M.Sc. Thesis
Master of Science in Engineering

DTU Compute
Department of Applied Mathematics and Computer Science

ABBA Iterative Methods for X-Ray Computed Tomography

Maria Knudsen (s171246)
Supervisors: Per Christian Hansen &
Emil Y. Sidky

Kongens Lyngby 2023

DTU Compute
Department of Applied Mathematics and Computer Science
Technical University of Denmark

Matematiktorvet
Building 303B
2800 Kongens Lyngby, Denmark
Phone +45 4525 3031
compute@compute.dtu.dk
www.compute.dtu.dk

Abstract

ABBA Iterative Methods for X-Ray Computed Tomography

X-ray computed tomography (CT) is used in many branches to examine the interior of an object. Solving CT problems involve projector matrices that are large, and even though they are sparse, they still require a lot of memory. To reduce memory, matrix-free implementations of the projector matrices are beneficial but it leads to unmatched projector pairs as the back projector **B** is not exactly the transpose of the forward projector **A**. The squared matrices **AB** and **BA** are not guaranteed to be symmetric and positive definite and, thus, standard iterative methods for CT problems such as CGLS are not guaranteed to converge. This thesis studies the behavior of GMRES when providing an unmatched projector pair, referred to as the ABBA iterative methods. To create the unmatched projector pair, we examine the two public software packages ASTRA and TIGRE, and one software package provided by Professor Emil Y. Sidky from University of Chicago.

Solving CT problems with GMRES can be very expensive in memory as increasing the number of iterations yields an increasing Krylov subspace. Thus, we will explore the influence of using restarted GMRES. Furthermore, we investigate different stopping criteria. We explore the Discrepancy Principle (DP) and the Normalized Cumulative Periodogram (NCP).

The results of the exploratory study showed that the ABBA iterative methods obtain semi-convergence for unmatched projector pairs both with and without restart. Using restarted GMRES implies a slower convergence making the solvers less sensitive to taking too many iterations. From the investigation of DP and NCP, we found that both stop too early. Hence, we did not find a good stopping criterion for the ABBA methods. Finally, based on our study, we can conclude that the ABBA methods are a good choice for CT problems with unmatched projector pairs.

The research code presented in this thesis has been published in a GitHub repository called ABBA-GMRES [23].

KEYWORDS: ABBA iterative methods, AB-GMRES, BA-GMRES, restarted GMRES, Krylov subspace method, unmatched projector pair.

Preface

This Master thesis was prepared at the Department of Applied Mathematics and Computer Science at the Technical University of Denmark in fulfillment of the requirements for acquiring a Master's degree in Mathematical Modelling and Computation. The workload of this Master thesis corresponds to 35 ECTS points.

I want to take a moment to thank those who have been an important part of my thesis. First of all, I want to thank my main supervisor Professor Per Christian Hansen. I am very grateful to have had you as my supervisor. Thanks for sharing your knowledge, giving me good feedback, and acknowledging and sharing my frustrations but most importantly, thanks for your support in difficult times.

Thanks to my co-supervisor Professor Emil Y. Sidky for your interest in this project, for providing the user domain for the thesis, and for your feedback. I would also like to thank Senior Researcher Jakob Sauer Jørgensen from DTU for providing knowledge and expertise. Moreover, I would like to thank him for the CGLS results that demonstrate why we even consider the ABBA methods.

Furthermore, I would like to thank Dr. Ander Biguri from University of Cambridge and Dr. Willem Jan Palenstijn from CWI in Amsterdam for good discussions and quick answers. Lastly, I want to thank my boyfriend Martin Haulund Gæde for the mental support and for taking the time to discuss theoretical and programming/computer-based questions. Moreover, I would like to thank him for the good and constructive feedback.

Kongens Lyngby, February 28, 2023

Maria Knudsen (s171246)

Symbols

General Notation

Type	**Space**	**Typeset**	Example
Scalar	$\mathbb{R}$	Minuscule	k
Vector	$\mathbb{R}^n$	Roman bold and minuscule	$\mathbf{x}$
Matrix	$\mathbb{R}^{m \times n}$	Roman bold and capital	$\mathbf{A}$

Operators

Operator	**Name**
$\|\cdot\|_2$	2-norm
$\|\cdot\|_F$	Frobenius-norm
$\mathcal{E}(\cdot)$	Expected value
$\mathcal{R}$	Radon transform
$\mathcal{R}^\#$	Back projection

Acronyms

Acronym	**Meaning**
BP	Back Projection
CGLS	Conjugate Gradient algorithm for Least Squares problems
CT	Computed Tomography
DFT	Discrete Fourier Transform
DP	Discrepancy Principle

FBP	Filtered Back Projection
FDK	Feldkamp Davis and Kress
FFT	Fast Fourier Transform
FP	Forward Projection
GMRES	Generalized Minimal Residuals
LSQR	Alternative implementation of CGLS
NCP	Normalized Cumulative Periodogram
SVD	Singular Values Decomposition
TSVD	Truncated Singular Value Decomposition
MV	Matrix-Vector

List of Symbols

Symbol	Description
$\mathbf{A}$	The forward projector or system matrix
$a_{kl}^{(j)}$	Line-pixel coefficient or weights
$\alpha_{j,k}$	Linear interpolation coefficient
α	Parameter that ensure non negative real part eigenvalues in Landweber
$\mathbf{B}$	Back projector
$\bar{\mathbf{b}}$	Measured data (or sinogram) without noise
$\mathbf{b}$	Measured data (or sinogram) with noise
b_l	The lth detector element
$\mathbf{c}$	A vector with the coefficients for each vector in the Krylov subspace $\mathbf{W}_k$
χ_{π_j}	Pixel indicator function
$\mathbf{d}$	An arbitrary vector containing measured data
$\mathrm{d}I$	Change in intensity
$\mathrm{d}l$	Thickness of object
$\mathbf{e}$	Noise
$\mathbf{e}_1$	A zero vector with one in its first element
e_{dp}	Relative error for $\mathbf{x}_{\mathrm{dp}}$
e_{ncp}	Relative error for $\mathbf{x}_{\mathrm{ncp}}$
e_{opt}	Relative error for $\mathbf{x}_{\mathrm{opt}}$
η	Noise level (standard deviation)

f	Image function consisting of intensity values
$f_\square$	Pixelated image function
f_{interp}	Linear interpolated value from two neighbouring pixel intensities
$\mathbf{f}$	Discrete image of f
g	Function of the measured data - the sinogram
γ	Length of a pixel
$\mathbf{H}$	Hessenberg matrix
$\tilde{\mathbf{H}}$	A partial part of the Hessenberg matrix
i	Iteration
i_{dp}	The number of iterations found with DP
i_{ncp}	The number of iterations found with NCP
i_{opt}	The optimal number of iterations
$\mathbf{I}$	Identity matrix
I	Intensity
I_0	Initial intensity
$I_{\theta,s}$	Intensity at (θ, s)
$\mathcal{K}_k$	Krylov subspace
K	Maximum number of iterations
l	Position on the line parametrization of the X-ray
L	Total number of periods in restarted GMRES
$L_{\theta,s}$	Line segment
$L_{\theta,s}^{(\pi_j)}$	Intersection length
m	The number $N_\theta N_s$
$\mathbf{M}$	An arbitrary square system matrix
μ	Linear attenuation coefficient
n	Number of pixels
N_s	Number of detectors
N_θ	Number of measured angles
ν_{dp}	Safety factor in DP
ω	Step length in Landweber
p	Restart parameter
$\mathbf{p}_i$	Power spectrum of residual vector
π	A pixel in $f_\square$
$\hat{\pi}_j$	An artificial pixel in $f_\square$
$\phi_i^{[k]}$	Filter factors
$\mathbf{Q}_k$	An orthogonal matrix computed from QR factorization at iteration k

$\mathbf{R}$	An upper triangular matrix computed from QR factorization
$R_{e,\mathrm{dp}}$	The error ratio for DP
$R_{e,\mathrm{ncp}}$	The error ratio for NCP
$R_{i,\mathrm{dp}}$	The iteration ratio for DP
$R_{i,\mathrm{ncp}}$	The iteration ratio for NCP
$\mathbf{r}_0$	Initial residual
$\mathbf{r}_k$	Residual at iteration k
$\hat{\mathbf{r}}_i$	DFT of residual vector at iteration i
s	The axis spanned by the projection angle
σ	Standard deviation
σ_i	The ith singular value
$\mathbf{\Sigma}$	Diagonal matrix with singular values
θ	Projection angle
$\mathbf{U}$	Matrix of left singular vectors
$\mathbf{V}$	Matrix of right singular vectors
$\mathcal{W}$	A suitable basis for a low-dimensional subspace
$\mathbf{W}_k$	Orthogonalized Krylov subspace at iteration k
$\bar{\mathbf{x}}$	Ground truth
$\mathbf{x}$	Approximated solution to $\bar{\mathbf{x}}$
$\mathbf{x}_0$	Initial guess of solution
$\mathbf{x}_k$	Reconstruction at iteration k
$\mathbf{x}_{\mathrm{dp}}$	DP reconstruction at i_{dp}
$\mathbf{x}_{\mathrm{fbp}}$	FBP reconstruction
$\mathbf{x}_{\mathrm{naive}}$	Naive solution
$\mathbf{x}_{\mathrm{ncp}}$	NCP reconstruction at i_{ncp}
$\mathbf{x}_{\mathrm{opt}}$	Reconstruction at the optimum
$\mathbf{y}_k$	Solution at iteration k to GMRES minimization problem

Contents

CHAPTER 1

Introduction

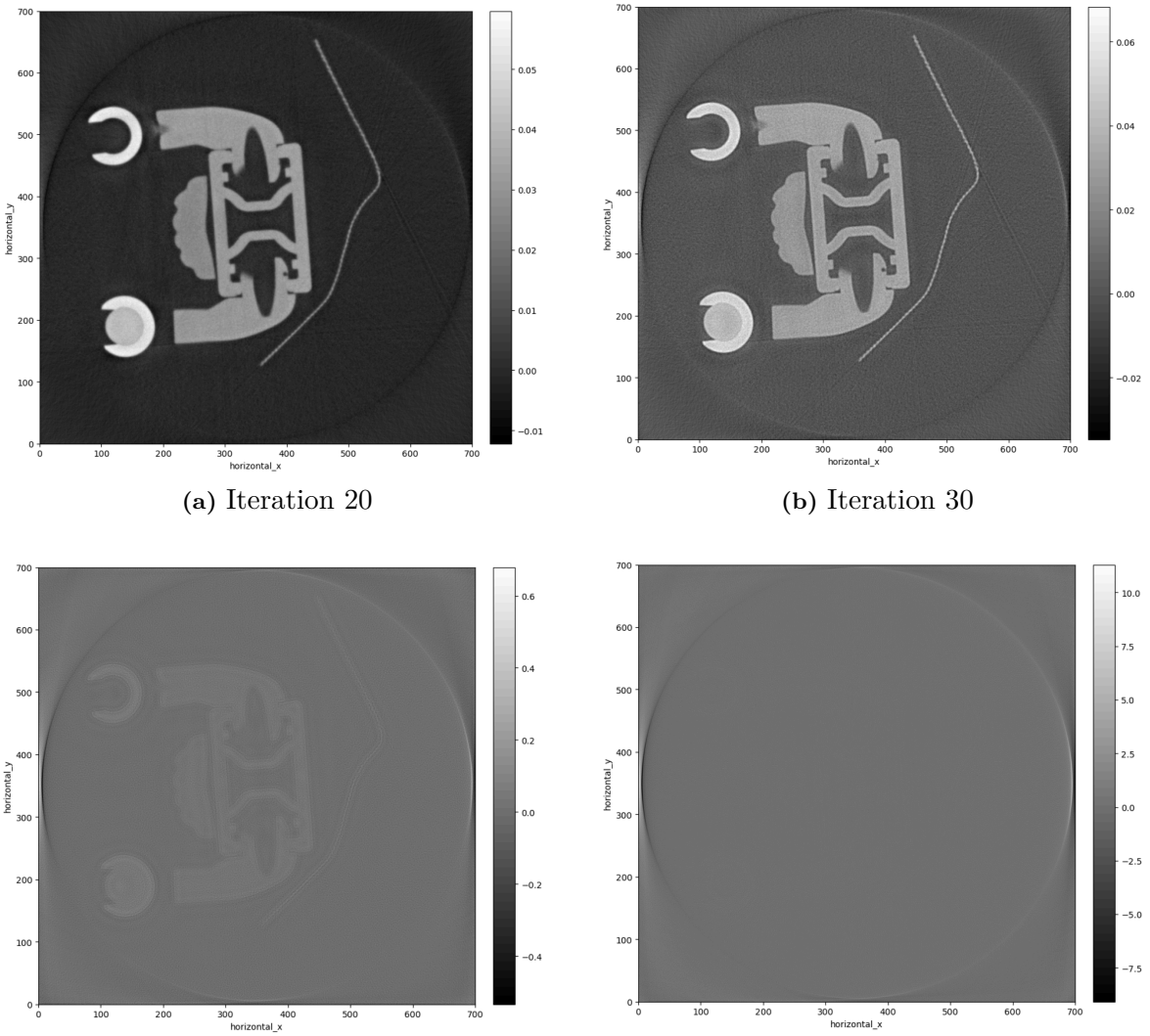
X-ray computed tomography (CT) is a widely used technique in medicine, materials science, industrial inspections, etc. CT contributes to a non-intrusive understanding of the interior of an object, as we can compute an image of the interior of an object from penetrating X-rays. CT works by; an X-ray source is placed on one side of the object after which the X-rays penetrate the object and are received in a detector. It then uses the measured attenuation of X-rays within the detector to reconstruct an image of the interior by solving a system of linear equations. These systems of equations can be very large and have millions of unknowns. Such large systems can be challenging to handle both because it is time-consuming but mostly because it requires a huge amount of memory. Thus, we need to find an efficient way for solving these large-scale computational problems.

CT problems, as well as many other problems that involve large systems of linear equations, can be solved by *iterative methods*. They are very efficient methods as they can solve systems only using matrix-vector products. Thereby, facilitating the possibility of *matrix-free* methods. In this thesis, we focus on *Krylov subspace methods* which is a class of iterative methods. We solve a least squares problem by solving the normal equations which require products with a matrix $\mathbf{A}$ referred to as the *forward projector* and its transpose $\mathbf{B}$ referred to as the *back projector*. We call this a *matched projector pair* and the forward projector models the projections of the X-rays onto the detector and the back projector models the reversed process.

To obtain better performance, we will consider matrix-free implementations of the projector pair. Hence, we will use different discretization schemes for the forward and back projections. We will consider the three discretization schemes for the forward projector: the *line model*, the *strip model*, and *Joseph's model*. Moreover, we will introduce the most common back projector within the CT community which is sometimes referred to as the *pixel-driven back projector*. Using matrix-free implementations of the projection models, thus, entails that we no longer use the exact transpose but have an *unmatched projector pair* instead. We, thereby, need to solve the unmatched normal equations with the squared matrices $\mathbf{AB}$ or $\mathbf{BA}$ which are not guaranteed to be symmetric and positive definite.

A common Krylov subspace method for CT problems is the *Conjugate Gradient Least*

Squares (CGLS). CGLS is only guaranteed to converge for matched projector pairs as it requires the matrices **AB** or **BA** to be symmetric and positive definite. Hence, CGLS is not guaranteed to converge for unmatched projector pairs. Senior Researcher Jakob Sauer Jørgensen from DTU Compute has provided a very good example of CGLS applied to the unmatched normal equations which are shown in Figure 1.1. The data is a CT scan of a Lego Gandalf figure. A slice of Gandalf's interior has been extracted with the *Core Imaging Library* (CIL) [21, 25] to create a 2D CT problem. Figure 1.1 shows the reconstructions of Gandalf's interior when having an unmatched projector pair and using CGLS. The results show that CGLS does not converge. Due to problems with the usual methods for solving CT problems, we are looking at the Krylov subspace method *GMRES* instead. More specifically, we will consider the *AB-GMRES* and *BA-GMRES* methods which we refer to as the *ABBA iterative methods*.



(a) Iteration 20 **(b)** Iteration 30

(c) Iteration 50 **(d)** Iteration 80

Figure 1.1: The images are from Senior Researcher Jakob Sauer Jørgensen from DTU Compute. Each image illustrates the reconstruction computed with CGLS at the i'th iteration.

The thesis is characterized by a lack of theory on the subject meaning we will focus on making numerical experiments for studying the ABBA iterative methods. We aim to determine the best possible implementation of the ABBA iterative methods based on experiments with among others the software packages ASTRA and TIGRE. We focus on implementing fast, robust, and memory-reduced solvers. The ABBA iterative methods can have a high memory complexity when the number of iterations increases and, thus, we will introduce restarted AB- and BA-GMRES and investigate their behavior and potential benefits.

Measurements from CT scans will include noise. This means that the ABBA methods will achieve *semi-convergence* and it is, therefore, crucial to have a good stopping criterion to obtain good reconstructions of the interior of an object. We will investigate two stopping criteria: the *Discrepancy Principle* (DP) and the *Normalized Cumulative Periodogram* (NCP).

In the end, we will use our final implementation of the ABBA iterative methods on real data. We will compare the reconstructions from the ABBA methods with a *filtered back projection* (FBP) reconstruction. Moreover, we will investigate the convergence behavior of the ABBA methods when using the Lego Gandalf data shown in Figure 1.1.

In short, this thesis answers the following questions:

- How do the ABBA iterative methods compare to existing methods?
- How are the ABBA iterative methods best implemented such that they utilize the fast operations in CT software packages?
- When considering different software packages, is there a difference in the forward and back projections and in how unmatched the projector pairs are?
- Do we see similar or different behavior of the ABBA iterative methods when using different software packages?
- How does the convergence behave when applying restart to AB- and BA-GMRES?
- What is the best strategy for choosing the restart parameter?
- What are the memory requirements, computational time, and robustness of the ABBA iterative methods?
- How robust are the different stopping criteria? Do they also work when we use restarted GMRES?

The research code presented in this thesis has been published in a GitHub repository called ABBA-GMRES [23] and is discussed in Appendix C.

1.1 Outline of Report

Chapter 2 will introduce *inverse problems* and introduce X-ray CT as an inverse problem. In Section 2.1, we introduce CT, both the laboratory setup needed to obtain the measurements and a bit of physical understanding of the processes within the data collection. We will then move on to the mathematical model of the CT inverse problem in Section 2.2. We will introduce the continuous CT model and then focus on discretizing it, to allow the usage of numerical methods. Thus, in Section 2.3, we discretize the inverse problem. Finally, in Section 2.4, we will discuss the use of *Singular Value Decomposition* (SVD) for analyzing our CT inverse problem.

Chapter 3 will address the challenges in solving CT problems. We will discuss possible methods for solving our inverse problem. Section 3.1 will briefly give the motivation for choosing *iterative solvers* for our reconstruction. In Section 3.2, we will further shrink the number of possible iterative solvers to Krylov subspace methods due to the desire for fast computations which results in unmatched projector pairs. We will then in Section 3.3 describe what a Krylov subspace is and why we end up choosing to look at GMRES. Section 3.4 will describe the GMRES algorithm and we will present a pseudo code for the implementation. GMRES requires a square matrix and since CT problems are often rectangular matrices, we need to introduce the AB/BA-GMRES methods which are described in Section 3.5. The important property of GMRES and AB/BA-GMRES is that they semi-convergence when noise is present in the measurements which we will discuss in Section 3.6.

Chapter 4 will investigate two public domain and one user domain software package to construct the unmatched forward and back projectors. Section 4.2 will introduce the ASTRA toolbox and Section 4.3 will introduce the TIGRE toolbox. The two public toolboxes ASTRA and TIGRE are compared in Section 4.4. Professor Emil Y. Sidky from University of Chicago has provided a user domain which we review in Section 4.5. Based on the analysis of the software packages, we decide which unmatched projector pairs we continue working with, and in Section 4.6 we determine how unmatched the forward and back projector pairs are. To reduce the scope of this chapter, we only include results for BA-GMRES while the results for AB-GMRES can be found in Appendix B.

