Spring 2010

The Study of Binary Steering Algorithms in Discrete Tomography

Brittany A. Cole

THE STUDY OF BINARY STEERING ALGORITHMS IN DISCRETE TOMOGRAPHY

by

BRITTANY A. COLE

(Under the Direction of Dr. Xiezhang Li)

ABSTRACT

In this research project we will study the binary steering technique in iterative methods for image reconstruction in discrete tomography (DT). We will compare the effectiveness of three common algorithms to solve binary systems. The methods used are the Algebraic Reconstruction Technique (ART), Cimmino (CIM), and Diagonally-Relaxed Orthogonal Projections (DROP) algorithms. These reconstruction algorithms will be transformed into binary reconstruction algorithms through the binary steering technique. A new binary steering technique was developed to improve the convergence of the binary steering algorithms. Numerical experiments were performed to demonstrate these improvements.

Key Words: Binary Steering, Discrete Tomography, Medical Imaging, Projections, Reconstruction methods/algorithms

2009 *Mathematics Subject Classification*: 65F10, 92C5

THE STUDY OF BINARY STEERING ALGORITHMS IN DISCRETE TOMOGRAPHY

by

BRITTANY A. COLE

B.S. in Mathematics, Spelman College, Atlanta, GA

A Thesis Submitted to the Graduate Faculty of Georgia Southern University in Partial Fulfillment of the Requirement for the Degree

MASTER OF SCIENCE

IN MATHEMATICS

STATESBORO, GEORGIA

2010

THE STUDY OF BINARY STEERING ALGORITHMS IN DISCRETE TOMOGRAPHY

by

BRITTANY A. COLE

Major Professor: Dr. Xiezhang Li

Committee: Dr. Jiehua Zhu

Dr. Yan Wu

Electronic Version Approved:

May 2010

ACKNOWLEDGMENTS

I wish to acknowledge the efforts of my thesis defense committee members, Drs. Y. Wu and J. Zhu. I would also like to thank Dr. Martha Abell, Mathematics Department Chair. I would especially like to thank my advisor, Dr. Xiezhang Li for his guidance on this project. And lastly, I am grateful to my parents for their encouragement and continuous support throughout my collegiate tenure.

TABLE OF CONTENTS

Page

LIST OF TABLES

Table Page

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

In general, tomography deals with the problem of determining shape and dimensional information of an object from a set of projections. The object corresponds to a function; and the problem posed is to reconstruct this function from its integrals or sums over subsets of its domain. It is typical in discrete tomography that only a few projections (line sums) are used. In this case, conventional techniques fail. The name discrete tomography is due to Larry Shepp, who organized the first meeting devoted to this topic (DIMACS Mini-Symposium on Discrete Tomography, September 19, 1994, Rutgers University). Discrete Tomography (DT) deals with the reconstruction of a function from its projections, when the function has a known finite range [KH99]. Knowing the discrete range can significantly reduce the number of projections required for a high-quality reconstruction. The reconstruction methods used in DT applications are usually based on some formulation as an optimization problem. According to theoretical and practical results, in some cases just a few views are sufficient for high-quality reconstruction of objects. This is an important difference between DT and classical tomographies such as computed tomography (CT), the standard commercial verisions of X-ray computed tomography [HK03].

Computed Tomography (CT) refers to computational synthesis of an image from external measurements of a spatially varying function in terms of projections. Line integrals are the most common measures, which are collectively known as projections. CT reconstruction algorithms are traditionally developed for real-valued underlying functions. However, the ranges of the underlying functions may often be discrete in industry and other applications. Therefore, DT has been developed to reconstruct an unknown function whose range is a given discrete set and domain that may be discrete

or continuous. The general CT reconstruction algorithms are not appropriate in this case. The known discrete range of the function may allow it to be determined from less data than what are necessary for general functions. So DT has its own theory and reconstruction methods. The most popular models in DT are line projections with a lattice of points and strip projection with a lattice of pixels/cells. The line-based projection model fits some applications but involves a major approximation since the X-ray beams of finite widths are simplified as line integrals. The strip-based projection model formulates projection equations according to the fractional areas of the intersection of each strip-shaped beam and the rectangular grid of an image to be reconstructed [ZLYW08].

A special case of discrete tomography deals with the reconstruction of a binary image from a small number of projections. The problem with reconstructing a binary image from a small number of projections generally leads to a large number of solutions. It is desirable to limit the class of possible solutions to only those that are typical of the class of the images which contains the image being reconstructed [CM99].

The binary steering process [CM99] is a method designed to change between consecutive steps of a nonbinary iterative image reconstruction algorithm in order to gradually steer the iterates towards a binary solution. In other words, it is a steering scheme by which nonbinary iterative reconstruction algorithms can be steered towards a binary solution of a binary problem [C01].

Reconstruction algorithms have many applications in image processing, medicine, three-dimensional statistical data security problems, computer tomography assisted engineering and design, and electron microscopy [HK03]. The mathematical theory

of DT is based mostly on discrete mathematics but also uses functional analysis, combinatorics, geometry, optimization, and algebra [KH99].

This paper will focus on the binary steering scheme on three particular DT algorithms for the binary image reconstruction. We will introduce three major reconstruction algorithms and test the accuracy of their approximations and compare the binary versus nonbinary versions of the algorithms. The sequences used in the binary steering scheme are also studied. A new way of performing the binary steering scheme is proposed to improve the image reconstruction. We also compare the effectiveness of the binary steering in order to make conclusions about the practicality of using one algorithm over another.

CHAPTER 2

ITERATIVE METHODS

2.1 Non-Binary Iterative Methods

The Non-Binary methods used in this project include the ART, CIM, DROP algorithms. This code has beeen outlined in previous research and is used in image processing [GTB08]. Real valued data in discrete tomography(DT) are used to compute the images. Each algorithm uses an iterative formula to approximate the image. Two different phantoms were used in the experimental data. The first phantom is an example of a simple image called "dtphan" that can be easily approximated. The second phantom is the more complex Shepp-Logan phantom; the Shepp-Logan phantom was created as a standard for computerized tomography (CT) image reconstruction simulations of the head. The phantom is also used frequently for magnetic resonance image (MRI) reconstruction and k-space simulations.

Problem Definition

Let $Ax = b$ be a system of linear equations representing the fully discretized model of a two-dimensional image reconstruction from a projection problem. The vector $x = (x_j)_{j=1}^n \in \mathbb{R}^n$, in the n-dimensional Euclidean space, is the *image vector* whose j-th component x_j has the value of the uniform grayness at the j-th pixel. The vector $b = (b_i)_{i=1}^m \in \mathbb{R}^m$ is the *measurement vector* whose i-th component b_i is the value of the i-th line integral through the unknown image. The $m \times n$ projection matrix A is a 0-1 matrix having its i-th row and j-th column element a_j^i equal to zero if the i-th ray does not intersect the j-th pixel, and equal to one if it does. [CM99]

2.1.1 ART Algorithm

The Algebraic Reconstruction Technique (ART) is an iterative algorithm for the reconstruction of a two-dimensional image from a series of one-dimensional angular projections which is used in Computed Tomography scanning. In numerical linear algebra the method is called the Kaczmarz method.

The Kaczmarz method [GBH70], which is based on the work of the Polish mathematician Stefan Kaczmarz, is a method for solving linear systems of equations $Ax = b$. It was rediscovered in the field of image reconstruction from projections by Richard Gordon, Robert Bender, and Gabor Herman in 1970, where it was named the Algebraic Reconstruction Technique (ART). It is applicable to any linear system of equations, but its computational advantage relative to other methods depends on the system being sparse. This method has been found efficacious in the area of image reconstruction from projections. It has been demonstrated to be superior, in some biomedical imaging applications, to other methods such as the filtered backprojection method.

The Kaczmarz method or ART [H09], is an iterative algorithm that has many applications ranging from computed tomography (CT) to signal processing. It can be obtained also by applying to the hyperplanes, described by the linear system, the method of successive projections onto convex sets (POCS). Given a real or complex matrix A and a real or complex vector b, respectively, the method computes an approximation of the solution of the linear systems of equations using the formula below.

Given a $m \times n$ real matrix A and a real vector $b \in \mathbb{R}^n$, where $0 < \lambda < 2$.

In the following formula $\|*\|$ and $\langle *, * \rangle$ are the Euclidean norm and the inner product in $\mathbb{R}^n$, respectively, m is the number of views, a^i is the i-th column of A^T (the transpose of A), λ_k are the relaxation parameters, M is the maximum number of iterations, and w_i^k are positive iteration dependent weights which must sum up (over i) to one, for every $k \geq 0$ [CM99].

Algebraic Reconstruction Technique- ART Algorithm [CEHN08]

Initialization: $x^0 \in \mathbb{R}^n$ is arbitrary.

Iterative Step: Given x^k compute.

for $k = 1$ to M do

for $i = 1$ to m do

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

end do

2.1.2 CIM Algorithm

The Cimmino method is a modificaton of the ART method. The change is to include the relaxation parameters with equal weights $w_i^k = \frac{1}{m}$ for simplicity. Using this system of weights we are able to take advantage of the sparsity of matrix A. This method is suitable for parallel computing.

Cimmino-CIM Algorithm [CEHN08]

Initialization: $x^0 \in \mathbb{R}^n$ is arbitrary.

Iterative Step: Given x^k compute.

for $k = 1$ to M do

for $i = 1$ to m do

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

end do

where $w_i^k = \frac{1}{m}$.

When the matrix A is sparse, which happens often in real-world applications, only a small number of the elements are non-zero but using the CIM algorithm the sum of their contributions is divided by the relatively large m, this slows down the progress of the algorithm. This created a need for a more effective algorithm, the DROP algorithm.

