BICAV: A Block-Iterative, Parallel Algorithm for Sparse Systems with Pixel-Related Weighting

Yair Censor

Department of Mathematics, University of Haifa, Mt. Carmel, Haifa 31905, Israel. (yair@math.haifa.ac.il)

Dan Gordon

Department of Computer Science, University of Haifa, Haifa 31905, Israel. (gordon@cs.haifa.ac.il)

Rachel Gordon

Department of Aerospace Engineering, The Technion – Israel Institute of Technology, Haifa 32000, Israel. (rgordon@tx.technion.ac.il)

May 15, 2000

First Revision: January 9, 2001
Second Revision: April 22, 2001
Third Revision: July 9, 2001

Abstract

Component averaging (CAV) was recently introduced by Censor, Gordon and Gordon as a new iterative parallel technique suitable for large and sparse unstructured systems of linear equations. Based on earlier work of Byrne and Censor, it uses diagonal weighting matrices, with pixel-related weights determined by the sparsity of the system matrix. CAV is inherently parallel (similarly to the very slowly converging Cimmino method) but its practical convergence on problems of image reconstruction from projections is similar to that of ART (Kaczmarz's sequential row-action algorithm). Parallel techniques are becoming more important for practical image reconstruction since they are relevant not only for supercomputers but also for the increasingly prevalent multiprocessor workstations. This study reports on experimental results with a block-iterative version of component averaging, called BICAV. When BICAV is optimized for block size and relaxation parameters, its very first iterates are far superior to those of CAV, and more or less on a par with ART. Similarly to CAV, BICAV is also inherently parallel. The fast convergence is demonstrated on problems of image reconstruction from projections, using the SNARK93 image reconstruction software package. Detailed plots of various measures of convergence, and reconstructed images are presented.

Keywords. *Block-iterative, component averaging, image reconstruction, parallel processing, pixel-related weighting, sparse systems.*

1 Introduction

Problems of image reconstruction from projections, after suitable discretization, can be represented by a system of linear equations

$$Ax = b, \tag{1.1}$$

where A is an $m \times n$ system matrix, m is the number of equations, n the size of the (unknown) image vector x, and b is the vector (of size m) of readings. In practice, the system (1.1) is often inconsistent, and one usually seeks a point $x^* \in \mathbb{R}^n$ which minimizes some predetermined optimization criterion. Even then, the problem is frequently ill-posed and there may be more than one optimal solution. The standard approach to dealing with that problem is via regularization, i.e., by aiming at an optimal solution with a minimal Euclidean norm.

Equation (1.1) may also be viewed as a special case of the *convex feasibility problem*, which is to find a point $x^* \in C = \bigcap_{i=1}^m C_i$, where C is the intersection of finitely many closed convex sets $C_i \subseteq \mathbb{R}^n$, $i = 1, 2, \ldots, m$, in the Euclidean space. In the inconsistent case, when C is empty, some iterative projection algorithms (mostly of the simultaneous type) converge to a point x^* which minimizes a certain *proximity function*. In image recovery

the task of estimating an image from the measurements of one or more signals, physically related to it, can often be modelled by a (not necessarily linear) convex feasibility problem, see, e.g., Combettes [20] or Stark and Yang [29]. Algorithmic schemes for this problem are, in general, either *sequential* or *simultaneous* or *block-iterative*, see, e.g., Censor and Zenios [18, Section 1.3] for a classification of projection algorithms into such classes, and the review paper of Bauschke and Borwein [4] for a variety of specific algorithms of these types.

A typical example of a sequential method for solving (1.1) is the row-action ART (Algebraic Reconstruction Technique)—see, e.g., Herman [21], Censor and Zenios [18, Algorithm 5.4.3]. Originating with Kaczmarz [24], this algorithm cyclically projects the current iterate onto the hyperplanes represented by the rows of the system (1.1). In the inconsistent case, it converges cyclically (Tanabe [30]), and for a fixed positive relaxation parameter, the limits of the cycles lie within a bounded distance from the geometric least-squares solution—see Censor, Eggermont and Gordon [15]. This distance approaches zero as the relaxation parameter tends to zero, so ART with small relaxation parameters can also be used to approach the geometric least squares solution arbitrarily closely. In practice, ART performs very well with a small relaxation parameter—see Herman [21] and Herman and Meyer [22].

In the simultaneous paradigm the current iterate x^k is operated upon simultaneously with respect to all sets (or with respect to all sets in the current "block", in the case of a block-iterative method), generating "intermediate points"

$$x^{k+1,i} \;=\; R_i\left(x^k\right), \quad i = 1, 2, \ldots, m. \tag{1.2}$$

Here, R_i are the algorithmic operators applied to x^k with respect to the sets C_i. For example, R_i could be $P_{C_i}(x^k)$—the least-Euclidean-distance projection of x^k onto the set C_i, also called the orthogonal projection. The next iterate x^{k+1} is then generated from the intermediate points by

$$x^{k+1} \;=\; R\left(\left\{x^{k+1,i}\right\}_{i=1}^{m}\right), \tag{1.3}$$

where R is another algorithmic operator, for example, the taking of a convex combination of the form $\sum_{i=1}^{m} \omega_i x^{k+1,i}$, with $\omega_i > 0,\ i = 1, 2, \ldots, m$, and $\sum_{i=1}^{m} \omega_i = 1$.

A significant property of simultaneous schemes is that they are inherently parallel since the projections onto the sets C_i (the hyperplanes in the linear case) are independent of each other and can, therefore, be done in parallel. A prototype of the simultaneous paradigm is the Cimmino algorithm [19], in which the current iterate is projected onto all the hyperplanes, and the next iterate is the weighted average of all the projections—see also Auslender [3] for the extension to convex sets. It is well-known that the initial

convergence rate of this algorithm is so slow as to render it impractical, even if it is executed in parallel—see, for example, our computational results in [17].

In our recent paper [17] we developed a fully simultaneous projection algorithm called CAV (*Component Averaging*), which uses diagonal weighting matrices with component-related weights. We also showed how to choose a specific weighting strategy, based on the *sparsity* of the system matrix, which strongly accelerates the initial convergence of the algorithm when applied to image reconstruction problems. Being simultaneous, CAV is inherently parallel (see Butnariu, Censor and Reich [9]) but its initial convergence rate is far superior to that of the original simultaneous Cimmino method. Initial iterates of ART were better than CAV, but eventually CAV produced slightly better results.

In this paper we present a block-iterative version of CAV (which we call BICAV) and report on experiments with this algorithm. We show that when BICAV is optimized with respect to block size and relaxation parameters, its initial convergence rate is significantly superior to that of CAV, and it is somewhat on a par with ART. BICAV retains the quality of inherent parallelism of CAV, although it naturally requires somewhat more communications between processors in order to broadcast the intermediate results of the block calculations to all the processors.

