defined in projection space. We can therefore decompose the problem by projection-space symmetry and assign each processor certain groups of symmetry-related projections. Recalling Table 1, there are $p = 55,800$ base-symmetry coincidence lines in thick-slice mode and $p = 144,150$ base-symmetry coincidence lines in thin-slice mode. Given these large values of p, decomposing the problem by projection-space symmetries does not limit the scalability of a parallelization scheme based on that data decomposition.

Specifically, we define $N = \sum_{l=1}^{p} N_l(b)$, where p is the number of base-symmetry angles, $b = 1, \ldots, p$ is a particular symmetry index, and $N_l(b)$ is the number of symmetry-related coincidence lines corresponding to base-symmetry index b. We then define

$$\begin{aligned} y &= \left[\begin{array}{cccc} y_{(1)}^T & y_{(2)}^T & \vdots\ \vdots & y_{(p)}^T \end{array}\right]^T, \\ \hat{y} &= \left[\begin{array}{cccc} \hat{y}_{(1)}^T & \hat{y}_{(2)}^T & \vdots\ \vdots & \hat{y}_{(p)}^T \end{array}\right]^T, \\ C &= \left[\begin{array}{cccc} C_{(1)} & C_{(2)} & \vdots\ \vdots & C_{(p)} \end{array}\right], \\ \hat{Y}_{(b)} &= \operatorname{diag}\left(\left[\hat{y}_{(b)}\right]_j,\ j = 1, \ldots, N_l(p)\right), \end{aligned}$$

where $y_{(b)}, \hat{y}_{(b)} \in \Re^{N_l(b)}$ and $C_{(b)} \in \Re^{n \times N_l(b)}$. Image-space vectors such as θ and ν are replicated on all processors, but during a back-projection operation, the vector normally used for ν temporarily stores information representing the back projection of a particular processor's portion of projection space. Thus we introduce $\nu_{(d)} \in \Re^n$, $d = 1, \ldots, D$, where D is the number of processors, and define a parallel version of the forward-and-back projection operation.

Algorithm 1 *forward-and-back project:*

```
   b   $\nu_{(d)} = 0$
f  b   for all $b$ assigned to $d$
f              $\hat{y}_{(b)} = C_{(b)}^T \theta$
   b           $\nu_{(d)} = \nu_{(d)} + C_{(b)} \hat{Y}_{(b)}^{-1} y_{(b)}$
f  b   end for
   b   $\nu = \sum_{d=1}^{D} \nu_{(d)}$   (global summation)
```

The parallelization scheme of Algorithm 1 may be restricted to perform forward-only and back-only projections. A forward-projection algorithm consists of those rows of Algorithm 1 marked with a bold "**f**." Similarly, a back-projection algorithm consists of those rows of Algorithm 1 marked with a bold "**b**." Notice that a global summation is required for the forward-and-back-projection algorithm and the back-projection algorithm but is not required for the forward-projection algorithm. The global summation is the only inter-processor communication operation required in any of the algorithms. All other operations can be performed independently on the processors, including the I/O operations if the processors have local disks.

4. Other ML Methods

In this section we investigate the applicability of the Algorithm 1 parallelization scheme to other methods for performing ML reconstructions. We shall assume that the methods discussed in this section are intended for Poisson-likelihood reconstruction in 3-D emission tomography, although our notation does not explicitly limit the scope of consideration. Similar ML methods exist for the transmission tomography problem [8, 21].

4.1 Regularized ML-EM Methods

Due to ill-posedness of the reconstruction problem, the objective function often must be regularized in order to obtain useful images [7]. The regularized ML objective function is

$$f_{RML}(\theta) = f_{ML}(\theta) - \gamma R(\theta), \tag{8}$$

where $R(\theta)$ is a smoothing function that penalizes local roughness in the image, and γ is a hyperparameter that controls the trade-off between agreement with the data (i.e., optimization of $f_{ML}(\theta)$) and agreement with a priori knowledge about the solution (i.e., optimization of $R(\theta)$). The constrained optimization problem in regularized ML-EM reconstruction is

$$\begin{array}{ll} \max & f_{RML}(\theta) \\ \text{s.t.} & \theta \geq 0. \end{array} \tag{9}$$

A number of simultaneous-update algorithms have been proposed for regularized ML reconstruction under restricted classes of priors. One such method, which requires an upper threshold for γ and a finite bound on $\|\nabla R(\theta)\|_\infty$, is Green's "One Step Late" (OSL) procedure [10]. One iteration of the OSL algorithm is computed as

$$\theta_i^{k+1} = \frac{\theta_i^k \nu_i^k}{q_i + \gamma \frac{\partial}{\partial\theta_i} R\left(\theta^k\right)}, \tag{10}$$

where ν^k can be computed in parallel in the manner of Algorithm 1. The computation of $\nabla R\left(\theta^k\right)$ does not involve any forward- or back-projection operations and is of negligible cost for most practical smoothing functions.