2.1.3 DROP Algorithm

This method allows diagonal component-wise relaxation in conjunction with orthogonal projections onto the individual hyperplanes of the system, which gives it the name diagonally relaxed orthogonal projections (DROP). Diagonal relaxation has been proven useful, by previous mathematicians, in accelerating the initial convergence of simultaneous and block-iterative projection algorithms. However, it was only available in conjunction with generalized oblique projections in which there is a special relation between the weighting and the oblique projections. In the context of this paper DROP is shown to be a modification of the classic ART and Cimmino methods mentioned above. This method replaces the factor $\frac{1}{m}$ in the CIM algorithm by a factor that depends only on the number of non-zero elements in the set, s_j.

Diagonally-Relaxed Orthogonal Projections- DROP Algorithm [CEHN08]

Initialization: $x^0 \in \mathbb{R}^n$ is arbitrary.

Iterative Step: Given x^k compute.

for $k = 1$ to M do

for $i = 1$ to m do

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

end do

where s_j is denoted by the number of nonzero elements in column j of the matrix. A.

2.2 Binary Steering Technique

The Binary Reconstruction Problem is to find a 0-1 vector x^* that is an acceptable approximation to the solution of the system $Ax = b$. Censor and Matej, [CM99] proposed a steering scheme by which non-binary iterative reconstruction algorithms can be steered towards a binary solution of a binary problem. This project piggybacks off of their scheme with a few modifications and further numeric experimentation.

The binary steering mechanism created by Censor and Matej was implemented and used in conjunction with three non-binary iterative algorithms ART, CIM, and DROP.

2.3 Binary Steering Mechanism

The binary steering mechanism has two operations that are used to binarize the iterative methods [CM99].

Step 1:

Given a real number x and two real parameters α and β such that $0 \leq \alpha \leq \beta \leq 1$ we define $\tilde{x}$ by

$$\tilde{x_j} = \begin{cases} 0, & x_j^k \leq \alpha_k \\ 1, & x_j^k \geq \beta_k \\ x_j & \text{otherwise.} \end{cases} \tag{2.1}$$

We binarize each iterate x^k before putting it into the nonbinary iterative algorithm, after the iteration has been performed, a conflict might arise between x^k and the output y^k of the nonbinary iterative algorithm [CM99]. Step 2 will define what it means to have a conflict between x^k and y^k.

Step 2:

Given two real numbers x and y and two real parameters α_k and β_k such that $0 \leq \alpha_k < t$ and $t < \beta_k \leq 1$, where t is given by a threshold and a fixed ϵ, $0 < \epsilon < 0.1$, we define z^k by

$$z_j^k = \begin{cases} t - \epsilon, & x_j^k \leq \alpha_k \ \ and \ \ y_j^k \geq t \\ t + \epsilon, & x_j^k \geq \beta_k \ \ and \ \ y_j^k \leq t \\ y & \text{otherwise.} \end{cases} \tag{2.2}$$

The steering mechanism consists of adding the *Binarizer* and the *Conflict settler* to any nonbinary algorithm [CM99].

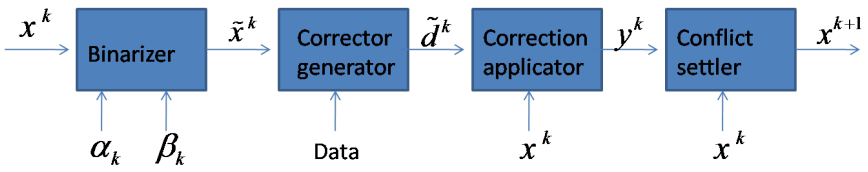

Figure 2.1: Binary steering of the nonbinary algorithm is accomplished by operations of binarization and conflict settlement

Each iteration of the overall process begins with a *(parital) binarization* of the current iterate x^k to form $\tilde{x^k}$ [CM99] (see figure). As the iterations proceed the values α_k keep increasing and the values of β_k keep decreasing so that more components fit into the desired binary 0-1 nature of the vector. The corrector $\tilde{d^k}$ is calculated from $\tilde{x^k}$ by any nonbinary iterative algorithm, not the original x^k, but it is applied by the *correction applicator* to x^k. If the resulting $y^k = x^k + \tilde{d^k}$ has a component that is greater than or equal to the current threshold value t, while its previous value x^k was below α_k, then we say there is a conflict and resolve the conflict by allowing x^k to be only as much as $t - \epsilon$. Thus our final approximate solution contains only 0's and 1's.

$$x_j^{k+1} = \begin{cases} 0, & x_j \leq 0.5 \\ 1, & x_j > 0.5 \end{cases} \tag{2.3}$$

The binary steering process takes the initial x^k and implements Step 1 to binarize each iterate x^k. Then takes a new real value y^k and compares both x^k and y^k in Step 2. The second step corrects any conflict between x^k and y^k and delivers a completely binary solution resulting in z^k which we will call our x^{k+1}.

CHAPTER 3

EXPERIMENTAL STUDY

3.1 The sequence $\{\alpha_k\}$

The original binary steering scheme is a linear sequence such that $\{\alpha_k\}$ and $\{\beta_k\}$ both approach 0.5. Where $\{\alpha_k\}$ approaches from 0 and increases to 0.5 and $\{\beta_k\}$ approaches from 1 and decreases to 0.5, until every $\left\{x^k\right\}$ has been put into sequential binarization.

However, we have contemplated other binary steering schemes in order to test the "speed" of the sequence. The goal was to see if changing the linear sequence to a quadratic, exponential, or square root sequence will decrease the time, number of iterations, and/or error; thus improving the binary steering scheme.

The following tables outline the results of the numerical experiments, where k is defined as each iteration number and M is the given maximum number of iterations in the sequence.

3.1.1 Term Definition

The *time* measures the number of seconds it takes the computer to run the given program. The *number of iterations* measures the amount of iterations it takes to reach the error tolerance of 0.1 or the given iteration maximum, whichever is the least. The *error* is measured as $\|b - A\tilde{x}\|_2$ where b is the desired solution and $\tilde{x}$ is the approximation. Finally, the *number difference* measures the amount of blocks in the approximation image that are different from the original image.

α_k	β_k	Time	Num. of Its.	Error	Number Difference
$\frac{k}{2M}$	$1-\alpha_k$	3.4	27	0	0
$\frac{k^2}{2M^2}$	$1-\alpha_k$	4.3	34	0	0
$1.5^{\frac{k}{M}}-1$	$1-\alpha_k$	3	24	0	0
$.5\sqrt{\frac{k}{M}}$	$1-\alpha_k$	3.3	26	0	0

Table 3.1: Binary Steering ART

The results in Table 3.1 shows that the exponential sequence for $\{\alpha_k\}$ and $\{\beta_k\}$ demonstrates at least approximately a 7 percent improvement in the time and number of iterations.

α_k	β_k	Time	Num. of Its.	Error	Number Difference
$\frac{k}{2M}$	$1-\alpha_k$	123.6	1000	54.9	144
$\frac{k^2}{2M^2}$	$1-\alpha_k$	123.6	1000	50.9	127
$1.5^{\frac{k}{M}}-1$	$1-\alpha_k$	123.6	1000	53.9	140
$.5\sqrt{\frac{k}{M}}$	$1-\alpha_k$	123.3	1000	473.6	1404

Table 3.2: Binary Steering CIM

However, the results in Table 3.2 shows that the quadratic sequence for $\{\alpha_k\}$ and $\{\beta_k\}$ demonstrates at least approximately a 6 percent improvement in the error and 9 percent in the number difference.

α_k	β_k	Time	Num. of Its.	Error	Number Difference
$\frac{k}{2M}$	$1-\alpha_k$	6	55	0	0
$\frac{k^2}{2M^2}$	$1-\alpha_k$	5.3	48	0	0
$1.5^{\frac{k}{M}}-1$	$1-\alpha_k$	6	55	0	0
$.5\sqrt{\frac{k}{M}}$	$1-\alpha_k$	4.9	44	0	0

Table 3.3: Binary Steering DROP

The results in Table 3.3 shows that the square root sequence for $\{\alpha_k\}$ and $\{\beta_k\}$ demonstrates at least approximately a 8 percent improvement in the time and number of iterations.

3.2 Comparison of Binary Steering vs. (Original) NonBinary Steering Algorithms

3.2.1 Numerical Experiment 1

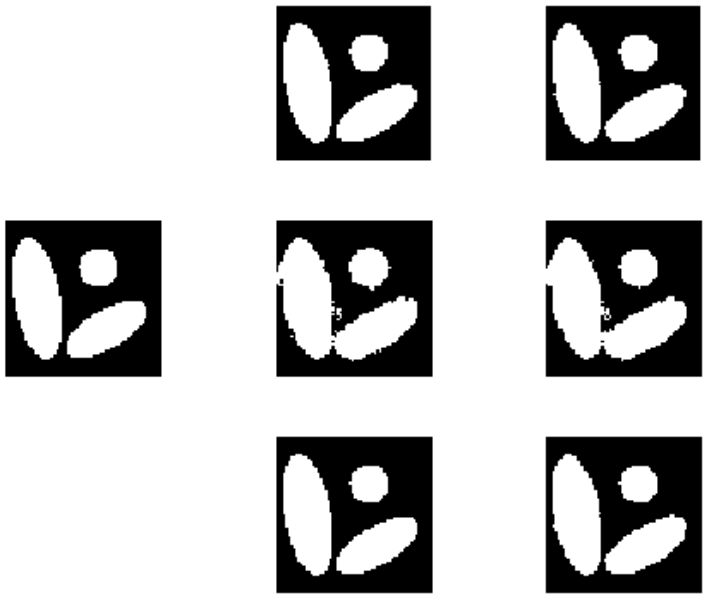

Figure 3.1: Phantom 1 (dtphan): First Column: Original Image, Second Column: Binary ART, Binary CIM, Binary DROP, Third Column:NonBinary ART, NonBinary CIM, NonBinary DROP

In this experiment we used the "dtphan" phantom, 64×64, which represents a simple image that can easily show the effectiveness of each algorithm. The dimension of the system A is 2650×4096, $\lambda_k = 1$ for simplicity, the maximum number of iterations is set at $M = 1000$, and $t = \frac{1}{2}$. Also the error is calculated as the relative error $= \frac{\left\|x^k - x^0\right\|_2}{\|x^0\|_2}$. The first image represents the original image and each subsequent

image are approximations made by the individual algorithms. As demonstrated in Figure 3.1 the ART and DROP algorithms appear to be the best approximations to the image using both the binary and nonbinary versions of each algorithm. The following graphs will show the exact results of the approximations.