Chapter 5 investigates the influence on the convergence history when applying restart and we will consider DP and NCP as stopping criteria. Section 5.1 show the results when applying restart to the ABBA methods and Section 5.2 show the results when applying the stopping criteria. The results for AB- and BA-GMRES are very similar and, therefore, we have only included the BA-GMRES results in the main text. The remaining results are shown in Appendix B.

Chapter 6 tests the ABBA methods on real data. Section 6.1 introduces a data set that includes a FBP reconstruction which makes it possible to compare the solution obtained

with the ABBA methods. Previously, we discussed a CT problem where CGLS failed to converge (see Figure 1.1) and in Section 6.2 the ABBA methods are used to solve the Gandalf problem but with a different unmatched projector pair.

Finally, the thesis ends with a discussion in Chapter 7 and a conclusion in Chapter 8 addressing the results obtained from the exploratory study of the ABBA iterative methods.

CHAPTER 2

Inverse Problems

The thesis aims to investigate the *ABBA iterative methods* for *X-ray computed tomography* (CT) problems. To fully understand the ABBA methods we must introduce the essential physical and mathematical models on which our analysis will be based.

This chapter is based on [17] and it will introduce CT which is a process that aims to construct an image of the interior of an object from exterior measurements. This is achieved by sending X-rays through the object of interest and then measuring the attenuation of the rays. An important property of X-rays is that materials have different degrees of attenuation of X-rays. Thus, by passing X-rays through an object, we will expect different attenuation based on which materials the X-rays passed through. We utilize this property in CT to reconstruct the interior of the object.

Mathematically, the fundamental process of CT is to solve an *inverse problem*. An inverse problem arises when we aim to recover some internal or hidden data from some external measurements. The process of obtaining these external measurements (output) is called the *forward problem* and it is shown in Figure 2.1. Thus, the inverse problem is to reconstruct the input from the output and the system. The system is a mathematical model of physics describing the relationship between the input and the output.

Figure 2.1: The forward problem is to compute the output from the input and the system. The inverse problem is to compute the input from the system and the output.

CT aims to reconstruct the interior of an object from our measurements of attenuated X-rays. The input is, therefore, the interior of the object, the output is the measured dampened X-rays, and the system is the physics describing this relationship. We will cover the fundamental physics of CT and how to mathematically model the inverse problem of CT.

The structure of the chapter follows:

- Introduction to CT scanning and the basic physical models.

- CT as an inverse problem.
- Analysis of the discretized CT problem.

2.1 Introduction to X-Ray Computed Tomography

X-ray computed tomography (CT) is an important tool in fields such as medicine and material science. CT is a non-intrusive way of gaining information about the interior of an object of interest. It is based on X-ray photons that are sent through the object. Some of these photons will be absorbed by the object. However, the amount of photons absorbed depends on the material's physical properties. CT utilizes this essential premise to reconstruct images by sending X-rays through an object from different angles and then counting how many were absorbed.

2.1.1 CT Scanning

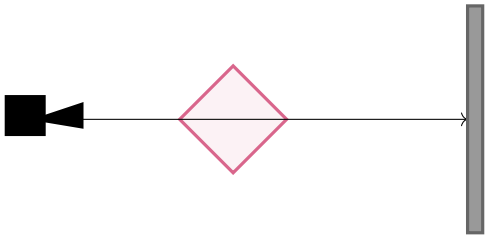


Figure 2.2: An illustration of a simple CT scanner setup.

To get a visual understanding of the setup and, thereby, the parts which make up the CT scanning process, consider Figure 2.2. This figure is a simplified version but it is good for an understanding of the different processes. To the left, we see an X-ray source that sends X-rays through the object (illustrated with a purple square), and to the right, we have a detector that detects the photons that did not get absorbed by the object.

To obtain a reconstruction of the interior of an object, we need a series of measurements from different angles and it is not enough with just one X-ray as shown in the simplified setup. Figure 2.3 shows an example of two X-ray projections onto the detector. This scan geometry is called the *parallel beam* projection. Figure 2.4 shows one projection for the scan geometry called *fan beam*. We will use both scan geometries in our studies. However, in the theory going forward, we will only talk about parallel beam since the principles are the same for the fan beam geometry.

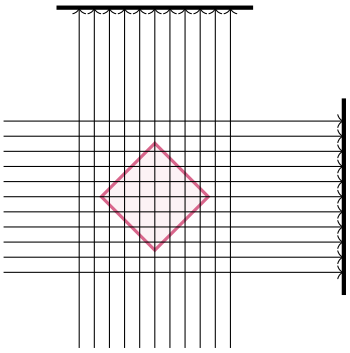


Figure 2.3: An illustration of a 2D parallel beam geometry with two projections.

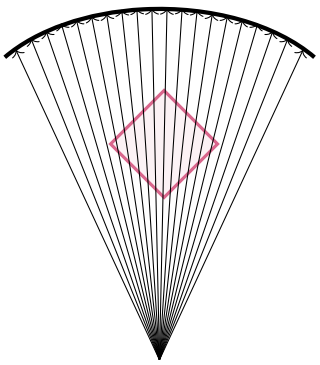


Figure 2.4: An illustration of a 2D fan beam geometry with one projection.

2.1.2 Physics

The final destination of our X-rays is the detector. The detector contains small sensors that register photons. An X-ray is a beam of photons with frequency ranging from 30 petahertz (PHz) to 30 exahertz (EHz) [17, Ch. 4]. When the X-ray passes through an object some of its photons can either interact with the material or pass through without any interaction. If a photon result in interaction with a material, it can either be absorbed or change its path [17, Ch. 4], however, in both cases the photon will not reach the detector. This leads to a decrease in the X-ray intensity, which we refer to as the *attenuation*.

This physical property of (monochromatic) X-rays is described by the linear first-order ordinary differential equation (for polychromatic X-rays, we refer to [17, Ch. 4]):

$$\frac{\mathrm{d}I}{\mathrm{d}l} = -\mu(l)I(l), \tag{2.1}$$

where l is the position on the line of the X-ray, I is the intensity at position l, dI is the change in intensity, $\mathrm{d}l$ is the thickness of the object, and μ is the *linear attenuation coefficient*. (2.1) is Lambert-Beers law and it can be solved by separation of variables. With the initial condition $I(0) = I_0$, we get the solution

$$I(l) = I_0 e^{-\int \mu(l)\mathrm{d}l}, \tag{2.2}$$

where I_0 is the initial intensity. (2.2) describes an exponential decay in intensity.

2.2 X-Ray Computed Tomography as an Inverse Problem

We will now derive a mathematical model for the forward problem [17, Ch. 5]. Figure 2.5 shows the geometry on which our mathematical model will be based. The line $L(\theta, s)$

is the direction of the X-rays at measurement angle θ and where the s-axis is emanating from θ. The lines $L(\theta, s)$ are orthogonal to the s-axis for parallel beam projections.

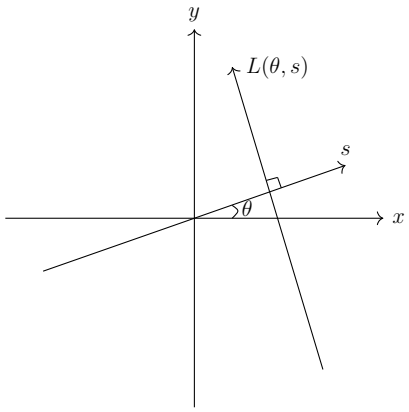


Figure 2.5: The geometry we use in our mathematical modeling of the CT problem.

The mathematical model of the forward problem for 2D parallel beam CT problems is given by the *Radon transform* denoted $\mathcal{R}$. Thus, the forward problem takes the form

$$\mathcal{R}[f](\theta, s) = g(\theta, s), \tag{2.3}$$

where $f : \mathbb{R}^2 \to \mathbb{R}$ is the image function of the object and g is a function of the measured data. f assigns an intensity $f(x, y)$ to each point. The intensity of a point (x, y) is based on the attenuation constant of the object at (x, y). We assume without loss of generality that f has local support on the unit disk $\mathbb{D}$ (illustrated by the blue circle in Figure 2.6), and is zero elsewhere.

The function g defines the projections of all line integrals along $L_{\theta,s}$ at a fixed angle θ as

$$g(\theta, s) = \int_{L_{\theta,s}} f(x, y) \, \mathrm{d}l, \quad s \in [-1, 1]. \tag{2.4}$$

An example of the function g is shown in Figure 2.6. In connection to Lambert-Beers law, we can rewrite (2.2) as

$$\log\left(\frac{I_0}{I(l)}\right) = \int_L \mu(l) \, \mathrm{d}l. \tag{2.5}$$

On this form, we see that we can write Lambert-Beers law as a Radon transform where

$$\begin{aligned} g(\theta, s) &= \log\left(\frac{I_0}{I_{\theta,s}}\right), \\ f(x, y) &= \mu(x, y), \end{aligned} \tag{2.6}$$

where (x, y) are points on the line $L_{\theta,s}$. The collection of g for all measurement angles θ_k, $k = 1, 2, \ldots$, are called the *sinogram*.

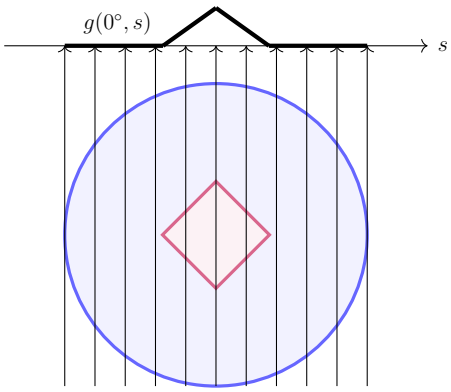


Figure 2.6: An image function with local support on the unit disk with a square object. An example is shown for the projection $g(\theta = 0°, s)$ which is obtained from integrating along the lines $L_{\theta=0°,s}$ for $s \in [-1, 1]$.

From the definition of g given in (2.4), we can rewrite (2.3) and the Radon transform yields

$$\mathcal{R}[f](\theta, s) = g(\theta, s) = \int_{L_{\theta,s}} f(x_1, x_2)\ \mathrm{d}l, \quad \theta \in [0°, 180°[, s \in [-1, 1], \tag{2.7}$$

which is the definition of the Radon transform. Thus, $\mathcal{R}[f]$ is defined as the projections at all angles.

For us to make numerical computations on a computer, we need to discretize (2.7). It is often straightforward to discretize the sinogram g as it is often measured by a digital detector which, therefore, already is divided into a discrete array of detector elements. Furthermore, the number of angular measurements is finite. So in the next section, we will consider discretizing the image f and the forward operator $\mathcal{R}$.

2.3 Discretization of X-Ray Computed Tomography Problems

This section is based on [17, Ch. 6 + 9]. In (2.7) the image function f is a continuous function. This function needs to be discretized. On computers, 2D images are represented by 2D arrays where we call the elements for pixels. We will denote a pixel as π_j and it is defined as

$$\pi_j = \{(x, y) \in \mathbb{R}^2 :\ x_j^{\text{left}} \leq x \leq x_j^{\text{right}} \text{ and } y_j^{\text{bottom}} \leq y \leq y_j^{\text{top}}\}. \tag{2.8}$$

Figure 2.7 shows an example of dividing the image into a 2D array of pixels. With n pixels, we also have n pixel values which we store in the vector

$$\mathbf{f} = \begin{pmatrix} f_1 \\ \vdots \\ f_n \end{pmatrix} \in \mathbb{R}^n. \tag{2.9}$$

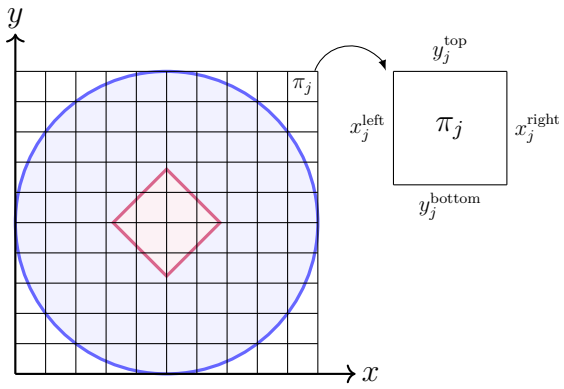


Figure 2.7: Example of a pixel grid.

Moreover, we define a pixel indicator function $\chi_{\pi_j} : \mathbb{R}^2 \to \{0, 1\}$ by

$$\chi_{\pi_j}(x, y) = \begin{cases} 1 & \text{if } (x, y) \in \pi_j, \\ 0 & \text{otherwise.} \end{cases} \tag{2.10}$$

With (2.9) and (2.10), we can then write $f(x, y)$ as a pixelated image function

$$f_\square(x, y) = \sum_{j=1}^{n} f_j \chi_{\pi_j}(x, y), \tag{2.11}$$

which is constant in each pixel and, therefore, $f_\square$ is a piecewise constant function in $\mathbb{R}^2$.

The lines $L_{\theta,s}$ intersect the pixels π, and the intersection between a line and a pixel is called the intersection length $L_{\theta,s}^{(\pi_j)}$. If the line do not intersect π_j then its value is zero, so $L_{\theta,s}^{(\pi_j)}$ is computed by

$$L_{\theta,s}^{(\pi_j)} = \begin{cases} \text{length of line segment } L_{\theta,s} \cap \pi_j, \\ 0 \text{ if } \mathrm{L_{\theta,s}} \cap \pi_\mathrm{j} = \emptyset. \end{cases} \tag{2.12}$$

This yields that the Radon transform of χ_{π_j} for pixel π_j at position (θ, s) is the line integral within π_j along $L_{\theta,s}$ as $\chi_{\pi_j} = 1$ when we are inside pixel π_j:

$$\mathcal{R}[\chi_{\pi_j}](\theta, s) = \int_{L_{\theta,s}} 1 \, \mathrm{d}l = L_{\theta,s}^{(\pi_j)}. \tag{2.13}$$

With this formulation, we have found a way to discretize the image function f and the Radon transform $\mathcal{R}$.

We consider the detectors to be placed in a 1D line. The detectors are adjacent and non-overlapping. For each angle θ_k a detector coordinate s_l is associated with the lth element

out of L detector elements in total. We assume s_l is in the middle of the detector element l if nothing else is stated. With this formulation of the detectors, we can construct a discrete system of the forward problem

$$b_l = \sum_{j=1}^{n} a_{kl}^{(j)} f_j, \quad l \in [1, 2, \ldots, L], \tag{2.14}$$

where $a_{kl}^{(j)}$ is the line-pixel coefficient or in some CT literature called the weight.

There are different possibilities for the discretization scheme for $a_{kl}^{(j)}$. We call these schemes for *projection models* and we will in the following sections consider three methods for modeling the forward projection. We will consider the *line model*, the *strip model*, and *Joseph's model*. The forward projection models all leads to constructing the sinogram. Moreover, we will briefly discuss the *back projection* method as it is an important operator in the reconstruction methods which we will consider in later chapters.

2.3.1 The Line Model

The first projection model we consider is called the line model and an illustration of the concept is shown in Figure 2.8. We want to determine the value of pixel π_j at detector s_l. A line runs from the center of detector element l and through pixel π_j. The line intersects π_j and the measured data at the lth detector will be the length of the line integral within π_j.

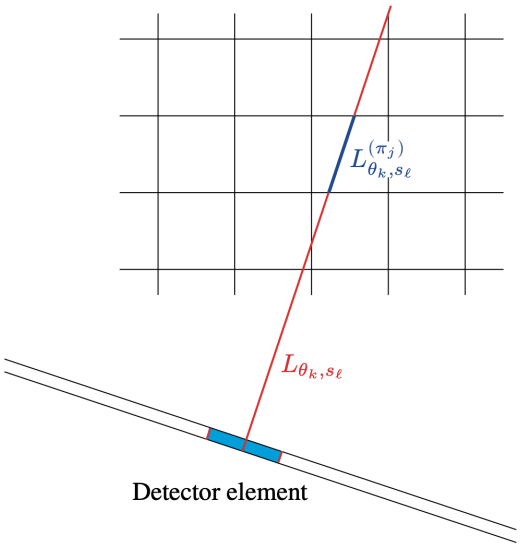


Figure 2.8: This illustration is from [17, Ch. 9] and it shows the concept of the line model.

To formulate the line model discretely, we consider the pixelated image function $f_\square$ in

(2.11) and taking the Radon transform of $f_\square$, we obtain

$$\mathcal{R}\{f_\square\}(\theta_k, s_l) = \sum_{j=1}^{n} f_j \mathcal{R}[\chi_{\pi_j}](\theta_k, s_l) = \sum_{j=1}^{n} f_j L^{(\pi_j)}_{\theta_k, s_l}. \tag{2.15}$$

Considering (2.14), we see that the weight $a^{(j)}_{kl} = L^{(\pi_j)}_{\theta_k, s_l}$.

2.3.2 The Strip Model

The second projection model we will introduce is the strip model. An illustration of the concept of this model for 2D parallel beam CT is shown in Figure 2.9. This model is based on the detector having a finite size. We let s_l^{left} and s_l^{right} denote the edges of detector element l and s_l denote the center. The intersecting lines are defined by the set $\{L_{\theta_k,s} : s_l^{\text{left}} \leq s_l \leq s_l^{\text{right}}\}$ and they form a strip in the (x, y) plane.

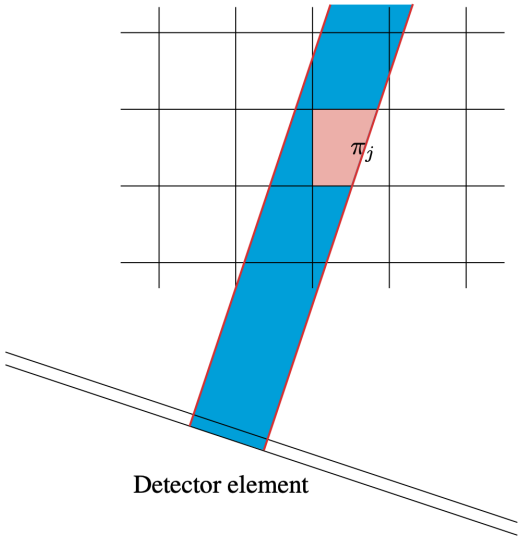


Figure 2.9: This illustration is from [17, Ch. 9] and it shows the concept of the strip model.

We again use $f_\square$ to formulate the discrete strip model, so the recorded data from pixel π_j in the lth element is given by the integral

$$\begin{aligned} \int_{s=s_l^{\text{left}}}^{s_l^{\text{right}}} \mathcal{R}[f_\square](\theta_k, s)\ \mathrm{d}s &= \sum_{j=1}^{n} f_j \int_{s=s_l^{\text{left}}}^{s_l^{\text{right}}} \mathcal{R}[\chi_{\pi_j}](\theta_k, s)\ \mathrm{d}s \\ &= \sum_{j=1}^{n} f_j \int_{s=s_l^{\text{left}}}^{s_l^{\text{right}}} L^{(\pi_j)}_{\theta_k, s_l}\ \mathrm{d}s \end{aligned} \tag{2.16}$$

Thus, we see that the weight computed from the strip model yields

$$a^{(j)}_{kl} = \int_{s=s_l^{\text{left}}}^{s_l^{\text{right}}} L^{(\pi_j)}_{\theta_k, s_l}\ \mathrm{d}s, \tag{2.17}$$

which is the area of π_j that the strip intersects, as shown in Figure 2.9.

2.3.3 Joseph's Model

The last forward projection model we will introduce is Joseph's model [22] (sometimes called the interpolation model). Figure 2.10 shows an illustration of the concept for this model when $\theta_k \in [-45°, 45°]$. The pixels in the grid are of size $\gamma \times \gamma$ which we will use in the derivation of the Joseph model.