4.2 Line Search Acceleration

The EM update equation (4) may be rewritten as a scaled steepest ascent iteration

$$\theta^{k+1} = \theta^k + W_k \nabla f_{ML}\left(\theta^k\right),$$

where $W_k = \operatorname{diag}\left\{\theta_i^k / q_i,\ i = 1, \ldots, n\right\}$ and $\nabla f_{ML}\left(\theta^k\right) = \nu^k - q$ is the gradient of (1). Likewise, the OSL update equation (10) may be rewritten

$$\theta^{k+1} = \theta^k + W_k \nabla f_{RML}\left(\theta^k\right),$$

where $\nabla f_{ML}\left(\theta^k\right) = \nu^k - q - \gamma \nabla R\left(\theta^k\right)$ is the gradient of (8). It has been observed that as the optimal solution is approached in EM, the distance moved at each iteration becomes very small [17]. For this reason, line searches have been proposed to accelerate the convergence of ML-EM [17] and regularized ML-EM [21]. The regularized ML-EM update can be enhanced with a line search for the steplength α_k that ensures feasibility and approximately maximizes the function along the search direction. (Thus for the OSL update, $\alpha_k$approximately maximizes $f\left(\theta^k + \alpha W_k \nabla f_{RML}\left(\theta^k\right)\right)$ over α.) The enhanced update is then

$$\theta^{k+1} = \theta^k + \alpha_k W_k \nabla f_{RML}\left(\theta^k\right). \tag{11}$$

The cost of the update (11) is dominated by the forward-and-back projection to compute ν^k, since $\nabla R\left(\theta^k\right)$ is easy to compute and the line search can be implemented inexpensively for this problem. The computation of ν^k can then be computed in the manner of Algorithm 1. Lange et al. have proved that regularized ML-EM algorithms, enhanced with a line search, converge to the optimal solution of (9) [21].

4.3 Block-Iterative ML-EM Algorithms

A number of authors have recently proposed *block-iterative* methods to improve the convergence of (unregularized) ML-EM [2, **?**, 13]. These methods partition the projection space vectors into m (unusually non-overlapping) blocks $S_l,\ l = 1, \ldots, m$. Each iteration requires m passes corresponding to the m blocks, where the entire image is updated with each pass, but the update is performed only on the current (lth) block of projection space. A popular example of this is Hudson and Larkin's "ordered subsets EM" (OSEM) algorithm [13], in which the lth pass of the kth iteration consists of the following operations:

$$\begin{aligned} \hat{y}_j^{k,l} &= \sum_{i=1}^{n} C_{i,j} \theta^{k,l}, \quad j \in S_{l,} \\ \theta_i^{k,l+1} &= \frac{\theta_i^{k,l}}{q_i^l} \sum_{j \in S_l} \frac{C_{i,j} y_j}{\hat{y}_j^{k,l}}, \quad i = 1, \ldots, n, \end{aligned}$$

where the outer iteration increments after the mth pass, i.e., $\theta^{k+1,1} \equiv \theta^{k,m+1}$, and $q^l = \sum_{j \in S_l} C_{i,j}$.

The two operations above can be performed in a single parallelized forward-and-back-projection operation over the S_l block of projection space. The only complication is that we now have two decompositions of projection space, one for partitioning the block subsets and the other for the symmetry-related groups which are assigned to the various processors. Usually m will be a small number, say 5, and since the number of base-symmetry projections p is generally quite large, the subset of base-symmetry projections assigned to a particular processor can simply be further decomposed into m processor-specific blocks for the OSEM algorithm. This subset-partitioning strategy also attempts to ensure an evenly balanced load across the processors within each OSEM block, although load balancing is more difficult in a block-update algorithm than in a simultaneous-update algorithm.

When the number of block subsets m is small, one pass of the OSEM algorithm increases $f_{ML}(\theta)$ by nearly the same amount as one iteration of ML-EM in the first several iterations. Since the cost one pass of the OSEM algorithm is approximately $1/m$ the cost of an EM iteration, the early-iteration convergence is improved. The OSEM algorithm does not converge to a solution of (3), but rather to a limit cycle of N distinct solutions that are within a proximity to the optimal solution. The "row action maximum likelihood algorithm" (RAMLA) of Browne and DePierro [2] is computationally similar to OSEM and converges to an optimal solution of (3). To our knowledge, no block-iterative algorithm has been proven to converge to the optimal solution of the regularized problem (9).