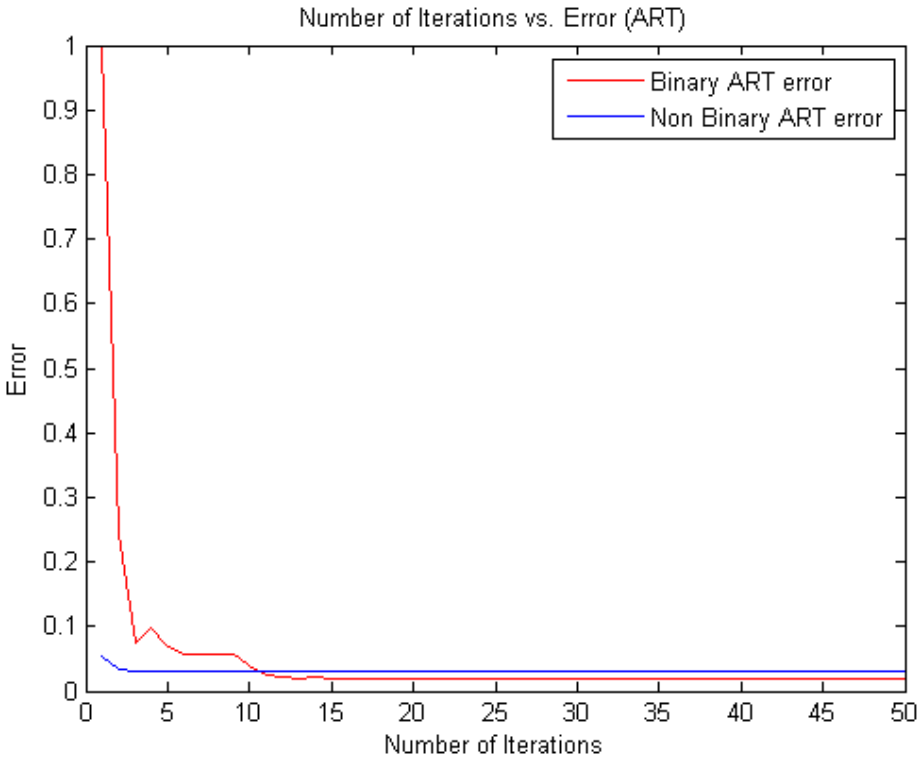

Figure 3.2: Binary ART vs. Non Binary ART

Figure 3.2 is a graph of the error in the binary ART algorithm versus the error in the nonbinary ART algorithm for the "dtphan" phantom. The error in the binary algorithm begins very large and rapidly decreases as the number of iterations increases, and dips below the error of the nonbinary algorithm. However, the error of the nonbinary algorithm remains mostly constant thus requiring less iterations to reach a feasible error tolerance. The maximum number of iterations for this method is $M = 50$.

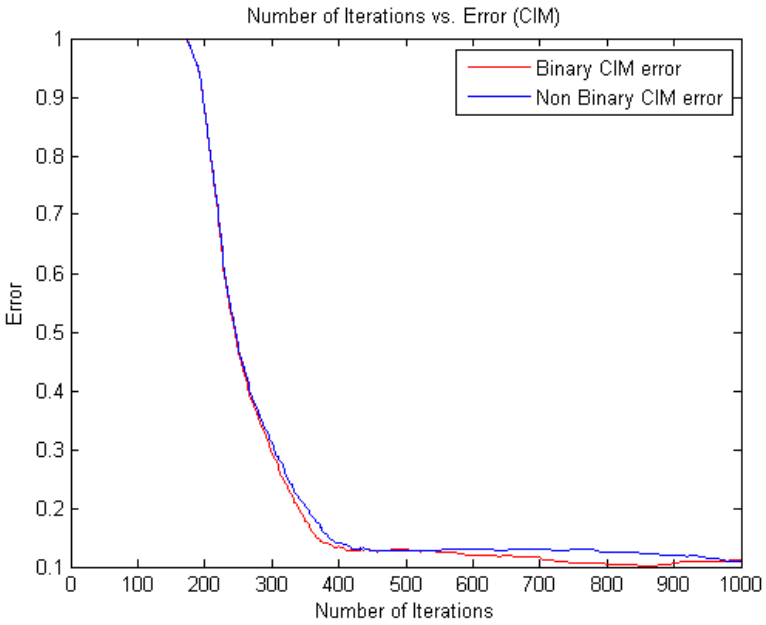

Figure 3.3: Binary CIM vs. Non Binary CIM

Figure 3.3 represents the error of the binary CIM algorithm versus the nonbinary CIM algorithm for the "dtphan" phantom. The error of the two are relatively the same for the maximum number of iterations which was set at 1000. These two methods converged the slowest in comparison with the ART and DROP methods, thus taking the largest number of iterations to converge. This requires a very large number of iterations to be error free. The error begins large for both and steadily decreases until it levels off and converges slowly.

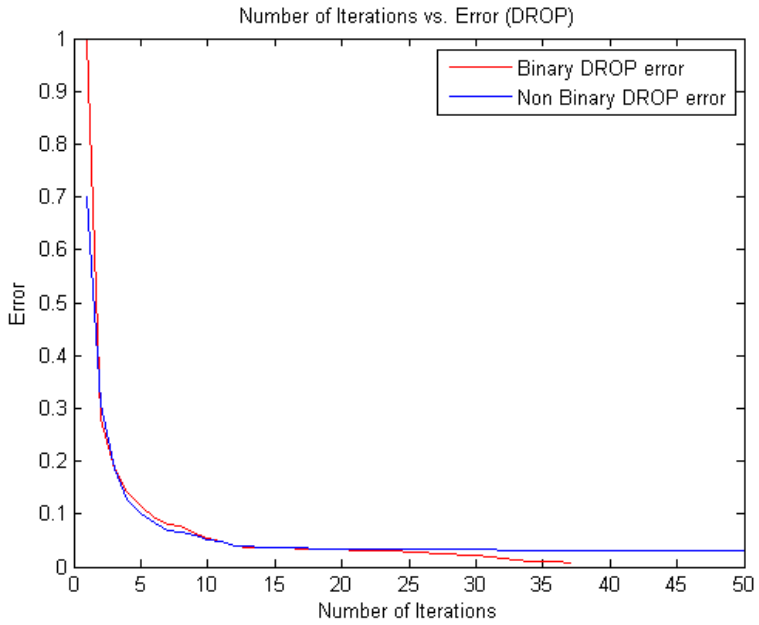

Figure 3.4: Binary DROP vs. Non Binary DROP

Figure 3.4 shows the graph of the error of the binary DROP algorithm versus the nonbinary DROP algorithm for the "dtphan" phantom. Both methods begin with a large error and quickly converge to minimal error. However, the binary algorithm converges completely to an error of 0.0 and uses less iterations than the nonbinary algorithm. The maximum number of iterations for this method is $M = 50$.

The binary DROP algorithm is the best algorithm for the approximation of the "dtphan" phantom. It is the most accurate and requires the least number of iterations to achieve that accuracy.

3.2.2 Numerical Experiment 2

The Shepp-Logan 64×64 phantom is often used in 2-D and 3-D reconstruction literature to present the quality of reconstruction algorithms. It is a well known phantom used in the biomedical imaging area and often used in MATLAB programs. This phantom is much more intricate than the previous image, therefore resulting in less sharp approximations. The dimension of the system A is 2650×4096, $\lambda_k = 1$ for simplicity, the maximum number of iterations is set at $M = 1000$, and $t = \frac{1}{2}$. Also the error is calculated as the relative error $= \frac{\left\|x^k - x^0\right\|_2}{\|x^0\|_2}$.

Figure 3.5: Phantom 2 (Shepp-Logan): First Column: Original Image, Second Column: Binary ART, Binary CIM, Binary DROP, Third Column: NonBinary ART, NonBinary CIM, NonBinary DROP

The above figure represents the original image as well as the approximation images that result from implementing the six approximation algorithms. As demonstrated in Figure 3.5, the DROP algorithm appears to be the best approximation to the image using both the binary and nonbinary versions of each algorithm.