While a formal convergence analysis of the fully simultaneous CAV method is available (in [17]), a mathematical validation of the new BICAV algorithm has only recently been achieved and will be published elsewhere (see comments at the end of Section 3). Thus, the encouraging experimental results, presented here, will be complemented by a mathematical study of BICAV.

A good starting point for our presentation is the general prototypical Block-Iterative Projections (BIP) method, developed by Aharoni and Censor [1], which uses the following iterative formula:

$$x_j^{k+1} \;=\; x_j^k + \lambda_k \left(\sum_{i=1}^{m} w_i^k \left(P_i \left(x^k \right) \right)_j - x_j^k \right), \quad j = 1, 2, \ldots, n, \tag{1.4}$$

where $x^k = (x_j^k) \in \mathbb{R}^n$ is the current iterate, x^{k+1} is the next iterate, $\{\lambda_k\}_{k \geq 0}$ are relaxation parameters, and $P_i(x^k)$ is the (orthogonal) projection of x^k onto the closed convex set C_i. Each $w^k = (w_i^k) \in \mathbb{R}^m$ is a *weight vector* whose components are the weights used at the k-th iterative step and they are nonnegative and sum up to one, for each k. BIP allows the weights to vary from one iteration to another and, by allowing some of the weights to take zero values, lets the iterations proceed along blocks which vary with iteration index in both size and composition.

If we choose in (1.4) $w^k = e^{i(k)}$, for all $k \geq 0$, where $e^t \in \mathbb{R}^n$ is the t-th standard basis

vector (having one in its t-th coordinate and zeros elsewhere), and $\{i(k)\}_{k\geq 0}$ is a *control sequence* of the algorithm all of whose indices are $1 \leq i(k) \leq m$, then the BIP scheme (1.4) becomes the well-known purely sequential POCS (Projections Onto Convex Sets) method. At the other extreme, if $w_i^k \neq 0$, for all $k \geq 0$ and all $i = 1, 2, \ldots, m$, then BIP becomes a fully simultaneous algorithm in which all sets $\{C_i\}$ are being acted upon in every iterative step.

Although the weights in BIP depend on the iteration index, they are not *component-related*, i.e., they cannot vary with the component index j. To the best of our knowledge, earlier projection algorithms for the convex feasibility problem, except for the recent method of Byrne and Censor [12, 13], did not allow the weights to be component-related (i.e., j-related or pixel-related). Both CAV [17] and the new BICAV algorithms use component-related weights for solving sparse systems of linear equations. More specific to image reconstruction, Mueller, Yagel and Wheller [26] have recently incorporated pixel-dependent weighting into ART reconstructions from enhanced modeling considerations.

The paper is laid out as follows. In Section 2 we briefly review the motivation and construction of the fully simultaneous CAV algorithm presented in [17]. In Section 3 we present BICAV, the block-iterative derivation of CAV, which is the object of the present study. Section 5 contains a report of our experimental computational work with BICAV and comparisons with other relevant iterative reconstruction methods.

2 Motivation and Definitions

Consider the case of linear equations in which the sets C_i are hyperplanes

$$H_i \stackrel{\Delta}{=} \left\{x \in \mathbb{R}^n \mid \langle a^i, x\rangle = b_i\right\}, \tag{2.1}$$

for $i = 1, 2, \ldots, m$, where $\langle \cdot, \cdot \rangle$ is the inner product and $a^i = (a_j^i)_{j=1}^n \in \mathbb{R}^n$, $a^i \neq 0$, and $b_i \in \mathbb{R}$ are given vectors and given real numbers, respectively. Then, for any $z \in \mathbb{R}^n$, the orthogonal projection of z onto H_i is

$$P_i(z) \;=\; z + \frac{b_i - \langle a^i, z\rangle}{\|a^i\|_2^2} a^i\,, \tag{2.2}$$

where $\|\cdot\|_2$ is the Euclidean norm.

In Cimmino's algorithm for the convex feasibility problem (Auslender [3]), with relaxation parameters and with equal weights, the next iterate x^{k+1} is the average of the projections of x^k on the closed and convex sets C_i, as follows:

Algorithm 2.1 (Cimmino):
Initialization: $x^0 \in \mathbb{R}^n$ *is arbitrary.*
Iterative Step: *Given* x^k *compute*

$$x^{k+1} = x^k + \frac{\lambda_k}{m} \sum_{i=1}^{m} \left(P_i(x^k) - x^k \right), \tag{2.3}$$

where $\{\lambda_k\}_{k \geq 0}$ *are relaxation parameters.*

Expanding the iterative step (2.3) according to (2.2) produces, for every component $j = 1, 2, \ldots, n$,

$$x_j^{k+1} = x_j^k + \frac{\lambda_k}{m} \sum_{i=1}^{m} \frac{b_i - \langle a^i, x^k \rangle}{\|a^i\|_2^2} a_j^i, \tag{2.4}$$

which is a special case of

$$x^{k+1} = x^k + \lambda_k \sum_{i=1}^{m} w_i \frac{b_i - \langle a^i, x^k \rangle}{\|a^i\|_2^2} a^i, \tag{2.5}$$

where the fixed weights $\{w_i\}_{i=1}^m$ must be positive for all i and $\sum_{i=1}^m w_i = 1$. In matrix notation, (2.4) can be written as

$$x^{k+1} = x^k + \lambda_k A^T D \left(b - Ax^k \right), \tag{2.6}$$

where $b = (b_i) \in \mathbb{R}^m$, A^T (the transpose of A) has a^i in its i-th column, and

$$D = \frac{1}{m} \operatorname{diag} \left(\frac{1}{\|a^1\|_2^2}, \frac{1}{\|a^2\|_2^2}, \ldots, \frac{1}{\|a^m\|_2^2} \right). \tag{2.7}$$

Our CAV algorithm [17] is similar in form to (2.6), but with a totally different diagonal matrix D.