4.4 Coordinate-Ascent Methods

Coordinate-ascent methods can be considered the "dual" of block-iterative methods in that they update only a single voxel or a group of voxels with each iteration. Coordinate-ascent algorithms are generally posed as solution methods for the *regularized* ML-EM algorithm. Recent papers have shown that carefully designed coordinate-ascent algorithms can result in faster convergence [1, 8].

Here we consider the single-coordinate update algorithm of Bouman et al. [1]. Defining $e^i \in \Re^n$ to be a vector with zeros in all elements except the ith, for which $e^i_i = 1$, one iteration of the algorithm moves only the single voxel indexed $i_k = (k \bmod n) + 1$, according to

$$\theta^{k+1} = \theta^k + e^{i_k}\Delta\theta_{i_k} = \theta^k + e^{i_k}\left(\theta^{k+1}_{i_k} - \theta^k_{i_k}\right).$$

The kth iteration attempts to minimize the regularized ML objective function over a single variable θ_{i_k}, or

$$\theta^{k+1}_{i_k} = \arg\min_{\xi \geq 0} f_{RML}\left(\theta^k + e^{i_k}\left(\xi - \theta^k_{i_k}\right)\right).$$

Taking a one-dimensional Taylor series expansion of the ML component of f_{RML} about the point θ^k_i yields a regularized quadratic approximation

$$\begin{aligned} & f_{RML}\left(\theta^k + e^{i_k}\left(\xi - \theta^k_{i_k}\right)\right) \approx Q(\xi) \\ = \; & \varphi_1\left(\xi - \theta^k_{i_k}\right) + \varphi_2\left(\xi - \theta^k_{i_k}\right)^2 + \\ & \gamma R\left(\theta^k + e^{i_k}\left(\xi - \theta^k_{i_k}\right)\right) + \text{const.} \end{aligned}$$

The quadratic $Q(\xi)$ can be minimized via a one-dimensional Newton method, or approximately minimized by taking a single Newton step.

Computation of the quadratic coefficients φ_1 and φ_2 dominates the cost of one iteration of this algorithm. Let us assume that at the start of the kth iteration, the forward-projection vector $\hat{y}$ was updated after the $(k-1)$st iteration. The coefficient φ_1 is computed via a dot product between the i_kth column of C and a projection-space vector $\zeta = e_N - \hat{Y}^{-1}y$, i.e. $\varphi_1 = C^T_{i_k}\zeta$. The computation of φ_2 also involves a dot product, although in this case the elements of the system matrix and projection-space vector are squared. (We shall encounter a similar operation in Section 5.3 with equation (17).) The important point here is that our general parallelization scheme fits surprisingly well to the calculation of φ_1 and φ_2. If projection space is partitioned, every processor can simply compute its portion of the dot product and then perform a global summation. Following the update of θ, the forward-projection vector $\hat{y}$ can be updated with the simple vector addition operation

$$\hat{y}^{k+1} = \hat{y}^k + \theta^{k+1}_{i_k}C_i,$$

which can easily be performed in parallel since the partitioning of projection-space vectors $\hat{y}$ and C_i is identical.

A couple of caveats are in order regarding the use of a projection-space parallelization scheme for coordinate-ascent algorithms. First, every single-coordinate update requires two scalar global summations, one each for the calculation of φ_1 and φ_2. Since n iterations are required for a single update of the entire image, every complete update requires $2n$ scalar global summations. This is in contrast to the lone n-element global summation at the end of every iteration of a simultaneous-update algorithm. The numerous global summations required by a coordinate-ascent algorithm can incur significant communication latency costs. Second, the partitioning of projection space into symmetry-related groups is much more difficult to implement in a coordinate-ascent algorithm. Incorporation of radially-symmetric voxel basis functions [22] and on-the-fly computation of the system matrix may be more appropriate for this class of algorithm.

4.5 Interior-Point Methods

Interior-point methods are constrained optimization methods that maintain strict feasibility (i.e., $\theta > 0$) at every iteration and approach the constraint boundaries only in the limit and from within the interior of the feasible region. They avoid the combinatorial difficulties associated with other constrained optimization methods such as active-set methods. Although interior-point methods were originally developed in the 1960's by Fiacco and McCormick, recent developments in linear programming [16, 23] and nonlinear programming [25] have revived interest in them within the optimization community. In [15], we proposed two interior-point methods for the regularized ML reconstruction problem: a specialized *logarithmic barrier method* and a *primal-dual method*.