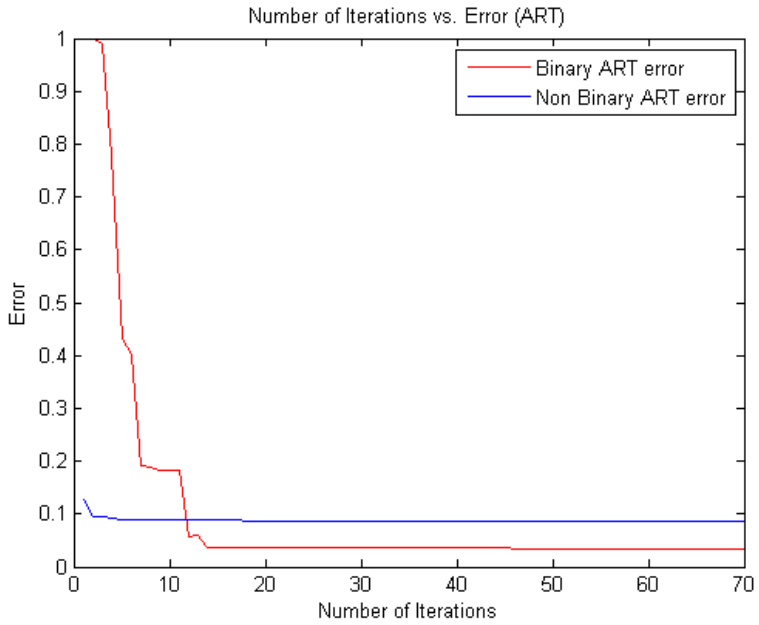

Figure 3.6: Binary ART vs. Non Binary ART

Figure 3.6 is a graph of the error in the binary ART algorithm versus the error in the nonbinary ART algorithm for the Shepp-Logan phantom. The error in the binary algorithm begins very large and rapidly decreases as the number of iterations increases, and dips below the error of the nonbinary algorithm all the way to an error of 0.0. However, the error of the nonbinary algorithm remains mostly constant thus requiring less iterations to reach a feasible error tolerance. The maximum number of iterations for this method is $M = 70$.

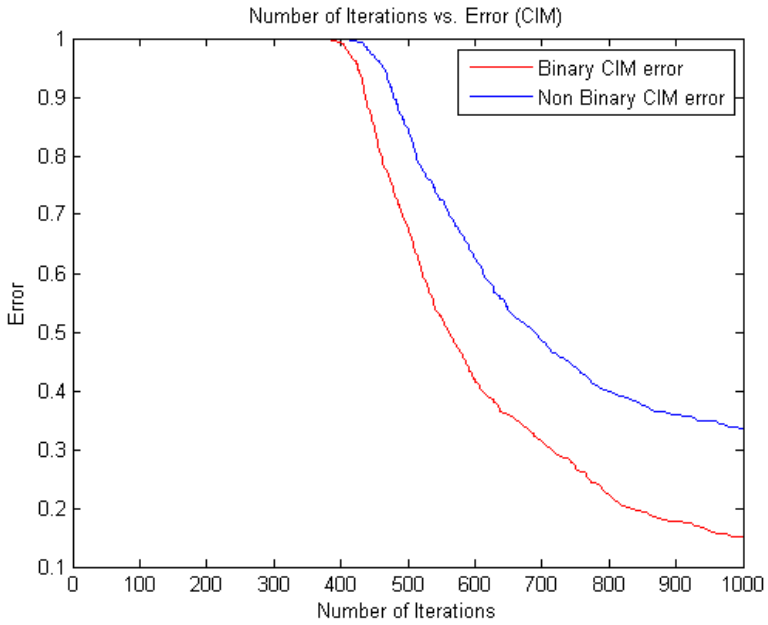

Figure 3.7: Binary CIM vs. Non Binary CIM

Figure 3.7 represents the error of the binary CIM algorithm versus the nonbinary CIM algorithm for the Shepp-Logan phantom. The error of the two are relatively the same for the maximum number of iterations which was set at 1000. These two methods converged the slowest in comparison with the ART and DROP methods, thus taking the largest number of iterations to converge. This requires a very large number of iterations to be error free. The error begins large for both and steadily decreases until it levels off and converges slowly. However, the binary algorithm converges more steap to a smaller error.

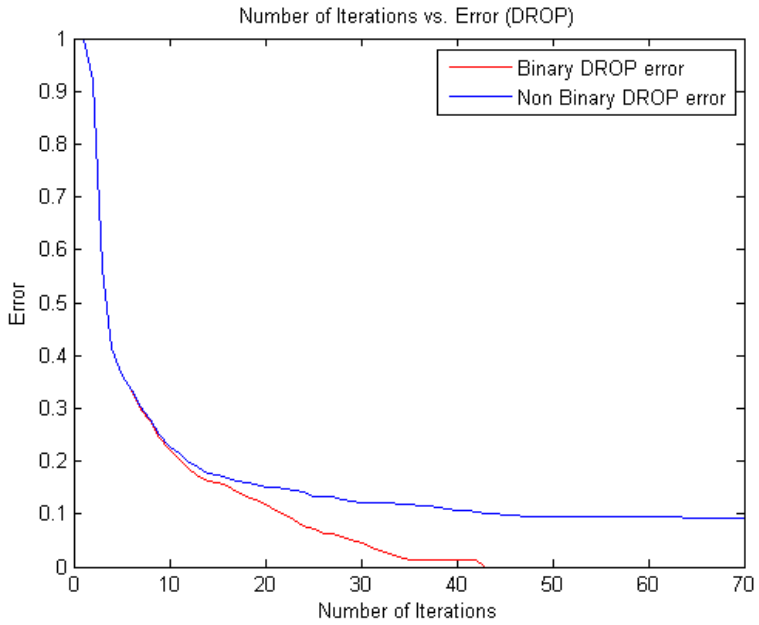

Figure 3.8: Binary DROP vs. Non Binary DROP

Figure 3.8 shows the graph of the error of the binary DROP algorithm versus the nonbinary DROP algorithm for the Shepp-Logan phantom. Both methods begin with a large error and quickly converge to minimal error. However, the binary algorithm converges completely to an error of 0.0 and uses less iterations than the nonbinary algorithm. The maximum number of iterations for this method is $M = 70$.

The binary DROP algorithm is the best algorithm for the approximation of the Shepp-Logan phantom. It is the most accurate and requires the least number of iterations to achieve that accuracy.

3.3 The Effects of the parameter λ_k

The following figures and graphics show the results of changing λ in the algorithms. It has been proven that $\lambda \in (0, 2)$ converges for these particular reconstruction algorithms. We investigated further to see the impact of the value of λ has on the algorithms both before and after using the binary steering scheme.

The objective is to find the "best value" of λ. We define "best value" as minimizing the error and number of iterations. From the numerical experiments conducted we can generally conclude that as λ approaches 2 both the error and number of iterations decrease. The dtphan phantom was used in these experiments. In this experiment we used the "dtphan" 64 phantom. Note: throughout this section λ_k is used as a constant denoted by λ, for simplicity.

3.3.1 Numerical Experiment 3

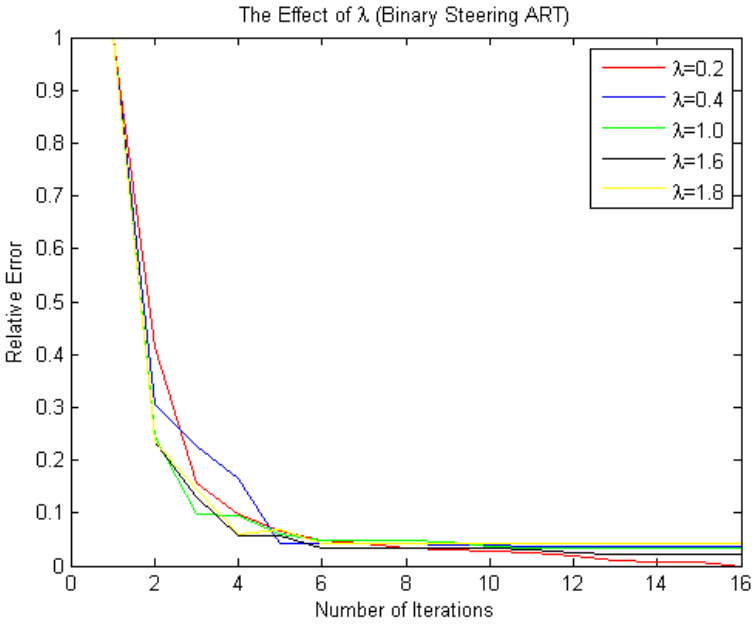

Figure 3.9: Binary Steering ART

The maximum number of iterations for this method is $M = 20$.

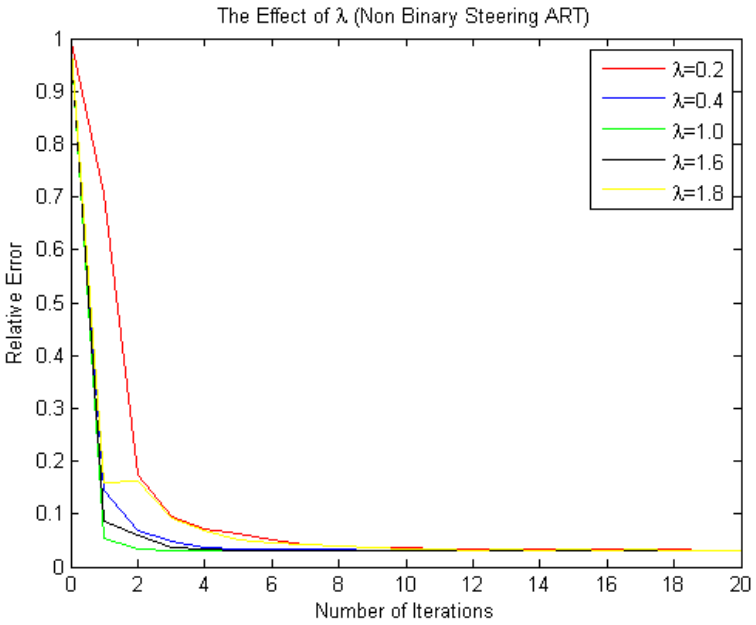

Figure 3.10: Non Binary Steering ART

The maximum number of iterations for this method is $M = 20$.

The value of λ in the ART method, both binary and nonbinary, does not greatly affect the error or the number of iterations needed for the algorithms to converge. However, for the values of $\lambda \geq 1$ show the best results. These results are represented in Figures 3.9 and 3.10.