Consider now the system (2.1). When it is sparse, only a relatively small number of the elements $a_j^1, a_j^2, \ldots, a_j^m$ are nonzero, but in (2.4) the sum of their contributions is divided by the relatively large m. This observation led us to consider a replacement of the factor $1/m$ in (2.4) by a factor that depends only on the *nonzero* elements in the set $\{a_j^1, a_j^2, \ldots, a_j^m\}$. For each $j = 1, 2, \ldots, n$, we denoted by s_j the number of nonzero elements of column j, and we wanted to replace (2.4) by

$$x_j^{k+1} = x_j^k + \frac{\lambda_k}{s_j} \sum_{i=1}^{m} \frac{b_i - \langle a^i, x^k \rangle}{\|a^i\|_2^2} a_j^i. \tag{2.8}$$

We then combined this idea of using the s_j's with the concepts developed by Byrne and Censor [12, 13] to obtain our CAV algorithm, which uses *oblique projections*, commonly

defined as follows. Let $H \triangleq \{x \in \mathbb{R}^n \mid \langle a, x\rangle = b\}$, with $a = (a_j) \in \mathbb{R}^n$, $b \in \mathbb{R}$ and $a \neq 0$, be a hyperplane. Let G be an $n \times n$ symmetric positive definite matrix and let $\|x\|_G^2 \triangleq \langle x, Gx\rangle$ be the associated *ellipsoidal norm*, see, e.g., Bertsekas and Tsitsiklis [6, Proposition A.28]. Given a point $z \in \mathbb{R}^n$, the oblique projection of z onto H with respect to G is the unique point $P_H^G(z) \in H$ for which

$$P_H^G(z) \;=\; \arg\min\{\|x - z\|_G \mid x \in H\}\,. \tag{2.9}$$

Solving this minimization problem leads to

$$P_H^G(z) \;=\; z + \frac{b - \langle a, z\rangle}{\|a\|_{G^{-1}}^2} G^{-1} a\,, \tag{2.10}$$

where G^{-1} is the inverse of G. For $G = I$, the unit matrix, Equation (2.10) yields the orthogonal projection of z onto H, as given by (2.2); see, e.g., Ben-Israel and Greville [5, Section 2.6].

In order to consider oblique projections onto H with respect to a diagonal matrix $G = \mathrm{diag}(g_1, g_2, \ldots, g_n)$ for which some diagonal elements might be zero, we introduced the following definition.

Definition 2.1 [17]: *Let $G = \mathrm{diag}(g_1, g_2, \ldots, g_n)$ with $g_j \geq 0$, for all $j = 1, 2, \ldots, n$, let $H = \{x \in \mathbb{R}^n \mid \langle a, x\rangle = b\}$ be a hyperplane with $a = (a_j) \in \mathbb{R}^n$ and $b \in \mathbb{R}$, and assume that $g_j = 0$ if and only if $a_j = 0$. The* generalized oblique projection *of a point $z \in \mathbb{R}^n$ onto H with respect to G is defined, for all $j = 1, 2, \ldots, n$, by*

$$(P_H^G(z))_j \;\triangleq\; \begin{cases} z_j + \dfrac{b - \langle a, z\rangle}{\displaystyle\sum_{\substack{l=1\\ g_l \neq 0}}^{n} \dfrac{a_l^2}{g_l}} \cdot \dfrac{a_j}{g_j}\,, & \text{if} \quad g_j \neq 0\,, \\[2ex] z_j\,, & \text{if} \quad g_j = 0\,. \end{cases} \tag{2.11}$$

This $P_H^G(z)$ reduces to (2.10) if $g_j \neq 0$ for all $j = 1, 2, \ldots, n$. It is not difficult to verify that this $P_H^G(z)$ belongs to H, that it solves (2.9) if we just replace $\|x - z\|_G$ there by $\langle x - z, G(x - z)\rangle$, and that it is uniquely defined, although other solutions of (2.9) may exist due to the possibly zero-valued g_j's.

Consider now a set $\{G_i\}_{i=1}^m$ of real diagonal $n \times n$ matrices $G_i = \mathrm{diag}(g_{i1}, g_{i2}, \ldots, g_{in})$ with $g_{ij} \geq 0$ for all $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$, such that $\sum_{i=1}^m G_i = I$. Referring to the sparsity pattern of A we made the following definition.

Definition 2.2 [17]: *A family $\{G_i\}_{i=1}^m$ of real diagonal $n \times n$ matrices with all diagonal elements $g_{ij} \geq 0$ and such that $\sum_{i=1}^m G_i = I$ will be called* sparsity pattern oriented (SPO, for short) with respect to an $m \times n$ matrix A *if, for every $i = 1, 2, \ldots, m$, $g_{ij} = 0$ if and only if $a_j^i = 0$.*

The CAV algorithm that we presented in [17] combined three features:

1. Each orthogonal projection onto H_i in (2.5) was replaced by a generalized oblique projection with respect to G_i.

2. The scalar weights $\{w_i\}$ in (2.5) were replaced by the diagonal weighting matrices $\{G_i\}$.

3. The actual weights were set inversely proportional to the number of nonzero elements in each column, as motivated by the discussion preceding Equation (2.8).

The iterative step resulting from the first two features has the form

$$x^{k+1} \;=\; x^k + \lambda_k \sum_{i=1}^{m} G_i \left(P_{H_i}^{G_i}(x^k) - x^k \right) , \tag{2.12}$$

or, equivalently, substituting from (2.11) for each $P_{H_i}^{G_i}$, we obtained

Algorithm 2.2 (Diagonal Weighting (DWE) for Linear Equations):
Initialization: $x^0 \in \mathbb{R}^n$ *is arbitrary.*
Iterative Step: *Given* x^k*, compute* x^{k+1} *by using, for* $j = 1, 2, \ldots, n$*, the formula:*

$$x_j^{k+1} \;=\; x_j^k + \lambda_k \sum_{\substack{i=1 \\ g_{ij} \neq 0}}^{m} \frac{b_i - \langle a^i, x^k \rangle}{\displaystyle\sum_{\substack{l=1 \\ g_{il} \neq 0}}^{n} \frac{(a_l^i)^2}{g_{il}}} \cdot a_j^i \, , \tag{2.13}$$

where $\{G_i\}_{i=1}^m$ *is a given family of diagonal SPO (with respect to A) weighting matrices as in Definition 2.2, and* $\{\lambda_k\}_{k \geq 0}$ *are relaxation parameters.*

Finally, we specified how we construct the diagonal matrices $\{G_i\}_{i=1}^m$ in order to utilize the s_j's appearing in (2.8). Define

$$g_{ij} \;\triangleq\; \begin{cases} \frac{1}{s_j} \, , & \text{if } a_j^i \neq 0 \, , \\ \\ 0 \, , & \text{if } a_j^i = 0 \, . \end{cases} \tag{2.14}$$

With this particular SPO family of G_i's we obtained our CAV algorithm:

Algorithm 2.3 (Component Averaging (CAV)):
Initialization: $x^0 \in \mathbb{R}^n$ *is arbitrary.*
Iterative Step: *Given* x^k*, compute* x^{k+1} *by using, for* $j = 1, 2, \ldots, n$*, the formula:*

$$x_j^{k+1} \;=\; x_j^k + \lambda_k \sum_{i=1}^{m} \frac{b_i - \langle a^i, x^k \rangle}{\displaystyle\sum_{l=1}^{n} s_l \left(a_l^i\right)^2} \cdot a_j^i \, , \tag{2.15}$$

where $\{\lambda_k\}_{k \geq 0}$ *are relaxation parameters and* $\{s_l\}_{l=1}^n$ *are as defined above.*

We showed in [17] that Algorithm 2.2, with $\lambda_k = 1$ for all $k \geq 0$, generates sequences $\{x^k\}$ which always converge, regardless of the initial point x^0 and independently from the consistency or inconsistency of the underlying system $Ax = b$. Moreover, it always converges to a minimizer of a certain proximity function.

