Distribution Agreement

In presenting this thesis or dissertation as a partial fulfillment of the requirements for an advanced degree from Emory University, I hereby grant to Emory University and its agents the non-exclusive license to archive, make accessible, and display my thesis or dissertation in whole or in part in all forms of media, now or hereafter known, including display on the world wide web. I understand that I may select some access restrictions as part of the online submission of this thesis or dissertation. I retain all ownership rights to the copyright of the thesis or dissertation. I also retain the right to use in future works (such as articles or books) all or part of this thesis or dissertation.

Signature:

_____ _____

Veronica Mejia Bustamante Date

Iterative Polyenergetic Digital Tomosynthesis Reconstructions
for Breast Cancer Screening

By

Veronica Mejia Bustamante
Doctor of Philosophy

Mathematics and Computer Science

James G. Nagy, PhD.
Advisor

Michele Benzi, PhD.
Committee Member

Alessandro Veneziani, PhD.
Committee Member

Accepted:

Lisa A. Tedesco, PhD.
Dean of the James T. Laney School of Graduate Studies

Date

Iterative Polyenergetic Digital Tomosynthesis Reconstructions for Breast Cancer Screening

By

Veronica Mejia Bustamante
M.Sc., Emory University, 2011
B.S. with Highest Honors, Emory University, 2008

Advisor: James G. Nagy, PhD.

An abstract of
A dissertation submitted to the Faculty of the
James T. Laney School of Graduate Studies of Emory University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy
in Mathematics and Computer Science

2013

Abstract

Iterative Polyenergetic Digital Tomosynthesis Reconstructions
for Breast Cancer Screening

By
Veronica Mejia Bustamante

In digital tomosynthesis imaging, multiple projections of an object are obtained along a small range of incident angles in order to reconstruct a pseudo 3D representation of the object. This technique is of relevant interest in breast cancer screening since it eliminates the problem of tissue superposition that reduces clinical performance in standard mammography. The challenge of this technique is that it is computationally and memory intensive, as it deals with millions of input pixels in order to produce a reconstruction composed of billions of voxels. Standard approaches to solve this large-scale inverse problem have relied on simplifying the physics of the image acquisition model by considering the x-ray beam to be monoenergetic, thus decreasing the number of degrees of freedom and the computational complexity of the solution. However, this approach has been shown to introduce beam hardening artifacts to the reconstructed volume. Beam hardening occurs when there is preferential absorption of low-energy photons from the x-ray by the object, thus changing the average energy of the x-ray beam.

This thesis presents an interdisciplinary collaboration to overcome the mathematical, computational, and physical constraints of standard reconstruction methods in digital tomosynthesis imaging. We begin by developing an accurate polyenergetic mathematical model for the image acquisition process and propose a stable numerical framework to iteratively solve the nonlinear inverse problem arising from this model. We provide an efficient and fast implementation of the volume reconstruction process that exploits the parallelism available on the GPU architecture. Under our framework, a full size clinical data set can be reconstructed in under five minutes. The implementation presented reduces storage and communication costs by implicitly storing operators and increasing kernel functionality. We show that our reconstructed volume has no beam hardening artifacts and has better image quality than standard reconstruction methods. Our reconstructions also provide a quantitative measure for each voxel of the volume, allowing the physician to see and measure the contrast between materials present inside the breast. The research presented in this thesis shows that large-scale medical image reconstructions can be done using physically accurate models by effectively harnessing the multi-threading power of GPUs.

Iterative Polyenergetic Digital Tomosynthesis Reconstructions for Breast Cancer Screening

By

Veronica Mejia Bustamante
M.Sc., Emory University, 2011
B.S. with Highest Honors, Emory University, 2008

Advisor: James G. Nagy, PhD.

A dissertation submitted to the Faculty of the
James T. Laney School of Graduate Studies of Emory University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy
in Mathematics and Computer Science

2013

Acknowledgements

I am forever indebted to Emory University which has been the stage for my life for almost a decade. I would not be where I am today without the opportunities, challenges, and support that the university with its faculty and staff have given me. I would especially like to thank the faculty and staff at Oxford College for their commitment to building a community of scholars and future leaders. It was in my years at Oxford that I developed the intellectual curiosity that led to this work and still guides me through the world today.

I would like to thank my advisor Professor James Nagy for his guidance and support through this work. Thank you so much for teaching me and seeing what I am capable of before I knew I could do it. Thank you for providing amazing opportunities for me to travel and to enhance my education by encouraging me to learn new things from colleagues and friends. It really was a pleasure working with you. Thank you also to my committee members Professors Michele Benzi and Alessandro Veneziani for your time and suggestions in completing this dissertation. It has been a privilege to be in your classrooms and learn from your expertise.

Thank you to the Mathematics and Computer Science department, faculty, staff, students and researchers. The department has been my home as an undergraduate and graduate student and I will miss it dearly. Thank you especially to Terry Ingram for all your help. Professor Vaidy Sunderam, thank you so much for introducing me to the world of computer science and parallel programing, it truly changed the course of my career. Thank you also to all my collaborators for your help in putting this work together.

Thank you to the Computational Physics team at Los Alamos National Laboratory and Dr. Ben Bergen for introducing me to the world of GPU programming by hosting me as a summer intern. Thank you also to the Numerical Device Modeling team and Dr. Tom Linton at Intel Corporation for introducing me to large scale parallel software engineering.

This work would not be possible without the support of my best friend and soulmate, my husband Aaron. You are my rock and my inspiration to be better. Thank you for accepting to take this journey with me from the beginning, for making this work a priority in our lives, and for keeping the faith in the moments it was hard for me to see the light. You keep me grounded and remind me what is truly important in life. In the words of Shawn Carter, “Never in bunches, just me and you, I loved your point of view cause you held no punches”.

Last but not least, I would like to thank my family for being with me every step of the way. Near and far, I have felt you close and enjoyed sharing the various points of this journey with you. To Gloria and Felipe, thank you for being my teammates and always believing in me, even when the rest of the world would not. Finally, I would like to acknowledge my father-in-law Gilbert, the one person who was most excited to see the completion of this dissertation. Your enthusiasm and curiosity for this work gave me the motivation to keep going forward when things became difficult. Thank you for your love and encouragement through this process and even though you cannot be with us to celebrate this accomplishment I know you are proud and we miss you.

Para mi madre, la mujer que me enseñó el valor del esfuerzo y la tenacidad con su ejemplo

To my mother, the woman who taught me the value of dedication and hard work by example

Contents

List of Figures **xii**

List of Tables **xv**

1 Introduction **2**

1.1 Background 2

1.2 Inverse Problems and Regularization 5

1.2.1 Linear Inverse Problems 6

1.2.2 Separable Inverse Problems 10

1.2.3 Nonlinear Inverse Problems 15

1.3 Outline of Work 17

1.4 Contributions 19

2 The Physics of X-ray Imaging and Digital Tomosynthesis **22**

2.1 Producing a Radiograph 23

2.1.1 X-ray Production 24

2.1.2 Interaction Between X-rays and Matter 26

2.1.3 The Detector 28

2.2 Computed Tomography 29

2.3 Digital Tomosynthesis 32

3 The Mathematics of Digital Tomosynthesis **35**

3.1 Modeling the Source-Detector Interaction 36

3.2 Beer's Law 37

3.3 The Radon Transform . . . 40
3.4 Standard Reconstruction Methods . . . 42
3.4.1 Backprojection . . . 42
3.4.2 Filtered Backprojection . . . 44
3.4.3 Maximum Likelihood Estimation Maximization Method (MLEM) . . . 46
3.4.4 Algebraic Reconstruction Techniques (ART) . . . 48
3.5 Beam Hardening . . . 49

4 The Linear Polyenergetic Model for Attenuation 52
4.1 Building the Forward Model . . . 53
4.2 Statistical Considerations . . . 55
4.3 The Iterative Reconstruction Framework . . . 56
4.3.1 Gradient Descent . . . 57
4.3.2 Newton Approach . . . 58
4.4 Summary . . . 61

5 The Quadratic Polyenergetic Model for Attenuation 62
5.1 Building the Forward Model . . . 63
5.1.1 Finding a Quadratic Fit . . . 63
5.1.2 Finding an Improved Quadratic Fit . . . 65
5.1.3 The Forward Projection Model . . . 69
5.2 Statistical Considerations . . . 70
5.3 The Iterative Reconstruction Framework . . . 71
5.3.1 Gradient Descent . . . 72
5.3.2 Newton Approach . . . 74
5.4 Summary . . . 79

6 The Polyenergetic Model for the Attenuation of Multiple Materials 80
6.1 Building the Forward Model . . . 81
6.2 Statistical Considerations . . . 84
6.3 The Iterative Reconstruction Framework . . . 86

6.3.1 Gradient Descent . . . 87
6.3.2 Newton Approach . . . 89
6.4 Summary . . . 93

7 A Simplified Polyenergetic Multimaterial Model 94
7.1 Building the Forward Model . . . 95
7.2 Statistical Considerations . . . 98
7.3 The Iterative Reconstruction Framework . . . 99
7.3.1 Gradient Descent . . . 100
7.3.2 Newton Approach . . . 103
7.4 Summary . . . 106

8 Implementation Considerations 107
8.1 Computational Intensity . . . 107
8.2 GPU Computing . . . 110
8.3 Programming using OpenCL . . . 111

9 Trace Functions 114
9.1 Siddon's Algorithm for the Exact Radiological Path . . . 114
9.2 A Task Parallel Version of Siddon's Raytrace . . . 121
9.3 Raytracing in Digital Tomosynthesis . . . 124
9.3.1 Implementation Considerations for Siddon's Algorithm . . . 125
9.3.2 Raytracing for Digital Tomosynthesis on a GPU . . . 126
9.3.3 Our Modified Raytrace vs. a Direct Implementation of Siddon's Algorithm . . . 130
9.4 The Backprojection Operation . . . 133
9.5 Summary . . . 135

10 Computational Design 136
10.1 Serial Implementation . . . 137
10.2 Matrix Product Optimizations . . . 139
10.3 Kernel Fusion Implementation . . . 144

10.4 Summary 148

11 Numerical Results **149**

11.1 Homogeneous Phantoms 151

11.1.1 Homogeneous Phantom with Glandular Tissue Inserts 151

11.1.2 Homogeneous Phantom with Micro-calcification Inserts 156

11.2 Heterogeneous Phantoms 160

11.2.1 Heterogeneous Phantom Glandular and Adipose Tissue Swirl 160

11.2.2 Heterogeneous Phantom with Glandular Tissue Inserts 162

11.2.3 Heterogeneous Phantom with Micro-calcification Inserts 166

11.3 Multimaterial Model Results 169

11.4 Summary 172

12 Conclusion **173**

Bibliography **176**

List of Figures

1.1 Tomosynthesis imaging device 3

2.1 X-ray emission at the atomic level 25
2.2 The Bremsstrahlung radiation process 26
2.3 A Bremsstrahlung curve 27
2.4 Linear attenuation coefficients 28
2.5 Mass attenuation coefficients 28
2.6 Limitations of a single angle radiograph 29
2.7 Distinction of objects by incident angles 30
2.8 Rotation of x-ray source 30
2.9 Geometry of a CT scanner 31
2.10 Digital Tomosynthesis imaging system geometry 33
2.11 Geometry of a reconstructed volume 34

3.1 Beer's law in one dimension 38
3.2 Radon transform of Shepp-Logan phantom 41
3.3 Example of a simple backprojecion 43
3.4 Backprojection of Shepp-Logan phantom 43
3.5 Filtered backprojection of Shepp-Logan phantom 45
3.6 Beam hardening in terms of energy intensity 50
3.7 Examples of beam hardening artifacts 51

8.1 Architecture design of a CPU vs a GPU 110

9.1 Voxels as the intersection of orthogonal planes 115
9.2 Possible intersections of the ray with the CT array 116
9.3 Potential ray intersections 123
9.4 Raytracing in digital tomosynthesis 126
9.5 Unfiltered raytrace vs. s direct implementation of Siddon's algorithm . . . 131
9.6 Filtered raytrace vs. a direct implementation of Siddon's algorithm in 2D 132
9.7 Filtered raytrace vs. a direct implementation of Siddon's algorithm in 3D 132

10.1 Gradient descent iteration serial scheme 139
10.2 Matrix vector product as a raytrace 140
10.3 3D Execution topology for raytrace 141
10.4 Matrix optimization algorithm scheme 145
10.5 Fused kernel algorithm scheme 147

11.1 Relative function value for homogeneous-glands phantom 152
11.2 SDNR for homogeneous-glands phantom 152
11.3 Features of interest in reconstruction of homogeneous-glands phantom . . 153
11.4 Profile for homogeneous-glands lesion 153
11.5 Homogeneous phantom with glandular tissue 155
11.6 Relative function value for homogeneous-calcs phantom 156
11.7 Features of interest in reconstruction of homogeneous-calcs phantom . . . 157
11.8 Comparison of ROI for homogeneous-calcs phantom 157
11.9 Boundary artifacts in reconstruction of homogeneous-calcs phantom . . . 158
11.10 Homogeneous phantom with micro-calcification inserts 159
11.11 Results for the heterogeneous swirl phantom 161
11.12 Relative function value for heterogeneous-swirl phantom 162
11.13 Spectral distribution function at entrance and exit of phantom 163
11.14 Relative function value for heterogeneous-glands phantom 163
11.15 Heterogenous phantom with glandular tissue 165
11.16 Features of interest in reconstruction of heterogeneous-glands phantom . . 166
11.17 Relative function value for heterogeneous-calcs phantom 167

11.18 Features of interest in reconstruction of heterogeneous-calcs phantom . . . 167
11.19 Heterogenous phantom with glandular tissue 168
11.20 Multi-material reconstruction of homogeneous-glandss phantom 170
11.21 Profile for homogeneous-glands lesion 170
11.22 Multi-material reconstruction of homogeneous-calcs phantom 171
11.23 Multi-material reconstruction of homogeneous-calcs phantom 171

List of Tables

10.1 Timings for matrix product operation 143
10.2 Timings for the optimized matrix product kernel call 144

11.1 Reconstruction time for homogeneous-glands phantom 154
11.2 Reconstruction time for homogeneous-calcs phantom 160
11.3 Reconstruction time for swirl phantom 162
11.4 Reconstruction time for heterogeneous-glands phantom 164
11.5 Reconstruction time for heterogeneous-calcs phantom 169

Variable Definitions

j	voxel index
N_v	number of voxels in the reconstructed volume
i	pixel index
N_p	number of pixels in the detector
e	index of the discrete value of energy
N_e	number of discrete energy levels
θ	projection index, represents incident angle
N_θ	number of projections acquired
N_m	number of attenuating materials to be modeled
$\mathbf{b}_\theta^i$	measured value for pixel i in the detector and projection angle θ
$\boldsymbol{\eta}^i$	the additional noise for pixel i
$\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i$	expected value of pixel i for projection angle θ
$\boldsymbol{\mu}_{j,e}$	*linear* attenuation coefficient for the composite materials present in voxel j attenuating at discrete energy level e
$\left(\frac{\mu}{\rho}\right)_{j,e}$	*mass* attenuation coefficient for the composite materials present in voxel j attenuating at energy level e
$\boldsymbol{\rho}_j$	density of the composite materials in voxel j
$u_{e,m}$	*linear* attenuation coefficient for material m at energy level e
$\left(\frac{u}{r}\right)_{e,m}$	*mass* attenuation coefficient for material m at energy level e
r_m	density of material m
$\mathbf{w}_{j,m}$	weight fraction (or percentage) of the m material in voxel j
s_e	energy fluence, defined as the product of x-ray energy with the number of incident photons at that energy
a_θ^{ij}	the length of the x-ray beam that passes through voxel j, incident onto pixel i starting from source angle θ
A_θ	raytrace matrix for incident projection angle θ

Chapter 1

Introduction

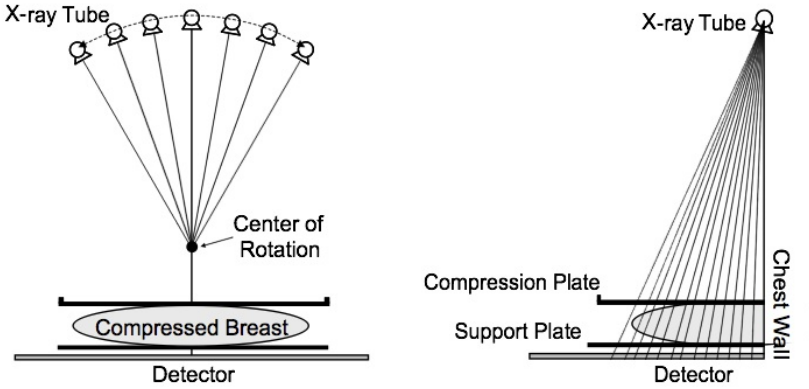
1.1 Background

In the United States breast cancer is the second leading cause of cancer death in women, with 1 in 8 expected to develop invasive breast cancer during their lifetime and 1 in 36 estimated to die because of the disease according to the American Cancer Society [4] . In 2012 alone, the American Cancer Society expects over 200,000 new cases of invasive breast cancer and over 63,000 new cases of carcinoma in situ (CIS is a non-invasive and earliest form of breast cancer) to be diagnosed in women. For this same year, the American Cancer Society estimates there will be around 40,000 breast cancer related deaths. Although the incidence rate for female breast cancer has been stable for the last decade and substantial progress has been made in identifying potential genetic, environmental, and behavioral risk factors, most physicians agree that early detection saves thousands of lives each year [4]. This is confirmed by the fact that chances of survival significantly decrease as the cancer reaches higher stages: the 5-year survival rate for breast cancer detected at stage I is 88% whereas the 5-year survival rate for cancer detected at stage IV is 15%.

In the later part of the 20th century, the introduction of screening mammography has been shown to be responsible for a reduction in breast cancer mortality [69]. However, even with the development of digital mammography to overcome the lower image quality, clinical efficacy, and higher radiation dosage of screen-film mammography, the standard planar mammography techniques have some limitations. Detection rates for mammography,

measured as sensitivity or the proportion of breast cancer detected when breast cancer is present, and specificity or the likelihood of the test being normal in the absence of cancer, vary widely with the population. In 2002 it was reported by Kolb et al [80] that breast density is the most significant predictor of mammographic sensitivity, ranging from 48% in women with very dense breast to 97% in women with fatty breasts [80]. Specificity was reported in the same study to range between 82% to 98%, but it was estimated in [42] that about 25% of women that receive screening mammography over a ten year period will receive at least one false positive result. The issues of breast density and tissue superposition are some of the factors leading to these false positive results in mammography screening.

Tomosynthesis imaging is a technique of relevant interest in breast cancer screening since it does away with the problem of tissue superposition which reduces clinical performance in standard mammography. In digital tomosynthesis imaging, multiple projections of an object are obtained along a small range of incident angles in order to reconstruct a pseudo 3D representation of the object. This is done by rotating the x-ray tube along an arch above the compressed breast as images of the breast are acquired at different angles (see Figure 1.1). The detector can be static or rotate along the same range of angles to remain perpendicular to the x-ray tube. The set of projections acquired by the imaging system is then used to reconstruct a pseudo 3D representation of the breast.



(a) Front view (b) Side view with X-ray tube at 0°

Figure 1.1: Example of the imaging system for breast tomosynthesis imaging

The challenge of this technique is that the reconstruction process is computationally and memory intensive, as it deals with millions of input pixels from the stack of acquired

projection images in order to produce a reconstructed pseudo 3D volume composed of billions of voxels. Standard approaches to solve this large scale inverse problem have relied in simplifying the physics of the image acquisition model by considering the x-ray beam to be monoenergetic (filtered backprojection, MLEM), thus decreasing the number of degrees of freedom and the computational complexity of the solution. However, this approach has been shown to introduce what is know as beam hardening artifacts, which are cupping-like shadows that surround the lesions of interest in the reconstructed volume. Beam hardening occurs when there is preferential absorption of low-energy photons from the x-ray by the object, thus changing the average energy of the x-ray beam. It is known that beam hardening is directly related to the assumption of a monoenergetic x-ray beam in the image acquisition process, and research has been done to develop post-processing imaging techniques that minimize the appearance of the artifacts [19]. As will be shown in this work, modeling the tomosynthesis imaging problem using a polyenergetic x-ray beam produces significantly better images and more meaningful reconstructions without the need to post-process the results. In order to develop an efficient reconstruction scheme for the digital tomosynthesis problem for breast imaging we need to dive into three different areas: the biomedical engineering of the image acquisition process, the numerical aspects and solution to the mathematical model, and the computational science required for a robust and fast implementation.

This thesis presents an interdisciplinary collaboration to overcome the mathematical, computational, and physical constraints of standard reconstruction methods in digital tomosynthesis imaging for breast cancer screening. We begin by describing the mathematics and developing a physically accurate polyenergetic forward model for the image acquisition process as well as propose a stable numerical framework to iteratively solve the nonlinear ill-posed inverse problem arising from this model. We exploit the multi-threaded parallelism available on the GPU architecture using OpenCL as the API to provide an efficient and fast implementation of the volume reconstruction framework that can reconstruct a full size clinical data set in under five minutes. We then show that our reconstructed volume has no beam hardening artifacts and better image quality than standard reconstruction methods. Our reconstructions also provide a quantitative measure for each voxel of the pseudo 3D

volume, allowing the physician to see and measure the contrast between materials present inside the breast. The research presented in this thesis shows that large scale, physically accurate medical image reconstructions can be done using smaller and less costly machines by effectively harnessing the multi-threading power of GPUs. This allows smaller hospitals and clinics that lack access to expensive clusters and supercomputers, to use off-the-shelf computing systems to run the reconstructions needed for diagnostic radiology.

1.2 Inverse Problems and Regularization

Inverse problems arise in a variety of imaging applications where we seek to reconstruct an image from indirect or noise polluted measurements obtained from an object. A simple example is image deblurring, where we want to remove the noise and blur from an image in order to obtain a representation of the object with higher resolution. The noise or blur in the image can come from different factors like the imaging system or the environment, and even with the use of precise imaging equipment in a controlled setting, it may not be possible to avoid a corrupted image. However, having some information about the blur, statistical properties of the noise, or an accurate mathematical representation of the image acquisition process can help produce very accurate reconstructions. The digital tomosynthesis reconstruction problem for breast cancer screening can be formulated as a large scale nonlinear ill-posed inverse problem. This section presents an introduction to the mathematics of ill-posed inverse problems in order to provide a background for the discussion of the reconstruction methods proposed in this thesis.

The general representation of an inverse problem is given by

$$\mathbf{b} = \mathbf{K}(\mathbf{x}_{\text{exact}}) + \boldsymbol{\eta} \tag{1.1}$$

where $\mathbf{K}(\cdot)$ is a function that models the image acquisition process, $\mathbf{x}_{\text{exact}}$ represents the true image of the object, the quantity $\boldsymbol{\eta}$ contains the noise and error that can be present in the data, and $\mathbf{b}$ is the observed data. The structure and properties of the operator $\mathbf{K}(\cdot)$ depend on the application. A major challenge that arises in solving inverse problem is that

often the problems are ill-posed. The definition proposed by Hadamard [59] of a well-posed problem is one that satisfies:

1. the solution is unique,
2. the solution exists for arbitrary data, and
3. the solution depends continuously on the data.

In their continuous mathematical form ill-posed inverse problems fail to satisfy at least one of the above criterion which makes finding an exact solution a difficult or even impossible task. Once the continuous problem is discretized in order to find a numerical solution, the ill-posedness of the problem often continues to be an issue in the form of ill-conditioning. For the tomosynthesis reconstruction problem, the ill-posedness of the continuous problem creates issues with the stability of approximate solutions since small changes in the observed data can cause large changes in the approximated solution. Regularization is needed in this case to deal with this instability.

1.2.1 Linear Inverse Problems

The simplest type of inverse problem is a linear inverse problem where the function $\mathbf{K}(\cdot)$ can be expressed as $\mathbf{K}(\mathbf{x}) = A\mathbf{x}$ where A is a known linear operator, or in the discrete setting, an $m \times n$ matrix. In this case the formulation of the inverse problem is

$$\mathbf{b} = A\mathbf{x}_{\text{exact}} + \boldsymbol{\eta} \tag{1.2}$$

and the goal is to find an approximation of $\mathbf{x}_{\text{exact}}$ given both $\mathbf{b}$ and A. In order to investigate this problem, consider the singular value decomposition (SVD) for the $m \times n$ matrix A

$$A = U\Sigma V^T$$

where U is an $m \times m$ orthogonal matrix (that is, $U^TU = I$, $UU^T = I$), V is an $n \times n$ orthogonal matrix, and Σ is an $m \times n$ diagonal matrix containing the nonnegative singular values $\sigma_1 \geq \sigma_2 \geq ... \geq 0$. If the matrix A is nonsingular, the inverse solution for the problem

is given by

$$\mathbf{x}_{\text{inv}} = A^{-1}\mathbf{b} = \sum_{i=1}^{n} \frac{\mathbf{u}_i^T \mathbf{b}_{\text{exact}}}{\sigma_i}\mathbf{v}_i + \sum_{i=1}^{n} \frac{\mathbf{u}_i^T \boldsymbol{\eta}}{\sigma_i}\mathbf{v}_i = \mathbf{x}_{\text{exact}} + \text{error} \tag{1.3}$$

where $\mathbf{u}_i, \mathbf{v}_i$ are the singular vectors of A (the columns of matrices of U and V respectively). In the case of matrices arising from ill-posed inverse problems, we have the following properties [31]:

1. The matrix A is severely ill-conditioned and the singular values σ_i decay steadily to zero without a significant gap that indicates numerical rank.

2. The singular vectors corresponding to small singular values have a higher frequency than the singular vectors corresponding to large singular values.

3. The components $\left|\mathbf{u}_i^T \mathbf{b}_{\text{exact}}\right|$ decay on average faster than the singular values σ_i. This is known as the *discrete Picard condition* [62].

These properties show that for ill-posed inverse problems the solution $\mathbf{x}_{\text{inv}}$ in Equation (1.3) will be dominated by the terms in the summation that correspond to small singular values since the division by σ_i will magnify the oscillations of the corresponding singular vectors $\mathbf{u}_i, \mathbf{v}_i$. We can try to limit the contribution of these dominating terms by filtering out or excluding these terms in the solution, using a technique that is called regularization by SVD filtering.

The idea in regularization by SVD filtering is to introduce what are called *filter factors* ϕ_i to the solution $\mathbf{x}_{\text{inv}}$ where $\phi_i \approx 1$ for large values of σ_i and $\phi_i \approx 0$ for small σ_i. The resulting filtered solution is of the form

$$\mathbf{x}_{\text{filt}} = \sum_{i=1}^{n} \phi_i \frac{\mathbf{u}_i^T \mathbf{b}}{\sigma_i}\mathbf{v}_i. \tag{1.4}$$

There are different choices for the factors ϕ_i [64, 62] including

- **Truncated SVD Filtering:** where given an appropriate tolerance level τ, the factors are given by $\phi_i = \begin{cases} 1 & \text{if } \sigma_i > \tau \\ 0 & \text{if } \sigma_i \leq \tau \end{cases}$.

- **Tikhonov Filtering:** the filter is given by $\phi_i = \frac{\sigma_i^2}{\sigma_i^2+\alpha^2}$ and we need an appropriate choice of α.

- **Exponential Filtering:** where the factors are given by $\phi_i = 1 - e^{-\frac{\sigma_i^2}{\alpha^2}}$ and we again need to choose a value for the parameter α.

The choice of the regularization parameter for each filter τ or α depends on the properties of the problem and of the matrix A, and finding the appropriate value can sometimes be a difficult task on its own.

Another class of regularization methods for linear inverse problems is known as variational regularization and has the form

$$\min_{\mathbf{x}} \left\{ ||\mathbf{b} - A\mathbf{x}||_2^2 + \alpha^2 \mathcal{J}(\mathbf{x}) \right\} \tag{1.5}$$

where the regularization parameter α and the regularization operator $\mathcal{J}$ are chosen depending on the problem. Variational regularization methods are flexible (for example, the least squares criterion can be exchanged with the Poisson log likelihood function [7, 8, 6]) and have the attractive property that constraints like nonnegativity in the solution can be added. Some choices for the regularization operator $\mathcal{J}$ [31] include Tikhonov [86, 99, 111, 112, 113], total variation [27, 101, 116], and sparsity constraints [26, 49, 115], where

- **Tikhonov:** the regularization operator is given by $\mathcal{J}(\mathbf{x}) = ||L\mathbf{x}||_2^2$ where L is typically chosen to be the identity matrix or some discrete approximation to a derivative operator like the Laplacian.

- **Total Variation:** the regularization operator is given by

$$\mathcal{J}(\mathbf{x}) = \left|\left| \sqrt{(D_h\mathbf{x})^2 + (D_v\mathbf{x})^2} \right|\right|_1$$

  where the matrices D_h and D_v represent the horizontal and vertical derivatives of the 2D image $\mathbf{x}$ (the method extends to 3D images without loss of generality). Implementing total variation regularization can be nontrivial for some problems [116, 27].

- **Sparsity:** for sparse reconstructions the regularization operator is given by

$$\mathcal{J}(\mathbf{x}) = ||\Phi\mathbf{x}||_1$$

  where the matrix Φ represents the basis in which the image is sparse and depends completely on the structure of the desired solution.

Variational regularization approaches can be costly and nontrivial to implement for some problems. Sometimes, a simpler and effective approach is to use iterative regularization, in which an iterative method is applied to the unregularized form of the problem

$$\min_{\mathbf{x}} \left\{ ||\mathbf{b} - A\mathbf{x}||_2^2 \right\} \tag{1.6}$$

until a "good" solution has been reached. If the inverse problem satisfies the *Picard condition*, many iterative methods will exhibit "semi-convergence" behavior, in which the early iterates are very good approximations of the solution and later iterates begin to be corrupted by errors in the data. This is because the early iterations will reconstruct the components of the solution that correspond to the larger singular values whereas the later iterations will reconstruct those components corresponding to smaller singular values. Thus, if we terminate the iteration process at the point where the solution begins to degrade because of the small singular value components, we can still achieve a good approximation to the exact solution. Iterative methods of this type include Landweber [81], steepest descent, conjugate gradient type methods (like LSQR [96, 97], GMRES, and MINRES), and statistics based methods like expectation-maximization [43, 63, 116]. The advantage of this type of regularization is that some of these iterative methods are very easy to implement, but as is the case with regularization parameters, finding a good iterate to stop the iteration process can be difficult.

A more sophisticated approach to regularization is called hybrid iterative-direct regularization, which as the name suggests, combines iterative regularization methods with variational approaches. In this hybrid approach an iterative Golub-Kahan (or Lanczos)

based method like LSQR is applied to the minimization problem

$$\min_{\mathbf{x}} \left\{ ||\mathbf{b} - A\mathbf{x}||_2^2 \right\} \tag{1.7}$$

and variational regularization is applied within the iterative process. The scheme works by projecting the full problem onto a Krylov subspace of smaller dimension (by using bidiagonalization) and applying variational regularization to the projected problem at each iteration. For more details on this technique see [15, 16, 61, 76, 77, 82, 94].

Another common approach is regularization using statistically motivated Bayesian methods. In these types of methods, the unknown image $\mathbf{x}$ is regarded as a random variable and certain properties that are known about the solution (like the probability density function) are used to solve the problem. Examples of this approach are given in [24, 10, 13, 14, 22, 23, 25].

The choice of regularization method depends significantly on the problem and the computational resources available. Once the regularization method has been chosen, estimating the appropriate regularization parameter for the method is a nontrivial question that can be a time consuming and computationally intensive. The right choice of regularization parameter is critical to reach an approximate solution to the problem; a very small parameter will not sufficiently limit the contribution of high frequency components to the solution and a very large parameter will result in an over-smoothed approximation to the solution. There are several techniques that have been developed to choose an appropriate regularization parameter, like generalized cross-validation [53], the discrepancy principle [43], L-curve [62], and others [21, 43, 62, 116], but just like the choice of regularization method, a regularization parameter is not a one size fits all approach.

1.2.2 Separable Inverse Problems

A second type of inverse problem is known as separable inverse problem. A separable inverse problem is one in which the unknown vector $\mathbf{x}$ can be separated into two components $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$, where one relates to the linear operator A and the other to the exact solution vector. This means we can write the operator as $\mathbf{K}(\mathbf{x}) = \mathbf{K}(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}) = A\left(\mathbf{x}^{(1)}\right)\mathbf{x}^{(2)}$ with

A a linear operator that is defined by the parameters $\mathbf{x}^{(1)}$, and the inverse problem is written as

$$\mathbf{b} = A\left(\mathbf{x}^{(1)}_{\text{exact}}\right)\mathbf{x}^{(2)}_{\text{exact}} + \boldsymbol{\eta} \tag{1.8}$$

The idea is to find an approximation of $\mathbf{x}^{(1)}_{\text{exact}}$ and $\mathbf{x}^{(2)}_{\text{exact}}$ given $\mathbf{b}$ and the parametric representation of A. Consider adapting the regularization approaches of the previous section to this problem. If we use the variational form of the Tikhonov method, the regularized separable inverse problem becomes

$$\min_{\mathbf{x}} \left\{ ||\mathbf{b} - A\left(\mathbf{x}^{(1)}\right)\mathbf{x}^{(2)}||_2^2 + \alpha^2 ||\mathbf{x}^{(2)}||_2^2 \right\}$$

which is equivalent to the problem

$$\min_{\mathbf{x}^{(1)},\mathbf{x}^{(2)}} \left|\left| \begin{bmatrix} A\left(\mathbf{x}^{(1)}\right) \\ \alpha I \end{bmatrix} \mathbf{x}^{(2)} - \begin{bmatrix} \mathbf{b} \\ \mathbf{0} \end{bmatrix} \right|\right|_2^2 \tag{1.9}$$

Notice that the least squares problem in (1.9) is not linear by definition, but it is linear in terms of the value $\mathbf{x}^{(2)}$. Described below are three different approaches to solve this type of problem that take advantage of the separation of the two unknowns $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}$ in order to find a good approximate solution to Equation (1.8).

The first approach to solve the regularized nonlinear least squares problem is to iteratively solve the fully-coupled problem. We can write Equation (1.9) as

$$\min_{\mathbf{x}} \phi(\mathbf{x}) = \min_{\mathbf{x}} ||\rho(\mathbf{x})||_2^2 \tag{1.10}$$

where $\rho(\mathbf{x}) = \rho\left(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}\right) = \begin{bmatrix} A\left(\mathbf{x}^{(1)}\right) \\ \alpha I \end{bmatrix} \mathbf{x}^{(2)} - \begin{bmatrix} \mathbf{b} \\ \mathbf{0} \end{bmatrix}$ and $\mathbf{x} = \begin{bmatrix} \mathbf{x}^{(1)} \\ \mathbf{x}^{(2)} \end{bmatrix}$ and solve the numerical optimization problem using an iterative algorithm. In general, iterative algorithms to solve this problem have the form

Algorithm 1.2.1 General Iterative Algorithm for Numerical Optimization

1: Choose the initial iterate $\mathbf{x}_0 = \begin{bmatrix} \mathbf{x}_0^{(1)} \\ \mathbf{x}_0^{(2)} \end{bmatrix}$
2: **for** $k = 1, 2, ...$ until converge **do**
3: choose a step direction $\mathbf{d}_k$
4: determine the step length τ_k
5: update the solution $\mathbf{x}_{k+1} = \mathbf{x}_k + \tau_k \mathbf{d}_k$
6: **end for**

There are different choices for the step direction $\mathbf{d}_k$ but some common ones that will be used in this work are the gradient descent direction and the Newton direction. The gradient descent direction is given by the first derivative of the cost function $\phi(\cdot)$, that is

$$\mathbf{d}_k = -\phi'(\mathbf{x}_k) = -\mathbf{J}_\phi^T \rho$$

where $\mathbf{J}_\phi$ is the Jacobian matrix given by

$$\mathbf{J}_\phi = \left[\frac{\partial \rho\left(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}\right)}{\partial \mathbf{x}^{(1)}} \quad \frac{\partial \rho\left(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}\right)}{\partial \mathbf{x}^{(2)}} \right].$$

The Newton direction is given by

$$\mathbf{d}_k = -\left(\hat{\phi}''(\mathbf{x}_k)\right)^{-1} \phi'(\mathbf{x}_k)$$

where $\hat{\phi}''$ is an approximation of the second derivative ϕ'' and, as in the gradient descent method, $\phi'(\mathbf{x}_k) = \mathbf{J}_\phi^T \rho$. For the well known Gauss-Newton method, which is generally used to solve nonlinear least squares problems, the approximation to the second derivative of the cost function is given as $\hat{\phi}'' = \mathbf{J}_\phi^T \mathbf{J}_\phi$.

A simpler approach to solve the separable nonlinear problem in Equation (1.9) is to decouple the problem in term of the unknowns and use a block coordinate descent algorithm. The iterative algorithm to solve the decoupled problem takes the form

Algorithm 1.2.2 Block Coordinate Descent Algorithm

1: Choose the initial iterate $\mathbf{x}_0 = \mathbf{x}_0^{(1)}$

2: **for** $k = 1, 2, ...$ until converge **do**

3: choose the regularization α_k and solve the linear problem
$\mathbf{x}_k^{(2)} = \arg\min_{\mathbf{x}^{(2)}} \left|\left|A\left(\mathbf{x}_k^{(1)}\right)\mathbf{x}_k^{(2)} - \mathbf{b}\right|\right|_2^2 + \alpha_k^2 \left|\left|\mathbf{x}_k^{(2)}\right|\right|_2^2$

4: given the solution above, solve the nonlinear problem
$\mathbf{x}_k^{(1)} = \arg\min_{\mathbf{x}^{(1)}} \left|\left|A\left(\mathbf{x}_k^{(1)}\right)\mathbf{x}_k^{(2)} - \mathbf{b}\right|\right|_2^2 + \alpha_k^2 \left|\left|\mathbf{x}_k^{(2)}\right|\right|_2^2$

5: **end for**

The regularization parameter α_k can be chosen using any of the regularization approaches for the linear inverse problem mentioned in the previous subsection. The solution of the nonlinear least squares problem in step 4 can be more involved than the linear least squares problem in step 3, but if the number of unknowns $\mathbf{x}^{(1)}$ is significantly less that the number of unknowns in $\mathbf{x}^{(2)}$ a Gauss-Newton method will be sufficient to solve the problem. One major drawback of this approach is that it can be very slow to converge in problems with tightly coupled variables [90].

One last approach to solve the regularized separable least squares problem in Equation (1.9) is the variable projection method [55, 54, 74, 95, 102]. If the number of unknowns in $\mathbf{x}^{(1)}$ is significantly less that the number of unknowns in $\mathbf{x}^{(2)}$ variable projection can work better than the block coordinate descent method described above. For the variable projection method we take advantage of the problem being linear with respect to $\mathbf{x}^{(2)}$ and express the cost function in terms of only $\mathbf{x}^{(1)}$ by implicitly eliminating $\mathbf{x}^{(2)}$. A Gauss-Newton iterative method is then used to solve the nonlinear least square problem. Consider the function

$$\psi(\mathbf{x}^{(1)}) = \phi\left(\mathbf{x}^{(2)}(\mathbf{x}^{(1)}), \mathbf{x}^{(2)}\right)$$

where the value $\mathbf{x}^{(2)}(\mathbf{x}^{(1)})$ is a solution to the least squares problem

$$\min_{\mathbf{x}^{(2)}} \phi\left(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}\right) = \min_{\mathbf{x}^{(2)}} \left|\left| \begin{bmatrix} A\left(\mathbf{x}^{(1)}\right) \\ \alpha I \end{bmatrix} \mathbf{x}^{(2)} - \begin{bmatrix} \mathbf{b} \\ \mathbf{0} \end{bmatrix} \right|\right|_2^2 . \tag{1.11}$$

Then we use a Gauss-Newton method to solve the problem

$$\min_{\mathbf{x}^{(1)}} \psi(\mathbf{x}^{(1)}) \tag{1.12}$$

where the derivative $\psi'(\mathbf{x}^{(1)})$ is given by

$$\psi'(\mathbf{x}^{(1)}) = \frac{d\mathbf{x}^{(2)}}{d\mathbf{x}^{(1)}}\frac{\partial\phi}{\partial\mathbf{x}^{(2)}} + \frac{\partial\phi}{\partial\mathbf{x}^{(1)}}$$

but since $\mathbf{x}^{(2)}(\mathbf{x}^{(1)})$ is a solution to equating (1.11), we have that the derivative is

$$\psi'(\mathbf{x}^{(1)}) = \frac{\partial\phi}{\partial\mathbf{x}^{(1)}} = \mathbf{J}_\psi^T\rho$$

where $\mathbf{J}_\psi$ is the Jacobian of the reduced cost function given by

$$\mathbf{J}_\psi = \frac{\partial}{\partial\mathbf{x}^{(1)}}\left[A(\mathbf{x}^{(1)})\mathbf{x}^{(2)}\right].$$

The algorithm is given as

Algorithm 1.2.3 Variable Projection gauss-Newton Algorithm

1: Choose the initial iterate $\mathbf{x}_0 = \mathbf{x}_0^{(1)}$

2: **for** $k = 1, 2, ...$ until converge **do**

3: choose the regularization α_k

4: solve $\mathbf{x}_k^{(2)} = \arg\min_{\mathbf{x}^{(2)}} \left|\left| \begin{bmatrix} A\left(\mathbf{x}_k^{(1)}\right) \\ \alpha_k I \end{bmatrix} \mathbf{x}^{(2)} - \begin{bmatrix} \mathbf{b} \\ \mathbf{0} \end{bmatrix} \right|\right|_2$

5: compute the residual $\mathbf{r}_k = \mathbf{b} - A\left(\mathbf{x}_k^{(1)}\right)\mathbf{x}_k^{(2)}$

6: compute the step direction $\mathbf{d}_k = \arg\min_{\mathbf{d}} ||\mathbf{J}_\psi\mathbf{d} - \mathbf{r}_k||_2$

7: find the step length τ_k

8: update the solution $\mathbf{x}_{k+1} = \mathbf{x}_k + \tau_k\mathbf{d}_k$

9: **end for**

Note that the advantage of this method is that since the cost function has been reduced

from $\phi(\mathbf{x}^{(1)}, \mathbf{x}^{(2)})$ to $\psi(\mathbf{x}^{(2)})$ the Jacobian $\mathbf{J}_\psi$ is significantly smaller than $\mathbf{J}_\phi$. Also, now there is only one convergence rate for the solution $\mathbf{x}^{(2)}$, as opposed to the two convergence rates that we need to examine in the block coordinate descent approach (one for step 3 and one for step 4 of Algorithm 1.2.2). The regularization methods described for the linear inverse problem in the previous section can be used in this algorithm as well.

1.2.3 Nonlinear Inverse Problems

Perhaps the most challenging type of inverse problem to solve is called a nonlinear inverse problem, which is one that cannot be classified as either linear or separable. This type of problem coupled with ill-posedness makes finding an approximation to the exact solution a very difficult task. In fact, the formulation and solution of the digital tomosynthesis reconstruction problem as an ill-posed nonlinear inverse problem is the full scope of this thesis. Using the SVD in order to study the ill-posedness of the problem as it was done for the linear case is not an option since there is no robust theoretical tool developed to do so, and even studying the linearization of a nonlinear inverse problem does guarantee a correct diagnosis of the degree of ill-posednes [45]. Furthermore, it is very difficult to establish the convergence properties for some nonlinear inverse problems, and the assumptions necessary to perform this kind of convergence analysis can sometimes be unrealistic [43, 44]. However, there exist general techniques to solve this type of inverse problem and some regularization methods can be extended to fit this case.

In its general form, a nonlinear inverse problem is expressed as

$$\mathbf{b} = \mathbf{K}(\mathbf{x}_{\text{exact}}) + \boldsymbol{\eta}$$

where the idea is to compute an approximation of $\mathbf{x}_{\text{exact}}$ given $\mathbf{b}$ and the function $\mathbf{K}(\cdot)$. In some physical applications, like digital tomosynthesis imaging, the function $\mathbf{K}(\cdot)$ (also called the *forward model*) cannot be known exactly, and a great amount of work needs to be done in order to accurately estimate it before attempting to approximate $\mathbf{x}_{\text{exact}}$. There are many approaches to solve this problem [5, 43, 44, 47, 67, 114, 116], we describe a few approaches in this work.

First, consider the nonlinear inverse problem of the form

$$\mathbf{b} = \mathbf{K}(\mathbf{x}) \tag{1.13}$$

which can also be formulated as

$$\mathbf{K}(\mathbf{x}) - \mathbf{b} = 0 \tag{1.14}$$

where we are now looking to find a zero of Equation (1.14). Now we can apply a Newton-type method to (1.14), updating the iterates using

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \tau_k \mathbf{d}_k$$

with step direction given by the solution of the linear system

$$\mathbf{J}(\mathbf{x}_k)\mathbf{d} = \mathbf{b} - \mathbf{K}(\mathbf{x}_k)$$

where $\mathbf{J}(\mathbf{x}_k)$ is the Jacobian matrix at iterate $\mathbf{x}_k$. Two drawbacks of this approach are that there is no guarantee for the existence and uniqueness of the solution because it depends on the properties of the matrix $\mathbf{J}(\mathbf{x}_k)$, and finding the step length τ_k can be nontrivial.

Another approach to solve a nonlinear problem of the form (1.13) is to solve the problem using an iterative nonlinear optimization algorithm. The nonlinear Landweber iteration takes the form

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{J}(\mathbf{x}_k)^T(\mathbf{b} - \mathbf{K}(\mathbf{x}_k))$$

which is the standard Landweber iteration in the case of a linear operator $\mathbf{K}(\cdot)$. Another choice is to use a gradient descent type method or a Newton-type method which have iterates of the form

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \tau_k \mathbf{d}_k$$

where $\mathbf{d}_k = -\nabla\mathbf{K}(\mathbf{x}_k)$ for gradient descent and $\mathbf{d}_k$ solves the linear system given by $\mathbf{H}(\mathbf{x}_k)\mathbf{d} = -\nabla\mathbf{K}(\mathbf{x}_k)$ for the Newton method. Here we denote $\nabla\mathbf{K}(\mathbf{x}_k)$ as the first derivative of the function $\mathbf{K}$ and $\mathbf{H}(\mathbf{x}_k)$ as the Hessian evaluated at $\mathbf{x}_k$. If the analytical form of

the Hessian $\mathbf{H}(\mathbf{x}_k)$ is not available, is difficult to compute, or requires significant storage, methods like Gauss-Newton, LBFGS and nonlinear Conjugate Gradient which estimate $\mathbf{H}(\mathbf{x}_k)$ can be used.

For nonlinear ill-posed inverse problems, the iterative approach above can be combined with regularization to provide meaningful solutions. In the case of coupling an iterative solver with variational regularization approaches, the goal is to solve a newly formulated problem

$$\min_{\mathbf{x}} \left\{ ||\mathbf{b} - \mathbf{K}(\mathbf{x})||_2^2 + \alpha^2 \mathcal{J}(\mathbf{x}) \right\} \tag{1.15}$$

given a regularization operator $\mathcal{J}$ and a corresponding regularization parameter α. Solving this problem provides a good amount of flexibility in the choice of $\mathcal{J}$ and α, but selecting the proper regularization method is difficult and estimating α can be computationally intensive. A simple and effective regularization approach for ill-posed nonlinear inverse problems is early termination, where the iteration process is terminated when the solution has reached certain criteria but before the problem has fully converged to the true inverse solution. Selecting the criteria to stop the iteration process is a problem on its own, but it is certainly less computationally expensive than solving the full problem in Equation (1.15).

Ultimately, the framework to solve a nonlinear ill-posed inverse problem depends on a strong choice of the problem's three components: an accurate estimate of the function $\mathbf{K}(\cdot)$ (the forward model), a numerically stable nonlinear optimization iterative solver, and a robust form of regularization. The work discussed in this thesis will describe our proposed choice for each of these three components in the digital tomosynthesis reconstruction problem.

1.3 Outline of Work

This thesis describes the three different areas of research we explored to solve the polyenergetic digital tomosynthesis reconstruction problem for breast cancer imaging, starting with the medical physics of the problem, continuing with the mathematical modeling, and concluding with the computational science needed for the implementation. We begin by describing the physics of the problem in Chapter 2, where we introduce the basics of x-ray

imaging and discuss the details of the image acquisition process in computed tomography and digital tomosythesis. This chapter highlights the need for a polyenergetic forward model as we show it is physically inaccurate to assume a monochromatic x-ray for the different projections acquired in tomography. Chapter 3 serves as an introduction to the mathematical models proposed for the digital tomosynthesis problem in this thesis, which are described in the subsequent Chapters 4-7. In Chapter 3 we provide a background for our statistical approach in formulating the digital tomosythesis forward model, describe Beer's law, and discuss the details and shortcomings of standard approaches used to solve this problem (filtered backprojection, MLEM, ARTs).

Chapter 4-7 present the mathematical contributions of this thesis, describing four different forward models for the digital tomosynthesis problem and our proposed reconstruction framework. Chapter 4 describes the linear model for attenuation which decomposes the attenuation function of the volume as a linear function of the glandular fractions in each voxel, assuming the presence of only adipose and glandular tissue inside the breast. Chapter 5 describes the quadratic model for attenuation, which expands on the linear model by finding a quadratic fit to account for the presence of air and micro-calcifications inside the breast. Chapter 6 proposes a linear polyenergetic multi-material attenuation model for the object by formulating the attenuation function in terms of the weight fractions of an arbitrary number of materials present inside the breast. This formulation allows us to capture additional materials of interest in breast cancer screening and provides the flexibility to extend our polyenergetic reconstruction framework to other objects or parts of the body. Chapter 7 describes a simplified multi-material model that is easier to implement and analyze than the model presented in Chapter 6, but still assumes the presence of adipose and glandular tissue as well as micro-calcifications inside the breast.

The computational science contributions of this thesis are described in Chapters 8-10. Chapter 8 describes the implementation considerations for the reconstruction framework, including computational complexity and the hardware used in our numerical results. Chapter 9 presents the kernels we use to perform our raytracing and backprojection operation in the reconstruction process. We show how the sorting operation and memory requirements of Siddon's algorithm are modified to run on a GPU. Chapter 10 describes our muti-threaded

implementation and the optimization techniques used to provide an efficient and fast reconstruction.

To conclude, Chapter 11 presents some numerical results to show the effectiveness and computation time of our polyenergetic digital tomosynthesis reconstruction frameworks on five real data sets taken of different phantom breast objects with known materials. Finally, Chapter 12 provides some concluding remarks.

1.4 Contributions

The contributions of this work are as follows:

- We propose four physically accurate polyenergetic forward models for the image acquisition process in digital tomosynthesis for breast cancer screening

    1. **The linear model for attenuation** takes into account the attenuation properties of adipose and glandular tissue, the two materials that compose the majority of the breast.
    2. **The quadratic model for attenuation** extends the linear model for attenuation to include the presence of air and micro-calcifications inside the imaged breast.
    3. **The linear model for multiple attenuating materials** which improves the previous two models by explicitly accounting for the presence of multiple arbitrary materials inside the imaged object. Accounting for each attenuating material independently allows for including more physical information about the composition of the imaged object and gives the physician a visual and quantitative representation of the type of tissue inside the breast. The flexibility of this model allows us to extend of polyenergetic digital tomosynthesis reconstruction framework to other objects or parts of the body.
    4. **The simplified linear model for multiple attenuating materials** is a simplified version of the linear model for multiple attenuating materials which assumes the density of all modeled material is approximately the same. This model

is less computationally intensive than the full model but still allows us to separate the multiple materials inside the breast.

- We develop a robust framework to solve some nonlinear inverse problems of the form

$$\mathbf{b} = \mathbf{K}(\mathbf{x})\boldsymbol{\rho} + \boldsymbol{\eta}$$

where $\mathbf{b}, \mathbf{x}, \boldsymbol{\rho}$ are vectors and $\mathbf{K}(\mathbf{x})$ is a large structured matrix defined by the set of parameters $\mathbf{x}$. This type of problem arises not only in digital tomosynthesis but also in whole-body computed tomography and other medical imaging applications.

- We provide an efficient implementation of the iterative reconstruction framework that takes advantage of the multi-threaded parallelism available in GPUs. The multi-threaded implementation is written in OpenCL making it scalable to larger problem sizes and portable to other architectures. The implementation is designed to increase throughput, minimize storage requirements, and maximize the utilization of hardware resources.

- We implement a very accurate and fast version of Siddon's algorithm to compute a raytrace in a GPU, allowing for large matrix vector multiplications to be done in seconds.

- The software we present in this thesis runs a full clinical size reconstruction in under five minutes, using a single core accelerated with a single GPU. The application framework can easily be extended to multiple GPUs, which may be needed in the case of several materials present inside the object or reconstructing a higher resolution volume like those needed in CT scans.

- The contribution of this work to medical imaging and diagnostic radiology is a fast reconstruction of the imaged breast based on a physically accurate polyenergetic x-ray model. The reconstructed volume is beam artifact free with each voxel containing a quantitative representation of the material present inside of it. A physically accurate model of the imaging system coupled with our efficient reconstruction framework

allows for a precise characterization of the imaged breast, giving the physician better information to make a correct clinical diagnosis of the patient.