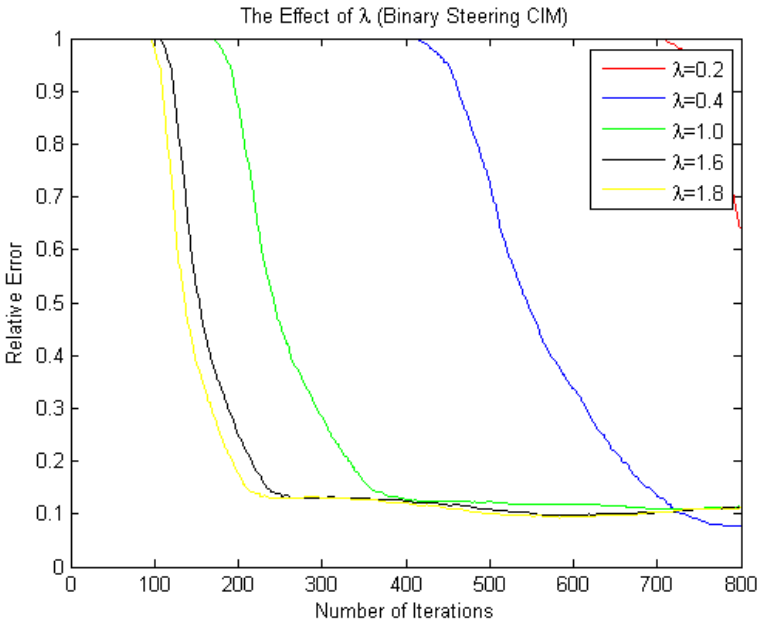

Figure 3.11: Binary Steering CIM

The maximum number of iterations for this method is $M = 800$.

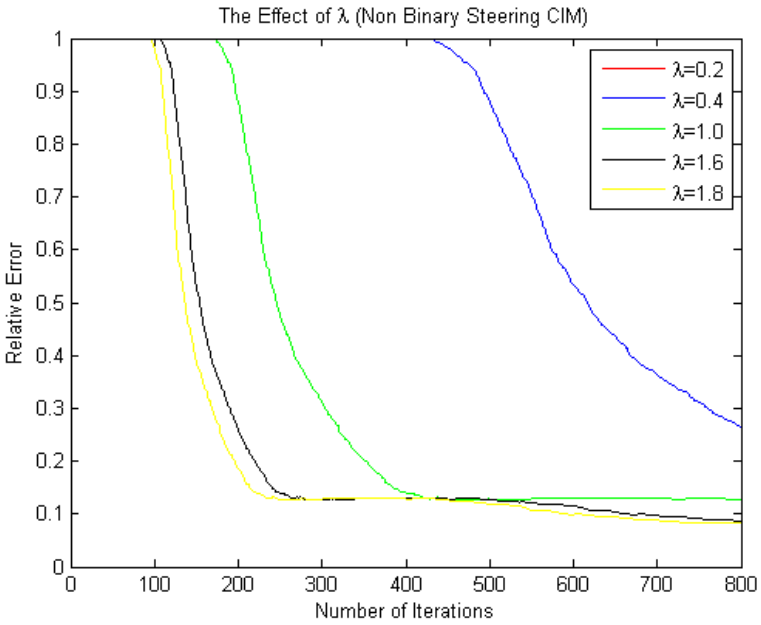

Figure 3.12: Non Binary Steering CIM

The maximum number of iterations for this method is $M = 800$.

The value of λ in the CIM method, both binary and nonbinary, greatly affects the error or the number of iterations needed for the algorithms to converge. However, for the values of $\lambda \geq 1.6$ show the best results. These results are represented in Figures 3.11 and 3.12.

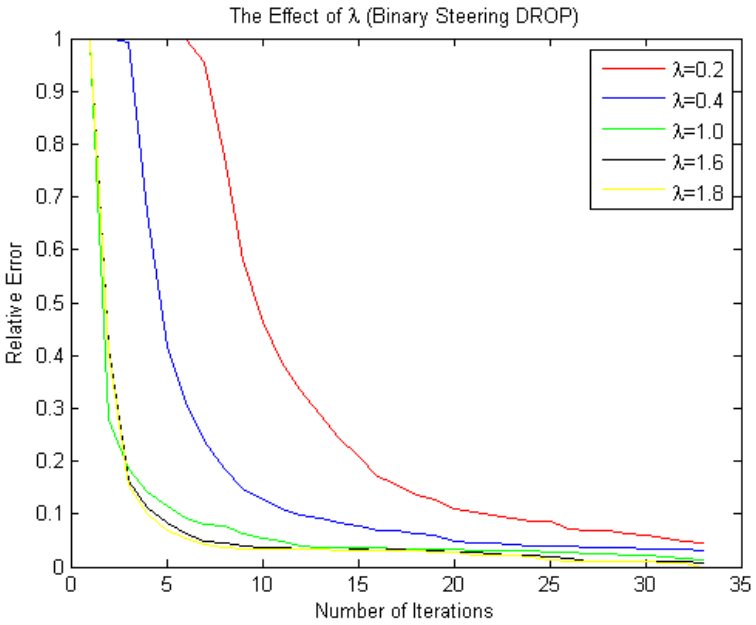

Figure 3.13: Binary Steering DROP

The maximum number of iterations for this method is $M = 50$.

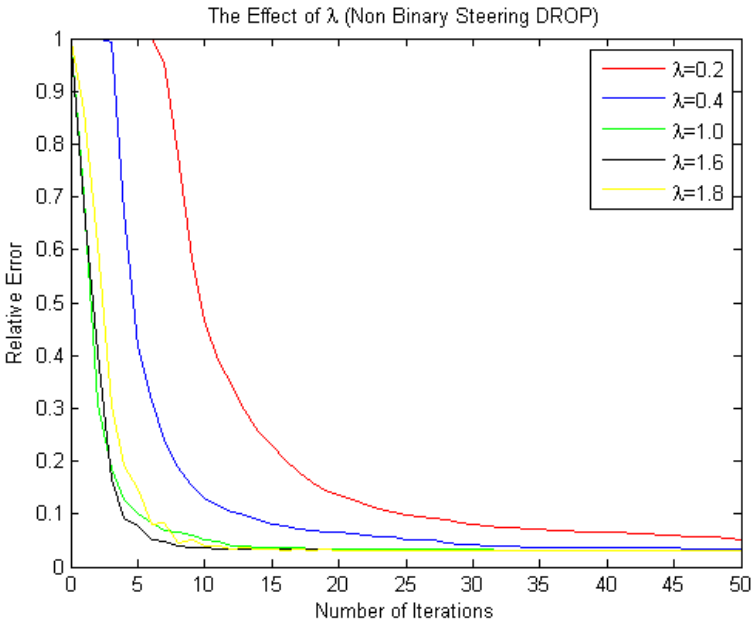

Figure 3.14: Non Binary Steering DROP

The maximum number of iterations for this method is $M = 50$.

The value of λ in the DROP method, both binary and nonbinary, does not greatly affect the error or the number of iterations needed for the algorithms to converge. However, for the values of $\lambda \geq 1$ show the best results. These results are represented in Figures 3.13 and 3.14.

3.3.2 Numerical Experiment 4

The following figures and graphics show the results of changing λ in the algorithms. It has been proven that $\lambda \in (0, 2)$ converges for these particular reconstruction algorithms. We investigated further to see the impact of the value of λ has on the algorithms both before and after using the binary steering scheme.

The objective is to find the "best value" of λ. We define "best value" as minimizing the error and number of iterations. From the numerical experiments conducted we can generally conclude that as λ approaches 2 both the error and number of iterations decrease. The Shepp-Logan 64×64 phantom was used in these experiments.

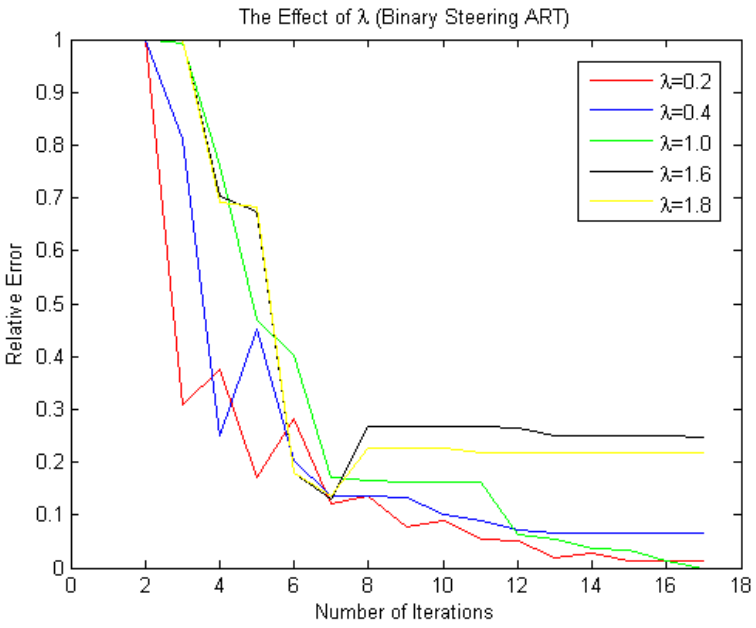

Figure 3.15: Binary Steering ART

The maximum number of iterations for this method is $M = 20$.