3 The Block-Iterative Component Averaging Algorithm (BICAV)

We now develop our block-iterative derivation of the CAV algorithm. The basic idea is to break up the system $Ax = b$ into "blocks" of equations and treat each block according to the CAV method, passing cyclically over all the blocks. This calls for the slight notational complication of having to deal with block indices. Throughout the following, M will be the number of blocks. For $t = 1, 2, \ldots, M$, let the block of indices $B_t \subseteq \{1, 2, \ldots, m\}$, be an ordered subset of the form $B_t = \{i_1^t, i_2^t, \ldots, i_{m(t)}^t\}$, where $m(t)$ is the number of elements in B_t, such that every element of $\{1, 2, \ldots, m\}$ appears in at least one of the sets B_t. For $t = 1, 2, \ldots, M$, let A_t denote the matrix formed by taking all the rows of A whose indices belong to the block of indices B_t, i.e.,

$$A_t \triangleq \begin{pmatrix} a^{i_1^t} \\ a^{i_2^t} \\ \vdots \\ a^{i_{m(t)}^t} \end{pmatrix}, \quad t = 1, 2, \ldots, M. \tag{3.1}$$

The iterative step of our proposed BICAV algorithm uses, for every block index $t = 1, 2, \ldots, M$, generalized oblique projections with respect to a family $\{G_i^t\}_{i=1}^m$ of diagonal matrices which are SPO with respect to A_t. The same family is also used to perform the diagonal weighting. The resulting iterative step has the form

$$x^{k+1} = x^k + \lambda_k \sum_{i \in B_{t(k)}} G_i^{t(k)} \left(P_{H_i}^{G_i^{t(k)}}(x^k) - x^k \right), \tag{3.2}$$

where $\{t(k)\}_{k \geq 0}$ is a *control sequence* according to which the $t(k)$-th block is chosen by the algorithm to be acted upon at the k-th iteration. Thus, we must have $1 \leq t(k) \leq M$, for all $k \geq 0$. The real numbers $\{\lambda_k\}_{k \geq 0}$ are user-chosen *relaxation parameters*. Substituting from (2.11) for each $P_{H_i}^{G_i^{t(k)}}$, we obtain:

Algorithm 3.1 (Block-Iterative Diagonal Weighting (BIDWE) for Linear Equations):

Initialization: $x^0 \in \mathbb{R}^n$ *is arbitrary.*
Iterative Step: *Given* x^k*, compute* x^{k+1} *by using, for* $j = 1, 2, \ldots, n$*, the formula:*

$$x_j^{k+1} \;=\; x_j^k + \lambda_k \sum_{\substack{i \in B_{t(k)} \\ g_{ij}^{t(k)} \neq 0}} \frac{b_i - \langle a^i, x^k \rangle}{\displaystyle\sum_{\substack{l=1 \\ g_{il}^{t(k)} \neq 0}}^{n} \frac{(a_l^i)^2}{g_{il}^{t(k)}}} \cdot a_j^i \,, \tag{3.3}$$

where, for each $t = 1, 2, \ldots, M$*,* $\{G_i^t\}_{i=1}^m$ *is a given family of diagonal SPO (with respect to* A_t*) weighting matrices, as in Definition 2.2, the control sequence is cyclic, i.e.,* $t(k) = k \bmod M + 1$*, for all* $k \geq 0$*,* $\{\lambda_k\}_{k \geq 0}$ *are relaxation parameters, and* $G_i^t = \operatorname{diag}(g_{i1}^t, g_{i2}^t, \ldots, g_{in}^t)$*.*

Finally, in order to achieve the acceleration, the diagonal matrices $\{G_i^t\}_{i=1}^m$ are constructed as in the original CAV algorithm [17], but with respect to each A_t. Let s_j^t be the number of nonzero elements $a_j^i \neq 0$ in the j-th column of A_t and define

$$g_{ij}^t \;\triangleq\; \begin{cases} \dfrac{1}{s_j^t}\,, & \text{if } a_j^i \neq 0, \\[2ex] 0\,, & \text{if } a_j^i = 0. \end{cases} \tag{3.4}$$

It is easy to verify that for each $t = 1, 2, \ldots, M$, $\sum_{i=1}^m G_i^t = I$ holds for these matrices. With these particular SPO families of G_i^t's we obtain our block-iterative algorithm:

Algorithm 3.2 (Block-Iterative Component Averaging (BICAV)):
Initialization: $x^0 \in \mathbb{R}^n$ *is arbitrary.*
Iterative Step: *Given* x^k*, compute* x^{k+1} *by using, for* $j = 1, 2, \ldots, n$*, the formula:*

$$x_j^{k+1} \;=\; x_j^k + \lambda_k \sum_{i \in B_{t(k)}} \frac{b_i - \langle a^i, x^k \rangle}{\displaystyle\sum_{l=1}^{n} s_l^{t(k)} (a_l^i)^2} \cdot a_j^i \,, \tag{3.5}$$

where $\{\lambda_k\}_{k \geq 0}$ *are relaxation parameters,* $\{s_l^t\}_{l=1}^n$ *are as defined above, and the control sequence is cyclic, i.e.,* $t(k) = k \bmod M + 1$*, for all* $k \geq 0$*.*

For the case $M = 1$ and $B_1 = \{1, 2, \ldots, m\}$, Algorithm 3.2 becomes fully simultaneous, i.e., it is the CAV algorithm of [17]. For $M = m$ and $B_t = \{t\}$, $t = 1, 2, \ldots, m$, BICAV simply becomes ART (the well-known Algebraic Reconstruction Technique—see, e.g., Herman [21]).

After the first version of this paper was ready, progress has been made in the mathematical study of the convergence of BICAV by Byrne [11, 10] and by Censor and Elfving [16].

These results, which will be published elsewhere, validate the convergence with relaxation parameters of CAV in the inconsistent case, and of BICAV in the consistent case. In [16] the algorithmic framework of BICAV is generalized to handle systems of linear inequalities and to include as a special case the well-known image reconstruction algorithm called SART, invented by Andersen and Kak [2]. The following result describes the behavior of BICAV on a system of linear equations in the consistent case.