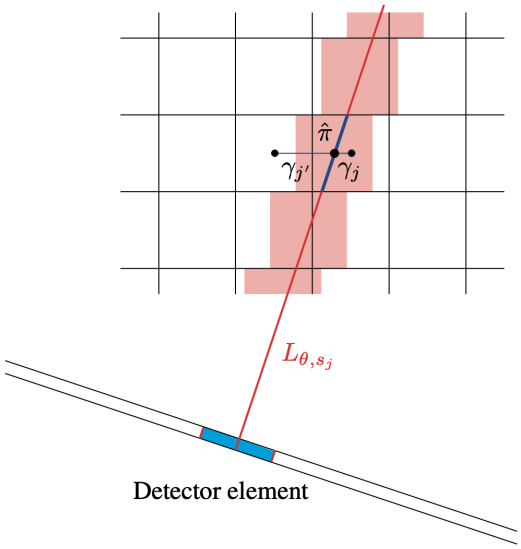


Figure 2.10: This illustration is from [17, Ch. 9] and it shows the concept of Joseph's model.

From the middle of the detector, we have the line L_{θ_k,s_l} (in Figure 2.10 its marked L_{θ,s_j}). For each row an artificial pixel $\hat{\pi}$ of size $\gamma \times \gamma$ is placed along the line, as illustrated by the pink squares in Figure 2.10. Each pixel $\hat{\pi}$ overlaps with two neighboring pixels in the row and its center is placed at the intersection between L_{θ_k,s_l} and the line segment between the two neighboring pixels. We can compute the intersection length of $\hat{\pi}$ using trivial trigonometry as

$$L_{\theta_k,s_l}^{(\pi_j)} = \frac{\gamma}{\cos\theta_k}. \tag{2.18}$$

With the creation of artificial pixels, we can use the line model on $\hat{\pi}$ instead of π. The advantage is that the intersection length is the same for all $\hat{\pi}$ associated with (θ_k, s_l). The pixel value of $\hat{\pi}$ is determined by the linear interpolation between the pixel value of the two neighbors denoted π_j and $\pi_{j'}$. We define γ_j and $\gamma_{j'}$ to be the lengths between the centers of π_j and $\pi_{j'}$, respectively, and the center of $\hat{\pi}$ so it applies that $\gamma_j + \gamma_{j'} = \gamma$. The pixel intensity of $\hat{\pi}$ found with linear interpolation then yields

$$f_{\text{interp}} = \frac{\gamma_{j'}}{\gamma} f_j + \frac{\gamma_j}{\gamma} f_{j'}, \tag{2.19}$$

where f_j and $f_{j'}$ is the pixel intensities for π_j and $\pi_{j'}$, respectively. The total contribution to the line integral along L_{θ_k,s_l} is obtained by multiplying f_{interp} with the intersection length (2.18). Thus, we obtain

$$\begin{aligned} L_{\theta_k,s_l} f_{\text{interp}} &= L_{\theta_k,s_l} \frac{\gamma_{j'}}{\gamma} f_j + L_{\theta_k,s_l} \frac{\gamma_j}{\gamma} f_{j'}, \\ &= \frac{\gamma}{\cos\theta_k} \frac{\gamma_{j'}}{\gamma} f_j + \frac{\gamma}{\cos\theta_k} \frac{\gamma_j}{\gamma} f_{j'}, \\ &= \frac{\gamma_{j'}}{|\cos\theta_k|} f_j + \frac{\gamma_j}{|\cos\theta_k|} f_{j'}. \end{aligned} \tag{2.20}$$

From (2.20), we see that

$$a_{kl}^{(j)} = \frac{\gamma_{j'}}{|\cos\theta_k|} \text{ and } a_{kl}^{(j')} = \frac{\gamma_j}{|\cos\theta_k|}. \tag{2.21}$$

Thus, we need to compute γ_j and $\gamma_{j'}$ and for this, we need to determine the location in the horizontal direction of the center in $\hat{\pi}$. The advantage of Joseph's model is that we can reuse computations for adjacent rows making it computationally efficient [17, Ch. 9].

2.3.4 Back Projection Model

We have until now covered the Radon transform and how it maps an image f to a sinogram g through projections. However, to solve an inverse problem we need to model the adjoint process of the forward propagation, namely the *back projection operator*. Thus, we must have a mathematical model that recovers an image f from its sinogram g.

For each pair (θ, s) we have a projection $g(\theta, s)$ which we "smear" back across the image along the line of integration. To recover the full image f, we can write this back projection as an operator $\mathcal{R}^{\#}$ that maps a sinogram g to an image f as an integration over θ [17, Ch. 6]:

$$\mathcal{R}^{\#}[g](x,y) = \int_0^{2\pi} g(\theta, x\cos\theta + y\sin\theta)\ \mathrm{d}\theta. \tag{2.22}$$

We can discretize (2.22) by replacing the integration over θ with a sum over the discrete measurement angles $\theta_1, \theta_2, \ldots, \theta_{N_\theta}$ [17, Ch. 9]. Hence, the center (x^j, y^j) of pixel π_j can be approximated by

$$\mathcal{R}^{\#}[g](x^j, y^j) = \frac{2\pi}{N_\theta} \sum_{k=1}^{N_\theta} g(\theta_k, x^j\cos\theta_k + y^j\sin\theta_k), \tag{2.23}$$

where $x^j\cos\theta_k + y^j\sin\theta_k$ defines the point $s_{j,k}$ on the detector. The concept is shown in Figure 2.11. It is clear from the figure, that $s_{j,k}$ will not be in the center of the detector element as it does not follow the line segment L_{θ_k,s_l} (referred to as $L_{\theta,s}$ in Figure 2.11).

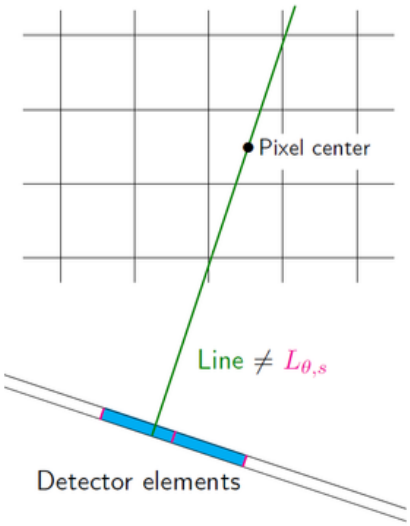


Figure 2.11: Illustration from [12] showing the concept of back projection.

To obtain an approximation of g at position $s_{j,k}$, we will need to interpolate between the discrete values of the sinogram. The value at $s_{j,k}$ depends on the two closest neighbour detector elements. We denote the center of these neighbours $s_{j,k}^{\text{left}}$ and $s_{j,k}^{\text{right}}$ such that $s_{j,k}^{\text{left}} \leq s_{j,k} \leq s_{j,k}^{\text{right}}$. If we use linear interpolation, we obtain

$$g(\theta_k, x^j \cos\theta_k + y^j \sin\theta_k) \approx \alpha_{j,k} g\left(\theta_k, s_{j,k}^{\text{left}}\right) + (1-\alpha_{j,k})\, g\left(\theta_k, s_{j,k}^{\text{right}}\right), \tag{2.24}$$

where $\alpha_{j,k}$ is the linear interpolation coefficient given by

$$\alpha_{j,k} = \left| \frac{s_{j,k} - s_{j,k}^{\text{right}}}{s_{j,k}^{\text{left}} - s_{j,k}^{\text{right}}} \right|. \tag{2.25}$$

Thus, (2.23) takes the form

$$\mathcal{R}^{\sharp}[g]\left(x^j, y^j\right) \approx \frac{2\pi}{N_\theta} \sum_{k=1}^{N_\theta} \alpha_{j,k} g\left(\theta_k, s_{j,k}^{\text{left}}\right) + (1-\alpha_{j,k})\, g\left(\theta_k, s_{j,k}^{\text{right}}\right). \tag{2.26}$$

There exist other interpolation schemes such as nearest-neighbor and spline interpolation. If we were to use one of those, we would need to derive the weights $\alpha_{j,k}$ which we will not cover here.

2.3.5 The System Matrix

From the derivations above, it is now possible to describe the CT problem as a system of linear equations. If we consider one view for θ_k, then we can set up a *system matrix* $\mathbf{A}_k \in \mathbb{R}^{N_s \times n}$ where N_s is the number of detectors and n is the number of pixels. The

entries in $\mathbf{A}_k$ will be the weights $a_{kl}^{(j)}$ from the chosen forward projection model. So for the view θ_k, the system matrix will take the form

$$\boldsymbol{A}_k = \begin{pmatrix} a_{k1}^{(1)} & a_{k1}^{(2)} & \cdots & a_{k1}^{(n)} \\ a_{k2}^{(1)} & a_{k2}^{(2)} & \cdots & a_{k2}^{(n)} \\ \vdots & \vdots & & \vdots \\ a_{kN_s}^{(1)} & a_{kN_s}^{(2)} & \cdots & a_{kN_s}^{(n)} \end{pmatrix}. \tag{2.27}$$

We will often have a lot of angles in a CT problem and the number of angles will be denoted N_θ. We can assemble the angles by stacking $\mathbf{A}_k$ such that we obtain a large system matrix $\mathbf{A}$:

$$\boldsymbol{A} = \begin{pmatrix} \boldsymbol{A}_1 \\ \vdots \\ \boldsymbol{A}_{N_\theta} \end{pmatrix}, \tag{2.28}$$

where the dimension of $\mathbf{A}$ is $m \times n$ and $m = N_\theta N_s$. The relationship between the matrix elements a_{ij} and the weights can be described by

$$a_{ij} = a_{k\ell}^{(j)} \quad \text{with} \quad i = (k-1)N_s + \ell, \tag{2.29}$$

where j, k, and l are related to pixel π_j, projection angle θ_k, and detector coordinate s_l, respectively. Note that $\mathbf{A}$ is sparse due to (2.12).

Now that we have a discrete system matrix including the forward projections, we can write the relation between the measured data on the detector $\mathbf{b} \in \mathbb{R}^m$ and the pixel values $\mathbf{f} \in \mathbb{R}^n$ by

$$\mathbf{A}\mathbf{f} = \mathbf{b} \tag{2.30}$$

We change the notation of $\mathbf{f}$ to $\mathbf{x}$ to be consistent with the linear algebra literature. The discretization of the forward problem then becomes

$$\mathbf{A}\bar{\mathbf{x}} = \bar{\mathbf{b}}, \tag{2.31}$$

where $\bar{\mathbf{x}} \in \mathbb{R}^n$ is the true image and $\bar{\mathbf{b}} \in \mathbb{R}^m$ is the noise-free measurements on the detector.

In CT, we do not know $\bar{\mathbf{x}}$ and, hence, the inverse problem is to determine $\bar{\mathbf{x}}$. Real data will inevitably include noise so instead of (2.31) we need to solve

$$\mathbf{A}\bar{\mathbf{x}} = \mathbf{b}, \quad \mathbf{b} = \bar{\mathbf{b}} + \mathbf{e}, \tag{2.32}$$

where $\mathbf{e} \in \mathbb{R}^m$ is an error term due to noise from data acquisition. We will in this report consider the noise to be Gaussian white noise meaning $\mathbf{e} \sim \mathcal{N}(0, 1)$. The naive solution to (2.32) if $\mathbf{A}$ is square and $\mathbf{A}^{-1}$ exist is

$$\mathbf{x}_{\text{naive}} = \mathbf{A}^{-1}\mathbf{b} = \mathbf{A}^{-1}\left(\bar{\mathbf{b}} + \mathbf{e}\right) = \bar{\mathbf{x}} + \mathbf{A}^{-1}\mathbf{e}. \tag{2.33}$$

If $\mathbf{A}$ is rectangular, one must instead solve the normal equations. The naive solution will be dominated by noise as the component $\mathbf{A}^{-1}\mathbf{e}$ is more dominating than $\bar{\mathbf{x}}$ due to $\mathbf{A}$ being ill-conditioned, see Example 2.1.

Example 2.1: Naive solution
From (2.33), we see that due to noise it is not possible to reconstruct $\bar{\mathbf{x}}$. We can illustrate this by considering a small 2D CT test problem. Our 2D test problem is chosen to have a parallel beam geometry and the image $\bar{\mathbf{x}}$ is the Shepp Logan phantom of size 64×64. The system matrix $\mathbf{A}$ is computed from the toolbox ASTRA [1] with the strip model as the forward projection and $\theta_k \in [0°, 179°]$. We add 0.003 relative noise to $\bar{\mathbf{b}}$. Figure 2.12 shows the ground truth $\bar{\mathbf{x}}$ and the naive solution $\mathbf{x}_{\text{naive}}$. The naive solution is far from the true solution as it is dominated by noise.

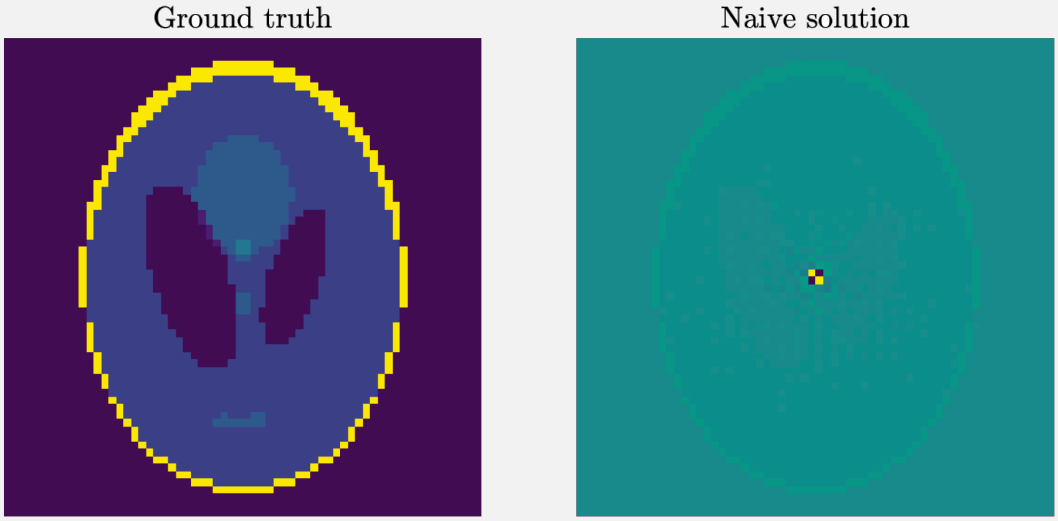


Figure 2.12: Example of a naive solution to the right and the ground truth to the right.

To get a better understanding of the properties of (2.32), we can use the *Singular Value Decomposition* (SVD) to analyze our problem.

2.4 Singular Value Decomposition

The Singular Value Decomposition is a powerful tool for analyzing linear systems. The SVD of a matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ where $m \geq n$ is a rank-revealing factorization that takes the form

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T = \sum_{i=1}^{n} \mathbf{u}_i \sigma_i \mathbf{v}_i^T, \tag{2.34}$$

where $\mathbf{\Sigma} \in \mathbb{R}^{n\times n}$ is a diagonal matrix with the singular values σ_i for $i = 1, \cdots, n$ sorted in a non-ascending order meaning $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 0$. $\mathbf{U} = \begin{pmatrix} \mathbf{u}_1 & \mathbf{u}_2 & \cdots & \mathbf{u}_n \end{pmatrix} \in \mathbb{R}^{m\times n}$ contain the left singular vectors and $\mathbf{V} = \begin{pmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_n \end{pmatrix} \in \mathbb{R}^{n\times n}$ contain the right singular vectors. Moreover, $\mathbf{U}$ and $\mathbf{V}$ are orthogonal matrices meaning $\mathbf{U}^T\mathbf{U} = \mathbf{V}^T\mathbf{V} = \mathbf{I}$. $\mathbf{V}$ is a basis of the domain and $\mathbf{U}$ is a basis of the image of the linear operator $\mathbf{A}$. Due to these vectors being the basis for our solution, it is interesting to study them.

For simplicity, we will shortly consider a 1D test problem for illustrative purposes. In the toolbox Regularization Tools [15] a 1D test function called gravity is given which computes $\mathbf{A}$, $\bar{\mathbf{x}}$, and $\bar{\mathbf{b}}$. We consider a problem of size 128×1 and we add 10^{-6} relative noise to $\bar{\mathbf{b}}$ to simulate noisy measurements. Figure 2.13 and 2.14 show some of the left and right singular vectors, respectively. From the figures, we see that when i increases so do the oscillations within $\mathbf{u}_i$ and $\mathbf{v}_i$. This means that singular vectors corresponding to high singular values represent smooth details of the solution whereas singular vectors corresponding to small singular values represent sharp and finer details in the solution. The same behavior is also seen when considering the 2D CT test problem from Section 2.3.5. Figure 2.15 and 2.16 show the left and right singular vectors, respectively.

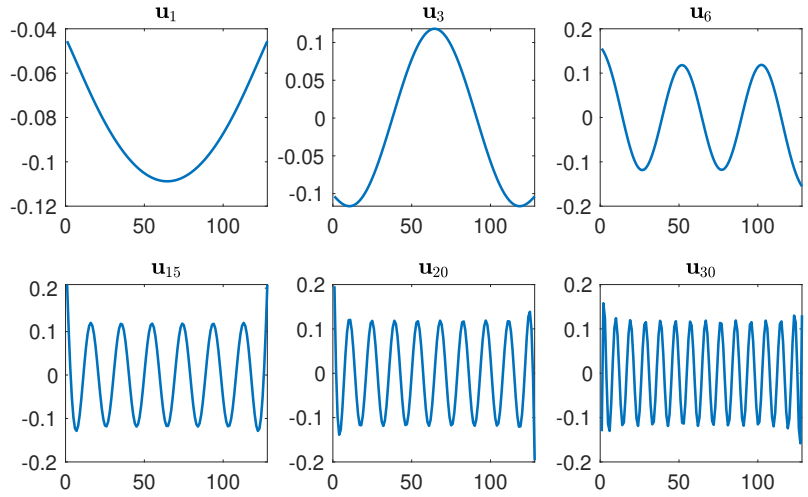


Figure 2.13: Examples of left singular vectors for the 1D test problem gravity.

The naive solution (2.33) can be written in terms of the SVD components as

$$\mathbf{x}_{\text{naive}} = \sum_{i=1}^{n} \frac{\mathbf{u}_i^T\mathbf{b}}{\sigma_i}\mathbf{v}_i = \sum_{i=1}^{n} \frac{\mathbf{u}_i^T\bar{\mathbf{b}}}{\sigma_i}\mathbf{v}_i + \sum_{i=1}^{n} \frac{\mathbf{u}_i^T\mathbf{e}}{\sigma_i}\mathbf{v}_i. \tag{2.35}$$

To investigate the behavior of the solution when noise is present, we can consider a *Picard plot*. In a Picard plot we plot the coefficients $|\mathbf{u}_i^T\mathbf{b}|$ and $|\mathbf{u}_i^T\mathbf{b}|/\sigma_i$ together with the singular values σ_i. An example of such a plot is shown in Figure 2.17 for our 1D test problem.

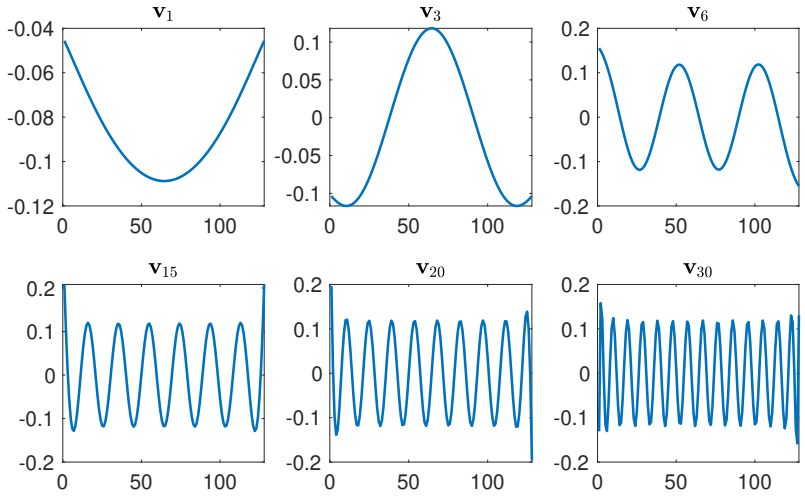


Figure 2.14: Examples of right singular vectors for the 1D test problem gravity.