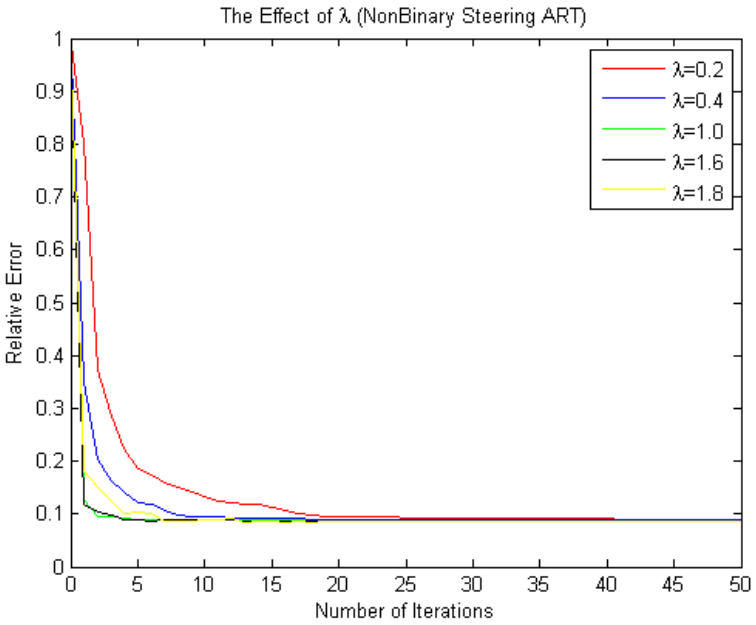

Figure 3.16: Non Binary Steering ART

The maximum number of iterations for this method is $M = 50$.

The value of λ in the ART method, both binary and nonbinary, does not greatly affect the error or the number of iterations needed for the algorithms to converge. However, for the values of $\lambda \geq 1$ show the best results. These results are represented in Figures 3.15 and 3.16.

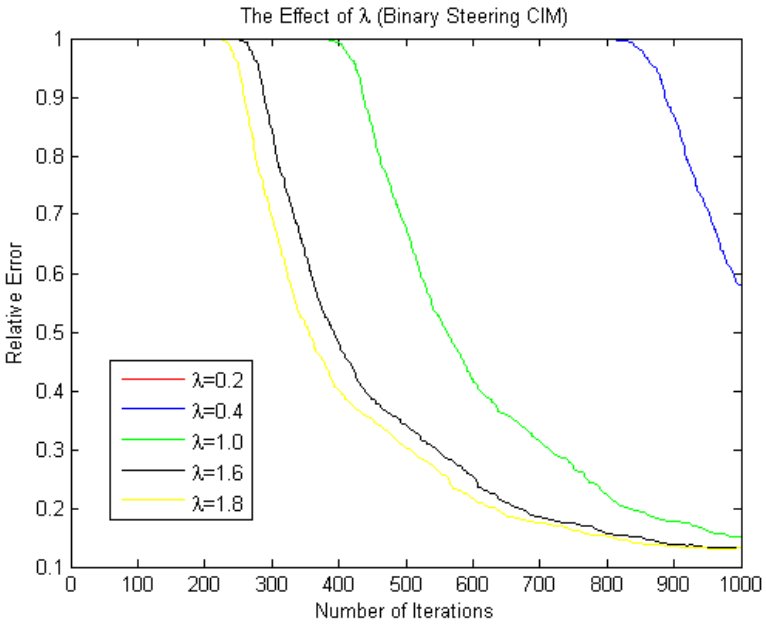

Figure 3.17: Binary Steering CIM

The maximum number of iterations for this method is $M = 1000$.

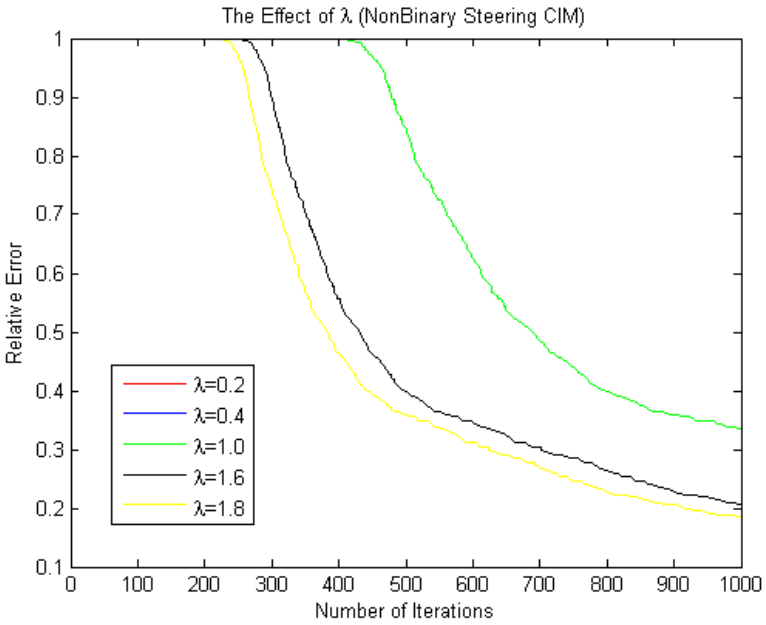

Figure 3.18: Non Binary Steering CIM

The maximum number of iterations for this method is $M = 1000$.

The value of λ in the CIM method, both binary and nonbinary, greatly affects the error or the number of iterations needed for the algorithms to converge. However, for the values of $\lambda \geq 1.6$ show the best results. These results are represented in Figures 3.17 and 3.18.

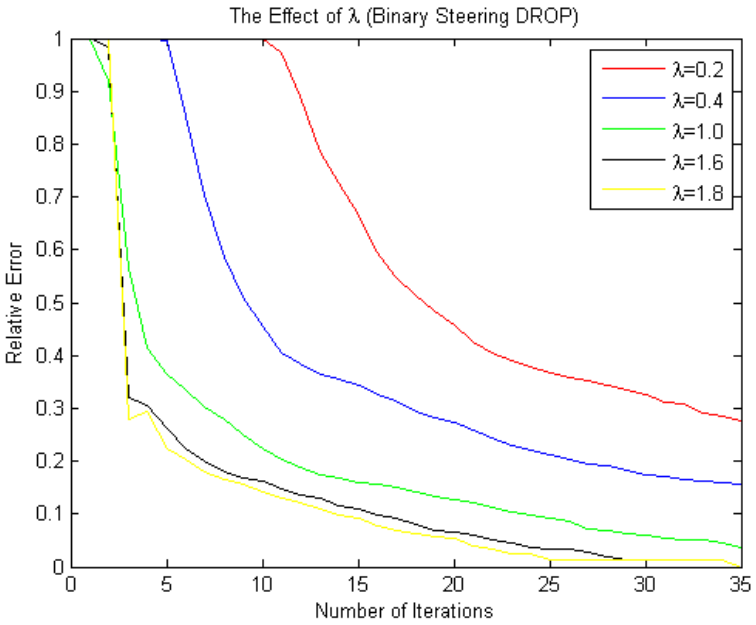

Figure 3.19: Binary Steering DROP

The maximum number of iterations for this method is $M = 50$.

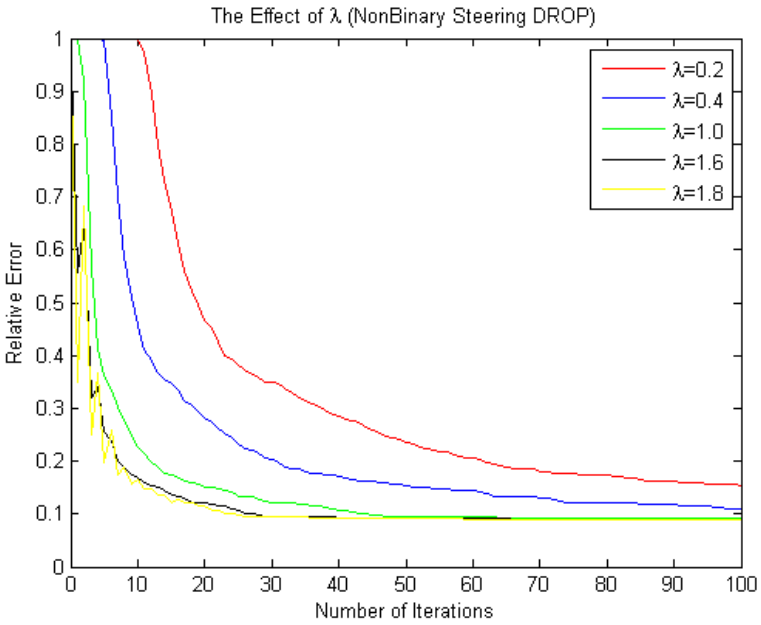

Figure 3.20: Non Binary Steering DROP

The maximum number of iterations for this method is $M = 100$.

The value of λ in the DROP method, both binary and nonbinary, does not greatly affect the error or the number of iterations needed for the algorithms to converge. However, for the values of $\lambda \geq 1$ show the best results. These results are represented in Figures 3.19 and 3.20.

CHAPTER 4

NEW BINARY STEERING SCHEME

4.1 The Scheme

We constructed a new binary steering scheme by adding two parameters in Step 1 of the binary steering scheme described in Chapter 2. The following is our revised binary steering Step 1.

(Revised) Step 1:

Given a real number x and four real parameters α_k , β_k, γ_k, and δ_k such that $0 \leq \alpha_k \leq \gamma_k \leq \delta_k \leq \beta_k \leq 1$ we define $\tilde{x}_j$ by

$$\tilde{x}_j = \begin{cases} 0, & x_j^k \leq \alpha_k \\ \gamma_k, & \gamma_k \leq x_j^k \leq \frac{1}{2} \\ \delta_k, & \frac{1}{2} < x_j^k \leq \delta_k \\ 1, & x_j^k \geq \beta_k \\ x_j, & \text{otherwise.} \end{cases} \tag{4.1}$$

Step 2 follows as previously stated. The results of these changes is the speed of convergence of the binary steering step. The γ_k and δ_k parameters act as additional binarizing steps in order to better rsolve conflicts between the original x^k and y^k.

4.2 Results of Numerical Testing

We apply line projections with a lattice of points. This special case of discrete tomography deals with the reconstruction of a binary image from a small number of projections. It is desirable to limit the number of projections so that we will not

end up with a large number of solutions and to only use the class of images which contains the image being reconstructed. In each of the following cases we have limited the number of projection directions to seven, eight , and twelve respectively.