Chapter 2

The Physics of X-ray Imaging and Digital Tomosynthesis

The medical advantages of x-ray imaging were discovered in the late 1800s when Röntgen successfully produced the first radiograph of his wife's left hand, beginning a revolution in non-invasive medicine and the diagnosis of fractures and lesions. Initially x-rays were considered to be a special form of light that had three important properties [46]:

1. As an x-ray travels through an object, the intensity of the ray will be decreased depending on the density of the material composition inside the object. This is called *attenuation*.

2. An x-ray beam travels on a straight path.

3. X-rays have an opaquing effect on photographic film, which was initially used to capture radiographs.

Although x-rays are useful in detecting contrast between very different attenuating mediums (like bone and tissue), they are incapable of detecting differences between materials of similar density (like adipose and glandular tissue). In addition, a single x-ray image has the inherent limitation of projecting a three dimensional object onto a two dimensional space, thus losing major volumetric information of the object. Technology advancements like modern computing and development of scintillation detectors allowed Allan Cormack

and Godfrey Hounsfield [89] to independently create a new imaging modality to overcome the issues posed in x-ray imaging. This imaging technique, called x-ray tomography (or slice imaging), collects multiple radiographic projections of the object at various incident angles along a 360 degree range to reconstruct a 3D representation.

Digital tomosynthesis is a form of computed tomography (CT) that collects x-ray projections of the object along a limited range of incident angles and reconstructs a *pseudo* 3D representation of the object. Thus, in order to study the mathematical formulation of the digital tomosynthesis reconstruction problem, it is important to understand the basic physics that contribute to the image acquisition process which includes x-ray imaging and tomography. In this chapter we cover the physical concepts of x-ray imaging, computed tomography, and digital tomosynthesis. We begin by describing the basic atomic properties of x-rays and the interaction of radiation with matter to produce radiographs. We also discuss the properties of computed tomography imaging to provide a background for the motivation of our reconstruction model in the coming chapters. Finally we describe the details of digital tomosynthesis, sketching the geometry of the imaging systems as well as the reconstructed output.

2.1 Producing a Radiograph

A single x-ray projection or radiograph, is produced by sending multiple x-ray beams of a known energy through an object and measuring their intensity after they exit the object. Originally, the exit intensity of the beam was detected by a sheet of photographic film since the x-rays that penetrate the object attenuate and thus have a less of an opaquing effect on the film than the x-rays not entering the object. The opaqued photographic film is the radiograph [46]. Today, the film has been replaced by a digital detector which is significantly more sensitive to the intensity of the exiting x-rays [20, 70, 28]. This section describes the physics involved in the three stages of producing a single x-ray projection: creating the x-ray beam, describing the interaction of an x-ray with matter, and measuring the exit intensity of the beam using a digital detector.

2.1.1 X-ray Production

An atom is the smallest particle that maintains the properties of an element. It is composed of positively charged particles called *protons*, negatively charged particles called *electrons*, and neutral charge particles called *neutrons*. Generally, atoms are electrically neutral because they have the same number of protons and electrons to balance positive and negative charges. Structurally, the protons and neutrons are bound together in the *nucleus* of the atom, while electrons constantly move along concentric spherical shells around the nucleus accelerated by the attractive coulomb forces. These electron shells are classified by their distance to the nucleus, the first (innermost) shell is called the *K shell*, the second shell is the *L shell*, the third shell is the *M shell* and so on. There is a maximum number of electrons that can occupy a given shell at all times. For example, the K shell can only hold up to two electrons, the L shell can hold up to eight electrons and the M shell can hold up to eighteen electrons. In reality this atomic structure can be much more complex as each shell can be further divided and there is a limit to the number of electrons in the outermost shell [20, 28], but for the purpose of the discussion at hand this high level overview suffices.

Each electron bound to an atom has a certain potential energy that determines the amount of energy needed to free the electron from its shell. This potential energy is inversely proportional to the distance of the electron from the nucleus, meaning that the electrons in the K shell of an atom are the most tightly bound whereas the outermost shell electrons require the least amount of energy to remove. The process of removing an electron from an atom is called *ionization* and the minimum amount of energy necessary to unbind an electron from an atom is called *ionization energy* which is measured in terms of *electron volts* (eV). An eV is the energy acquired by an electron accelerated through 1 V of potential difference.

An x-ray is electromagnetic radiation with energy of 100 eV or more that has the ability to ionize matter and penetrate substances. Electromagnetic radiation is energy emitted as the result of interactions between charged particles, and it can propagate in the form of a wave or a particle (called a *photon*). A photon is a pocket of energy and has no mass or charge. There are different ways to produce x-rays: by electron transitions at the atomic

level like in characteristic x-rays and Auger-electron emission, or by accelerating charged particles to create photons as in the Bremsstrahlung "braking radiation" process.

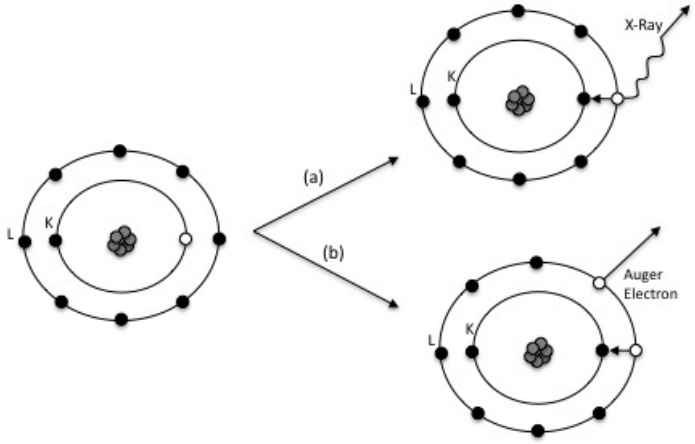


Figure 2.1: X-ray emission at the atomic level: (a) Characteristic x-rays created by an electron from a higher shell filling a vacancy in a lower shell. (b) Auger-electrons are created when an outer shell electron fills an inner shell vacancy and the energy produced by the transition ejects another electron from the atom.

Characteristic x-rays are emitted when an electron vacancy is created in an inner shell and an electron from an outer shell transitions to fill the hole as shown in Figure 2.1 [28]. This type of x-ray is characteristic to each element, since the energy emitted depends on the number of protons that make up the atom. An Auger-electron emission occurs similarly to a characteristic x-ray but results in a double ionization of the atom (see Figure 2.1). When there is an electron vacancy in an inner shell, an electron from an outer shell will transition to fill the hole, but instead of the balance of energy being emitted as a characteristic x-ray the energy is absorbed by another electron which is then ejected from the atom.

In diagnostic imaging, x-rays are produced as a result of the Bremsstrahlung process. During the Bremsstrahlung process, an electron is accelerated towards the nucleus of a target atom. While the electron is inside the positively charged electric field of the nucleus, it will experience a sudden deceleration or loss of kinetic energy which is instantaneously released in the form of radiation (see Figure 2.2 [20]). Hence the name Bremsstrahlung which is German for "braking radiation". Note that the energy that is emitted by a photon produced during the Bremsstrahlung process can have any value up to the total kinetic

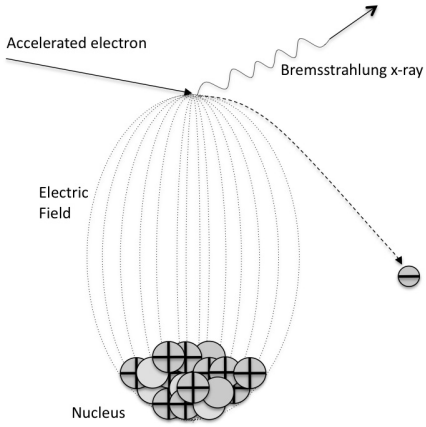


Figure 2.2: The Bremsstrahlung radiation process. An accelerated electron enters the positively charged electrical field of the nucleus and kinetic energy is transformed into an x-ray.

energy of the accelerated electron. Therefore, when x-rays are produced by accelerating multiple electrons inside an x-ray tube at the same time, the resulting x-ray beam will be a continuous spectrum of energies (a polyenergetic x-ray beam) as opposed to a single energy value (a monoenergetic x-ray beam). Figure 2.3 shows an example of the x-ray spectrum of energies used for breast imaging.

2.1.2 Interaction Between X-rays and Matter

As shown in the previous section, a beam of x-rays produced by the Bremsstrahlung process is made up of many photons that have different energy values which are expressed in terms of a continuous spectrum distribution. Once an x-ray enters an object, a series of interactions take place between the x-ray beam the matter composing the object that result in the process *attenuation*. Attenuation is the removal of select photons from the x-ray beam. The probability and type of interaction (absorption or scattering) that occurs to cause attenuation depends on the material inside the object and the energy of the photon. We measure the attenuation properties of a particular material at a given energy level by the *linear attenuation coefficient* and the *mass attenuation coefficient*.

The linear attenuation coefficient describes how easy the material is penetrated by a given energy, measuring the fraction of photons that are removed from a monochromatic

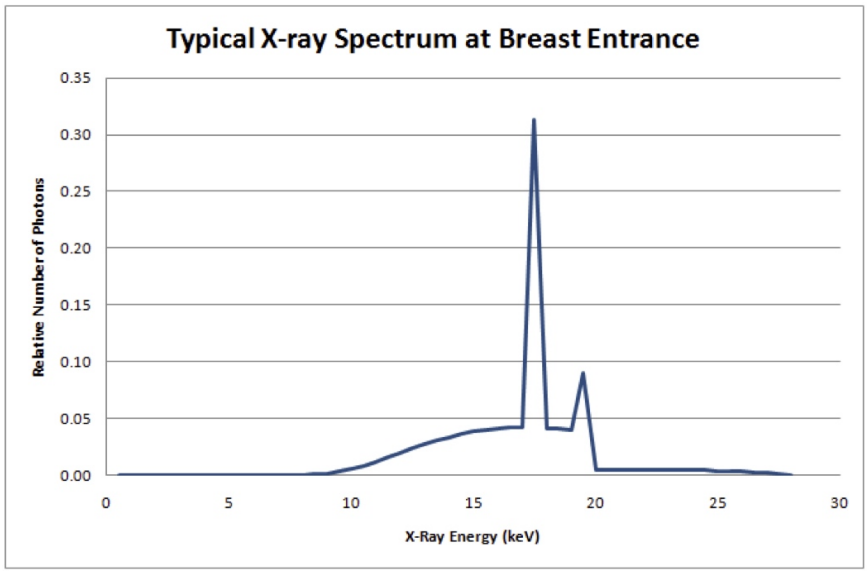


Figure 2.3: The continuous spectrum of energies produced by the Bremsstrahlung process.

x-ray per unit of thickness. Hence, a large coefficient suggests that a monoenergetic x-ray beam significantly weakens as it penetrates the material, whereas a small coefficient indicates that the beam is barely altered while it interacts with the object. The linear attenuation coefficient depends strongly on the energy of the incident photon and the nature of the material including the density and atomic number. Therefore, a single material will have a different linear attenuation coefficient for each energy level, usually expressed in units of inverse centimeters (cm^{-1}). Figure 2.4 shows the linear attenuation coefficients of skin, adipose and glandular tissue, iodine, and micro-calcifications in the range of 10 keV to 200 keV. In general, the linear attenuation coefficients decrease as the energy increases, except in the presence of an absorption edge (like in iodine).

The mass attenuation coefficient removes the dependence of the linear attenuation on the density of the material by measuring how much the material attenuates the energy per unit mass. In other words, it reflects the probability of interaction between the incident photons and the atoms making up the material. There is a special relationship between the linear and mass attenuation coefficient since the mass attenuation coefficient for a material at a given energy level $\left(\frac{\mu}{\rho}\right)_e$ is defined as the linear attenuation coefficient μ_e normalized to unit density ρ, or

$$\mu_e = \frac{\mu_e}{\rho} = \left(\frac{\mu}{\rho}\right)_e . \tag{2.1}$$

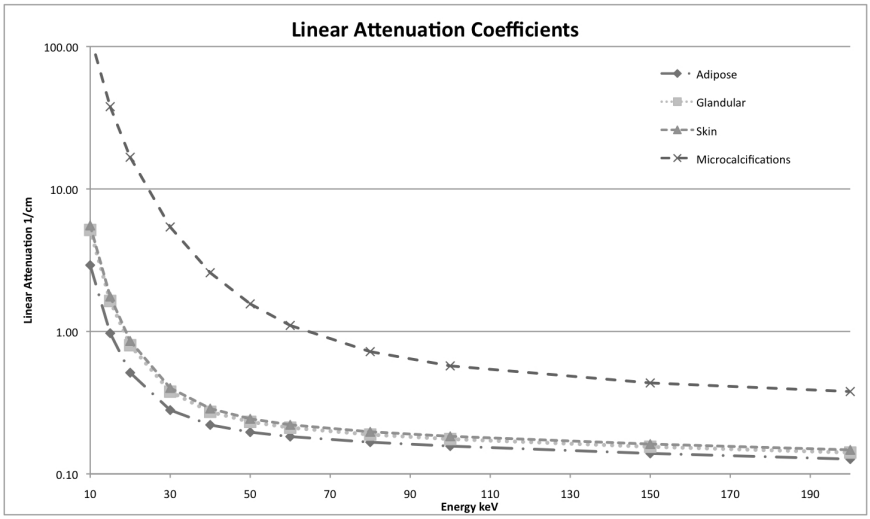


Figure 2.4: Linear attenuation coefficients for some materials relevant in breast imaging

The mass attenuation coefficient is expressed in terms of units cm^2/g. Figure 2.5 shows the mass attenuation coefficients in the energy range 10keV to 50keV for glandular tissue, adipose tissue, and micro-calcifications which are relevant materials in breast imaging.

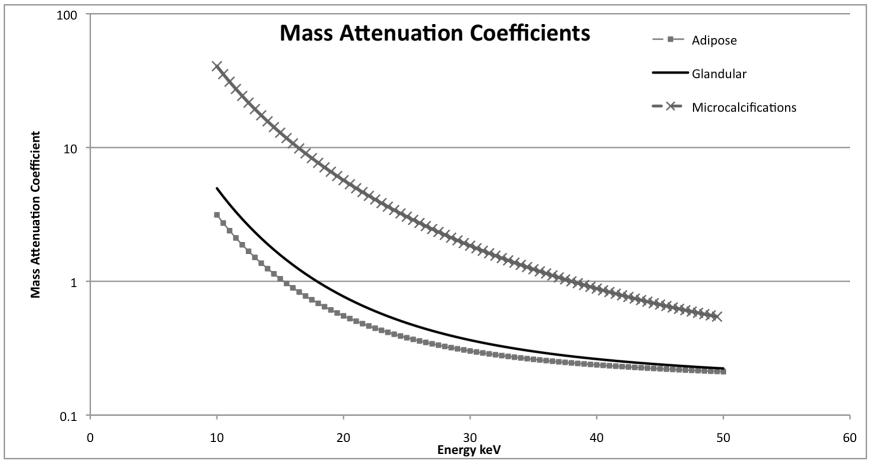


Figure 2.5: Mass attenuation coefficients for some materials relevant in breast imaging

2.1.3 The Detector

After an x-ray beam is produced by the Bremsstrahlung process and it undergoes attenuation by penetrating the imaged object, it arrives at the detector. The detector is some form of plate that sits underneath the object and measures the intensity of the x-ray beams as they exit the object. Early on, the detector was a sheet photographic film that was opaqued

by the x-rays, more so by the rays not attenuated by the object. Today mostly digital detectors are used in place of photographic film plates, particularly in mammographic screenings, since they produce better image quality at lower radiation dosages [20]. A digital detector produces an image where each pixel represents the intensity of energy measured at that particular location. The energy intensity can be measured as a count or as an accumulation of energy. The physics of imaging detectors is beyond the scope of this work, for reference see [50, 28, 70, 20].

2.2 Computed Tomography

Computed tomography (CT) started to be clinically available in the early 1970s with the advent of the modern computing era [20, 66]. This imaging modality revolutionized diagnostic medicine as it significantly reduced the practice of exploratory medicine. In CT, multiple x-ray projections of an object are taken along a 360 degree rotation and are then processed by a computer to produce *tomographs* of the patient. As opposed to standard radiography, CT is capable of creating a 3D representation of the object, reduce the problem of superposition and differentiate between similar types of materials like glandular and adipose tissue.

The problem of superposition in tomography is eliminated by obtaining different projections of an object along incident angles. Figure 2.6 shows an example of the limitation

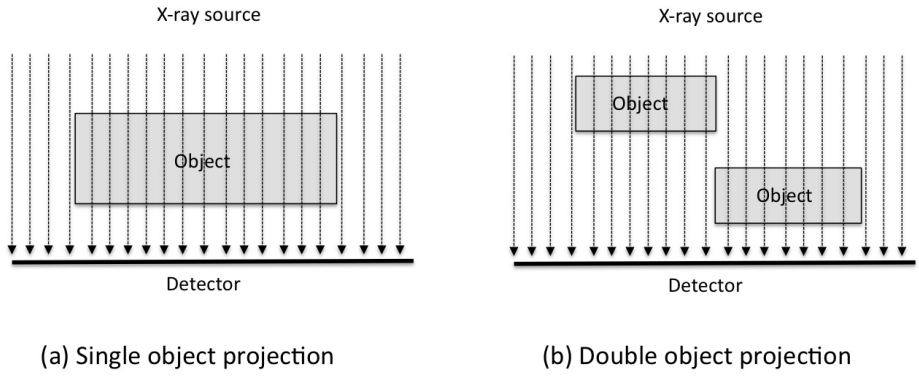


Figure 2.6: A single x-ray projection of one or two objects is indistinguishable.

of taking an x-ray projection from one single angle. The left hand side image is that of

one single object and the right hand side image shows the x-ray projection of two separate objects. It is clear that both scenarios are very different in terms of size and position of the object, however, the x-ray projection produced will be very similar and likely show the presence of one single object. By taking an x-ray projection of the right hand side case at an angle (like that shown in Figure 2.7) the two distinct objects will appear separately in the x-ray image. We could rotate the x-ray tube along more angles, as in Figure 2.8, to obtain more information about the distinct lesions inside the object.

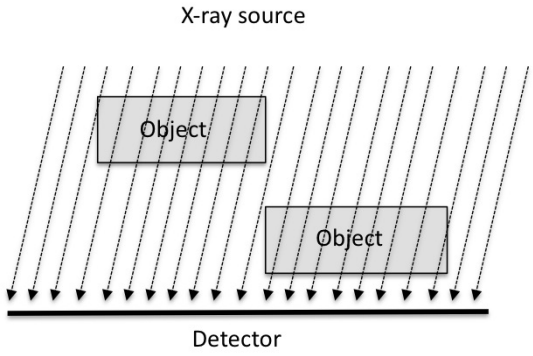


Figure 2.7: Using an angled x-ray tube to acquire the x-ray projection of two objects allows for distinction of individual lesions.

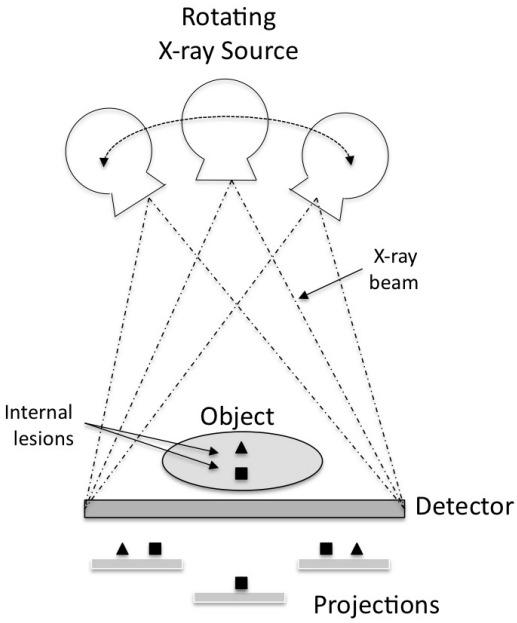


Figure 2.8: Multiple projections of an object obtained by a rotating x-ray source allow for distinction of individual lesions.

A modern CT scanner can capture 800 images of a patient along a 360 degree rotation in about five seconds, to produce a representation of the inside of the human body in terms of a stack of cross sectional images. This stack of images can be post-processed to create a 3D representation of the object that allows the physician to detect the presence of cancer, ruptured disks, aneurysms, and a variety of other pathologies [20]. The scheme for a typical CT scan is shown in Figure 2.9 where the object (in this case the patient) lays still while the x-ray tube and detector rotate simultaneously around it to obtain multiple projections along different angles. The collected data is then processed by a reconstruction

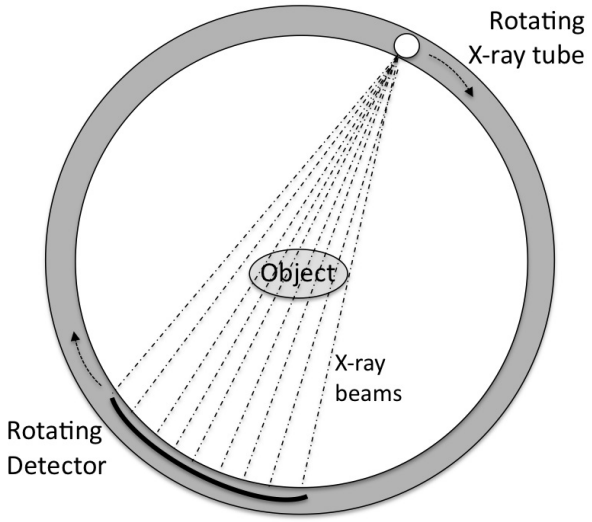


Figure 2.9: Geometry of a CT scanner. Both x-ray source and detector rotate along 360 degrees about an object to obtain hundreds of projections.

algorithm in a computer to output the 3D volume used for diagnosis. The widespread availability of scanners and the effectiveness of CT scans in diagnostic imaging has made this imaging modality replace radiography in many instances. In the United States alone, it is estimated that more than 60 million CT scans are performed annually. However, the benefits of a CT scan come at the price of radiation exposure since the patient is subject to significantly higher radiation levels than radiography by the multiple projections acquired [18]. Radiation exposure is a growing problem in the United States where about half of all the medical radiation exposure is due to CT scans [20]. Digital tomosynthesis can sometimes be an alternative to CT scans that seeks to lower radiation exposure while still providing an accurate depiction of the patient.

2.3 Digital Tomosynthesis

Digital tomosynthesis is an imaging technique that maintains the benefits of a CT scan but reduces the radiation exposure to the patient. It is used in cases where a full 3D representation of the object is not absolutely necessary, like in breast cancer screening and chest imaging [38]. In tomosynthesis, a form of what is called *limited angle* tomography, multiple projections of the object are obtained along a smaller range of incident angles in order to produce a *pseudo* 3D representation of the object. Whereas a full CT scan obtains hundreds of projections along a 360 degree range to reconstruct a representation of the object at an isotropic spatial resolution (a true 3D volume), digital tomosynthesis obtains 15-30 projections along a range of up to 45 degrees and reconstructs a representation of the object at a much lower resolution in the z-direction (in depth direction that is perpendicular to the projection plane labeled x-y). This smaller range of rotation allows for the collection of data at enough incident angles to capture lesions of interest while requiring a smaller radiation dosage.

There are different geometries of motion for the x-ray tube and detector in digital tomosythesis imaging depending on the part of the body that is being imaged. For example the parallel path geometry, where the source moves parallel to the detector, is mostly used for chest and abdominal tomosynthesis imaging. Another example geometry used for cone-beam CT is the complete isocentric motion geometry in which both the x-ray tube and the detector rotate along the same arc. The last example is partial isocentric motion described in Figure 2.8 where the x-ray tube rotates along an arc above the stationary detector. Most breast tomosynthesis is done with partial isocentric motion geometry since the construction of the imaging equipment is easier under these conditions [38]. In this thesis we focus on reconstruction algorithms for the partial isocentric motion geometry but our work can be extended in a straight forward manner to the complete isocentric motion and cone-beam CT.

The imaging system for digital breast tomosynthesis is described in Figure 2.10 where the x-ray tube rotates about a defined center of rotation along an arc perpendicular to the detector. The center of rotation is located along the source to imager distance (sid) and

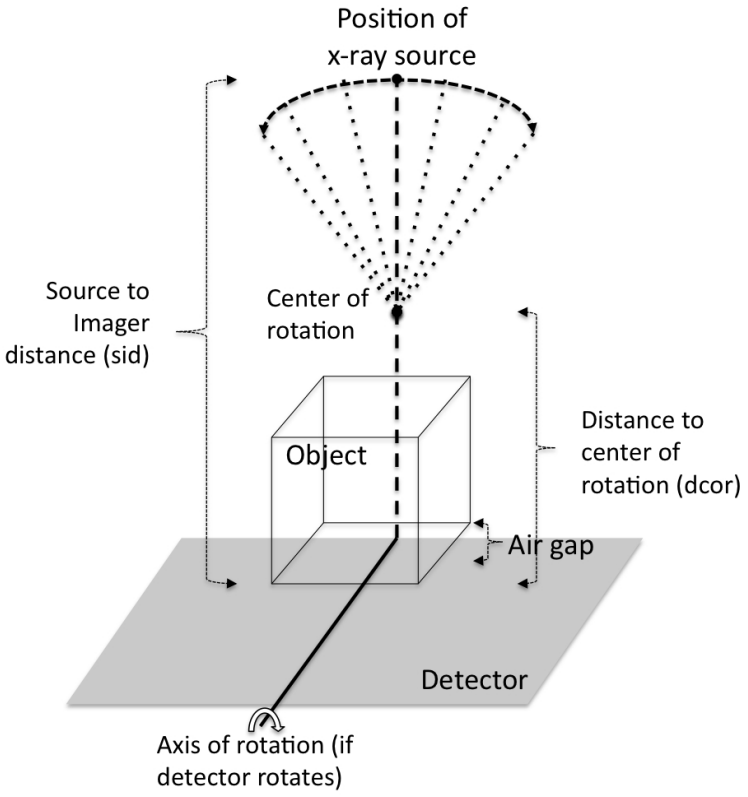


Figure 2.10: Geometry of the digital tomosynthesis imaging system.

can sometimes be found in the plane of the detector. The object (or the breast in digital breast tomosynthesis) sits stationary at a small distance above the detector, called the air gap. The detector, which can be stationary or rotate with the x-ray source, captures the images of the object at each incident angle and outputs the stack of projections that will be inputed to a reconstruction algorithm in order to produce a pseudo 3D representation of the object.

The geometry of the reconstructed pseudo 3D volume is shown in Figure 2.11. The reconstruction is a stack of tomographs that represent the different planes inside the object. Since digital tomosynthesis uses a limited rotation angular range, the resolution in the z-direction (the number of tomographs in the stack) is significantly less than the resolution of each individual tomograph. The slice thickness indicates the distance inside the object that the tomographs represent, but for our case the pseudo 3D volume contains equidistant tomographs.

This thesis focuses in the process of building a reconstructed pseudo 3D volume for

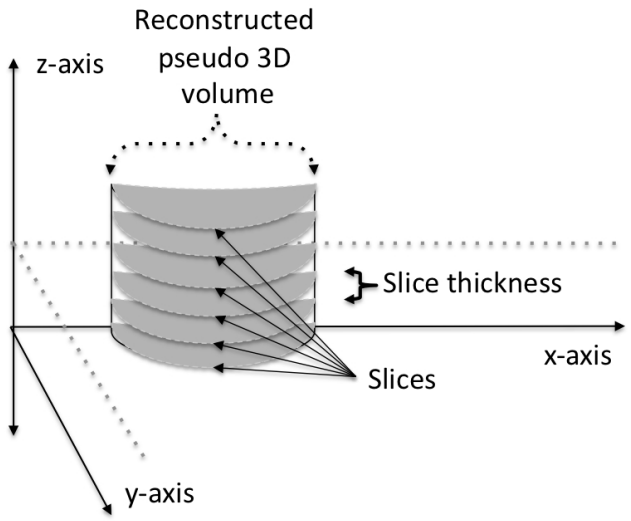


Figure 2.11: Reconstructed pseudo 3D volume is a stack of slices (or tomographs).

digital breast tomosynthesis after the projection data has been acquired by an imaging system with geometry similar to that shown in Figure 2.10. The process of reconstructing the pseudo 3D volume is mathematically and computationally challenging since we deal with millions of input pixels in order to produce a physically accurate reconstruction composed of billions of voxels. Standard approaches to solve this large scale inverse problem have relied in simplifying the physics of the image acquisition model by considering the x-ray beam to be monoenergetic (as opposed to modeling it as a spectral distribution produced by the Bremsstrahlung described in Section 2.1.1), thus decreasing the number of degrees of freedom and the computational complexity of the solution. However, this approach has been shown to introduce what is know as beam hardening artifacts to the reconstructed volume. Beam hardening and standard mathematical approaches used reconstruct the pseudo 3D volume are described in the next chapter.

Chapter 3

The Mathematics of Digital Tomosynthesis

As described in Section 1.2.3 a general inverse problem can be formulated as

$$\mathbf{b} = \mathbf{K}(\mathbf{x}_{\text{exact}}) + \boldsymbol{\eta} \tag{3.1}$$

where $\mathbf{b}$ is the observed measurement, $\mathbf{K}$ is a function that describes the forward process applied to $\mathbf{x}_{\text{exact}}$ to generate the measurements, and $\boldsymbol{\eta}$ represents any noise that may be present in the observed data. In digital tomosynthesis we use this formulation of a general inverse problem, where $\mathbf{b}$ represents the acquired projection data, $\mathbf{K}$ (also known as the forward model) represents the mathematical model that estimates the physical process to acquire the projection data $\mathbf{b}$, $\mathbf{x}_{\text{exact}}$ is the imaged object, and $\boldsymbol{\eta}$ represents any signal noise affecting the projection data.

In order to solve this inverse problem, we first need to focus on developing a physically accurate mathematical representation of the forward model $\mathbf{K}$. This chapter serves as an introduction to the forward models proposed for the digital tomosynthesis problem in this thesis which are described in chapters four through seven. We begin by providing a background for the statistical approach in our reconstruction process and introducing Beer's law, which is the basis of our forward models. We end the chapter by outlining the details as well as shortcomings of standard approaches used to solve this problem, expanding on

the concepts of the Radon transform and beam hardening.

3.1 Modeling the Source-Detector Interaction

In the reconstruction framework we propose for the digital tomosynthesis problem in the following chapters, we assume that the interaction between the source and the detector can be modeled as a statistical distribution. In this section, we provide the motivation for this assumption.

As noted in the previous chapter, an x-ray projection is produced by emitting a number of photons of a certain energy from a source (the x-ray tube) as a beam that travels through the imaged object and lands on a specific pixel (or group of pixels) in the detector. The projection image is given by the energy measured at each pixel in the detector. Since the source emits a finite quantity of photons, statistically we can represent the source as a Poisson distribution with intensity $\bar{\lambda}$ [46, 48] . Therefore, if the source emits N photons, then N is a random variable with probability distribution

$$P_s(k = N) = \frac{\bar{\lambda}^N e^{-\bar{\lambda}}}{N!}. \tag{3.2}$$

In addition, because an incident photon will either be counted or not counted by a receiving pixel, we can represent each pixel in the detector as a Bernoulli variable. If all pixels in the detector are calibrated the same, we can describe the detector as a group of independent Bernoulli variables with probability $0 \leq p \leq 1$ of the photon being counted. Then we have that the probability of a single pixel counting k photons given that N photons arrive is given by a conditional probability distribution that can be expressed as

$$P_d(k|N) = \begin{cases} \binom{N}{k} p^k (1-p)^{N-k} & k = 0, 1, ..., N \\ 0 & k > N \end{cases} \tag{3.3}$$

To find the probability of a pixel in the detector observing k photons, we use the definition of conditional probability and the two probability distributions described above. By

multiplying both distributions we have

$$\begin{aligned}
P_o(d=k) &= \sum_{N=k}^{\infty} P_s(N) P_d(k|N) \\
&= \sum_{N=k}^{\infty} \frac{\bar{\lambda}^N e^{-\bar{\lambda}}}{N!} \binom{N}{k} p^k (1-p)^{N-k} \\
&= e^{-\bar{\lambda}} \sum_{N=k}^{\infty} \frac{1}{k!(N-k)!} (\bar{\lambda}p)^k (\bar{\lambda}(1-p))^{N-k} \\
&= \frac{(\bar{\lambda}p)^k}{k!} e^{-\bar{\lambda}} e^{\bar{\lambda}(1-p)} \\
&= \frac{(\bar{\lambda}p)^k}{k!} e^{-\bar{\lambda}p}
\end{aligned}$$

which is a Poisson distribution with mean $\bar{\lambda}p$. This shows that we can model the variable that represents the number of photons counted by each pixel in the detector as the realization of Poisson random variable with mean related to the intensity of the source and the probability of detection. Moreover, the photon count for each individual pixel in the detector is independent from other pixels in the detector. This shows why in medical imaging, it is standard to model measurements due to x-ray transmission and photon counts as independently distributed Poisson random variables [32, 48, 119]. In the next section we introduce Beer's law in order to determine the expected value $\bar{\lambda}p$ for the Poisson distribution we use in our reconstruction algorithms.

3.2 Beer's Law

Beer's Law, also called the Beer-Lambert Law, describes the relationship between the attenuation of an x-ray beam and the concentration of attenuating material inside a medium through which the x-ray travels. In other words, it describes how the intensity of an x-ray is altered by traveling through an object. The law can be described in simple terms as an ordinary differential equation. In one dimension (see Figure 3.1), consider an x-ray beam of intensity $I(\ell)$ with initial intensity I_{in} traveling through an attenuating material from point a to point b. If the attenuating material is divided into infinitesimal slices of size $d\ell$,

we can express the difference in the intensity of the x-ray after crossing a single slice, called

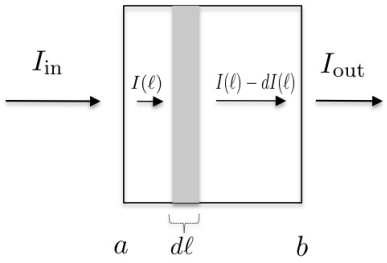


Figure 3.1: Beer's Law in one dimension. The x-ray passes through a cross-sectional area of length $d\ell$ and loses a fraction of intensity that depends on the attenuating medium.

$dI(\ell)$, as a fraction of the intensity entering the slice, represented by $\mu I(\ell)$, multiplied by the length of the slice $d\ell$. Note that μ, which is the fraction of the intensity of the x-ray left after it crosses each infinitesimal slice, depends on the linear attenuation coefficient of the material. Thus we have the ordinary differential equation

$$dI(\ell) = -\mu I(\ell) d\ell. \tag{3.4}$$

Similarly, in three dimensions we can describe Beer's law by assuming that the x-ray travels in a straight line L that is given parametrically in the form

$$L = \{\mathbf{x}_0 + \ell\mathbf{v} : \ell \in \mathbb{R}\} \tag{3.5}$$

where $||\mathbf{v}||_2 = 1, \mathbf{v} \in \mathbb{R}^3$ [46]. If we let the function $I(\mathbf{x})$ represent the x-ray flux at the point $\mathbf{x} \in \mathbb{R}^3$, then we can represent $i(\ell) = I(\mathbf{x}_0 + \ell\mathbf{v})$ as the intensity of the x-ray along the parametric line. Additionally, if we know the linear attenuation coefficient of the material at each point along the path of the x-ray beam by the function $\mu(\mathbf{x})$, we can express $m(\ell) = \mu(\mathbf{x}_0 + \ell\mathbf{v})$. By Equation (3.4), we can set up the relationship

$$\frac{d}{d\ell}[i(\ell)] = -m(\ell)i(\ell)$$

which is equivalent to

$$\frac{d}{d\ell}\log(i(\ell)) = -m(\ell). \tag{3.6}$$

Integrating Equation (3.6) along the x-ray path from $\ell = a$ to $\ell = b$ we have

$$\log\left(\frac{i(b)}{i(a)}\right) = -\int_a^b m(\ell)d\ell \tag{3.7}$$

and thus we obtain a more common expression of Beer's law

$$i(b) = i(a)\exp\left[-\int_a^b m(\ell)d\ell\right], \tag{3.8}$$

which says that the exit intensity of an x-ray beam traveling through an attenuating medium is given by the product of the initial intensity of the x-ray and the exponential of the line integral of the linear attenuation function along the path of the beam. The intensity of the x-ray beam is measured in units of $\mathrm{eV}/(\ell \times \mathrm{cm}^2)$. It is important to note that the derivation of Beer's law shows that attenuation is an isotropic process– the path of the x-ray beam has no influence on the level of attenuation as the direction vector $\mathbf{v}$ was chosen to be arbitrary.

Note that according to Beer's law in Equation (3.8), in order to estimate the exit intensity of the x-ray beam $i(b)$, we need to know the initial intensity of the x-ray beam $i(a)$. If we are modeling a *monoenergetic* (also called *monochromatic*) x-ray, which is an x-ray made up of photons of one energy or "color", the initial intensity of the beam will be a single known value. However, as described in section 2.1.1, the x-rays used in diagnostic imaging are a result of the Bremsstrahlung process which produces a *polyenergetic* beam, one consisting of photons of multiple energies described in terms of a *spectral distribution function* $s(e)$ (see Figure 2.3). In this case, we can say that the intensity of the x-ray beam given by the photons of energy e is $s(e)de$ [46]. Moreover, if e_1 and e_2 are neighboring energy values, then the intensity of the beam given by photons that have energies within the interval $[e_1, e_2]$ is found by

$$\int_{e_1}^{e_2} s(e)de.$$

If we are modeling the interaction of a polyenergetic x-ray with the object, we need to take into account the fact that the attenuation of the material will be different for each photon energy in the beam. This means that at a given point inside the object, the

attenuation function depends on both the path of the x-ray and the energy of the photons that interact with the matter at that point. Then, we can measure the exit intensity of a polyenergetic x-ray Ψ as an extension of Beer's law by measuring the the total energy output at the source with $\int_0^\infty s(e)de$ and writing the new attenuation function $\mu(\mathbf{x}_0+\ell\mathbf{v}, e)$. This gives

$$\Psi = \int_0^\infty s(e) \exp\left[-\int_L \mu(\mathbf{x}_0 + \ell\mathbf{v}, e)dL\right] de \tag{3.9}$$

where L represents the path that the beam travels and μ represents the attenuation of the material along the path.

Equation (3.9) is the general form of the forward problem for digital tomosyntheis imaging. In terms of the nonlinear inverse problem shown in Equation (3.1), the measured data $\mathbf{b}$ is given by Ψ, our imaged object $\mathbf{x}_{\text{exact}}$ is represented in terms of its attenuation properties μ, and the forward model $\mathbf{K}(\mu)$ is the full nonlinear integral equation. Our aim in the reconstruction process is to solve for μ in Equation (3.9) which is a very hard problem to solve given the nonlinearity on the unknown. In the remainder of this chapter and this thesis we will develop the mathematical and computational framework to solve this integral equation without simplifying its formulation.

3.3 The Radon Transform

Equations (3.8) and (3.9) describe the intensity of a single monoenergetic or polyenergetic x-ray as it penetrates the object. However, in the image acquisition process for digital tomosynthesis, each projection is obtained by sending multiple x-rays to travel through the object at once. We can combine the formulation of Beer's law for all the different x-ray beams into one single expression, identifying each x-ray by the pair θ, t. To do this, we begin by linearizing the formulation of Beer's law for a monoenergetic x-ray in Equation (3.7), letting $\Phi_{t,\theta}$ represent the left-hand side of the equation for each ray. In order to generalize the right-hand side of Equation (3.7), we expand the representation of our parametric line L to the set of all oriented lines in the space $\mathbb{R}^2$ by constructing the orthogonal unit vectors

$$\boldsymbol{\omega}(\theta) = (\sin(\theta), \cos(\theta)) \ , \ \hat{\boldsymbol{\omega}}(\theta) = (-\cos(\theta), \sin(\theta))$$

to express

$$L_{\theta,t} = \{t\boldsymbol{\omega} + s\hat{\boldsymbol{\omega}} : s \in \mathbb{R}\}$$

where t is the distance of the oriented line to the origin. Therefore, we can model Beer's law for all monoenergetic x-rays penetrating the object by

$$\Phi_{t,\theta} = -\int_{L_{t,\theta}} \mu(t\boldsymbol{\omega} + s\hat{\boldsymbol{\omega}})\ dL_{t,\theta}.$$

If we denote the function

$$\mathcal{R}\mu(t,\theta) = \Phi_{t,\theta}$$

for $t_1 < t < t_2$ and $\theta_1 < \theta < \theta_2$, then the forward model for the acquisition of one projection in digital tomosynthesis using a monoeregetic x-ray is given by the set of line integrals

$$\mathcal{R}\mu(t,\theta) = -\int_{L_{t,\theta}} \mu(t\boldsymbol{\omega} + s\hat{\boldsymbol{\omega}})\ dL_{t,\theta}. \tag{3.10}$$

which is by definition the Radon transform of the function μ [46]. Figure 3.2 shows the Radon transform of the famous Shepp-Logan phantom which is a standard phantom used in tomography [71, 84]. The transform was computed using the radon routine in MATLAB®. The object is considered two dimensional and each each column of the right-hand side image

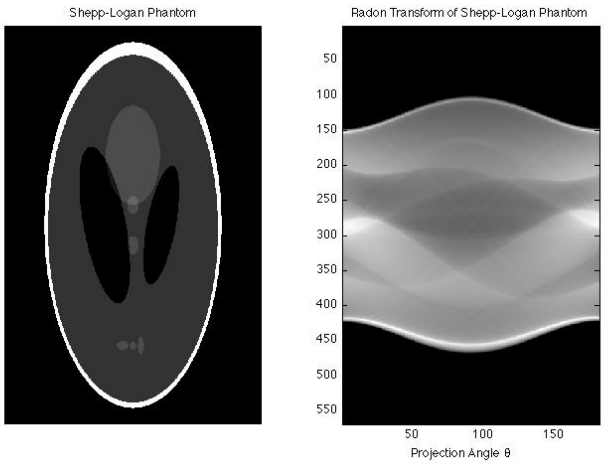


Figure 3.2: Radon transform of Shepp-Logan phantom

represents a one dimensional projection of the phantom.

We have shown that each projection of an object obtained in tomography using a monoenergetic source can be modeled as the Radon transform of the function representing the attenuation of the materials composing the object. The reconstruction problem in this case is much easier to solve in terms of the well developed mathematical theory of the Radon transform. However, it is important to note that the assumption of a monoenergetic x-ray for clinical situations is far from the true image acquisition described in the previous chapter. Simplifying the polyenergetic x-ray in Equation (3.9) to the monoeregetic case of Equation (3.8) makes for an easier mathematical solution but it comes at the price of artifacts in the reconstruction as will be discussed later in this chapter.

3.4 Standard Reconstruction Methods

3.4.1 Backprojection

If the Radon transform represents the forward model for an x-ray projection in the monochromatic case, we can attempt to solve the reconstruction problem by finding the inverse or an approximation of the inverse of the Radon transform. The easiest approximation to the inverse is the process called *backprojection* in which the attenuation function μ is reconstructed by averaging the projection data over all the lines that pass through a given point. Figure 3.3 shows the idea of backprojection in simple terms, where the lines represent each individual x-ray beam [20]. The projection data **A, B, C,** and **D** is the linear sum of the attenuation of each box that an individual x-ray crosses– the discretized line integral. If we are trying to find the value corresponding to the attenuation of a single box, we can trace back all rays that traveled through the box and determine the value by looking at the projection data of all these rays. That is, we fix a point $\mathbf{x}$ in the object and we "backproject" all the x-rays that passed through the point.

In terms of the Radon transform, we can consider the set of all lines

$$L_{\theta,t} = \{t\boldsymbol{\omega} + s\hat{\boldsymbol{\omega}} : s \in \mathbb{R}\}$$

of a fixed the direction $\boldsymbol{\omega}$, and the line that passes through the point $\mathbf{x}$ is given by the inner

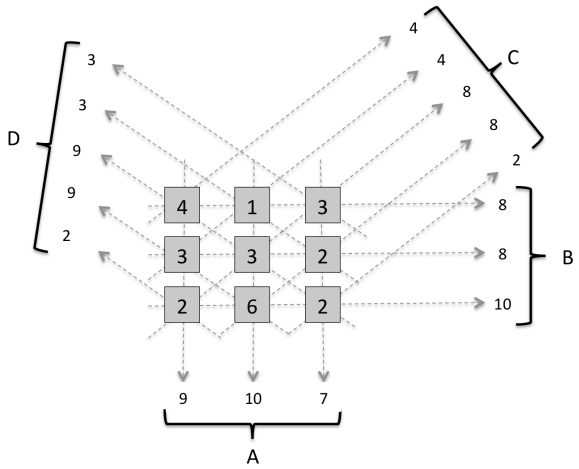


Figure 3.3: Example of a simple backprojecion. **A, B, C, D** represent the projection data and the 3×3 boxes in the center of the image represent the unknown attenuation coefficients of each unit in the discretized object. The attenuation is measured as integers for the purpose of explanation. The goal of a backprojection is to reconstruct the numbers in the 3×3 boxes given the projection data [20].

product $t = \langle \mathbf{x}, \boldsymbol{\omega}(\theta) \rangle$ [46]. The value of the attenuation at point $\mathbf{x}$ is the average, given by the backprojection formula

$$\tilde{\mu}(\mathbf{x}) = \frac{1}{2\pi} \int_0^{2\pi} \mathcal{R}\mu(t, \theta) d\theta. \tag{3.11}$$

The backprojection operation applied to the Radon transform of the Shepp-Logan phantom in Figure 3.2 is shown in Figure 3.4. The reconstruction was done using the unfiltered iradon routine in MATLAB®. Note that this is not a very good reconstruction of the

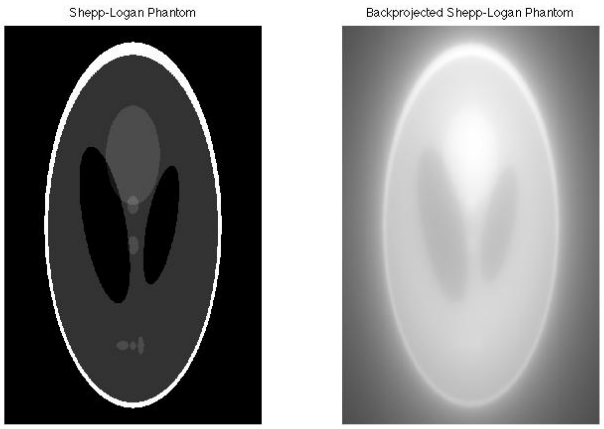


Figure 3.4: Backprojection of Shepp-Logan phantom

phantom because it is corrupted by some type blur. The backprojection formula alone is not accurate enough for reconstructions in tomography, but it can be refined or *filtered* to produce excellent results. In fact, filtered backprojection is the standard reconstruction technique for computed tomography and digital tomosynthesis in modern imaging devices.

3.4.2 Filtered Backprojection

In filtered backprojection the Radon transform is altered or *filtered* in some fashion to smooth the components of the function that produce the blur in a regular backprojection. The type of filter used depends on the scanner geometry and the properties of the problem. Mathematically, the process of filtered backprojection is an analytical inversion of the Radon transform that uses the Fourier transform. The relationship between these two integral transforms is expressed by the *central slice theorem*.

Theorem 3.4.1. (Central slice theorem) *Let μ be an absolutely integrable function in the domain of $\mathcal{R}$. For any real number r and unit vector $\boldsymbol{\omega}$, we have the identity*

$$\int_{-\infty}^{\infty} \mathcal{R}\mu(t,\boldsymbol{\omega}) e^{-itr} dt = \hat{\mu}(r\boldsymbol{\omega})$$

where $\hat{\mu}(r\boldsymbol{\omega})$ is the Fourier trasnform of $\mu(\mathbf{x})$. For proof of the theorem see [46].

In order to simplify notation, let $\widetilde{\mathcal{R}\mu}(r,\boldsymbol{\omega}) = \hat{\mu}(r\boldsymbol{\omega})$. If we use polar coordinates to express the Fourier inversion formula we have

$$\mu(\mathbf{x}) = \frac{1}{[2\pi]^2} \int_0^{2\pi} \int_{-\infty}^{\infty} \hat{\mu}(r\boldsymbol{\omega}) \mathrm{e}^{ir\langle \mathbf{x},\boldsymbol{\omega}\rangle} r dr d\boldsymbol{\omega}.$$

By Theorem 3.4.1 and since the Radon transform is an even function we can say that $\widetilde{\mathcal{R}\mu}(r,\boldsymbol{\omega}) = \widetilde{\mathcal{R}\mu}(-r,-\boldsymbol{\omega})$ and, furthermore that

$$\mu(\mathbf{x}) = \frac{1}{[2\pi]^2} \int_0^{\pi} \int_{-\infty}^{\infty} \widetilde{\mathcal{R}\mu}(r,\boldsymbol{\omega}) |r| \mathrm{e}^{ir\langle \mathbf{x},\boldsymbol{\omega}\rangle} dr d\boldsymbol{\omega}. \tag{3.12}$$

The expression for the attenuation function $\mu(\mathbf{x})$ in Equation (3.12) suggests that we can express filtered backprojection operation as a two step process.

1. The first step is the application of a *filter* to the Radon transform that acts on the parameter t whose output is

$$\mathcal{GR}\mu(\mathbf{x}) = \frac{1}{2\pi}\int_{-\infty}^{\infty} \widetilde{\mathcal{R}\mu}(r,\boldsymbol{\omega})|r|\mathrm{e}^{ir\langle \mathbf{x},\boldsymbol{\omega}\rangle}dr \tag{3.13}$$

   which is the inner integral of Equation (3.12).

2. The second step is the backprojection formula in Equation (3.11) applied to the filtered Radon transform in Equation (3.13) as

$$\mu(\mathbf{x}) = \frac{1}{2\pi}\int_{0}^{\pi} \mathcal{GR}\mu(\langle \mathbf{x},\boldsymbol{\omega}\rangle,\boldsymbol{\omega})d\boldsymbol{\omega} \tag{3.14}$$

   which is the radial integral in Equation (3.13).

Figure 3.5 shows the result of a filtered backprojection reconstruction of the Shepp-Logan phantom using a cropped Ram-Lak filter (also called ramp filter). By filtering the backpro-

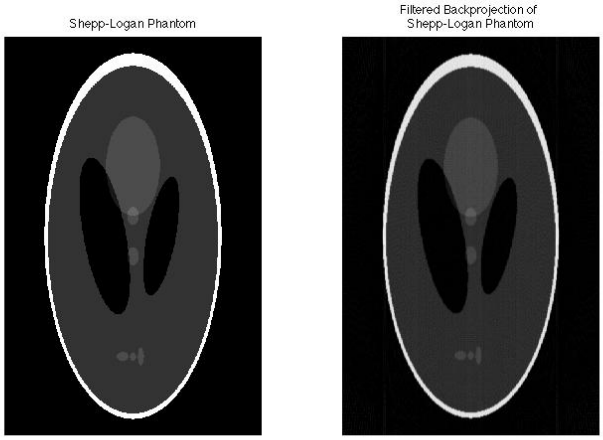


Figure 3.5: Filtered backprojection of Shepp-Logan phantom

jection, the blur that corrupted the reconstruction in Figure 3.5 has disappeared. Much work has been done in the development and implementation of filter functions for the tomography reconstruction problem, for more details see [46, 87, 66]. However, as in backprojection, we are assuming that the x-rays used to obtain each projection are monoeregetic, since we use the Radon transform to model the image acquisition process.

3.4.3 Maximum Likelihood Estimation Maximization Method (MLEM)

The MLEM method provides a statistical based reconstruction approach that eliminates the systematic biases introduced to the projection data by the logarithm operation in filtered backprojection (Equation (3.7)). The idea in MLEM to use the actual measured data, $i(a)$ and $i(b)$ in Equation (3.7), in the forward model as opposed to the logarithm of their ratio. Using the raw data is preferential, especially in low-count scans which are more sensitive to the biases introduced by the nonlinearity of the logarithm function [48, 19]. In addition, using a statistical method is a more reasonable approach to solve the reconstruction problem given the possibility of noise in the acquired projections (see Section 1.2).