Theorem 3.1 (BICAV for Linear Equalities [16, Theorem 7.1]) *Let $0 < \varepsilon \leq \lambda_k \leq 2 - \varepsilon$, for all $k \geq 0$, where ε is an arbitrarily small but fixed constant, and assume consistency of the system $Ax = b$. Then any sequence $\{x^k\}_{k\geq 0}$, generated by Algorithm 3.2 (BICAV), converges to a solution of the system $Ax = b$.*

The next theorem shows that any sequence $\{x^k\}_{k\geq 0}$, generated by the fully simultaneous Algorithm 2.3 (CAV), converges to a weighted least squares solution of the system of equations $Ax = b$, regardless of its consistency, for relaxation parameters in the interval $[\varepsilon, 2 - \varepsilon]$. Only the case of unity relaxation, i.e., $\lambda_k = 1$, for all $k \geq 0$, was shown in [17], where CAV was first proposed; the proof in [17] was adapted from [12] and [13, Algorithm 4.2].

Theorem 3.2 (CAV for Linear Equalities in the Inconsistent Case [16, Theorem 7.3]) *Let $0 < \varepsilon \leq \lambda_k \leq 2 - \varepsilon$, for all $k \geq 0$, where ε is an arbitrarily small but fixed constant, Then any sequence $\{x^k\}_{k\geq 0}$, generated by Algorithm 2.3 (CAV), for linear equations, converges to a weighted least squares solution with weight matrix $M_{CAV} = \mathrm{diag}\{1/||a^i||_S^2 \mid i = 1, 2, \ldots, m\}$ and with $S = \mathrm{diag}\{s_j \mid j = 1, 2, \ldots, n\}$, where s_j is the number of nonzero elements in the j-th column of A.*

4 Problem Description, Convergence Measures and Test Cases

In the medical application of Transmission Computerized Tomography (TCT), a planar cross-section of the body is considered and the tissue's attenuation of X-rays everywhere in the cross-section has to be reconstructed. This unknown function of two variables has real nonnegative values and is called the *image* or *picture*. The fundamental model in the *finite series-expansion approach*, see, e.g., Censor [14], is formulated as follows. A Cartesian grid of square picture-elements, called *pixels*, is introduced into the region of interest so that it covers the whole picture that has to be reconstructed. The pixels are numbered in some agreed manner, say from 1 (top left corner pixel) to n (bottom

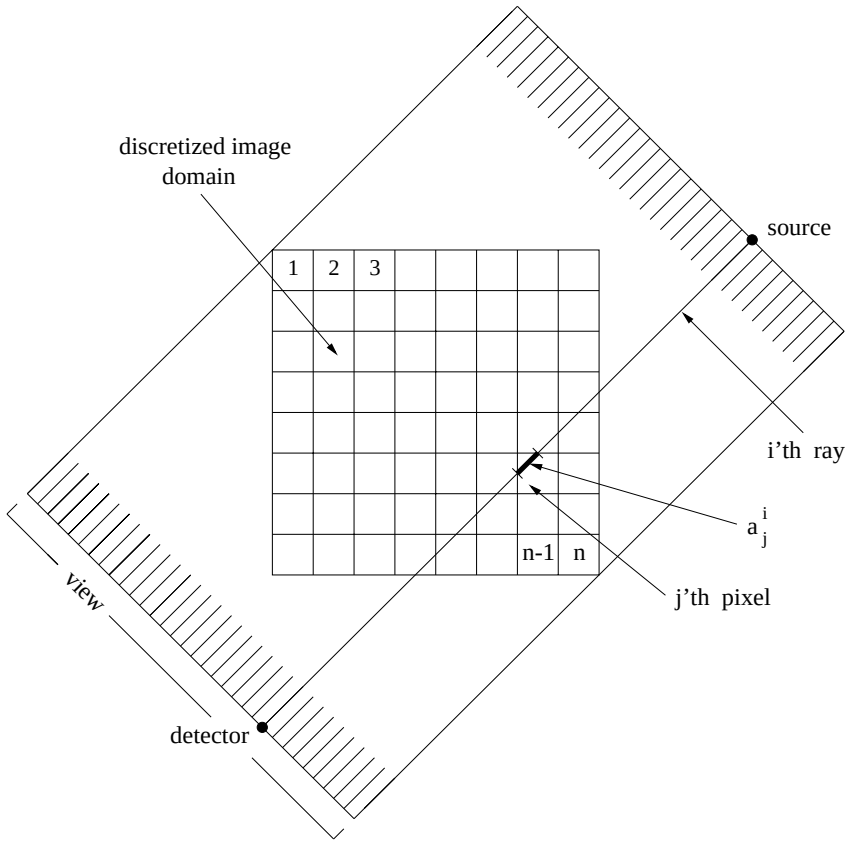

Figure 1: The fully discretized model for transmission tomography image reconstruction.

right corner pixel); see Figure 1. The X-ray attenuation function is assumed to take a constant uniform value x_j throughout the j-th pixel, for $j = 1, 2, \ldots, n$. Sources of X-rays and detectors are assumed to be points, and the rays between them—lines. It is further assumed that the length of intersection of the i-th ray with the j-th pixel, denoted by a_j^i, for all $i = 1, 2, \ldots, m$, $j = 1, 2, \ldots, n$, represents the contribution of the j-th pixel to the total attenuation along the i-th ray.

The physical measurement of the total attenuation along the i-th ray, denoted by b_i, represents the line integral of the unknown attenuation function along the path of the ray. Therefore, in this fully discretized model, each line integral is approximated by a finite sum and the model is described by a system of linear equations

$$\sum_{j=1}^{n} x_j a_j^i = b_i \,, \quad i = 1, 2, \ldots, m \,. \tag{4.6}$$

Here, $b = (b_i) \in \mathbb{R}^m$ is the *measurements vector*, $x = (x_j) \in \mathbb{R}^n$ is the *image vector* and the $m \times n$ matrix $A = (a_j^i)$ is the *projection matrix*.

All our reconstruction algorithms were implemented within SNARK93, a software package for testing and evaluating algorithms for image reconstruction, see Browne, Herman

and Odhner [8]. Three different measures were used for comparisons: *distance*, *relative error*, and *standard deviation*, calculated by SNARK93, and defined in [8, Section 5.10] as follows.

Let x_j^k and $\tilde{x}_j$ denote the density assigned to the j-th pixel of the reconstruction after k iterations, and the density of the j-th pixel in the phantom, respectively. Let S denote the set of indices j of pixels which are in the region of interest and let α be the number of elements in S. The *average* value of the reconstructed image x^k is given by

$$\rho_k \triangleq \frac{1}{\alpha} \sum_{j \in S} x_j^k \tag{4.7}$$

and the *variance* of x^k is given by

$$v_k \triangleq \frac{1}{\alpha} \sum_{j \in S} (x_j^k - \rho_k)^2. \tag{4.8}$$

The *standard deviation* of the reconstructed image x^k is then

$$\sigma_k \triangleq \sqrt{v_k}\,. \tag{4.9}$$

Similarly, we define the average value $\tilde{\rho}$, variance $\tilde{v}$ and standard deviation $\tilde{\sigma}$ for the phantom, in terms of the phantom values $\tilde{x}_j$.