The following charts demonstrate the difference between using the original binary steering scheme and the revised scheme we have constructed.

Tables 4.1 and 4.2 are the results of using the "dtphan" phantom. The seven projection directions used in Table 4.1 are D_1 = [0 1; 1 0; 1 1; 1 -1; 1 2; 2 -1; 1 -2]. The eight projection directions used in Table 4.2 are D_2 = [0 1; 1 0; 1 1; 1 -1; 1 2; 2 -1; 1 -2; 2 1].

Method	Time	Num. of Its.	Error	Number Difference
ART	26.5	200	14.7	49
ART with $\{\gamma_k\}$ and $\{\delta_k\}$	26.5	200	10.7	20
CIM	26.4	200	138.9	393
CIM with $\{\gamma_k\}$ and $\{\delta_k\}$	26.5	200	82.0	261
DROP	15.8	138	0	0
DROP with $\{\gamma_k\}$ and $\{\delta_k\}$	12.9	113	0	0

Table 4.1: $M = 200$, D_1 = [0 1; 1 0; 1 1; 1 -1; 1 2; 2 -1; 1 -2]

Method	Time	Num. of Its.	Error	Number Difference
ART	4.3	26	0	0
ART with $\{\gamma_k\}$ and $\{\delta_k\}$	4.1	25	0	0
CIM	32.2	200	150.4	402
CIM with $\{\gamma_k\}$ and $\{\delta_k\}$	32.5	200	105.3	263
DROP	12.7	91	0	0
DROP with $\{\gamma_k\}$ and $\{\delta_k\}$	9.6	69	0	0

Table 4.2: $M = 200$, $D_2 = [0\ 1;\ 1\ 0;\ 1\ 1;\ 1\ -1;\ 1\ 2;\ 2\ -1;\ 1\ -2;\ 2\ 1]$

The new binary steering scheme including the γ_k and δ_k sequences have shown improvement in all three algorithms for the "dtphan" phantom. The greatest improvement is in the binary CIM algorithms where both the error and the number difference has decreased. In the binary DROP algorithms the number of iterations have decreased in both cases. Lastly, the binary ART algorithm using seven different directions decreased the error and number difference, but the binary ART algorithm using eight different projection directions basically had no change.

Table 4.3 are the results from using the Shepp-Logan phantom. The twelve projection directions used in Table 4.3 are D_3 = [0 1; 1 0; 1 1; 1 -1; 1 3; 3 -1; 1 -3; 3 1; 2 3; 3 -2; 2 -3; 3 2].

Method	Time	Num. of Its.	Error	Number Difference
ART	9.0	19	0	0
ART with $\{\gamma_k\}$ and $\{\delta_k\}$	14.3	30	0	0
CIM	469.6	1000	42.0	103
CIM with $\{\gamma_k\}$ and $\{\delta_k\}$	468.8	1000	39.6	90
DROP	26.6	64	0	0
DROP with $\{\gamma_k\}$ and $\{\delta_k\}$	24.2	58	0	0

Table 4.3: $M = 200$ for ART and DROP, $M = 1000$ for CIM, D_3 = [0 1; 1 0; 1 1; 1 -1; 1 3; 3 -1; 1 -3; 3 1; 2 3; 3 -2; 2 -3; 3 2]

The new binary steering scheme including the γ_k and δ_k sequences have shown improvement in the binary CIM and DROP algorithms for the Shepp-Logan phantom. However, the new scheme actually made the binary ART algorithm worse in this case. The greatest improvement is in the binary CIM algorithms where both the error and the number difference has decreased. In the binary DROP algorithms the number of iterations have decreased.

In the original binary steering method the components of x^k, are adjusted to a binary integer less than α_k and greater than β_k. In our new binary steering scheme, in addition to the method mentioned in Chapter 2, all of the components of x_k, which are located in the interval (γ_k, δ_k), are adjusted to either γ_k or δ_k. This speeds up the *binarization* of the algorithm.

CHAPTER 5

CONCLUSION

5.1 Summary

In conclusion, tomography deals with the problem of determining shape and dimensional information of an object from a set of projections. The object corresponds to a function; and the problem posed is to reconstruct this function from its integrals or sums over subsets of its domain. It is typical in discrete tomography that only a few projections (line sums) are used. In this case, conventional techniques fail. The reconstruction methods used in DT applications are usually based on some formulation as an optimization problem. According to theoretical and practical results, in some cases just a few views are sufficient for high-quality reconstruction of objects.

Reconstruction algorithms have many applications in image processing, medicine, three-dimensional statistical data security problems, computer tomography assisted engineering and design, and electron microscopy. The mathematical theory of DT is based mostly on discrete mathematics but also uses functional analysis, combinatorics, geometry, optimization, and algebra.

The binary steering process [CM99] is a method designed to change between consecutive steps of a nonbinary iterative image reconstruction algorithm in order to gradually steer the iterates towards a binary solution. In other words, it is a steering scheme by which nonbinary iterative reconstruction algorithms can be steered towards a binary solution of a binary problem. This paper focused on the binary steering scheme on three particular nonbinary discrete tomography (DT) algorithms to recontruct the problem into a binary problem.

Three major reconstruction algorithms were introduced and manipulated in order to test the accuracy of their approximations and to compare and contrast the binary versus nonbinary versions of the algorithms. Also to compare the effectiveness of the different algorithms in order to make conclusions about the practicality of using one algorithm over another.

Our numerical results uncovered an improved binary steering scheme in most cases as well as a deeper understanding of the λ_k weight on our algorithms. We were also able to test different sequences for our α_k and β_k.

5.2 Future Study

Future work in this field will include further exploration of the new binary steering scheme using the sequences $\{\gamma_k\}$ and $\{\delta_k\}$. As well as proving that there is one particular λ that is optimal for all the reconstruction algorithms and proving that this value has the "best" convergence. We would also like to explore the impact the number of projection directions has on our binary steering algorithms.

CHAPTER 6

APPENDIX

6.1 Main Program

```
% This program reconstructs an image by CS based interior tomography
% algorithm
clear;
%load dtphan2.mat;
%D = [0 1; 1 0; 1 1; 1 -1; 1 2; 2 -1; 1 -2; 2 1];
load shepplogan64.mat
D = [0 1; 1 0; 1 1; 1 -1; 1 3; 3 -1; 1 -3; 3 1; 2 3; 3 -2; 2 -3; 3 2];
[m,n] = size(f);
% set up a system whose exact solution is reshape(F',m*n,1)
[A, b] = setup(f, D);
exactx = reshape(f', m*n,1);
%err3 = norm(b-A*exactx)
% solve the system by calling a function you developed
M = 200;
lambda = 1;
tic;
[x1, numit1, err1] = bisteer_ART(A, b, M);
time1 = toc;
tic;
[x2, numit2, err2] = NONbisteer_ART(A, b, M);
time2 = toc;
```

```
tic;
[x3, numit3, err3] = bisteer_CIM(A, b, M);
time3 = toc;
tic;
[x4, numit4, err4] = NONbisteer_CIM(A, b, M, lambda);
time4 = toc;
tic;
[x5, numit5, err5] = bisteer_DROP(A, b, M);
time5 = toc;
tic;
[x6, numit6, err6] = NONbisteer_DROP(A, b, M);
time6 = toc;
G = G';
G1 = reshape(x1,n,m);  G1 = G1';
G2 = reshape(x2,n,m);  G2 = G2';
G3 = reshape(x3,n,m);  G3 = G3';
G4 = reshape(x4,n,m);  G4 = G4';
G5 = reshape(x5,n,m);  G5 = G5';
G6 = reshape(x6,n,m);  G6 = G6';
% output the image G and compare it with the original phantom F
figure(1);
subplot(3,3,4); imshow(f);
subplot(3,3,2); imshow(G1);
subplot(3,3,3); imshow(G2);
subplot(3,3,5); imshow(G3);
```

```
subplot(3,3,6); imshow(G4);
subplot(3,3,8); imshow(G5);
subplot(3,3,9); imshow(G6);
disp('-----------------------------------------------------')
disp('                Binary Steering ART');
disp('-----------------------------------------------------')
disp('Time     Iterations       Error      Number Difference')
numit1;
err_realsol1 = norm(A*x1-b);
num_diff1 = sum(sum(abs(f-G1)));
fprintf('%4.1f    %6d     %11.1f     %11d\n\n',  time1, numit1, err_realsol1, num_diff1)
disp('-----------------------------------------------------')
disp('                NONBinary Steering ART');
disp('-----------------------------------------------------')
disp('Time     Iterations       Error      Number Difference')
numit2;
err_realsol2 = norm(A*x2-b);
num_diff2 = sum(sum(abs(f-G2)));
fprintf('%4.1f    %6d     %11.1f     %11d\n\n',  time2, numit2, err_realsol2, num_diff2)
disp('-----------------------------------------------------')
disp('                Binary Steering CIM');
disp('-----------------------------------------------------')
disp('Time     Iterations       Error      Number Difference')
numit3;
err_realsol3 = norm(A*x1-b);
```

```
num_diff3 = sum(sum(abs(f-G3)));
fprintf('%4.1f    %6d     %11.1f     %11d\n\n',  time3, numit3, err_realsol3, num_diff3)
disp('-----------------------------------------------------')
disp('                NONBinary Steering CIM');
disp('-----------------------------------------------------')
disp('Time     Iterations       Error      Number Difference')
numit4;
err_realsol4 = norm(A*x1-b);
num_diff4 = sum(sum(abs(f-G4)));
fprintf('%4.1f    %6d     %11.1f     %11d\n\n',  time4, numit4, err_realsol4, num_diff4)
disp('-----------------------------------------------------')
disp('                Binary Steering DROP');
disp('-----------------------------------------------------')
disp('Time     Iterations       Error      Number Difference')
numit5;
err_realsol5 = norm(A*x1-b);
num_diff5 = sum(sum(abs(f-G5)));
fprintf('%4.1f    %6d     %11.1f     %11d\n\n',  time5, numit5, err_realsol5, num_diff5)
disp('-----------------------------------------------------')
disp('                NONBinary Steering DROP');
disp('-----------------------------------------------------')
disp('Time     Iterations       Error      Number Difference')
numit6;
err_realsol6 = norm(A*x1-b);
num_diff6 = sum(sum(abs(f-G6)));
```

```
fprintf('%4.1f    %6d     %11.1f     %11d\n\n',  time6, numit6, err_realsol6, num_diff6)
t1 = 1 : numit1;  err1a = err1(1:numit1);
t2 = 1 : numit2;  err2a = err2(1:numit2);
t3 = 1 : numit3;  err3a = err3(1:numit3);
t4 = 1 : numit4;  err4a = err4(1:numit4);
t5 = 1 : numit5;  err5a = err5(1:numit5);
t6 = 1 : numit6;  err6a = err6(1:numit6);
figure(2);
plot(t1, err1a,'r',t2, err2a, 'b');
legend('Binary ART error','Non Binary ART error')
title('Number of Iterations vs. Error (ART)')
xlabel('Number of Iterations');
ylabel('Error');
figure(3);
plot(t3, err3a, 'r', t4, err4a, 'b');
legend('Binary CIM error', 'Non Binary CIM error');
title('Number of Iterations vs. Error (CIM)');
xlabel('Number of Iterations');
ylabel('Error');
figure(4);
plot(t5, err5a, 'r', t6, err6a, 'b');
legend('Binary DROP error','Non Binary DROP error');
title('Number of Iterations vs. Error (DROP)');
xlabel('Number of Iterations');
ylabel('Error');
```