To build the MLEM reconstruction framework, let $\bar{s}^i$ be the average number of photons that the detector would read for the ith x-ray in the absence of an attenuating object. Then by Beer's law we have that the intensity for x-ray i at the detector is

$$\mathbf{b}^i = \bar{s}^i \exp\left[\int_L \mu(\mathbf{x_0} + \ell\mathbf{v})dL\right]$$

where L is the path of the ray and μ represents the attenuation of the object. In the medical imaging community, it is standard practice to model the measurements obtained by x-ray transmission as independently distributed Poisson random variables with additive noise [48, 46, 87, 66]. The additive noise variable accounts for background effects such as scattering and has known statistical properties. If the variable y^i denotes the number of photons for x-ray i measured at the detector, we can write its expected value using Beer's law by

$$E[y^i] = \bar{s}^i \exp\left[\int_L \mu(\mathbf{x})dL\right] + \eta^i \tag{3.15}$$

where η^i represents additive noise. To discretize the model, we can write the attenuation function μ in terms of the voxel basis (pixel basis in 2D) $\chi_j(\mathbf{x})$, where $\chi_j(\mathbf{x}) = 1$ inside voxel j and zero elsewhere. This allows us to write the integral

$$\int_L \mu(\mathbf{x})dL = \sum_{j=1}^{N_v} a^{ij}\boldsymbol{\mu}_j$$

where N_v is the number of voxels, μ_j is the attenuation of voxel j, and a^{ij} represents the length of the x-ray beam traveling through voxel j. If we organize all values a^{ij} in the raytrace matrix A, we have $y^i \sim \text{Poisson}(\bar{\mathbf{y}}(\boldsymbol{\mu}_{true}))$ where

$$\bar{\mathbf{y}}^i(\boldsymbol{\mu}) = \bar{s}^i \exp\left(-\left[A\boldsymbol{\mu}\right]^i\right) + \boldsymbol{\eta}^i$$

and the value

$$\left[A\boldsymbol{\mu}\right]^i = \sum_{j=1}^{N_v} a^{ij}\boldsymbol{\mu}_j.$$

Knowing the probability distribution for the projection measurements, we can use the theory of maximum likelihood estimation to find the reconstructed volume. For the Poisson variable $\mathbf{y}^i$ we have the likelihood function

$$p(\mathbf{y};\boldsymbol{\mu}) = \prod_{i=1}^{N_p} \frac{\exp(-\bar{\mathbf{y}}^i(\boldsymbol{\mu}))[\bar{\mathbf{y}}^i(\boldsymbol{\mu})]^{\mathbf{y}^i}}{\mathbf{y}^i!}$$

and its respective negative log-likelihood function

$$L(\mathbf{y};\boldsymbol{\mu}) = -\log(p(\mathbf{y};\boldsymbol{\mu})) = \sum_{i=1}^{N_p} \left[\bar{\mathbf{y}}^i(\boldsymbol{\mu}) - \mathbf{y}^i \log[\bar{\mathbf{y}}^i(\boldsymbol{\mu})] + \log\left(\mathbf{y}^i!\right)\right].$$

The reconstructed volume using the MLEM method is the vector $\boldsymbol{\mu}_{MLE}$ that solves the numerical optimization problem

$$\boldsymbol{\mu}_{MLE} = \min_{\boldsymbol{\mu}>\mathbf{0}} \left\{L(\mathbf{y};\boldsymbol{\mu})\right\}.$$

This numerical optimization problem can be solved using a myriad of numerical algorithms available.

Formulating the reconstruction problem in MLEM as a numerical optimization problem gives much more flexibility in the reconstruction than filtered backprojection. For MLEM methods we can incorporate things like regularization, additional constraints, and penalty functions to the reconstruction process in order to refine the solution volume. However, just like filtered backprojection, the MLEM approach makes an initial assumption that the pro-

jection data is obtained using monoenergetic x-rays. As described in the previous chapter, this assumption is not realistic when x-rays are produced using the Bremsstrahlung radiation process. The consequences of this assumption are artifacts caused by beam hardening which is discussed at the end of this chapter.

3.4.4 Algebraic Reconstruction Techniques (ART)

A class of iterative approaches to compute a reconstruction from projection data is called algebraic reconstruction techniques (ART). This class of techniques is based on variations of Kaczmarz's method (or method of projections) in numerical linear algebra. ARTs can be a good alternative to filtered backprojection in tomography reconstructions since they can incorporate relaxation parameters in order to control the effects of inconsistent data and the presence of noise in the projections. This section briefly describes how ARTs and iterative methods can be applied to tomography reconstructions, for further details see [66, 87, 56, 46].

The idea in ARTs is to express the image acquisition process in tomography and digital tomosynthesis as a linear system that can be solved iteratively. If we assume a monoenergetic x-ray, the image acquisition process is described in terms of a linear relationship between the observed data and the unknown attenuation function by the Radon transform in Equation (3.10). In order to build the linear system, we must choose a set of basis functions for the problem. In ARTs, the basis of choice is the pixel basis because it is relatively easy to compute and allows for a straight forward discretization of the line integrals in Equation (3.10). If the observed data at pixel i for a projection at incident angle θ is denoted $\mathbf{b}_{i,\theta}$ and the unknown attenuation of voxel j in the object is denoted by $\boldsymbol{\mu}_j$, then linear system is given by

$$\mathbf{b}_\theta = -A_\theta \boldsymbol{\mu}$$

where the ith row of A_θ is called $\mathbf{a}_i^\theta$ and represents a vector whose entry $\mathbf{a}_{i,j}^\theta$ is the length of the intersection of the x-ray beam at incident angle θ with voxel j that contributed (or "landed") on pixel i. That is, the inner product of row i of matrix A_θ and attenuation vector $\boldsymbol{\mu}$ is the discretization of a line integral in the Radon transform. If we are dealing

with N_θ projections, we can write the right-hand side vector and the linear operator as

$$\mathbf{b} = \begin{bmatrix} \mathbf{b}_1 \\ \vdots \\ \mathbf{b}_{N_\theta} \end{bmatrix} \quad \text{and} \quad A = \begin{bmatrix} A_1 \\ \vdots \\ A_{N_\theta} \end{bmatrix}.$$

Using the expanded linear system set-up above, we apply ARTs method to the system

$$\mathbf{b} = -A\boldsymbol{\mu}.$$

ARTs are a good alternative to filtered backprojection reconstructions because of the many variations that exist and the freedom they provide in choosing a set of basis functions [66]. This means that the reconstruction is not necessarily directly related to the inversion of the Radon transform [46]. Also, they can accommodate complicated beam profiles in the construction of the linear operator. However, the measurement matrix in ARTs can become very large for clinical size problems and computing or storing its entries becomes a challenge. Furthermore, similarly to filtered backprojection and MLEM, the only way to linearize the digital tomosynthesis model from Beer's law is by assuming a monoenergetic x-ray, which has its consequences as described in the next section.

3.5 Beam Hardening

The standard methods for the computed tomography and the digital tomosyntheis reconstruction problem described in the previous section make a very strong initial assumption: the projection data is produced using monoenergetic x-rays. If we have a monoenergetic forward model for the image acquisition process, then the problem simplifies from Equation (3.9) to Equation (3.7) and we can study the reconstruction process in terms of integral transforms using well known analytical tools. However, while reasonably sound, this assumption is far from the true physics of the image acquisition process (see Section 2.1.1) and can lead to errors in the form of artifacts in the computed solution.

Recall that in diagnostic imaging, x-ray beams are a result of the Bremsstrahlung pro-

cess, which produces x-rays composed of photons of different energies expressed in terms of a spectral distribution function (see Figure 2.3). In reality, as a polyenergetic x-ray beam penetrates the imaged object, lower energy photons are preferentially absorbed by the matter, causing a shift in the spectral distribution function. This effect is called *beam hardening* and is shown in Figure 3.6. Here we see that the spectrum of energies in the beam that

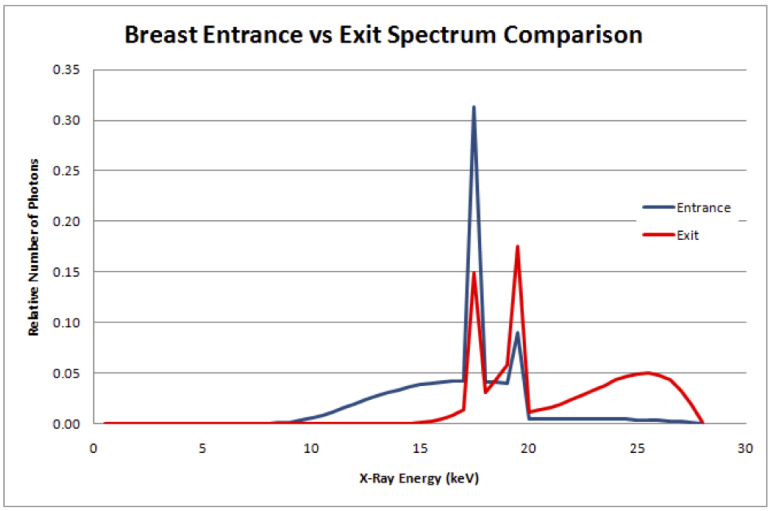
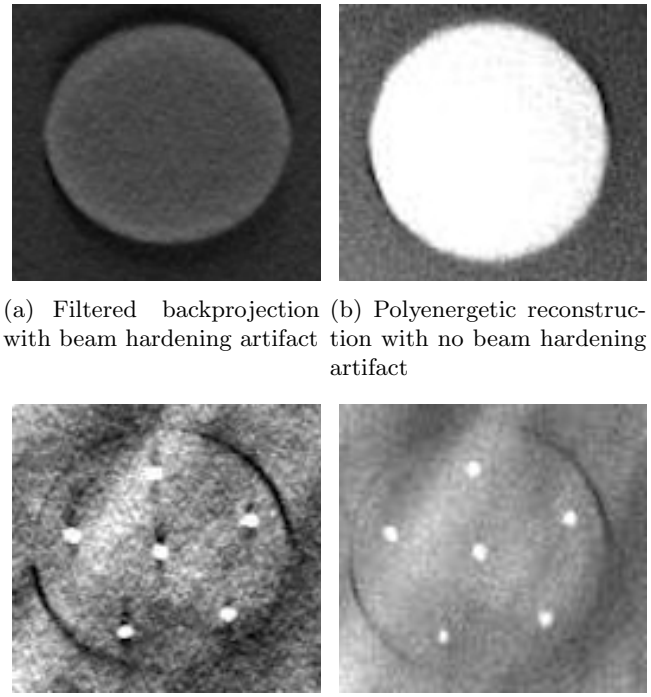


Figure 3.6: Beam hardening in terms of energy intensity. The entrance intensity of the spectrum is higher than the exit intensity. Low energy photons are absorbed by the object, thus changing the average energy of the beam. Mean energy at entrance is 16.8keV and mean energy at exit is 21.5keV.

enters the object is very different from the spectrum of energies that exits the object and that the mean value of the x-ray beam is changed. This is the reason why the attenuation of a particular material is dependent both on its atomic properties and the energy of the x-ray since lower energy x-rays change more (or disappear) as they penetrate.

Modern x-ray machines can account for beam hardening by using certain filters, adding certain material to the beam, or post-processing the results [20, 46, 72], but the reconstruction can still suffer from *beam hardening artifacts* which are artifacts caused by this effect [19]. Beam hardening artifacts are dark streaks or dark cupping-like shadows around lesions of interest like those shown in Figures 3.7(a) and 3.7(c). The left-hand side reconstructions where done using a filtered backprojection algorithm and the right-hand side reconstructions were done using the polyenergetic reconstruction methods proposed in this thesis. Figure 3.7 shows that the real benefit of a polyenergetic reconstruction that directly



(a) Filtered backprojection with beam hardening artifact

(b) Polyenergetic reconstruction with no beam hardening artifact

(c) Filtered backprojection with beam hardening artifacts around each feature

(d) Polyenergetic reconstruction with no beam hardening artifacts

Figure 3.7: Examples of beam hardening artifacts. Figures 3.7(a) and 3.7(c) show beam hardening artifacts in the form of shadow-like objects cupping the features reconstructed using a filtered backprojection method. Figures 3.7(b) and 3.7(d) show a reconstruction of the same features using the polyenergetic methods described in this thesis with no beam hardening artifacts.

solves the digital tomosynthesis problem using Equation (3.9) is a physically accurate and artifact-free depiction of the patient. In the next four chapters we develop the theory of four different forward models for the polyenergetic digital tomosynthesis problem and the framework to produce clean and accurate reconstructions for breast cancer diagnosis.

Chapter 4

The Linear Polyenergetic Model for Attenuation

In 2010 Chung, Nagy, and Sechopoulos proposed a mathematical model for digital breast tomosynthesis reconstruction that explicitly accounts for the polyenergetic nature of the x-ray beam [32]. In their work, they showed it is possible to produce digital tomosythesis reconstructions that are free of beam artifacts using the polyenergetic formulation of Beer's law in Equation (3.9). Their idea was to decompose the attenuation function in the forward projection model in terms of the percentage of each material composing the breast. This way they were able to reconstruct the breast in terms of the glandular fractions (the percentage of glandular tissue inside each voxel), as opposed to the standard approaches of Section 3.4 which reconstruct the breast in terms of density values. Using glandular fractions, they were able to describe the unknown attenuation function by a given structure, solve the polyenergetic reconstruction problem, and derive a physically meaningful quantitative measure for each voxel in the resulting volume. In this chapter we describe the reconstruction framework described in [32] as a basis for the models described in the next chapters.

4.1 Building the Forward Model

In the single material model Chung, Nagy and Sechopoulus began by considering each voxel in the discretized volume to be composed of only two materials: adipose tissue $\mathbf{w}_{j,1}$ and glandular tissue $\mathbf{w}_{j,2}$. Therefore, for a given energy e, the linear attenuation of voxel j in the discretized volume can then be expressed as a sum of the product of the amount of each material present in the voxel and the respective linear attenuation of the material. That is

$$\boldsymbol{\mu}_{e,j} = u_{e,1}\mathbf{w}_{j,1} + u_{e,2}\mathbf{w}_{j,2} \tag{4.1}$$

where $u_{e,1}, u_{e,2}$ are the linear attenuation coefficients for adipose and glandular tissue at energy level e [12], and the values $\mathbf{w}_{j,1}$ and $\mathbf{w}_{j,2}$ contain the percentage of each material present in the voxel j. By assumption each voxel j is composed of only two materials, so we can say that

$$\mathbf{w}_{j,1} + \mathbf{w}_{j,2} = 1$$

or, the composition of adipose and glandular tissue add up to 100% of the composition of voxel j. Without loss of generality, the adipose variable $\mathbf{w}_{j,1}$ can be eliminated by writing

$$\mathbf{w}_{j,1} = 1 - \mathbf{w}_{j,2}$$

which results in the new linear attenuation model for voxel j

$$\boldsymbol{\mu}_{e,j} = [u_{e,2} - u_{e,1}]\mathbf{w}_{j,2} + u_{e,1} \tag{4.2}$$

Equation (4.2) models the attenuation of each voxel j for energy level e, where $u_{e,1}, u_{e,2}$ are known linear fit coefficients that depend on the particular energy level e and the linear attenuation coefficients of adipose and glandular tissue. The vector $\mathbf{w}_{:,2}$ is called the glandular fraction because it contains the percentage of glandular tissue present inside the breast. To formulate the reconstruction problem as a nonlinear inverse problem, we begin by building the forward model using the attenuation function for the object developed in Equation (4.2) and applying Beer's law. From Equation (3.9), Beer's law for a single

polyenergetic x-ray i is given by

$$\Psi^i = \int_0^\infty s(e) \exp\left[-\int_L \mu(\mathbf{x}, e) dL\right] de.$$

This integral equation can be discretized by dividing the object into N_v voxels and using a finite number of levels of energy to approximate the spectral distribution function. The discrete formulation of Beer's law for a polyenergetic x-ray is then given by

$$\mathbf{b}^i = \sum_{e=1}^{N_e} s_e \exp\left[-\sum_{j=1}^{N_v} a^{ij} \boldsymbol{\mu}_{e,j}\right] \tag{4.3}$$

where N_e represents the number of discrete levels of energy in the x-ray beam, $\boldsymbol{\mu}_{e,j}$ is the attenuation of voxel j for energy level e, and a^{ij} represents the length of the x-ray beam traveling through voxel j. Using Equation (4.2) we have that the monochromatic x-ray for pixel i is

$$\begin{aligned}\sum_{j=1}^{N_v} \mu_{e,j} a^{ij} =& [u_{e,2} - u_{e,1}] \sum_{j=1}^{N_v} a^{ij} \mathbf{w}_{j,2} + u_{e,1} \sum_{j=1}^{N_v} a^{ij} \\ =& [u_{e,2} - u_{e,1}] \mathbf{a}^{i:} \bullet \mathbf{w}_{j,:} + u_{e,1} \mathbf{a}^{i:} \bullet \mathbf{1}\end{aligned}$$

where the symbol $\bullet$ denotes a dot product operation and $\mathbf{1}$ is a column vector of ones of length N_v. Note that the scalars a^{ij} can be organized for the given projection angle θ into the matrix A_θ, and the row vector $\mathbf{a}_\theta^{i,:}$ will represent the i-th row of the matrix. The forward model for pixel i in the projection taken at incident angle θ is represented as

$$\mathbf{b}_\theta^i = \sum_{e=1}^{N_e} s_e \exp\left(-[(u_{e,2} - u_{e,1}) \mathbf{a}_\theta^{i:} \bullet \mathbf{w}_{:,2} + u_{e,1} \mathbf{a}_\theta^{i:} \bullet \mathbf{1}]\right) \tag{4.4}$$

where the scalar s_e is the value of the spectral distribution function at energy e, representing the number of incident photons at that energy (called the *energy fluence*). Using matrix notation to simplify the model, the entire projection acquired at incident angle θ is

$$\mathbf{b}_\theta = \sum_{e=1}^{N_e} s_e \exp\left(-[(u_{e,2} - u_{e,1}) A_\theta \mathbf{w}_{:,2} + u_{e,1} A_\theta \mathbf{1}]\right) \tag{4.5}$$

Equation (4.5) is the discrete forward model for a single incident angle for the polyenergetic image acquisition process in digital tomosynthesis.

4.2 Statistical Considerations

As discussed in Section 3.1 the source-detector interaction in digital tomosynthesis can be modeled as a Bernoulli random variable (the detector) receiving a Poisson random variable as "input" (the source). This allows us to model the variable representing the energy measured by each pixel in the detector as a Poisson random variable. In the medical imaging community this is standard practice, modeling the measurements obtained by x-ray transmission as independently distributed Poisson random variables with additive noise. Given the forward model in Equation (4.4), the expected value of the measured data for pixel i at angle θ is

$$E(\mathbf{b}_\theta^i, \mathbf{w}_{:,2}) = \sum_{e=1}^{N_e} s_e \exp\left(-[(u_{e,2} - u_{e,1})\mathbf{a}_\theta^{i:} \bullet \mathbf{w}_{:,2} + u_{e,1}\mathbf{a}_\theta^{i:} \bullet \mathbf{1}]\right) + \bar{\boldsymbol{\eta}}^i$$

where $\bar{\boldsymbol{\eta}}^i$ represents error in the observed data that is attributed to noise or scatter. This error is also assumed to follow a Poisson distribution whose statistical mean is known or can be approximated.

This means that each pixel i in the observed projection data at incident angle θ is a realization of the Poisson random variable

$$\mathbf{b}_\theta^i \sim \text{Poisson}(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}_\theta).$$

Using the theory of maximum likelihood estimation, we can say that the probability of observing the projection data $\mathbf{b}_\theta$ given the volume $\mathbf{w}_{:,2}$ represented in terms of its attenuation, is modeled by the *likelihood function*

$$p(\mathbf{b}_\theta, \mathbf{w}_{:,2}) = \prod_{i=1}^{N_p} \frac{e^{-(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)^{\mathbf{b}_\theta^i}}{\mathbf{b}_\theta^i!} \tag{4.6}$$

for all projections $\theta = 1, ..., N_\theta$, where N_p represents the total number of pixels in the de-

tector. Therefore, in order to find the volume that has the highest probability of having produced the observed projection data, we need to find the vector $\mathbf{w}_{:,2}$ that maximizes Equation (4.6). In other words, we want the maximizer of this likelihood function because it will give us the vector of glandular fractions $\mathbf{w}_{:,2}$ that most likely produced the projections $\mathbf{b}_\theta$ measured by the detector. This vector will be the reconstructed breast in digital tomosynthesis.

4.3 The Iterative Reconstruction Framework

During the reconstruction process we are trying to find the vector of glandular fractions $\mathbf{w}_{:,2}$ that maximizes the likelihood function in Equation (4.6). However, in numerical optimization it is more common to find the minimizer of a function so we transform the likelihood function into a *negative log-likelihood function* and find its minimum. Finding the vector $\mathbf{w}_{:,2}$ that maximizes this likelihood function is equivalent to minimizing the log-likelihood function

$$-L(\mathbf{b}_\theta, \mathbf{w}_{:,2}) = -\log(p(\mathbf{b}_\theta, \mathbf{w}_{:,2})).$$

Here we have that

$$-L(\mathbf{b}_\theta, \mathbf{w}_{:,2}) = \sum_{i=1}^{N_p} (\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i) - \mathbf{b}_\theta^i \log(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}_\theta) + c \tag{4.7}$$

where $\mathbf{b}_\theta^i$ is the observed data at pixel i for projection θ, c is a constant, and by the previous section

$$\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i = \sum_{e=1}^{N_e} s_e \exp\left(-[(u_{e,2} - u_{e,1})\mathbf{a}_\theta^{i:} \bullet \mathbf{w}_{:,2} + u_{e,1}\mathbf{a}_\theta^{i:} \bullet \mathbf{1}]\right) + \bar{\boldsymbol{\eta}}^i. \tag{4.8}$$

Using the negative log-likelihood function the reconstruction problem is now a numerical optimization problem formulated as

$$\mathbf{w}_{MLE} = \arg \min_{\mathbf{w}_{:,2}} \left\{ \sum_{\theta=1}^{N_\theta} -L(\mathbf{b}_\theta, \mathbf{w}_{:,2}) \right\}. \tag{4.9}$$

Many iterative methods exist to find $\mathbf{w}_{MLE}$ above, but Chung, Nagy and Sechopoulos proposed using a gradient descent algorithm or a Newton approach as described in Section

1.2. The authors chose these methods because they can handle the nonlinearity of the unknown, the possibility of having a non-convex cost function, and the presence of noise. Also, gradient descent methods or Newton approaches allow the user to incorporate regularization to suppress the noise in the data (see Chapter 1). In the case of a non-convex cost function, Chung, Nagy and Sechopoulos show analytically that the if the matrix A_θ is full rank, and the inequality

$$\mathbf{b}^i - (\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i) \leq \frac{\min_e(u_{e,2} - u_{e,1})}{\max_e(u_{e,2} - u_{e,1})} \left(\frac{\mathbf{b}_\theta^i}{\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i} \right) \bar{\mathbf{b}}_\theta^i$$

is satisfied for all i then their proposed polyenergetic cost function is convex with respect to the glandular fractions $\mathbf{w}_{:,2}$.

4.3.1 Gradient Descent

The gradient descent iteration is given by

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} - \alpha_k \nabla L(\mathbf{b}_\theta, \mathbf{w}_{:,2}^{(k)}) \tag{4.10}$$

where the scalar α_k is the step length for each iteration k and the step direction

$$\nabla L(\mathbf{b}_\theta, \mathbf{w}_{:,2}^{(k)}) = \frac{\partial}{\partial \mathbf{w}_{j,2}} \left(-L(\mathbf{b}_\theta, \mathbf{w}_{:,2}^{(k)}) \right)$$

is the derivative of the cost function evaluated at the current iterate. For the given negative log-likelihood function in Equation (4.7), we can calculate the derivative as

$$\frac{\partial}{\partial \mathbf{w}_{j,2}} \left(-L(\mathbf{b}_\theta, \mathbf{w}_{:,2}^{(k)}) \right) = \sum_{i=1}^{N_p} \left(1 - \frac{\mathbf{b}_\theta^i}{\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i} \right) \frac{\partial \bar{\mathbf{b}}^i}{\partial \mathbf{w}_{j,2}}$$

where $\dfrac{\partial \bar{\mathbf{b}}^i}{\partial \mathbf{w}_{j,2}}$ is the derivative of Equation (4.8)

$$\frac{\partial \bar{\mathbf{b}}_\theta^i}{\partial \mathbf{w}_{j,2}} = -a_\theta^{ij} \sum_{e=1}^{N_e} s_e(u_{e,2} - u_{e,1}) \exp\left(-[(u_{e,2} - u_{e,1})\mathbf{a}^{i:} \bullet \mathbf{w}_{:,2} + u_{e,1}\mathbf{a}^{i:} \bullet \mathbf{1}]\right). \tag{4.11}$$

Using matrix notation, we can write the function $\nabla L(\mathbf{b}_\theta, \mathbf{w}_{:,2}^{(k)})$ by a matrix vector product

$$\nabla L(\mathbf{b}_\theta, \mathbf{w}_{:,2}^{(k)}) = A_\theta^T \mathbf{v}_\theta^{(k)} \tag{4.12}$$

where the matrix A_θ contains all the scalars a_θ^{ij} (called the *raytrace operator*) and each entry i of the vector $\mathbf{v}_\theta^{(k)}$ is given by

$$\mathbf{v}_\theta^{(k)}(i) = \left(\frac{\mathbf{b}_\theta^i}{\overline{\mathbf{b}}_\theta^i + \overline{\boldsymbol{\eta}}^i} - 1 \right) \sum_{e=1}^{N_e} s_e(u_{e,2} - u_{e,1}) \exp(-[(u_{e,2} - u_{e,1})A_\theta \mathbf{w}_{:,2}^{(k)} + u_{e,1}A_\theta \mathbf{1}]).$$

Note that the exponential operator in the equation above is applied entry-wise. The product of the transpose raytracing matrix A_θ^T with a vector is called *backprojection* and it is the discrete form of a type of the backprojection reconstruction described in section 3.4.

4.3.2 Newton Approach

Chung, Nagy and Sechopoulos also proposed a Newton type approach to find $\mathbf{w}_{MLE}$ in Equation (4.9). In numerical optimization Newton type methods, introduced in section 1.2, are known to converge much faster to the solution than gradient descent methods. However, their implementation is expensive in terms of memory and computation, since we must construct the Hessian of the cost function (or its approximation) and an inner linear solver is required to find the step direction. In addition, Newton methods can be more sensitive to noise in the data. In their paper Chung et al show that the Hessian of the cost function for the proposed polyenergetic model can be approximated by a product of the raytrace matrix A_θ and its transpose with a diagonal matrix. In addition, they overcome the sensitivity to noise in the data by employing regularization on the form of terminating

the iterations early (see Section 1.2). A Newton iteration takes the form

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \alpha_k \mathbf{s}^{(k)} \tag{4.13}$$

where the scalar α_k is the step length for each iteration k and the step direction is given by the vector $\mathbf{s}^{(k)}$. In this Newton approach the step direction is found as the solution to the linear system

$$\mathbf{H}_\theta(\mathbf{w}^{(k)})\mathbf{s}^{(k)} = -\nabla L(\mathbf{b}_\theta, \mathbf{w}_{:,2}^{(k)}) \tag{4.14}$$

where $\mathbf{H}_\theta(\mathbf{w}^{(k)})$ is the Hessian matrix of the cost function, and $\nabla L(\mathbf{b}_\theta, \mathbf{w}_{:,2}^{(k)})$ the gradient of the cost function, both evaluated at the k-th iterate. Therefore, each entry of the matrix $\mathbf{H}_\theta(\mathbf{w}^{(k)})$ is given by

$$h_\theta^{j\ell} = \frac{\partial}{\partial \mathbf{w}_{\ell,2}} \left(\frac{\partial}{\partial \mathbf{w}_{j,2}} (-L(\mathbf{b}_\theta, \mathbf{w}_{:,2}^{(k)})) \right) = \frac{\partial}{\partial \mathbf{w}_{\ell,2}} \left(\sum_{i=1}^{N_p} \left(1 - \frac{\mathbf{b}_\theta^i}{\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i} \right) \frac{\partial \bar{\mathbf{b}}_\theta^i}{\partial \mathbf{w}_{j,2}} \right)$$

which can be simplified to

$$h_\theta^{j\ell} = \sum_{i=1}^{N_p} \left\{ \left(1 - \frac{\mathbf{b}_\theta^i}{\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i} \right) \frac{\partial}{\partial \mathbf{w}_{\ell,2}} \left(\frac{\partial \bar{\mathbf{b}}_\theta^i}{\partial \mathbf{w}_{j,2}} \right) + \frac{\partial}{\partial \mathbf{w}_{\ell,2}} \left(1 - \frac{\mathbf{b}_\theta^i}{\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i} \right) \frac{\partial \bar{\mathbf{b}}_\theta^i}{\partial \mathbf{w}_{j,2}} \right\}. \tag{4.15}$$

The first term can be expanded using the second derivative $\frac{\partial}{\partial \mathbf{w}_{\ell,2}} \left(\frac{\partial \bar{\mathbf{b}}_\theta^i}{\partial \mathbf{w}_{j,2}} \right)$ of Equation (4.11) which is

$$\frac{\partial}{\partial \mathbf{w}_{\ell,2}} \left(\frac{\partial \bar{\mathbf{b}}_\theta^i}{\partial \mathbf{w}_{j,2}} \right) = a_\theta^{i\ell} a_\theta^{ij} \sum_{e=1}^{N_e} s_e (u_{e,2} - u_{e,1})^2 \exp(-[(u_{e,2} - u_{e,1}) \mathbf{a}_\theta^{i:} \cdot \mathbf{w}_{:,2} + u_{e,1} \mathbf{a}_\theta^{i:} \cdot \mathbf{1}]). \tag{4.16}$$

The second term can be expanded using Equation (4.11) to obtain

$$\frac{\partial \bar{\mathbf{b}}_\theta^i}{\partial \mathbf{w}_{j,2}} \frac{\partial}{\partial \mathbf{w}_{\ell,2}} \left(1 - \frac{\mathbf{b}_\theta^i}{\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i} \right) = \frac{\partial \bar{\mathbf{b}}_\theta^i}{\partial \mathbf{w}_{j,2}} \left\{ -\mathbf{b}_\theta^i \frac{\partial}{\partial \mathbf{w}_{\ell,2}} \left(\frac{1}{\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i} \right) \right\}$$

$$= \frac{\partial \bar{\mathbf{b}}_\theta^i}{\partial \mathbf{w}_{j,2}} \left\{ \frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)^2} \frac{\partial \bar{\mathbf{b}}^i}{\partial \mathbf{w}_{\ell,2}} \right\}$$

$$= \frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)^2} \frac{\partial \bar{\mathbf{b}}_\theta^i}{\partial \mathbf{w}_{j,2}} \frac{\partial \bar{\mathbf{b}}_\theta^i}{\partial \mathbf{w}_{\ell,2}}$$

which is equivalent to

$$\frac{a_\theta^{ij} a_\theta^{i\ell} \mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)^2} \left(\sum_{e=1}^{N_e} s_e (u_{e,2} - u_{e,1}) \exp(-[(u_{e,2} - u_{e,1}) \mathbf{a}^{i:} \bullet \mathbf{w}_{:,2} + u_{e,1} \mathbf{a}^{i:} \bullet \mathbf{1}]) \right)^2 . \tag{4.17}$$

By Equation (4.16) and Equation (4.17), the (j, ℓ) entry of the Hessian $\mathbf{H}_\theta(\mathbf{w}^{(k)})$ of the cost function evaluated at $\mathbf{w}^{(k)}$ is

$$\begin{aligned} h_\theta^{j\ell} = & \sum_{i=1}^{N_p} a_\theta^{i\ell} a_\theta^{ij} \left(1 - \frac{\mathbf{b}_\theta^i}{\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i} \right) \sum_{e=1}^{N_e} s_e (u_{e,2} - u_{e,1})^2 \exp(-[(u_{e,2} - u_{e,1}) \mathbf{a}_\theta^{i:} \cdot \mathbf{w}_{:,2}^{(k)} + u_{e,1} \mathbf{a}_\theta^{i:} \cdot \mathbf{1}]) \\ & + a_\theta^{i\ell} a_\theta^{ij} \frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)^2} \left(\sum_{e=1}^{N_e} s_e (u_{e,2} - u_{e,1}) \exp(-[(u_{e,2} - u_{e,1}) \mathbf{a}_\theta^{i:} \cdot \mathbf{w}_{:,2}^{(k)} + u_{e,1} \mathbf{a}_\theta^{i:} \cdot \mathbf{1}]) \right)^2 \end{aligned}$$

To simplify notation, consider writing each entry of $\mathbf{H}_\theta(\mathbf{w}^{(k)})$ as the sum

$$h_\theta^{j\ell} = \sum_{i=1}^{N_p} a_\theta^{i\ell} a_\theta^{ij} \boldsymbol{\omega}_\theta^j \tag{4.18}$$

with the a_θ^{ij} being the entries of the raytracing matrix A_θ and the jth entry of the vector $\boldsymbol{\omega}$ as

$$\begin{aligned} \boldsymbol{\omega}_\theta^{(k)}(j) = & \left(1 - \frac{\mathbf{b}_\theta^i}{\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i} \right) \sum_{e=1}^{N_e} s_e (u_{e,2} - u_{e,1})^2 \exp(-[(u_{e,2} - u_{e,1}) \mathbf{a}^{i:} \bullet \mathbf{w}_{:,2}^{(k)} + u_{e,1} \mathbf{a}^{i:} \bullet \mathbf{1}]) \\ & + \frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)^2} \left(\sum_{e=1}^{N_e} s_e (u_{e,2} - u_{e,1}) \exp(-[(u_{e,2} - u_{e,1}) \mathbf{a}^{i:} \bullet \mathbf{w}_{:,2}^{(k)} + u_{e,1} \mathbf{a}^{i:} \bullet \mathbf{1}]) \right)^2 . \end{aligned}$$

This leads to expressing the Hessian matrix of the cost function by

$$\mathbf{H}_\theta(\mathbf{w}^{(k)}) = A_\theta^T \Omega_\theta^{(k)} A_\theta \tag{4.19}$$

where $\Omega_\theta^{(k)}$ is a diagonal matrix whose entries are the entries in vector $\boldsymbol{\omega}$. By expressing the Hessian matrix $\mathbf{H}_\theta(\mathbf{w}^{(k)})$ in this form, we can compute the step direction $\mathbf{s}_k$ for the Newton method in Equation (4.14) by solving the linear system

$$A_\theta^T \Omega_\theta^{(k)} A_\theta \mathbf{s}_k = -\nabla L(\mathbf{b}_\theta, \mathbf{w}_{:,2}^{(k)}). \tag{4.20}$$

which is the normal equation formulation of the least squares problem

$$\min_{\mathbf{s}_k} ||W_k^{\frac{1}{2}} A_\theta \mathbf{s}^{(k)} - W_k^{-\frac{1}{2}} \mathbf{v}^{(k)} k||_2^2 \tag{4.21}$$

with $W_k = \Omega_\theta^{(k)}$ and $\mathbf{v}^{(k)} = \nabla L(\mathbf{b}_\theta, \mathbf{w}_{:,2}^{(k)})$. If the entries of matrix W_k are positive, this linear system can be solved using a conjugate gradient least squares method at each iteration.

4.4 Summary

In this chapter we have introduced the basic reconstruction framework for the polyenergetic digital tomosynthesis problem proposed by Chung, Nagy and Sechopoulos in [32]. By incorporating the polyenergetic nature of the x-ray beam directly into the forward problem, this approach produces reconstructions that significantly reduce the presence of beam artifacts and provide a physically meaningful measure of the attenuation of the volume. The aim of this thesis is to expand this model by developing a functional implementation of the reconstruction framework that works for clinical data and incorporates additional arbitrary materials in the model of the object. We would like to use of this polyenergetic reconstruction framework in a clinical setting for breast cancer diagnosis and extend the polyenergetic digital tomosynthesis reconstruction method to other objects and parts of the human body. In the next three chapters we describe an improvement to this linear model for attenuation that accounts for the presence of more than two materials inside the breast.

Chapter 5

The Quadratic Polyenergetic Model for Attenuation

In the previous chapter we described a linear model for attenuation in digital tomosynthesis in breast imaging that uses glandular fractions to reconstruct the unknown volume. The forward model in this reconstruction framework assumes the breast is composed of only glandular and adipose tissue and explicitly accounts for one while it implicitly accounts for the other. This representation of the breast falls short in a clinical setting, because in reality, the breast is composed of more materials like skin, micro-calcifications, and air. In order to create a physically accurate representation of the breast we need to modify the linear model for attenuation to include the attenuating properties of additional materials that can not be accounted by the percent glandular fraction.

In this chapter we present a forward model for the digital tomosynthesis reconstruction problem that accounts for the presence of air and micro-calcifications in the imaged breast in addition to glandular and adipose tissue. We do this by developing a quadratic fit for the attenuation function that expands linear attenuation model from the previous chapter. This new forward model allows us to generate digital tomosynthesis reconstructions of the complete region imaged by the detector, not just the inside of the breast. This way we can use the quadratic model for attenuation to reconstruct clinical data sets. The work in the theory and implementation of the model described in this chapter was done in collaboration

with Dr. Julianne Chung at Virginia Tech.

5.1 Building the Forward Model

5.1.1 Finding a Quadratic Fit

From Equation (4.2) in the previous chapter, we have that for a given energy level e, the attenuation of voxel j is given as the linear function

$$\boldsymbol{\mu}_{e,j} = [u_{e,2} - u_{e,1}]\mathbf{w}_{j,2} + u_{e,1}$$

where $u_{e,1}$ and $u_{e,2}$ are the known linear attenuation coefficients of adipose and glandular tissue respectively and the value $\mathbf{w}_{j,2}$ is the glandular fraction (percentage of glandular tissue) inside voxel j. In reality, the imaged breast may contain micro-calcifications, which are calcium deposits inside the breast tissue. These features are important to include in our model, as their patterns can sometime signal breast cancer [85]. In addition, we would like to account for the presence of air surrounding the imaged breast.

To expand the attenuation function, we would like to build a quadratic representation of the attenuation function $\boldsymbol{\mu}_{e,j}$ that is almost linear in some range $[\hat{x}, \hat{x} + 100]$, where $\mathbf{x}_j = \hat{x}$ represents a voxel is only composed of adipose tissue and $\mathbf{x}_j = \hat{x} + 100$ represents a voxel that is composed of 100% glandular tissue. The attenuation for voxel j at energy level e can be represented as the function

$$\mu(e, \mathbf{x}_j) = \alpha_e \mathbf{x}_j^2 + \beta_e \mathbf{x}_j + \gamma_e.$$

Our goal is to find the constants α_e, β_e, and γ_e that allow us to satisfy the following criteria:

- If $\mathbf{x}_j = 0$ we want the voxel attenuation to be equal to the attenuation of air, that is:

$$\mu(e, 0) = \gamma(e) = \mu_{air}(e).$$

- If the voxel is composed of all adipose tissue, then

$$\mu(e, \hat{x}) = \alpha_e \hat{x}^2 + \beta_e \hat{x} + \gamma_e$$

which according to the linear model is

$$\mu(e, \hat{x}) = \alpha_e \hat{x}^2 + \beta_e \hat{x} + \gamma_e = [u_{e,2} - u_{e,1}] \cdot 0 + u_{e,1} = u_{e,1}.$$

- If the voxel is composed of all glandular tissue, then the attenuation of the voxel is

$$\mu(e, \hat{x} + 100) = \alpha_e [\hat{x} + 100]^2 + \beta_e [\hat{x} + 100] + \gamma_e$$

which according to the linear model is

$$\mu(e, \hat{x} + 100) = \alpha_e [\hat{x} + 100]^2 + \beta_e [\hat{x} + 100] + \gamma_e = [u_{e,2} - u_{e,1}] \cdot 1 + u_{e,1} = u_{e,2}.$$

- We want the quadratic model to be linear in the range between points $(\hat{x}, \mu(e, \hat{x}))$ and $(\hat{x} + 100, \mu(e, \hat{x} + 100))$. This means that at the midpoint of this interval, given by $(\hat{x} + 50, \mu(e, \hat{x} + 50))$, the derivative is constant and equal to the derivative of the linear model. That is,

$$\mu'(e, \hat{x} + 50) = 2\alpha_e [\hat{x} + 50] + \beta_e = u_{e,2} - u_{e,1}.$$

These conditions introduce a linear system for each energy level e given by

$$\begin{bmatrix} 0 & 0 & 1 \\ \hat{x}^2 & \hat{x} & 1 \\ (\hat{x} + 100)^2 & \hat{x} + 100 & 1 \\ 2(\hat{x} + 50) & 1 & 0 \end{bmatrix} \begin{bmatrix} \alpha_e \\ \beta_e \\ \gamma_e \end{bmatrix} = \begin{bmatrix} \mu_{air} \\ u_{e,1} \\ u_{e,2} \\ u_{e,2} - u_{e,1} \end{bmatrix}. \tag{5.1}$$

We can solve for α_e, β_e, and γ_e to obtain the attenuation function at each energy level e.

5.1.2 Finding an Improved Quadratic Fit

The quadratic fit developed in the previous section is a good approach to include air and micro-calcifications in the attenuation model, but the polynomial developed can be too general and not approximate the linear model sufficiently well. In this section, we build on the principles of the previous quadratic fit and include additional constraints to improve the model. We want to optimize our choice of nodes to create the polynomial fit to have a better handle on how close the quadratic polynomial resembles the original linear model for attenuation.

The idea in extending the linear attenuation model to a quadratic attenuation model is to express the attenuation of voxel j for energy level e as a polynomial of the form

$$\boldsymbol{\mu}_{e,j} = \alpha_e \mathbf{x}_j^2 + \beta_e \mathbf{x}_j + \gamma_e \tag{5.2}$$

where the variable $\mathbf{x}_j$ is now a transformation of the percent glandular fraction value $\mathbf{w}_j$ of the voxel. In this case, we let the value $\mathbf{w}_j = 1$ (which means the voxel is composed of 100% glandular tissue) be equivalent to $\mathbf{x}_j = x_{gland}$. Here, the value x_{gland} represents the value of 100% glandular tissue in a new range, as opposed to the previous fit where $x_{gland} = \hat{x} + 100$.

Assume that we know the values in this new range for $x_{air}, x_{calc}, x_{adip}$, and x_{gland}. Then, using the quadratic fit of Equation (5.2) , the value reconstructed for each voxel j in the volume is inside the interval $\mathbf{x}_j \in [x_{air}, x_{calc}]$, noting that the special values of x_{adip}, and x_{gland} fall within this interval. The conditions we will use to develop this new quadratic fit are as follows:

- The quadratic polynomial is positive inside the interval $\mathbf{x}_j \in [x_{air}, x_{calc}]$ in order to guarantee monotonicity.

- If the voxel in question is composed of air, then we have the condition that the quadratic polynomial in Equation (5.2) at x_{air} is equal to the value of the linear attenuation of air. That is, if j_{air} is the index of a voxel composed of 100% air we

have

$$\mu(e, j_{air}) = \alpha_e x_{air}^2 + \beta_e x_{air} + \gamma_e = \mu_{air}(e).$$

- If the voxel in question is composed of adipose tissue, then we want to have the quadratic polynomial at the voxel be equal to the attenuation of adipose tissue. That is, if j_{adip} is the index of a voxel composed of 100% adipose tissue we have

$$\mu(e, j_{adip}) = \alpha_e x_{adip}^2 + \beta_e x_{adip} + \gamma_e = \mu_{adipose}(e).$$

  By the linear model, we have $\mu_{adipose}(e) = u_{e,1}$.

- Similarly, if the voxel in question is composed of glandular tissue and j_{gland} is it's voxel index, then

$$\mu(e, j_{gland}) = \alpha_e x_{gland}^2 + \beta_e x_{gland} + \gamma_e = \mu_{glandular}(e).$$

  In this case, if the voxel is composed of 100% glandular tissue, then by the linear model for attenuation we have

$$\mu_{glandular}(e) = [u_{e,2} - u_{e,1}] \cdot 1 + u_{e,1} = u_{e,2}.$$

Putting these conditions together we have nonlinear separable optimization problem, where we want to find the values of x_{air}, x_{adip}, and x_{gland} and the values $\alpha(e), \beta(e), \gamma(e)$ for all energies $e = 1 : N_e$.

To set up the nonlinear separable inverse problem, begin by constructing the vector of unknowns

$$\bar{x} = \begin{bmatrix} x_{air} \\ x_{adip} \\ x_{gland} \end{bmatrix},$$

which are the same for all energy levels e, and constructing the matrix of unknown fit

coefficients

$$\mathbf{F} = \begin{bmatrix} \alpha_1 & \alpha_2 & ... & \alpha_{N_e} \\ \beta_1 & \beta_2 & ... & \beta_{N_e} \\ \gamma_1 & \gamma_2 & ... & \gamma_{N_e} \end{bmatrix},$$

with each column being the set of fit coefficients for each energy level. The known data for each energy level can be organized as columns of a matrix, like

$$\mathbf{B} = \begin{bmatrix} \mu_{air}(e_1) & \mu_{air}(e_2) & ... & \mu_{air}(e_{N_e}) \\ u_{1,1} & u_{2,1} & ... & u_{N_e,1} \\ u_{1,2} & u_{2,2} & ... & u_{N_e,2} \end{bmatrix}.$$

If we define the Vandermonde matrix $\mathbf{X}$ as

$$\mathbf{X}(\bar{x}) = \begin{bmatrix} x_{air}^2 & x_{air} & 1 \\ x_{adip}^2 & x_{adip} & 1 \\ x_{gland}^2 & x_{gland} & 1 \end{bmatrix},$$

then for each energy e we have the linear system

$$\mathbf{X}(\bar{x})\mathbf{F}(:, e) = \mathbf{B}(:, e) \tag{5.3}$$

which is a nonlinear separable inverse problem (see Section 1.2).

To develop the quadratic fit, we need to solve the linear system in Equation (5.3) for all energies e. As described in Section 1.2, we can solve this nonlinear separable inverse problem by solving the least squares problem given by

$$\min_{\bar{x},\mathbf{F}} ||\mathbf{B} - \mathbf{X}(\bar{x})\mathbf{F}||_2^2$$

where $\bar{x}$ and $\mathbf{F}$ are our unknown. If $\mathbf{B} = \mathbf{X}(\bar{x})\mathbf{F}$, then we can write $\mathbf{F} = \mathbf{X}^{-1}(\bar{x})\mathbf{B}$ and solve

the equivalent optimization problem

$$\begin{aligned} \min_{\bar{x}} \left|\left|\mathbf{B} - \mathbf{X}(\bar{x})\mathbf{X}^{-1}(\bar{x})\mathbf{B}\right|\right|_2^2 \\ \text{s.t. } C(\bar{x})\mathbf{F} > 0 \end{aligned} \tag{5.4}$$

where the constrain matrix is $C(\bar{x})$ guarantees that the quadratic polynomial and its first derivative are positive. The matrix $C(\bar{x})$ given as

$$C(\bar{x}) = \begin{bmatrix} x_{air}^2 & x_{air} & 1 \\ x_{adip}^2 & x_{adip} & 1 \\ x_{gland}^2 & x_{gland} & 1 \\ 2x_{air} & 1 & 0 \\ 2x_{adip} & 1 & 0 \\ 2x_{gland} & 1 & 0 \end{bmatrix}.$$

Once the solution $\bar{x}$ is obtained from the least squares problem, the values for matrix $\mathbf{F}$ are recovered by $\mathbf{F} = \mathbf{X}^{-1}(\bar{x})\mathbf{B}$ where the inverse of the Vandermonde matrix $\mathbf{X}$ is

$$\mathbf{X}^{-1}(\bar{x}) = \begin{bmatrix} \frac{1}{(x_{air}-x_{adip})(x_{air}-x_{gland})} & \frac{-1}{(x_{air}-x_{adip})(x_{adip}-x_{gland})} & \frac{1}{(x_{air}-x_{gland})(x_{adip}-x_{gland})} \\ \frac{-(x_{adip}+x_{gland})}{(x_{air}-x_{adip})(x_{air}-x_{gland})} & \frac{x_{air}+x_{gland}}{(x_{air}-x_{adip})(x_{adip}-x_{gland})} & \frac{x_{air}+x_{adip}}{(x_{air}-x_{gland})(x_{gland}-x_{adip})} \\ \frac{x_{adip}x_{gland}}{(x_{air}-x_{adip})(x_{air}-x_{gland})} & \frac{-x_{adip}x_{gland}}{(x_{air}-x_{adip})(x_{adip}-x_{gland})} & \frac{x_{air}x_{adip}}{(x_{air}-x_{gland})(x_{adip}-x_{gland})} \end{bmatrix}.$$

Finally, to determine the value for x_{calc} we use the quadratic formula to obtain

$$\begin{aligned} \mu_{calc}(e) &= \alpha_e[x_{calc}(e)]^2 + \beta_e x_{calc}(e) + \gamma_e \\ x_{calc}(e) &= \frac{-\beta_e \pm \sqrt{\beta_e^2 - 4\alpha_e(\gamma_e - \mu_{calc}(e))}}{2\alpha_e} \\ x_{calc} &= \text{mean}_e x_{calc}(e). \end{aligned}$$

Note that under this quadratic fit, the object will be reconstructed in terms of the variable $\mathbf{x}_j$ which is a transformation of the glandular fractions $\mathbf{w}_{j,2} \in [0, 1]$. To convert from the new units of the reconstructed volume to the original glandular fractions we use

the transformation

$$\mathbf{w}_j = \frac{x_j - x_{adip}}{x_{gland} - x_{adip}}. \tag{5.5}$$

5.1.3 The Forward Projection Model

The forward model using a quadratic attenuation function differs slightly from the linear model described in the previous chapter. Mainly, in the quadratic case we are reconstructing an unknown vector $\mathbf{x}$ whose entries are the linear transformation of the percent glandular fractions $\mathbf{w}_{:,2}$ described in Equation (5.5). Hence, in the quadratic model reconstruction framework, we have an additional step to recover the values $\mathbf{w}_{:,2}$ from the solution $\mathbf{x}$.

To begin building the forward model we use the discrete formulation of Beer's law for a polyenergetic x-ray given in the previous chapter by

$$\mathbf{b}^i = \sum_{e=1}^{N_e} s_e \exp\left[-\sum_{j=1}^{N_v} a^{ij}\boldsymbol{\mu}_{e,j}\right].$$

As before, in this equation N_e is the number of discrete levels of energy in the x-ray beam, N_v is the number of voxels in the object, $\boldsymbol{\mu}_{e,j}$ is the attenuation of voxel j for energy level e, a^{ij} represents the length of the x-ray beam traveling through voxel j, and s_e is the value of the spectral distribution function at energy e. Using Equation (5.2) we have that the the monochromatic ray for pixel i at incident angle θ is

$$\begin{aligned}\sum_{j=1}^{N_v}\boldsymbol{\mu}_{e,j}a_\theta^{ij} =& \sum_{j=1}^{N_v}[\alpha_e\mathbf{x}_j^2 + \beta_e\mathbf{x}_j + \gamma_e]a_\theta^{ij}\\ =&\alpha_e\sum_{j=1}^{N_v}a_\theta^{ij}\mathbf{x}_j^2 + \beta_e\sum_{j=1}^{N_v}a_\theta^{ij}\mathbf{x}_j + \gamma_e\sum_{j=1}^{N_v}a_\theta^{ij}\\ =&\alpha_e\mathbf{a}_\theta^{i:}\bullet\mathbf{x}^2 + \beta_e\mathbf{a}_\theta^{i:}\bullet\mathbf{x} + \gamma_e\mathbf{a}_\theta^{i:}\bullet\mathbf{1}\end{aligned}$$

where $\mathbf{a}_\theta^{i:}$ is the ith row of the raytrace matrix A_θ, $\mathbf{1}$ is a column vector of ones of length N_v, and the fit coefficients $\alpha_e, \beta_e, \gamma_e$ are determined by the process outlined in the previous section. Note that the exponent $\mathbf{x}^2$ is applied entry-wise. Applying Beer's law gives the

value of pixel i for the projection taken at angle θ as

$$\mathbf{b}_\theta^i = \sum_{e=1}^{N_e} s_e \exp\left(-[\alpha_e \mathbf{a}_\theta^{i:} \bullet \mathbf{x}^2 + \beta_e \mathbf{a}_\theta^{i:} \bullet \mathbf{x} + \gamma_e \mathbf{a}_\theta^{i:} \bullet \mathbf{1}]\right)$$

where the exponential operation is applied entry-wise. Using matrix notation the entire projection acquired at incident angle θ is given by

$$\mathbf{b}_\theta = \sum_{e=1}^{N_e} s_e \exp\left(-[\alpha_e A_\theta \mathbf{x}^2 + \beta_e A_\theta \mathbf{x} + \gamma_e A_\theta \mathbf{1}]\right). \tag{5.6}$$

Equation (5.6) is the forward model for incident angle θ in the digital tomosynthesis reconstruction problem using the quadratic model for attenuation.

5.2 Statistical Considerations

Similar to the reconstruction model for linear attenuation, we model the variable representing the energy measured by each pixel in the detector as a Poisson random variable with additional noise. This is standard practice in the imaging community where modeling the measurements obtained by x-ray transmission as independently distributed Poisson random variables with additive noise. Refer to section 3.1 to see how the source-detector interaction in digital tomosynthesis can be modeled as a Bernoulli random variable (the detector) receiving a Poisson random variable as "input" (the source). Hence, given the forward model in Equation (5.6), the expected value of the measured data for pixel i at angle θ is

$$E(\mathbf{b}_\theta^i, \mathbf{x}) = \sum_{e=1}^{N_e} s_e \exp\left[-\left(\alpha_e \mathbf{a}_\theta^{i:} \bullet \mathbf{x}^2 + \beta_e \mathbf{a}_\theta^{i:} \bullet \mathbf{x} + \gamma_e \mathbf{a}_\theta^{i:} \bullet \mathbf{1}\right)\right] + \bar{\boldsymbol{\eta}}^i$$

where $\mathbf{a}_\theta^{i:}$ is the i-th row of A_θ and $\bar{\boldsymbol{\eta}}^i$ represents the additive noise. This noise is generally due to scatter and is also assumed to follow a Poisson distribution whose statistical mean is known or can be approximated. This means that each pixel of the obtained data is a realization of the Poisson random variable

$$\mathbf{b}_\theta^i \sim \text{Poisson}(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)$$

Using the theory of maximum likelihood estimation, we have that the probability of observing image $\mathbf{b}_\theta$ for all θ given the volume $\mathbf{x}$ where N_p represents the total number of pixels in the detector is determined by the likelihood function

$$p(\mathbf{b}_\theta, \mathbf{x}) = \prod_{i=1}^{N_p} \frac{e^{-(\overline{\mathbf{b}}_\theta + \overline{\boldsymbol{\eta}}^i)} (\overline{\mathbf{b}}_\theta + \overline{\boldsymbol{\eta}}^i)^{\mathbf{b}_\theta^i}}{\mathbf{b}_\theta^i!}. \tag{5.7}$$

Maximizing the likelihood function above, will result in the value of $\mathbf{x}$ that has the highest probability of having produced projection $\mathbf{b}_\theta$. As in the previous chapter, the goal of the reconstruction process is to find the value $\mathbf{x}$ that maximizes Equation (5.7).