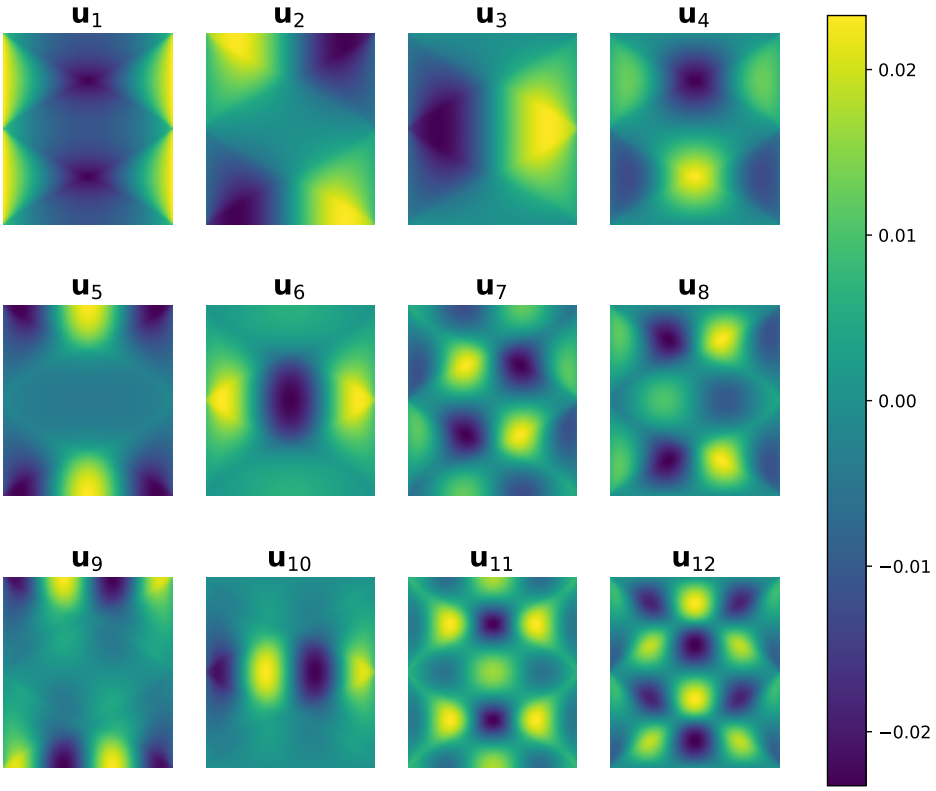


Figure 2.15: The first 12 left singular vectors $\mathbf{u}_i$ with the strip model as the forward projector.

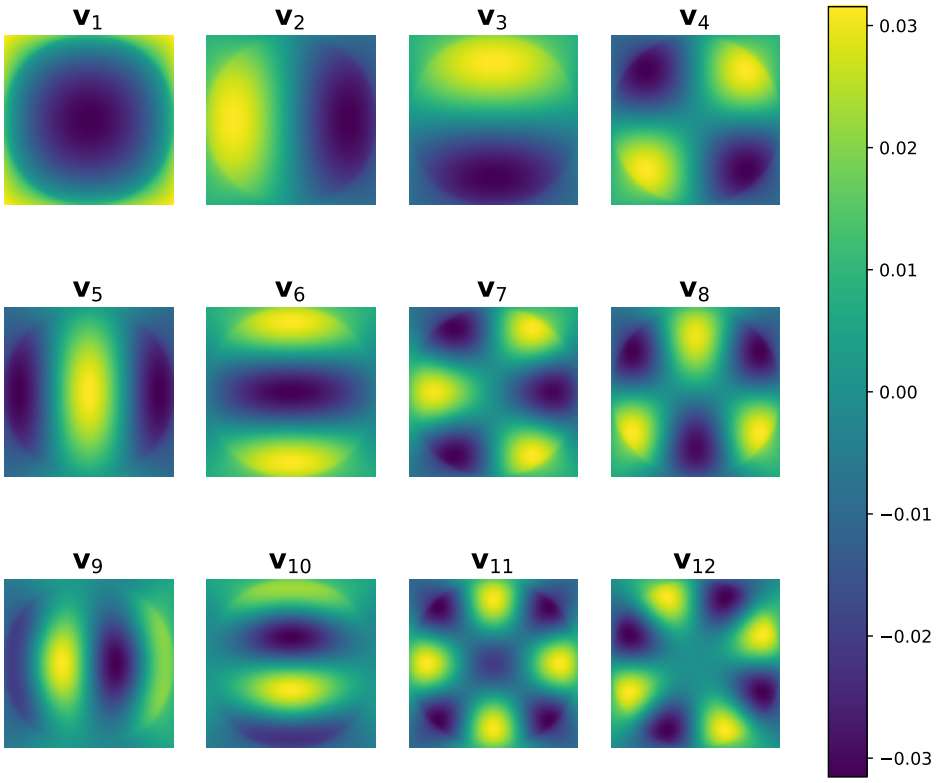


Figure 2.16: The first 12 right singular vectors $\mathbf{v}_i$ with the strip model as the forward projector.

Inspections of the Picard plot can help us analyze our inverse problem. We will obtain good reconstructions without dominating noise when the *discrete Picard condition* is satisfied [13, Ch. 3]. The discrete Picard condition is fulfilled if the coefficients $|\mathbf{u}_i^T\mathbf{b}|$ on average decay faster than σ_i until it reaches a level where σ_i levels off due to machine precision. In the example shown in Figure 2.17, we see that on average the coefficients $|\mathbf{u}_i^T\mathbf{b}|$ decay faster than the singular values σ_i until we reach the level of the noise at 10^{-6}. This is approximately at $i = 15$ and the following solutions will be dominated by the noise as the Picard condition is no longer satisfied. When the discrete Picard condition is no longer satisfied, the scalars $|\mathbf{u}_i^T\mathbf{b}|/\sigma_i$ increase significantly.

Considering (2.35), the values $\mathbf{u}_i^T\mathbf{e}$ are very small and they change very little. σ_i decreases as i increases. So the scalar $\mathbf{u}_i^T\mathbf{e}/\sigma_i$ will increase drastically as $i \to n$ due to $\sigma_i \to 0$ which we see in Figure 2.17. The scalar $\mathbf{u}_i^T\mathbf{e}/\sigma_i$ is multiplied with $\mathbf{v}_i$, so the high frequent vectors in $\mathbf{V}$ will have large amplitudes. These high frequent vectors $\mathbf{v}_i$ with large amplitudes are the reason for obtaining bad reconstruction from the naive solution as noise is dominating the solution.

Considering (2.35), we can investigate the solution as i increases. Figure 2.18 shows the

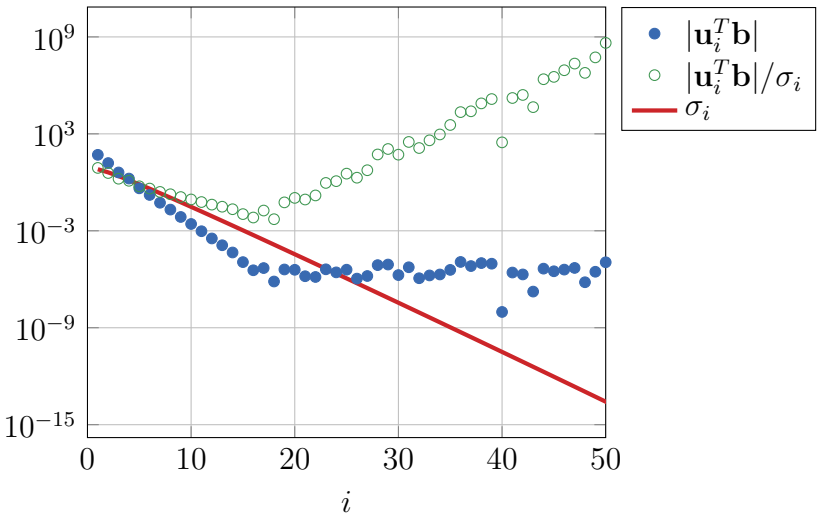


Figure 2.17: A Picard plot for the 1D gravity problem.

solution $\mathbf{x}_k$ given by

$$\mathbf{x}_k = \sum_{i=1}^{k} \frac{\mathbf{u}_i^T\mathbf{b}}{\sigma_i}\mathbf{v}_i, \tag{2.36}$$

for the 1D test problem. We see that $\mathbf{x}_k$ moves towards the true solution, however, at some point the noise begins to dominate and the values of $\mathbf{x}_k$ explode. Note, (2.36) is the regularization technique *Truncated Singular Value Decomposition* (TSVD).

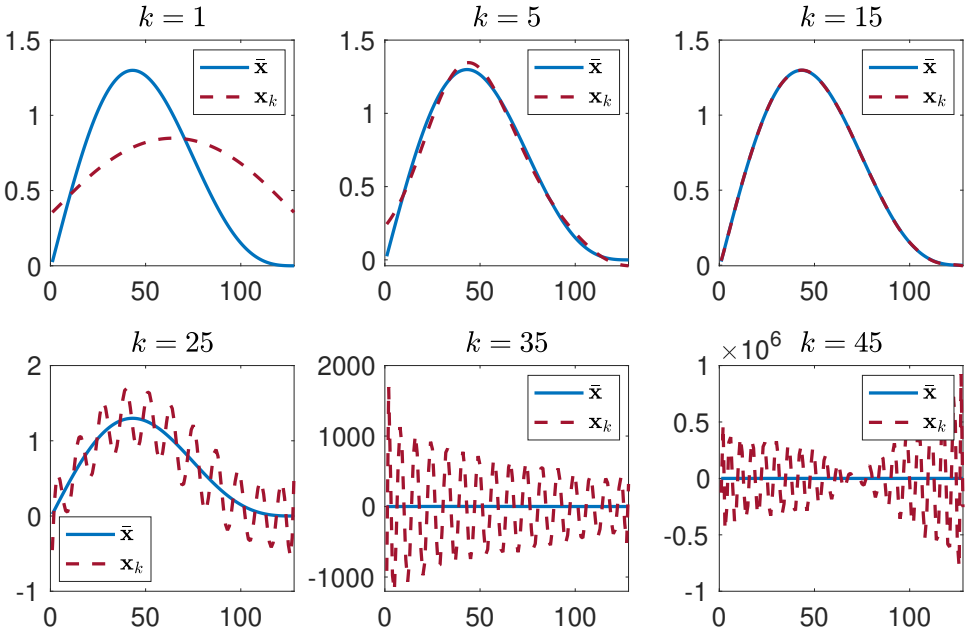


Figure 2.18: Example of (2.36) for $k = 1, 5, 15, 25, 35, 45$. The solid line is ground truth $\bar{\mathbf{x}}$ and the dashed line is the solution $\mathbf{x}_k$.

Finally, we relate to CT and Figure 2.19 shows the Picard plot for the first 2400 quantities of our 2D parallel beam test problem. We do not obtain as nice a plot as in Figure 2.17, but we can see that $|\mathbf{u}_i^T\mathbf{b}|$ on average decay faster than σ_i.

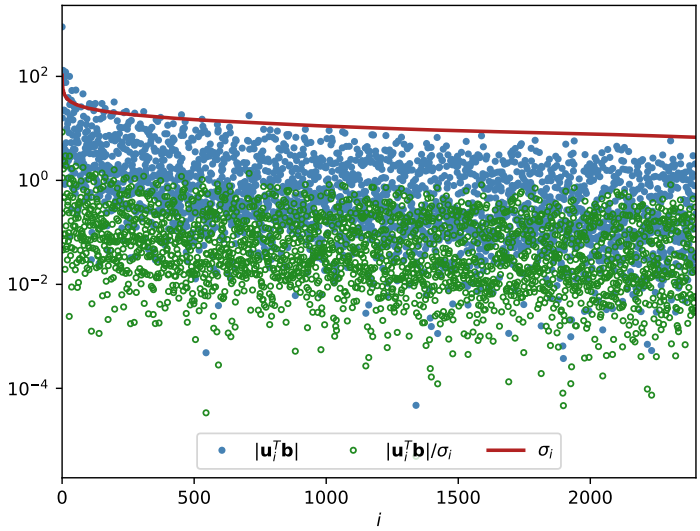


Figure 2.19: A Picard plot for the first 2400 quantities.

2.5 Summary

In this chapter, we introduced the physics behind CT and described their underlying mathematical models. We then investigated different discretization models used to describe the components of the linear system

$$\mathbf{A}\mathbf{x} = \mathbf{b}. \tag{2.37}$$

Here we focused on the forward and backward projection operators $\mathbf{A}$ and $\mathbf{A}^T$, respectively. For the forward projections, we considered the line, the strip, and Joseph's models. Solving the inverse problem (2.37) is not straightforward as the measurements $\mathbf{b}$ include noise and we saw that naively solving inverse problems can lead to unusable results. To get a better understanding of the issue, we did an SVD analysis on a CT test problem and illustrated that we can compute usable results using regularization.

CHAPTER 3

Iterative Methods

Chapter 2 introduced the mathematical model of our X-ray CT problem and described what challenges we face in connection with interference by noise in real data. This chapter considers how to reconstruct images when noise is present in the measurements.

CT problems are often large-scale rectangular problems with high sparsity. This is important information when looking at the possible reconstruction methods. We need to find a method that can handle such large problems and we need to think about how we store our data and make our computations most efficient, as problems can be too large to store on a computer. This yields looking at different discretization schemes for $\mathbf{A}$ and its transpose referred to as $\mathbf{B}$ resulting in an *unmatched projector pair*.

For reconstructions of CT problems, iterative methods are very popular. We will in this chapter discuss the advantage of such methods compared to direct methods. Moreover, we will cover the difficulties with CT problems concerning many iterative solvers. This will lead to the introduction of the iterative solver that we will study in this thesis, namely the *Generalized Minimal Residual* (GMRES).

GMRES is a *Krylov subspace method* which can be interpreted using our understanding of the singular vectors from the SVD. There exist several iterative Krylov subspace methods, but many face challenges with unmatched projector pairs as the matrices $\mathbf{AB}$ and $\mathbf{BA}$ are not guaranteed to be symmetric and positive definite. We will investigate the ABBA iterative methods, which are modifications to GMRES, to understand its advantage when solving systems with unmatched projectors. Moreover, this chapter will cover *semi-convergence* which is a property that is important to handle correctly for iterative solvers to obtain a good reconstruction.

Finally, we will consider two adjustments to the ABBA iterative methods. To obtain more efficient algorithms, we introduce restarted GMRES as this leads to better scaling. Furthermore, we will look at two stopping criteria, namely *Discrepancy Principle* (DP) and *Normalized Cumulative Periodogram* (NCP).

The structure of the chapter follows:

- How unmatched projector pairs arise

- Introduction to Krylov subspace methods and the challenge with unmatched projector pairs
- The iterative method Generalized Minimal Residuals
- The ABBA iterative methods
- Semi-convergence of inverse problems
- Reducing memory with restarted GMRES
- Early termination with the stopping criteria DP and NCP

3.1 Motivation

In Chapter 2, we showed how we could formulate our CT problem as a system of linear equations $\mathbf{Ax} = \mathbf{b}$. If $\mathbf{A}$ is square and has full rank, meaning that all singular values are non-zero, then the inverse of $\mathbf{A}$ exists. But as we saw in (2.33) the solution will include noise and we often deal with over/under-determined systems. Thus, we turn to other methods for solving our inverse problem.

A direct method for solving a linear system of equations would be to use e.g. SVD, *LU factorization* or *Gaussian elimination*. In Chapter 2, we used TSVD to obtain a regularized solution (see Figure 2.18). CT problems are large-scale sparse problems and TSVD has poor scaling $\mathcal{O}(mn^2)$ [13, Ch. 3] which does not exploit sparsity. Thus, we deem the TSVD reconstruction technique to be infeasible for sufficiently large problems.

Furthermore, $\mathbf{A}$ is often rank defect in CT problems, and the methods LU factorization and Gaussian elimination are designed for full rank square matrices. They are, therefore, not desired for CT reconstructions. Moreover, it is often not possible to store $\mathbf{A}$ and, therefore, we need to consider iterative methods instead of direct methods.

3.2 Matched and Unmatched Projector Pairs

When $\mathbf{A}$ is rectangular it means that solving the inverse problem relates to solving the normal equations

$$\mathbf{A}^T\mathbf{Ax} = \mathbf{A}^T\mathbf{b}, \tag{3.1}$$

which gives a least squares solution. $\mathbf{A} \in \mathbb{R}^{m\times n}$ is the forward projector covered in Section 2.3 which we can obtain from either the line, strip, or Joseph's model. $\mathbf{A}^T$ is the back projector which we will denote $\mathbf{B} \in \mathbb{R}^{n\times m}$. So (3.1) can be written as

$$\mathbf{BAx} = \mathbf{Bb}. \tag{3.2}$$

When $\mathbf{B} = \mathbf{A}^T$ we call (3.2) for the *matched normal equations*. Iterative solvers usually rely on a sequence of matrix-vector products with $\mathbf{A}$ and $\mathbf{B}$ to solve (3.2). Thus, modern implementations focus on fast matrix-vector products for better performance. Here, it can be preferred to choose different discretization schemes and model approximations for $\mathbf{A}$ and $\mathbf{B}$ [20]. This results in $\mathbf{B} \neq \mathbf{A}^T$ which means that we now solve the *unmatched normal equations* (3.2) with the unmatched projector pair $\mathbf{BA}$.

The simplest iterative method is called *Landweber* (or gradient decent) and it is given by

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \omega \mathbf{A}^T \left(\mathbf{b} - \mathbf{A}\mathbf{x}^{(k)} \right), \quad k = 1, 2, \ldots, \tag{3.3}$$

where ω is the step length and k is the number of iterations. Here we assume that we have a matched projector pair. When having an unmatched projector pair the method yields

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \omega \mathbf{B} \left(\mathbf{b} - \mathbf{A}\mathbf{x}^{(k)} \right), \quad k = 1, 2, \ldots. \tag{3.4}$$

Often in practice, the unmatched projector pair $\mathbf{BA}$ has complex eigenvalues with a negative real part, thus, $\mathbf{BA}$ is not positive definite and Landweber is not guaranteed to converge cf. [16] and [9]. Thus, for Landweber to converge, we would need to modify this method by shifting all the eigenvalues [8, Theorem 1.1 (Brauer)] of $\mathbf{BA}$ to the right, such that we avoid eigenvalues with negative real parts. This yields

$$\mathbf{x}^{(k+1)} = (1 - \alpha\omega)\mathbf{x}^{(k)} + \omega \mathbf{B} \left(\mathbf{b} - \mathbf{A}\mathbf{x}^{(k)} \right), \quad k = 1, 2, \ldots, \tag{3.5}$$

where α is a parameter that controls how much we would need to shift the negative real part eigenvalues to obtain only positive real parts. We would need to tune both α and ω. Furthermore, we would need to compute the smallest eigenvalues to ensure convergence. Since CT problems can deal with a massive amount of unknowns, this is not preferable so instead of looking at Landweber, we will consider Krylov subspace methods.