The logarithmic barrier method solves the constrained problem by solving a sequence of unconstrained subproblems. Each subproblem is parameterized by a barrier parameter μ_k and involves minimization of the augmented objective function

$$F(\theta, \mu_k) = f(\theta) - \sum_{i=1}^{n} \log\theta_i,$$

where $f(\theta) \equiv -f_{RML}(\theta)$. The barrier penalty term $\sum_{i=1}^{n} \log\theta_i$ enforces strict feasibility. These subproblems are solved for a descending sequence of barrier parameters $0 < \mu_k \to 0$. Each subproblem solution satisfies a perturbed version of the Karush-Kuhn-Tucker (KKT) optimality conditions [9, 26]. Under mild conditions, it can be shown that as $\mu \to 0$, the subproblem solutions approach the optimal solution to the constrained problem (9) [9]. The logarithmic barrier method updates only the primal (image) variables θ.

The *primal-dual* method is closely related to the logarithmic barrier method. In both the logarithmic barrier and primal-dual methods, a sequence of subproblems parameterized by a descending sequence $0 < \mu_k \to 0$ are solved approximately, and the corresponding subproblem solutions satisfy the same perturbed KKT conditions. In contrast to the logarithmic barrier method however, the primal-dual method actively updates both the primal image variables and the Lagrangian *dual variables*. The actively-computed dual variables enable the primal-dual method to converge more rapidly than the logarithmic barrier method.

In [15], we demonstrate that interior-point methods developed for the (regularized or unregularized) ML reconstruction problem converge significantly faster than EM algorithms and require significantly less computation. Although we do not discuss the details of our methods here, we feel it worth mentioning that the dominant computational operations are forward-and-back-projection operations that are readily parallelizable as per Algorithm 1. There are essentially three operations that dominate the interior point methods: 1) a matrix-vector product that is computed during a preconditioned conjugate gradient iteration (in Section 5.3 we shall discover that this is equivalent to a forward-and-back projection); 2) computation of a diagonal matrix preconditioner in the aforementioned preconditioned conjugate gradient (in Section 5.3 we shall show with equation (17) that this is roughly equivalent to a back-projection), and 3) updating the gradient, which is essentially a forward-and-back projection.

5. Least-Squares Methods

The simplified Poisson-measurement model that underlies the ML reconstruction methods may not be accurate in certain tomographic imaging problems. For these problems, a more appropriate iterative approach is to minimize a least-squares functional such as $\frac{1}{2}\left\|y - C^T\theta\right\|_2^2$ which may or may not be subject to non-negativity constraints. Such a formulation is actually applicable to *all* tomographic imaging problems, since the forward process is always modeled as $z = C^T\theta$, regardless of the statistical interpretation. We shall discover that the least-squares methods are also dominated by forward- and back-projection operations and that the parallelization strategy presented in Section 3 is also applicable to the vast majority of least-squares methods.

5.1 ART-Family Methods

The *algebraic reconstruction technique* (ART) is an iterative method that attempts to solve the inconsistent and generally overdetermined system of equations $C^T x = y$ [11, 12]. ART is a fully sequential-update algorithm, meaning that the kth iteration of ART updates the entire image estimate but operates on only the j_kth measurement. The ART update is given by

$$\theta^{k+1} = \theta^k + \sigma_k \frac{y_{j_k - C_{j_k}^T \theta^k}}{\|A_{j_k}\|_2^2} A_{j_k},$$

where $A_{i,j} = C_{i,j}/q_i,\ \forall i, j,\ A_{j_k}$ is the j_kth column vector of $A = [A_{i,j}]$, θ^k is the current (kth iteration) estimate of x, σ_k is a scalar *relaxation parameter*, and j_k is the projection element of the current iteration. The update is usually cyclic, so that $j_k = (k \bmod N) + 1$. In the inconsistent case, when $\lim\limits_{k\to 0} \sigma_k = 0$, ART converges to a solution of the minimum norm problem

$$\min \sum_{j=1}^{N} \frac{y_j - C_j^T\theta}{\|A_{j_k}\|_2^2}.$$

Without relaxation, ART converges to a limit cycle of N vectors in a vicinity of the minimum-norm solution.

Unfortunately, the parallelization strategy of Section 3 cannot be applied to a purely sequential-update algorithm such as ART. Block-iterative variants of ART such as the *variable-block ART* algorithm [3], however, are indeed amenable to our parallelization scheme.

5.2 Generalized Landweber Iterations

The generalized Landweber iteration is a scaled steepest descent method, where each iteration is given by the simultaneous update equation

$$\theta^{k+1} = \theta^k + M_k^{-1} \nabla f\left(\theta^k\right), \tag{12}$$

where $\nabla f\left(\theta^k\right)$ is the gradient of the objective function evaluated at the current point θ^k, and M_k is a shaping matrix or *preconditioner* [6, 30]. We shall discuss preconditioners in Section 5.3.