5.3 The Iterative Reconstruction Framework

The goal of the reconstruction process is to iteratively solve for the vector of $\mathbf{x}$ that maximizes the likelihood function in Equation (5.7). This vector is our reconstructed volume in terms of a transformation of the percent glandular fractions. However, in numerical optimization it is more common to find the minimizer rather than the maximizer of a function. Therefore, we can transform the problem of maximizing Equation (5.7) is to minimizing the negative log-likelihood function

$$-L(\mathbf{b}_\theta, \mathbf{x}) = -\log(p(\mathbf{b}_\theta, \mathbf{x})).$$

Here we have that the function

$$-L(\mathbf{b}_\theta, \mathbf{x}) = \sum_{i=1}^{N_p} (\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i) - \mathbf{b}_\theta^i \log(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i) + c \tag{5.8}$$

where $\mathbf{b}_\theta$ is the observed data for projection θ, the scalar c is a constant, and $\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i$ is the expected value of the pixel i in the projection given in the previous section as

$$\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i = \sum_{e=1}^{N_e} s_e \exp\left[-\left(\alpha_e \mathbf{a}_\theta^{i:} \bullet \mathbf{x}^2 + \beta_e \mathbf{a}_\theta^{i:} \bullet \mathbf{x} + \gamma_e \mathbf{a}_\theta^{i:} \bullet \mathbf{1}\right)\right] + \bar{\boldsymbol{\eta}}^i. \tag{5.9}$$

The reconstruction problem for digital tomosynthesis can now be expressed as the numerical optimization problem

$$\mathbf{x}_{MLE} = \arg \min_{\mathbf{x}} \left\{ \sum_{\theta=1}^{N_\theta} -L(\mathbf{b}_\theta, \mathbf{x}) \right\}. \tag{5.10}$$

To solve the optimization problem above, we will follow the same approach that Chung, Nagy, and Sechopoulos proposed in [32] to reconstruct the unknown volume using the the linear model for attenuation. We will describe a gradient descent algorithm and a Newton approach with a line search to solve this problem since these methods can deal with the nonlinearity of the function, the possibility of having a non-convex cost function, and the presence of noise. Using a gradient descent method or a Newton approach also allows for incorporating regularization to limit the effect of noise in the data to the solution.

5.3.1 Gradient Descent

The gradient descent iteration represents a line search method that moves along the direction of the negative derivative at every step. Choosing the descent direction in this method to be the negative derivative of the function is natural, since it is the direction of which the function decreases most rapidly. To show this, using Taylor's theorem we have the function expansion of the cost function $L(\mathbf{b}_\theta, \mathbf{x}^{(k)} + \alpha \mathbf{s}^{(k)})$ for any vector $\mathbf{s}^{(k)}$ and parameter α is

$$L(\mathbf{b}_\theta, \mathbf{x}^{(k)} + \alpha \mathbf{s}^{(k)}) = L(\mathbf{b}_\theta, \mathbf{x}^{(k)}) + \alpha [\mathbf{s}^{(k)}]^T \nabla L(\mathbf{b}_\theta, \mathbf{x}^{(k)}) + \mathcal{O}(\alpha^2).$$

When the direction $\mathbf{s}^{(k)}$ is a descent direction, the angle between the vector $\mathbf{s}^{(k)}$ and the gradient $\nabla L(\mathbf{b}_\theta, \mathbf{x}^{(k)})$ (denoted $\Theta^{(k)}$) has a negative value of cosine. Therefore

$$[\mathbf{s}^{(k)}]^T \nabla L(\mathbf{b}_\theta, \mathbf{x}^{(k)}) = ||[\mathbf{s}^{(k)}|| \; ||\nabla L(\mathbf{b}_\theta, \mathbf{x}^{(k)})|| \cos \Theta^{(k)} < 0$$

and thus we have that

$$L(\mathbf{b}_\theta, \mathbf{x}^{(k)} + \alpha \mathbf{s}^{(k)}) < L(\mathbf{b}_\theta, \mathbf{x}^{(k)}).$$

If we let $\mathbf{s}^{(k)} = \nabla L(\mathbf{b}_\theta, \mathbf{x}^{(k)})$, then $\cos(\Theta^{(k)}) = -1$ is the minimum it can be, and thus we have the most decrease (see [90, 37] for further details on the gradient descent method). We

choose to solve the problem in Equation (5.10) using a gradient descent method because of the size of the problem and the computational intensity of higher derivatives of the cost function $L(\mathbf{b}_\theta, \mathbf{x}^{(k)})$ (see section 5.3.2) .

The gradient descent iteration has the form

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \alpha_k \nabla L(\mathbf{b}_\theta, \mathbf{x}^{(k)}) \tag{5.11}$$

where α_k is the step length for each iteration k and the step direction

$$\nabla L(\mathbf{b}_\theta, \mathbf{x}^{(k)}) = \frac{\partial}{\partial \mathbf{x}_j}\left(-L(\mathbf{b}_\theta, \mathbf{x}^{(k)})\right)$$

is the derivative of the cost function evaluated at the current iterate. For the given negative log-likelihood function in Equation (5.10), we can calculate the derivative as

$$\begin{aligned}\frac{\partial}{\partial \mathbf{x}_j}\left(-L(\mathbf{b}_\theta, \mathbf{x}^{(k)})\right) =& \frac{\partial}{\partial \mathbf{x}_j}\left(\sum_{i=1}^{N_p}(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i) - \mathbf{b}_\theta^i \log(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i) + c\right) \\ =& \sum_{i=1}^{N_p}\left(1 - \frac{\mathbf{b}_\theta^i}{\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i}\right)\frac{\partial(\bar{\mathbf{b}}^i + \bar{\boldsymbol{\eta}}^i)}{\partial \mathbf{x}_j}\end{aligned}$$

where $\dfrac{\partial(\bar{\mathbf{b}}^i + \bar{\boldsymbol{\eta}}^i)}{\partial \mathbf{x}_j}$ is the derivative of the expected value in Equation (5.9), given by

$$\frac{\partial \bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i}{\partial \mathbf{x}_j} = -a_\theta^{ij}\sum_{e=1}^{N_e} s_e[2\alpha_e \mathbf{x}_j + \beta_e]\exp\left(-(\alpha_e \mathbf{a}_\theta^{i:} \bullet \mathbf{x}^2 + \beta_e \mathbf{a}_\theta^{i:} \bullet \mathbf{x} + \gamma_e \mathbf{a}_\theta^{i:} \bullet \mathbf{1})\right). \tag{5.12}$$

Using matrix notation we can express the gradient $\nabla L(\mathbf{b}_\theta, \mathbf{x}^{(k)})$ as

$$\nabla L(\mathbf{b}_\theta, \mathbf{x}^{(k)}) = 2\mathbf{x}^{(k)} \cdot \times (A_\theta^T \mathbf{v}^{(k)}) + A_\theta^T \mathbf{y}^{(k)} \tag{5.13}$$

where the matrix A_θ is the raytracing matrix, $\mathbf{x}^{(k)}$ is the current iterate $\mathbf{x}^{(k)}$ with the operation $\cdot\times$ being an entry-wise multipliction, and the vectors $\mathbf{v}^{(k)}, \mathbf{y}^{(k)}$ are given by

$$\mathbf{v}^{(k)}(i) = \left(\frac{\mathbf{b}_\theta^i}{\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i} - 1\right)\sum_{e=1}^{N_e} s_e \alpha_e \exp\left(-\left[\alpha_e \mathbf{a}_\theta^{i:} \bullet [\mathbf{x}^{(k)}]^2 + \beta_e \mathbf{a}_\theta^{i:} \bullet \mathbf{x}^{(k)} + \gamma_e \mathbf{a}_\theta^{i:} \bullet \mathbf{1}\right]\right)$$

$$\mathbf{y}^{(k)}(i) = \left(\frac{\mathbf{b}_\theta^i}{\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i} - 1 \right) \sum_{e=1}^{N_e} s_e \beta_e \exp\left(- \left[\alpha_e \mathbf{a}_\theta^{i:} \bullet [\mathbf{x}^{(k)}]^2 + \beta_e \mathbf{a}_\theta^{i:} \bullet \mathbf{x}^{(k)} + \gamma_e \mathbf{a}_\theta^{i:} \bullet \mathbf{1} \right] \right).$$

To solve Equation (5.10) using a gradient descent method, the framework is as follows

Algorithm 5.3.4 Gradient Descent Algorithm For the Quadratic Attenuation Model

1: Find $\alpha_e, \beta_e, \gamma_e$ as a separable nonlinear inverse problem using the lease squares formulation in Equation (5.4)
2: Choose the initial volume $\mathbf{x}^{(0)}$
3: **for** $k = 1, 2, ...$ until converge **do**
4: Compute the vectors $\mathbf{v}^{(k)}$ and $\mathbf{y}^{(k)}$ for all θ
5: Find the step direction for all θ using $\nabla L(\mathbf{b}_\theta, \mathbf{x}^{(k)}) = 2\mathbf{X}^{(k)} A_\theta^T \mathbf{v}^{(k)} + A_\theta^T \mathbf{y}^{(k)}$
6: Determine the step length α_k
7: Update the solution $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \alpha_k \nabla L(\mathbf{b}_\theta, \mathbf{x}^{(k)})$
8: **if** $\sum_{\theta=1}^{N_\theta} L(\mathbf{b}_\theta, \mathbf{x}^{(k)}) > \sum_{\theta=1}^{N_\theta} L(\mathbf{b}_\theta, \mathbf{x}^{(k-1)})$ **then**
9: Refine parameter α_k; no sufficient descent
10: **end if**
11: **end for**

In this method regularization is enforced by early termination, as later iterates begin to be corrupted by noise (see section 1.1). For digital breast tomosynthesis reconstructions, only a few iterations of gradient descent are needed (less than twenty in most cases). There is currently no automated convergence criteria, convergence is determined once the relative value of the function stagnates. For more details see Chapter 11.

5.3.2 Newton Approach

We are also interested in developing a Newton method for the reconstruction process using the quadratic model for attenuation. Newton methods are sometimes preferred to a gradient descent approach because they are known to converge much faster to the solution. However, implementing a Newton method is expensive in terms of computation and storage because they require building the Hessian of the cost function and inner linear solver to find the step direction. In addition, Newton methods are more sensitive to noise in the data. In this

section we find an analytical representation of the Hessian of the cost function and reduce the computation requirements by expressing it as a product of A_θ, A_θ^T and some simple matrices.

The Newton approach described in this work is a line search method that moves along the Newton direction for every iteration (see [90, 37] for further details). The Newton direction is derived from the quadratic term of the Taylor approximation of the cost function $L(\mathbf{b}_\theta, \mathbf{x}^{(k)})$. If we approximate the cost function $L(\mathbf{b}_\theta^i, \mathbf{x}^{(k)} + \alpha \mathbf{s}^{(k)})$ for any vector $\mathbf{s}^{(k)}$ and parameter α by its Taylor expansion, we have

$$L(\mathbf{b}_\theta, \mathbf{x}^{(k)} + \alpha \mathbf{s}^{(k)}) = L(\mathbf{b}_\theta, \mathbf{x}^{(k)}) + \alpha [\mathbf{s}^{(k)}]^T \nabla L(\mathbf{b}_\theta, \mathbf{x}^{(k)}) + \alpha [\mathbf{s}^{(k)}]^T \nabla^2 L(\mathbf{b}_\theta, \mathbf{x}^{(k)}) \mathbf{s}^{(k)} + \mathcal{O}(\alpha^3).$$

Define the function $m(\mathbf{s}^{(k)}) = L(\mathbf{b}_\theta, \mathbf{x}^{(k)}) + \alpha [\mathbf{s}^{(k)}]^T \nabla L(\mathbf{b}_\theta, \mathbf{x}^{(k)}) + \alpha [\mathbf{s}^{(k)}]^T \nabla^2 L(\mathbf{b}_\theta, \mathbf{x}^{(k)}) \mathbf{s}^{(k)}$. If the term $\nabla^2 L(\mathbf{b}_\theta, \mathbf{x}^{(k)})$ is positive definite, then the vector $\mathbf{s}^{(k)}$ that minimizes $m(\mathbf{s}^{(k)})$ is given by

$$\mathbf{s}^{(k)} = \nabla^2 L^{-1}(\mathbf{b}_\theta, \mathbf{x}^{(k)}) \nabla L(\mathbf{b}_\theta, \mathbf{x}^{(k)})$$

and this is the Newton search direction. If $\nabla^2 L(\mathbf{b}_\theta, \mathbf{x}^{(k)})$ is positive definite we have that $\nabla L(\mathbf{b}_\theta, \mathbf{x}^{(k)})^T \mathbf{s}^{(k)} < 0$ and thus $\mathbf{s}^{(k)}$ is a descent direction where we find that

$$L(\mathbf{b}_\theta, \mathbf{x}^{(k)} + \alpha \mathbf{s}^{(k)}) < L(\mathbf{b}_\theta, \mathbf{x}^{(k)}),$$

or in other words, the cost function decreases along the direction of the vector $\mathbf{s}^{(k)}$.

For our problem, a Newton iteration takes the form

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k \mathbf{s}^{(k)} \tag{5.14}$$

where α_k is the step length and $\mathbf{s}^{(k)}$ is the step direction given by the solution to the linear system

$$\mathbf{H}_\theta(\mathbf{x}^{(k)}) \mathbf{s}^{(k)} = -\nabla L(\mathbf{b}_\theta, \mathbf{x}^{(k)}) \tag{5.15}$$

where $\mathbf{H}_\theta(\mathbf{x}^{(k)})$ is the Hessian matrix of the cost function and $\nabla L(\mathbf{b}_\theta, \mathbf{x}^{(k)})$ is the gradient of the cost function both evaluated at the k-th iterate. Therefore, we can write each entry

(j, ℓ) of the matrix $\mathbf{H}_\theta(\mathbf{x}^{(k)})$ as

$$h_\theta^{j\ell} = \frac{\partial}{\partial \mathbf{x}_\ell}\left(\frac{\partial}{\partial \mathbf{x}_j}(-L(\mathbf{b}_\theta, \mathbf{x}^{(k)}))\right) = \frac{\partial}{\partial \mathbf{x}_j}\left(\sum_{i=1}^{N_p}\left(1 - \frac{\mathbf{b}_\theta^i}{\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i}\right)\frac{\partial \bar{\mathbf{b}}_\theta^i}{\partial \mathbf{x}_j}\right)$$

which expands to the expression

$$h_\theta^{j\ell} = \sum_{i=1}^{N_p}\left\{\left(1 - \frac{\mathbf{b}_\theta^i}{\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i}\right)\frac{\partial}{\partial \mathbf{x}_\ell}\left(\frac{\partial \bar{\mathbf{b}}_\theta^i}{\partial \mathbf{x}_j}\right) + \frac{\partial \bar{\mathbf{b}}_\theta^i}{\partial \mathbf{x}_j}\frac{\partial}{\partial \mathbf{x}_\ell}\left(1 - \frac{\mathbf{b}_\theta^i}{\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i}\right)\right\}. \tag{5.16}$$

In order to expand the first term in Equation (5.16) we need to compute $\dfrac{\partial^2(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}{\partial \mathbf{x}_\ell \partial \mathbf{x}_j}$. Using Equation (5.12) we have

$$\begin{aligned}\frac{\partial^2(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}{\partial \mathbf{x}_\ell \partial \mathbf{x}_j} =& \frac{\partial}{\partial \mathbf{x}_\ell}\left(-a_\theta^{ij}\sum_{e=1}^{N_e} 2 s_e \alpha_e \mathbf{x}_j \exp\left(-\left[\alpha_e \mathbf{a}_\theta^{i:} \cdot [\mathbf{x}^{(k)}]^2 + \beta_e \mathbf{a}_\theta^{i:} \cdot \mathbf{x}^{(k)} + \gamma_e \mathbf{a}_\theta^{i:} \cdot \mathbf{1}\right]\right)\right) \\ &+ \frac{\partial}{\partial \mathbf{x}_\ell}\left(-a_\theta^{ij}\sum_{e=1}^{N_e} s_e \beta_e \exp\left(-\left[\alpha_e \mathbf{a}_\theta^{i:} \bullet [\mathbf{x}^{(k)}]^2 + \beta_e \mathbf{a}_\theta^{i:} \bullet \mathbf{x}^{(k)} + \gamma_e \mathbf{a}_\theta^{i:} \bullet \mathbf{1}\right]\right)\right).\end{aligned}$$

If we let

$$M_{\theta,i}(\mathbf{x}^{(k)}, e) = \alpha_e \mathbf{a}_\theta^{i:} \bullet [\mathbf{x}^{(k)}]^2 + \beta_e \mathbf{a}_\theta^{i:} \bullet \mathbf{x}^{(k)} + \gamma_e \mathbf{a}_\theta^{i:} \bullet \mathbf{1}$$

to simplify notation, then the equation above expands to

$$\begin{aligned}\frac{\partial^2(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}{\partial \mathbf{x}_\ell \partial \mathbf{x}_j} =& 4 a_\theta^{ij} a_\theta^{i\ell} \mathbf{x}_j \mathbf{x}_\ell \sum_{e=1}^{N_e} s_e \alpha_e^2 \exp\left(-M_{\theta,i}(\mathbf{x}^{(k)}, e)\right) \\ &+ 2 a_\theta^{ij} a_\theta^{i\ell}[\mathbf{x}_j + \mathbf{x}_\ell] \sum_{e=1}^{N_e} s_e \alpha_e \beta_e \exp\left(-M_{\theta,i}(\mathbf{x}^{(k)}, e)\right) \\ &+ a_\theta^{ij} a_\theta^{i\ell} \sum_{e=1}^{N_e} s_e \beta_e^2 \exp\left(-M_{\theta,i}(\mathbf{x}^{(k)}, e))\right).\end{aligned} \tag{5.17}$$

To compute the second portion of each entry $h^{j\ell}$ in the Hessian we write the second term in Equation (5.16) as

$$\frac{\partial(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}{\partial \mathbf{x}_j}\frac{\partial}{\partial \mathbf{x}_\ell}\left(1 - \frac{\mathbf{b}_\theta^i}{\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i}\right) = \frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)^2}\frac{\partial(\bar{\mathbf{b}}^i + \bar{\boldsymbol{\eta}})}{\partial \mathbf{x}_j}\frac{\partial(\bar{\mathbf{b}}^i + \bar{\boldsymbol{\eta}}^i)}{\partial \mathbf{x}_\ell}. \tag{5.18}$$

Using Equations (5.12), (5.17), and (5.18) we can then write the (j,ℓ) entry of the Hessian matrix $\mathbf{H}_\theta(\mathbf{x}^{(k)})$ as

$$
\begin{aligned}
h_\theta^{j\ell} = &\sum_{i=1}^{N_p} 4a_\theta^{ij}a_\theta^{i\ell}\mathbf{x}_j\mathbf{x}_\ell\left(1-\frac{\mathbf{b}_\theta^i}{\bar{\mathbf{b}}_\theta^i+\bar{\boldsymbol{\eta}}^i}\right)\sum_{e=1}^{N_e} s_e\alpha_e^2\exp\left(-M_{\theta,i}(\mathbf{x}^{(k)},e)\right)\\
&+\sum_{i=1}^{N_p} 2a_\theta^{ij}a_\theta^{i\ell}[\mathbf{x}_j+\mathbf{x}_\ell]\left(1-\frac{\mathbf{b}_\theta^i}{\bar{\mathbf{b}}_\theta^i+\bar{\boldsymbol{\eta}}^i}\right)\sum_{e=1}^{N_e} s_e\alpha_e\beta_e\exp\left(-M_{\theta,i}(\mathbf{x}^{(k)},e)\right)\\
&+\sum_{i=1}^{N_p} a_\theta^{ij}a_\theta^{i\ell}\left(1-\frac{\mathbf{b}_\theta^i}{\bar{\mathbf{b}}_\theta^i+\bar{\boldsymbol{\eta}}^i}\right)\sum_{e=1}^{N_e} s_e\beta_e^2\exp\left(-M_{\theta,i}(\mathbf{x}^{(k)},e)\right)\\
&+\sum_{i=1}^{N_p} \frac{a_\theta^{ij}a_\theta^{i\ell}\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i+\bar{\boldsymbol{\eta}}^i)^2}\left[\sum_{e=1}^{N_e} s_e\beta_e\exp\left(-M_{\theta,i}(\mathbf{x}^{(k)},e)\right)\right]^2\\
&+\sum_{i=1}^{N_p} \frac{4a_\theta^{ij}a_\theta^{i\ell}\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i+\bar{\boldsymbol{\eta}}^i)^2}\mathbf{x}_j\mathbf{x}_\ell\left[\sum_{e=1}^{N_e} s_e\alpha_e\exp\left(-M_{\theta,i}(\mathbf{x}^{(k)},e)\right)\right]^2\\
&+\sum_{i=1}^{N_p} \frac{2(a_\theta^{ij}a_\theta^{i\ell})^2\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i+\bar{\boldsymbol{\eta}}^i)^2}\mathbf{x}_j\left[\sum_{e=1}^{N_e} s_e\alpha_e\exp\left(-M_{\theta,i}(\mathbf{x}^{(k)},e)\right)\right]\left[\sum_{e=1}^{N_e} s_e\beta_e\exp\left(-M_{\theta,i}(\mathbf{x}^{(k)},e)\right)\right]\\
&+\sum_{i=1}^{N_p} \frac{2a_\theta^{ij}a_\theta^{i\ell}\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i+\bar{\boldsymbol{\eta}}^i)^2}\mathbf{x}_\ell\left[\sum_{e=1}^{N_e} s_e\alpha_e\exp\left(-M_{\theta,i}(\mathbf{x}^{(k)},e)\right)\right]\left[\sum_{e=1}^{N_e} s_e\beta_e\exp\left(-M_{\theta,i}(\mathbf{x}^{(k)},e)\right)\right].
\end{aligned}
$$

To express the Hessian matrix $\mathbf{H}_\theta(\mathbf{x}^{(k)})$ in terms of the raytracing matrix A_θ, consider denoting the vectors

$$
\begin{aligned}
\mathbf{d}_i = &\left(1-\frac{\mathbf{b}_\theta^i}{\bar{\mathbf{b}}_\theta^i+\bar{\boldsymbol{\eta}}^i}\right)\sum_{e=1}^{N_e} s_e\alpha_e^2\exp\left(-M_{\theta,i}(\mathbf{x}^{(k)},e)\right)\\
&+\frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i+\bar{\boldsymbol{\eta}}^i)^2}\left[\sum_{e=1}^{N_e} s_e\alpha_e\exp\left(-M_{\theta,i}(\mathbf{x}^{(k)},e)\right)\right]^2,
\end{aligned}
$$

$$
\begin{aligned}
\mathbf{p}_i = &\left(1-\frac{\mathbf{b}_\theta^i}{\bar{\mathbf{b}}_\theta^i+\bar{\boldsymbol{\eta}}^i}\right)\sum_{e=1}^{N_e} s_e\alpha_e\beta_e\exp\left(-M_{\theta,i}(\mathbf{x}^{(k)},e)\right)\\
&+\frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i+\bar{\boldsymbol{\eta}}^i)^2}\left[\sum_{e=1}^{N_e} s_e\alpha_e\exp\left(-M_{\theta,i}(\mathbf{x}^{(k)},e)\right)\right]\left[\sum_{e=1}^{N_e} s_e\beta_e\exp\left(-M_{\theta,i}(\mathbf{x}^{(k)},e)\right)\right]
\end{aligned}
$$

and

$$\mathbf{q}_i = \left(1 - \frac{\mathbf{b}_\theta^i}{\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i}\right) \sum_{e=1}^{N_e} \rho(e)\beta_e^2 \exp\left(-M_{\theta,i}(\mathbf{x}^{(k)}, e)\right) + \frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)^2} \left[\sum_{e=1}^{N_e} s_e\beta_e \exp\left(-M_{\theta,i}(\mathbf{x}^{(k)}, e)\right)\right]^2.$$

Then we can express the entry $h_\theta^{j\ell}$ as

$$h_\theta^{j\ell} = 4\mathbf{x}_j^{(k)}\mathbf{x}_\ell^{(k)} \sum_{i=1}^{N_p} a_\theta^{ij} a_\theta^{i\ell} \mathbf{d}^{(k)} + 2(\mathbf{x}_j^{(k)} + \mathbf{x}_\ell^{(k)}) \sum_{i=1}^{N_p} a_\theta^{ij} a_\theta^{i\ell} \mathbf{p}^{(k)} + \sum_{i=1}^{N_p} a_\theta^{ij} a_\theta^{i\ell} \mathbf{q}^{(k)}.$$

Then if the matrix $\mathbf{H}_\theta(\mathbf{x}^{(k)})$ is symmetric positive definite, we can find the step direction at each iteration for the Newton method $\mathbf{s}^{(k)}$ by solving the linear system

$$\mathbf{H}_\theta(\mathbf{x}^{(k)})\mathbf{s}^{(k)} = -\nabla L(\mathbf{b}_\theta, \mathbf{x}^{(k)}).$$

To solve Equation (5.10) using a Newton method, the framework is as follows

---

Algorithm 5.3.5 Newton Method Algorithm For the Quadratic Attenuation Model

---

1: Find $\alpha_e, \beta_e, \gamma_e$ as a separable nonlinear inverse problem using the lease squares formulation in Equation (5.4)

2: Choose the initial volume $\mathbf{x}_0$

3: **for** $k = 1, 2, ...$ until converge **do**

4: Compute the vectors $\mathbf{v}^{(k)}$ and $\mathbf{y}^{(k)}$ for all θ

5: Find the gradient using $\nabla L(\mathbf{b}_\theta, \mathbf{x}^{(k)}) = 2\mathbf{X}^{(k)} A_\theta^T \mathbf{v}^{(k)} + A_\theta^T \mathbf{y}^{(k)}$ using Equation (4.19)

6: Compute vectors $\mathbf{d}^{(k)}, \mathbf{p}^{(k)}, \mathbf{q}^{(k)}$ to build the Hessian $\mathbf{H}_\theta(\mathbf{x}^{(k)})$

7: Find the step direction $\mathbf{s}^{(k)}$ by solving the linear system $\mathbf{H}_\theta(\mathbf{x}^{(k)})\mathbf{s}^{(k)} = -\nabla L(\mathbf{b}_\theta, \mathbf{x}^{(k)})$

8: Determine the step length α_k

9: Update the solution $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \alpha_k \mathbf{s}^{(k)}$

10: **if** $\sum_{\theta=1}^{N_\theta} L(\mathbf{b}_\theta, \mathbf{x}^{(k)}) > \sum_{\theta=1}^{N_\theta} L(\mathbf{b}_\theta, \mathbf{x}^{(k-1)})$ **then**

11: Refine parameter α_k; no sufficient descent

12: **end if**

13: **end for**

As in the gradient descent approach, in the Newton method we use regularization by early termination since the later iterates begin to be corrupted by noise as is the case for ill-posed problems (see Section 1.1). For digital breast tomosynthesis reconstructions, the Newton method requires only a few iterations (less than ten in most cases) and the number of inner iterations depends on the conditioning of the Hessian matrix. There is currently no automated convergence criteria, convergence is determined once the relative value of the function stagnates.

5.4 Summary

In this chapter we have expanded the reconstruction method proposed in [32] to account for the presence of air and micro-calcifications in the reconstructed breast. This extension allows their polyenergetic approach to reconstruct large clinical data sets. However, this quadratic approach is still limited in terms of the number and type of materials used in the forward model, and, in addition, fitting a quadratic polynomial to the linear model for attenuation introduces numerical errors to the reconstructions (shown in Chapter 11). In the next chapter, we address the limitations of the quadratic model for attenuation by developing a new attenuation model that allows for an arbitrary number and type of attenuating materials inside the object. Using arbitrary materials creates a more flexible model that can be used for polyenergetic digital tomosynthesis reconstructions of objects other than the breast.

Chapter 6

The Polyenergetic Model for the Attenuation of Multiple Materials

In the previous two chapters we have formulated the digital tomosynthesis reconstruction problem for breast cancer screening as a nonlinear ill-posed inverse problem, where the goal is to approximate the true volume from the given set of projection images. From the discussion of Chapters 2 and 3, we have established that it is important to take into account the polyenergetic nature of the x-ray source in the forward model in order to eliminate the presence of beam hardening artifacts in the reconstruction (Section 3.5). By expressing the attenuation of the object as a function of the glandular fractions, we have shown that the integral equation for the polyenergetic digital tomosynthesis problem using Beer's law, as posed in Equation (3.9), can be solved iteratively.

So far, we have presented two polyenergetic reconstruction frameworks that can improve the reconstructions obtained using the standard approaches of filtered backprojection and MLEM. The reconstruction framework described in Chapter 4, developed by Chung et al. [32], eliminates beam hardening artifacts by modeling explicitly the polyenergetic nature of the x-ray spectrum. However, this model is of limited scope to full clinical size problems because it only accounts for the presence of two attenuating materials inside the imaged breast (namely glandular and adipose tissue) ignoring the presence of air and microcalcifications. An attempt to extend the linear attenuation model is presented in Chapter

5, where the quadratic model for attenuation is described. The quadratic model improves the linear model in that it accounts for air and micro-calcifications present in the imaged area, but the model is limited to the problem of breast tomosynthesis imaging since adding more attenuating materials to the quadratic fit is a complex task.

This chapter proposes a linear polyenergetic multi-material model for the digital breast tomosynthesis reconstruction problem that overcomes the shortcomings of the models previously described. The derivation and mathematical aspects of the multi-material attenuation function for each voxel are discussed, and the reconstruction framework using both a gradient descent and Newton optimization approach is described. The goal of the multi-material attenuation model is to provide flexibility in the choice and number of attenuating materials present inside the imaged object. This allows us to extend the proposed model to digital tomosythesis imaging of other objects or parts of the body.

6.1 Building the Forward Model

We begin to derive the multi-material attenuation model by considering the *mass* attenuation coefficient (as oposed to the *linear* attenuation coefficient from the previous models) for a given voxel j and a particular energy e. The mass attenuation coefficient of a substance is the ratio of the energy absorption of the material per unit mass. As described in Section 2.1.2, for a given voxel in the discretized object, the *mass* attenuation coefficient is the ratio of the *linear* attenuation coefficient for the voxel and the density of the composite materials inside the voxel. That is, if $\boldsymbol{\mu}_{e,j}$ denotes the linear attenuation coefficient for voxel j at energy e and $\boldsymbol{\rho}_j$ denotes the density of the composite materials inside the voxel, then we can write

$$\frac{\boldsymbol{\mu}_{e,j}}{\boldsymbol{\rho}_j} = \left(\frac{\mu}{\rho}\right)_{e,j} = \textit{mass} \text{ attenuation coefficient at voxel } j \text{ for energy } e.$$

Therefore, using the equation

$$\boldsymbol{\mu}_{e,j} = \boldsymbol{\rho}_j \left(\frac{\mu}{\rho}\right)_{e,j}. \tag{6.1}$$

we can recover the *linear* attenuation coefficient from the *mass* attenuation coefficient if the composite density of the voxel $\boldsymbol{\rho}_j$ is known.

To build the attenuation function for the forward model, assume there are N_m arbitrary attenuating materials composing the object. Let each of the values $\{\mathbf{w}_{j,1}, \mathbf{w}_{j,2}, ..., \mathbf{w}_{j,N_m}\}$ represent the weight fractions (or percentage) of each material indexed $1, 2, ..., N_m$ present inside voxel j. This means that for voxel j we have

$$\sum_{m=1}^{N_m} \mathbf{w}_{j,m} = 1,$$

or in other words, the percentage of all materials present adds up to one. Recall that the discrete formulation of Beer's law for a polyenergetic x-ray is given in Equation (3.9) by

$$\mathbf{b}_\theta^i = \sum_{e=1}^{N_e} s_e \exp\left(-\sum_{j=1}^{N_v} a_\theta^{ij} \boldsymbol{\mu}_{e,j}\right)$$

where N_e is the number of discrete energies, s_e is the value of the spectral distribution function for energy indexed e, N_v is the number of voxels, and a_θ^{ij} is the length of the x-ray passing through voxel j and contributing to pixel i in projection θ. The value $\boldsymbol{\mu}_{e,j}$ is the *linear* attenuation of voxel j for energy level e. Using Equation (6.1) we have that the forward model for the acquisition of projection θ in terms of the mass attenuation and density of each voxel is

$$\mathbf{b}_i^\theta = \sum_{e=1}^{N_e} s_e \exp\left(-\sum_{j=1}^{N_v} a_\theta^{ij} \boldsymbol{\rho}_j \left(\frac{\mu}{\rho}\right)_{e,j}\right). \tag{6.2}$$

Since we are modeling the presence of multiple materials that have different attenuating properties, we need to decompose the density $\boldsymbol{\rho}_j$ and the mass attenuation $\left(\frac{\mu}{\rho}\right)_{e,j}$ of voxel j in terms of the materials present inside it.

Consider decomposing the *composite material* mass attenuation coefficient $\left(\frac{\mu}{\rho}\right)_{e,j}$ for a given voxel j at energy level e. If we know the weight fraction of each material $\mathbf{w}_{j,m}$ present in voxel j, we can express the composite material mass attenuation coefficient as a linear combination of the mass attenuation coefficient for each individual material and the respective weight fractions. To do this, we let r_m represent the density of material m, and let $u_{e,m}$ represent the linear attenuation coefficient of material m for energy level e. Then,

by Equation (6.1), the mass attenuation for each material is given as the ratio of the linear attenuation coefficient and the density of the material, that is

$$\frac{u_{m,e}}{r_m} = \left(\frac{u}{r}\right)_{m,e}.$$

Thus, we can write the composite material mass attenuation coefficient for voxel j at energy level e using N_m materials as

$$\left(\frac{\mu}{\rho}\right)_{e,j} = \sum_{m=1}^{N_m} \mathbf{w}_{j,m} \left(\frac{u}{r}\right)_{m,e}. \tag{6.3}$$

Now consider decomposing the density of the voxel in terms of the N_m materials assumed to be present inside. If we know the density of each material r_m and the weight fractions $\mathbf{w}_{j,m}$ for all m, we can write the density of the composite material inside voxel j as

$$\boldsymbol{\rho}_j = \frac{1}{\sum_{m=1}^{N_m} \frac{\mathbf{w}_{j,m}}{r_m}}. \tag{6.4}$$

Using Equations (6.3) and (6.4), we can write the linear attenuation function for each voxel j at energy level e in terms of the mass attenuation and density of each material by

$$\boldsymbol{\mu}_{e,j} = \frac{\sum_{m=1}^{N_m} \mathbf{w}_{j,m} \left(\frac{u}{r}\right)_{m,e}}{\sum_{m=1}^{N_m} \frac{\mathbf{w}_{j,m}}{r_m}}. \tag{6.5}$$

This allows us to write the forward projection model in Equation (6.2) for pixel i and projection θ in terms of the attenuating properties of each material present in the object by

$$\mathbf{b}_\theta^i = \sum_{e=1}^{N_e} s_e \exp\left(-\sum_{j=1}^{N_v} a_\theta^{ij} \frac{\sum_{m=1}^{N_m} \mathbf{w}_{j,m} \left(\frac{u}{r}\right)_{m,e}}{\sum_{m=1}^{N_m} \frac{\mathbf{w}_{j,m}}{r_m}} \right). \tag{6.6}$$

In this forward model, the object will be represented by N_m vectors $\{\mathbf{w}_{:,1}, \mathbf{w}_{:,2}, ..., \mathbf{w}_{:,N_m}\}$ that contain the weight fractions for each composing material. All the weight fractions can be combined using Equation (6.5) to obtain one single attenuation vector for the object.

We can eliminate one variable from the forward projection model using the relation between the weight fractions of the materials for voxel j. If

$$\sum_{m=1}^{N_m} \mathbf{w}_{j,m} = 1,$$

we can arbitrarily choose material $m = 1$ to eliminate and write the weight fractions for voxel j as

$$\mathbf{w}_{j,1} = 1 - \sum_{m=2}^{N_m} \mathbf{w}_{j,m}.$$

Using the variable elimination and Equation (6.6), we have the new forward projection model

$$\mathbf{b}_\theta^i = \sum_{e=1}^{n_e} s_e \exp\left(- \sum_{j=1}^{N_v} a_\theta^{ij} \frac{\left(\frac{u}{r}\right)_{1,e} + \sum_{m=2}^{N_m} \mathbf{w}_{j,m} \left(\left(\frac{u}{r}\right)_{m,e} - \left(\frac{u}{r}\right)_{1,e} \right)}{\frac{1}{r_1} + \sum_{m=2}^{N_m} \mathbf{w}_{j,m} \left(\frac{1}{r_m} - \frac{1}{r_1} \right)} \right). \tag{6.7}$$

Equation 6.7 is the proposed image acquisition model for one incident angle in the digital tomosynthesis problem given multiple attenuating materials inside the object.

6.2 Statistical Considerations

In a similar manner as the reconstruction process for the quadratic model for attenuation described in Chapter 5, we consider the statistical properties of the image acquisition process. We model the variable representing the energy measured by each pixel in the detector as a Poisson random variable with additional background noise. Refer to Section 3.1 to see how the source-detector interaction in digital tomosynthesis can be modeled as a Bernoulli random variable (the detector) receiving a Poisson random variable as "input" (the source).

For simplicity of notation, let $\mathbf{w}$ denote the set of vectors representing the weight fractions for the $N_m - 1$ materials included in the model (i.e. for N_m materials we let $\mathbf{w} = \{\mathbf{w}_{:,2}, ..., \mathbf{w}_{:,N_m,}\}$). Given the forward model in (6.7), the expected value of the measured data for pixel i in projection θ is

$$E(\mathbf{b}_\theta^i; \mathbf{w}) = \sum_{e=1}^{n_e} s_e \exp\left(-\sum_{j=1}^{N_v} a_\theta^{ij} \frac{\left(\frac{u}{r}\right)_{1,e} + \sum_{m=2}^{N_m} \mathbf{w}_{j,m} \left(\left(\frac{u}{r}\right)_{m,e} - \left(\frac{u}{r}\right)_{1,e} \right)}{\frac{1}{r_1} + \sum_{m=2}^{N_m} \mathbf{w}_{j,m} \left(\frac{1}{r_m} - \frac{1}{r_1} \right)} \right) + \bar{\boldsymbol{\eta}}^i \quad (6.8)$$

where the notation $\mathbf{w}_{j,m}$ denotes a *scalar* jth entry of the vector containing the weight fractions for material m (i.e. $\mathbf{w}_{m,j} = \mathbf{w}_m(j)$) and $\bar{\boldsymbol{\eta}}^i$ represents additive noise in the image acquisition. The noise is generally due to scatter and is also assumed to follow a Poisson distribution whose statistical mean is known or can be approximated. For notation purposes, let $E(\mathbf{b}_\theta^i; \mathbf{w}) = \bar{\mathbf{b}}_i^\theta + \bar{\boldsymbol{\eta}}^i$. We then have that each pixel i of the observed projection data at incident angle θ is a realization of the Poisson random variable

$$\mathbf{b}_\theta^i \sim \text{Poisson}\left(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i\right).$$

Using the theory of maximum likelihood estimation, we know that the probability of observing image $\mathbf{b}_\theta$ for all θ given the set of weight fractions $\mathbf{w} = \{\mathbf{w}_{:,2}, ..., \mathbf{w}_{:,N_m,}\}$ for the imaged object is determined by the Poisson likelihood function

$$p(\mathbf{b}_\theta; \mathbf{w}) = \prod_{i=1}^{N_p} \frac{e^{-(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)} (\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)^{\mathbf{b}_\theta^i}}{\mathbf{b}_\theta^i!} \quad (6.9)$$

with $\mathbf{b}_\theta^i$ being the observed data and N_p representing the total number of pixels in the detector. As in the previous chapters, our goal is to find the set of weight fractions $\mathbf{w}$ that maximize the likelihood function above. The set of weight fractions that maximize Equation (6.9) represent the volume that most likely produced the observed projection $\mathbf{b}_\theta$ and thus, our reconstructed volume.

6.3 The Iterative Reconstruction Framework

The goal of the reconstruction process is to find the set of vectors $\mathbf{w} = \{\mathbf{w}_{:,2}, ..., \mathbf{w}_{:,N_m,}\}$ that maximize the Poisson likelihood function in Equation (6.9). This set of vectors represent the weight fractions for each material composing the reconstructed volume. In practice, for numerical optimization problems we try to find the minimizer rather than the maximizer of a function. Therefore, we want to transform the problem of maximizing Equation (6.9) to minimizing the negative log-likelihood function. We can transform the Poisson likelihood function into its corresponding negative log-likelihood function by taking the negative logarithm of $p(\mathbf{b}_\theta; \mathbf{w})$

$$-L(\mathbf{b}_\theta; \mathbf{w}) = -\log(p(\mathbf{b}_\theta^i; \mathbf{w}))$$

to obtain

$$-L(\mathbf{b}_\theta; \mathbf{w}) = \sum_{i=1}^{N_p} (\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i) - \mathbf{b}_\theta^i \log(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i) + c \tag{6.10}$$

where c is a constant, N_p is the number of pixels in the detector, and $\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i$ is the expected value of the measurement at pixel i for projection at incident angle θ given in Equation (6.8). Since the logarithm is a monotone operation, finding the minimizer of the negative log-likelihood function in Equation (6.10) is analytically equivalent to finding the maximizer of the Poisson likelihood function.

The reconstruction problem is to iteratively find the maximum likelihood estimator (MLE) $\mathbf{w}_{MLE}$ by solving the numerical optimization problem

$$\mathbf{w}_{MLE} = \arg \min_{\mathbf{w}} \left\{ \sum_{\theta=1}^{N_\theta} -L(\mathbf{b}_\theta; \mathbf{w}) \right\}. \tag{6.11}$$

In this multi-material framework, we are reconstructing a set of $N_m - 1$ unknown vectors that represent the weight fractions of each material modeled. Note that this problem differs from the linear attenuation model reconstruction problem described in Chapter 4 in that we are interested in reconstructing a set of weight fraction vectors as opposed to a single unknown vector containing the percent glandular fractions of the breast. Therefore, the optimization problem in this model has increased the computational complexity as we have

$(N_m - 2) \times N_v$ more unknown variables. As in the quadratic attenuation reconstruction framework, we use a gradient descent approach and a Newton method to solve the optimization problem described by Equation (6.11).

6.3.1 Gradient Descent

As discussed in Section 5.3.1, the gradient descent iteration is a line search method that moves along the direction of the negative derivative at every step. The negative derivative direction is the direction in which the function decreases most rapidly. The iteration is given by

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} - \alpha_k \nabla L(\mathbf{b}_\theta, \mathbf{w}^{(k)}) \tag{6.12}$$

where α_k is the step length for each iteration k and the step direction $\nabla L(\mathbf{b}_\theta, \mathbf{w}^{(k)})$ is the derivative of the cost function evaluated at the current iterate. Note that in the multi-material model, the step direction is given as the derivative of the cost function with respect to all $(N_m - 1) \times N_v$ unknowns, namely

$$\nabla L(\mathbf{b}_\theta, \mathbf{w}^{(k)}) = \frac{\partial}{\partial \mathbf{w}_{j,m}} \Big(-L(\mathbf{b}_\theta, \mathbf{w}^{(k)}) \Big)$$

where the index $m = 2, 3, ..., N_m$ represents the number of materials and the index $j = 1, 2, ..., N_v$ corresponds to the voxels.

To simplify notation in the derivation process, consider writing the attenuation function $\boldsymbol{\mu}_{e,j}$ for voxel j at energy level e in Equation (6.5) as the quotient of two functions.

1. Let the function $f(e, \mathbf{w}) = \left(\frac{u}{r}\right)_{1,e} + \sum_{m=2}^{N_m} \mathbf{w}_{j,m} \left[\left(\frac{u}{r}\right)_{m,e} - \left(\frac{u}{r}\right)_{1,e} \right]$ be the numerator of $\boldsymbol{\mu}_{e,j}$. Then its first derivative with respect to the j-th voxel of the m-th material is given by
$$\frac{\partial f(e, \mathbf{w})}{\partial \mathbf{w}_{j,m}} = \left(\frac{u}{r}\right)_{m,e} - \left(\frac{u}{r}\right)_{1,e}.$$

2. Let the function $g(e, \mathbf{w}) = \frac{1}{r_1} + \sum_{m=2}^{N_m} \mathbf{w}_{j,m} \left[\frac{1}{r_m} - \frac{1}{r_1} \right]$ be the denominator of $\boldsymbol{\mu}_{e,j}$. Then

its first derivative with respect to the j-th voxel of the m-th material is given by

$$\frac{\partial g(e,\mathbf{w})}{\partial \mathbf{w}_{j,m}} = \frac{1}{r_m} - \frac{1}{r_1}.$$

Given $\boldsymbol{\mu}_{e,j}(\mathbf{w}) = \frac{f(e,\mathbf{w})}{g(e,\mathbf{w})}$ we have that the first derivative of the attenuation function with respect to the j-th voxel of the m-th material is

$$\begin{aligned}\frac{\partial \boldsymbol{\mu}_{e,j}}{\partial \mathbf{w}_{j,m}} =& \frac{\partial}{\partial \mathbf{w}_{j,m}} \frac{f(e,\mathbf{w})}{g(e,\mathbf{w})} \\ =& \frac{\partial f(e,\mathbf{w})}{\partial \mathbf{w}_{j,m}} \cdot \frac{1}{g(e,\mathbf{w})} - \frac{f(e,\mathbf{w})}{[g(e,\mathbf{w})]^2} \frac{\partial g(e,\mathbf{w})}{\partial \mathbf{w}_{j,m}}.\end{aligned} \tag{6.13}$$

To compute each entry of the step direction vector $\nabla L(\mathbf{b}_\theta, \mathbf{w})$, we use Equation (6.10) to obtain

$$\frac{\partial(-L(\mathbf{b}_\theta,\mathbf{w}))}{\partial \mathbf{w}_{j,m}} = \sum_{i=1}^{N_p} \left(1 - \frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}\right) \frac{\partial(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}{\partial \mathbf{w}_{j,m}}. \tag{6.14}$$

where $\dfrac{\partial(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}{\partial \mathbf{w}_{j,m}}$ is the derivative of the expected value in Equation (6.8) with respect to the j-th voxel of the m-th material for projection θ given by

$$\begin{aligned}\frac{\partial(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}{\partial \mathbf{w}_{j,m}} =& \frac{\partial}{\mathbf{w}_{j,m}} \sum_{e=1}^{n_e} s_e \exp\left(-\sum_{j=1}^{N_v} a_\theta^{ij} \boldsymbol{\mu}_{e,j}\right) \\ =& -a_\theta^{ij} \sum_{e=1}^{n_e} s_e \frac{\partial \boldsymbol{\mu}_{e,j}}{\partial \mathbf{w}_{j,m}} \exp\left(-\sum_{j=1}^{N_v} a_\theta^{ij} \boldsymbol{\mu}_{e,j}\right).\end{aligned} \tag{6.15}$$

This gives the $(m-2) \times N_v + j$ entry of the step direction vector as

$$\frac{\partial(-L(\mathbf{b}_\theta,\mathbf{w}))}{\partial \mathbf{w}_{j,m}} = \sum_{i=1}^{N_p} a_\theta^{ij} \left(\frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)} - 1\right) \sum_{e=1}^{n_e} s_e \frac{\partial \boldsymbol{\mu}_{e,j}}{\partial \mathbf{w}_{j,m}} \exp\left(-\sum_{j=1}^{N_v} a_\theta^{ij} \boldsymbol{\mu}_{e,j}\right). \tag{6.16}$$

To solve Equation (6.11) using a gradient descent method, the framework is as follows

Algorithm 6.3.6 Gradient Descent Algorithm For the Multi-material Attenuation Model

1: Choose the initial set of volumes representing the weight fractions for each material given by $\mathbf{w}^{(0)} = \begin{bmatrix} \mathbf{w}_{:,2}^{(0)} \\ \vdots \\ \mathbf{w}_{:,N_m,}^{(0)} \end{bmatrix}$

2: **for** $k = 1, 2, ...$ until converge **do**

3: Evaluate the forward model using Equation (6.7)

4: Find the step direction vector $\nabla L(\mathbf{b}_\theta, \mathbf{w}^{(k)})$ for all θ using Equation (6.16)

5: Determine the step length α_k

6: Update the solution $\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} - \alpha_k \nabla L(\mathbf{b}_\theta, \mathbf{w}^{(k)})$

7: **if** $\sum_{\theta=1}^{N_\theta} L(\mathbf{b}_\theta, \mathbf{w}^{(k)}) > \sum_{\theta=1}^{N_\theta} L(\mathbf{b}_\theta, \mathbf{w}^{(k-1)})$ **then**

8: Refine parameter α_k; no sufficient descent

9: **end if**

10: **end for**

The gradient descent method is known to converge very slowly for some problems, but as is shown in Chapter 11, for digital breast tomosynthesis reconstructions only a few iterations of this method are needed (less than twenty in most cases) to produce quality reconstructions. We choose to solve Equation (6.11) using this method because of the computational intensity of the higher derivatives of the cost function. In numerical experiments it has been noted that a step length $\alpha_k < 1$ produces steady descent with few line search iterations. For this problem, the method uses regularization by early termination since later iterates begin to be corrupted by noise (see Section 1.1). As in the case for the quadratic model for attenuation, convergence is determined as the point where the relative function value stagnates.

6.3.2 Newton Approach

Although a working implementation of a Newton method is not presented in this thesis, the numerical aspects of using a Newton approach are described in this section. As discussed in Section 5.3.2, a Newton method is a line search strategy with search direction given by

the second derivative term in the Taylor series expansion of the cost function. This search direction has a much faster rate of convergence than the gradient descent method (typically quadratic). However, the search direction is not guaranteed to be a descent direction unless the Hessian of the cost function is symmetric positive definite.

A Newton iteration takes the form

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \alpha_k \mathbf{s}^{(k)} \tag{6.17}$$

where α_k is the step length and $\mathbf{s}^{(k)}$ is the step direction. Here, the step direction $\mathbf{s}^{(k)}$ is the solution to the linear system

$$\mathbf{H}_\theta(\mathbf{w}^{(k)})\mathbf{s}^{(k)} = -\nabla L(\mathbf{b}_\theta, \mathbf{w}^{(k)}) \tag{6.18}$$

where $\mathbf{H}_\theta(\mathbf{w}^{(k)})$ is the Hessian matrix of the cost function and $\nabla L(\mathbf{b}_\theta, \mathbf{w}^{(k)})$ is the gradient of the cost function both evaluated at the k-th iterate.