3.3 Krylov Subspace Methods

In our SVD analysis (see Section 2.4), we showed what happened when we included more and more singular vectors and singular values to our solution using TSVD (2.36) (see Figure 2.18). The solution $\mathbf{x}_k$ approached $\bar{\mathbf{x}}$ until the noise started dominating. Thus, we can say that the best achievable approximation $\mathbf{x}_k$ to $\bar{\mathbf{x}}$ lies within a low-dimensional subspace of $\mathbb{R}^n$. For our large-scale CT problems, it is computationally infeasible and sometimes impossible to compute the SVD. However, we still want our reconstruction $\mathbf{x}$ to consist of the low-frequency components as this reduces the interference of noise. Based on this, we can write our least squares problem for our inverse problem as

$$\min_{\mathbf{x}} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2 \quad \text{s.t.} \quad \mathbf{x} \in \mathcal{W}_k = \text{span}\{w_1, \cdots, w_k\}, \tag{3.6}$$

where $\mathcal{W}_k \in \mathbb{R}^{n \times k}$ is a suitable basis for the low-dimensional subspace. As just discussed, the optimal set of basis vectors for $\mathbf{A}$ would be the singular vectors. However, CT

problems are too large for computing the SVD and that is why we will now consider the Krylov subspace which is defined as [13, Ch. 6]:

$$\mathcal{K}_k \equiv \text{span}\{\mathbf{A}^T\mathbf{b}, (\mathbf{A}^T\mathbf{A})\mathbf{A}^T\mathbf{b}, (\mathbf{A}^T\mathbf{A})^2\mathbf{A}^T\mathbf{b}, \cdots, (\mathbf{A}^T\mathbf{A})^{k-1}\mathbf{A}^T\mathbf{b}\}. \tag{3.7}$$

From (3.7), we see that the dimension of the Krylov subspace is at most k. The Krylov subspace is closely related to the power method [10] which seeks to find the largest eigenpair of a matrix as the subspace is spanned by the iteration vectors. In our setting, we are computing the eigenpairs of $\mathbf{A}^T\mathbf{A}$ which corresponds to computing the singular values σ_i and the right singular vectors v_i of $\mathbf{A}$. The basis vectors within the subspace $\mathcal{K}_k$ are not suitable to use as a basis for our solution of (3.6) as the basis vectors of $\mathcal{K}_k$ converge to the principle eigenvector of $\mathbf{A}^T\mathbf{A}$ and, thus, are not close to being orthogonal. However, we can still use the subspace $\mathcal{K}_k$ to form a well-suited subspace $\mathcal{W}_k$ by orthonormalizing the Krylov vectors of (3.7). Example 3.1 illustrates the importance of orthonormalizing the Krylov vectors and why they can be interpreted as an approximation to a basis of singular vectors.

There are many Krylov subspace methods we can choose from, e.g., *Conjugate Gradient Least Squares* [7] (CGLS), CGLS's stabilized version LSQR [24], or GMRES [26]. CGLS and LSQR both solve least squares problems for symmetric and positive definite matrices while GMRES solve problems with quadratic matrices which do not need to be symmetric or positive definite. As the unmatched system matrix $\mathbf{BA}$ is neither symmetric nor positive semi-definite [16, p. 7], we cannot use the CGLS and LSQR reconstruction methods. GMRES, on the other hand, can handle non-symmetric and non-positive semi-definite matrices. Another advantage is that we do not need to tune any parameters and it is easy to implement.

Example 3.1: Approximating singular vectors with a Krylov subspace
In this example from [13, Ch. 6], we illustrate why we need to orthoganalize the Krylov vectors which are shown in Figure 3.1. The top row shows the basis vectors of $\mathcal{K}_k$ for $k = 1, 2, \ldots, 5$ before making them orthogonal and the second row from the top shows the right singular vectors multiplied with the 5 Krylov vectors. This shows that all Krylov vectors are orthogonal to all but the first singular vector and, therefore, none of the other singular vectors are represented. This is not a good basis as we need more than just the first singular vector to obtain a good solution (as we saw in Figure 2.18).

The third row from the top is the orthogonalized Krylov vectors for $k = 1, 2, \ldots, 5$ and the bottom row is the multiplication with the right singular vectors. We see that more right singular vectors are described by the Krylov subspace as Krylov vectors now are orthogonal to the different singular vectors.

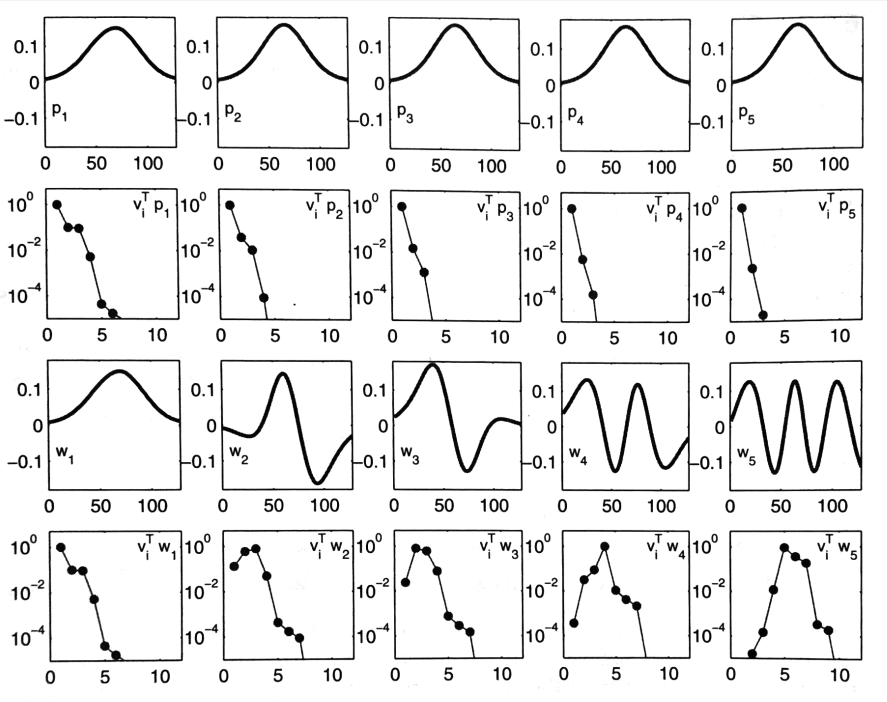


Figure 3.1: Illustration from [13, Ch. 6, Fig. 6.6]. The top row shows an example of the 5 first Krylov subspace vectors from (3.7). The second row from the top shows all singular vectors multiplied by each Krylov vector. The third row from the top shows the orthongonalized Krylov vectors. The bottom row shows the singular vectors multiplied by the orthogonal Krylov vectors.

3.4 GMRES: The Generalized Minimal Residual Algorithm

We consider an arbitrary linear system given by

$$\mathbf{M}\mathbf{x} = \mathbf{d}, \tag{3.8}$$

where $\mathbf{M} \in \mathbb{R}^{n \times n}$, $\mathbf{x} \in \mathbb{R}^n$, and $\mathbf{d} \in \mathbb{R}^n$.

The Generalized Minimal Residual (GMRES) method [26] is designed to handle non-symmetric square matrices and it seeks to minimize the residual

$$\underset{\mathbf{x}}{\operatorname{argmin}} \|\mathbf{d} - \mathbf{M}(\mathbf{x}_0 + \mathbf{x})\|_2 = \underset{\mathbf{x}}{\operatorname{argmin}} \|\mathbf{r}_0 - \mathbf{M}\mathbf{x}\|_2, \tag{3.9}$$

by iteratively expanding a Krylov subspace $\mathcal{K}_k = \operatorname{span}\{\mathbf{r}_0, \mathbf{M}\mathbf{r}_0, \mathbf{M}^2\mathbf{r}_0, \cdots, \mathbf{M}^{k-1}\mathbf{r}_0\}$ where $\mathbf{r}_0 = \mathbf{d} - \mathbf{M}\mathbf{x}_0$ for $\mathbf{x}_0 \neq \mathbf{0}$ and $\mathbf{r}_0 = \mathbf{d}$ for $\mathbf{x}_0 = \mathbf{0}$. This yields minimizing

$$\mathbf{r}_k = \mathbf{r}_0 - \mathbf{M}\mathbf{x}_k, \quad \mathbf{x}_k \in \mathcal{K}_k. \tag{3.10}$$

To obtain the solution $\mathbf{x}_k$ which minimizes $\mathbf{r}_k$ as much as possible, we can use least squares. For each iteration k, the solution space obtained from the Krylov subspace expands and the matrix containing this information is denoted $\mathbf{W}_k$.

We can think of $\mathbf{x}_k$ as a linear combination of the Krylov subspace vectors, $\mathbf{x}_k = \mathbf{W}_k\mathbf{c}$, where $\mathbf{c}$ determines the weights for how much of each column we need to represent $\mathbf{x}_k$. Hence, we can rewrite (3.10) as

$$\mathbf{r}_k = \mathbf{r}_0 - \mathbf{M}\mathbf{W}_k\mathbf{c}. \tag{3.11}$$

We want to minimize the 2-norm of the residuals in (3.11) and the only thing that is not fixed is the weights $\mathbf{c}$. Thus, we can write the least squares solution as

$$\underset{\mathbf{c}}{\operatorname{argmin}} \left\|\mathbf{r}_0 - \mathbf{M}\mathbf{W}_k\mathbf{c}\right\|_2. \tag{3.12}$$

The vectors spanning the Krylov subspace are not necessarily orthogonal and normalized but it is an advantage to work with (see Example 3.1). We can orthogonalize the basis vectors of the Krylov subspace by a QR factorization, modified Gram-Schmidt, or other methods to obtain an orthonormal basis as $\mathbf{Q}_k$ consists of orthonormal vectors that span the subspace $\mathcal{K}_k$. So instead of solving (3.12), we solve

$$\underset{\mathbf{y}_k}{\operatorname{argmin}} \left\|\mathbf{r}_0 - \mathbf{M}\mathbf{Q}_k\mathbf{y}_k\right\|_2. \tag{3.13}$$

Thus, the relation between $\mathbf{c}$ and $\mathbf{y}_k$ is $\mathbf{y}_k = \frac{\mathbf{W}_k}{\mathbf{Q}_k}\mathbf{c}$.

It is computationally expensive to solve (3.13) for large system matrices $\mathbf{M}$. Therefore, GMRES utilizes Arnoldi iterations [2] to reduce the cost of solving (3.13). Arnoldi's method uses a Gram-Schmidt method to make an orthonormal basis $\mathbf{Q}_k$ of the Krylov subspace $\mathcal{K}_k$ and produces a Hessenberg matrix which is an upper triangular matrix with a sub-diagonal:

$$\mathbf{H}_k = \begin{bmatrix} h_{1,1} & h_{1,2} & h_{1,3} & \cdots & h_{1,k} \\ h_{2,1} & h_{2,2} & h_{2,3} & \cdots & h_{2,k} \\ 0 & h_{3,2} & h_{3,3} & \cdots & h_{3,k} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & h_{k,k-1} & h_{k,k} \end{bmatrix}. \tag{3.14}$$

The following holds for $\mathbf{H}$:

$$\mathbf{H}_k = \mathbf{Q}_k^T\mathbf{M}\mathbf{Q}_k. \tag{3.15}$$

Given an orthonormal basis $\mathbf{Q}_k$ of the kth Krylov subspace $\mathcal{K}_k$ of $\mathbf{M}$ then $\mathbf{H}_k$ is the low-dimensional representation of $\mathbf{M}$ in the basis $\mathbf{Q}_k$. (3.15) can be rewritten as

$$\mathbf{M}\mathbf{Q}_k = \mathbf{Q}_k\mathbf{H}_k. \tag{3.16}$$

We introduce $\tilde{\mathbf{H}}_k$ which is the same as $\mathbf{H}_k$ except it has an additional row with a non-zero element $h_{k+1,k}$:

$$\tilde{\mathbf{H}}_k = \begin{bmatrix} h_{1,1} & h_{1,2} & h_{1,3} & \cdots & h_{1,k} \\ h_{2,1} & h_{2,2} & h_{2,3} & \cdots & h_{2,k} \\ 0 & h_{3,2} & h_{3,3} & \cdots & h_{3,k} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & 0 & h_{k,k-1} & h_{k,k} \\ 0 & \cdots & \cdots & 0 & h_{k+1,k} \end{bmatrix}. \tag{3.17}$$

Then $\tilde{\mathbf{H}}_k$ satisfy the relation

$$\begin{aligned} \mathbf{M}\mathbf{Q}_k &= \mathbf{Q}_{k+1}\tilde{\mathbf{H}}_k, \\ \mathbf{M}\mathbf{q}_k &= h_{1,k}\mathbf{q}_1 + \cdots + h_{k,k}\mathbf{q}_k + h_{k+1,k}\mathbf{q}_{k+1}. \end{aligned} \tag{3.18}$$

Inserting (3.18) into (3.13) yields

$$\underset{\mathbf{y}_k}{\operatorname{argmin}} \left\| \mathbf{r}_0 - \mathbf{Q}_{k+1}\tilde{\mathbf{H}}_k\mathbf{y}_k \right\|_2. \tag{3.19}$$

Using the definition of our Krylov subspace, we arrive at

$$\underset{\mathbf{y}_k}{\operatorname{argmin}} \left\| \mathbf{q}_1\|\mathbf{r}_0\|_2 - \mathbf{Q}_{k+1}\tilde{\mathbf{H}}_k\mathbf{y}_k \right\|_2 = \underset{\mathbf{y}_k}{\operatorname{argmin}} \left\| \mathbf{Q}_{k+1}\left(\mathbf{e}_1\|\mathbf{r}_0\|_2 - \tilde{\mathbf{H}}_k\mathbf{y}_k\right) \right\|_2, \tag{3.20}$$

where $\mathbf{e}_1 = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \end{bmatrix}^T \in \mathbb{R}^{k+1}$. Solving (3.20) is significantly cheaper than solving (3.13) because we only have to factorize a matrix of size $(k+1) \times k$ instead of $n \times k$.

[26] provides the pseudo code for implementing GMRES which is shown in Algorithm 1. For an effective implementation, a QR factorization is a good choice when solving the minimization problem in line 12 of the algorithm. There is an effective way of updating a QR-factorization of $\mathbf{H}_k$, but we will not cover that in this thesis.

3.4.1 Convergence

A great strength of GMRES and other orthogonalized Krylov subspace solvers is their fast convergence. Given an $n \times n$ problem of full rank, then GMRES will converge to a solution in at most n steps [26]. This occurs as the orthogonalized basis $\mathbf{Q}_k$ approximates the right singular vectors and will, therefore, constitute a full basis of range($\mathbf{M}$) after n iterations.

However, we are not guaranteed consistency of our system as it is depending on the properties of the system. Considering a full rank consistent system, meaning the system matrix $\mathbf{M}$ is non-singular (we can write $\mathbf{M}^{-1}$) and $\mathbf{d} \in$ range($\mathbf{M}$), GMRES will converge to the true solution $\bar{\mathbf{x}}$. When the system is consistent but rank defect there are infinitely

Algorithm 1 GMRES

Choose initial $\mathbf{x}_0$
$\mathbf{r}_0 = \mathbf{d} - \mathbf{M}\mathbf{x}_0$
$\mathbf{w}_1 = \mathbf{r}_0 / \|\mathbf{r}_0\|_2$
for $j = 1, 2, \ldots, k$ **do**
    $\mathbf{q}_j = \mathbf{M}\mathbf{w}_j$
    for $i = 1, 2, \ldots, j$ **do**
        $h_{i,j} = (\mathbf{q}_j, \mathbf{w}_i)$
        $\mathbf{q}_j = \mathbf{q}_j - h_{i,j}\mathbf{w}_i$
    end for
    $h_{j+1,j} = \|\mathbf{q}_j\|_2$
    $\mathbf{w}_{j+1} = \mathbf{q}_j / h_{j+1,j}$
    $\mathbf{y}_k = \text{argmin}_{\mathbf{y}} \|\|\mathbf{r_0}\|_2 \mathbf{e}_1 - \mathbf{H}_k\mathbf{y}\|_2$
    $\mathbf{x}_k = \mathbf{x}_0 + \begin{bmatrix}\mathbf{w}_1 & \mathbf{w}_2 & \cdots & \mathbf{w}_k\end{bmatrix} \mathbf{y}_k$
end for

many solutions for $\bar{\mathbf{x}}$ and we will not know which of those solutions GMRES will converge to.

In the case where the system is inconsistent, meaning we have a singular system matrix $\mathbf{M}$ and $\mathbf{d} \notin \text{range}(\mathbf{M})$, GMRES do not converge to $\bar{\mathbf{x}}$. When $\mathbf{d} \notin \text{range}(\mathbf{M})$, we have noise in our data. The noise will make $\mathbf{d}$ lie in a higher dimensional space and this will make GMRES converge to $\mathbf{x}_{\text{naive}}$. However, GMRES will initially move towards $\bar{\mathbf{x}}$ for a certain number of iterations before converging to $\mathbf{x}_{\text{naive}}$. This is called *semi-convergence* and we will talk more about this in Section 3.6.

3.5 ABBA Iterative Methods

We have shown how GMRES can solve linear systems. However, GMRES is restricted to work on square matrices and in CT, the system matrix is usually rectangular, thus, we need to use AB/BA-GMRES. We consider our inverse problem (2.32) as a least squares problem

$$\min_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2, \tag{3.21}$$

where $\mathbf{A}^{m\times n}$. For a rectangular CT problem, we can have that $m \geq n$ (over-determined) or $m < n$ (under-determined which means we have a lack of angles). When the system is over-determined, (3.21) is equivalent to solving the normal equations

$$\mathbf{A}^T\mathbf{A}\mathbf{x} = \mathbf{A}^T\mathbf{b}, \tag{3.22}$$

where we are certain a solution exists. When we have an under-determined system (3.21) yields solving the normal equations

$$\mathbf{A}\mathbf{A}^T\mathbf{u} = \mathbf{b}, \quad \mathbf{x} = \mathbf{A}^T\mathbf{u}. \tag{3.23}$$

In CT problems, $\mathbf{A}$ and $\mathbf{A}^T$ denote the forward and backward projectors. It can be beneficial to not use the same discretization scheme for $\mathbf{A}$ and $\mathbf{A}^T$, and we will therefore denote the backward projector $\mathbf{B}$ instead of $\mathbf{A}^T$. If $\mathbf{B} = \mathbf{A}^T$, we have a matched projector pair and GMRES is equivalent to CGLS [7].

We will, in this thesis, consider solving the over- and under-determined unmatched CT problems. We refer to the method solving the under-determined system as AB-GMRES and solving the over-determined system as BA-GMRES. We refer to these methods as the ABBA iterative methods [18].

AB-GMRES solves the unmatched normal equations on the form

$$\mathbf{ABu} = \mathbf{b}, \quad \mathbf{x} = \mathbf{Bu}, \tag{3.24}$$

which yields minimizing

$$\min_{\mathbf{u}} \|\mathbf{b} - \mathbf{ABu}\|_2, \quad \mathbf{x} = \mathbf{Bu}, \tag{3.25}$$

where $\mathbf{u}_k \in \mathbf{u}_0 + \mathcal{K}_k(\mathbf{AB}, \mathbf{b})$ and $\mathbf{x}_k \in \mathbf{x}_0 + \mathcal{K}_k(\mathbf{BA}, \mathbf{Bb})$. The pseudo-code for implementing AB-GMRES is shown in Algorithm 2.

For BA-GMRES the unmatched normal equations take the form

$$\mathbf{BAx} = \mathbf{Bb}, \tag{3.26}$$

where we minimize

$$\min_{\mathbf{x}} \|\mathbf{Bb} - \mathbf{BAx}\|_2. \tag{3.27}$$

Thus, it means that BA-GMRES forms the iterates $\mathbf{x}_k \in \mathbf{x}_0 + \mathcal{K}_k(\mathbf{BA}, \mathbf{Bb})$ that minimizes (3.27). The pseudo code for implementing BA-GMRES is shown in Algorithm 3.

We see, that the solution $\mathbf{x}_k$ for both AB- and BA-GMRES is found in the same Krylov subspace spanned by $\mathcal{K}_k(\mathbf{BA}, \mathbf{Bb})$:

$$\mathcal{K}_k(\mathbf{BA}, \mathbf{Bb}) = \text{span}\{\mathbf{Bb}, \mathbf{BABb}, \cdots, (\mathbf{BA})^{k-1}\mathbf{Bb}\}. \tag{3.28}$$

In Appendix A.1 it is shown that GMRES and BA-GMRES are mathematically equivalent. The same can be done for GMRES and AB-GMRES.