A common objective function in such problems is the *weighted least squares* (WLS) objective

$$\begin{aligned} f_{WLS}(\theta) &= \frac{1}{2}\left\|D^{-\frac{1}{2}}\left(y - C^T\theta\right)\right\|_2^2 & (13)\\ &= \frac{1}{2}\left(y - C^T\theta\right)^T D^{-1}\left(y - C^T\theta\right), & (14)\end{aligned}$$

where D is typically a diagonal weighting matrix with diagonal elements that correspond to estimates of the noise variance of the measurements [6]. When the measurement vector is assumed to be a realization of a Gaussian-distributed random vector, a solution that minimizes $f_{WLS}(\theta)$ is also a ML solution. Since the gradient of (13) is given by

$$\nabla f_{WLS}(\theta) = CD^{-1}\left(C^T\theta - y\right),$$

the cost of one iteration of (12) is equal to that of a forward projection $z = C^T \theta$ followed by a backprojection $C(z - y)$, plus the cost of forming and applying the preconditioner. The forward- and back-projection operations can be decomposed in parallel and computed in the same loop, as per Algorithm 1.

5.3 Conjugate Gradient Iterations

The unconstrained WLS objective function $f_{WLS}(\theta)$ can also be minimized by solving the *normal equations*

$$\left(CD^{-1}C^T\right)\theta = C^T D^{-1} y. \tag{15}$$

The linear conjugate gradient (CG) method for solving symmetric, positive semidefinite systems of equations such as (15) [26] has been applied to the WLS reconstruction problem [5, 18]. The quality of the estimate for θ improves with each CG iteration. As such, an inexact solution for θ can be obtained from the CG method within (hopefully) a reasonable number of iterations. The cost of a single CG iteration is dominated by a matrix-vector product of the form

$$CD^{-1}C^T v, \tag{16}$$

where v is a working vector in the CG algorithm. The matrix-vector product (16) can be implemented as a parallelized forward projection $z = C^T v$ followed by a parallelized back projection $CD^{-1}z$. Alternatively, (16) can be implemented in parallel as a forward-and-back projection in the manner of Algorithm 1.

When working in exact arithmetic, the CG method converges in n iterations. In inexact arithmetic, the convergence rate of the CG method depends upon the condition number of the matrix $CD^{-1}C^T$ [26]. The convergence rate of the CG method can be improved through the use of a preconditioning matrix M_k. Specifically, the *preconditioned CG method* solves the transformed system of equations

$$M_k^{-1}\left(CD^{-1}C^T\right)\theta = M_k^{-1} C^T D^{-1} y.$$

The implementation of the preconditioned CG method is quite similar to that of the "plain" linear CG method, although every iteration of the preconditioned CG method requires an application of the preconditioner in the form $M_k^{-1}w$, where w is a working vector in the preconditioned CG algorithm.

Numerous methods have been proposed for preconditioning the least-squares problem. In general, M_k should either be easy to invert or products of the form $M_k^{-1}\nabla f\left(\theta^k\right)$ should be easy to apply, and transformation of the system of equations via the preconditioner should improve convergence. One such preconditioner is the diagonal of the Hessian of the objective function, which in the case of $\nabla^2 f_{WLS}(\theta) = CD^{-1}C^T$ (where D is a diagonal matrix) is given by

$$[M_k]_i = \sum_{j=1}^{N} \frac{C_{i,j}^2}{D_{i,i}}, \quad i = 1, \ldots, n. \tag{17}$$

Equation (17) is similar to the computation of φ_2 in the coordinate-ascent algorithm of Section 4.4 and to the computation of the diagonal preconditioner in the interior-point methods of Section 4.5. All of these operations are essentially implemented with a back-projection operation that is a bit more expensive than a "standard" back-projection due to the squaring of the system-matrix elements. Other preconditioners attempt to exploit the approximately block-Toeplitz structure of the Hessian of the least-squares objective function. Such preconditioners are generally implemented via multidimensional FFT's [5] or inverse filtering operations [6] which, if sufficiently expensive, may require a parallelization strategy that is outside the framework of Algorithm 1.

The solutions that minimize the least-squares objective function $f_{LS}(\theta)$ and the weighted least-squares objective function $f_{WLS}(\theta)$ may be noisy due to ill-posedness. As with the ML methods, the remedy here is regularization. The regularized, weighted least-squares objective function is given by $f_{RWLS}(\theta) = f_{WLS}(\theta) + \gamma R(\theta)$. The unconstrained problem of minimizing $f_{RWLS}(\theta)$ can be solved via a nonlinear CG iteration [19]. Each iteration of the nonlinear CG method requires a gradient calculation $\nabla f_{RWLS}(\theta) = \nabla f_{WLS}(\theta) + \gamma \nabla R(\theta)$, which is dominated by the cost of $\nabla f_{WLS}(\theta)$. Recalling from Section 5.2 that $\nabla f_{WLS}(\theta)$ is computed by a forward-and-back projection, it is clear that Algorithm 1 can also be used for the nonlinear CG. The convergence rate of most nonlinear CG methods is only marginally better than that of steepest descent methods. A faster solution of the unconstrained minimizer of $f_{RWLS}(\theta)$ may be obtained via a *truncated-Newton* method (see [24, 26].)