Recall that in the multi-material attenuation model describe in Equation (6.5), we represent the reconstructed volume in terms of $N_m - 1$ vectors $\mathbf{w}_{:,m} \in \mathbb{R}^{N_v}$ that correspond to the weight fractions of each material inside the object. To compute the Hessian of the cost function $L(\mathbf{b}_\theta, \mathbf{w}^{(k)})$, we then need the entry $(m-2) \times N_v + j$ of the first derivative of the cost function, which is given in Equation (6.14) as

$$\frac{\partial(-L(\mathbf{b}_\theta, \mathbf{w}))}{\partial \mathbf{w}_{j,m}} = \sum_{i=1}^{N_p} \left(1 - \frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}\right) \frac{\partial(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}{\partial \mathbf{w}_{j,m}}$$

where $j = 1, 2, .., N_v$ is the voxel index and $m = 2, 3, ..., N_m$ is the index of the materials modeled. Therefore, we can express the Hessian matrix for the problem as a block matrix, with $(m-1) \times (m-1)$ blocks of size $N_v \times N_v$. Let $j, \ell = 1, 2, ..., N_v$, then the (j, ℓ) entry of the Hessian matrix for block (m, n) is

$$h_{m,n}^{j\ell} = \frac{\partial^2(-L(\mathbf{b}_\theta, \mathbf{w}))}{\partial \mathbf{w}_{\ell,n} \partial \mathbf{w}_{j,m}}$$

which can be expanded to the equation

$$\sum_{i=1}^{N_p} \frac{\partial}{\partial \mathbf{w}_{\ell,n}} \left(1 - \frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}\right) \frac{\partial(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}{\partial \mathbf{w}_{j,m}} + \left(1 - \frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}\right) \frac{\partial}{\partial \mathbf{w}_{\ell,n}} \left(\frac{\partial(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}{\partial \mathbf{w}_{j,m}}\right).$$

To build the Hessian matrix we begin by using the equality

$$\frac{\partial}{\partial \mathbf{w}_{\ell,n}} \left(1 - \frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}\right) \frac{\partial(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}{\partial \mathbf{w}_{j,m}} = \frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)^2} \frac{\partial(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}{\partial \mathbf{w}_{j,m}} \frac{\partial(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}{\partial \mathbf{w}_{\ell,n}}$$

and Equation (6.15) to compute the first term in the summation. The second term in the summation of each entry of the Hessian follows from the equation

$$\begin{aligned}\frac{\partial}{\partial \mathbf{w}_{\ell,n}} \left(\frac{\partial(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}{\partial \mathbf{w}_{j,m}}\right) = & -a_\theta^{ij} \sum_{e=1}^{n_e} s_e \frac{\partial \boldsymbol{\mu}_{e,j}}{\partial \mathbf{w}_{j,m}} \frac{\partial}{\partial \mathbf{w}_{\ell,n}} \left[\exp\left(-\sum_{j=1}^{N_v} a_\theta^{ij} \boldsymbol{\mu}_{e,j}\right)\right] \\ & - a_\theta^{ij} \sum_{e=1}^{n_e} s_e \frac{\partial^2 \boldsymbol{\mu}_{e,j}}{\partial \mathbf{w}_{\ell,n} \partial \mathbf{w}_{j,m}} \exp\left(-\sum_{j=1}^{N_v} a_\theta^{ij} \boldsymbol{\mu}_{e,j}\right)\end{aligned}$$

where the derivative of the exponential is

$$\frac{\partial}{\partial \mathbf{w}_{\ell,n}} \exp\left(-\sum_{j=1}^{N_v} a_\theta^{ij} \boldsymbol{\mu}_{e,j}\right) = -a_\theta^{i\ell} \frac{\partial \boldsymbol{\mu}_{e,j}}{\partial \mathbf{w}_{\ell,n}} \exp\left(-\sum_{j=1}^{N_v} a_\theta^{ij} \boldsymbol{\mu}_{e,j}\right)$$

and the second derivative of the attenuation function is

$$\frac{\partial^2 \boldsymbol{\mu}_{e,j}}{\partial \mathbf{w}_{\ell,n} \partial \mathbf{w}_{j,m}} = \frac{\dfrac{\partial f(e,\mathbf{w})}{\partial \mathbf{w}_{j,m}} \dfrac{\partial g(e,\mathbf{w})}{\partial \mathbf{w}_{\ell,n}} - \dfrac{\partial g(e,\mathbf{w})}{\partial \mathbf{w}_{j,m}} \dfrac{\partial f(e,\mathbf{w})}{\partial \mathbf{w}_{\ell,n}}}{g(e,\mathbf{w})^2} - 2\frac{\dfrac{\partial g(e,\mathbf{w})}{\partial \mathbf{w}_{j,m}} \dfrac{\partial g(e,\mathbf{w})}{\partial \mathbf{w}_{\ell,n}} f(e,\mathbf{w})}{g(e,\mathbf{w})^3}.$$

Then we have the the expanded (j,ℓ) entry in the block (m,n) of the Hessian matrix is

given by the equation

$$
\begin{aligned}
h_{m,n}^{j\ell} =& \sum_{i=1}^{N_p} \frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)^2} \frac{\partial(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}{\partial \mathbf{w}_{j,m}} \frac{\partial(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}{\partial \mathbf{w}_{\ell,n}} \\
& - \sum_{i=1}^{N_p} a_\theta^{ij} a_\theta^{i\ell} \left(\frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)} - 1 \right) \sum_{e=1}^{n_e} s_e \frac{\partial \boldsymbol{\mu}_{e,j}}{\partial \mathbf{w}_{j,m}} \frac{\partial \boldsymbol{\mu}_{e,j}}{\partial \mathbf{w}_{\ell,n}} \exp\left(- \sum_{j=1}^{N_v} a_\theta^{ij} \boldsymbol{\mu}_{e,j} \right) \\
& + \sum_{i=1}^{N_p} a_\theta^{ij} \left(\frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)} - 1 \right) \sum_{e=1}^{n_e} s_e \frac{\partial^2 \boldsymbol{\mu}_{e,j}}{\partial \mathbf{w}_{\ell,n} \partial \mathbf{w}_{j,m}} \exp\left(- \sum_{j=1}^{N_v} a_\theta^{ij} \boldsymbol{\mu}_{e,j} \right).
\end{aligned} \tag{6.19}
$$

If the block matrix $\mathbf{H}_\theta(\mathbf{w}^{(k)})$ is symmetric positive definite, the Newton search direction guarantees descent and we find it by solving the linear system

$$\mathbf{H}_\theta(\mathbf{w}^{(k)})\mathbf{s}^{(k)} = -\nabla L(\mathbf{b}_\theta, \mathbf{w}^{(k)}).$$

at each iteration.

Given the Hessian matrix for the multi-material problem in Equation (6.19), we can only establish that $\mathbf{H}_\theta(\mathbf{w}^{(k)})$ is symmetric, not necessarily positive definite. This is why at this point we are unable to provide a functional implementation of the Newton method to solve the digital tomosynthesis reconstruction problem using the multi-material attenuation model. However, if the matrix $\mathbf{H}_\theta(\mathbf{w}^{(k)})$ is shown to be positive definite, the framework to solve Equation (6.11) using a Newton method is as follows

---

Algorithm 6.3.7 Newton Method For the Multi-material Attenuation Model

---

1: Choose the initial set of volumes representing the weight fractions for each material given by $\mathbf{w}^{(0)} = \begin{bmatrix} \mathbf{w}_{:,2}^{(0)} \\ \vdots \\ \mathbf{w}_{:,N_m,}^{(0)} \end{bmatrix}$

2: **for** $k = 1, 2, ...$ until converge **do**

3: Evaluate the forward model using Equation (6.7)

4: Build the gradient vector $\nabla L(\mathbf{b}_\theta, \mathbf{w}^{(k)})$ for all θ using Equation (6.16)

5: Build the Hessian matrix using $\mathbf{H}_\theta(\mathbf{w}^{(k)})$ Equation (6.19)

Find the step direction $\mathbf{s}^{(k)}$ by solving the linear system $\mathbf{H}_\theta(\mathbf{w}^{(k)})\mathbf{s}^{(k)} = -\nabla L(\mathbf{b}_\theta, \mathbf{w}^{(k)})$
Determine the step length α_k
Update the solution $\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} - \alpha_k \mathbf{w}^{(k)}$
if $\sum_{\theta=1}^{N_\theta} L(\mathbf{b}_\theta, \mathbf{w}^{(k)}) > \sum_{\theta=1}^{N_\theta} L(\mathbf{b}_\theta, \mathbf{w}^{(k-1)})$ **then**
    Refine parameter α_k; no sufficient descent
end if
end for

For this method, the step length is the natural Newton method step length $\alpha_k = 1$. The number of inner iterations to solve the linear system that determines the step direction depends on the conditioning of the Hessian matrix. Similarly to the gradient descent approach described earlier in the chapter, in the Newton method regularization is done by early termination since the later iterates begin to be corrupted by noise as is the case for ill-posed problems (see Section 1.1). There is currently no automated convergence criteria, convergence is determined once the relative value of the function stagnates.

6.4 Summary

In this chapter we have presented a multi-material forward model for the polyenergetic digital tomosynthesis reconstruction problem for breast cancer screening. In the multi-material model, we formulate the attenuation function of the object in terms of the weight fractions of an arbitrary number of materials N_m present inside the object. This formulation allows us to expand the quadratic model for attenuation presented in Chapter 5 to capture additional materials of interest in breast cancer screening, like skin and solutions used for contrast of lesions. In addition, by keeping the formulation of the model independent of glandular fractions, we can extend the reconstruction framework to tomosynthesis imaging of other objects or parts of the body. However, as noted in Section 6.3.2, we are unable to analyze the derivatives of the cost function to derive conditions for a positive definite Hessian matrix. In the next chapter we describe a simplified version of the multi-material model that allows us to include multiple arbitrary attenuating materials inside the imaged object but is easier to analyze and implement.

Chapter 7

A Simplified Polyenergetic Multimaterial Model

In this work we seek to develop a physically accurate forward model of the image acquisition process for polyenergetic digital breast tomosynthesis that accounts for the presence of multiple attenuating materials inside the imaged object. We began with the linear attenuation model in Chapter 4, where the volume was reconstructed in terms of the percent glandular fraction of each voxel to determine the presence of glandular and adipose tissue inside the object. This model allowed us to reduce the presence of beam hardening artifacts by using a polyenergetic formulation of Beer's law, and gave us insight into the the advantages using glandular fractions to represent the attenuation function of the object. However, the linear model for attenuation is limiting in terms of the number and type of materials modeled. The quadratic model developed in Chapter 5 extends the linear attenuation model to include the presence of air and micro-calcifications inside the breast, allowing for a full reconstruction of the imaged area. Unfortunately, as in the linear attenuation model, the quadratic model is limiting in terms of reconstructing only the percent glandular fractions of the volume and cannot be generalized to arbitrary materials present inside the volume. In Chapter 6 we presented the multi-material attenuation model where the volume is reconstructed in terms of the weight fractions of an arbitrary number of attenuating materials. This model allowed us to explicitly account for the attenuation properties of each material in the object, thus

giving us a more physically accurate depiction of the image acquisition process and extend the reconstruction framework to digital tomosynthesis imaging of other parts of the body. The downside of the multi-material model is that the additional materials modeled increase the number of unknowns and make the problem more complicated and computationally intensive.

In this chapter we develop a simplified forward model from the multi-material attenuation function described the previous chapter using some simple assumptions. This simplified model allows us to assume the presence of an arbitrary number of attenuating materials inside the breast while reducing the computational complexity of the reconstruction framework. We show how the simplifying assumptions are valid for clinical problems and derive the forward model for the polyenergetic problem. Finally, we summarize the reconstruction framework using both a gradient descent and Newton optimization approach.

7.1 Building the Forward Model

In the last chapter, we built the multi-material attenuation model by writing Beer's law in terms of the mass attenuation coefficient and density of each voxel. For a given voxel in the discretized object, the *mass* attenuation coefficient is given as a ratio of the *linear* attenuation coefficient for the voxel and the density of the composite materials at that voxel. That is, if $\boldsymbol{\mu}_{e,j}$ denotes the linear attenuation coefficient for voxel j at energy e and $\boldsymbol{\rho}_j$ denotes the density of the composite materials inside the voxel, we have

$$\left(\frac{\mu}{\rho}\right)_{e,j} = \textit{mass} \text{ attenuation coefficient at voxel } j \text{ for energy } e.$$

Moreover, the *linear* attenuation coefficient of the voxel $\boldsymbol{\mu}_{e,j}$ for energy level e is given by

$$\boldsymbol{\mu}_{e,j} = \boldsymbol{\rho}_j \left(\frac{\mu}{\rho}\right)_{e,j} \tag{7.1}$$

Using the discrete version of Beer's law for a polyenergetic ray we have the forward projection model

$$\mathbf{b}_\theta^i = \sum_{e=1}^{N_e} s_e \exp\left(-\sum_{j=1}^{N_v} a_\theta^{ij} \boldsymbol{\mu}_{j,e}\right)$$

where, as before, s_e is the value of the spectral distribution function for energy level e and a_θ^{ij} is the length of the x-ray passing through voxel j and contributing to pixel i of projection θ. We can write the forward model in terms of the mass attenuation and density of the voxel by

$$\mathbf{b}_\theta^i = \sum_{e=1}^{N_e} s_e \exp\left(-\sum_{j=1}^{N_v} a_\theta^{ij} \boldsymbol{\rho}_j \left(\frac{\mu}{\rho}\right)_{e,j} \right). \tag{7.2}$$

Assume there are N_m attenuating materials present inside the voxel j where the set $\{\mathbf{w}_{j,1}, ..., \mathbf{w}_{j,N_m}\}$ represents the weight fractions (or percentage) of each material m inside the voxel j and

$$\sum_{m=1}^{N_m} \mathbf{w}_{j,m} = 1.$$

As in the previous chapter, we can decompose the voxel's mass attenuation coefficient for energy level e in terms of the weight fractions of each material by

$$\left(\frac{\mu}{\rho}\right)_{e,j} = \sum_{e=1}^{N_e} \mathbf{w}_{m,j} \left(\frac{u}{r}\right)_{m,e} \tag{7.3}$$

where the mass attenuation is

$$\left(\frac{u}{r}\right)_{m,e} = \frac{u_{e,m}}{r_m},$$

or the ratio of the linear attenuation coefficient to the density of the m-th material. Similarly, to decompose the composite density of voxel j in terms of the materials present inside we write

$$\boldsymbol{\rho}_j = \frac{1}{\sum_{m=1}^{N_m} \frac{\mathbf{w}_{j,m}}{r_m}} \tag{7.4}$$

where the density of each material is r_m and the weight fractions are $\mathbf{w}_{j,m}$. Note, that if we are modeling N_m materials that have very similar densities, we have

$$r_1 \approx r_2 \approx ... \approx r_{N_m}$$

and thus we can define the average density $\bar{r}$ as

$$\bar{r} = \text{mean}(r_1, r_2, ..., r_{N_m}).$$

This assumption allows us to rewrite the denominator in Equation (7.4) as

$$\sum_{m=1}^{N_m} \frac{\mathbf{w}_{j,m}}{r_m} \approx \frac{1}{\bar{r}}.$$

and, the forward model in Equation (7.2) using Equation (7.3) becomes

$$\mathbf{b}_\theta^i = \sum_{e=1}^{n_e} s_e \exp\left(-\sum_{j=1}^{N_v} a_\theta^{ij} \sum_{m=1}^{N_m} \mathbf{w}_{j,m} \bar{r} \left(\frac{u}{r}\right)_{m,e} \right).$$

Note, that since $\bar{r} \approx r_m$ for al materials $m = 1, 2, ..., N_m$ we have that

$$\left(\frac{u}{r}\right)_{m,e} = \frac{u_{e,m}}{r_m} \approx u_{e,m}$$

and the forward projection model above becomes

$$\mathbf{b}_\theta^i = \sum_{e=1}^{n_e} s_e \exp\left(-\sum_{j=1}^{N_v} a_\theta^{ij} \sum_{m=1}^{N_m} \mathbf{w}_{j,m} u_{m,e} \right). \tag{7.5}$$

Equation 7.5 is the forward projection model for one incident angle assuming the presence of multiple attenuating materials of similar densities inside the object. This model is an appropriate forward model for the digital tomosynthesis problem when the object is composed of an arbitrary number of similar materials, like adipose and glandular tissue. It is important to note that this model is a simplification of the multi-material model for attenuation presented in Chapter 6 and a generalization of the linear attenuation model presented in Chapter 4.

As in the previous chapter, without loss of generality we eliminate the variable $\mathbf{w}_{j,1}$ from the forward projection using

$$\mathbf{w}_{j,1} = 1 - \sum_{m=2}^{N_m} \mathbf{w}_{j,m}$$

and the simplified multi-material attenuation function becomes

$$\boldsymbol{\mu}_{e,j} = u_{1,e} + \sum_{m=2}^{N_m} \mathbf{w}_{j,m} \left(u_{m,e} - u_{1,e} \right). \tag{7.6}$$

This new attenuation function gives us the new forward projection model for incident projection angle θ

$$\mathbf{b}_\theta^i = \sum_{e=1}^{n_e} s_e \exp \left[- \sum_{j=1}^{N_v} a_\theta^{ij} \left(u_{1,e} + \sum_{m=2}^{N_m} \mathbf{w}_{j,m} \left(u_{m,e} - u_{1,e} \right) \right) \right] \tag{7.7}$$

where $u_{e,m}$ represents the linear attenuation coefficient of the material m for the energy level e.

7.2 Statistical Considerations

In the same way as the reconstruction process for the multi-material attenuation model described in Chapter 6, we consider the statistical properties of the image acquisition process. The observed data for pixel i of projection θ is modeled as the realization of independently distributed Poisson random variables with additional background noise. Given the forward model in Equation (7.7) the expected value of the measured data for pixel i at angle θ is

$$E(\mathbf{b}_\theta^i; \mathbf{w}) = \sum_{e=1}^{n_e} s_e \exp \left[- \sum_{j=1}^{N_v} a_\theta^{ij} \left(u_{1,e} + \sum_{m=2}^{N_m} \mathbf{w}_{j,m} \left(u_{m,e} - u_{1,e} \right) \right) \right] + \bar{\boldsymbol{\eta}}^i \tag{7.8}$$

where $u_{m,e}$ is the linear attenuation for the m-th material at energy level e, $\mathbf{w}_{j,m}$ denotes a *scalar* found as the j-th entry of the vector containing the weight fractions for material m, and the variable $\mathbf{w} = \{\mathbf{w}_{2,:}, ..., \mathbf{w}_{N_m,:}\}$ is the set of vectors representing the weight fractions for the $N_m - 1$ materials included in the model. If we let $E(\mathbf{b}_\theta^i) = \bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i$ we have that

each pixel i in the observed projection θ follows the Poisson distribution given by

$$\mathbf{b}_\theta^i \sim \text{Poisson}\left(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i\right).$$

Using the theory of maximum likelihood estimation, we know that the probability of observing projection $\mathbf{b}_\theta$ for all θ given the set of weight fractions $\mathbf{w} = \{\mathbf{w}_{:,2}, ..., \mathbf{w}_{:,N_m},\}$ for the object is determined by the Poisson likelihood function

$$p(\mathbf{b}_\theta; \mathbf{w}) = \prod_{i=1}^{N_p} \frac{e^{-(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)^{\mathbf{b}_\theta^i}}{\mathbf{b}_\theta^i!} \tag{7.9}$$

with $\mathbf{b}_\theta$ being the observed data and N_p representing the total number of pixels in the detector. As in the previous chapters, our goal is to find the set of weight fractions $\mathbf{w}$ that maximize the likelihood function above. The set of weight fractions that maximize Equation (7.9) represent the volume that most likely produced the observed projection $\mathbf{b}_\theta$ and thus, our reconstructed volume.

7.3 The Iterative Reconstruction Framework

Given the forward model in Equation (7.7) we are interested in reconstructing the vectors $\mathbf{w}_m$ for each material $m = 2, 3, ..., N_m$ representing the unknown weight fractions of the object. Here we follow the same maximum likelihood estimator approach outlined in Chapter 6, modified to fit the image acquisition model in Equation (7.7). We begin by formulating the reconstruction problem as a numerical optimization problem where we try to find the set of weight fractions $\mathbf{w}$ that maximize the likelihood function in Equation (7.9). This set of weight fractions corresponds to the volume that was most likely to produce the observed projection data $\mathbf{b}_\theta^i$.

By the monotonicity of the logarithm function, the problem of finding the maximizer of Equation (7.9) is equivalent to finding the value of $\mathbf{w}$ that minimizes the negative log-

likelihood function given by

$$\begin{aligned} -L(\mathbf{b}_\theta, \mathbf{w}) &= -\log(p(\mathbf{b}_\theta; \mathbf{w})) \\ &= \sum_{i=1}^{N_p} (\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i) - \mathbf{b}_\theta^i \log(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i) + c \end{aligned} \tag{7.10}$$

where c is a constant. The reconstruction problem is formulated as finding the maximum likelihood estimator $\mathbf{w}_{MLE}$ that solves the optimization problem

$$\mathbf{w}_{MLE} = \arg \min_{\mathbf{w}} \left\{ \sum_{\theta=1}^{N_\theta} -L(\mathbf{b}_\theta, \mathbf{w}) \right\}. \tag{7.11}$$

In this simplified multi-material framework, we are still reconstructing a set of $N_m - 1$ unknown vectors that represent the weight fractions of each material modeled as in Chapter 6, but by the simplified forward model in Equation (7.7) our cost function has easier derivatives to compute. This allows us to analyze the Hessian of the cost function and write the derivatives in terms of the raytracing matrix A_θ. In this section we derive the necessary functions to solve the optimization problem in Equation (7.11) using a gradient descent and Newton method.

7.3.1 Gradient Descent

Recall from the previous chapters that the gradient descent iteration is given by

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} - \alpha_k \nabla L(\mathbf{b}_\theta, \mathbf{w}^{(k)}) \tag{7.12}$$

where α_k is the step length for each iteration k and the step direction $\nabla L(\mathbf{b}_\theta, \mathbf{w}^{(k)})$ is the derivative of the cost function evaluated at the current iterate. Similar to the multi-material attenuation model, the step direction is given as the derivative of the cost function with respect to all $(N_m - 1) \times N_v$ unknowns, namely

$$\nabla L(\mathbf{b}_\theta, \mathbf{w}^{(k)}) = \frac{\partial}{\partial \mathbf{w}_{j,m}} \left(-L(\mathbf{b}_\theta, \mathbf{w}^{(k)}) \right)$$

where the variable $m = 2, 3, ..., N_m$ represents the number of materials and the variable $j = 1, 2, ..., N_v$ corresponds to the voxels. We then have that

$$\frac{\partial(-L(\mathbf{b}_\theta, \mathbf{w}^{(k)}))}{\partial \mathbf{w}_{j,m}} = \sum_{i=1}^{N_p} \left(1 - \frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}\right) \frac{\partial(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}{\partial \mathbf{w}_{j,m}}. \tag{7.13}$$

Using Equation (7.8) we know that the derivative of the expected value for each pixel i in projection θ with respect to the j-th voxel of the m-th material is

$$\frac{\partial(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}{\partial \mathbf{w}_{j,m}} = -a_\theta^{ij} \sum_{e=1}^{n_e} s_e \left(u_{m,e} - u_{1,e}\right) \exp\left(-\sum_{j=1}^{N_v} a_\theta^{ij} \boldsymbol{\mu}_{e,j}\right). \tag{7.14}$$

This gives the $(m-2) \times N_v + j$ entry of the step direction vector as

$$\frac{\partial(-L(\mathbf{b}_\theta, \mathbf{w}^{(k)}))}{\partial \mathbf{w}_{j,m}} = \sum_{i=1}^{N_p} a_\theta^{ij} \left(\frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_i^\theta + \bar{\boldsymbol{\eta}}^i)} - 1\right) \sum_{e=1}^{n_e} s_e \left(u_{m,e} - u_{1,e}\right) \exp\left(-\sum_{j=1}^{N_v} a_\theta^{ij} \boldsymbol{\mu}_{e,j}\right).$$

In terms of the raytracing matrix A_θ, we can express the derivative of the cost function $\nabla L(\mathbf{b}_\theta, \mathbf{w}^{(k)})$ as a backprojection operation by

$$\nabla(-L(\mathbf{b}_\theta, \mathbf{w})) = \begin{bmatrix} A_\theta^T \mathbf{v}_2^{(k)} \\ \vdots \\ A_\theta^T \mathbf{v}_{N_m}^{(k)} \end{bmatrix} \tag{7.15}$$

where the i-th entry of the vector $\mathbf{v}_m^{(k)}$ is given by

$$\mathbf{v}_m^{(k)}(i) = \left(\frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)} - 1\right) \sum_{e=1}^{n_e} s_e \left(u_{m,e} - u_{1,e}\right) \exp\left(-\sum_{j=1}^{N_v} a_\theta^{ij} \boldsymbol{\mu}_{e,j}\right) \tag{7.16}$$

with

$$\boldsymbol{\mu}_{e,j} = u_{1,e} + \sum_{m=2}^{N_m} \mathbf{w}_{j,m}^{(k)} \left(u_{m,e} - u_{1,e}\right).$$

To solve Equation (7.11) using a gradient descent method, the framework is as follows

Algorithm 7.3.8 Gradient Descent Algorithm For the Simplified Multi-material Attenuation Model

1: Choose the initial set of volumes representing the weight fractions for each material given by $\mathbf{w}^{(0)} = \begin{bmatrix} \mathbf{w}_{:,2}^{(0)} \\ \vdots \\ \mathbf{w}_{:,N_m,}^{(0)} \end{bmatrix}$

2: **for** $k = 1, 2, ...$ until converge **do**

3: Evaluate the forward model using Equation (7.8)

4: Compute the vectors $\mathbf{v}_m^{(k)}$ for all materials

5: Find the derivative $\nabla L(\mathbf{b}_\theta, \mathbf{w}^{(k)}) = \begin{bmatrix} A_\theta^T \mathbf{v}_2^{(k)} \\ \vdots \\ A_\theta^T \mathbf{v}_{N_m}^{(k)} \end{bmatrix}$ for all θ

6: Determine the step length α_k

7: Update the solution $\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} - \alpha_k \nabla L(\mathbf{b}_\theta, \mathbf{w}^{(k)})$

8: **if** $\sum_{\theta=1}^{N_\theta} L(\mathbf{b}_\theta, \mathbf{w}^{(k)}) > \sum_{\theta=1}^{N_\theta} L(\mathbf{b}_\theta, \mathbf{w}^{(k-1)})$ **then**

9: Refine parameter α_k; no sufficient descent

10: **end if**

11: **end for**

The gradient descent method is known to converge very slowly for some problems but as is shown in Chapter 11, for digital breast tomosynthesis reconstructions only a few iterations of this method are needed (less than twenty in most cases) to produce quality reconstructions. We choose to solve the reconstruction problem using this method because of the computational intensity of the higher derivatives of the cost function. In numerical experiments it has been noted that a step length $\alpha_k < 1$ produces steady descent with few line search iterations. For this problem, the method uses regularization by early termination since later iterates begin to be corrupted by noise (see Section 1.1). As in the case for the quadratic model for attenuation, convergence is determined as the point where the relative function value stagnates.

7.3.2 Newton Approach

Although a working implementation of a Newton method for the simplified multi-material attenuation model is not presented in this thesis, the numerical aspects of using a Newton approach are described in this section. As discussed in Section 5.3.2, a Newton method is a line search strategy with search direction given by the second derivative term in the Taylor series expansion of the cost function. This search direction has a much faster rate of convergence than the gradient descent method (typically quadratic) but in order to guarantee descent of the cost function, the Hessian matrix must be symmetric positive definite. A Newton iteration takes the form

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \alpha_k \mathbf{s}^{(k)}$$

where α_k is the step length and the step direction $\mathbf{s}^{(k)}$ is the solution to the linear system

$$\mathbf{H}(\mathbf{w}^{(k)})\mathbf{s}^{(k)} = -\nabla L(\mathbf{b}_\theta, \mathbf{w}^{(k)}).$$

In order to compute the Hessian of the cost function, note that the first derivative of the cost function is given in Equation (7.13) as

$$\frac{\partial(-L(\mathbf{b}_\theta, \mathbf{w}))}{\partial \mathbf{w}_{j,m}} = \sum_{i=1}^{N_p} \left(1 - \frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}\right) \frac{\partial(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}{\partial \mathbf{w}_{j,m}}.$$

for $j = 1, 2, .., N_v$ and $m = 2, 3, ..., N_m$. The Hessian in this case can be expressed as a block matrix, with each block having size $N_v \times N_v$ and indexed (m, n) where $m, n = 2, 3, ..., N_m$. Therefore, for $\ell = 1, 2, ..., N_v$ the (j, ℓ) entry of the (m, n) block of the Hessian is given as

$$h_{m,n}^{j\ell} = \sum_{i=1}^{N_p} \frac{\partial}{\partial \mathbf{w}_{\ell,n}} \left(1 - \frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}\right) \frac{\partial(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}{\partial \mathbf{w}_{j,m}} + \left(1 - \frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}\right) \frac{\partial}{\partial \mathbf{w}_{\ell,n}} \left(\frac{\partial(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}{\partial \mathbf{w}_{j,m}}\right).$$

The first term in the summation for each entry in the Hessian matrix is given by

$$\frac{\partial}{\partial \mathbf{w}_{\ell,n}} \left(1 - \frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}\right) \frac{\partial(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}{\partial \mathbf{w}_{j,m}} = \frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)^2} \frac{\partial(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}{\partial \mathbf{w}_{\ell,n}} \frac{\partial(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}{\partial \mathbf{w}_{j,m}} \tag{7.17}$$

which, using Equation (7.14), is equal to

$$a_\theta^{i\ell} a_\theta^{ij} \frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i))^2} \sum_{e=1}^{n_e} s_e c_{n,e} \exp\left(-\sum_{j=1}^{N_v} a_\theta^{ij} \boldsymbol{\mu}_{e,j} \right) \sum_{e=1}^{n_e} s_e c_{m,e} \exp\left(-\sum_{j=1}^{N_v} a_\theta^{ij} \boldsymbol{\mu}_{e,j} \right)$$

where the values $c_{n,e} = (u_{n,e} - u_{1,e})$ and $c_{m,e} = (u_{m,e} - u_{1,e})$ represent the difference between the linear attenuation coefficient at energy level e for material n (m) and the first material.

The second term in the sum for each entry of the Hessian is computed using Equation (7.14) to obtain

$$\left(1 - \frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}\right) \frac{\partial^2 (\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}{\partial \mathbf{w}_{\ell,n} \partial \mathbf{w}_{j,m}} = a_\theta^{i\ell} a_\theta^{ij} \left(1 - \frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}\right) \sum_{e=1}^{n_e} s_e c_{m,e} c_{n,e} \exp\left(-\sum_{j=1}^{N_v} a_\theta^{ij} \boldsymbol{\mu}_{e,j} \right)$$

for $j, \ell = 1, 2, ..., N_v$ and $m, n = 2, 3, ..., N_m$. Therefore, the (j, ℓ) entry of the (m, n) block of the Hessian matrix is given by the equation

$$\begin{aligned} h_{m,n}^{j\ell} = & \sum_{i=1}^{N_p} a_\theta^{i\ell} a_\theta^{ij} \frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i))^2} \sum_{e=1}^{n_e} s_e c_{n,e} \exp\left(-\sum_{j=1}^{N_v} a_\theta^{ij} \boldsymbol{\mu}_{e,j} \right) \sum_{e=1}^{n_e} s_e c_{m,e} \exp\left(-\sum_{j=1}^{N_v} a_\theta^{ij} \boldsymbol{\mu}_{e,j} \right) \\ & + \sum_{i=1}^{N_p} a_\theta^{i\ell} a_\theta^{ij} \left(1 - \frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}\right) \sum_{e=1}^{n_e} s_e c_{m,e} c_{n,e} \exp\left(-\sum_{j=1}^{N_v} a_\theta^{ij} \boldsymbol{\mu}_{e,j} \right). \end{aligned}$$

In matrix vector notation, we can write each the (m, n) block of the Hessian matrix as

$$H_{m,n} = A_\theta^T \Omega_{m,n}^{(k)} A_\theta \tag{7.18}$$

for $m, n = 2, 3, ..., N_m$ where A_θ is the raytracing matrix and the matrix $\Omega_{m,n}$ is a diagonal matrix whose entries are given by

$$\begin{aligned} \boldsymbol{\omega}_{m,n}^{(k)}(i) = & \frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)^2} \sum_{e=1}^{n_e} s_e c_{n,e} \exp\left(-\sum_{j=1}^{N_v} a_\theta^{ij} \boldsymbol{\mu}_{e,j} \right) \sum_{e=1}^{n_e} s_e c_{m,e} \exp\left(-\sum_{j=1}^{N_v} a_\theta^{ij} \boldsymbol{\mu}_{e,j} \right) \\ & + \left(1 - \frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)}\right) \sum_{e=1}^{n_e} s_e c_{m,e} c_{n,e} \exp\left(-\sum_{j=1}^{N_v} a_\theta^{ij} \boldsymbol{\mu}_{e,j} \right). \end{aligned} \tag{7.19}$$

In this thesis we only provide the numerical details for the Newton reconstruction framework and a functional implementation is left as future work. The process to solve Equation (7.11) using a Newton method is as follows

---

Algorithm 7.3.9 Newton Framework for the Simplified Multi-material Attenuation Model

---

1: Choose the initial set of volumes representing the weight fractions for each material given by $\mathbf{w}^{(0)} = \begin{bmatrix} \mathbf{w}_{:,2}^{(0)} \\ \vdots \\ \mathbf{w}_{:,N_m,}^{(0)} \end{bmatrix}$

2: **for** $k = 1, 2, ...$ until converge **do**

3: Evaluate the forward model using Equation (7.8)

4: Compute the vectors $\mathbf{v}_m$ for $m = 2, 3, ..., N_m$ using Equation (7.16)

5: Build the gradient vector $\nabla L(\mathbf{b}_\theta, \mathbf{w}^{(k)}) = \begin{bmatrix} A_\theta^T \mathbf{v}_2^{(k)} \\ \vdots \\ A_\theta^T \mathbf{v}_{N_m}^{(k)} \end{bmatrix}$ for all θ

6: Compute the vectors $\boldsymbol{\omega}_{m,n}$ for $m, n = 2, 3, ..., N_m$ using Equation (7.19)

7: Build the Hessian matrix $\mathbf{H}(\mathbf{w}^{(k)})$ using Equation (7.18)

8: Find the step direction $\mathbf{s}^{(k)}$ by solving the linear system $\mathbf{H}(\mathbf{w}^{(k)})\mathbf{s}^{(k)} = -\nabla L(\mathbf{b}_\theta, \mathbf{w}^{(k)})$

9: Determine the step length α_k

10: Update the solution $\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} - \alpha_k \mathbf{w}^{(k)}$

11: **if** $\sum_{\theta=1}^{N_\theta} L(\mathbf{b}_\theta, \mathbf{w}^{(k)}) > \sum_{\theta=1}^{N_\theta} L(\mathbf{b}_\theta, \mathbf{w}^{(k-1)})$ **then**

12: Refine parameter α_k; no sufficient descent

13: **end if**

14: **end for**

For this approach the natural Newton method step length $\alpha_k = 1$. The number of inner iterations to solve the linear system that determines the step direction depends on the conditioning of the Hessian matrix. Similarly to the gradient descent approach described earlier in the chapter, in the Newton method regularization is done by early termination since the later iterates begin to be corrupted by noise as is the case for ill-posed problems (see Section 1.1). Convergence is determined once the relative value of the function stagnates.

7.4 Summary

In this chapter we have presented a simplified version of the multi-material forward model for the polyenergetic digital tomosynthesis reconstruction problem used in breast cancer screening. The simplified model is derived by assuming the presence of similar attenuating materials inside the breast, that is, materials that have similar density. This assumption is physically sound, since the breast is composed of adipose tissue, glandular tissue, and skin, all materials that have similar density. Some of the benefits of simplifying the multi-material model presented in Chapter 6 is that we have easier derivatives, we can write the Hessian matrix in terms of the raytracing matrix, and we can establish conditions for the convexity of the cost function. Keep in mind that in this simplified multi-material model, we still formulate the attenuation function of the object in terms of the weight fractions of an arbitrary number of N_m materials present inside the object, giving us the advantage of modeling an arbitrary number of materials for the problem. This model also allows us to expand both the linear attenuation model presented in Chapter 4 and the quadratic model for attenuation presented in Chapter 5 in order to capture additional materials of interest in breast cancer screening, like skin and solutions used for contrast of lesions. In addition, by keeping the formulation of the model independent of glandular fractions, we can extend the reconstruction framework to tomosynthesis imaging of other objects or parts of the body.

So far in this thesis, we have presented four forward models for the polyenergetic digital tomosynthesis problem for breast cancer imaging. As will be shown by the numerical results in Chapter 11, these models have improved image quality and less artifacts than the reconstructions provided by MLEM and filtered backprojection. In the next chapters, we discuss the computational science considerations needed to implement an effective reconstruction framework to solve the digital tomosynthesis problem for clinical size data sets.

Chapter 8

Implementation Considerations

The previous chapters have described the four different forward models for the image acquisition process we propose in this thesis as well as a gradient descent and Newton framework to solve the digital tomosynthesis reconstruction problem. As we will see in the next chapters, the implementation of a gradient descent or Newton approach is nontrivial given the size of the problem for clinical tomosynthesis reconstructions. In this chapter, we evaluate the computational intensity of the problem and discuss our motivation to use a GPU for the implementation. We describe the basics of GPU computing using OpenCL as the API as an introduction to the kernel design and implementation optimizations described in the next two chapters.

8.1 Computational Intensity

From a numerical analysis standpoint, solving the digital tomosynthesis reconstruction problem as proposed in this work is ultimately solving the optimization problem given by

$$\mathbf{w}_{MLE} = \arg \min_{\mathbf{w}} \left\{ \sum_{\theta=1}^{N_\theta} -L(\mathbf{b}_\theta, \mathbf{w}) \right\} \tag{8.1}$$

where the unknowns are the set of vectors corresponding to the weight fractions of each material $\mathbf{w} = \{\mathbf{w}_1, ..., \mathbf{w}_n\}$, and n depends on the chosen forward model. In theory, the problem is relatively simple to solve, given that the analytical first and second derivatives

of the cost function $L(\mathbf{b}_\theta, \mathbf{w})$ are known. In reality, however, finding a solution to this very large scale optimization problem is a challenge, and a creative implementation scheme is necessary in order to produce timely reconstructions using a reasonable amount of computational resources.

We would like to develop an implementation of the reconstruction process that allows healthcare facilities to produce a tomosynthesis reconstruction for one patient in-house or using an architecture that is readily available. Our goal is to produce better quality digital tomosynthesis reconstructions in a short amount of time that is comparable to the time it takes to produce a reconstruction using the filtered backprojection or MLEM approaches. Therefore, we must be able to solve the optimization problem in Equation (8.1) in a matter of minutes using off-the-shelf computing architectures.

The difficulty in solving Equation (8.1) is the size of the problem. Consider using the simplified multi-material model described in Chapter 7 as the forward model. We have that each function $L(\mathbf{b}_\theta, \mathbf{w})$ is given by

$$-L(\mathbf{b}_\theta, \mathbf{w}) = \sum_{i=1}^{N_p} (\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i) - \mathbf{b}_\theta^i \log(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i) + c$$

where c is a constant and, if we model three attenuating materials, the value $\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i$ is given by the equation

$$\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i = \sum_{e=1}^{n_e} s_e \exp\left[- \left(u_{2,e}\ \mathbf{a}_\theta^{i:} \bullet \mathbf{w}_{:,2} + u_{3,e}\ \mathbf{a}_\theta^{i:} \bullet \mathbf{w}_{:,3} + u_{1,e}\ \mathbf{a}_\theta^{i:} \bullet \mathbf{1} \right) \right] + \bar{\boldsymbol{\eta}}^i.$$

Therefore, for one single evaluation of the cost function in Equation (8.1) we have that

1. We must evaluate $L(\mathbf{b}_\theta, \mathbf{w})$ for all projection angles N_θ, where N_θ is typically 15-45 for a clinical size problem. In our numerical experiments we use $N_\theta = 15$.

2. To evaluate each $L(\mathbf{b}_\theta, \mathbf{w})$ we need to evaluate the expected value $\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i$ for all pixels N_p. In our clinical data the size of the detector is 1280×2048 meaning 2.6×10^6 pixels.

3. To evaluate $\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i$ for each pixel i we need to perform two operations:

    a. A dot product of the i-th row of the raytracing matrix A_θ with each of the vectors

containing the weight fractions and a vector of ones. Note that the size of each of the vectors in this dot product is the total number of voxels in the reconstructed volume N_v, which is about 1.3×10^8 for our clinical size problems.

b. We also need to perform a summation operation over all discretized energy levels N_e. In our clinical size problems, the number of energy levels typically ranges between 40 and 70.

Therefore, in order to do a single evaluation of Equation (8.1), we must complete a nested loop that goes over the number of projection angles, the number of pixels in the detector, the number of discrete energy levels, and the total number of voxels that discretize the object. Furthermore, the computation of the first derivative of the cost function can be even more intensive. To find the gradient of the cost function $\nabla(-L(\mathbf{b}_\theta, \mathbf{w}))$ for all $\theta = 1, ..., N_\theta$ we need to compute

$$\nabla(-L(\mathbf{b}_\theta, \mathbf{w})) = \begin{bmatrix} A_\theta^T \mathbf{v}_2 \\ A_\theta^T \mathbf{v}_3 \end{bmatrix}$$

where the i-th entry of the vector $\mathbf{v}_m$ is given by

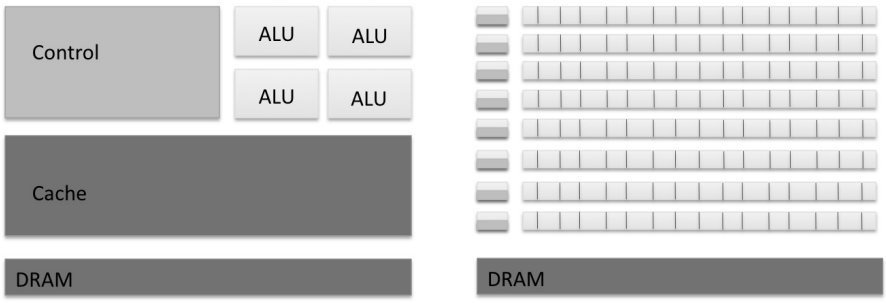
$$\mathbf{v}_m(i) = \left(\frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)} - 1 \right) \sum_{e=1}^{n_e} s_e \left(u_{m,e} - u_{1,e} \right) \exp\left(-u_{2,e} \mathbf{a}_\theta^{i:} \cdot \mathbf{w}_{:,2} - u_{3,e} \mathbf{a}_\theta^{i:} \cdot \mathbf{w}_{:,3} - u_{1,e} \mathbf{a}_\theta^{i:} \cdot \mathbf{1} \right).$$

Hence, in addition to the quantities already computed in the function evaluation, we must compute two separate loops over all energy levels N_e for the vectors $\mathbf{v}_2$ and $\mathbf{v}_3$ and two matrix transpose-vector products with each A_θ which is of size $N_p \times N_v$ or 2.6×10^6 by 1.3×10^8. Clearly, a single gradient descent iteration for this problem is both memory and computation intensive, and without the right implementation approach it can take several hours to complete.

We choose to solve the reconstruction problem for the digital tomosynthesis using a GPU to accelerate computations. GPUs are powerful cards capable of achieving hundreds of gigaflops with the right program design. In the rest of this chapter we describe the GPU architecture and the OpenCL API as an introduction to the implementation details presented in the next two chapters for the application described in this thesis.

8.2 GPU Computing

According to Nvidia, GPU computing is by definition "the use of a GPU (graphics processing unit) together with a CPU to accelerate general-purpose scientific and engineering applications" [93]. A CPU is single core or a multicore microprocessor that has been optimized for sequential performance by using advanced control logic to execute a set instructions in parallel or out-of-order. In addition, CPUs are equipped with large cache memories that hide latency for both instructions and data. In contrast, a GPU is a many-core microprocessor dedicated to arithmetic throughput, as a large number of cores work together to execute compute intensive portions of a program. The design of a GPU and a CPU is fundamentally different as shown in Figure 8.1. GPUs are built to optimize execution throughput by



(a) CPU design (b) GPU design

Figure 8.1: Architecture design of a CPU vs a GPU [79].

managing a massive number of compute intensive threads using context switching to hide latencies. This reduces the complexity of the control logic needed to handle each thread, thus maximizing the chip area dedicated to floating point computations [79].

In other words, GPUs provide hundreds of simpler cores with small caches that can execute hundreds of flop intensive threads in parallel, as opposed to CPUs which provide a handful of sophisticated processors that deliver optimized performance for sequential programs. Each architecture provides its own benefits and a program that is well suited to run on a CPU does not necessarily perform well on a GPU and vice-versa. This is why in GPU computing, a significant portion of the software development process is spent in detecting the compute intensive operations and extracting the parallelism from the application.

In this thesis we use the Nvidia Tesla^{TM} C2070 GPU [92] for our numerical experiments. The specifics of this Tesla card include 448 CUDA cores with a frequency of 1.15GHz that achieve a theoretical peak of 515 Gflops in double precision and 1.03 Tflops in single precision. In addition, the card has 6GB of global memory at a memory speed of 1.5GHz and bandwidth of 144GB/sec. The CPU we used is the Intel Xeon(R) X5472 @ 3GHz.

8.3 Programming using OpenCL

To develop our GPU implementation we use the Open Computing Language (OpenCL) which was developed and is mantained by the Khronos Group [110]. OpenCL is a heterogeneous programming framework used to develop applications that execute across different types of devices that are not necessarily made by the same vendor. The framework can be seen as an alternative to CUDA [88] which was developed by Nvidia to build applications that run on their branded platforms. We choose to develop the application in this thesis using OpenCL as opposed to CUDA because of the portability that the OpenCL framework provides. Our goal is to ultimately allow healthcare facilities to run the digital tomosynthesis reconstruction application we developed in whatever device is available to them. In this section we describe the basic details of the OpenCL framework and the particular considerations needed to design the kernel functions presented in the next two chapters. For an extensive discussion on programming GPUs using OpenCL see [51] or programming using CUDA see [79].

We begin our overview of OpenCL by describing the four parts (or models) of the specification as presented in [51]. The first model, called the *platform model*, provides an abstraction of the hardware. The model identifies the *host* which is a single CPU that coordinates the execution of OpenCL code in one or more *devices*. As OpenCL is designed for heterogeneous computing, the devices may be a combination of CPUs and GPUs that are capable of executing OpenCL C code. In our application, the platform model sets up the host to coordinate the execution of functions in a single GPU.

The second model is the *execution model* and it manages the execution of OpenCL C code in the device. This is the portion of the application responsible for establishing the

number of threads (called *work-items*) that will be executed in a given topology. It is through the execution model that host to device communication is coordinated. Another very important part of the execution model is the *kernel*, which is the part of the OpenCL program that actually executes on the device. The kernel is an instance of the parallel operation and will be executed by each work-item enqueued. That is, to run the most computationally intensive portions of a program in a GPU, the operation needs to be divided into equal tasks that can be identified by a unique index to be executed by an individual thread. The task to be executed by the work-item is the kernel. As we will see in the next chapters, the kernels we use have a very specific purpose, are arithmetic intense, require few global memory accesses, and do not communicate. This is a key aspect of kernel design, work-items cannot communicate with each other unless they belong to a subgroup. The execution model is thus responsible for loading the memory needed to run computations on the device, designing the execution of the multiple work-items, and transferring the results back to the CPU once the kernel execution is completed.

The third model for the specification is called the *memory model* and defines the abstract memory hierarchy of the execution. The hierarchy begins with *global* memory which is memory that can be accessed by all work-items. Global memory is the slowest type of memory to each processor and is the area where the data transferred from the host to the device resides until it is explicitly moved elsewhere. The second type of memory is called *local* memory and it is accessible to a specified group of work-items called a *workgroup*. The size and dimension of each workgroup is specified in the execution model. Local memory has a much shorter latency and higher bandwidth than global memory. The third type of memory in the hierarchy is called *private* memory and it is accessible to one individual work-item. This is usually register data that has the fastest access to the processor. In addition, there is a fourth type of memory called *constant* memory that is accessible to all work-items but is designed for read-only data that is accesses by all work-items at once. Constant memory resides in part of the global memory.

The last model in the specification is the *programming model* which describes how the threads are physically executed in the hardware. This model is architecture dependent as each device, be it a GPU or a CPU, has a different number of processing elements to execute

the work-items.

The most important part to understand from this discussion of GPU computing and OpenCL is that the multi-threaded parallelism of a GPU is fundamentally different from the multi-threading parallelism in a CPU. A multi-threaded program in a CPU can either run multiple instances of a function using a handful of very sophisticated cores, or it can utilize complex control logic to execute a single program in parallel by issuing sequential instructions to the multiple cores in or out of order. In contrast, a GPU runs a program in parallel by executing a user defined kernel hundreds of times by its hundreds of simpler cores. Thus, in GPU programming, the responsibility of finding the right compute intensive portions of the program lies in the developer's understanding of the application.

In the next two chapters we describe the most computationally intensive portions of the digital tomosynthesis reconstruction framework and redesign the algorithms needed to perform these operations to make them GPU friendly. For every kernel we design, we enforce that it does not require too many local variables since the processing unit executing the task will have limited fast memory and that every instance of a kernel be self-contained or completely independent of any neighboring work-item. This allows us to solve the reconstruction problem to produce clinical size reconstructions in minutes. Compared to serial runs, using a GPU allows our implementation to obtain speed-ups of up to $30\times$ for most problems.

Chapter 9

Trace Functions

In the forward models for the digital tomosynthesis problem proposed in this thesis, we must analyze how the x-ray beam travels through the object to reach the detector and generate the photon counts measured by the imaging system. To establish the behavior of the x-ray beam inside the object we need to take into account the physical composition of the object and the geometry of the ray. If we discretize the 3D object into voxels with individual attenuation coefficients, we can follow the ray through each voxel and determine the level of absorption of energy from the ray. The attenuation coefficients for the given object can be known a priori but the length and direction of the ray as it travels through each voxel in the object is completely determined by the geometry of the imaging system at the time of the image acquisition. Determining the length of the ray through each of the voxels inside the discretized object is know in the imaging and radiation therapy community as raytracing and can be done using different algorithms.

9.1 Siddon's Algorithm for the Exact Radiological Path

In 1985, Robert Siddon [105] formulated an exact and efficient algorithm to calculate the radiological path through a 3D CT array. The radiological path is defined as

$$d = \sum_i \sum_j \sum_k \ell(i,j,k)\rho(i,j,k) \tag{9.1}$$

where voxel (i, j, k) has respective density $\rho(i, j, k)$ and $\ell(i, j, k)$ is the length of the ray traveled inside the voxel. For this problem, Siddon's algorithm scales with the sum of linear dimensions of the CT array as opposed to the number of terms in the sum if d were to be calculated directly.

The general idea in Siddon's algorithm is to consider the individual voxels in the discretization of the 3D object as intersecting equidistant orthogonal planes. Therefore, we can interpret the intersection of the ray with an individual voxel as the intersection of the ray with two separate planes. This allows us to compute the intersection of the ray with all planes recursively given the initial intersection point. Figure 9.1 shows this idea in two dimensions. Note that using this approach, we no longer consider the ray intersecting a

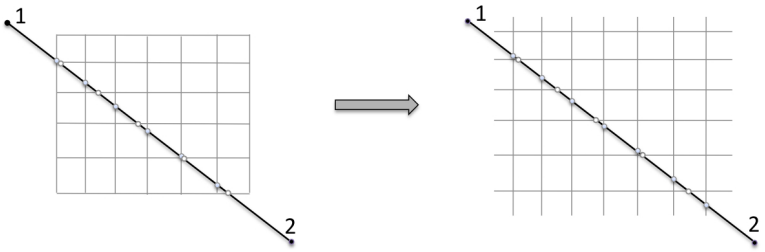


Figure 9.1: In two dimensions, transforming voxels to equally spaced orthogonal planes

voxel but we consider the ray's intersection with three types of planes (x-planes, y-planes, z-planes) where the intersections for each type of plane are independent of one another.

The parametric representation of the ray starting at point 1 and ending at point 2 is given by the set of equations

$$\begin{aligned} x(\alpha) &= x_1 + \alpha(x_2 - x_1) \\ y(\alpha) &= y_1 + \alpha(y_2 - y_1) \\ z(\alpha) &= z_1 + \alpha(z_2 - z_1) \end{aligned} \tag{9.2}$$

where $\alpha = 0$ at point 1, $\alpha = 1$ at point 2, and the triplet $(x(\alpha), y(\alpha), z(\alpha))$ gives the location of point α along the ray. To use this framework, we need to consider the four possibilities for the intersection of the ray with the CT array representing the discretized object (shown in Figure 9.2).

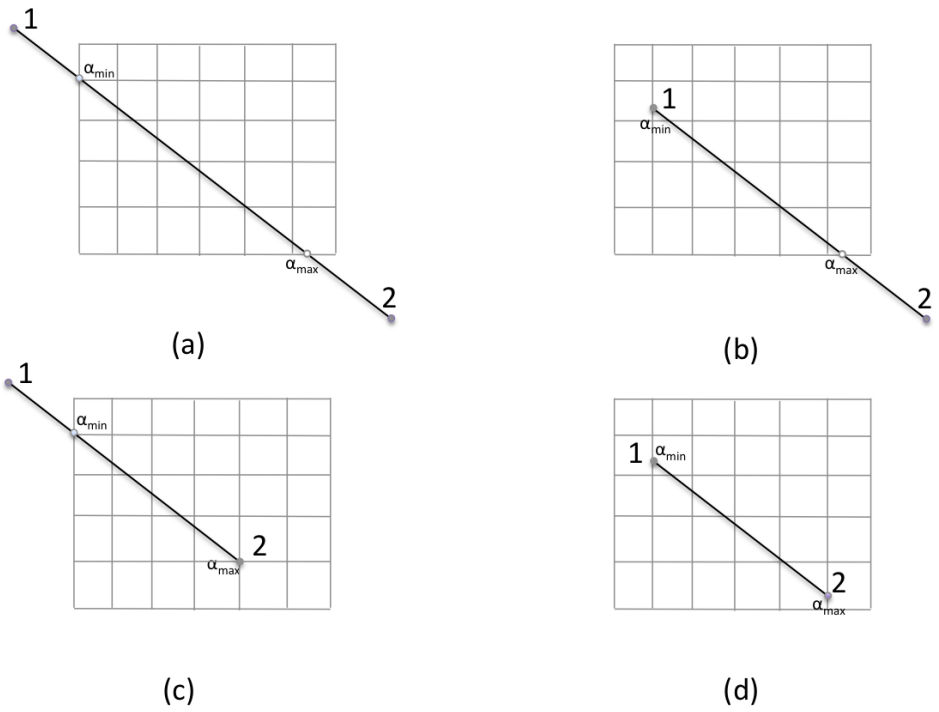


Figure 9.2: Possible intersections of the ray with the CT array

The possibilities of intersection are:

(a) Point 1 and point 2 are outside the CT array, so the ray begins and ends outside the object. In this case, the voxels that have an intersection with the ray must have parametric value $\alpha \in [\alpha_{min}, \alpha_{max}]$.

(b) Point 1 is inside the CT array and point 2 is outside the CT array. That is, the ray begins inside the object and ends outside of it. The intersected voxels for this case must have parametric value $\alpha \in [0, \alpha_{max}]$.