The *distance* between x^k and the phantom $\tilde{x}$ is

$$\delta_k \triangleq \begin{cases} \dfrac{1}{\tilde{\sigma}} \sqrt{\dfrac{1}{\alpha} \displaystyle\sum_{j \in S} (x_j^k - \tilde{x}_j)^2}, & \text{if } \tilde{\sigma} > 0, \\[2ex] \sqrt{\displaystyle\sum_{j \in S} (x_j^k - \tilde{x}_j)^2}, & \text{if } \tilde{\sigma} \leq 0. \end{cases} \tag{4.10}$$

With $\tau \triangleq \sum_{j \in S} |\tilde{x}_j|$ the *relative error* of x^k is defined by

$$\epsilon_k \triangleq \begin{cases} \dfrac{1}{\tau} \displaystyle\sum_{j \in S} |x_j^k - \tilde{x}_j|, & \text{if } \tau > 0\,, \\[2ex] \displaystyle\sum_{j \in S} |x_j^k - \tilde{x}_j|, & \text{if } \tau \leq 0. \end{cases} \tag{4.11}$$

The performance of the different algorithms is demonstrated on the reconstruction of the Herman head phantom [21, Section 4.3], which is specified by a set of ellipses, with a specific attenuation value attached to each elliptical region. The values of b_i, $i = 1, \ldots, m$, are calculated by computing the line integrals through the elliptical regions (without

reference to the discretization). Thus, the system (4.6) is basically inconsistent, because the left-hand-side is only an approximation to the actual integrals. This matches the real-life situation where the b_i's are actual X-ray readings through an object but the region of interest is discretized as above.

We examined four test cases characterized by two different image resolutions and varying numbers of projections and rays per projection. The four cases thus had differing numbers of variables and equations, as shown in Table 1.

	Equations	Variables	Image Size	Projections	Rays
Case 1	13,137	13,225	115×115	151	87
Case 2	26,425	13,225	115×115	151	175
Case 3	126,655	119,025	345×345	365	347
Case 4	232,275	119,025	345×345	475	489

Table 1: The four different test cases.

In all our experiments, the relaxation parameter remained constant throughout the iterations, i.e., $\lambda_k = \lambda$ for all $k \geq 0$. In [17] we already determined that for such problems, the optimal relaxation parameter for CAV is $\lambda = 2.0$. We also used a small relaxation parameter $\lambda = 0.1$ for ART, since it was found to produce good results without impeding the convergence rate. Although all the experiments were carried out on the four test cases, we present convergence plots only for Cases 1–3 since Case 4 was very similar to the others. Furthermore, the results for the distance measure and standard deviation were very similar to the relative error, so they are omitted. For the reconstructed images, we present only Case 2.

5 Experimental Results

In this section, we outline the general setting of the experiments and present their results. Subsection 5.1 explains how optimal relaxation parameters were chosen for BICAV and compares the relative performance of five different algorithms: ART, CAV and BICAV, together with the Cimmino (CIM) algorithm (Algorithm 2.1) and the BIP method (Equation (1.4)). Subsection 5.2 contains our main set of experiments, involving ART, CAV and BICAV. In Subsection 5.3, we examine the behavior of ART, CAV and BICAV when noise is introduced into the equations. We also experiment with Equation (2.8) and its block version, and display images obtained after very few (5) iterations.

All the methods were implemented (sequentially) on a Digital Alpha workstation running at 433MHz. The different algorithms are compared on the basis of the measures of

convergence and on image quality. We limit our work here only to the above-mentioned *projection algorithms*, leaving out many other iterative image reconstruction algorithms, such as the EM algorithm, see, e.g., Lange and Karson [25] or Hudson and Larkin [23] for the block-iterative version of the EM algorithm, called: the "Ordered Subsets EM" (OSEM) algorithm.

All our experiments were initiated with $x^0 = 0$. Note that CAV is the same as BICAV with one block and ART is identical to BICAV if every block contains exactly one equation. We henceforth use the term *iteration* to refer to one whole sweep through all equations of the system, so the time for a single iteration is the same for all algorithms. Thus, the number of iterations is a faithful timing basis for comparing the different algorithms.

In the implementation of BICAV, we initially compute, for every $t = 1, 2, \ldots, M$, all the values s_j^t, $j = 1, 2, \ldots, n$. These are used to compute the denominators of Equation (3.5), which are stored and used in subsequent iterations. We also experimented with different values of M – the number of blocks. For each value of M, we experimented with various values of the relaxation parameter λ in order to determine an optimal relaxation parameter for that number of blocks – see Subsection 5.1.

5.1 Preliminary Experiments

We first show a typical set of experiments with BICAV, to demonstrate its behavior with different relaxation parameters and to show how we chose their optimal values. We present in detail the example of BICAV with 10 blocks, executed on Case 2. The relative errors are shown in Figure 2. The relaxation parameters varied from 0.25 to 2.0, and the figures demonstrate the typical behavior of BICAV: The optimal results are obtained for some relaxation parameter strictly less than 2.0 (which is optimal for CAV). Starting from the low relaxation parameter, each successive value of λ produces visibly better performance, until the optimal value, and larger values of λ produce worse results. Even so, we can see that there is a wide range of values of λ which achieve good results: BICAV with 10 blocks achieves good convergence measures for all values of $\lambda \geq 0.4$ within 25 iterations.

Note that the optimal relaxation parameter depends on the number of blocks M (of BICAV): When $M = 1$, BICAV coincides with CAV, whose optimal relaxation parameter is 2. As M increases, the optimal relaxation parameter decreases. When M equals the number of equations, BICAV is identical to ART, and requires a relaxation parameter of 0.1.

The next set of experiments is intended to demonstrate the relative performance of BICAV with the following algorithms, of which only the first three (like BICAV) are inherently

parallel:

- The Cimmino algorithm (CIM) – Algorithm 2.1. We used relaxation parameter 2, which was determined experimentally to be the best of all values for which CIM provably converges.
- The Block-Iterative Projection algorithm (BIP) – Equation (1.4) with 10 blocks and P_i as in (2.2). Here we also used the optimal relaxation parameter of 2.
- CAV, with its known optimal relaxation parameter of 2 [17].
- ART with a relaxation parameter of 0.1 - known experimentally to achieve excellent results.

We ran BICAV with 10 blocks and optimal relaxation parameter of 1.4. The results of these comparisons are presented in Figure 3, which shows the relative errors. These plots demonstrate that there is a very clear and distinct difference between the initial convergence rate of the algorithms. ART and BICAV are almost identical in their fast initial rate of convergence. CAV is slower, but it achieves the same measures of convergence within about 25 iterations. Between CIM and BIP, BIP is clearly much better, but neither of them comes close to the performance of the other three within the number of iterations that were examined.