5.4 Interior-Point Methods

Although the least-squares objective functions $f_{LS}(\theta)$, $f_{WLS}(\theta)$, and $f_{RWLS}(\theta)$ are well defined outside of the non-negative orthant, negative solutions have no physical interpretation in most tomographic applications. As such, negative elements in the solution vector contribute mainly to image noise. To improve image quality, the regularized weighted least-squared problem (for example) can then be posed as the constrained minimization problem

$$\begin{array}{ll} \min & f_{RWLS}(\theta) \\ \text{s.t.} & \theta \geq 0. \end{array} \tag{18}$$

The problem (18), being similar in nature to the problem (9), is also well suited for interior-point methodology. Interior-point, least-squares reconstructions are also dominated by forward-and-back projections and back-only projections. The interested reader is referred to [15]. Once again, the computational structure of Algorithm 1 is applicable.

6. Performance Tests

In this section we report on the results of a number of computational tests that were designed to measure the performance of Algorithm 1 as applied to a simultaneous-update method (EM) and a block-iterative method (OSEM). In all cases, the methods tested exploit the sparsity and symmetries in the system matrix as described in Section 3.1. For all tests reported in this section, a single dataset consisting of 2.5M counts collected from a small animal PET scanner was used. Reconstructions were performed in "thick-slice" and "thin-slice" mode according to the dimensions of Table 1.

Table 2. Cost comparison between full and sparse EM iterations on 2.5M-count study using 10 120-MHz SP processors.

	thick-slice	thin-slice
sparse gradient eval.	3.42 min.	7.23 min.
full gradient eval.	6.74 min.	2.9 hrs.
density of y	0.466	.0397
sparse/full ratio	0.507	.0415

We first consider the effect of exploiting measurement-data sparsity in the EM algorithm, as described in Section 2.1. Table 2 compares the cost of the sparse implementation of Algorithm 1 (as per (7)) with the cost of a "full" implementation of Algorithm 1. Noting that the sparse implementation skips the columns of C corresponding to the zero-valued elements of y, we would expect the improvement in the sparse implementation to be similar to the density of y. Indeed, Table 2 suggests that this is the case.

The parallel performance of the sparse implementation of Algorithm 1 is plotted in Figure 4. Since all base-symmetry chords must be read from disk every iteration, the I/O throughput requirement can be high. On architectures such as the IBM SP2 with fast local disks, the processors can act independently until the global summation at the end of Algorithm 1. As such, the strong performance indicated by Figure 4, and in particular, by the thick-slice relative efficiency plot, is not surprising. The favorable efficiency plot also suggests that the sparse implementation of Algorithm 1 does not seriously degrade the load balance between the processors. Due to the high per-node memory requirements of thin-slice reconstructions, we were unable to perform a single-node test in thin-slice mode, and consequently thin-slice efficiency numbers are not available.

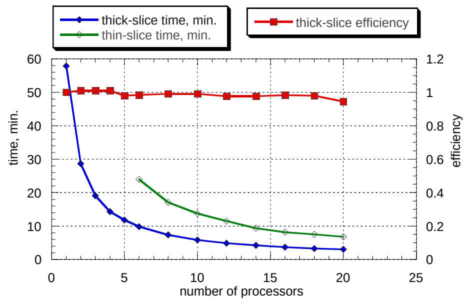

Figure 4. Cost of one EM iteration (both thick-slice and thin-slice) and relative efficiency (thick-slice only) as a function of number of processors. Tests performed on a 2.5M-count study on 66-MHz IBM RISC/6000 SP processors.

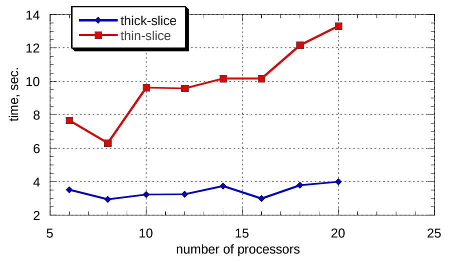

Figure 5. Cost of global summation at the end of EM algorithm as a function of number of processors. Test conditions are identical to Figure 4.