(c) Point 1 is outside the CT array and point 2 is inside the CT array. Here we have that the ray begins outside the object and ends inside the object. In this case we have that the parametric value for the voxels intersected is $\alpha \in [\alpha_{min}, 1]$.

(d) The last case is when point 1 and point 2 are both inside the CT array, which happens when the ray is completely contained inside the object. The voxels intersected in this case have parametric value $\alpha \in [0, 1]$.

For the geometry of the digital tomosynthesis problem the source is always outside the

object, so we are either in case (a) or case (c), depending on the space between the object and the detector.

Given the above set up, Siddon's algorithm computes the intersection of the ray with the individual voxels in several steps. First, we determine the range for the parametric values of α and calculate the sets $\{\alpha_x\}, \{\alpha_y\}, \{\alpha_z\}$ for each type of plane. Next, we merge the three sets $\{\alpha_x\}, \{\alpha_y\}, \{\alpha_z\}$ in ascending order and append the computed values α_{min} and α_{max} to the set. The length of the ray, given in units of ray length, contained in a particular voxel is obtained by finding the difference between adjacent α values in the merged sets. Then for each length, we find the index (i, j, k) of the corresponding voxel and sum the products of all lengths and voxel densities to determine the radiological path length d.

For a CT array with $(N_x - 1, N_y - 1, N_z - 1)$ voxels, the orthogonal planes (equidistant in parallel) can be written recursively as

$$\begin{aligned} x_{plane}(i) &= x_{plane}(1) + (i-1)d_x \\ y_{plane}(j) &= y_{plane}(1) + (j-1)d_y \\ z_{plane}(k) &= z_{plane}(1) + (k-1)d_z \end{aligned}$$

where d_x, d_y, d_z are the distances between the planes along each spatial dimension. Note that these values also correspond to lengths of the side of the voxels. We also denote the coordinates for point 1 by (x_1, y_1, z_1) and point 2 by (x_2, y_2, z_2).

To begin the recursion in the algorithm, we must find the parametric values for the initial and final intersection of the ray with the orthogonal planes, given by α_{min} and α_{max}. The values are

$$\alpha_{min} = \max(0, \min[\alpha_x(1), \alpha_x(N_x)], \min[\alpha_y(1), \alpha_y(N_y)], \min[\alpha_z(1), \alpha_z(N_z)]) \tag{9.3}$$

$$\alpha_{max} = \min(1, \max[\alpha_x(1), \alpha_x(N_x)], \max[\alpha_y(1), \alpha_y(N_y)], \max[\alpha_z(1), \alpha_z(N_z)]) \tag{9.4}$$

where $\alpha_*(1)$ is the parametric value of α for the intersection of the ray with the first $*$-plane and $\alpha_*(N_*)$ is the parametric value for the intersection of the ray with the last $*$-plane.

Using Equation (9.2), we have that $\alpha_x(1), \alpha_x(N_x)$ are given by

$$\alpha_x(1) = \frac{x_{plane}(1) - x_1}{x_2 - x_1},\ \alpha_x(N_x) = \frac{x_{plane}(N_x) - x_1}{x_2 - x_1} \tag{9.5}$$

and similarly for y and z. Note that if the parametric value for the final intersection is less than the parametric value for the initial intersection (that is, $\alpha_{max} \leq \alpha_{min}$), then the ray does not intersect the object. In addition, if any of these values $x_2 - x_1$, $y_2 - y_1$, or $z_2 - z_1$ is equal to zero, then the ray is perpendicular to the plane and the α value for this intersection is ignored.

Since we are interested in the intersection of the ray with the object, we would like to determine minimum and maximum indices of the planes that represent the object. These planes will have parametric values $(\alpha_{min}, \alpha_{max})$. The range of the indices for each type of plane is denoted by (i_{min}, i_{max}), (j_{min}, j_{max}), and (k_{min}, k_{max}). For the x-planes we have

if $x_2 - x_1 \geq 0$ **then**

$i_{min} = N_x - \frac{x_{plane}(N_x) - \alpha_{min}(x_2 - x_1) - x_1}{d_x}$,

$i_{max} = 1 + \frac{x_1 + \alpha_{max}(x_2 - x_1) - x_{plane}(1)}{d_x}$

else

$i_{min} = N_x - \frac{x_{plane}(N_x) - \alpha_{max}(x_2 - x_1) - x_1}{d_x}$,

$i_{max} = 1 + \frac{x_1 + \alpha_{min}(x_2 - x_1) - x_{plane}(1)}{d_x}$

end if

and similarly for the y-planes and z-planes.

So far, we have determined the range of indices (i_{min}, i_{max}), (j_{min}, j_{max}) and (k_{min}, k_{max}) for the planes that correspond to the object and the range of parametric values for the points where the ray intersects the object $(\alpha_{min}, \alpha_{max})$. Using this information, we can build three independent sets of parametric values $\{\alpha_x\}, \{\alpha_y\}, \{\alpha_z\}$ that represent the intersection of the ray with all the orthogonal planes in our range of interest. For the x-planes we have that the set occurs in either of the following orders:

if $x_2 - x_1 \geq 0$ **then**

$\{\alpha_x\} = \{\alpha_x(i_{min}), \alpha_x(i_{min} + 1), ..., \alpha_x(i_{max} - 1), \alpha_x(i_{max})\}$

else

$\{\alpha_x\} = \{\alpha_x(i_{max}), \alpha_x(i_{max}-1), ..., \alpha_x(i_{min}+1), \alpha_x(i_{min})\}$

end if

where $\alpha_x(i) = \frac{x_{plane}(i)-x_1}{x_2-x_1} = \alpha_x(i-1) + \frac{d_x}{x_2-x_1}$. If follows similarly for the y-planes and z-planes. Note that each term in a set $\{\alpha_*\}$ corresponds to an intersection of the ray with an x, y, or z plane and the terms in the sets are in ascending order. Therefore, if we merge the three sets $\{\alpha_x\}, \{\alpha_y\}, \{\alpha_z\}$ into $\{\alpha\}$ and append $\alpha_{min}, \alpha_{max}$ we obtain a global set containing all intersections of the ray with the discretized object. For convenience we index the merged sets as

$$\{\alpha\} = \{\alpha(0), \alpha(1), ..., \alpha(n)\}.$$

If the global $\{\alpha\}$ set is sorted in ascending order, two adjacent terms in the set will correspond to the intersection of the ray with a particular voxel.

Define the value

$$d_{12} = \sqrt{(x_2-x_1)^2 + (y_2-y_1)^2 + (z_2-z_1)^2}.$$

Then, the length of the ray traveled between two plane intersections globally labeled $m-1$ and m is given by the eqaution

$$\ell(m) = d_{12}(\alpha(m) - \alpha(m-1)).$$

The voxel corresponding to the intersections m and $m-1$ is indexed by $(i(m), j(m), k(m))$ and contains a point along the ray which is the midpoint of the two intersections of the voxel. Let the midpoint of the ray inside the voxel be given by the parametric value $\alpha_{mid} = \frac{\alpha(m)-\alpha(m-1)}{2}$, then the index for the voxel intersected is

$$\begin{aligned} i(m) &= 1 + \frac{x_1 + \alpha_{mid}(x_2-x_1) - x_{plane}(1)}{d_x} \\ j(m) &= 1 + \frac{y_1 + \alpha_{mid}(y_2-y_1) - y_{plane}(1)}{d_y} \\ k(m) &= 1 + \frac{z_1 + \alpha_{mid}(z_2-z_1) - z_{plane}(1)}{d_z}. \end{aligned} \tag{9.6}$$

The radiological path can then be written as

$$d = \sum_{n=1}^{m} \ell(m)\rho(i(m), j(m), k(m)) = d_{12} \sum_{n=1}^{m} [\alpha(m) - \alpha(m-1)]\rho(i(m), j(m), k(m))$$

where $\rho(i(m), j(m), k(m))$ is the density of the voxel $i(m), j(m), k(m)$.

Siddon's algorithm for a 3D CT array path calculation is as follows

---

Algorithm 9.1.10

1: Determine the coordinates of the start and end of the ray (x_1, y_1, z_1) and (x_2, y_2, z_2)

2: Let d_x, d_y, d_z denote the size of the voxels in the CT array along each dimension

3: Compute $\alpha_x(1), \alpha_x(N_x), \alpha_y(1), \alpha_y(N_y), \alpha_z(1)$, and $\alpha_z(N_z)$

4: Determine the parametric range $(\alpha_{min}, \alpha_{max})$

5: Find index ranges (i_{min}, i_{max}), (j_{min}, j_{max}) and (k_{min}, k_{max})

6: Find the sets $\{\alpha_x\}, \{\alpha_y\}, \{\alpha_z\}$

7: Merge the sets and sort to find the global set $\{\alpha\}$

8: Calculate the length ℓ for all voxels

9: Compute the midpoint of the ray inside all voxels $\{\alpha_{mid}\}$

10: Compute the index (i, j, k) for the voxel with the corresponding ℓ and α_{mid}

11: Sum the product of the voxel density with the ray length ℓ for all voxels

$$d = \sum_{i} \sum_{j} \sum_{k} \ell(i, j, k)\rho(i, j, k)$$

Note that a direct implementation of Algorithm 9.1.10 is naturally sequential. Siddon's algorithm spends 40% of the time finding the voxel indices $i(m), j(m), k(m)$ from the α parameters and 26% of the time sorting the global set $\{\alpha\}$.

In the digital tomosynthesis forward problem, we are interested in using an efficient version Siddon's algorithm to evaluate the polyenergetic formulation of Beer's law given in Equation (3.9). To do this we need to develop a parallel implementation of Algorithm 9.1.10 that eliminates the sorting operation and takes advantage of the resources available in the GPU architecture. In the next section we describe a task parallel version of Siddon's algorithm that is the basis of our implementation. The specific details of the connection between Siddon's algorithm and our reconstruction process are described in Section 9.3.

9.2 A Task Parallel Version of Siddon's Raytrace

Raytracing is frequently used in dose calculations for radiation therapy (or radiotherapy), where radiation beams are sent to cancerous tumors to kill malignant cells. In this context, is important to avoid directing the beam toward healthy cells, or in the very least, try to minimize the effect the radiation dose will have on these cells. This is where raytracing is needed in the radiation therapy process, to understand the effect of the radiation beam on the different materials present inside the human body near the area of the tumor. In 2009, de Greef et al. [34] developed an approach to accelerate the raytrace computations needed for radiotherapy dose calculations using a GPU. Their approach is based on Siddon's algorithm for the exact calculation of the radiological path length. We describe their algorithm as a basis for our implementation approach.

The general idea of the accelerated algorithm is the same as Siddon's: consider the discretized object as three sets of intersecting orthogonal planes and determine the intersections of the ray with each plane. The radiological path length is given by the summation of the length of the ray traveled through each voxel multiplied by the electron density of the voxel. In our tomosynthesis work, we consider the innate quantity of the voxel to be the attenuation coefficient of the material that composes the object as opposed to the electron density. A big difference between Siddon's algorithm and the algorithm developed by de Greef et al is that the latter allows for non-equidistant z-planes in the application. The CT scans used for radiotherapy treatment can be obtained with different slice thickness, so the raytrace must be adjusted for this case. Thus, the implementation in [34] relies on having an array that contains the positions of each plane in the z direction called $\underline{z}_{plane}$. However, this only affects the computation of the z-dependent variables.

The task parallelism exploited by the implementation in [34] calls for each individual GPU thread to perform a raytrace to a single voxel in the region of interest. Therefore, each raytrace is independent of one another, making this the suitable kernel for the GPU. Algorithm 9.2.11 shows their schematic implementation.

Algorithm 9.2.11

1: Use Siddon's formulas to compute the parametric value α_{min}

2: Use Siddon's formulas to compute $i_{min}, j_{min}, k_{min}$

3: Set $\alpha_{current} = \alpha_{min}$ and find the α values for potential next intersecting planes:

$\alpha_{x,next} = \alpha_x(i_{min})$, $\alpha_{y,next} = \alpha_y(j_{min})$, $\alpha_{z,next} = \alpha_z(k_{min})$

4: Initialize v, v_{old} which contain the index of the voxels

5: **while** $\alpha_{current} < 1.0$ **do**

6: **if** $(\alpha_{y,next} < \alpha_{x,next})$ & & $(\alpha_{y,next} < \alpha_{z,next})$ **then**

7: $\alpha_{mid} = \frac{\alpha_{current}+\alpha_{y,next}}{2}$ (Ray intersects the y-plane)

8: $\ell = \alpha_{y,next} - \alpha_{current}$, set $\alpha_{current} = \alpha_{y,next}$

9: $\alpha_{y,next} = \alpha_{y,next} + \frac{d_y}{Y_2 - Y_1}$ (Potential next y-plane)

10: update voxel index v

11: **else if** $(\alpha_{x,next} < \alpha_{z,next})$ **then**

12: $\alpha_{mid} = \frac{\alpha_{current}+\alpha_{x,next}}{2}$ (Ray intersects the x-plane)

13: $\ell = \alpha_{x,next} - \alpha_{current}$, set $\alpha_{current} = \alpha_{x,next}$

14: $\alpha_{x,next} = \alpha_{x,next} + \frac{d_x}{X_2 - X_1}$ (Potential next x-plane)

15: update voxel index v

16: **else**

17: $\alpha_{mid} = \frac{\alpha_{current}+\alpha_{z,next}}{2}$ (Ray intersects the z-plane)

18: $\ell = \alpha_{z,next} - \alpha_{current}$, set $\alpha_{current} = \alpha_{z,next}$

19: $\alpha_{z,next} = \frac{z_{plane}[k] - Z_1}{Z_2 - Z_1}$ (Potential next z-plane)

20: update v, k

21: **end if**

22: **if** $\rho(v_{old}) \neq 0$ **then**

23: *radiological path length = radiological path length* $+\rho(v_{old})\ell$

24: **end if**

25: $v_{old} = v$

26: **end while**

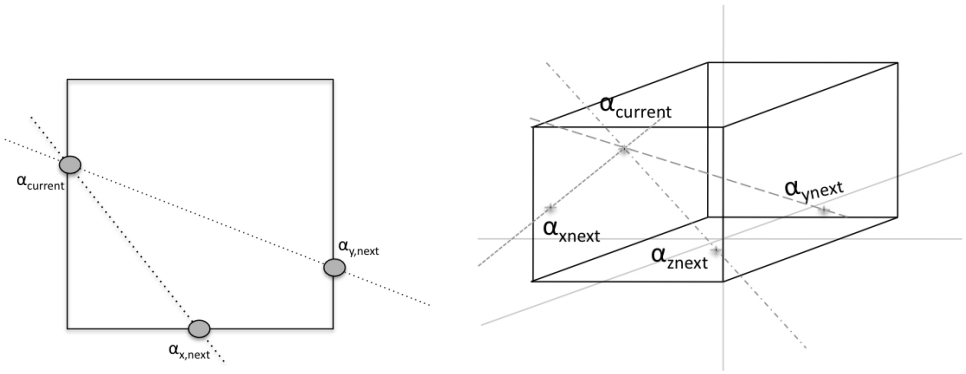
27: *radiological path length* $= d_{12} \times$ (*radiological path length*)

The idea of Algorithm 9.2.11 is to begin by using the formulas presented in Siddon's algorithm to determine the initial intersection of the ray, and then follow the ray through

the future intersecting planes using recursion patterns. To determine the initial intersection we find the initial parametric value α_{min} and the indices for the initial planes inside the object $i_{min}, j_{min}, k_{min}$. Note that once the ray intersects the first plane, there are three possible intersections for the ray to have next:

1. it can intersect an x-plane,
2. it can intersect a y-plane, or
3. it can intersect a z-plane.

We can use Siddon's formulas to calculate the three candidates for the next α parameter denoted $\alpha_{x,next}, \alpha_{y,next}$, or $\alpha_{z,next}$. By Siddon's approach and the fact that the global set $\{\alpha\}$ is in ascending order, we know that the next intersecting plane will be the one corresponding to the minimum of $\alpha_{x,next}, \alpha_{y,next}$, or $\alpha_{z,next}$. Figure 9.3 shows an example of the potential next plane intersections of the ray in two and three dimensions. When we determine the



(a) Potential ray intersections 2D (b) Potential ray intersections 3D

Figure 9.3: Example of the potential intersections the ray can have.

next parametric value of α we can then compute the length of the ray through the voxel (the difference between the current α and the previous α), the midpoint of the ray inside the voxel (α_{mid}), and the voxel index v. After these values are computed, the process can begin again and the next intersecting plane is found. This process is repeated until the parametric value $\alpha_{current}$ is greater than one since we are in case (c) of Figure 9.2 where the ray begins outside the CT array and ends inside the object.

9.3 Raytracing in Digital Tomosynthesis

Geometrically, raytracing is the process of determining the length of the intersection of a given ray with each of the voxels in a discretized object. This operation is inherent to the digital tomosynthesis problem as we are interested in the interaction of an x-ray sent from a source at incident angle θ with the matter present in the object. Recall from chapters four through seven that the discrete formulation of Beer's law for the polyenergetic digital tomosynthesis problem is

$$\mathbf{b}_\theta^i = \sum_{e=1}^{N_e} s_e \exp\left(-\sum_{j=1}^{N_v} a_\theta^{ij} \boldsymbol{\mu}_{e,j}\right) \tag{9.7}$$

where N_e is the number of discrete energies, s_e is the value of the spectral distribution function for energy level e, N_v is the number of voxels, $\boldsymbol{\mu}_{e,j}$ is the attenuation of the voxel j for energy level e, and a_θ^{ij} is the length of the x-ray passing through voxel j and contributing to pixel i in projection θ. Note that the summation inside the exponential in the equation above is similar to the formula for the radiological path length in Equation (9.1). For the tomosythesis problem, we can say that $\boldsymbol{\mu}_{e,j}$ corresponds to the density of the voxel and a_θ^{ij} is the length of the ray.

The evaluation of the summation formula

$$\sum_{j=1}^{N_v} a_\theta^{ij} \boldsymbol{\mu}_{e,j}$$

appears at several points in the reconstruction frameworks outlined in the previous chapters. We need this quantity to evaluate the expected value of each pixel in a given projection, evaluate the derivatives of the cost function, and build the Hessian matrix. Since this formula depends on both the pixel i and the incident angle θ, for a given reconstruction problem we would be evaluating this summation as many times as the number of pixels in the detector (which can generally be about 2500 by 4000 pixels) and for as many incident angles as the number of projections (typically 15 to 30). Therefore, using Algorithm 9.1.10 to evaluate each instance of this summation is too expensive in terms of memory and

computation time. We need to develop an efficient algorithm to evaluate the summation expression millions of times in the reconstruction process.

9.3.1 Implementation Considerations for Siddon's Algorithm

Implementing a direct formulation of Algorithm 9.1.10 in a GPU in posses several challenges. First, as de Greef et al [34] point out, the amount of memory needed for a worst case scenario geometry is more than the memory currently available on a GPU. Consider the size of the clinical data sets we are interested in reconstructing for this thesis where we are raytracing an object that has been discretized into $2048 \times 1280 \times 50$ voxels. A worst case scenario geometry would be to assume that the source is just outside the CT array representing the object and the path of the ray is along the main diagonal. This means the ray will intersect roughly half of the planes in each direction. If we compute the parametric sets $\{\alpha_x\}, \{\alpha_y\}, \{\alpha_z\}$ we have $1024, 640$, and 25 floating point numbers for each array. When we merge the arrays into the set $\{\alpha\}$, we produce at most 1024+624+25 floating point numbers. Up to this point we have roughly 2GB of storage for the parametric values resulting from tracing a single ray from the source to *one* pixel. If the detector can be as large as 2500 by 4000 pixels and we are using between 15 and 30 projection angles, the storage costs are unreasonable. This aspect alone shows that this type of implementation is not suitable for a GPU because the raytrace for all pixels in the detector will not fit in the available memory for any device in the current market. If we perform each raytrace separately, the CPU-GPU memory communication will out weigh the reduction of the computational cost.

Another consideration that we must make if we were to use Siddon's approach directly on a GPU is how to merge the sets $\{\alpha_x\}, \{\alpha_y\}, \{\alpha_z\}$ into the global set $\{\alpha\}$ and sort the values. A sorting operation is notoriously slow on a GPU since it is a memory bound kernel. For our digital tomosynthesis reconstruction, we need raytracing to happen very fast, and even optimizing the sorting operation can still make this process very slow in comparison to other approaches. In order to use Siddon's algorithm efficiently and accurately on a GPU, we have to redesign the raytracing kernel.

9.3.2 Raytracing for Digital Tomosynthesis on a GPU

Our goal is to raytrace the x-ray beam starting at the position of the source, through the discretized object, and ending at the center of each pixel in the detector (see Figure 9.4). This means, each pixel in the detector will perform a raytrace using a call to Siddon's

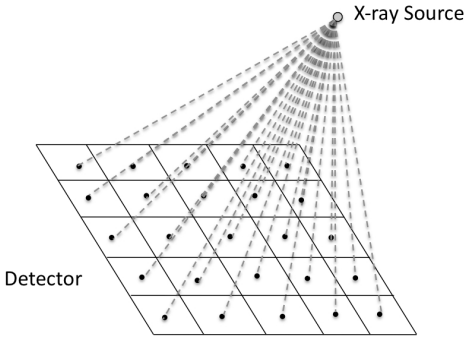


Figure 9.4: Raytracing in digital tomosynthesis. Multiple rays from a single source ending at the center of each pixel in the detector to produce one projection.

algorithm. To do this, we need to develop "light weight" tasks that can be mapped to GPU threads. These threads must have the following two properties:

1. A task should not require too many local variables, the processing unit executing the task will have limited "fast" memory.

2. A task should be self-contained. That is, a task should be completely independent of any neighboring task.

In our implementation, an independent task is a single pixel raytracing from the source to its location on the detector. Therefore, in order to make the task of each pixel light weight, we must address the two computational weaknesses of Algorithm 9.1.10 which are the amount of memory needed for a single raytrace and the sorting of the parametric values $\{\alpha\}$. The iterative sorting idea described in Algorithm 9.2.11 effectively addresses these two issues.

In Algorithm 9.2.11 the computation of the sets $\{\alpha_x\}, \{\alpha_y\}, \{\alpha_z\}$ and the sorting of the global $\{\alpha\}$ is combined using a stepping approach. Given a starting $\{\alpha\} = \{\alpha(0)\}$ the idea is to determine the three candidates $\alpha_{x,next}, \alpha_{y,next}, \alpha_{z,next}$ for the next element $\alpha(1)$, select

the correct value of $\alpha(1)$ and discard the other two values. This decreases the memory requirement for each task significantly as only the parametric values on the set $\{\alpha\}$ are stored, and it eliminates the need to have a global sorting operation since a three element sort is done before adding a new value to the global parametric set. Once the right $\alpha(1)$ value is selected, we can use it to compute the length of the ray inside the voxel and the voxel index. The product of the voxel density and the length of the ray through the voxel is added to a running sum and the loop moves on to find the next $\alpha(2)$ value for the set $\{\alpha\}$. Using this stepping approach, the task for each pixel only needs to store the geometry of the problem, size of the voxel, two adjacent α values from the set $\{\alpha\}$ and the variable storing the total radiological path. This process continues until the end of the ray is reached.

Our implemented raytrace operation for the digital tomosynthesis reconstruction problem is a modification of the approach used in [34]. Algorithm 9.3.12 shows our raytracing kernel for any pixel in the detector and any projection angle. Note that in order to generalize the kernel to all the pixels in the detector we need to take into account the fact that the x-rays emitted by the source can approach the center of the pixels in the detector from different directions along each dimension. To do this, we add checkpoints inside the kernel to determine the direction in which the ray approaches for each pixel. For example, if the source is to the left of a pixel, $x_2 - x_1 > 0$, if it is to the right then $x_2 - x_1 < 0$, and if $x_2 - x_1 = 0$ the ray is parallel to the x-planes and no intersections of this type are considered.

Our raytrace kernel for a single pixel in the detector is given as follows:

Algorithm 9.3.12

1: Determine the position of the start of the ray (x_1, y_1, z_1)

2: Find the center of the pixel (x_2, y_2, z_2).

3: Compute the ray length $d_{12} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$

4: Use Siddon's formulas to compute the parametric range $(\alpha_{min}, \alpha_{max})$

5: Use Siddon's formulas and the direction the ray approaches the pixel to compute $(i_{min}, j_{min}, k_{min})$ and $(i_{max}, j_{max}, k_{max})$

6: **if** $(X_2 - X_1) \geq 0$ **then**

$\alpha_{x,next} = \frac{X_{plane}(1) - i_{min} d_x - X_1}{X_2 - X_1}$

else

$\alpha_{x,next} = \frac{X_{plane}(1) - (i_{max} - 1) d_x - X_1}{X_2 - X_1}$

end if

if $(Y_2 - Y_1) \geq 0$ **then**

$\alpha_{y,next} = \frac{Y_{plane}(1) - j_{min} d_y - Y_1}{Y_2 - Y_1}$

else

$\alpha_{y,next} = \frac{Y_{plane}(1) - (j_{max} - 1) d_y - Y_1}{Y_2 - Y_1}$

end if

if $(Z_2 - Z_1) \geq 0$ **then**

$\alpha_{z,next} = \frac{Z_{plane}(1) - k_{min} d_z - Z_1}{Z_2 - Z_1}$

else

$\alpha_{z,next} = \frac{Z_{plane}(1) - (k_{max} - 1) d_z - Z_1}{Z_2 - Z_1}$

end if

If $\alpha_{x,next} = 0, \alpha_{y,next} = 0$, or $\alpha_{z,next} = 0$ set it to a number greater than one.

Set $\alpha_{current} = \alpha_{min}$

Initialize the resulting value for the pixel: $pixel\_value = 0$

if $\alpha_{max} < \alpha_{min}$ **then**

The ray does not intersect the volume to reach the center of the pixel

else

while $\alpha_{current} < \alpha_{max}$ (while the ray is inside the CT array) **do**

if $(\alpha_{y,next} < \alpha_{x,next})$ & & $(\alpha_{y,next} < \alpha_{z,next})$ **then**

$\alpha_{mid} = \frac{\alpha_{current} + \alpha_{y,next}}{2}$ (Ray intersects the x-plane)

$\ell = \alpha_{y,next} - \alpha_{current}$; $\alpha_{current} = \alpha_{y,next}$

$\alpha_{y,next} = \alpha_{y,next} + \frac{d_y}{Y_2 - Y_1}$ (Potential next y-plane)

else if $(\alpha_{x,next} < \alpha_{z,next})$ **then**

$\alpha_{mid} = \frac{\alpha_{current} + \alpha_{x,next}}{2}$ (Ray intersects the x-plane)

$\ell = \alpha_{x,next} - \alpha_{current}$; $\alpha_{current} = \alpha_{x,next}$

$\alpha_{x,next} = \alpha_{x,next} + \frac{d_x}{X_2 - X_1}$ (Potential next x-plane)

else

37: $\alpha_{mid} = \frac{\alpha_{current}+\alpha_{z,next}}{2}$ (Ray intersects the z-plane)

38: $\ell = \alpha_{z,next} - \alpha_{current}$; $\alpha_{current} = \alpha_{z,next}$

39: $\alpha_{z,next} = \alpha_{z,next} + \frac{d_z}{Z_2 - Z_1}$ (Potential next x-plane)

40: **end if**

41: Find the coordinates of the current voxel
$i = \frac{X_1 - \alpha_{mid}(X_2 - X_1) - X_{plane}(1)}{d_x}$, $j = \frac{Y_1 - \alpha_{mid}(Y_2 - Y_1) - Y_{plane}(1)}{d_y}$,
$k = \frac{Z_1 - \alpha_{mid}(Z_2 - Z_1) - Z_{plane}(1)}{d_z}$

42: **if** voxel (i, j, k) is part of the discretized volume **then**

43: $pixel\_value = pixel\_value + d_{12} \times \ell \times volume(i, j, k)$

44: **end if**

45: **end while**

46: **end if**

We begin Algorithm 9.3.12 by determining the starting point of the ray and the center of the pixel in question (the end of the ray). Then use Siddon's formulas to calculate the range of the parametric values given by $(\alpha_{min}, \alpha_{max})$. In our case, we must find a value for α_{max} because the ray can end outside the volume for some pixels in the detector. This is also why we must find all three index ranges $(i_{min}, j_{min}, k_{min})$ and $(i_{max}, j_{max}, k_{max})$ to determine the last voxel the ray crosses before it exits the 3D object. The next step is to determine the potential intersections of the ray with the orthogonal planes. We use Siddon's formulas to determine the possible α_{next} values. If $(x_2 - x_1) \geq 0$, then by Siddon's argument the set $\{\alpha_x\} = \{\alpha_x(i_{min}), ..., \alpha_x(i_{max})\}$, so we find the next potential parametric value by

$$\alpha_{x,next} = \alpha_x(i_{min}) = \frac{x_{plane(1)} - i_{min} d_x - x_1}{x_2 - x_1}.$$

However, if $(x_2 - x_1) < 0$ then the set corresponding to the parametric values will occur as $\{\alpha_x\} = \{\alpha_x(i_{max}), ..., \alpha_x(i_{min})\}$, so we find

$$\alpha_{x,next} = \alpha_x(i_{max}) = \frac{x_{plane}(1) - (i_{max} - 1) d_x - x_1}{x_2 - x_1}.$$

The process follows similarly for the sets $\{\alpha_y\}$ and $\{\alpha_z\}$. Note that if the values $\alpha_{x,next}, \alpha_{y,next}$, or $\alpha_{z,next}$ are equal to zero at any point, we know that the ray will not cross any more planes in that dimension so we set the corresponding value α_{next} to a number greater than one to avoid falling into an infinite loop.

After the initialization process we are ready to begin tracking the ray. Here we check if the ray actually enters the object by determining if $\alpha_{min} < \alpha_{max}$ and break if necessary. If the ray does enter the object, we use a modification of the loop in Algorithm 9.2.11 to take advantage of the equidistant z-planes. This eliminates the need to have a separate array containing the position of the z-planes, and thus an additional access to global memory. For our algorithm, once the while loop begins we do not need to request global memory accesses other than to retrieve the attenuation coefficient of the voxel. Inside the while loop, the ray will be tracked from the starting intersection with the object to the last voxel the ray crosses before it exits the object to reach the center of the pixel. At each instance of the while loop, a voxel length is calculated, the corresponding voxel index for that length is computed, and the sum variable is updated. The output of a single call of this algorithm is a single value that represents the energy measured by the individual pixel in the detector after the x-ray attenuates through the object. The set of values collected for all the pixels in the detector represents a single monochromatic projection of the object acquired by the imaging system.

9.3.3 Our Modified Raytrace vs. a Direct Implementation of Siddon's Algorithm

In order to determine the accuracy of our algorithm we simulate sample projections using a direct implementation of Siddon's algorithm (Algorithm 9.1.10) and our own modified raytrace (Algorithm 9.3.12). We begin by considering a raytrace of a the two dimensional object. In this setup, we have a one dimensional detector and a two dimensional object that is discretized into 2D voxels, each having their own attenuation coefficient. For the simulated sample projections, we assume homogeneous object with attenuation coefficient one. This allows us to further evaluate the accuracy of the geometry of the ray, since the resulting value for each pixel in the detector should be the length of the portion of the ray

that travels through the object, starting at the source and ending at the center of the pixel in the detector. We generated 15 projections of the object at different incident angles.

While generating our sample projections, we noticed an oscillating artifact begin to appear in some of the results. The projections generated using Algorithm 9.1.10, our modified raytrace, and the difference between the two can be seen in Figure 9.5. This oscillation is

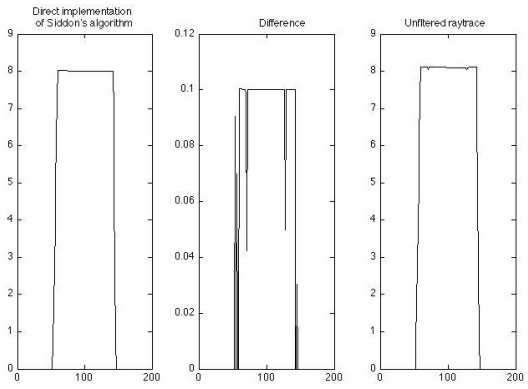


Figure 9.5: Comparison of resulting projections in 2D between direct Siddon's raytrace and unfiltered tomosynthesis raytrace

more noticeable in some of the projections and a close look at the difference between the two outputs shows the relative error between our approach and the direct implementation of Siddon's raytrace is about 2%. We performed the same comparison for a three dimensional problem, where the object is discretized into 3D voxels and the detector is two dimensional. In this case the artifacts are more prominent and create a distinct grid that almost annihilates the expected shape of the raytrace in the projection (see Figure 9.7). The artifacts are due an overestimation of the length of the ray by the while loop in Algorithm 9.3.12.

To fix the overestimation of the length of the ray we propose conditions to terminate the while loop earlier. The artifact is purely caused by round off error and it is easily eliminated by running the while loop for as long as $\alpha_{max} - \alpha_{current} > \tau$ where τ is a tolerance level depending on machine epsilon. For our experiments, using a tolerance $\tau = 10^{-10}$ for double precision and $\tau = 10^{-6}$ for single precision produces excellent results. Figures 9.6 and 9.7 show how our modified raytrace compares to a direct implementation of Siddon's algorithm in double precision for two and three dimensional problems.

Using a tolerance level to terminate the while loop in our modified raytrace shows that

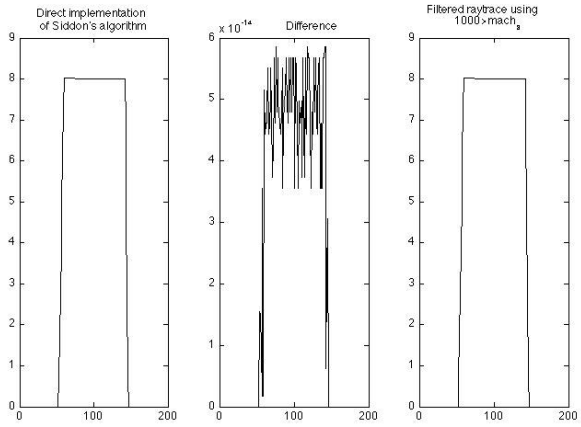


Figure 9.6: Comparison of resulting projections in 2D between direct Siddon's raytrace and filtered tomosynthesis raytrace using a tolerance of $1000\times$ $mach_\varepsilon$

Algorithm 9.3.12 is an accurate implementation of Siddon's algorithm that can run using multiple threads on a GPU. In numerical experiments, we see that our modified raytracing algorithm performs significantly better than a direct implementation of Algorithm 9.1.10. For example, in serial we see that our modified raytrace runs, on average, in about 23% of the time that Siddon's algorithm runs, being accurate to fourteen digits in double precision. For the digital tomosynthesis problem, we use Algorithm 9.3.12 to evaluate the forward models and the derivatives of our cost function in the reconstruction process.

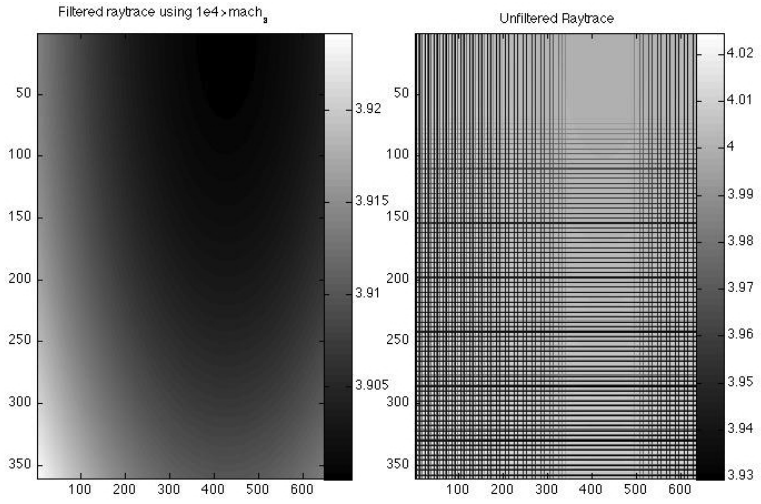


Figure 9.7: Left: In 3D a filtered raytrace using a tolerance of $1000\times$ $mach_\varepsilon$. Right: In 3D the same problem using and unfiltered raytrace

9.4 The Backprojection Operation

As described in Section 3.4.1, the backprojection operation is crucial to the reconstruction process in digital tomosynthesis. Recall from the previous discussions that in numerical analysis terms, performing a raytrace operation is the equivalent to performing a matrix multiplication with the raytrace matrix A_θ. That is, if the vector $\mathbf{v}$ represents the discretized volume, the operation $A_\theta\mathbf{v}$ will result in a monochromatic projection of the object at incident angle θ. We choose to use a raytrace algorithm to perform the matrix vector products as opposed to storing the full matrix A_θ because of storage limitations. For example, for a clinical size problem, the size of each A_θ for $\theta = 1, 2, ..., N_\theta$ is $N_p \times N_v$ where the values $N_p = 2048 \times 1280$, $N_v = 2048 \times 1280 \times 50$, and $N_\theta = 15$ to 30. Even storing A_θ in a sparse matrix format is unreasonably expensive because the structure of the matrix is completely determined at the time of image acquisition.

In this chapter, we have thus far developed a modified raytrace kernel from Siddon's algorithm to efficiently compute $A_\theta\mathbf{v}$ in a GPU. This allows us to evaluate very quickly any of the forward models for the reconstruction problem presented in the previous chapters. However, for the each iteration of the polyenergetic reconstruction problem we also need to perform a matrix-vector product using the transpose of the raytracing matrix A_θ^T. In medical imaging, the product of A_θ^T and a vector $\mathbf{p}$ representing the observed projection of an object is called a backprojection, and the result is a vector $\mathbf{v}$ representing the density of each voxel in the discretized object. Extending our modified raytrace to compute a backprojection is not a straightforward process and can be very slow. Therefore, we need to develop a backprojection kernel that can be executed by independent threads in a GPU to compute the product $A_\theta^T\mathbf{p}$.

Our backprojection operator extends on the idea presented in Figure 3.3. The goal is to trace back all rays to a given voxel, and average the measured data in pixel i along all voxels that the ray crossed giving more "weight" to those voxels where the length of the ray was longer. That is, for each ray, we compute the length traveled through all voxels, and the density of the voxel is set to be a weighted average of the energy measured by the pixel at the end of the ray. If we drive the parallelism of the backprojection operation by

the resulting voxel value, we can enqueue the backprojection kernel for each voxel. This leads to independent identical "light weight" tasks that map to GPU threads. Algorithm 9.4.13 shows our backprojection operation for a given voxel indexed (x, y, z) using a single projection $\mathbf{p}_\theta$.

Algorithm 9.4.13 Backprojection Operation

1: For given voxel (x, y, z) center of the voxel (x_2, y_2, z_2)

2: Use the positions of the center of the voxel to determine the range of parametric values for the ray inside the voxel
$(\alpha_{x,min}, \alpha_{x,max})$, $(\alpha_{y,min}, \alpha_{y,max})$ and $(\alpha_{z,min}\ , \alpha_{z,max})$.

3: Find the parametric values $\alpha_{min} = \max(0, \alpha_{x,min}, \alpha_{y,min}, \alpha_{z,min},)$ and
$\alpha_{max} = \min(1.0, \alpha_{x,max}, \alpha_{y,max}, \alpha_{z,max},)$

4: **if** $\alpha_{max} < \alpha_{min}$ **then**

5: The ray does not cross the voxel

6: **end if**

7: Compute the length of the ray from the source to the center of the voxel
$d_{12} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$

8: The length of the ray inside the voxel is a fraction of the length of the ray from the source to the center of the voxel, given by $\ell = d_{12} \times (\alpha_{max} - \alpha_{min})$

9: Retrieve the corresponding value for the pixel in the projection $\mathbf{p}(i, j)$

10: The value of the density of the voxel is given by $volume(x, y, z) = \ell \times \mathbf{p}_\theta(i, j)$

For multiple projections N_θ, we perform N_θ backprojections and add the density values for corresponding voxels.

The idea of algorithm 9.4.13 is to limit the scope of the raytrace to the voxel in question. First, we begin by determining the center of the voxel and the range of parametric values $(\alpha_{x,min}, \alpha_{x,max})$, $(\alpha_{y,min}, \alpha_{y,max})$ and $(\alpha_{z,min}\ , \alpha_{z,max})$ that any ray crossing the voxel can have. Then, we find the *global* values for α_{min} and α_{max} using Sddon's formulas to determine if the ray crosses the voxel in question. If the ray crosses the voxel, we calculate the length of the ray as a fraction of the length of the ray from the source to the voxel. Then we find the index of the pixel in the projection data that will be averaged over the

path of the ray and we compute the density of the voxel.

9.5 Summary

In this chapter we have developed two efficient algorithms to perform the matrix vector products $A_\theta \mathbf{v}$ (called a raytrace) and $A_\theta^T \mathbf{p}$ (called a backprojection) where A_θ is the raytracing matrix in the digital tomosythesis problem. An efficient and fast approach to compute these products is crucial to the polyenergetic reconstruction frameworks described in Chapters 4-7 since these are the building blocks of the cost function and its derivatives. We choose to use a raytrace and backprojection algorithm to perform the matrix vector products as opposed to storing the full matrix A_θ because of storage limitations. For our problems, the size of each A_θ is $N_p \times N_v$ where the values $N_p = 2048 \times 1280$, $N_v = 2048 \times 1280 \times 50$, and $N_\theta = 15$ to 30. Even storing A_θ in a sparse matrix format is unreasonablly expensive because the structure of the matrix is completely determined at the time of image acquisition.

To perform the product $A_\theta \mathbf{v}$ we have described Siddon's algorithm and its shortcomings. Using the work in [34] we developed a raytracing kernel that does not require a sort operation and uses limited memory to store its variables. Our new raytrace kernel can run as fast as $4\times$ the direct implementation of Siddon's algorithm. In addition, we can enqueue an arbitrary number of threads to execute the kernel in parallel, thus making it scalable and allowing us to compute a full projection in one single GPU function call.

To perform the product $A_\theta^T \mathbf{p}$ we have outlined a simple backrojection algorithm that computes the voxel densities as a weighted average of the pixel value. As in the case with the raytracing kernel, the backprojecton kernel can be enqueued by an arbitrary number of threads at once, each representing a voxel in the discretized object. In the next chapter we discuss the software design for the reconstruction framework we have proposed for the digital tomosynthesis problem, including the details of execution topology and kernel functionality. In addition, we describe how the functionality of the raytrace and backprojection kernel can be extended to produce fast tomosynthesis reconstructions.

Chapter 10

Computational Design

As described in Chapter 8, the digital tomosynthesis reconstruction problem is computationally and memory intensive as we deal with millions of input pixels to reconstruct a volume discretized in terms of billions of voxels. In Chapter 9 we described an optimized raytracing and backprojection operation that reduces storage requirements and allows us extract the parallelism of the application by mapping the most intensive computational tasks to the hardware of a GPU. This allow us to perform matrix-vector products in seconds using one single GPU. However, optimizing the raytrace and backprojection operations is not enough to speed-up the reconstruction process to a reasonable time. Ultimately, we would like our reconstructions to run in a matter of minutes in order to compete with the reconstructions provided by the imaging device.

In this chapter we describe the implementation details of our application to solve the polyenergetic digital breast tomosynthesis reconstruction problem using a single GPU. The application is written in C++ using OpenCL as the API. We focus on efficient handling of memory communication and increasing computational performance to achieve the best results. We discuss the execution details of the raytracing and backprojection kernels and present three implementations of the reconstruction algorithm, a serial approach, an approach focused on the optimization of the matrix products by threading for functionality, and a fused kernel approach. We show that of all three implementations a fused kernel approach is the most effective in our reconstruction framework, producing viable reconstructions in less than five minutes.

10.1 Serial Implementation

We begin by considering the reconstruction framework in serial as a basis for our GPU optimization comparisons. Recall from Chapters 4-7 that the reconstruction framework we propose in this thesis is either a gradient descent algorithm or a Newton method that finds the maximum likelihood estimator for the Poisson based likelihood function arising from the forward model. That is, ultimately, we are solving the optimization problem

$$\mathbf{w}_{MLE} = \arg\ \min_{\mathbf{w}} \left\{ \sum_{\theta=1}^{N_\theta} -L(\mathbf{b}_\theta, \mathbf{w}) \right\}. \tag{10.1}$$

where $\mathbf{w} = \{\mathbf{w}_{:,2}, ..., \mathbf{w}_{:,N_m}\}$ is the set of unknown weight fractions that depend on the forward model used. The cost function is given by

$$-L(\mathbf{b}_\theta, \mathbf{w}) = \sum_{i=1}^{N_p} (\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i) - \mathbf{b}_\theta^i \log(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i) + c \tag{10.2}$$

and $\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i$ depends on the forward model used from Chapter 4-7. Note that the first and second derivatives of the function $L(\mathbf{b}_\theta, \mathbf{w})$ are known. In this thesis we discuss the implementation details of the gradient descent algorithm, a fully functional Newton approach implementation is left as future work.

Schematically, the iterative process for any of the four forward models proposed in Chapters 4-7 is shown in Figure 10.1. For the purpose of this discussion we will use the simplified multi-material model for attenuation described in Chapter 7 using $N_m = 3$, that is, three materials are modeled. This gives the expected value of all pixels as

$$\bar{\mathbf{b}}_\theta + \bar{\boldsymbol{\eta}} = \sum_{e=1}^{n_e} s_e \exp\left(-\left[c_{3,e} A_\theta \mathbf{w}_{:,3} + c_{2,e} A_\theta \mathbf{w}_{:,2} + c_{1,e} A_\theta \mathbf{1}\right]\right) + \bar{\boldsymbol{\eta}} \tag{10.3}$$

where the scalars $c_{3,e}$, $c_{2,e}$, and $c_{1,e}$ depend on the linear attenuation coefficients of each material modeled and $\mathbf{w}_{:,*}$ represent the weight fractions of the two explicit materials modeled.

For notation purposes, note that

$$[c_{3,e}A_\theta \mathbf{w}_{:,3} + c_{2,e}A_\theta \mathbf{w}_{:,2} + c_{1,e}A_\theta \mathbf{1}]_i = \sum_{j=1}^{N_v} a_\theta^{ij} \boldsymbol{\mu}_{e,j}$$

hence, for this problem we have the derivative

$$\nabla(-L(\mathbf{b}_\theta, \mathbf{w})) = \begin{bmatrix} A_\theta^T \mathbf{v}_2 \\ A_\theta^T \mathbf{v}_3 \end{bmatrix} \tag{10.4}$$

where the i-th entry of the vector $\mathbf{v}_2$ and $\mathbf{v}_3$ is given by

$$\begin{aligned} \mathbf{v}_2(i) &= \left(\frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)} - 1 \right) \sum_{e=1}^{n_e} s_e \left(u_{2,e} - u_{1,e}\right) \exp\left(- \sum_{j=1}^{N_v} a_\theta^{ij} \boldsymbol{\mu}_{e,j} \right) \\ \mathbf{v}_3(i) &= \left(\frac{\mathbf{b}_\theta^i}{(\bar{\mathbf{b}}_\theta^i + \bar{\boldsymbol{\eta}}^i)} - 1 \right) \sum_{e=1}^{n_e} s_e \left(u_{3,e} - u_{1,e}\right) \exp\left(- \sum_{j=1}^{N_v} a_\theta^{ij} \boldsymbol{\mu}_{e,j} \right). \end{aligned} \tag{10.5}$$

According to Figure 10.1, we begin each gradient descent iteration by evaluating the cost function $\sum_{\theta=1}^{N_\theta} -L(\mathbf{b}_\theta, \mathbf{w}^{(k)})$ as a three step process. First we compute the three matrix-vector multiplications $A_\theta \mathbf{w}_{:,3}$ $A_\theta \mathbf{w}_{:,2}$ and $A_\theta \mathbf{1}$ needed inside the exponential. Next, we apply the fit using the coefficients $c_{3,e}$, $c_{2,e}$, and $c_{1,e}$ to evaluate $\bar{\mathbf{b}}_\theta + \bar{\boldsymbol{\eta}}$. Finally, we complete the cost function evaluation by performing the summations over all pixels and all angles as described in Equations (10.1) and (10.2). After we complete the function evaluation, we begin to compute the derivative of the function in two steps. First we use the matrix-vector products from the function evaluation to compute the vectors $\mathbf{v}_2$ and $\mathbf{v}_3$ corresponding to each material. Then we perform the matrix transpose-vector products using our backprojection algorithm to find $A_\theta^T \mathbf{v}_2$, $A_\theta^T \mathbf{v}_3$ and stack the results. To end the iteration we do a line search to guarantee descent of the function and then start the gradient iteration with a new starting $\mathbf{w}^{(k+1)}$. The process is repeated until convergence or until we are unable to guarantee descent in the value of the function.

For a clinical size data set, five gradient descent iterations take one to two hours to reconstruct the volume as shown in the results of Chapter 11. However, generally we need more than five iterations to obtain good results. We use the serial reconstruction results

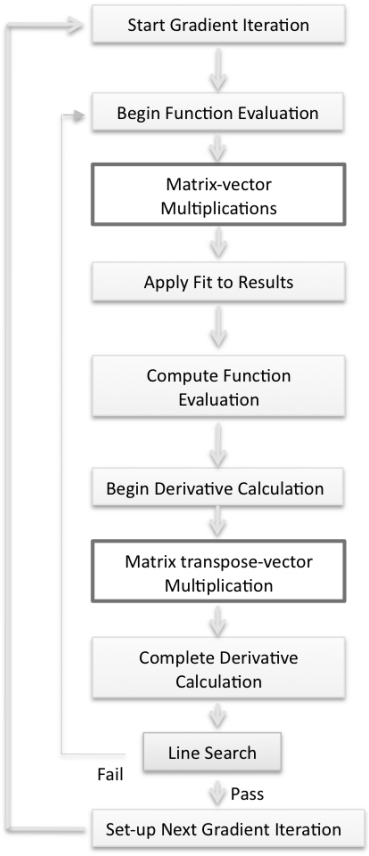


Figure 10.1: Gradient descent iteration serial scheme

shown in this section as a baseline to improve the computation time of our application with the goal of achieving a full clinical size reconstruction in minutes.

10.2 Matrix Product Optimizations

The first approach to accelerate the performance of the reconstruction algorithm using a single GPU is to thread the application for functionality, that is, create a kernel for each computationally intensive portion of the process and allow the device to run that portion before continuing the process. Profiling the serial code shows that the reconstruction algorithm spends the majority of the time computing the matrix-vector products.

We begin to improve the serial implementation by looking at our execution of the matrix-vector and matrix-transpose vector products. For the linear model for attenuation described in Chapter 4, each gradient descent iteration requires two matrix-vector products and one

matrix transpose vector product for each A_θ. For the more realistic quadratic and simplified multi-material cases (modeling three materials) described in Chapters 5 and 7, we need to perform at least three matrix-vector products and two matrix transpose vector products per iteration for each A_θ. In our numerical experiments we use data sets containing $N_\theta = 15$ projection angles, so for the quadratic and simplified multi-material cases we need at least 45 matrix-vector products and 30 matrix transpose vector products per gradient descent iteration. Note that each raytracing matrix A_θ is very large, size $N_p \times N_v$ which, for our sample data, translates to 1280×2048 rows by $1280 \times 2048 \times 50$ columns. Although we know that the matrix A_θ is sparse, it is neither symmetric nor structurally symmetric and the structure of the matrix is hard to estimate as it is completely determined at the time of image acquisition. The best thing we can say about the sparsity of A_θ is that the maximum number of nonzeros per row is $\dfrac{n_x + n_y + n_z}{2}$ which corresponds to a ray traversing the discretized object along the main diagonal.

Although we do not build the matrices A_θ explicitly, we are able to perform very fast matrix-vector and matrix transpose-vector products with these matrices by implementing Algorithms 9.3.12 and 9.4.13 described in Chapter 9. Using a modified version of Siddon's

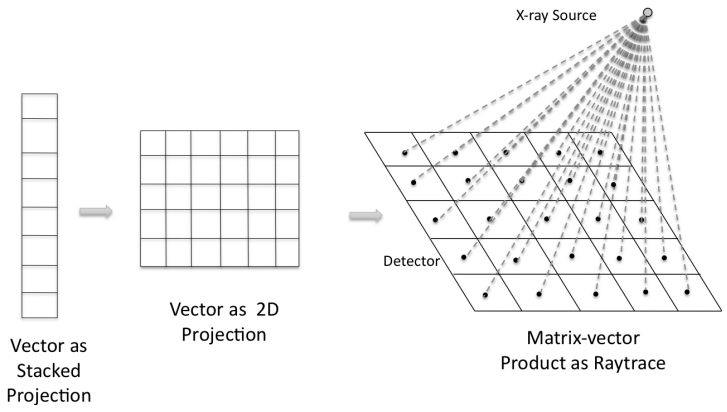


Figure 10.2: Matrix vector product as a raytrace

algorithm [105] we are able to perform the matrix products as a raytrace by reverting to the inherent two dimensional index of each entry in the vector, since each vector physically represents a stacked projection (see Figure 10.2).