Algorithm 2 AB-GMRES

Choose initial $\mathbf{x}_0$
$\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$
$\mathbf{w}_1 = \mathbf{r}_0 / \|\mathbf{r}_0\|_2$
for $k = 1, 2, \ldots, K$ **do**
    $\mathbf{q}_k = \mathbf{A}\mathbf{B}\mathbf{w}_k$
    for $i = 1, 2, \ldots, k$ **do**
        $h_{i,k} = \mathbf{q}_k^T \mathbf{w}_i$
        $\mathbf{q}_k = \mathbf{q}_k - h_{i,k}\mathbf{w}_i$
    end for
    $h_{k+1,k} = \|\mathbf{q}_k\|_2$
    $\mathbf{w}_{k+1} = \mathbf{q}_k / h_{k+1,k}$
    $\mathbf{y}_k = \operatorname{argmin}_{\mathbf{y}} \|\|\mathbf{r_0}\|_2 \mathbf{e}_1 - \mathbf{H}_k\mathbf{y}\|_2$
    $\mathbf{x}_k = \mathbf{x}_0 + \mathbf{B}\begin{bmatrix}\mathbf{w}_1 & \mathbf{w}_2 & \cdots & \mathbf{w}_k\end{bmatrix}\mathbf{y}_k$
    $\mathbf{r}_k = \mathbf{b} - \mathbf{A}\mathbf{x}_k$
    Stopping rule goes here
end for

Algorithm 3 BA-GMRES

Choose initial $\mathbf{x}_0$
$\mathbf{r}_0 = \mathbf{B}(\mathbf{b} - \mathbf{A}\mathbf{x}_0)$
$\mathbf{w}_1 = \mathbf{r}_0 / \|\mathbf{r}_0\|_2$
for $k = 1, 2, \ldots, K$ **do**
    $\mathbf{q}_k = \mathbf{B}\mathbf{A}\mathbf{w}_k$
    for $i = 1, 2, \ldots, k$ **do**
        $h_{i,k} = \mathbf{q}_k^T \mathbf{w}_i$
        $\mathbf{q}_k = \mathbf{q}_k - h_{i,k}\mathbf{w}_i$
    end for
    $h_{k+1,k} = \|\mathbf{q}_k\|_2$
    $\mathbf{w}_{k+1} = \mathbf{q}_k / h_{k+1,k}$
    $\mathbf{y}_k = \operatorname{argmin}_{\mathbf{y}} \|\|\mathbf{r_0}\|_2 \mathbf{e}_1 - \mathbf{H}_k\mathbf{y}\|_2$
    $\mathbf{x}_k = \mathbf{x}_0 + \begin{bmatrix}\mathbf{w}_1 & \mathbf{w}_2 & \cdots & \mathbf{w}_k\end{bmatrix}\mathbf{y}_k$
    $\mathbf{r}_k = \mathbf{b} - \mathbf{A}\mathbf{x}_k$
    Stopping rule goes here
end for

The convergence of the ABBA methods is discussed in [18].

3.6 Semi-Convergence of ABBA

So far, we have seen how GMRES solves a linear system, and how to apply it to CT problems (AB/BA-GMRES). However, as illustrated in Section 2.4, we need to consider the impact of noise on the right-hand side. We want to solve the noise-free system

$$\mathbf{A}\bar{\mathbf{x}} = \bar{\mathbf{b}}, \tag{3.29}$$

but we have the noisy system

$$\mathbf{A}\mathbf{x} = \bar{\mathbf{b}} + \mathbf{e}. \tag{3.30}$$

From Section 2.4, we saw that the optimal solution is spanned by a subset of the singular vectors of $\mathbf{A}$, where the inclusion of further vectors would deteriorate the solution quality. GMRES similarly tries to find an optimal solution in the Krylov subspace which is an approximation of the singular vectors. Recall the results shown in Figure 2.18; $\mathbf{x}_k$ in the initial iterations approach $\bar{\mathbf{x}}$ but at later iterations, the noise starts to dominate and the solution converges to the naive solution. This is called semi-convergence. GMRES has the same behavior as was briefly discussed in Section 3.4.1.

To obtain the solution closest to $\bar{\mathbf{x}}$, we aim to stop iterating when the convergence behavior changes. Thus, the regularized solution will not be too perturbed by the noise within the data. Through the Picard condition, we know when to stop including

SVD components in our reconstruction. As long as the Picard condition is satisfied, we approach ground truth $\bar{\mathbf{x}}$. When the Picard condition is no longer satisfied, we move away from the true solution as we converge to the naive solution $\mathbf{x}_{\text{naive}}$. So, we know from the SVD analysis that the optimal solution $\mathbf{x}_{\text{opt}}$ lies in a low-dimensional subspace which is good for the GMRES algorithm as this corresponds to performing fewer iterations. Example 3.2 illustrate the concept of semi-convergence.

Example 3.2: Concept of semi-convergence
Figure 3.2 shows an example of the convergence history for our 2D test problem when we use AB-GMRES with the matched transpose $\mathbf{B} = \mathbf{A}^T$ (the same applies to unmatched $\mathbf{B}$). The blue line illustrates the convergence towards $\bar{\mathbf{x}}$ and the red line illustrates when we diverge and converge to $\mathbf{x}_{\text{naive}}$. The black dot illustrates the optimal reconstruction $\mathbf{x}_{\text{opt}}$ as it is closest to the true solution $\bar{\mathbf{x}}$.

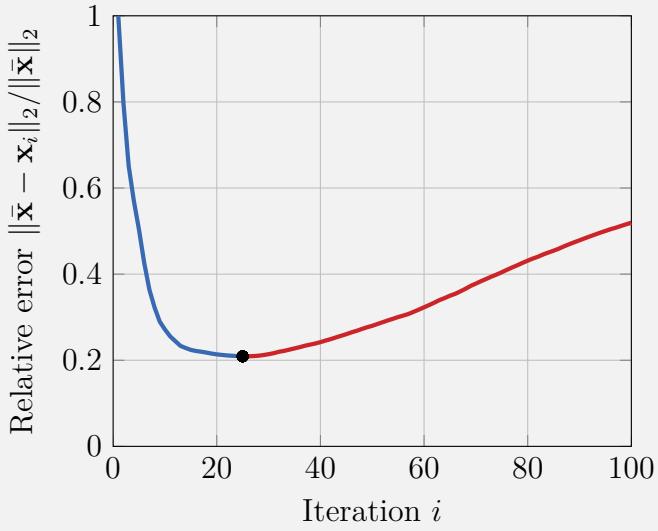


Figure 3.2: The convergence history of AB-GMRES for the 2D CT test problem. The dot illustrates the optimal solution $\mathbf{x}_{\text{opt}}$.

3.7 Restarted GMRES

So far, we have seen that the ABBA methods can find a minimizer to the unmatched normal equations, however, the ABBA methods can be computationally heavy to use. This is due to the Krylov subspace which in each iteration increase in size as it expands with $\mathbb{R}^{n\times 1}$ for each iteration (in the Algorithms 2 and 3 it is the expansion of the matrix $\mathbf{W}_k = \begin{bmatrix}\mathbf{w}_1 & \mathbf{w}_2 & \cdots & \mathbf{w}_k\end{bmatrix}$). Thus, after K iterations the Krylov subspace is of size $n \times K$ which is both expensive to do computations with but mostly it requires a lot of memory. Therefore, we introduce *restarted GMRES* [18], which aims to keep the dimension of the Krylov subspace low so that we can reduce the memory requirement of storing it.

Restarting the ABBA methods means that after p iterations, we will restart the algorithms with a new initial guess $\mathbf{x}_0^{(l+1)} = \mathbf{x}_p^{(l)}$ where l illustrate the period $k = 1, \ldots, p$. This yields that our Krylov subspace will at most be of size $\mathbb{R}^{n \times p}$. The pseudo codes are given in Algorithm 4 and 5. In Example 3.3 there is an illustration of the restart concept.

Algorithm 4 AB-GMRES(p)

Choose initial $\mathbf{x}_0$
$\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$
for $l = 1, 2, \ldots, L$ **do**
  $\mathbf{w}_1 = \mathbf{r}_0 / \|\mathbf{r}_0\|_2$
  for $k = 1, 2, \ldots, p$ **do**
    $\mathbf{q}_k = \mathbf{A}\mathbf{B}\mathbf{w}_k$
    for $i = 1, 2, \ldots, k$ **do**
      $h_{i,k} = \mathbf{q}_k^T \mathbf{w}_i$
      $\mathbf{q}_k = \mathbf{q}_k - h_{i,k}\mathbf{w}_i$
    end for
    $h_{k+1,k} = \|\mathbf{q}_k\|_2$
    $\mathbf{w}_{k+1} = \mathbf{q}_k / h_{k+1,k}$
    $\mathbf{y}_k = \text{argmin}_{\mathbf{y}} \|\|\mathbf{r_0}\|_2 \mathbf{e}_1 - \mathbf{H}_k \mathbf{y}\|_2$
    $\mathbf{x}_k = \mathbf{x}_0 + \mathbf{B}\left[\mathbf{w}_1, \mathbf{w}_2, \cdots, \mathbf{w}_k\right] \mathbf{y}_k$
    $\mathbf{r}_k = \mathbf{b} - \mathbf{A}\mathbf{x}_k$
    Stopping rule goes here
  end for
  $\mathbf{x}_0 = \mathbf{x}_k$
  $\mathbf{r}_0 = \mathbf{r}_k$
end for

Algorithm 5 BA-GMRES(p)

Choose initial $\mathbf{x}_0$
$\mathbf{res} = \mathbf{b} - \mathbf{A}\mathbf{x}_0$
for $l = 1, 2, \ldots, L$ **do**
  $\mathbf{r}_0 = \mathbf{B}(\mathbf{res})$
  $\mathbf{w}_1 = \mathbf{r}_0 / \|\mathbf{r}_0\|_2$
  for $k = 1, 2, \ldots, p$ **do**
    $\mathbf{q}_k = \mathbf{B}\mathbf{A}\mathbf{w}_k$
    for $i = 1, 2, \ldots, k$ **do**
      $h_{i,k} = \mathbf{q}_k^T \mathbf{w}_i$
      $\mathbf{q}_k = \mathbf{q}_k - h_{i,k}\mathbf{w}_i$
    end for
    $h_{k+1,k} = \|\mathbf{q}_k\|_2$
    $\mathbf{w}_{k+1} = \mathbf{q}_k / h_{k+1,k}$
    $\mathbf{y}_k = \text{argmin}_{\mathbf{y}} \|\|\mathbf{r_0}\|_2 \mathbf{e}_1 - \mathbf{H}_k \mathbf{y}\|_2$
    $\mathbf{x}_k = \mathbf{x}_0 + \left[\mathbf{w}_1, \mathbf{w}_2, \cdots, \mathbf{w}_k\right] \mathbf{y}_k$
    $\mathbf{r}_k = \mathbf{b} - \mathbf{A}\mathbf{x}_k$
    Stopping rule goes here
  end for
  $\mathbf{x}_0 = \mathbf{x}_k$
  $\mathbf{res} = \mathbf{r}_k$
end for

The convergence of AB-GMRES(p) and BA-GMRES(p) is not well known as there is no research that we know of that has tried restarting the ABBA methods for unmatched projector pairs. Hence, we will study the convergence history in Chapter 5. We will consider the convergence both as a function of iterations but also compare it to the computation time. In the following sections, we refer to AB/BA-GMRES without restart with the symbol ∞ (AB/BA-GMRES(∞)).

Example 3.3: Illustrating the concept of restarted GMRES
Figure 3.3 illustrates the concept of restarted GMRES. Having K iterations, we subdivide those into L equally spaced intervals depending on the choice of the restart parameter p ($L = K/p$). Then the ABBA methods are run on each of these sub-intervals $l = 1, \ldots, L$ using the previously found solution $\mathbf{x}_p^{(l)}$ as our new initial guess for $l + 1$ and for each of the sub-interval the Krylov subspace is reset. It is clear from both of the algorithms and Figure 3.3 that when we reach $l = L$ in the algorithms then we have taken as many iterations (K) as for the ABBA methods without restart denoted AB-GMRES(∞) and BA-GMRES(∞).

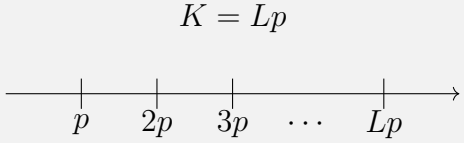


Figure 3.3: Illustrating the subdivision when using restarted GMRES. Instead of taking K iterations, we take p iterations L times. This leads to a reduced memory requirement for $\mathbf{W}_k$.

3.7.1 Computational Cost of AB/BA-GMRES(∞)

We will derive the number of floating-point operations (FLOPs) needed when using AB/BA-GMRES(∞). Due to inconsistencies in the literature, we state that the small s in FLOPs refers to it being in plural.

We start by considering AB-GMRES without restart, algorithm 2. We start by performing one matrix-vector (MV) product when computing $\mathbf{r_0}$. For each iteration k, we need to make two MV products when computing $\mathbf{q}_k$ in line 5. If we make a smart implementation, we can avoid one MV product when updating $\mathbf{x}_k$ in line 13. To do this, we simply save $\mathbf{B}\mathbf{w}_k$ from line 5 (when computing $\mathbf{q}_k$). Finally, if we need to compute the residuals $\mathbf{r}_k$ it requires one MV product in line 14. In total for all iterations $k = 1, \ldots, K$, we will compute $3K + 1 \sim 3K$ MV products. If we do not need to compute the residuals, we only need to compute $2K + 1 \sim 2K$ MV products. An overview of the MV products is shown in Appendix A.2.

To turn this into FLOPs, we consider Figure 3.4. The number of elements within $\mathbf{A}$ and $\mathbf{B}$ is mn. Each row in $\mathbf{A}$, represents an image of size $N \times N$. In Figure 3.4, we see an X-ray through that image illustrating that it can hit at most $2N - 1$ pixels. As the projectors are sparse, we obtain through experiments[1] that the number of non-zero elements within the row of $\mathbf{A}$ is on average N. Thus, the inner product between a row-vector from the sparse projector $\mathbf{A}$ and the full image $\mathbf{x}$ is on average $2N$. As we have

[1]Test: Construct $\mathbf{A}$. `Z = np.zeros(np.shape(A)), Z[A > 0] = 1, np.mean(np.sum(Z,1))`

m rows, computing one MV product costs

$$C_{\text{mv}} \leq m2N \text{ FLOPs.} \tag{3.31}$$

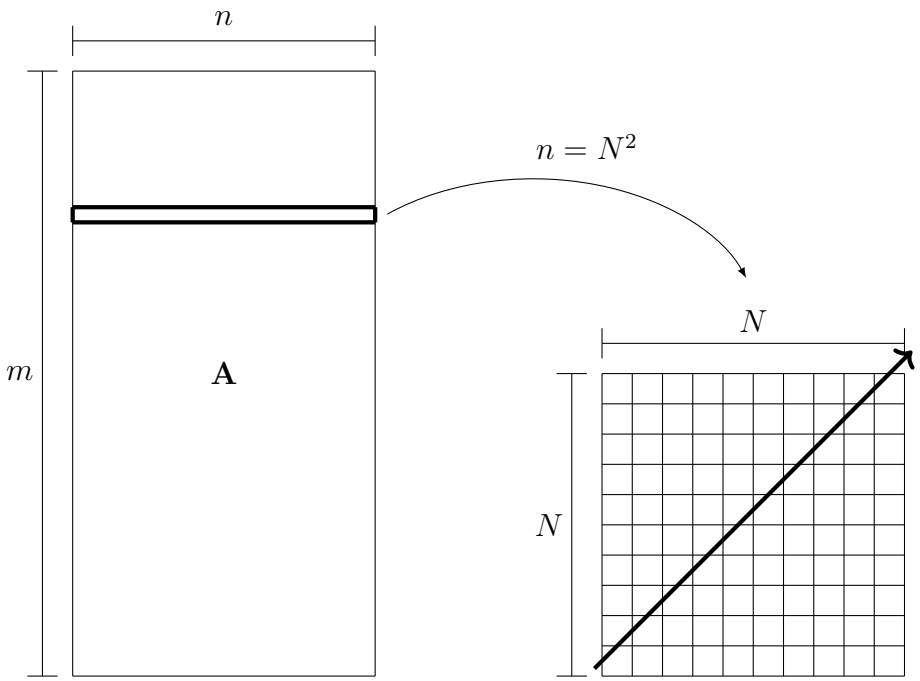


Figure 3.4: Extrapolating a row from **A** to show an example of how an X-ray moves across the image of interest. This X-ray touches as many pixels as possible.

The Gram-Schmidt orthogonalization in lines 7 and 8 requires combined $4n$ FLOPs. As it takes $2n$ FLOPs to compute $\mathbf{q}_k^T \mathbf{w}_i$ and $2n$ FLOPs to update $\mathbf{q}_k$. Thus, for each k'th iteration we obtain

$$C_{\text{orth}}(k) = k4n \text{ FLOPs.} \tag{3.32}$$

So running the AB-GMRES algorithm without restart, meaning $k = 1, \ldots, K$, the computations of the MV products will cost

$$3KC_{\text{mv}} = 3K2mN \text{ FLOPs or } 2KC_{\text{mv}} = 2K2mN \text{ FLOPs,} \tag{3.33}$$

depending on the computations of the residual. All Gram-Schmidt orthogonalizations combined will cost

$$\sum_{k=1}^{K} C_{\text{orth}}(k) = \sum_{k=1}^{K} k4n = 4n \sum_{k=1}^{K} k \approx 4n \frac{1}{2} K^2 = 2nK^2 \text{ FLOPs.} \tag{3.34}$$

Thus, AB-GMRES(∞) cost in total $3K2mN + 2nK^2$ FLOPs when computing $\mathbf{r}_k$ and $2K2mN + 2nK^2$ FLOPs if we do not need to compute $\mathbf{r}_k$. Note, we have disregarded the least squares problem in line 12 as the cost is relatively small compared to all other contributions.

The cost of BA-GMRES(∞), algorithm 3, is almost the same as for AB-GMRES(∞). The only difference is, that the residual $\mathbf{r}_0$ is computed differently. In BA-GMRES(∞), $\mathbf{r}_0 = \mathbf{B}(\mathbf{b} - \mathbf{A}\mathbf{x}_0)$ which means the cost is two MV products instead of one. Thus, the total number of MV products is $3K + 2 \sim 3K$ if we compute $\mathbf{r}_k$ in each iteration and $2K + 2 \sim 2K$ if we do not. Thus, BA-GMRES(∞) cost in total $3K2mN + 2nK^2$ FLOPs or $2K2mN + 2nK^2$ FLOPs, respectively. Thus, the cost is the same if we ignore the initial MV products.

3.7.2 Computational Cost of AB/BA-GMRES(p)

If we consider the ABBA methods with restart, we have that $K = Lp$ (see Figure 3.3). p illustrates the number of iterations taken before we restart the algorithm (referred to as one period) and L illustrates the number of periods. An overview of the MV products is shown in Appendix A.2.

Computing the number of FLOPs, we follow the same structure as for AB-GMRES(∞) and consider Algorithm 4. Before starting the first period, $\mathbf{r}_0$ is computed and it cost one MV. For each period l, we compute $2p$ MV products when computing $\mathbf{q}_k$ and p MV products if we compute the residual $\mathbf{r}_k$ for each iteration k. If we do not need to compute $\mathbf{r}_k$ in each iteration, we only need to make one MV product per period as it is used as the initial guess in the next period. Hence, when having L periods, the number of MV products are $1 + 3pL \sim 3K$ when computing $\mathbf{r}_k$ for each iteration and $1 + 2pL + L \sim 2K + L$ when only computing the residual when the period ends. The orthogonalization cost for each period $2np^2$ FLOPs. Thus, AB-GMRES(p) cost in total $(3K)2mN + 2np^2L = (3K)2mN + 2npK$ FLOPs (when computing $\mathbf{r}_k$ for each k) or $(2K + L)2mN + 2npK$ FLOPs (when computing $\mathbf{r}_k$ for only $k = p$). The gray box shows the total number of FLOPs for AB-GMRES(∞) and AB-GMRES(p). We have utilized that $p = KL^{-1}$ to reach derivations resembling each other as much as possible. The parenthesis illustrates the lowest cost (when only computing $\mathbf{r}_{k=p}$).