We have observed that a (usually small) disparity exists between the cost of a back-only projection and that of a forward-only projection. This disparity is attributable to the cost of the global summation, which is performed by the back-only projections but not the forward-only projections. In thin-slice reconstructions, this disparity grows as the number of processors increases. We thus suspect that the relative efficiency of thin-slice implementations of Algorithm 1 is not as strong as in the thick-slice case, due to the higher cost of the global summation. Figure 5 compares the cost of the global summation operation in thick-slice and thin-slice mode.

Finally we consider the parallel performance of block-iterative algorithms such as OSEM. In Figure 6, the computational cost and relative efficiency of one thick-slice OSEM iteration is plotted as a function of number of processors. A comparison of the OSEM plot with the earlier EM plot (Figure 4) reveals that the performance of OSEM clearly

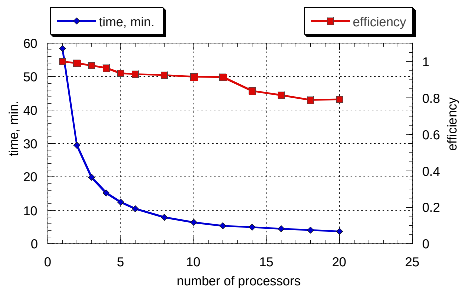

Figure 6. Cost and relative efficiency of a single OSEM iteration consisting of 5 passes (i.e., 5 subsets) as a function of number of processors. Tests performed on a 2.5M-count study in thick-slice mode using 66-MHz IBM RISC/6000 SP processors.

does not match that of EM. In the present (Figure 6) test, one OSEM iteration consists of five passes corresponding to five subsets. After each pass, a global summation is performed, so that the interprocessor communication costs in a 5-subset block-iterative algorithm are 5 times the costs of a simultaneous-update iteration. Another contributing factor to performance degradation in block-iterative algorithms is slightly worsened load balance due to the double partitioning of projection-space vectors.

7. Conclusion

As computer technology continues to advance, the advantages posed by iterative reconstruction methods will make them more widely used in many applications. On large-scale 3-D reconstruction problems, iterative methods remain computationally challenging and are taxing to numerous resources, including CPU, disk, and memory. The motivation for parallel computing methods for the iterative reconstruction problem is therefore clear. Over the years, numerous authors have proposed a multitude of methods, which can be roughly categorized into three computational categories: simultaneous update methods, block-iterative methods , and coordinate ascent methods.

The parallelization scheme of Algorithm 1 is based upon a data decomposition strategy that partitions the projection-space vectors. The applicability of Algorithm 1 to simultaneous-update algorithms is straightforward. Since Algorithm 1 requires no interprocessor communication until the last operation (a global summation) parallel performance is understandably strong. Algorithm 1 is capable of exploiting the sparsity and symmetries present in the system matrix in a computationally efficient manner. In methods that optimize a Poisson-likelihood criterion, sparsity of the measurement vector can be exploited to achieve a significant improvement in computational performance. With the exception of sequential update algorithms such as ART, our parallelization strategy of partitioning the measurement-space vectors can be extended to non-simultaneous methods.

8. Acknowledgments

The computational work in this paper was performed on the IBM RS/6000 SP2 parallel computer at the Division of Computer Research and Technology, National Institutes of Health. We are grateful to Mike Green and Jürgen Seidel of the Department of Nuclear Medicine, National Institutes of Health, for kindly providing us with the small animal data and Monte Carlo simulation data.

References

[1] C. Bouman and K. Sauer, “A unified approach to statistical tomography using coordinate descent optimization,” *IEEE Trans. Imag. Proc.*, vol. 5 (1996), pp. 480-492.

[2] J. Browne and A.R. DePierro, “A row-action alternative to the EM algorithm for maximizing likelihoods in emission tomography,” *IEEE Trans. Med. Imag.*, vol. 15, no. 4 (1996), pp. 687-699.

[3] Y. censor and S.A. Zenios, *Parallel Optimization: Theory, Algorithms, and Applications*, Oxford Press, New York, 1997.

[4] C.M. Chen, S.Y. Lee, and Z.H. Cho, “On parallelizing the EM algorithm for PET image reconstruction,” IEEE Trans. Parallel Dist. Sys., vol. 5 (1994), pp. 860-873.

[5] G. Chinn and S-C. Huang, “A general class of preconditioners for statistical iterative reconstruction of emission computed tomography,” *IEEE Trans. Med. Imag.*, vol. 16 (1997), pp. 1-10.

[6] N.H. Clinthorne, T-S. Pan, P-C. Chiao, W.L. Rogers, and J.A. Stamos, “Preconditioning methods for improved convergence rates in iterative reconstructions”, *IEEE Trans. Med. Imag.*, vol. 12 (1993), pp. 78-83.