However, even by using an optimized raytracing algorithm, enqueuing a kernel call to

perform a single matrix product A_θ is not an efficient utilization of GPU resources, as we have to transfer data from CPU to GPU at each matrix product kernel call and then we have to combine the results for all products using the CPU. Recall that for a single matrix-vector product $A_\theta \mathbf{u}$, each thread needs to know the vector $\mathbf{u}$ which represents the discretized volume and the position of the source in the imaging system for projection angle θ. Since we know the source position for all projections at the start of the reconstruction process and this information is a small $3 \times N_\theta$ array of doubles containing the coordinates of the source, we can load the geometry information for all projections into the device once for the full reconstruction time. In this set-up, there is still no thread communication or synchronization needed, as each thread will be computing a single output value for the pixel corresponding to its thread ID in the execution model. This ensures that there is no chance of a race condition or data hazards. Therefore, by having the vector and geometry

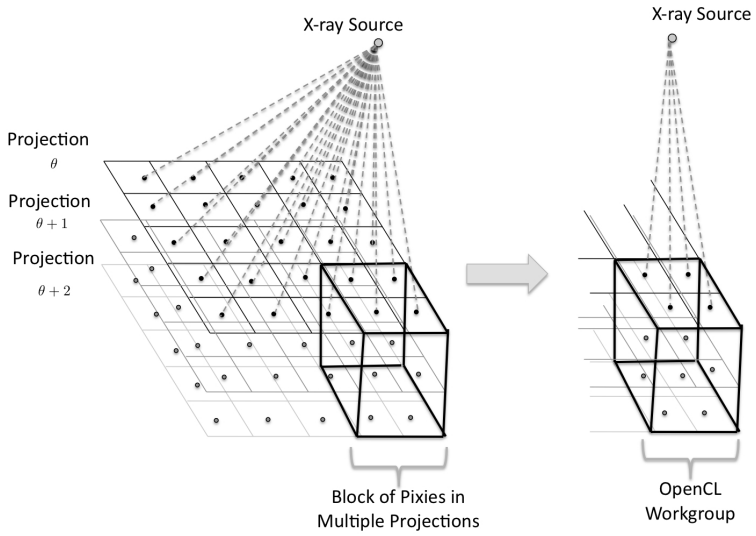


Figure 10.3: 3D Execution topology for raytrace kernel. Note that the thread blocks are consecutive pixels and pixels that are traced by the same ray along multiple projections which ensures memory coalescing.

information in the device, we can avoid the overhead costs of enqueueing each $A_\theta \mathbf{u}$ for $\theta = 1, ..., N_\theta$ individually. Furthermore, we can take advantage of a 3D execution topology to enqueue a single kernel call that will produce all N_θ products at once. The parallelism for the raytrace is driven by the output data and the execution topology is shown in Figure 10.3.

The execution model shown in Figure 10.3 allows us to take advantage of data locality and memory coalescing since a single workgroup will be composed of threads that represent either neighboring pixels or pixels traced by the same ray along multiple projections. Table 10.1 shows the advantages of enqueueing all matrix-vector products and matrix transpose-vector products in a single kernel for all angles as opposed to issuing multiple individual calls for each product. The vector $\mathbf{x}_\theta$ represents a projection at incident angle θ, the vector $\mathbf{x}$ without subscript represents a vector of all projection data $\mathbf{x}_\theta$ for $\theta = 1, ..., N_\theta$ stacked, and the matrix A without subscript represents the operator

$$A = \begin{bmatrix} A_1 \\ A_2 \\ \vdots \\ A_{N_\theta} \end{bmatrix}, \text{ and } A^T = \begin{bmatrix} A_1^T & A_2^T & ... & A_{N_\theta}^T \end{bmatrix}. \tag{10.6}$$

The results of Table 10.1 give us two important conclusions. First, the raytracing Algorithm 9.3.12 and the backprojection Algorithm 9.4.13 run much faster in a GPU as opposed to a CPU. A single raytrace $A_\theta \mathbf{v}$ can be done in less than two seconds on the device as opposed to over 30 seconds in serial, representing a speed-up of over 20×. In addition, a single backprojection $A_\theta^T \mathbf{x}_\theta$ can be done in less than six seconds in the GPU as opposed to over 130 seconds in serial, representing a speed up of over 22×. The discrepancy in the time to compute a raytrace and backprojection is attributed to the fact that even though the two algorithms fundamentally computing the same quantities, they are not the same. The second conclusion obtained from Table 10.1 is that the overhead cost of enqueuing multiple kernels is significant. Note that enqueueing 15 raytrace kernels independently will take approximately 24 seconds whereas a single kernel call that accounts for all projection angles will run in less than 10 seconds. The same can be said about the backprojection kernel where multiple calls to the kernel will take about 30 seconds as opposed to the almost 6 seconds it takes for a single kernel execution.

We can optimize the performance of the matrix product kernels even further by using the available 6GB memory in the Nvidia Tesla 2070c. If we perform the matrix-vector

	Matrix-vector product		Matrix transpose-vector product	
Implementation Type	one angle $A_\theta \mathbf{v}$	all angles $A\mathbf{v}$	one angle $A_\theta^T \mathbf{x}_\theta$	all angles $A^T\mathbf{x}$
Serial	33.58 sec	472 sec	9.8 sec	130.8 sec
GPU	1.65 sec	9.94 sec	2.05 sec	5.8 sec

Table 10.1: Times for the execution of the matrix-vector and matrix transpose-vector product operations for one angle θ and all angles. The size of matrix A_θ is 1280×2048, the number of angles (N_θ) is 15, the size of vector $\mathbf{v}$ is $1280 \times 2048 \times 50$, and the size of vector $\mathbf{x}_\theta$ is 1280×2048.

product for all angles at once as described above, the amount of memory we are utilizing in double precision for the vector representing the discretized volume is just over 1GB and the amount of memory needed for the projection data is less than 0.5GB. The geometry information including the position of source, the size and number of voxels and pixels, the projection angles and others is negligible, less than 100 doubles. Therefore, we have about two thirds of the available global memory of the device unused. Since we know ahead of the kernel call the number of matrix-vector and matrix transpose-vector products we have per iteration (depends on the forward model of the problem) we can optimize communication time by extending the kernel to compute multiple products in one call. That is, instead of enqueueing a single kernel call to compute $A\mathbf{v}$ where A is as in Equation (10.6), we can compute the matrix-vector product $A\begin{bmatrix}\mathbf{u} & \mathbf{v} & \mathbf{w}\end{bmatrix}$ by a simple extension of the raytracing kernel to take three volumes as input and output three projections. We do this by having each thread calculate three output values as if we had the same ray going through three different volumes and landing in three different detectors. The thread is responsible for following the ray, it takes into account the attenuation of each volume $\mathbf{u}$, $\mathbf{v}$ and $\mathbf{w}$, and it reports as output the value of the pixel corresponding to the thread ID at each of the three detectors. The only additional operations we are requiring from the raytracing kernel (Algorithm 9.3.12) is to keep track of two more *pixel_values* quantities in line 43, one for each new input vector $\mathbf{v}$ and $\mathbf{w}$. Similarly for the backprojection operation, we "backproject" to two volumes from the projections of two different detectors.

By performing the multiple matrix-vector products $A\begin{bmatrix}\mathbf{u} & \mathbf{v} & \mathbf{w}\end{bmatrix}$ and the matrix transpose-

	Matrix-vector product		Matrix transpose-vector product	
Implementation	$A\mathbf{u}$	$A\begin{bmatrix}\mathbf{u} & \mathbf{v} & \mathbf{w}\end{bmatrix}$	$A^T\mathbf{x}$	$A^T\begin{bmatrix}\mathbf{x} & \mathbf{y}\end{bmatrix}$
Serial	472 sec	490 sec	130.8 sec	148.83
GPU	9.94 sec	10.94 sec	5.8 sec	7.8 sec

Table 10.2: Times to compute matrix-vector or matrix transpose-vector products using a single kernel call for one or multiple products. The size of matrix A_θ is 1280×2048, the number of angles (N_θ) is 15, the size of vectors $\mathbf{u}$, $\mathbf{v}$, and $\mathbf{w}$ is $1280 \times 2048 \times 50$, and the size of vectors $\mathbf{x}$ and $\mathbf{y}$ is $1280 \times 2048 \times 15$.

vector products $A^T\begin{bmatrix}\mathbf{x} & \mathbf{y}\end{bmatrix}$ in one single kernel call, we avoid the communication and control overhead of enqueueing two or three separate kernel calls while maximizing the utilization of the available device memory. Table 10.2 shows the gains of performing these products in one call. As can be seen, for $A\begin{bmatrix}\mathbf{u} & \mathbf{v} & \mathbf{w}\end{bmatrix}$ we can compute the three products together in less than 11 seconds as opposed to about 30 seconds needed to compute all three products individually. Similarly, for $A^T\begin{bmatrix}\mathbf{x} & \mathbf{y}\end{bmatrix}$ we can compute two products together in less than 8 seconds as opposed to the almost 12 seconds needed to do both products as independent function calls. We can also see in this table that the cost of performing the products together is not significant, even in serial. For the GPU implementation we have an additional cost of less than one second for the raytrace and about two seconds for the backprojection. Figure 10.4 shows the schematic representation of the gradient descent process using the GPU to compute the matrix products.

10.3 Kernel Fusion Implementation

Optimizing the matrix product kernels and performing all products on the GPU as shown in Figure 10.4 provides a partial speed up of the reconstruction process. However, we can optimize the implementation even further. Profiling the new reconstruction implementation shows that at each iteration the application spends less than 10% of the time computing the matrix-vector and matrix transpose-vector products, and about 33% of the time in the evaluation of the cost function in Equation (10.1). This is because evaluating Equation

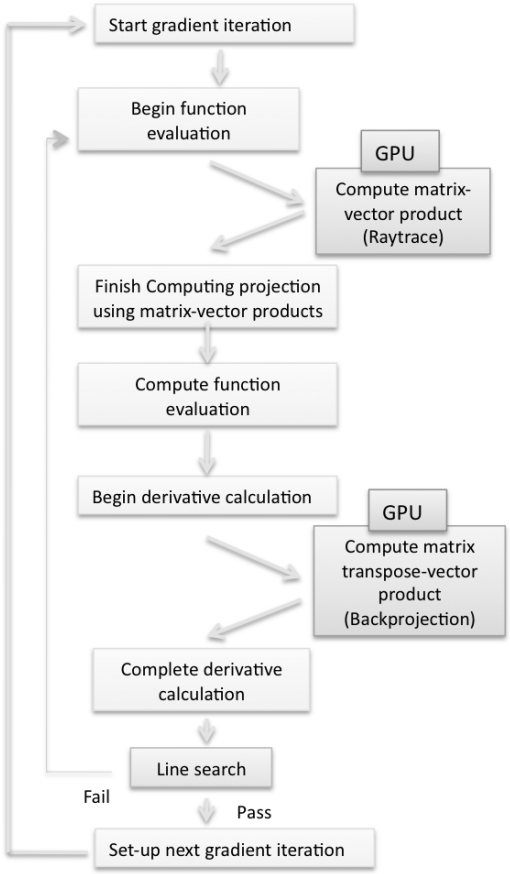


Figure 10.4: Matrix optimization algorithm scheme. The GPU performs the matrix products and the CPU applies fit coefficients to exponential inside the cost function and derivative evaluation, performs outer summations for the cost function and derivatives.

(10.1) given the results of the raytracing operation is a memory bound computation. In order to complete the cost function evaluation in Equation (10.1), we need to apply the fit coefficients to the results of the raytrace and sum over all energies in Equation (10.3), sum over all pixels in Equation (10.2), and sum over all angles in Equation (10.3). We can express this as a triple nested loop where the inner loop goes over all energies (order of a ten), the outer loop goes over all all pixels (order of a million), and the outside loop goes over all angles (order of ten). Loading this triple nested loop as a kernel into the device is possible but problematic because the power of a GPU is its ability to perform fast calculations and every layer of this loop is memory intensive.

A successful approach to speed up the computation of the function evaluation is to enhance the functionality of the raytracing kernel to apply the fit coefficients and process the

energy dependent loop for each pixel. Instead of unloading the quantities $A\mathbf{w}_{:,2}$, $A\mathbf{w}_{:,3}$, and $A\mathbf{1}$ from the device to evaluate Equation (10.3), we can keep these quantities in the device and load the fit coefficients into the GPU (given we have sufficient memory available) to perform the fit inside the kernel. In addition, we can use the exponential function provided by the OpenCL standard to compute the exponential portion. Therefore, we can expand the functionality of the kernel to compute the equation

$$\bar{\mathbf{b}}_\theta + \bar{\boldsymbol{\eta}} = \sum_{e=1}^{n_e} s_e \exp\left(-\left[c_{3,e} A_\theta \mathbf{w}_{:,3} + c_{2,e} A_\theta \mathbf{w}_{:,2} + c_{1,e} A_\theta \mathbf{1}\right]\right) + \bar{\boldsymbol{\eta}}$$

for all angles N_θ in one single kernel call. We can expand the functionality of the kernel even further by computing the inner loops

$$\sum_{e=1}^{n_e} s_e \left(u_{2,e} - u_{1,e}\right) \exp\left(-\sum_{j=1}^{N_v} a_\theta^{ij} \boldsymbol{\mu}_{e,j}\right)$$

and

$$\sum_{e=1}^{n_e} s_e \left(u_{2,e} - u_{1,e}\right) \exp\left(-\sum_{j=1}^{N_v} a_\theta^{ij} \boldsymbol{\mu}_{e,j}\right)$$

of the vectors $\mathbf{v}_2$ and $\mathbf{v}_3$ which are necessary to evaluate $\nabla(-L(\mathbf{b}_\theta, \mathbf{w}))$ in Equation (10.4). Once the kernel execution is completed, we have a single loop over all pixels to complete the evaluation of Equations (10.2) and (10.5). Note that these loops can run in the CPU very fast since it is only three lines of memory bound computations that use the cache to hide memory latency.

We call this a kernel fusion implementation because we have fused the nested loops into one kernel call as opposed to having separate kernels for the matrix product, the application of the fit coefficients to the data, and the evaluation of the vectors needed to compute the derivative of the cost function. Schematically, this approach is shown in Figure 10.5. The cost of extending the raytrace kernel is not significant. For a small size trial problem using 15 projections of size 360×648 pixels reconstructing a volume of 467×648×39 voxels, the fused kernel takes approximately 2.3 seconds to compute the matrix-vector products $A\begin{bmatrix}\mathbf{w}_{:,2} & \mathbf{w}_{:,3} & \mathbf{1}\end{bmatrix}$, the values $\bar{\mathbf{b}}_\theta + \bar{\boldsymbol{\eta}}$ for all N_θ and the derivative vectors $\mathbf{v}_2$ and $\mathbf{v}_3$. If we

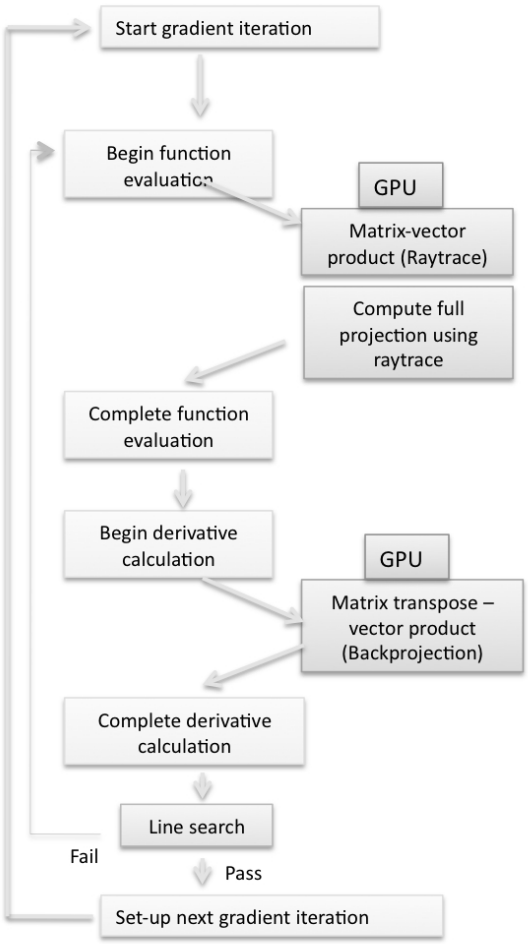


Figure 10.5: Fused kernel implementation scheme. The matrix-vector product kernel maintains vectors in the device to apply the energy fit coefficients and the exponential operation as well as compute the vectors needed for the derivative calculation.

run the optimized matrix-vector product kernel we have a computation time of 1.8 seconds for the raytrace and about 18 seconds for the function evaluation loop in the GPU. This represents a speed up of 8× for this operation. If we consider the clinical size problems fused in our numerical results, where we have 15 projections of size 1280×2048 pixels to reconstruct a volume of size 1280×2048×50, the advantage of a fused function evaluation approach is more noticeable. Performing the optimized matrix products on the GPU as described in the previous section, we have a computation time of about 11 seconds for $A\begin{bmatrix}\mathbf{w}_{:,2} & \mathbf{w}_{:,3} & \mathbf{1}\end{bmatrix}$ and 90 seconds to complete the evaluation of $\bar{\mathbf{b}}_\theta + \bar{\boldsymbol{\eta}}$ for all N_θ and the derivative vectors $\mathbf{v}_2$ and $\mathbf{v}_3$. However, using the fused kernel approach we can compute

all the above quantities in about 11 seconds, representing a speed up of about 9× for the operation.

One final optimization we maintain in this fused kernel approach is that CPU to GPU communication during the reconstruction process is absolutely minimal. The kernel calls for the raytracing and backprojection operation are almost identical since in both cases we just need to communicate to the geometry of the problem and the necessary vectors to the device. We have created a function call prior to the beginning of the reconstruction to load all necessary variables into the global memory of the GPU. This leads to only having to transfer the necessary vectors to the device at the time of enqueueing the kernel. In this approach the application spends most of the computation time running in the GPU and we have much faster line search computations. There is no additional optimizations needed by profiling the code because, as can be seen in the next chapter, with this application we are performing clinical size patient reconstructions in under five minutes.

10.4 Summary

In this chapter we have discussed the major implementation details for the digital breast tomosynthesis reconstruction problem. We have developed three approaches to solve the problem, a serial implementation, an implementation threaded for functionality, and a fused kernel approach using a single GPU. We have shown that the bottle necks of the application are the matrix products and the triple nested loop in the function evaluation. Avoiding an explicit matrix-vector product and matrix transpose-vector product and using a raytracing algorithm we can compute fast matrix products on a GPU. By fusing the function evaluation loops and extending the functionality of the raytracing kernel we reduce the bottle neck created by the memory bound computations of the function evaluation. Allowing minimal communication between CPU and GPU also reduces the time the GPU sits idle. Using all these performance optimizations, we show in the next chapter that our fused kernel approach attains the best performance and achieves our computational time goal, as it reconstructs the full clinical size data in under five minutes.

Chapter 11

Numerical Results

In this chapter we present some numerical results to show the effectiveness and computation time of our polyenergetic digital tomosynthesis reconstruction frameworks on five real data sets taken of different phantom breast objects with known materials. The five phantom breast objects are divided into two categories: two homogeneous phantoms and three heterogeneous phantoms. The two homogeneous phantoms are composed of a consistent background material with inserts of glandular tissue (called the *homogeneous-glands* data set) and micro-calcifications (called the *homogeneous-calcs* data set) which are two materials of interest in breast cancer detection. The background in these phantoms allows us to display the distinct features of the inserts for each data set and to quantitatively evaluate the image reconstruction quality. The three heterogeneous phantoms have a mixed background material that provides a more realistic representation of actual breast tissue. The heterogeneous phantoms are composed of a swirl of glandular and adipose tissue (called the *heterogeneous-swirl* data set), a swirl of adipose and glandular tissue with glandular tissue inserts (called the *heterogeneous-glands* data set), and a swirl of adipose and glandular tissue with micro-calcification inserts (called the *heterogeneous-calcs* data set). For the heterogeneous phantoms we are observing the detection of the given material inserts and the small features of the mixed background.

We compare the tomographs produced using our polyenergetic reconstruction frameworks against a filtered backprojection reconstruction and an MLEM reconstruction of each phantom. The projection images of each phantom object were acquired using a clinical

breast tomosynthesis system (Selenia Dimensions, Hologic Inc.) that is used exclusively for research. This system acquires 15 projections over a 15 degree angular range and produces a reconstruction of the imaged volume using the filtered backprojection (FBP) algorithm (see Section 3.4). Filtered backprojection is often used in commercial systems because it is a well-understood method that computes reconstructions very quickly. However, as discussed in Section 3.4, both filtered backprojection and the MLEM reconstruction approximate Beer's law by assuming a monoenergetic x-ray model. As we will see in this chapter, this assumption leads to artifacts that are not present in our polyenergetic reconstructions. To compare the reconstructions of the polyenergetic frameworks with the filtered backprojection reconstructions produced by the tomosynthesis system and the MLEM reconstructions, the 2 or 3 separate materials reconstructed were combined to obtain one image representing the estimated linear attenuation coefficient of each voxel, and these were then transformed into Hounsfield Units (HU) [46].

For our numerical results, each phantom was reconstructed using the polyenergetic reconstruction frameworks (the linear model for attenuation, the quadratic model for attenuation, and the simplified multi-material model for attenuation), and the monoenergetic reconstruction approaches (filtered backprojection and MLEM). For each data set we compare some slices (see Figure 2.11) of the reconstructions and highlight particular features of interest. We also show the behavior of the relative function value in our iterative processes. Finally, we compare the runtime for our efficient GPU implementation described in chapter 10 to show how it outperforms a serial implementation to gain up to 28$\times$ speed-up. For these results, the linear model for attenuation refers to the model described in Chapter 4 where we are assuming the presence of only adipose and glandular tissue inside the object, the quadratic model for attenuation described in Chapter 5 expands on the linear model by accounting for the presence of air and micro-calcifications inside the object, and the simplified multi-material model described in Chapter 7 assumes the presence of adipose and glandular tissue as well as micro-calcifications in the object. Reconstructions using the multi-material model described in Chapter 6 are shown separately at the end of this chapter.

A special thank you to Dr. Ioannis Sechopoulos with the Department of Radiology and

Imaging Sciences and Winship Cancer Institute at Emory University and Steve S. J. Feng with Georgia Institute of Technology for acquiring the phantom data and processing the signal analysis information.

11.1 Homogeneous Phantoms

11.1.1 Homogeneous Phantom with Glandular Tissue Inserts

The first phantom we used was made up of four stacked semi-circular, 1 cm thick plates consisting of a material equivalent to a homogeneous mixture of 50% breast adipose and 50% breast glandular tissue. A fifth plate was inserted in the middle consisting of a material equivalent to adipose tissue with nine 1 cm diameter holes throughout. The nine holes in the center plate were filled with 1 cm diameter targets consisting of different materials. For the images involving soft tissue lesions, six of the hole-filling targets consisted of an adipose-glandular tissue mixture with varying glandular-to-adipose ratios: 0% : 100%, 20% : 80%, 40% : 60%, up to 100% : 0%. The remaining three holes were filled with other targets that are not used for this analysis. We begin by using this homogeneous phantom because its constant background allows for simpler objective analysis of the lesion signal and image quality.

Figure 11.1 shows the behavior of the relative function value for 15 gradient descent iterations of our polyenergetic reconstructions. All three polyenergetic reconstructions required less than 15 iterations to produce a good reconstruction of the phantom. It is important to note that we currently do not have an automatic approach to choosing regularization parameters (in this case, the stopping iteration), and so our choice of stopping at iteration 15 was determined by experimentation. Figure 11.5 shows the reconstructed slices for the phantom using our polyenergetic frameworks, filtered backprojection, and MLEM. The reconstructions were produced using 5 to 8 iterations of gradient descent for the polyenergetic frameworks and 10 iterations for MLEM.

The signal difference-to-noise ratio (SDNR), which is a common image quality metric,

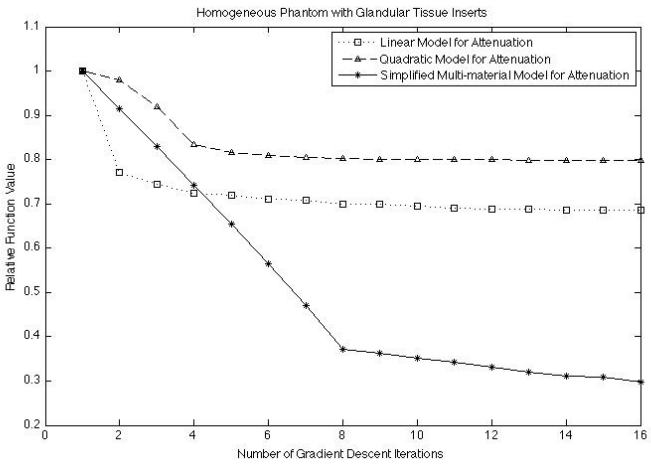


Figure 11.1: Relative function value of gradient descent iterations for the reconstruction of the homogeneous phantom with glandular tissue inserts. Note that the relative function value for all three methods begins to stagnate after 10 iterations. As expected, this is when the ill-posedness of the problem corrupts the solution.

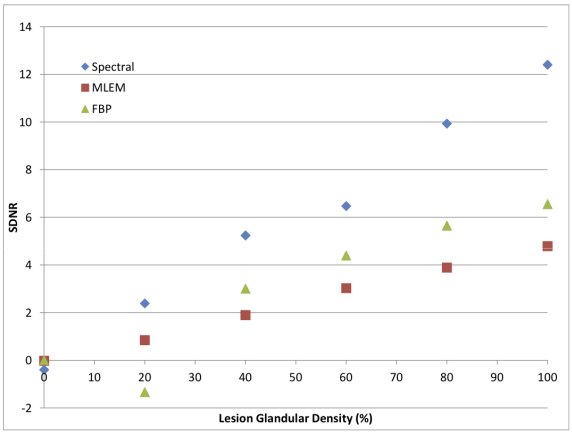


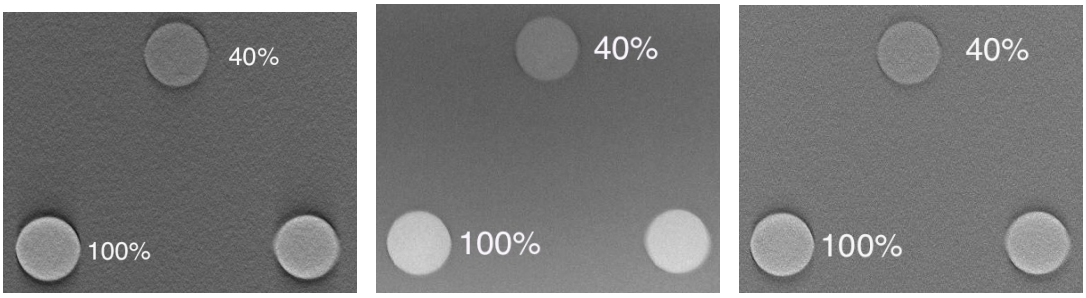
Figure 11.2: Signal difference-to-noise ratio (SDNR) for each glandular insert of the homogeneous-glands data set. A higher ratio signifies a more distinguishable lesion in the image.

was computed for each of the glandular inserts using the equation

$$\text{SDNR} = \frac{\mu_{\text{signal}} - \mu_{\text{back}}}{\sigma_{\text{back}}}$$

where μ_{signal} denotes the mean value of the voxels inside a region of interest (ROI) that represent a lesion, μ_{back} denotes the mean value of the voxels inside the ROI that represent the background, and σ_{back} denotes the standard deviation of the background ROI. Figure 11.2

shows the SDNR for each of the glandular tissue inserts in the phantom using the filtered backprojection reconstruction, the MLEM reconstruction, and our simplified multi-material reconstruction. It is clear that our simplified multi-material reconstruction achieves a higher SDNR for all glandular tissue inserts. A zoomed-in version of the some of the glandular lesions for the filtered backprojection, simplified multi-material model, and MLEM are shown in Figure 11.3. Note that in addition to having more distinguishable features of interest



(a) Filtered Backprojection (b) Simplified Multi-material (c) MLEM

Figure 11.3: A close up of tomograph 25 for the filtered backprojection, our simplified multi-material model for attenuation, and MLEM reconstructions showing the percentage glandular tissue insert. The results of Figures 11.3(a) and 11.3(c) assume a monoenergetic x-ray beam where as Figure 11.3(b) uses a polyenergetic spectrum in the forward model.

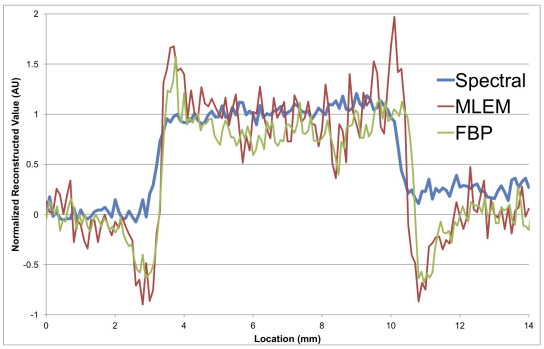


Figure 11.4: Vertical profile through the center of the 100% glandular tissue lesion shown in Figure 11.3 for the filtered backprojection, MLEM, and simplified multi-material reconstrucution. The values were normalized by subtracting the average of a background region and dividing by the average of signal region in each corresponding to display profile in the same scale.

in the image, our polyenergetic reconstruction 11.3(b) does not show the cupping artifacts

GPU Implementation Reconstruction Times			
Number of Gradient Descent Iterations	Linear Model for Attenuation	Quadratic Model for Attenuation	Simplified Multi-material Model for attenuation
5	6.3 min	3.8 min	2.6 min
8	11.2 min	6.1 min	4.4 min
15	23.2 min	11.4 min	11 min

Serial Implementation Reconstruction Times			
Number of Gradient Descent Iterations	Linear Model for Attenuation	Quadratic Model for Attenuation	Simplified Multi-material Model for attenuation
5	180 min	112 min	68 min
15	603 min	340 min	290 min

Table 11.1: Runtime in minutes for reconstruction of the homogeneous phantom with glandular tissue inserts data set. Note that not all iterations for each model take the same amount of time because of the line search operation. Some iterations take longer than others as the step length parameter must be refined.

around the glandular tissue inserts present in the monoenergetic reconstructions Figures 11.3(a) and 11.3(c).

In order to further compare these lesions of interest in the reconstructed phantom, Figure 11.4 shows the vertical signal profile through the center of the 100% glandular tissue lesion. The vertical signal profile shows that the filtered backprojection and MLEM reconstruction produce the same cupping artifacts around the glandular insert. Both curves show a very similar spike and drop at the edges of the feature as they seem to overestimate the signal at the boundary of insert and underestimate the background signal at the boundary of the lesion. In contrast, our simplified multi-material model produces a very stable curve that clearly marks the boundaries of the lesion without overestimating it and reconstructs the signal of the background equally everywhere around the lesion.

The reconstruction times for the serial implementation and our GPU implementation as described in Chapter 10 are shown in Table 11.1. It is important to note that our GPU implementation can outperform serial runtime by 26× up to 30× speed-up for this data set. This shows that quality reconstructions with our polyenergetic framework can be obtained in under five minutes (8 iterations).

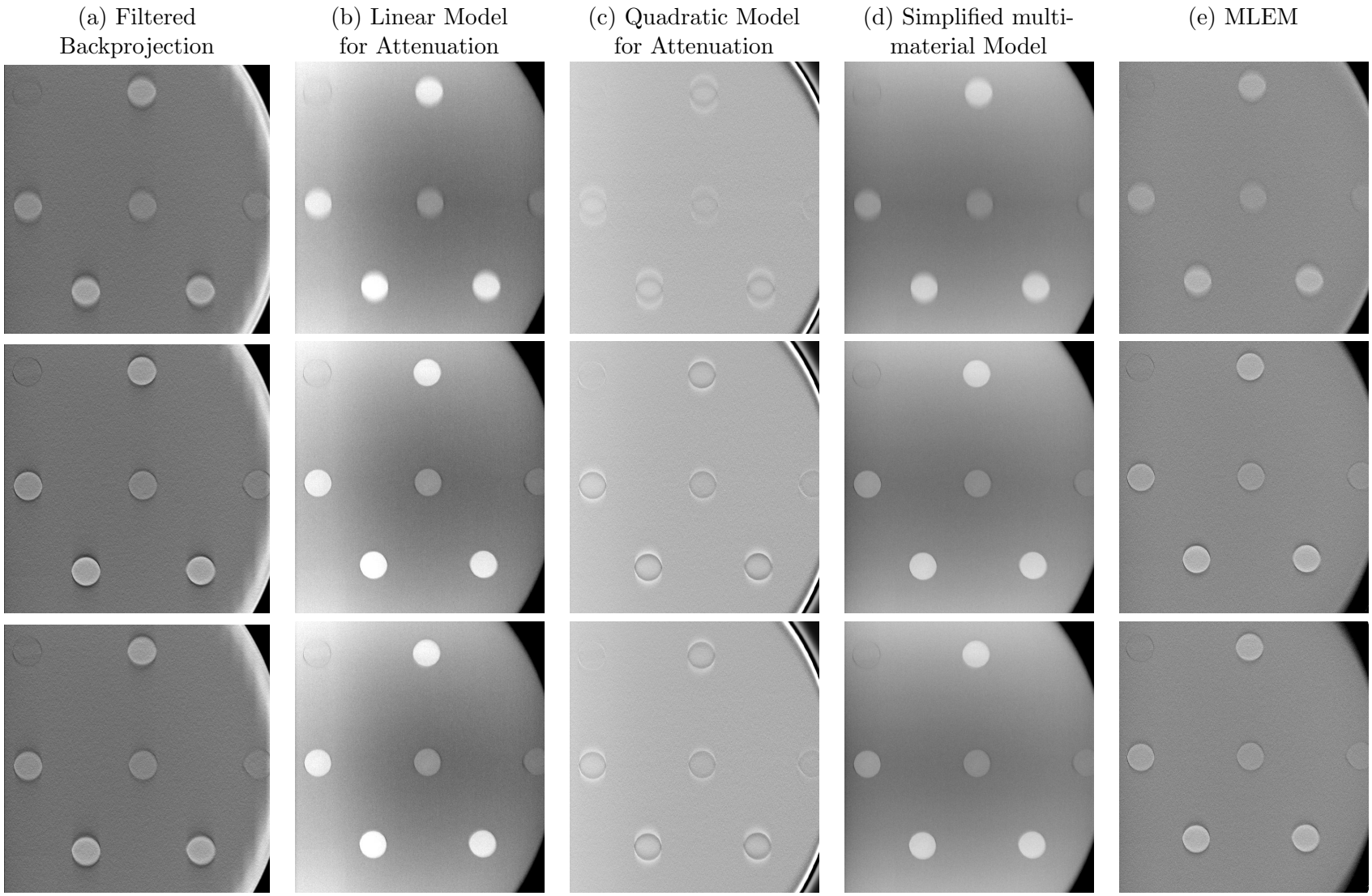


Figure 11.5: Results for homogeneous phantom with glandular tissue inserts. Tomographs 15 (top row), 25 (middle row), and 29 (bottom row) of the reconstructed pseudo 3D volume are displayed where each column represents the slices reconstructed using the given method. The middle row is the center of the reconstructed volume, where glandular tissue feature is most distinguishable.

11.1.2 Homogeneous Phantom with Micro-calcification Inserts

The second homogeneous phantom was set-up similarly to the phantom described in the previous section but we inserted a material equivalent to micro-calcifications in the center plate. The phantom consisted of four stacked semi-circular, 1 cm thick plates consisting of a material equivalent to a homogeneous mixture of 50% breast adipose and 50% breast glandular tissue with a fifth plate consisting of a material equivalent to adipose tissue in the center. The center plate contained nine holes of 1 cm diameter that were filled with a material equivalent 100% breast adipose and added inserts representing calcium specks of diameter sizes 0.130mm, 0.165mm, 0.196mm, 0.230mm, 0.290 and 0.400 mm.

Figure 11.6 shows the behavior of the relative function value for 15 gradient descent iterations of our polyenergetic reconstructions. All three polyenergetic reconstructions re-

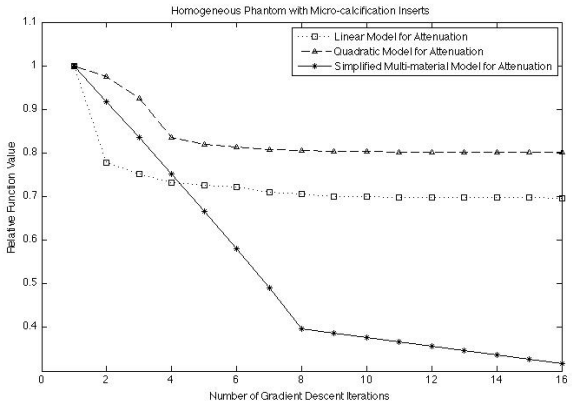
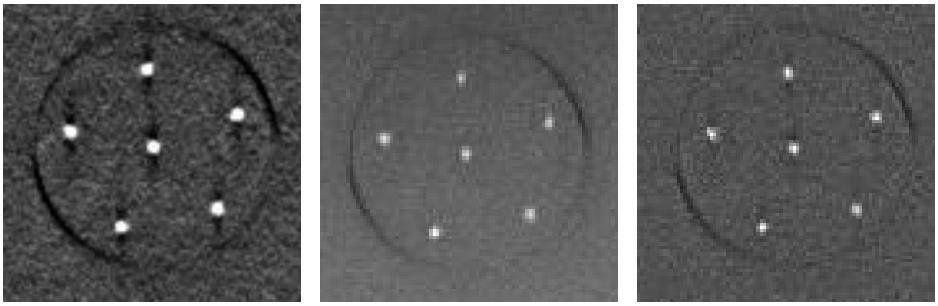


Figure 11.6: Relative function value of gradient descent iterations for the reconstruction of the homogeneous phantom with glandular tissue inserts.

quired less than 15 iterations to produce a good reconstruction of the phantom. Figure 11.10 shows the reconstructed slices for the phantom using our polyenergetic frameworks, filtered backprojection, and MLEM. The reconstructions were produced using 5 to 8 iterations of gradient descent for the polyenergetic frameworks and 10 iterations for MLEM.

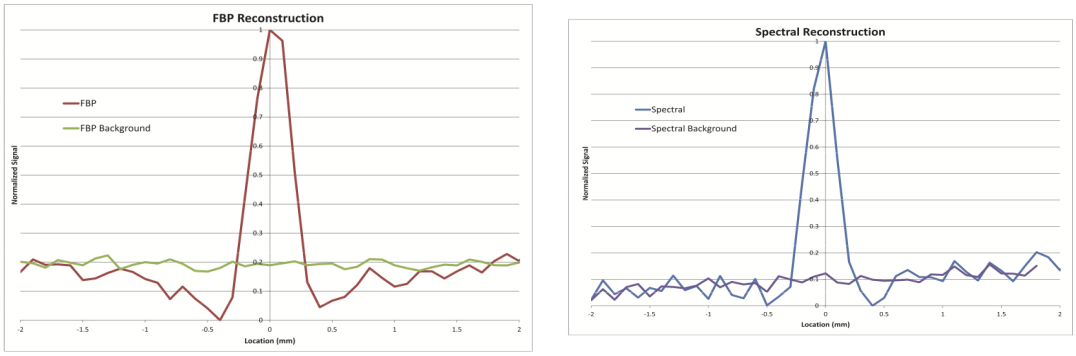
Figure 11.7 shows a close up of the micro-calcification features for the filtered backkprojection, our simplified multi-material model, and MLEM reconstructions. In this figure it is clear that our polyenergetic framework reduces the presence of shadow-like artifacts around



(a) Filtered Backprojection (b) Simplified Multi-material (c) MLEM

Figure 11.7: A close up of tomograph 25 for the filtered backprojection, simplified multi-material model for attenuation, and MLEM reconstructions shows the presence of beam hardening artifacts. Figures 11.7(a) and 11.7(c) assume a monoenergetic x-ray beam whereas Figure 11.7(b) uses a polyenergetic spectrum in the forward model. Note the cupping artifacts around the glandular feature for the monoenergetic reconstructions which are not present in our reconstruction.

the lesions of interest. A comparison of the vertical signal profile for this micro-calcification cluster in the filtered backprojection and in our polyenergetic simplified multi-material model reconstruction is shown in Figure 11.8. Each graph corresponds to the vertical profile



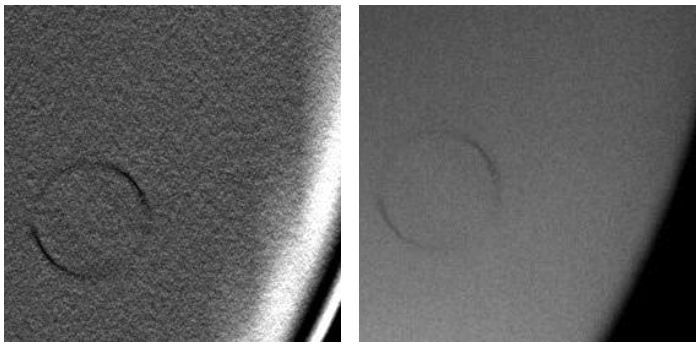
(a) Filtered Backprojection (b) Simplified Multi-material

Figure 11.8: A comparison of the vertical signal profile for this micro-calcification cluster in the filtered backprojection (FBP) and simplified multi-material model (spectral) reconstructions

through the center of the middle speck and the vertical profile of the background next to the speck. The spike in the graph represents the signal corresponding to the micro-calcification. To allow for a comparison between the reconstruction methods using the same scale, the

voxel values of both the signal and background profiles for each method were normalized by subtracting the minimum value of the corresponding signal profile and then dividing by the maximum value. Note that in the profile for filtered backprojection in Figure 11.8(a) the signal at the micro-calcification speck is lower than the background signal at least 1 mm from the center in each direction, while in the profile for the simplified multi-material model in Figure 11.8(b) the signal matches the background almost immediately, with some slight and narrower reduction only present towards one side. This drop in the signal for the micro-calcification corresponds to the cupping-like artifact in Figure 11.7(a). It can also be seen that the graph in Figure 11.8(b) produces a narrower peak than the peak in 11.8(a), representing improved spatial resolution by the simplified multi-material model reconstruction over the filtered backprojection results.

Another feature of interest in the reconstructions is shown in Figure 11.9. Here we see that our reconstructions eliminate the boundary artifacts that are present in the filtered backprojection reconstruction provided by the imaging system.



(a) Filtered Backprojection (b) Simplified Multi-material

Figure 11.9: A close up of tomograph 25 for the filtered backprojection and simplified multi-material model for attenuation shows a boundary artifact for the filtered backprojection reconstruction not present in our polyenergetic reconstruction.

The reconstruction times for the serial implementation and our GPU implementation as described in Chapter 10 are shown in Table 11.2. It is important to note that our GPU implementation can outperform serial runtime by 25× up to 28× speed-up. Quality reconstructions that account for a polyenergetic spectrum using the simplified multi-material model for attenuation can be obtained in under five minutes (8 iterations).

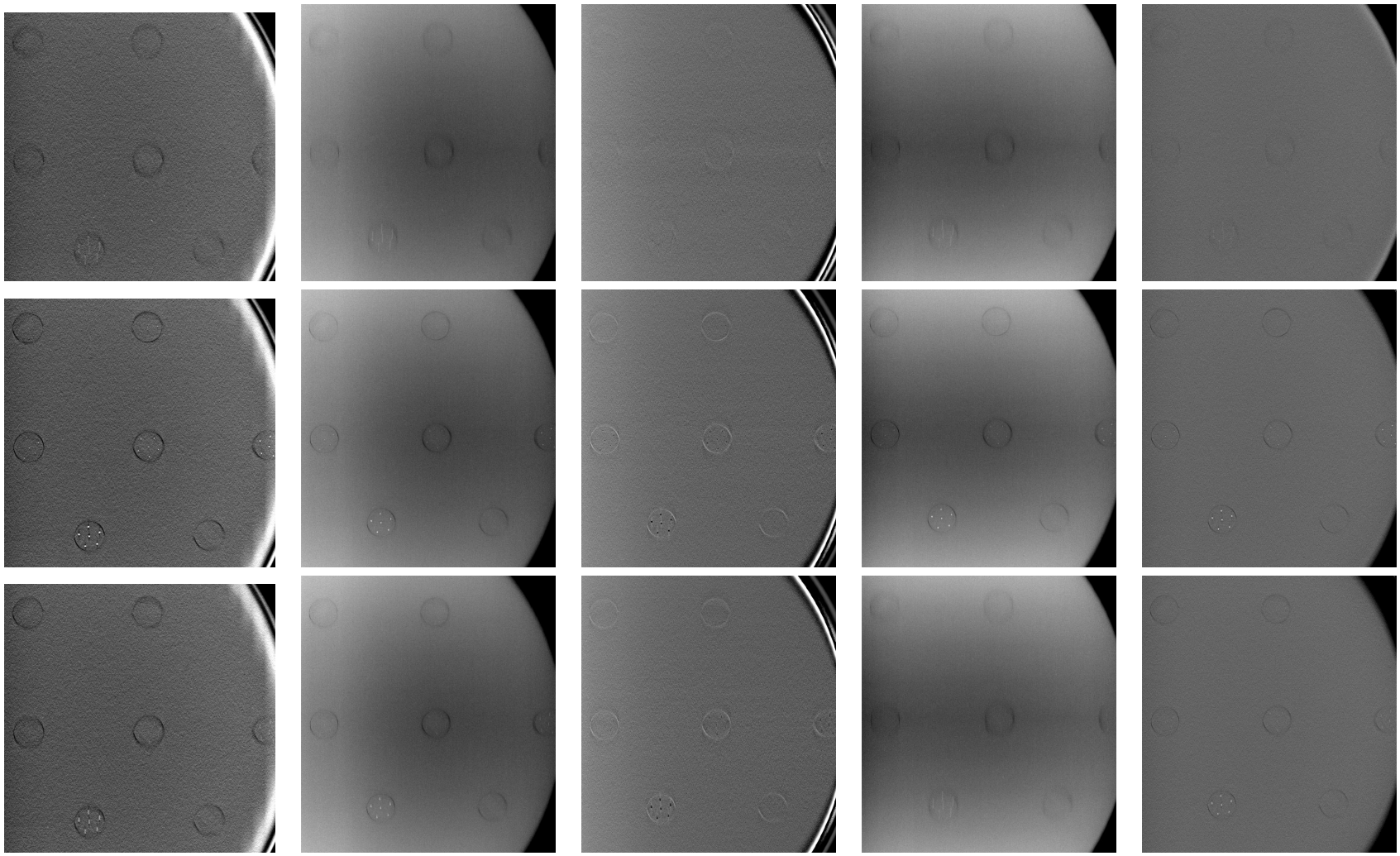


Figure 11.10: Results for homogeneous phantom with micro-calcification inserts. Tomographs 14 (top row), 25 (middle row), and 29 (bottom row) of the reconstructed pseudo 3D volume are displayed where each column represents the slices reconstructed using the given method. The middle row is the center of the reconstructed volume, where micro-calcification feature is most distinguishable.

GPU Implementation Reconstruction Times			
Number of Gradient Descent Iterations	Linear Model for Attenuation	Quadratic Model for Attenuation	Simplified Multi-material Model for attenuation
5	6.4 min	3.8 min	2.6 min
8	11.1 min	6.1 min	4.4 min
15	21.9 min	11.4 min	10.6 min

Serial Implementation Reconstruction Times			
Number of Gradient Descent Iterations	Linear Model for Attenuation	Quadratic Model for Attenuation	Simplified Multi-material Model for attenuation
5	179 min	111 min	67 min
15	572 min	336 min	285 min

Table 11.2: Runtime in minutes for reconstruction of the homogeneous phantom with microcalcification inserts data set.

11.2 Heterogeneous Phantoms

11.2.1 Heterogeneous Phantom Glandular and Adipose Tissue Swirl

The first heterogeneous phantom used was made up of five stacked semi-circular, 1 cm thick plates composed of a heterogeneous mixture of materials equivalent to breast adipose a breast glandular tissue. This heterogeneous phantom provides a more realistic representation of actual breast tissue. In addition, we use a heterogeneous phantom because the the presence of overlapping signals allows us to better test the abilities of the tomosynthesis system and the reconstruction framework. However, in general for heterogeneous phantoms, we can only provide a qualitative evaluation because of the random structure of the background.

Figure 11.11 shows some slices of the reconstructed volume using filtered backprojection and our polyenergetic frameworks (the linear model for attenuation, the quadratic model for attenuation, and the simplified muIti-material model). It can be seen that our polyenergetic frameworks depict the features of the heterogeneous background as well as the filtered backprojection reconstruction. In addition, we see that our linear model for attenuation and the simplified multi-material model for attenuation give a smoother reconstruction

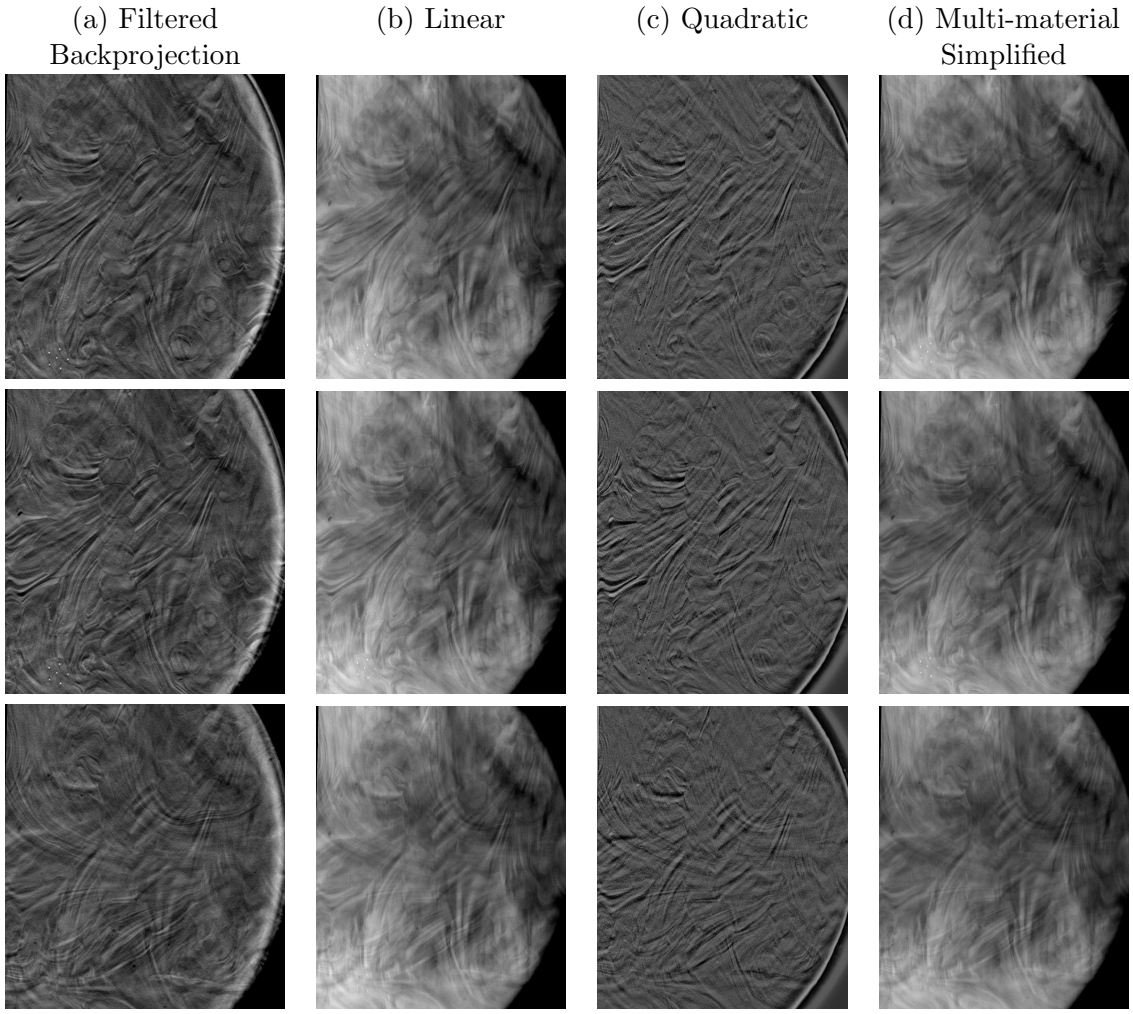


Figure 11.11: Tomographs 25 (top row), 29 (middle row), and 34 (bottom row) of the reconstructed pseudo 3D volume are displayed where each column represents the slices reconstructed using the given method. The middle row is the center of the reconstructed volume, where micro-calcification feature is most distinguishable.

that does not have the boundary artifacts present in the filtered backprojection slices or the reconstruction produced using a quadratic model for attenuation.

Figure 11.6 shows the behavior of the relative function value for 15 gradient descent iterations of our polyenergetic reconstructions. All three polyenergetic reconstructions required less than 15 iterations to produce a good reconstruction of the phantom, but it is important to note that our simplified multi-material forward model provides a steepest descent in the relative function value. In Figure 11.13 we show the spectral distribution for the x-ray at entrance and exit of the phantom. Note that the phantom absorbs the low-energy photons

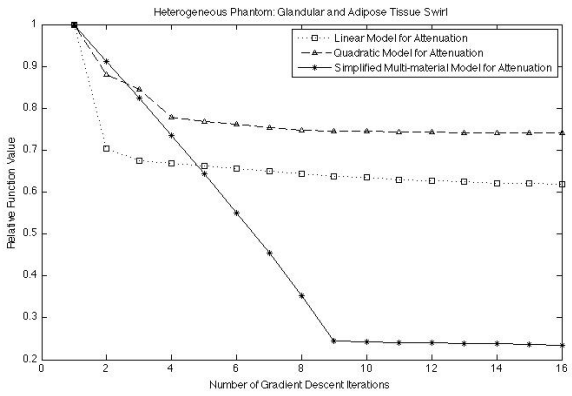


Figure 11.12: Relative function value of gradient descent iterations for the reconstruction of the heterogeneous phantom with adipose and glandular tissue swirl.

in the x-ray beam, thus changing the average energy of the spectrum. This is the beam hardening effect discussed in Section 3.5. Finally, Table 11.3 shows the reconstruction times for this phantom using the GPU implementation framework described in Chapter 10.