FLOP costs:
AB-GMRES(∞):

$$(3K)2mN + 2nK^2 \text{ FLOPs} \quad \Big((2K)2mN + 2nK^2 \text{ FLOPs}\Big) \tag{3.35}$$

AB-GMRES(p):

$$(3K)2mN + 2nK^2L^{-1} \text{ FLOPs} \quad \Big((2K + L)2mN + 2nK^2L^{-1} \text{ FLOPs}\Big) \tag{3.36}$$

The only difference between AB-GMRES(p) and BA-GMRES(p) when computing FLOPs is that BA-GMRES(p) makes one MV product more than AB-GMRES(p) for each period as the residual is multiplied with $\mathbf{B}$ ($\mathbf{r}_0 = \mathbf{B}(\mathbf{b} - \mathbf{A}\mathbf{x}_0)$). Thus, if we compute $\mathbf{r}_k$ in each iteration BA-GMRES(p) cost $(3K + L)2mN + 2nK^2L^{-1}$ FLOPs and if we only

compute $\mathbf{r}_{k=p}$ it cost $(2K + 2L)2mN + 2nK^2L^{-1}$ FLOPs. The results are also shown in the gray box where the parenthesis illustrates the lowest cost (when only computing $\mathbf{r}_{k=p}$).

FLOP costs:
BA-GMRES(∞):

$$(3K)2mN + 2nK^2 \text{ FLOPs} \quad \Big((2K)2mN + 2nK^2 \text{ FLOPs}\Big) \tag{3.37}$$

BA-GMRES(p):

$$(3K + L)2mN + 2nK^2L^{-1} \text{ FLOPs} \quad \Big((2K + 2L)2mN + 2nK^2L^{-1} \text{ FLOPs}\Big) \tag{3.38}$$

In conclusion, when using restart in BA-GMRES, we will compute L MV products more compared to not using restart when computing the residual in each iteration k and $2L$ MV products more if we consider the case where we only compute the residual at $k = p$. However, the number of FLOPs within the Gram-Schmidt orthogonalization is decreased with L^{-1} when using restart.

For AB-GMRES, restart also uses L^{-1} times fewer FLOPs in the orthogonalization step compared to not using restart. Considering the number of MV products, we see that there is no difference for AB-GMRES(∞) and AB-GMRES(p) if we compute the residual for each k while if we only compute it at $k = p$ then AB-GMRES(p) makes L MV products more compared to not using restart.

3.8 Stopping Criteria

As discussed previously, the ABBA methods will semi-converge and we would like to stop at the iteration where the solution $\mathbf{x}_i$ is closest to the true solution $\bar{\mathbf{x}}$. Note, that we have shifted notation of the iteration from k to i. We will now consider how to stop our algorithms. When working on real data, we do not know $\bar{\mathbf{x}}$ and, thus, we need to include some stopping criteria. We will in this thesis look at the *Discrepancy Principle* (DP) and the *Normalized Cumulative Periodogram* (NCP) as two choices for stopping the solvers.

3.8.1 Discrepancy Principle

The inverse problem we solve is

$$\mathbf{A}\bar{\mathbf{x}} = \mathbf{b} + \mathbf{e}. \tag{3.39}$$

We do this by minimizing $\|\mathbf{A}\mathbf{x}_i - \mathbf{b}\|_2$ and we would like to stop iterating when we are left with $\|\mathbf{e}\|_2$ on the right-hand side:

$$\|\mathbf{A}\mathbf{x}_i - \mathbf{b}\|_2 = \|\mathbf{e}\|_2. \tag{3.40}$$

Thus, minimizing $\|\mathbf{A}\mathbf{x}_i - \mathbf{b}\|_2$ leads to finding a solution $\mathbf{x}_i$ which satisfies $\|\mathbf{A}\mathbf{x}_i - \mathbf{b}\|_2 = 0$. As shown in (3.40), we need the least squares fit $\|\mathbf{A}\mathbf{x}_i - \mathbf{b}\|_2$ to equal the 2-norm of the noise and not zero. Thus, if we know what the noise level $\|\mathbf{e}\|_2$ is, then we can find $\mathbf{x}_i \approx \bar{\mathbf{x}}$ from (3.40). If we only have a noise level approximation, we use a safety factor $\nu_{\text{dp}} > 1$ to make our algorithms stop before they include noise. So the Discrepancy Principle [13, Ch. 5.2] is given as

$$\|\mathbf{A}\mathbf{x}_i - \mathbf{b}\|_2 = \nu_{\text{dp}}\|\mathbf{e}\|_2. \tag{3.41}$$

In many cases (3.41) cannot be satisfied exactly and we instead look at $\|\mathbf{A}\mathbf{x}_i - \mathbf{b}\|_2 \geq \nu_{\text{dp}}\|\mathbf{e}\|_2$. This means that we need to find the iteration i which satisfies

$$\|\mathbf{A}\mathbf{x}_i - \mathbf{b}\|_2 \geq \nu_{\text{dp}}\|\mathbf{e}\|_2 > \|\mathbf{A}\mathbf{x}_{i+1} - \mathbf{b}\|_2. \tag{3.42}$$

We will refer to the iteration satisfying (3.42) as i_{dp} which is easy to check in an iterative solver. A disadvantage of this method is that it relies on a good guess (or knowledge) of the noise level $\|\mathbf{e}\|_2$ (or η which is the standard deviation). If we do not have such information it is difficult to get good results from the DP. Thus, we also look at using the NCP criterion.

3.8.2 Normalized Cumulative Periodogram

Recall the Picard plot from Figure 2.17. As long as the Picard condition is satisfied, $\mathbf{u}_i^T\mathbf{b}$ will overall decay until the level off at η where the Picard condition is no longer satisfied. Thus, $\bar{\mathbf{b}}$ will be the dominating factor when the Picard condition is satisfied and we will converge toward the true solution. When we reach the level off, $\mathbf{e}$ begins to dominate. This information is utilized in the Normalized Cumulative Periodogram (NCP) [13, Ch. 5.5].

NCP and DP are closely related. In DP, we only utilized information of the noise level. With NCP, we want to answer the question: "*When can the residual vector be considered as noise?*" [13, Ch. 5.5, p. 98, l. 39–40]. We do not want to compute the SVD components nor make any visual inspections of the Picard plot. The approach for NCP is to consider $\bar{\mathbf{b}}$ as a "signal" which looks different than the noise $\mathbf{e}$ such that we can distinguish between satisfying the Picard condition and not satisfying it.

In the case where we have white Gaussian noise, we can use a discrete Fourier transform (DFT) to answer the question about when we can consider the residual vector as noise. Thus, we can write the DFT of the residual vector $\mathbf{r}_i$ as

$$\hat{\mathbf{r}}_i = \text{dft}(\mathbf{r}_i) = \left((\hat{\mathbf{r}}_i)_1, (\hat{\mathbf{r}}_i)_2, \ldots, (\hat{\mathbf{r}}_i)_m\right)^T \in \mathbb{C}^m. \tag{3.43}$$

To make fast computations, we use the fast Fourier transform (FFT) instead of DFT. We can then write the power spectrum of $\mathbf{r}_i$ as

$$\mathbf{p}_i = \left(|(\hat{\mathbf{r}}_i)_1|^2, |(\hat{\mathbf{r}}_i)_2|^2, \ldots, |(\hat{\mathbf{r}}_i)_{q+1}|^2\right)^T, \quad q = \lfloor m/2 \rfloor, \tag{3.44}$$

where $\mathbf{p}_i \in \mathbb{R}^{q+1}$. Furthermore, q is the largest integer satisfying $q \leq m/2$. The elements within $\mathbf{p}_i$ describe the power in the signal $\mathbf{r}_i$. The first element, $(\mathbf{p}_i)_1$, represents the lowest frequency (0th frequency) and $(\mathbf{p}_i)_{q+1}$ representing the highest frequency.

From the above, we can define NCP as

$$\mathbf{c}(\mathbf{r}_i)_j = \frac{(p_i)_2 + \ldots + (p_i)_{j+1}}{(p_i)_2 + \ldots + (p_i)_{q+1}}, \quad j = 1, \ldots, q. \tag{3.45}$$

We refer to $\mathbf{c}(\mathbf{r}_i)_j \in \mathbb{R}^q$ as the NCP vector and its elements include the cumulated sum of the power spectrum. $\mathbf{c}(\mathbf{r}_i)_j$ is normalized meaning the largest value of an element in $\mathbf{c}(\mathbf{r}_i)_q$ cannot exceed 1. Moreover, when computing the NCP the first component of $\mathbf{p}_i$ is not included. An NCP curve is defined by plotting $(j, \mathcal{E}(c(\mathbf{r}_i)_j))$ where $\mathcal{E}(\cdot)$ represent the expected value.

In the perfect setting, a residual vector consisting of only white Gaussian noise will have a flat power spectrum which means $\mathcal{E}((\mathbf{p}_i)_2) = \mathcal{E}((\mathbf{p}_i)_3) = \cdots = \mathcal{E}((\mathbf{p}_i)_q)$. Hence, the NCP curve will be a straight line going from $(0, 0)$ to $(q, 1)$. However, real noise does not have an ideal flat spectrum but we can still expect that the NCP is close to a straight line. In [13, Ch. 5.5] it is stated, that we can with a 5% significance level expect the NCP curve to lie within the Kolmogorov-Smirnoff limits $\pm 1.35q^{-1/2}$ of the straight line. This means that as long as our residual vector only includes white noise, we will stay within the Kolmogorov-Smirnoff limit and obtain solutions that comply with the discrete Picard condition. Unfortunately, real data often include other types of noise than only white Gaussian noise. Thus, achieving a straight line is not possible. So to make NCP more robust for such residual vectors, we look at which residual vectors entail the NCP to be closest to a straight line. This can be implemented by computing

$$d(i) = \|\mathbf{c}(\mathbf{r}_i) - \mathbf{c}_{\text{white}}\|_2, \tag{3.46}$$

where $\mathbf{c}_{\text{white}} = \left(1/q, 2/q, \ldots, 1\right)^T$. Choosing the $d(i)$ with the smallest value will give us the residual vector which is closest to the straight line. The iteration fulfilling this requirement will be denoted i_{NCP}.

To implement this, we have followed the same structure as in [14] which is a MATLAB implementation of NCP for CT problems that has been used in [16].

3.9 Summary

This chapter has discussed different choices of reconstruction models to suppress the influence of noise. As CT problems are large-scale sparse problems and often rank

defect, direct methods such as SVD, LU factorization, and Gaussian elimination are not desired for CT reconstructions.

Modern implementations of the CT problem involve fast matrix-vector products which are achieved with an unmatched projector pair, leading to us solving the unmatched normal equations. Unmatched normal equations can result in a minimization problem where the operator is not symmetric positive definite. This property ruins the convergence of many iterative methods and we, therefore, considered the ABBA iterative methods. The ABBA methods can handle projector pairs **AB** and **BA** that are neither symmetric nor positive semi-definite and they will converge [18].

We then considered improving the algorithms as GMRES is memory intensive and subject to semi-convergence. The Krylov subspace expands in each iteration and we, therefore, introduced restarted GMRES which yields resetting the Krylov subspace after every p'th iteration to keep the memory requirement lower. Applying restart to the ABBA methods with unmatched projector pairs is not well studied and, thus, we do not know how this will affect the convergence (see Chapter 5 for results).

Lastly, we discuss how to stop the ABBA methods when they reach semi-convergence. Two stopping rules are considered, namely, DP and NCP. Again, we do not have any theory on how they will behave to our problem. However, the advantage of DP is that it is very easy to implement but it also requires a good knowledge of the noise level. The advantage of NCP is that we do not have to know anything about our problem, we can just simply apply it.

CHAPTER 4

Constructing an Unmatched Projector Pair

The ABBA iterative methods need a forward and back projector. This thesis will investigate two open-source toolboxes for constructing these projector pairs. We will look at the *All Scales Tomographic Reconstruction Antwerp* (ASTRA) Toolbox [3, 29] and the *Tomographic Iterative GPU-based Reconstruction* (TIGRE) Toolbox [28, 6]. The presentation of those toolboxes in this thesis suffers from insufficient documentation.

ASTRA and TIGRE can be used to solve CT inverse problems as they include different reconstruction methods. However, we will only use their implementations of the forward and back projectors as building blocks for our ABBA methods. We will discuss more details about the software packages in this chapter. Furthermore, we will through simulated data compare the projection models from the two software packages to see if we obtain any new information.

ASTRA and TIGRE are public domains and we would like to also test the ABBA methods on a user domain. With the user domain, we will make it possible for the user to take their code for a forward and back projector instead of using public toolboxes which may be preferred in some studies. The user domain is provided by Professor Emil Y. Sidky from Chicago University. Finally, we will choose which projector pairs we will continue working with and we will study how unmatched they are.

The structure of the chapter follows:

- Introduction to the ASTRA toolbox.
- Introduction to the TIGRE toolbox.
- Comparison of the two public domain toolboxes.
- Introduction to the user domain.
- Study the difference within the projector pairs.

4.1 The Setup

This thesis will focus on 2D parallel beam and fan beam CT problems. We will both work with CPU and GPU implementations. All our work is performed on a desktop computer with a Windows 11 × 64 operating system. The specifications are given in Table 4.1.

Computer Specifications	
Memory:	16 GB, 2933 MHz, DDR4
GPU:	NVIDIA GeForce RTX 2080 @ 1.7 GHz
CPU:	Intel(R) Core(TM) i9-9900K CPU @ 3.60GHz
	- Number of cores: 8
	- With hyperthreading: 16

Table 4.1: The relevant computer specifications about the desktop computer used for this thesis.

We have implemented the reconstruction methods AB-GMRES(∞) from Algorithm 2 and BA-GMRES(∞) from Algorithm 3. The methods need a forward projector, a back projector, a sinogram, and the initial condition of the image which we choose to be $\mathbf{x}_0 = \mathbf{0}$. The algorithms can handle matrix-free forward and back projectors.

We will investigate the projectors in the public domains and based on the results, we will choose a public domain to continue working with. The study is based on comparing the convergence history obtained with the matrices $\mathbf{A}$ and $\mathbf{B}$ from the different software packages and seeing how equivalent they are. Finally, we will also look at how unmatched the projector pairs are for the chosen software domains.

4.2 ASTRA

ASTRA is a tomographic reconstruction toolbox with both a Python and a MATLAB interface. ASTRA supports both 2D parallel and fan beam geometries and 3D parallel and cone beam geometries. There are several reconstruction methods within the toolbox such as FBP, SIRT, SART, and CGLS, however, we will not consider these in this report as we focus on the ABBA methods.

According to [29], ASTRA has based most of the code on C++ and the core parts are handled on a GPU card through the NVIDIA CUDA language to make the algorithms most efficient. However, ASTRA also makes it possible to use a CPU for the computations. Not only does ASTRA provide fast implementations of the software, but it also has a modular design meaning that, e.g., projector pairs can be used independently which we utilize in this project. We will use ASTRA to create our forward and back projectors. Another advantage of ASTRA is that it not only provides a matrix implementation of the forward projectors but also a matrix-free implementation of both the forward and back projectors.

ASTRA provides the three forward projection models discussed in Chapter 2, namely the line, strip, and Joseph's model. All three models are only available when the CPU is used and only for the parallel beam geometry. If we have fan beam geometry and work on the CPU then the forward projection can be either modeled by the line or strip but not Joseph. The GPU implementation only has the Joseph forward projector. Table 4.2 shows an overview of the forward projectors within ASTRA.

	Line	Strip	Joseph
	Parallel beam		
ASTRA, CPU	✓	✓	✓
ASTRA, GPU	✗	✗	✓
	Fan beam		
ASTRA, CPU	✓	✓	✗
ASTRA, GPU	✗	✗	✓

Table 4.2: The forward projection models in ASTRA.

The documentation of the ASTRA toolbox [4] is limited, but we have been in contact with Dr. Willem Jan Palenstijn from CWI in Amsterdam who is the main programmer of ASTRA. The GPU version of ASTRA uses the back projection model described in Section 2.3.4 for the parallel beam geometry while for the fan beam geometry, it is unclear what has been implemented and there are no known reference[2]. If we use the CPU version of ASTRA, the back projector is the exact transpose of $\mathbf{A}$ meaning $\mathbf{B} = \mathbf{A}^T$. This has been verified in our study. Table 4.3 shows an overview of the back projection depending on the working space and the geometry. Note, the back projector is only given as a matrix-free implementation.

	Parallel beam	Fan beam
ASTRA, CPU	$\mathbf{B} = \mathbf{A}^T$	$\mathbf{B} = \mathbf{A}^T$
ASTRA, GPU	From Section 2.3.4	Not documented[2]

Table 4.3: The back projection operators in ASTRA.

4.3 TIGRE

The tomographic reconstruction toolbox TIGRE consists of fast reconstruction algorithms and projection models for 3D X-ray problems. TIGRE is only 3D-based and, thus, only includes parallel and cone beam geometries. The toolbox is based on CUDA

[2]Dr. Willem Jan Palenstijn states in a mail: "*It is (mostly) exactly the same model as for the parallel beam case. A ray is traced from the source through the center of the voxel to the detector, and linear interpolation is used to interpolate the sinogram values at the point the ray hits. The difference is that a weighting factor is added to compensate for the fact that the ray density varies depending on the magnification factor of the pixel/projection.*"

(meaning only GPU-based) which, therefore, forces the user to use single precision floating points. It has both a MATLAB and a Python interface. The documentation of TIGRE is limited, however, on GitHub they refer to the Ph.D. thesis [5] for a more detailed description.

TIGRE provides the forward projection models line and Joseph. The line model is according to [5] implemented from [11] which according to them is an improvement of the line model. In Section 2.3.4, we discuss Joseph's model with a linear interpolation which is also what they use in TIGRE except for one condition: according to [5, p. 59, l. 3–5], *"the implementation in TIGRE additionally blocks the user from choosing a sampling rate larger than a voxel by limiting it to that size and changing the interpolation to nearest neighbour"*. In Tigre, both the line and Joseph's forward projections are possible no matter the geometry. An overview of the forward projection models in TIGRE is given in Table 4.4.

	Line	Strip	Joseph
	Parallel beam		
TIGRE, GPU	✓	✗	✓
	Cone beam		
TIGRE, GPU	✓	✗	✓

Table 4.4: The forward projection models in TIGRE.

TIGRE includes a matched back projector and a FDK back projector. We will only consider the unmatched FDK back projector which is described in [5] (only for the cone beam geometry). Unfortunately, it is not mathematically clear from the thesis what is exactly implemented so we contacted Dr. Ander Biguri (the author of [5]) but no additional mathematical expression where given[3]. However, he informs us that the back projection model used for parallel beam is the same as the one discussed in Section 2.3.4. The back projection models (considered in this thesis) are summarized in Table 4.5.

	Parallel beam	Fan beam
TIGRE, GPU	From Section 2.3.4	Not documented[3]

Table 4.5: The back projection operators in TIGRE.

4.4 Comparing the Public Toolboxes: ASTRA and TIGRE

We have now looked at two public domains for obtaining our forward and back projector for the ABBA methods. We will in this section compare the implementations of both the

[3]Dr. Ander Biguri states in a mail: "*It's the same as in Section 2.3.4, but the weight of the value put into each voxel now can not be just $2\pi/N$, it requires some weight that accounts for the cone/fan angle. Then Section 4.4 of the thesis [5] explains what those weights are.*"

forward and back projections and based on the results we will choose one to focus on in the future studies of this report. Moreover, we will demonstrate that our implementation of the ABBA methods works for different choices of unmatched projector pairs. In this section, we work with AB-GMRES(∞) and BA-GMRES(∞).

In TIGRE, we use the setup shown in Table 4.6. In ASTRA, we only need to specify a setup when we use the fan beam geometry and this is also shown in Table 4.6. The distance in ASTRA between the center of rotation and the detector array must be zero to obtain matching setups and, thereby, matching sinograms. Physically, it does not make sense to have zero distance, but mathematically it is possible as it is just a scaling factor when we increase the distance (a more detailed explanation is found in Appendix A.3).