[7] G. Demoment, “Image reconstruction and restoration: overview of common estimation structures and problems,” *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 37 (1989), pp. 2024-2036.

[8] J.A. Fessler, E.P. Ficaro, N.H. Clinthorne, and K. Lange, “Grouped-coordinate ascent algorithms for penalized-likelihood transmission image reconstruction,” *IEEE Trans. Med. Imag.*, vol. 16 (1997), pp. 166-175.

[9] A.V. Fiacco and G.P. McCormick, *Nonlinear Programming: Sequential Unconstrained Minimization Techniques,* John Wiley and Sons, New York, 1968.

[10] P.J. Green, “On use of the EM algorithm for penalized likelihood estimation,” *J. Roy. Stat. Soc.* B, vol. 52 (1990), pp. 443-452.

[11] G.T. Herman, *Image Reconstruction from Projections: the Fundamentals of Computerized Tomography,* Academic Press, New York, 1980.

[12] G.T. Herman and L.B. Meyer, “Algebraic reconstruction techniques can be made computationally efficient,” *IEEE Trans. Med. Imag.*, vol. 12 (1993), pp. 600-609.

[13] H.M. Hudson and R.S. Larkin, “Accelerated image reconstruction using ordered subsets of projection data,” *IEEE Trans. Med. Imag.*, vol. 13 (1994), pp. 601-609.

[14] C.A. Johnson, J. Seidel, R.E. Carson, W.R. Gandler, A. Sofer, M.V. Green, and M.E. Daube-Witherspoon, “Evaluation of 3D reconstruction algorithms for a small animal PET camera,” *IEEE Trans Nucl. Sci.*, vol. 44. (1997), pp. 1303-1308.

[15] C.A. Johnson, *Nonlinear Optimization for Volume PET Reconstructions*, Ph.D. dissertation, Department of Operations Research and Engineering, George Mason University, Fairfax, VA, 1997.

[16] N. Karmarkar, “A new polynomial-time algorithm for linear programming,” *Combinatorica*, vol. 4, (1984), pp.49-57.

[17] L. Kaufman, “Implementing and accelerating the EM algorithm for positron emission tomography,” *IEEE Trans. Med. Imag.*, vol. 6 (1987), pp. 37-51.

[18] S. Kawata and O. Nalcioglu, “Constrained iterative reconstruction by the conjugate gradient method,” *IEEE. Trans. Med. Imag.*, vol. 4 (1985), pp. 65-71.

[19] D.S. Lalush and B.M.W. Tsui, “A fast and stable maximum a posteriori conjugate gradient reconstruction algorithm,” *Med. Phys.*, vol. 22 (1995), pp. 1273-1284.

[20] K. Lange and R. Carson, “EM Reconstruction Algorithms for Emission and Transmission Tomography,” *J. Comp. Assist. Tomogr.*, vol. 8 (1984), pp. 306-316.

[21] K. Lange, M. Bahn, and R. Little, “A theoretical study of some maximum likelihood algorithms for emission and transmission tomography,” *IEEE Trans. Med. Imag.*, vol. 6 (1987), pp. 106-114.

[22] R.M. Lewitt, “Alternatives to voxels for image representation in iterative reconstruction algorithms,” *Phys. Med. Biol.*, vol. 37 (1992), pp. 705-716.

[23] R.D.C. Monteiro and I. Adler, “Interior point path following primal-dual algorithms: Part I: Linear programming,” *Math. Programming,* vol. 44 (1989), pp. 27-41.

[24] S.G. Nash and A. Sofer, “Block truncated-Newton methods for parallel optimization,” *Math. Programming.* vol. 45 (1989), pp. 529-546.

[25] S.G. Nash and A. Sofer, “A barrier method for large-scale constrained optimizaton,” *ORSA J. Comp.*, vol. 5, no. 1 (1993), pp. 40-53.

[26] S.G. Nash and A. Sofer, *Linear and Nonlinear Programming*, McGraw-Hill, New York, 1996.

[27] A. Rosenfeld and A.C. Kak, *Digital Picture Processing,* Academic Press, San Diego, 1982.

[28] L.A. Shepp and Y. Vardi, “Maximum Likelihood Reconstruction for Emission Tomography,” *IEEE Trans Med. Imag.*, vol. 1 (1982), pp. 113-122.

[29] L. Xing and T.Y. Chen, “Iterative methods for inverse treatment planning,” *Phys. Med. Biol.*, vol. 41 (1996), p. 2107-2123.

[30] X.L. Xu, J.S. Liow, and S.C. Strother, “Iterative algebraic reconstruction algorithms for emission computed tomography: a united framework and its application to positron emission tomography,” *Med. Phys.*, vol. 20, (1993), pp. 1675-1684.