Number of Gradient Descent Iterations	Linear Model for Attenuation	Quadratic Model for Attenuation	Simplified Multi-material Model for attenuation
5	9.5 min	4.4 min	3.1 min
15	29.6 min	13.3 min	15.3 min

Table 11.3: Runtime in minutes for reconstruction of the swirl phantom. Note that not all iterations for each model take the same amount of time because of the line search operation. Some iterations take longer than others as the step length parameter must be refined.

11.2.2 Heterogeneous Phantom with Glandular Tissue Inserts

The second heterogeneous phantom used consisted of four stacked semi-circular, 1 cm thick plates composed of a material equivalent to a heterogeneous mixture of breast adipose and breast glandular tissue. A fifth plate was inserted in the middle consisting of the same heterogeneous material mixture as the other plates with nine 1 cm diameter holes throughout. The six holes in the center plate were filled with 1 cm diameter targets consisting of an adipose-glandular tissue mixture with varying glandular-to-adipose ratios: 0% : 100%, 20% : 80%, 40% : 60%, up to 100% : 0%. The remaining three holes were filled with other targets that are not used for this analysis. As in the previous phantom, the heterogeneous mixture provides a more realistic representation of actual breast tissue and overlapping

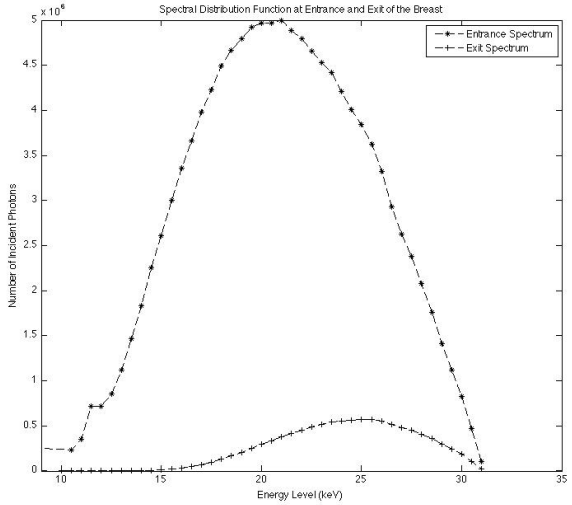


Figure 11.13: Spectral distribution function at entrance and exit of breast for heterogeneous swirl phantom. Note the preferential absorption of low energy photons.

signals allows us to better test the abilities of the reconstruction framework. However, here we can only provide a qualitative evaluation because of the random structure of the background.

Figure 11.14 shows the behavior of the relative function value for 15 gradient descent iterations of our polyenergetic reconstructions which is similar to the behavior observed in the previous phantoms. Again, we see that the relative function value for our linear

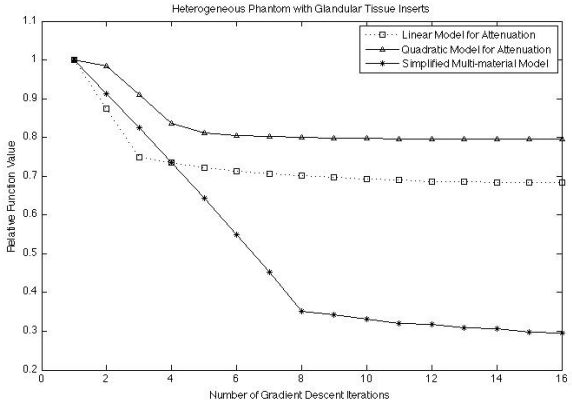


Figure 11.14: Relative function value of gradient descent iterations for the reconstruction of the heterogeneous phantom with glandular tissue inserts.

and quadratic models for attenuation stagnates sooner than the relative function value for the simplified multi-material attenuation model, suggesting the latter numerically provides

GPU Implementation Reconstruction Times			
Number of Gradient Descent Iterations	Linear Model for Attenuation	Quadratic Model for Attenuation	Simplified Multi-material Model for attenuation
5	6.8 min	3.8 min	2.6 min
8	9.9 min	6.1 min	4.4 min
15	19 min	11.8 min	12.7 min

Serial Implementation Reconstruction Times			
Number of Gradient Descent Iterations	Linear Model for Attenuation	Quadratic Model for Attenuation	Simplified Multi-material Model for attenuation
5	178 min	112 min	67 min
15	602 min	347 min	324 min

Table 11.4: Runtime in minutes for reconstruction of the heterogeneous phantom with glandular tissue inserts.

a better reconstruction framework. All three polyenergetic reconstructions required less than 15 iterations to produce a good reconstruction of the phantom and the tomographs are compared against the filtered backprojection and MLEM results in Figure 11.15. The reconstructions shown were produced using 5 to 8 iterations of gradient descent for the polyenergetic frameworks and 10 iterations for MLEM.

The reconstruction times for the serial implementation and our GPU implementation as described in Chapter 10 are shown in Table 11.2. For this data set, we see similar speed-up as in the homogeneous phantoms, namely 25× up to 29× of our GPU implementation over serial runtime. This shows that our reconstruction framework is capable of handling the more complicated breast-like heterogeneous background structures in about the same time as the easier homogeneous case. For this phantom, we provided quality reconstructions that account for a polyenergetic spectrum using the simplified multi-material model for attenuation in under five minutes (8 iterations).

Figure 11.16 shows a zoomed-in version of the some of the glandular lesions for the filtered backprojection, simplified multi-material model, and MLEM reconstructions of Figure 11.15. Note that in addition more distinguishable features of interest in the image, our polyenergetic reconstruction 11.3(b) shows no cupping artifacts around the glandular tissue inserts and a smoother image.

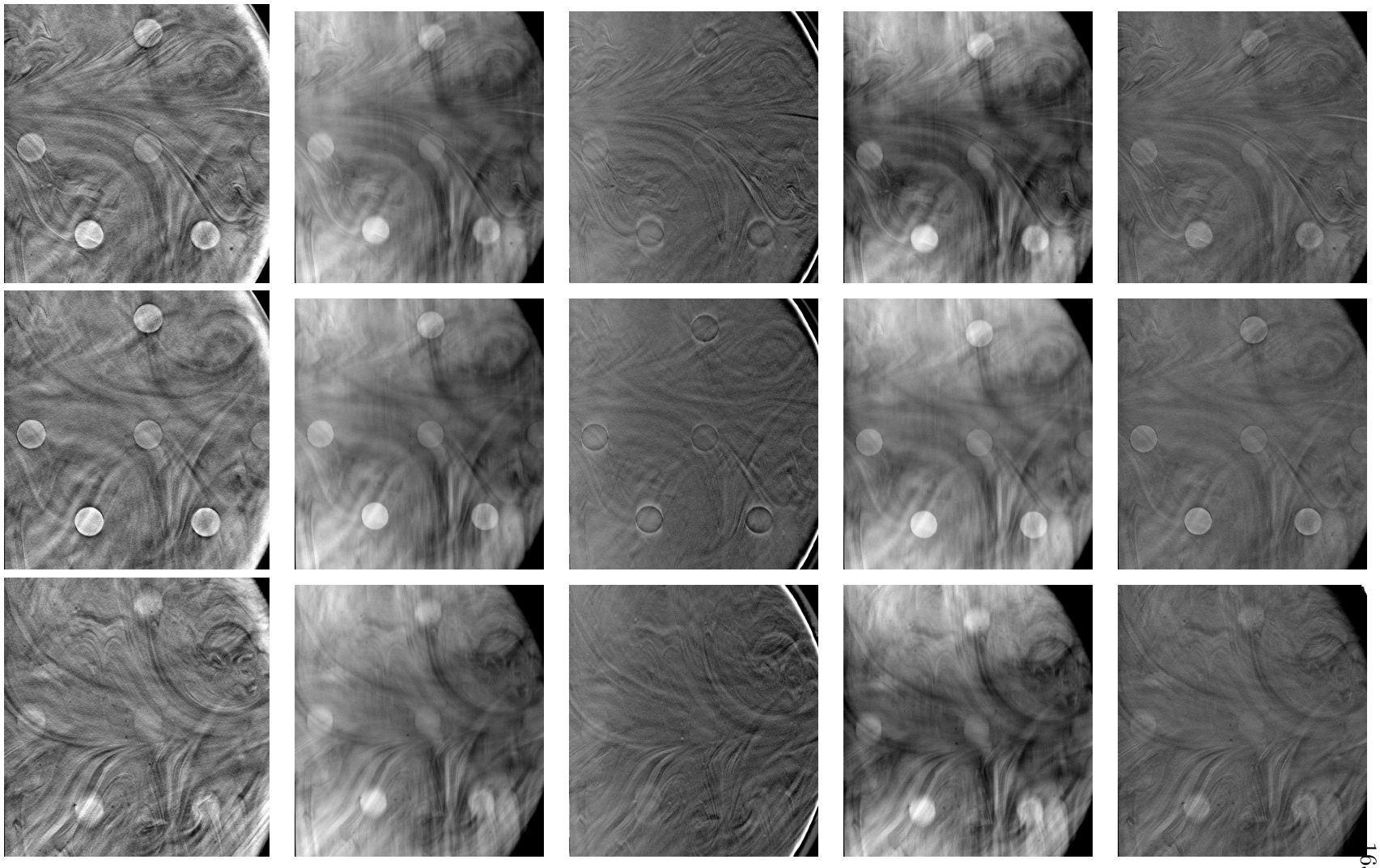
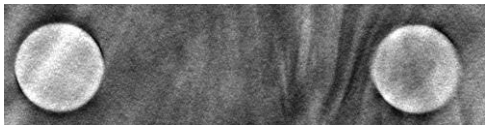


Figure 11.15: Results for heterogenous phantom with glandular tissue inserts. Tomographs 18 (top row), 25 (middle row), and 38 (bottom row) of the reconstructed pseudo 3D volume are displayed where each column represents the slices reconstructed using the given method. The middle row is the center of the reconstructed volume, where glandular tissue feature is most distinguishable.



(a) Filtered Backprojection

(b) Simplified Multi-material

(c) MLEM

Figure 11.16: A close up of tomograph 25 for the filtered backprojection, simplified multi-material model for attenuation, and MLEM reconstructions shows the presence of beam artifacts. Figures 11.16(a) and 11.16(c) assume a monoenergetic x-ray beam where as Figure 11.16(b) uses a polyenergetic spectrum in the forward model. Note the cupping artifacts around the glandular feature for the monoenergetic reconstructions which are not present in our reconstruction. Also, the monoenergetic reconstructions show noise.

11.2.3 Heterogeneous Phantom with Micro-calcification Inserts

The last phantom reconstructed consisted of five stacked semi-circular, 1 cm thick plates composed of a materials equivalent to a heterogeneous mixture breast adipose and breast glandular tissue. The middle plate contained nine 1 cm diameter holes throughout. Six of the holes were filled with a material equivalent to the heterogeneous background mixture with added inserts representing calcium specks of diameter sizes 0.130mm, 0.165mm, 0.196mm, 0.230mm, 0.290 and 0.400 mm. As in the previous phantom, the heterogeneous mixture provides a more realistic representation of actual breast tissue and overlapping signals allows us to better test the abilities of the reconstruction framework. In addition, the calcification specs test our framework's ability to distinguish small features of interest present inside the randomly structured background of breast tissue. In this section we provide only a qualitative evaluation of the reconstruction.

Figure 11.17 shows the behavior of the relative function value for 15 gradient descent iterations of our polyenergetic reconstructions which is similar to the behavior observed in the previous phantoms. Again, we see a steeper descent in the relative function value

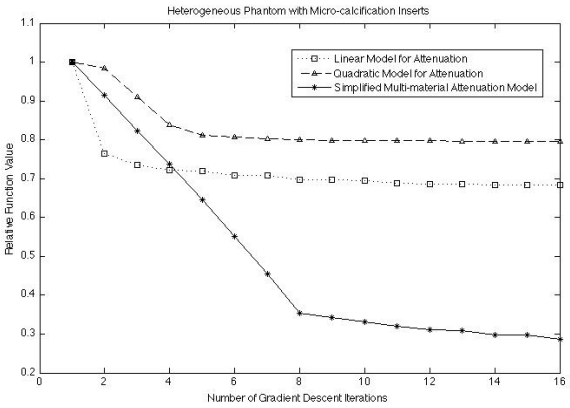
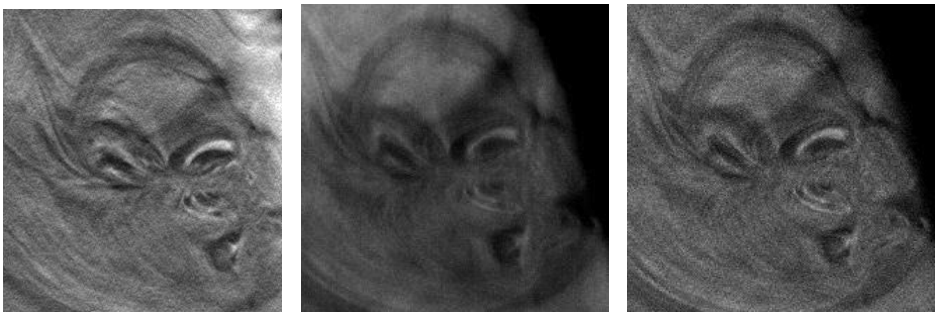


Figure 11.17: Relative function value of gradient descent iterations for the reconstruction of the homogeneous phantom with glandular tissue inserts.

of the simplified multi-material model, suggesting the it may be the better forward model numerically of those proposed in this thesis. All three polyenergetic reconstructions required less than 15 iterations to produce a good reconstruction of the phantom and the tomographs are compared against the filtered backprojection and MLEM results in Figure 11.19.



(a) Filtered Backprojection (b) Simplified Multi-material (c) MLEM

Figure 11.18: A close up of tomograph 43 for the filtered backprojection, simplified multi-material model for attenuation, and MLEM reconstructions shows our polyenergetic reconstruction picks up the heterogeneous background features.

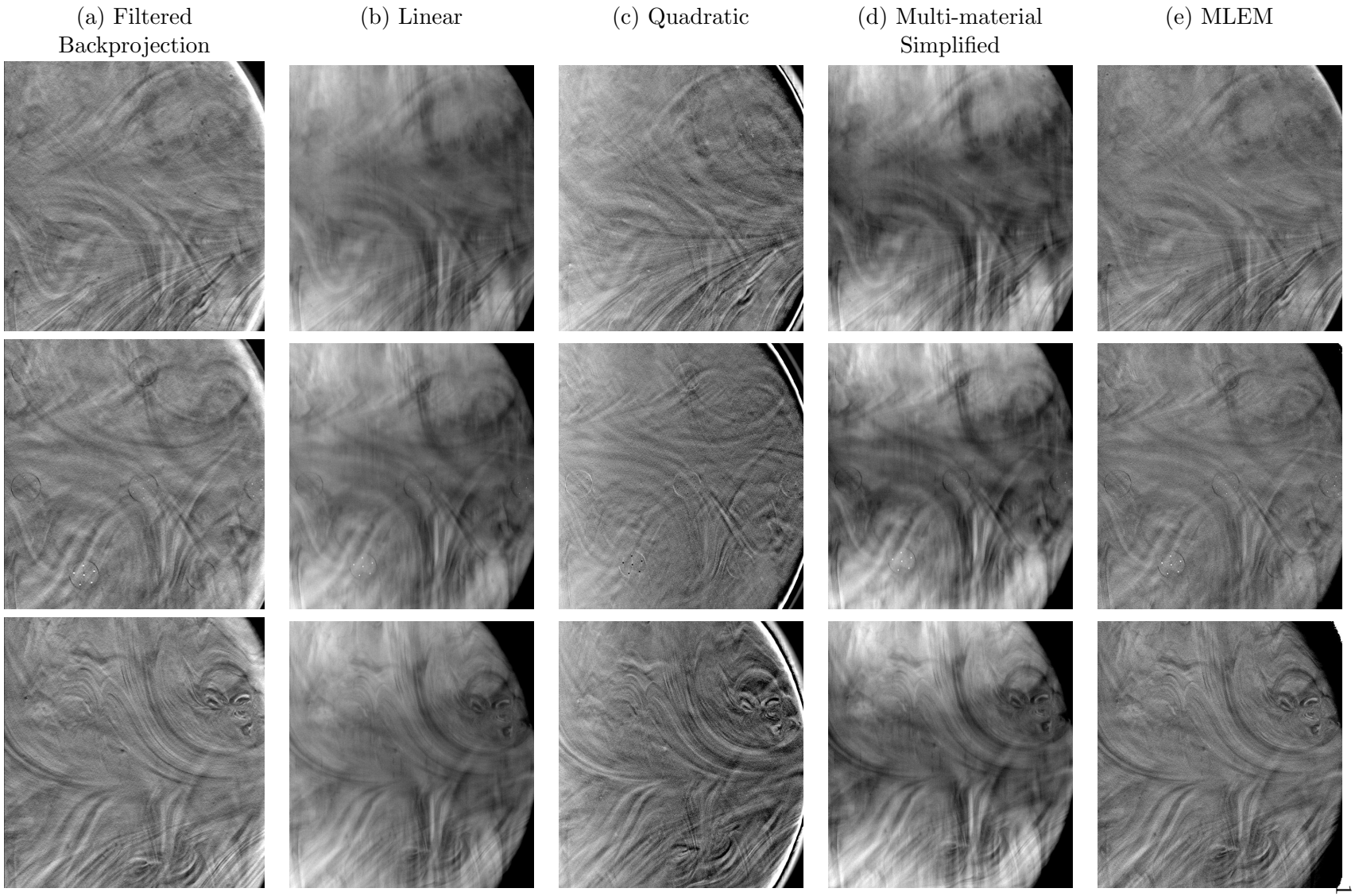


Figure 11.19: Results for heterogenous phantom with glandular tissue inserts. Tomographs 18 (top row), 25 (middle row), and 38 (bottom row) of the reconstructed pseudo 3D volume are displayed where each column represents the slices reconstructed using the given method. The middle row is the center of the reconstructed volume, micro-calcification feature is most distinguishable.

GPU Implementation Reconstruction Times			
Number of Gradient Descent Iterations	Linear Model for Attenuation	Quadratic Model for Attenuation	Simplified Multi-material Model for attenuation
5	6.9 min	3.8 min	2.6 min
8	11.6 min	6.1 min	4.4 min
15	23.3 min	11.8 min	10.6 min

Serial Implementation Reconstruction Times			
Number of Gradient Descent Iterations	Linear Model for Attenuation	Quadratic Model for Attenuation	Simplified Multi-material Model for attenuation
5	178 min	112 min	67 min
15	603 min	347 min	313 min

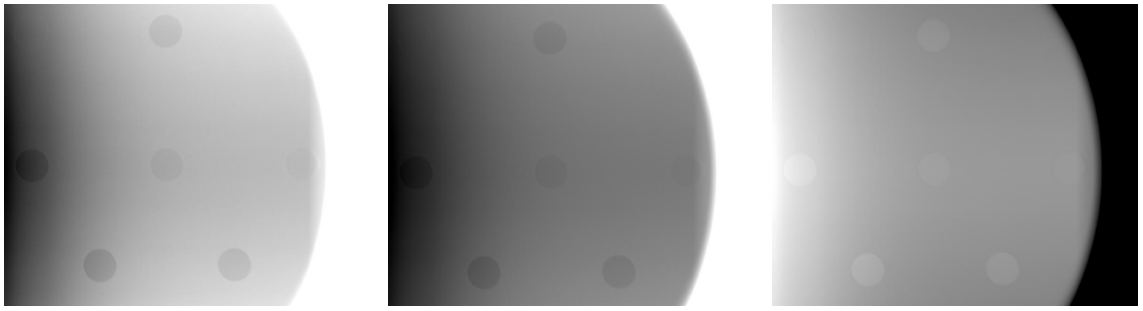
Table 11.5: Runtime in minutes for reconstruction of the heterogeneous phantom with micro-calcification inserts.

The reconstructions shown were produced using 5 to 8 iterations of gradient descent for the polyenergetic frameworks and 10 iterations for MLEM. Figure 11.18 shows a close up of the background features for the filtered backprojection, our simplified multi-material model, and MLEM reconstructions. Note that the features of the heterogeneous mixture are accurately reconstructed by our algorithm. In addition, the images produced using the simplified multi-material forward model are not as grainy as the other two reconstructions.

The reconstruction times for the reconstruction of this data set using the serial implementation and our GPU implementation as described in Chapter 10 are shown in Table 11.5. Again, we have that our GPU implementation can outperform serial runtime by up to 28× speed-up. A quality reconstruction for this phantom that accounts for a polyenergetic spectrum using the simplified multi-material model for attenuation can be obtained in under five minutes (8 iterations).

11.3 Multimaterial Model Results

To show the effectiveness of the multi-material model described in Chapter 6 we reconstructed the two homogeneous phantoms. First, we have the homogenous phantom with glandular tissue inserts which was reconstructed using the multi-material model for attenua-



(a) Glandular weight fractions (b) Adipose weight fractions (c) Air weight fractions

Figure 11.20: Tomograph 25 of the reconstruction of the homogeneous-glands phantom using the full multi-material model. The reconstruction was done using two gradient descent iterations

tion assuming the presence of glandular tissue, adipose tissue, and air. The weight fractions for each material in the middle tomograph of the reconstructed volume after two gradient descent iterations are shown in Figure 11.20. As expected, the glandular and adipose weight fractions capture the inside of the phantom since the materials are complementary for this problem, and the values of the glandular weight fractions are higher around the features of interest. Note also that the air weight fractions do not depict the inside of the phantom as well as the other two images since we are decomposing the volume into individual materials and there is no air inside the phantom. Figure 11.20 shows the signal profile of the three

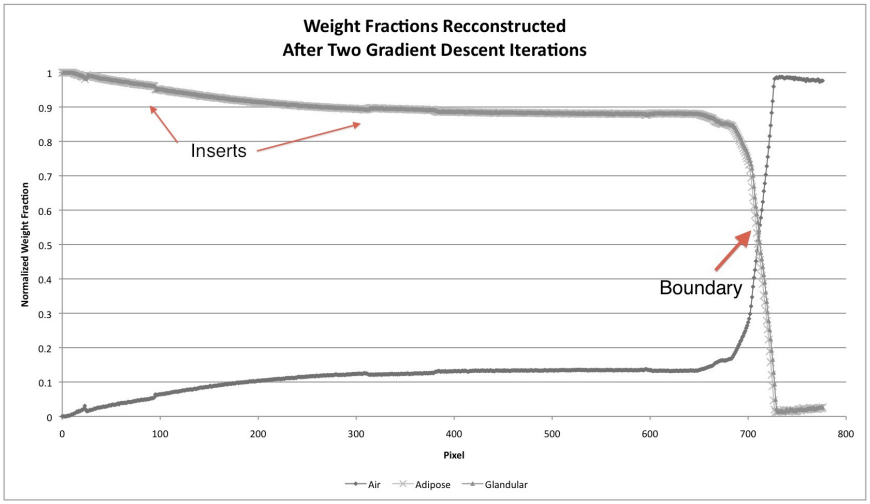
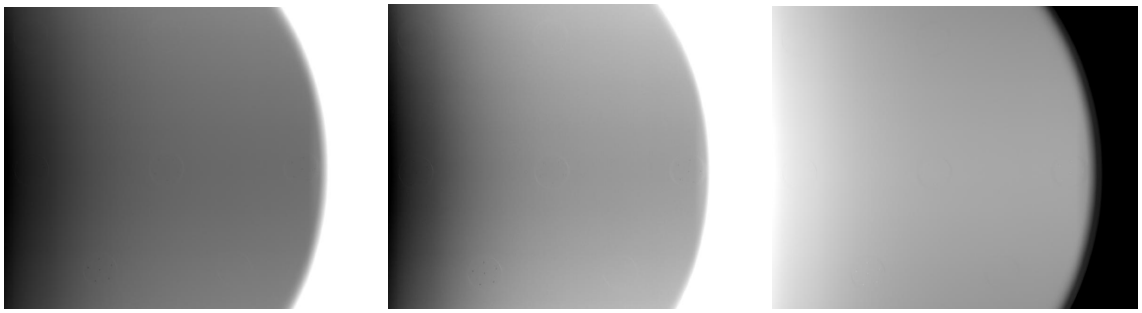


Figure 11.21: Vertical profile through the center of the weight fractions reconstructed using the full multi-material model from Figure 11.20. The values were normalized. The arrows show the increase of the glandular fractions corresponding to the glandular inserts and the boundary of the phantom when the voxels are mostly composed of air.

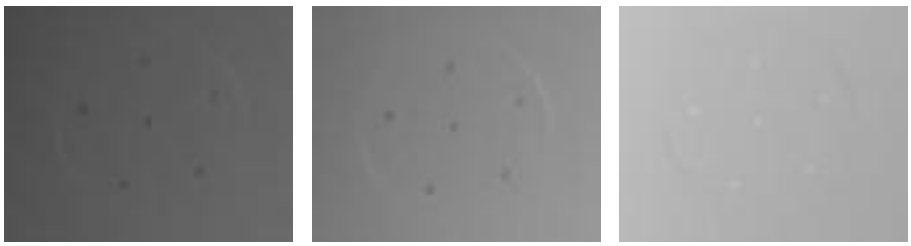


(a) Adipose weight fractions (b) Micro-calcifications weight fractions (c) Air weight fractions

Figure 11.22: Tomograph 25 of the reconstruction of the homogeneous-calcs phantom using the full multi-material model. The reconstruction was done using two gradient descent iterations

images through three glandular inserts in the center of the images. In this graph we can clearly see where the boundary of the phantom lies, as the weight fractions for air increase and the weight fractions of glandular tissue decrease. The problem is not fully converged with two gradient descent iterations but the results show that the method is making progress in the right direction. We leave an efficient implementation of the multi-material model as future work in this thesis.

The weight fractions corresponding to the middle tomograph for the second homogeneous phantom are shown in Figure 11.23. We assumed the presence of adipose tissue,



(a) Adipose (b) Micro-calcifications (c) Air

Figure 11.23: Close up of the micro-calcifications feature of Figure 11.23.

micro-calcifications, and air and used two gradient descent iterations to reconstruct the volume. Clearly, the method has not fully converged, but the results are promising and very similar to the previous phantom. As expected, the weight fractions for air and micro-calcifications barely reconstruct the details of the inserts in the middle plate of the phantom.

A close up of a calcification lesion is shown in Figure 11.23.

11.4 Summary

In this chapter we have shown through numerical results that the polyenergetic forward models can provide an accurate reconstruction of the breast. Our reconstructions show that the polyenergetic nature of the x-ray beam more accurately models the physics of the problem, while standard approaches such as filtered backprojection and MLEM assume a monoenergetic x-ray beam. Thus, our polyenergetic model should, and does, produce better reconstructions than standard algorithms. Moreover, we showed that iterative methods that take into account the various materials composing the object can be efficiently implemented in current computing architectures. The effectiveness of our polyenergetic reconstruction was illustrated with real data taken of an object with known materials that simulates an actual breast. This data was acquired using a digital tomosynthesis system, not simulated. This validates the work presented in this thesis as a reconstruction framework that can work in a clinical setting. In addition, we have shown using our optimized implementation described in Chapter 10, the time it takes for our reconstruction process is comparable to standard methods. Using a single GPU we are able to obtain close to 30$\times$ speed-up in our algorithms making a full reconstruction in less than five minutes.

Chapter 12

Conclusion

The research presented in this thesis shows that large scale, physically accurate medical image reconstructions can be done using smaller and less costly machines by effectively harnessing the multi-threading power of GPUs. This allows smaller hospitals and clinics that lack access to expensive clusters and supercomputers, to use off-the-shelf computing systems to run the reconstructions needed for diagnostic radiology.

In this work we have proposed four physically accurate forward models for the image acquisition process that explicitly take into account the polyenergetic nature of the x-ray beam as well as the attenuation of each material present inside the imaged breast at each discretized energy level. Using the standard mathematical model for transmission radiography, where we assume that x-ray transmission imaging is a realization of a Poisson random process, we employ a Maximum Likelihood Estimator approach (MLE) to formulate the reconstruction problem as a large scale ill-posed nonlinear inverse problem. We solve this nonlinear inverse problem in its numerical optimization formulation and describe a gradient descent and a Newton framework to reconstruct the volume. Gradient descent methods are slower to converge in terms of number of iterations, but are more robust, and require less storage and only the first derivative of the cost function, making them our method of choice. Newton methods provide faster convergence, but are characterized by costly inner iterations and require storage of the first derivative and the Hessian of the cost function, making them more expensive for our problem. In addition, our framework employs regularization by early termination in order to restrict the influence of noise in the

input data.

We provide an efficient implementation of the iterative reconstruction framework that takes advantage of the multi-threaded parallelism available in GPUs. The multi-threaded implementation is written in OpenCL making it scalable and portable to other architectures. The implementation is designed to increase throughput, minimize storage requirements, and maximize the utilization of hardware resources. The most computationally intense portions of the reconstruction process are the matrix-vector and matrix transpose-vector products of the raytrace matrix, A_θ, which are needed when evaluating the cost function and its corresponding gradient. Storing A_θ and computing these products explicitly is unrealistic in terms of memory, thus we compute the entries $a^{(ij)}$ as needed using our optimized raytracing and backprojection algorithms. To perform these products on the GPU, we drive the parallelism through the output data and exploit a 3D execution topology. Assigning work groups to sequential rows in the matrix takes advantage of thread warps and coalesced memory accessing, and reduces thread divergence because of the structure of A_θ.

We focus on performing the majority of our computation in the device, thus minimizing host to device communication. To do this we extend functionality and fuse subsequent kernels, avoiding the overhead cost of data transfer and enqueueing a new kernel. Explicit kernel fusion also allows us to reduce on chip memory needs, since by performing the complete function evaluation in the device we are able to avoid storage and communication of intermediate quantities. By extending kernel functionality and keeping in-process data on the chip, we keep multiprocessors working the majority of the time. The implementation we present in this thesis runs a full clinical size reconstruction in under five minutes, using a single core accelerated with a single GPU. The application framework can easily be extended to multiple GPUs, which may be needed in the case of several materials present inside the object or reconstructing a higher resolution volume like those needed in CT scans.

The contribution of this work to medical imaging and diagnostic radiology is a fast reconstruction of the imaged breast based on a physically accurate polyenergetic x-ray model. The reconstructed volume shows significantly less beam hardening artifacts and each voxel contains quantitative information of the object composition. Compared to the standard filtered backprojection or MLEM methods used by the tomosynthesis imaging

system, our polyenergetic x-ray reconstruction shows better image quality and less artifacts. A physically accurate model of the imaging system coupled with our efficient reconstruction framework allows for a precise characterization of the imaged breast, giving the physician better information to make a correct clinical diagnosis of the patient.

Bibliography

[1] Akcelik, V., Flath, H. P., Ghattas, O., Hill, J., Van Bloemen Waanders, B., and Wilcox, L. C. Fast algorithms for Bayesian uncertainty quantification in large-scale linear inverse problems based on low-rank partial Hessian approximations. `http://www.cs.sandia.gov/~bartv/papers/sisc_omar_flath_2010.pdf`, 2010.

[2] Alvarez, R. E., and Macovski, A. Energy-selective reconstructions in x-ray computerized tomography. *Phys. Med. Biol 21*, 5 (1976), 733–744.

[3] Alvarez, R. E., Seibert, J. A., and Thompson, S. K. Comparison of dual energy detector system performance. *Med. Phys. 31*, 3 (2004), 556–565.

[4] American Cancer Society. Breast cancer facts & figures 2011-2012. Atlanta: American Cancer Society, Inc.

[5] Bachmayr, M., and Burger, M. Iterative total variation schemes for nonlinear inverse problems. *Inverse Problems 25*, 10 (2009).

[6] Bardsley, J., and Vogel, C. R. A nonnegativly constrained convex programming method for image reconstruction. *SIAM J. Sci. Comput. 25*, 4 (2004), 1326–1343.

[7] Bardsley, J. M. An efficient computational method for total variation-penalized poisson likelihood estimation. *Inverse Problems and Imaging 2*, 2 (2008), 167–185.

[8] Bardsley, J. M. Stopping rules for a nonnegatively constrained iterative method for ill-posed poisson imaging problems. *BIT 48*, 4 (2008), 651–664.

[9] Bardsley, J. M. An efficient MCMC method for uncertainty quantification in inverse problems. Tech. Rep. UM Math Tech Report #28, Mathematics Department, University of Montana, 2010.

[10] Bardsley, J. M., Calvetti, D., and Somersalo, E. Hierarchical regularization for edge-preserving reconstruction of PET images. *Inverse Problems 26*, 3 (2010), doi: 10.1088/0266–5611/26/3/035010.

[11] Bazalova, M., Carrier, J., Beaulieu, L., and Verhaegen, F. Dual-energy CT-based material extraction for tissue segmentation in Monte Carlo dose calculations. *Phys. Med. Biol. 53*, 9 (2008), 2439–2456.

[12] Berger, M. J., Hubbell, J. H., Seltzer, S. M., Chang, J., Coursey, J. S., Sukumar, R., Zucker, D. S., and Olsen, K. *XCOM: Photon Cross Section Database*. National Institute of Standards and Technology, 2010. `http://www.nist.gov/pml/data/xcom/index.cfm`.

[13] Besag, J. On the statistical analysis of dirty pictures (with discussion). *J. Royal Stat. Soc., Ser. B 48*, 5–6 (1986), 259–302.

[14] Besag, J. Towards Bayesian image analysis. *J. Appl. Stat. 16* (1989), 395–406.

[15] Björck, Å. A bidiagonalization algorithm for solving large and sparse ill-posed systems of linear equations. *BIT 28*, 3 (1988), 659–670.

[16] Björck, Å., Grimme, E., and van Dooren, P. An implicit shift bidiagonalization algorithm for ill-posed systems of linear equations. *BIT 34*, 4 (1994), 510–534.

[17] Bleuet, P., Guillemaud, R., and Magnin, I. E. Resolution improvement in linear tomosynthesis with an adapted 3d regularization scheme. *Proc. SPIE 4682* (2002), 117–125.

[18] Brenner, D. J., and Hall, E. J. Computed tomography – An increasing source of radiation exposure. *New Engl. J. Med. 357*, 22 (2007), 2277–2284.

[19] Brooks, R., and Di Chiro, G. Beam hardening in x-ray reconstructive tomography. *Phys. Med. Biol. 21* (1976), 390–398.

[20] Bushberg, J. T., Seibert, J. A., Leidholdt, E. M., and Boone, J. M. *The Essential Physics of Medical Imaging*, 3 ed. Lippincott Williams & Wilkins, Philadelphia, PA, 2011.

[21] Calvetti, C., Hansen, P. C., and Reichel, L. L-curve curvature bounds via lanczos bidiagonalization. *Electron. Trans. Numer. Anal. 14* (2002), 20–35.

[22] Calvetti, D., and Somersalo, E. Image inpainting and bootstrap priors. *Image and Vision Computing 24* (2006), 782–793.

[23] Calvetti, D., and Somersalo, E. Gaussian hypermodel and recovery of blocky objects. *Inverse Problems 23* (2007), 733–754.

[24] Calvetti, D., and Somersalo, E. *Introduction to Bayesian Scientific Computing: Ten Lectures on Subjective Computing*, vol. 2. Springer Science+Business Media, LLC, New York, 2007.

[25] Calvetti, D., and Somersalo, E. Hypermodels in the Bayesian imaging framework. *Inverse Problems 24*, 3 (2008), doi: 10.1088/0266–5611/24/3/034013.

[26] Candés, E. J., Romberg, J. K., and Tao, T. Stable signal recovery from incomplete and inaccurate measuerments. *Comm. Pure Appl. Math. LIX* (2006), 1207–1223.

[27] Chan, T. F., and Shen, J. *Image Processing and Analysis: Variational, PDE, Wavelet, and Stochastic Methods*. SIAM, Philadelphia, PA, 2005.

[28] Chandra, R. *Nuclear medicine physics: the basics*, 5 ed. Williams & Wilkins, Baltimore, MD, 1998.

[29] Chen, P., and Barner, K. A multi-resolution statistical reconstruction for digital tomosynthesis. *Submitted to IEEE Trans. Med. Imaging* (2005).

[30] Chung, J. *Numerical Approaches for Large-Scale Ill-Posed Inverse Problems.* PhD thesis, Emory University, 2009. `http://www.cs.umd.edu/~jmchung/Home/CV_files/Julianne_Chung_Dissertation.pdf`.

[31] Chung, J., Knepper, S., and Nagy, J. G. Large-scale inverse problems in imaging. In *Handbook of Mathematical Methods in Imaging*, O. Scherzer, Ed. Springer, New York, 2010.

[32] Chung, J., Nagy, J., and Sechopoulos, I. Numerical algorithms for polyenergetic digital breast tomosynthesis reconstruction. *SIAM J. Imaging Sci. 3*, 1 (2010), 133–152.

[33] Cranley, K., Gilmore, B. J., Fogarty, G. W. A., and Desponds, L. *Catalogue of Diagnostic X-Ray Spectra and Other Data.* Institute of Physics and Engineering in Medicine, York, 1997.

[34] de Greef, M., Creeze, J., van Eijk, J. C., Pool, R., and Bel, A. Accelerated ray tracing for radiotherapy dose calculations on a GPU. *Med. Phys 36*, 9 (2009), 4095–4102.

[35] De Man, B., Nuyts, J., Dupont, P., Marchal, G., and Suetens, P. An iterative maximum-likelihood polychromatic algorithm for CT. *IEEE Trans. Med. Imaging 20* (2001), 999–1008.

[36] Delaney, A. H., and Bresler, Y. Globally convergent edge-preserving regularized reconstruction: An application to limited-angle tomography. *IEEE Trans. Image Process. 7*, 2 (1998), 204–221.

[37] Dennis, J. E., and Schnabel, R. B. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations.* SIAM, Philadelphia, 1996.

[38] Dobbins III, J. T. Tomosynthesis imaging: At a translational crossroads. *Med. Phys. 36*, 6 (2009), 1956–1967.

[39] Dobbins III, J. T., and Godfrey, D. J. Digital x-ray tomosynthesis: current state of the art and clinical potential. *Phys. Med. Biol 48* (2003), R65–R106.

[40] Elbakri, I. A., and Fessler, J. A. Statistical image reconstruction for polyenergetic x-ray computed tomography. *IEEE Trans. Med. Imaging 21*, 2 (2002), 89–99.

[41] Elbakri, I. A., and Fessler, J. A. Segmentation-free statistical image reconstruction for polyenergetic x-ray computed tomography with experimental validation. *Phys. Med. Biol. 48*, 15 (2003), 2453–2477.

[42] Elmore, J. G., Barton, M. B., Moceri, V. M., Polk, S., Arena, P. J., and Fletcher, S. W. Ten-year risk of false positive screening mammograms and clinical breast examinations. *New England Journal of Medicine 338*, 16 (1998), 1089–1096.

[43] Engl, H. W., Hanke, M., and Neubauer, A. *Regularization of Inverse Problems*. Kluwer Academic Publishers, Dordrecht, 2000.

[44] Engl, H. W., and Kügler, P. Nonlinear inverse problems: Theoretical aspects and some industrial applications. In *Multidisciplinary Methods for Analysis Optimization and Control of Complex Systems*, V. Capasso and J. Périaux, Eds. Springer, Berlin, 2005, pp. 3–48.

[45] Engl, H. W., Kunisch, K., and Neubauer, A. Convergence rates for Tikhonov regularisation of nonlinear ill-posed problems. *Inverse Problems 5*, 4 (1999), 523–540.

[46] Epstein, C. L. *Introduction to the Mathematics of Medical Imaging, Second Edition*. SIAM, Philadelphia, PA, 2007.

[47] Eriksson, J., and Wedin, P. Truncated Gauss-Newton algorithms for ill-conditioned nonlinear least squares problems. *Optimization Methods and Software 19*, 6 (2004), 721–737.

[48] Fessler, J. A. Statistical image reconstruction methods for transmission tomography. In *Handbook of Medical Imaging, Medical Image Processing and Analysis*, M. Sonka and J. M. Fitzpatrick, Eds., vol. 2. SPIE, Bellingham, WA, 2000.

[49] Figueiredo, M. A. T., Nowak, R. D., and Wright, S. J. Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems. *IEEE J. Selected Topics in Signal Processing 1*, 4 (2007), 586–597.

[50] Fredenberg, E., Lundqvist, M., Cederström, B., Aslund, M., and Danielsson, M. Energy resolution of a photon-counting silicon strip detector. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment 613*, 1 (2010), 156–162.

[51] Gaster, B., Howes, L., Kaeli, D. R., Mistry, P., and Schaa, D. *Heterogeneous computing with OpenCL*. Morgan Kaufmann, 2011.

[52] Godfrey, D. J., McAdams, H. P., and Dobbins III, J. T. Optimization of the matrix inversion tomosynthesis (MITS) impulse response and modulation transfer function characteristics for chest imaging. *Med. Phys. 33* (2006), 655–667.

[53] Golub, G., Heath, M., and Whaba, G. Generalized cross-validation as a method for choosing a good ridge paremeter. *Technometrics 21* (1979), 215–223.

[54] Golub, G., and Pereyra, V. The differentiation of pseudo-inverses and nonlinear least squares whose variables separate. *SIAM J. Numer. Anal. 10* (1973), 413–432.

[55] Golub, G., and Pereyra, V. Separable nonlinear least squares: the variable projection method and its applications. *Inverse Problems 19* (2003), R1–R26.

[56] Gordon, R., Bender, R., and Herman, G. T. Algebraic reconstruction techniques (ART) for three-dimensional electron microscopy and x-ray photography. *Journal of Theoretical Biology 29*, 3 (1970), 471 – 481.

[57] Grama, A., Karypis, G., Kumar, V., and Gupta, A. *Introduction to Parallel Computing (2nd Edition)*, 2 ed. Addison Wesley, 2003.

[58] Habin, K. What's the status of FDA approval for tomosynthesis? *ABC News OnCall Breast Cancer Center*, `http://abcnews.go.com/Health/OnCallPlus/`.

[59] Hadamard, J. Sur les problèmes aux dérivées partielles et leur signification physique. *Princeton University Bulletin 13* (1902), 49–52.

[60] Hammerstein, G. R., Miller, D. W., White, D. R., Masterson, M. E., Woodard, H. Q., and Laughlin, J. S. Absorbed radiation dose in mammography. *Radiology 130*, 2 (1979), 485–491.

[61] Hanke, M. On lanczos based methods for the regularization of discrete ill-posed problems. *BIT 41*, 5 (2001), 1008–1018.

[62] Hansen, P. C. *Rank-deficient and discrete ill-posed problems.* SIAM, Philadelphia, PA, 1997.

[63] Hansen, P. C. *Discrete Inverse Problems: Insight and Algorithms.* SIAM, Philadelphia, PA, 2010.

[64] Hansen, P. C., Nagy, J. G., and O'Leary, D. P. *Deblurring Images: Matrices, Spectra and Filtering.* SIAM, Philadelphia, PA, 2006.

[65] Heismann, B., and Balda, M. Quantitative image-based spectral reconstruction for computed tomography. *Med. Phys. 36* (2009), 4471–4485.

[66] Herman, G. T. *Fundamentals of Computerized Tomography: Image Reconstruction from Projections.* Advances in pattern recognition. Springer, 2009.

[67] Hofmann, B. Regularization of nonlinear problems and the degree of ill-posedness. In *Inverse Problems: Principles and Applications in Geophysics, Technology, and Medicine*, G. Anger, R. Gorenflo, H. Jochmann, H. Moritz, and W. Webers, Eds. Akademie Verlag, Berlin, 1993.

[68] Hubbell, J. H., and Seltzer, S. M. Tables of X-ray mass attenuation coefficients and mass energy-absorption coefficients from 1keV to 20 MeV for elements Z=1 to 92 and 48 additional substances of dosimetric interest. `http://physics.nist.gov/PhysRefData/XrayMassCoef/cover.html`.

[69] Humphrey, L. L., Helfand, M., Chan, B. K., and Woolf, S. H. Breast cancer screening: A summary of the evidence for the US preventive services task force. *Annals of Internal Medicine 137*, 5_Part_1 (2002), 347–360.

[70] Iniewski, K., Ed. *Medical Imaging: Principles, Detectors, and Electronics.* Wiley, Hoboken, NJ, 2009.

[71] Jain, A. *Fundamentals of digital image processing.* Prentice-Hall information and system sciences series. Prentice Hall, 1989.

[72] Joseph, P. M., and Spital, R. D. A method for correcting bone induced artifacts in computed tomography scanners. *J. of Comput. Assist. Tomog. 2*, 1 (1978), 100–108.

[73] Kastanis, I., Arridge, S., Stewart, A., Gunn, S., Ullberg, C., and Francke, T. 3D digital breast tomosynthesis using total variation regularization. In *Lecture Notes in Computer Science*, vol. 5116. Springer, 2008.

[74] Kaufman, L. A variable projection method for solving separable nonlinear least squares problems. *BIT 15* (1975), 49–57.

[75] Kelley, C. T. *Iterative methods for optimization*, vol. 18. SIAM, 1987.

[76] Kilmer, M. E., Hansen, P. C., and Español, M. I. A projection-based approach to general-form Tikhonov regularization. *SIAM J. Sci. Comput. 29*, 1 (2007), 315–330.

[77] Kilmer, M. E., and O'Leary, D. P. Choosing regularization parameters in iterative methods for ill-posed problems. *SIAM J. Matrix Anal. Appl. 22* (2001), 1204–1221.

[78] Kinahan, P. E., Alessio, A. M., and Fessler, J. A. Dual energy CT attenuation correction methods for quantitative assessment of response to cancer therapy with PET/CT imaging. *Technology in Cancer Research and Treatment 5*, 4 (2006), 319–327.

[79] Kirk, D. B., and Wen-mei, W. H. *Programming massively parallel processors: a hands-on approach.* Morgan Kaufmann, 2010.

[80] Kolb, T. M., Lichy, J., and Newhouse, J. H. Comparison of the performance of screening mammography, physical examination, and breast US and evaluation of

factors that influence them: An analysis of 27,825 patient evaluations1. *Radiology 225*, 1 (2002), 165–175.

[81] Landweber, L. An iteration formula for Fredholm integral equations of the first kind. *Amer. J. of Math. 73*, 3 (1951), 615–624.

[82] Larsen, R. M. Lanczos bidiagonalization with partial reorthogonalization. *DAIMI PB 27*, 537 (1998).

[83] Macovski, A., Alvarez, R. E., Chan, J. L.-H., Stonestrom, J. P., and Zatz, L. M. Energy dependent reconstruction in x-ray computerized tomography. *Comp. Biol. Med. 6*, 4 (1976), 325–336.

[84] MATLAB. *Student version 7.10.0 (R2010a)*. The MathWorks Inc., Natick, Massachusetts, 2010.

[85] Mayo Clinic Staff. Breast calcifications. `http://www.mayoclinic.com/health/breast-calcifications/MY00101`, August 2010.

[86] Miller, K. Least squares methods for ill-posed problems with a prescribed bound. *SIAM J. Math Anal 1*, 1 (1970), 52–74.

[87] Natterer, F. *The Mathematics of Computerized Tomography*. Classics in Applied Mathematics. SIAM, 2001.

[88] Nickolls, J., Buck, I., Garland, M., and Skadron, K. Scalable parallel programming with cuda. *Queue 6*, 2 (2008), 40–53.

[89] Nobelprize.org. Physiology or medicine 1979 - press release. `http://www.nobelprize.org/nobel_prizes/medicine/laureates/1979/press.html`.

[90] Nocedal, J., and Wright, S. *Numerical Optimization*. Springer, New York, 1999.

[91] Noh, J., Fessler, J. A., and Kinahan, P. E. Statistical sinogram restoration in dual-energy CT for PET attenuation correction. *IEEE Trans. Med. Imaging 28*, 11 (2009), 1688–1702.

[92] NVIDIA. *Fermi Compute Architecture Whitepaper.* `http://www.nvidia.com/object/workstation-solutions-tesla.html`.

[93] NVIDIA. What is GPU computing? `http://www.nvidia.com/object/what-is-gpu-computing.html`.

[94] O'LEARY, D. P., AND SIMMONS, J. A. A bidiagonalization-regularization procedure for large scale discretizations of ill-posed problems. *SIAM J. Sci. Stat. Comp. 2* (1981), 474–489.

[95] OSBORNE, M. Separable least squares, variable projection, and the gauss-newton algorithm. *Electronic Transactions on Numerical Analysis 28*, 2 (2007), 1–15.

[96] PAIGE, C. C., AND SAUNDERS, M. A. LSQR: An algorithm for sparse linear equations and sparse least squares. *ACM Trans. Math. Software 8*, 1 (1982), 43–71.

[97] PAIGE, C. C., AND SAUNDERS, M. A. LSQR: An algorithm for sparse linear equations and sparse least squares. *ACM Trans. Math. Software 8*, 2 (1982), 195–209.

[98] PARK, J. M., FANKEN, E. A., GARG, M., FAJARDO, L. L., AND NIKLASON, L. T. Breast tomosynthesis: Present considerations and future applications. *RadioGraphics 27* (2007), S231–S240.

[99] PHILLIPS, D. L. A technique for the numerical solution of certain integral equations of the first kind. *J. Assoc Comput Mach 9*, 1 (1962), 84–97.

[100] RATHEE, S., KOLES, Z. J., AND OVERTON, T. R. Image restoration in computed tomography: Estimation of the spatially variant point spread function. *IEEE Trans. Med. Imag. 11*, 4 (1992), 539–545.

[101] RUDIN, L., OSHER, S., AND FATEMI, E. Nonlinear total variation based noise removal algorithms. *Physica D 60* (1992), 259–268.

[102] RUHE, A., AND WEDIN, P. Algorithms for separable nonlinear least squares problems. *SIAM Review 22*, 3 (1980), 318–337.

[103] Sechopoulos, I. *Investigation of physical processes in digital x-ray tomosynthesis imaging of the breast.* PhD thesis, Georgia Institute of Technology, 2007.

[104] Sechopoulos, I., Suryanarayanan, S., Vedantham, S., D'Orsi, C. J., and Karellas, A. Scatter radiation in digital tomosynthesis of the breast. *Med. Phys 34* (2007), 564–576.

[105] Siddon, R. Fast calculation of the exact radiological path for a three dimensional CT array. *Med. Phys. 12*, 2 (1985), 252–255.

[106] Sidky, E. Y., Kao, C., and Pan, X. Accurate image reconstruction from few-views and limited-angle data in divergent-beam CT. *Journal of X-Ray Science and Technology 14*, 2 (2006), 119–139.

[107] Sidky, E. Y., Yu, L., Pan, X., Zou, Y., and Vannier, M. A robust method of x-ray source spectrum estimation from transmission measurements: Demonstrated on computer simulated, scatter-free transmission data. *J. Applied Physics 97* (2005), 124701–1:124701–11.

[108] Stenner, P., Berkus, T., and Kachelriess, M. Emperical dual energy calibration (EDEC) for cone-beam computed tomography. *Med. Phys. 34* (2007), 3630–3641.

[109] Sukovic, P., and Clinthorne, N. H. Penalized weighted least-squares image reconstruction for dual energy x-ray transmission tomography. *IEEE Trans. Med. Imaging 19*, 11 (2000), 1075–1081.

[110] The Khronos Group. OpenCL (Open Computing Language) version 1.1. `http://www.khronos.org/opencl/`.

[111] Tikhonov, A. N. Regularization of incorrectly posed problems. *Sov Math Dockl 4* (1963), 1624–1627.

[112] Tikhonov, A. N. Solution of incorrectly formulated problems and the regularization method. *Sov Math Dockl 4* (1963), 1035–1038.

[113] Tikhonov, A. N., and Arsenin, A. N. *Solutions of ill-posed problems.* Winston & Sons, Washington, 1977.

[114] Tikhonov, A. N., Leonov, A. S., and Yagola, A. G. *Nonlinear Ill-Posed Problems, Volumes I and II.* Chapman and Hall, London, 1998.

[115] Tsaig, Y., and Donoho, D. L. Extensions of compressed sensing. *Signal Processing 86*, 3 (2006), 549–571.

[116] Vogel, C. R. *Computational Methods for Inverse Problems.* SIAM, Philadelphia, PA, 2002.

[117] Wen-mei, W. H. *GPU Computing Gems Emerald Edition.* Morgan Kaufmann, 2011.

[118] Ying, Z., Naidu, R., and Crawford, C. R. Dual energy computed tomography for explosive detection. *J. X-Ray Sci. Tech. 14* (2006), 235–256.

[119] Zhang, Y., Chan, H.-P., Sahiner, B., Wei, J., Goodsitt, M. M., Hadjiiski, L. M., Ge, J., and Zhou, C. A comparative study of limited-angle cone-beam reconstruction methods for breast tomosynthesis. *Med. Phys. 33* (2006), 3781–3795.