5.2 Main Experiments

Our main experiments demonstrated here were performed on ART, CAV and BICAV. BICAV was executed with three different block sizes, and for each block size we used the optimal relaxation parameters. Convergence plots for test Cases 1–3 are show in Figure 4. BICAV is shown with 5, 10 and 30 blocks.

On the whole, the results indicate that BICAV behaves somewhat similarly to ART. Occasionally, BICAV is better, and occasionally it is slightly worse. As to CAV, it is always initially worse than ART and BICAV, but in most cases, after several iterations, ART and BICAV begin to deteriorate while CAV continues to improve, eventually overtaking them. An interesting observation, requiring further research, can be noted by comparing Case 1 with 5 blocks and Case 2 with 10 blocks: In both cases the number of equations per block is approximately 2600, and in both cases the optimal relaxation parameter is very similar (close to 1.5). A similar situation occurs with Cases 3 and 4 (Case 4 not shown). We conjecture that for a fixed number of variables, the main factor affecting the optimal relaxation parameter in BICAV is the absolute number of equations in a block.

Figure 5 shows the phantom and reconstructed images for Case 2 for the three methods, after 10, 20 and 60 iterations. BICAV is shown for the choice of 10 blocks, with its optimal

relaxation parameter. These images bear out the fact that on the whole, BICAV behaves very similarly to ART, while CAV produces images that are initially "fuzzy", but after some 60 iterations, the images are on a par with those produced by ART and BICAV after some 10 iterations. Note also that both ART and BICAV deteriorate gradually after 10 iterations. Cases 1, 3 and 4 exhibit similar behavior.

In order to provide the best visualization of the reconstructions, each image is displayed with its pixel values linearly mapped to grey levels between 0 and 255. This method shows up artifacts better than mapping all images according to the same scale.

5.3 Other Experiments

In this subsection we take one of our four cases (Case 2) and perform on it two more sets of experiments in order to gain a more detailed view of the capabilities of BICAV.

The first set of experiments compares the performance of ART, CAV and BICAV on the data of Case 2, but with noise added to the readings, i.e., to the right-hand-side of Equation (1.1). The noise consisted of multiplying each $b_j,\ j = 1, 2, \ldots, m$, with a random number from a Gaussian distribution with an average of 1.0 and a standard deviation of 0.05. Note that we are experimenting with *transmission* CT, so our experiment with noise differs from the accepted practice in emmission CT. The relative error results are shown in Figure 6. ART and BICAV now behave differently: Whereas ART reaches its best output in four iterations, BICAV requires eight, but its results are slightly better. Further iterations of ART and BICAV produce worse results, but CAV continues to improve for about 30 iterations and then very gradually deteriorates. Note that with the addition of noise, the optimal relaxation parameter for BICAV is smaller than for the regular case.

For the image comparisons, we compared the three methods at 4, 8, 15 and 30 iterations. For each method, we picked the best image – this is shown in Figure 7. Note that the number of iterations for the best image of each method matches the number of iterations at which the relative error is minimal (Figure 6). We see from this that CAV at its best (30 iterations) is almost identical to BICAV at its best (8 iterations), and both are distinctively better than the best image obtained with ART (4 iterations). We conclude from this that in the presence of noise, CAV and BICAV are preferable to ART, provided more computation time is available.

All three methods continued to deteriorate with further iterations, with ART being the worst and CAV the best. After 1000 iterations, the following relative errors were obtained: ART - 1.0177, BICAV - 0.642, CAV - 0.4391. However, all three methods continued to improve the *residual* of the solution, where the residual of the k-th iteration is defined

as $\|Ax^{(k)} - b\|$. The consequence of this is that the residual should not be used as an indicator of the quality of reconstruction. The experiments on phantoms provide a guide as to how many iterations to perform on real data.

The second set of experiments is concerned with Equation (2.8) and the behavior of the various algorithms at very few (5) iterations. Even though Equation (2.8) is sparsity oriented in its averaging, it is not the same as CAV because it employs orthogonal projections and not oblique projections like CAV. In matrix form, (2.8) can be written as

$$x^{k+1} \;=\; x^k + \lambda_k S A^T N (b - Ax^k)\,, \tag{5.1}$$

where $S = \mathrm{diag}(1/s_1, 1/s_2, \ldots, 1/s_m)$ and $N = \mathrm{diag}(1/\|a^1\|_2^2, 1/\|a^2\|_2^2, \ldots, 1/\|a^n\|_2^2)$. The effect of N is identical to normalizing all the equations before the iterations, i.e., the i-th equation is divided by $\|a^i\|_2$. Equation (5.1) appears to be similar in form to the *generalized Landweber iteration*—see, e.g., Björck [7] and Trussell and Civanlar [31] for further information. Recent applications of Landweber-type algorithms in image reconstruction from projections appear in Pan and Yagle [27] and Pan *et al.* [28]. In fact, (5.1) is quite distinct from the generalized Landweber iteration of [28], because the matrix N does not appear there, the matrix to the left of A^T is a certain polynomial matrix (called a *shaping matrix*), and the scalar is related to the singular values of A.

Experiments with iteration formula (2.8) produced some interesting results. As far as convergence plots were concerned, the differences between CAV and (2.8) were negligble, but in some of the cases, and especially in the early iterations, CAV produced better images. We also experimented with a block version of (2.8), where in each block we divide by s_j^t, which is the number of nonzero elements of the j-th column of A_t. As with (2.8), this block version of (2.8) produced convergence plots similar to those of BICAV, but the early iteration images of BICAV were again superior.

Figure 8 shows the images produced after five iterations by ART, BICAV (10 blocks), and the block version of Equation (2.8) (10 blocks), for Case 2. Again, BICAV is very similar to ART, while the block version of (2.8) exhibits visible artifacts. These images demonstrate that convergence plots alone are insufficient for comparing different algorithms; reconstructed images are essential. The important point here is that the oblique projections used by CAV and BICAV produce better results than the orthogonal projections of (2.8).

6 Conclusions

We have introduced here a block-iterative version (BICAV) of our component averaging algorithm CAV [17]. After suitable optimization of block sizes and relaxation parameters, BICAV combines the best features of ART and CAV:

• The initial convergence rate of BICAV is very similar to ART (and better than CAV).
• Similarly to CAV, it is inherently parallel in structure.
• For noisy images, CAV and BICAV produce better images than ART, but require more iterations.
• Oblique projections, as used in CAV and BICAV, produce better images than the orthogonal projections of Equation (2.8).

The significance of good parallel iterative methods lies in the fact that multiprocessor workstations can be expected in the near future to be widely prevalent alongside state-of-the-art medical equipment. With regard to the parallel implementation of BICAV, it should be noted that it requires somewhat more communication overhead than CAV, because the results of the projections within one block need to be combined and redistributed before they can be used for the next block. In view of this, one should attempt to minimize the number of blocks. Our results indicate that 10 blocks provides a reasonable choice.