6.2 Code for Numerical Experiments 3 and 4

```
% This program reconstructs an image by CS based interior tomography
% algorithm
clear;
load dtphan2.mat;
D = [0 1; 1 0; 1 1; 1 -1; 1 2; 2 -1; 1 -2; 2 1];
%load shepplogan64.mat
%D = [0 1; 1 0; 1 1; 1 -1; 1 3; 3 -1; 1 -3; 3 1; 2 3; 3 -2; 2 -3; 3 2];
[m,n] = size(f);
% set up a system whose exact solution is reshape(F',m*n,1)
[A, b] = setup(f, D); tic;
normb= norm(b);
exactx = reshape(f', m*n,1);
% solve the system by calling a function you developed
M = 50;
lambda =0.2;
[x, numit1, err1] = Bisteer_DROP(A, b, M, lambda);
%[x, numit1, err1] = NONBisteer_DROP(A, b, M, lambda);
%[x, numit1, err1] = Bisteer_CIM(A, b, M, lambda);
%[x, numit1, err1] = NONBisteer_CIM(A, b, M, lambda);
%[x, numit1, err1] = Bisteer_ART(A, b, M, lambda);
%[x, numit1, err1] = NONBisteer_ART(A, b, M, lambda);
lambda =0.4;
[x, numit2, err2] = Bisteer_DROP(A, b, M, lambda);
%[x, numit2, err2] = NONBisteer_DROP(A, b, M, lambda);
```

```
%[x, numit2, err2] = Bisteer_CIM(A, b, M, lambda);
%[x, numit2, err2] = NONBisteer_CIM(A, b, M, lambda);
%[x, numit2, err2] = Bisteer_ART(A, b, M, lambda);
%[x, numit2, err2] = NONBisteer_ART(A, b, M, lambda);
lambda =1.0;
[x, numit3, err3] = Bisteer_DROP(A, b, M, lambda);
%[x, numit3, err3] = NONBisteer_DROP(A, b, M, lambda);
%[x, numit3, err3] = Bisteer_CIM(A, b, M, lambda);
%[x, numit3, err3] = NONBisteer_CIM(A, b, M, lambda);
%[x, numit3, err3] = Bisteer_ART(A, b, M, lambda);
%[x, numit3, err3] = NONBisteer_ART(A, b, M, lambda);
lambda =1.6;
[x, numit4, err4] = Bisteer_DROP(A, b, M, lambda);
%[x, numit4, err4] = NONBisteer_DROP(A, b, M, lambda);
%[x, numit4, err4] = Bisteer_CIM(A, b, M, lambda);
%[x, numit4, err4] = NONBisteer_CIM(A, b, M, lambda);
%[x, numit4, err4] = Bisteer_ART(A, b, M, lambda);
%[x, numit4, err4] = NONBisteer_ART(A, b, M, lambda);
lambda =1.8;
[x, numit5, err5] = Bisteer_DROP(A, b, M, lambda);
%[x, numit5, err5] = NONBisteer_DROP(A, b, M, lambda);
%[x, numit5, err5] = Bisteer_CIM(A, b, M, lambda);
%[x, numit5, err5] = NONBisteer_CIM(A, b, M, lambda);
%[x, numit5, err5] = Bisteer_ART(A, b, M, lambda);
%[x, numit5, err5] = NONBisteer_ART(A, b, M, lambda);
```

```
err1=err1/normb; err1 =[1; err1];
err2=err2/normb; err2 =[1; err2];
err3=err3/normb; err3 =[1; err3];
err4=err4/normb; err4 =[1; err4];
err5=err5/normb; err5 =[1; err5];
numit_f= min(numit1, numit2);
numit_s= min(numit_f, numit3);
numit_t= min(numit_s, numit4);
numit_l= min(numit_t, numit5);
cycle= 0: numit_l;
plot(cycle, err1(1:numit_l+1),'r', cycle, err2(1:numit_l+1),'b',
cycle, err3(1:numit_l+1),'g', cycle, err4(1:numit_l+1),'k', cycle,
err5(1:numit_l+1),'y');
legend('\lambda=0.2','\lambda=0.4','\lambda=1.0','\lambda=1.6','\lambda=1.8');
title('The Effect of \lambda (Binary Steering DROP)');
xlabel('Number of Iterations');
ylabel('Relative Error');
```

6.3 Code for New Binary Steering Scheme

```
% M-file bisteer_ART.m
% defined bisteer_ART function
%This function demonstrates the ART algorithm using the new binary steering method.
function [xx, numit1, err1] = bisteer_ART(A, b, M, lambda)
[m,n]=size(A);
```

```
eps = 0.1;
eps2 = 0.01;
x = zeros(n,1);
xt = zeros(n,1);
y = zeros(n,1);
z = zeros(n,1);
lambda = 1;
numit = M;
for j = 1 : M
    alpha = j/(2*M); beta = 1- alpha; gamma = 1/2 -alpha; delta = 1/2 + alpha;
    for k = 1:n
      if x(k) <= alpha
        xt(k) = 0;
      elseif x(k) >=gamma && x(k) <=1/2
          xt(k) = gamma;
      elseif x(k) > 1/2 && x(k) <= delta
          xt(k) = delta;
      elseif x(k) >= beta
        xt(k) = 1;
      else
        xt(k) = x(k);
      end
    end
    xtemp = x;
    for i = 1: m
```

```
     coef(i) =( b(i) - A(i,:)*xt)/norm(A(i,:),2)^2;
     xtemp = xtemp + lambda*coef(i)*A(i,:)';
 end
 y = xtemp;
for k=1:n
  if x(k) <= alpha && y(k) <= alpha
    z(k)= 0;
  elseif   x(k) >= beta && y(k) >= beta
    z(k) = 1;
  elseif x(k) <= alpha && y(k) >= 0.5
    z(k)= 0.5 - eps;
  elseif x(k) >= beta && y(k) <= 0.5
    z(k)= 0.5 + eps;
  else
    z(k) = y(k);
  end
end
x = z;
xx = zeros(n, 1);
xx(find(x>=0.5)) = 1;
err1 = norm(b - A*xx);
if err1 < eps2
     numit = j;
    break;
end
```

```
end
```

BIBLIOGRAPHY

[C01] Y. Censor, *Binary Steering in Discrete Tomography Reconstruction with Sequential and Simultaneous Iterative Algorithms*

[CEHN08] Y. Censor, T. Elfving, GT Herman, and T. Nikazad, *On diagonally-relaxed orthogonal projection methods.*, SIAM Journal on Scientific Computing, 30:473-504 (2008).

[CM99] Y. Censor and S. Matej, *Binary Steering of Nonbinary Iterative Algorithms.* In Discrete Tomography: Foundations, Algorithms, and Applications, 285-296, GT Herman, A Kuba, Eds. Boston: Birkhauser.(Boston) (1999).

[GTB08] MH Gach, C. Tanase, and F. Boada, *Systems Engineering, International Conference on*, In Systems Engineering, ICSENG '08. 19^{th} International Conference on, Vol. 0, pp. 521-526 (2008).

[GBH70] R. Gordon, R. Bender, and GT Herman, *Algebraic reconstruction techniques (ART) for threedimensional electron microscopy and x- ray photography.* Journal of Theoretical Biology, Vol. 29, pp. 471-481, (1970).

[H09] GT Herman, *Fundamentals of computerized tomography: Image reconstruction from projection*, 2nd edition, Springer, (2009).

[HK03] GT Herman and A. Kuba, *Discrete Tomography in Medical Imaging*, Proceedings of the IEEE, Vol. 91, No. 10 (2003).

[JW02] M. Jiang and G. Wang, *Development of iterative algorithms for image reconstruction*, IOS Press (2002).

[KH99] A. Kuba and GT Herman, *Chapter 1: Discrete Tomography: A Historical Overview* (1999).

[ZLYW08] J. Zhu, X. Li, Y. Ye, and G. Wang, *Analysis on the strip-based projection model for discrete tomography*, Science Direct Journal on Discrete Applied Mathematics, 156:2359-2367 (2008).