Our use of the Diagonal Weighting (DWE) algorithm for linear equations (Algorithm 2.2) is motivated by a choice of diagonal weighting matrices (Equation (2.14)), with sparsity-related weights, which strongly and significantly accelerate the fully simultaneous Cimmino algorithm. This usage leads to the CAV method, Algorithm 2.3. BICAV, the block-iterative version of CAV, improves on CAV: Its initial rate of convergence is almost identical to ART, but in contrast to ART, it is also inherently parallel.

Future research on CAV and BICAV will concentrate on their behavior on other sparse systems coming from different real-world problems. With regard to convergence analysis, CAV is now known to converge in the inconsistent case with relaxation parameters up to 2, while BICAV is known to converge for the same relaxation parameters, but only in the consistent case. There is still a need for further research in this direction. Another topic for further research is an analysis of the initial rate of convergence of BICAV, with emphasis on the relation between the block size and the optimal relaxation parameter.

Acknowledgements. We thank Tommy Elfving, Robert Lewitt and Samuel Matej for their enlightening comments and discussions on this work. Special thanks are due to

Charles Byrne for sharing with us his research notes [11] and technical report [10]. Thanks are also due to the reviewers whose comments led to many improvements in the paper. This research was supported by research grants 293/97 and 592/00 from the Israel Science Foundation founded by the Israel Academy of Sciences and Humanities. The work of Y. Censor was also supported by NIH grant HL-28438 at the Medical Image Processing Group (MIPG), Department of Radiology, Hospital of the University of Pennsylvania, Philadelphia, PA, USA.

References

[1] R. Aharoni and Y. Censor. Block-iterative projection methods for parallel computation of solutions to convex feasibility problems. *Linear Algebra and Its Applications*, **120**:165–175, 1989.

[2] A.H. Andersen and A.C. Kak. Simultaneous algebraic reconstruction technique (SART): A superior implementation of the art algorithm. *Ultrasonic Imaging*, **6**:81–94, 1984.

[3] A. Auslender. *Optimization: Méthodes Numériques.* Masson, Paris, 1976.

[4] H.H. Bauschke and J.M. Borwein. On projection algorithms for solving convex feasibility problems. *SIAM Review*, **38**:367–426, 1996.

[5] A. Ben-Israel and T.N.E. Greville. *Generalized Inverses: Theory and Applications.* John Wiley & Sons, New York, 1974.

[6] D.P. Bertsekas and J.N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods.* Prentice-Hall, Englewood Cliffs, New Jersey, USA, 1989.

[7] Å. Björck. *Numerical Methods for Least squares Problems.* Society for Industrial and Applied Mathematics (SIAM), Philadelphia, Pennsylvania, USA, 1996.

[8] J.A. Browne, G.T. Herman, and D. Odhner. SNARK93: A programming system for image reconstruction from projections. Technical Report MIPG198, The Medical Image Processing Group (MIPG), Dept. of Radiology, University of Pennsylvania, 1993.

[9] D. Butnariu, Y. Censor, and S. Reich (Editors). *Inherently Parallel Algorithms in Feasibility and Optimization and their Applications.* Elsevier Science Publishers, Amsterdam, the Netherlands, 2001.

[10] C. Byrne. Notes on block-iterative or ordered subset methods in image reconstruction. Technical report, Dept. of Mathematical Sciences, University of Massachusetts at Lowell, September 2000.

[11] C. Byrne. Private communication. June–July, 2000.

[12] C. Byrne and Y. Censor. Proximity function minimization for separable, jointly convex Bregman distances, with applications. Technical report, February 1998.

[13] C. Byrne and Y. Censor. Proximity function minimization using multiple Bregman projections, with applications to entropy optimization and Kullback-Leibler distance minimization. Technical Report, June 1999. Revised: May 2000. *Annals of Operations Research*, to appear.

[14] Y. Censor. Finite series-expansion reconstruction methods. *Proceedings of the IEEE*, **71**:409–419, 1983.

[15] Y. Censor, P.P.B. Eggermont, and D. Gordon. Strong underrelaxation in Kaczmarz's method for inconsistent systems. *Numerische Mathematik*, **41**:83–92, 1983.

[16] Y. Censor and T. Elfving. Block-iterative algorithms with diagonally scaled oblique projections for the linear feasibility problem. Technical report, April 2001.

[17] Y. Censor, D. Gordon, and R. Gordon. Component averaging: An efficient iterative parallel algorithm for large and sparse unstructured problems. *Parallel Computing*, **27**:777–808, 2001.

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

[19] G. Cimmino. Calcolo approssimato per soluzioni dei sistemi di equazioni lineari. *La Ricerca Scientifica XVI, Series II, Anno IX*, **1**:326–333, 1938.

[20] P.L. Combettes. The convex feasibility problem in image recovery. *Advances in Imaging and Electron Physics*, **95**:155–270, 1996.

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

[22] G.T. Herman and L.B. Meyer. Algebraic reconstruction techniques can be made computationally efficient. *IEEE Transactions on Medical Imaging*, **MI-12**:600–609, 1993.

[23] H.M. Hudson and R.S. Larkin. Accelerated image reconstruction using ordered subsets of projection data. *IEEE Transactions on Medical Imaging*, **MI-13**:601–609, 1994.

[24] S. Kaczmarz. Angenäherte Auflösung von Systemen linearer Gleichungen. *Bulletin de l'Académie Polonaise des Sciences et Lettres*, **A35**:355–357, 1937.

[25] K. Lange and R. Carson. EM reconstruction algorithms for emission and transmission tomography. *Journal of Computer Assisted Tomography*, **8**:306–316, 1984.

[26] K. Mueller, R. Yagel, and J.J. Wheller. Anti-aliased three-dimensional cone-beam reconstruction of low-contrast objects with algebraic methods. *IEEE Transactions on Medical Imaging*, **MI-18**:519–537, 1999.

[27] T.-S. Pan and A.E. Yagle. Acceleration of Landweber-type algorithms by suppression of projection on the maximum singular vector. *IEEE Transactions on Medical Imaging*, **MI-11**:479–487, 1992.

[28] T.-S. Pan, A.E. Yagle, N.H. Clinthorne, and W.L. Rogers. Acceleration and filtering in the generalized Landweber iteration using a variable shaping matrix. *IEEE Transactions on Medical Imaging*, **MI-12**:278–286, 1993.

[29] H. Stark and Y. Yang. *Vector space projections: A Numerical Approach to Signal and Image Processing, Neural Nets, and Optics.* John Wiley & Sons, New York, 1998.

[30] K. Tanabe. Projection method for solving a singular system of linear equations and its applications. *Numerische Mathematik*, **17**:203–214, 1971.

[31] H.J. Trussell and M.R. Civanlar. The Landweber iteration and projection onto convex sets. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-33**:1632–1634, 1985.