TIGRE	
Distance to source-detector in mm	1000
Distance to source origin in mm	1000
Variable to define the accuracy of 'interpolated' projection	0.5
ASTRA	
Distance between the center of rotation and the detector array	0
Distance between the source and the center of rotation	1000

Table 4.6: Setup for TIGRE and ASTRA (only need to specify when using fan beam geometry).

The detectors in the two software packages do not have the same starting point and, therefore, we add $\pi/2$ to the angles in TIGRE. A final important thing is that the sinograms computed from TIGRE and ASTRA will be reversed in the x-axis when comparing the sinograms. Thus, to compare the sinograms one of them must be reversed in the x-axis.

TIGRE is based on 3D CT problems but we will use it on a 2D CT problem. Thus, the setup for the image is $\mathbb{R}^{1\times N\times N}$ and for the detector we have $\mathbb{R}^{1\times N_s}$. Furthermore, we set the pixel width in both ASTRA and TIGRE to one. We will test the toolboxes on the Shepp Logan phantom (see Figure 2.12 to the left) of size 128×128. For parallel beam geometries, we have that $\theta \in [0, \pi]$ and the number of angles is 181. When considering fan beam geometry, we have 181 angles within $\theta \in [0, 2\pi]$.

In the next sections, we will compare the forward and back projectors in ASTRA and TIGRE. The projectors are matrix-free, but we will construct the matrices $\mathbf{A}$ and $\mathbf{B}$ as it will make it easier to compare. The matrices $\mathbf{A}$ and $\mathbf{B}$ can be constructed one column at a time. We insert a zero vector $\mathbf{x}$ with only one element being one. The element being one is the column we will create from the forward and back projection functions.

4.4.1 Forward Projection

This section investigates the forward projectors given in the toolboxes. The aim is to see if the implementation for the same operator gives the same result or if we gain more information from one of the toolboxes.

Considering Table 4.7, we see that ASTRA on a CPU and TIGRE have two models in common, namely the line and Joseph's models while ASTRA GPU only has Joseph's model in common with TIGRE. To obtain a comparable $\mathbf{A}$, we must flip the entries in the columns up/down (reverse in x-direction) and flip the entries in the rows right/left (reverse in y direction) for the forward projector generated in TIGRE such that the geometries are the same. This is due to ASTRA and TIGRE not using the same setup.

	Line	Strip	Joseph
	Parallel beam		
ASTRA, CPU	✓	✓	✓
ASTRA, GPU	✗	✗	✓
TIGRE, GPU	✓	✗	✓
	Fan beam		
ASTRA, CPU	✓	✓	✗
ASTRA, GPU	✗	✗	✓
TIGRE, GPU	✓	✗	✓

Table 4.7: An overview of the forward projection models in the toolboxes for parallel and fan beam geometries.

The line model:
It is only ASTRA CPU that has the line model implemented and, thus, we must compare TIGRE GPU with ASTRA CPU. To compare how equivalent the line model implementations are, we can construct the matrices $\mathbf{A}_{A_{\text{CPU}},\text{l}}$ for ASTRA's line model and $\mathbf{A}_{T,\text{l}}$ for TIGRES's line model. Moreover, we will add a subscript p illustrating parallel beam geometry and f for fan beam geometry. The relative error can then be computed as

$$\frac{\|\mathbf{A}_{\text{p},A_{\text{CPU}},\text{l}} - \mathbf{A}_{\text{p},T,\text{l}}\|_F}{\|\mathbf{A}_{\text{p},A_{\text{CPU}},\text{l}}\|_F} = 0.0012, \tag{4.1}$$

and

$$\frac{\|\mathbf{A}_{\text{f},A_{\text{CPU}},\text{l}} - \mathbf{A}_{\text{f},T,\text{l}}\|_F}{\|\mathbf{A}_{\text{f},A_{\text{CPU}},\text{l}}\|_F} = 0.0075. \tag{4.2}$$

We see for both parallel and fan beam that the line models from ASTRA and TIGRE are approximately the same ($\mathbf{A}_{\text{p},A_{\text{CPU}},\text{l}} \approx \mathbf{A}_{\text{p},T,\text{l}}$ and $\mathbf{A}_{\text{f},A_{\text{CPU}},\text{l}} \approx \mathbf{A}_{\text{f},T,\text{l}}$) as the relative error is less than 1%. Figure 4.1 shows the structure of the matrices $\mathbf{A}_{\text{p},A_{\text{CPU}},\text{l}}$ (to the left) and $\mathbf{A}_{\text{p},T,\text{l}}$ (to the right) which also looks similar as the results imply.

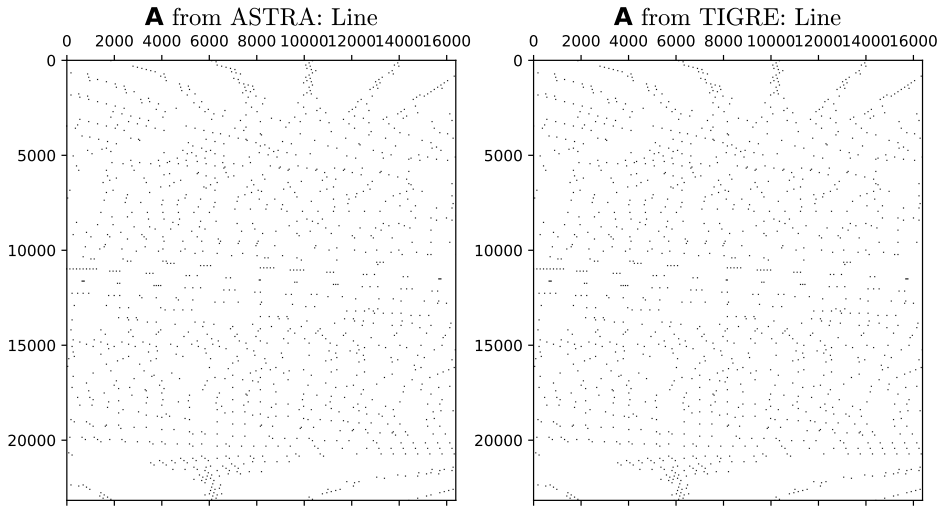


Figure 4.1: The structure of the forward projection matrix $\mathbf{A}$ when using the line model in ASTRA and TIGRE.

Figure 4.2 shows the true sinogram $\bar{\mathbf{b}}$ (without noise) for parallel beam geometry constructed from TIGRE and ASTRA with the line model. The sinogram from TIGRE has been reversed in the x-axis such that they have the same setup. The relative error between these sinograms are

$$\frac{\|\bar{\mathbf{b}}_{\mathrm{A_{p,CPU,l}}} - \bar{\mathbf{b}}_{\mathrm{p,T,l}}\|_2}{\|\bar{\mathbf{b}}_{\mathrm{A_{p,CPU,l}}}\|_2} = 7.4074 \cdot 10^{-5}. \tag{4.3}$$

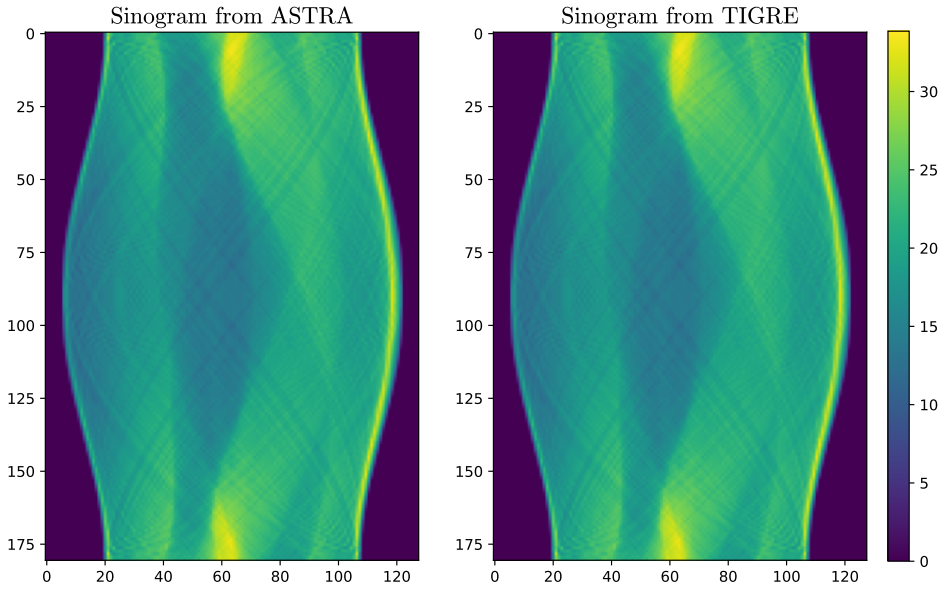


Figure 4.2: A noise-free sinogram from ASTRA (left) and TIGRE (right) with parallel beam geometry.

From (4.3), we see that the two sinograms are approximately the same which is also visually confirmed when looking at Figure 4.2. Furthermore, this was also the conclusion from (4.1) and (4.2).

The Joseph model:
As ASTRA has both a CPU and a GPU version of Joseph's model, we will compare both implementations with TIGRE's GPU implementation. If we consider the relative errors, we obtain

$$\frac{\|\mathbf{A}_{\mathrm{p},A_{\mathrm{CPU}},\mathrm{J}} - \mathbf{A}_{\mathrm{p},T,\mathrm{J}}\|_F}{\|\mathbf{A}_{\mathrm{p},A_{\mathrm{CPU}},\mathrm{J}}\|_F} = 0.7098, \tag{4.4}$$

and

$$\frac{\|\mathbf{A}_{\mathrm{p},A_{\mathrm{GPU}},\mathrm{J}} - \mathbf{A}_{\mathrm{p},T,\mathrm{J}}\|_F}{\|\mathbf{A}_{\mathrm{p},A_{\mathrm{GPU}},\mathrm{J}}\|_F} = 0.7098. \tag{4.5}$$

The relative errors for ASTRA CPU and GPU compared to TIGRE are the same. Thus, ASTRA's implementation on CPU and GPU gives the same result. From (4.4) and (4.5), we see that $\mathbf{A}_{\mathrm{p},A,\mathrm{J}} \neq \mathbf{A}_{\mathrm{p},T,\mathrm{J}}$ as the relative error is approximately 71%.

The GPU version of ASTRA has also Joseph's model for fan beam geometries and the relative error is

$$\frac{\|\mathbf{A}_{\mathrm{f},A_{\mathrm{GPU}},\mathrm{J}} - \mathbf{A}_{\mathrm{f},T,\mathrm{J}}\|_F}{\|\mathbf{A}_{\mathrm{f},A_{\mathrm{GPU}},\mathrm{J}}\|_F} = 0.7018, \tag{4.6}$$

which is also a very big difference. It seems odd that what should be the same models are so far from each other. We, therefore, investigate this a bit further by again looking at the matrices to see if we can spot the differences.

Figure 4.3 shows the matrices $\mathbf{A}_{\mathrm{p},A_{\mathrm{GPU}},\mathrm{J}}$ and $\mathbf{A}_{\mathrm{p},T,\mathrm{J}}$. We see that TIGRE is missing some of the structure in the upper left and lower right corners compared to ASTRA. Multiple experiments have been made with different choices of parameters. All experiments end up with TIGRE missing some parts in the forward projector compared to ASTRA. Figure 4.4 and 4.5 show two rows from the matrices illustrating that the X-ray is not passing through the entire image in TIGRE as it does in ASTRA. Thus, it seems there is a bug in TIGRE's code.

Conclusion:
From our analysis of the forward projectors, we can conclude that the line model is approximately the same for ASTRA and TIGRE as it varies very little (less than 1%). Joseph's model on the other hand does not give the same results for the two software packages as they differ by approximately 70%. The difference appears in the structure of $\mathbf{A}$. It seems some part is missing in the upper left and lower right corners of $\mathbf{A}$ constructed with TIGRE compared to $\mathbf{A}$ from ASTRA.

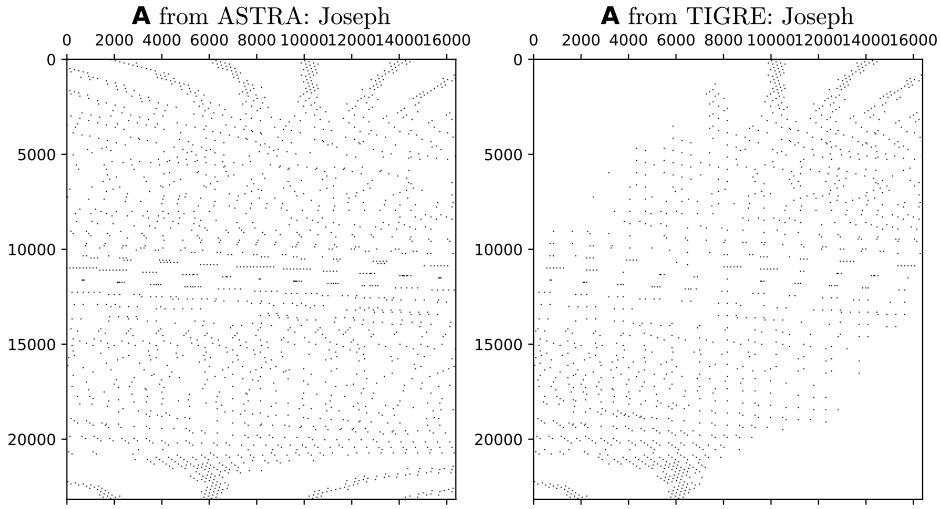


Figure 4.3: The structure of the forward projection matrix $\mathbf{A}$ when using the Joseph model in ASTRA GPU (left) and TIGRE (right).

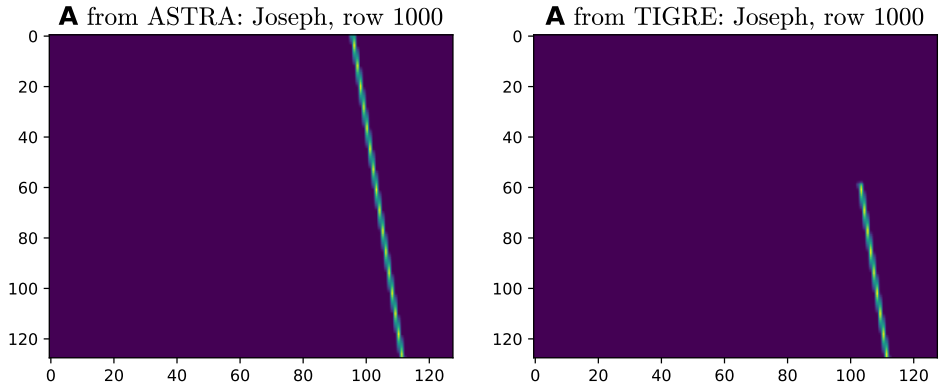


Figure 4.4: Illustrating the X-ray moving throw the image from row 1000 of $\mathbf{A}_{\mathrm{p},A_{\mathrm{GPU}},\mathrm{J}}$ and $\mathbf{A}_{\mathrm{p},T,\mathrm{J}}$ from Figure 4.3.

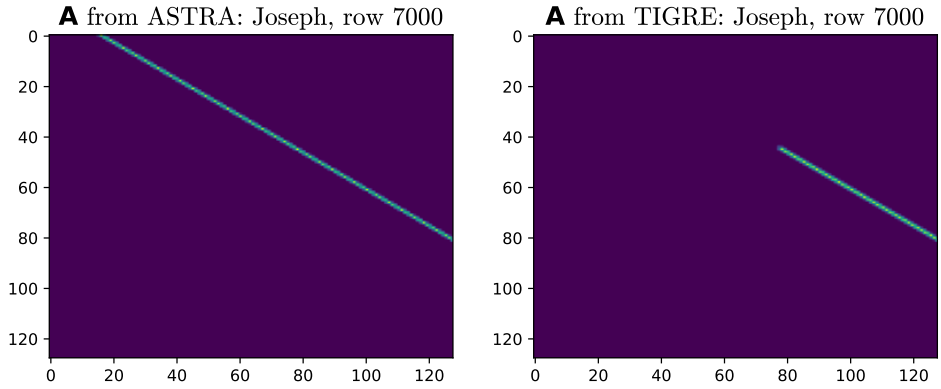


Figure 4.5: Illustrating the X-ray moving throw the image from row 7000 of $\mathbf{A}_{\mathrm{p},A_{\mathrm{GPU}},\mathrm{J}}$ and $\mathbf{A}_{\mathrm{p},T,\mathrm{J}}$ from Figure 4.3

4.4.2 Back Projection

In this section, we will compare the back projectors in ASTRA and TIGRE. An overview of the back projectors in the software packages is shown in Table 4.8. The back projection is dependent on the choice of a forward projector.

The back projection operators are compared by computing the matrix $\mathbf{B}$ from the matrix-free implementations of the back projector in ASTRA and TIGRE. To obtain a comparable $\mathbf{B}$, we must reverse both in x- and y-direction in TIGRE's back projector such that the geometries are the same (just as we did for the forward projectors).

	Parallel beam	Fan beam
ASTRA, CPU	$\mathbf{B} = \mathbf{A}^T$	$\mathbf{B} = \mathbf{A}^T$
ASTRA, GPU	From Section 2.3.4	Not documented
TIGRE, GPU	From Section 2.3.4	Not documented

Table 4.8: An overview of the back projection operators in the toolboxes for parallel and fan beam geometries.

The back projector is dependent on the choice of the forward projector. Thus, we will consider the different options and compute the relative errors. The forward projectors had the same scaling but this is not the case for the back projectors. Therefore, we look at the normalized relative error computed by

$$\text{relative error} = \frac{\left\| \frac{\mathbf{B}_A}{\|\mathbf{B}_A\|_F} - \frac{\mathbf{B}_T}{\|\mathbf{B}_T\|_F} \right\|_F}{\left\| \frac{\mathbf{B}_A}{\|\mathbf{B}_A\|_F} \right\|_F}, \tag{4.7}$$

and the different options for creating back projectors are shown in Table 4.9.

	Parallel beam	Fan beam
Line (CPU)	0.3051	0.3126
Joseph (CPU)	0.1318	×
Joseph (GPU)	0.0049	0.0380

Table 4.9: The normalized relative error (4.7) between the back projections computed from TIGRE and ASTRA (CPU and GPU) for different choices of the forward projector. The CPU and GPU marks are based on ASTRA as TIGRE always uses GPU and also only creates an inexact transpose.

From Table 4.9, we can conclude that the back projectors are very similar (a relative error between 0.5 to 3.8%) when we use the GPU version of ASTRA. Meaning, we compare an inexact transpose of the forward projector from ASTRA with the inexact transpose from TIGRE. For the two CPU-based back projectors, we obtain a relative error between 13 to 31% but we also compare an exact transpose from ASTRA with an inexact transpose from TIGRE.

4.4.3 The Line Model from ASTRA and TIGRE

We will now use the projector pairs constructed from ASTRA and TIGRE in AB-GMRES(∞) and BA-GMRES(∞) and study their convergence histories. Note, that only the CPU-based version of ASTRA can produce a line model. Thus, we will solve our CT problem with a matched projector pair when using ASTRA CPU and an unmatched projector pair when using TIGRE.

The sinograms provided for the ABBA methods are the ones created from the same toolboxes as the projector pairs come from (see Figure 4.2). To simulate noisy data, 0.03 relative Gaussian white noise is added to the sinograms before handing them to the solvers.

Figure 4.6 shows the convergence history for BA-GMRES(∞) when we have parallel beam geometry and we use the line model as the forward projector. Figure 4.7 shows the same but for fan beam geometry instead of parallel beam geometry. From both figures, we see the same overall behavior for the convergence and we see semi-convergence. ASTRA CPU reaches the optimum at iteration 16 and TIGRE reaches the optimum at iteration 17 with approximately the same relative error (19 to 20%).

The same overall behavior is obtained with AB-GMRES(∞). The only difference is that the optimum is at iteration 14 for ASTRA CPU and 15 for TIGRE, see Appendix B.1 Figure B.1 and B.2.

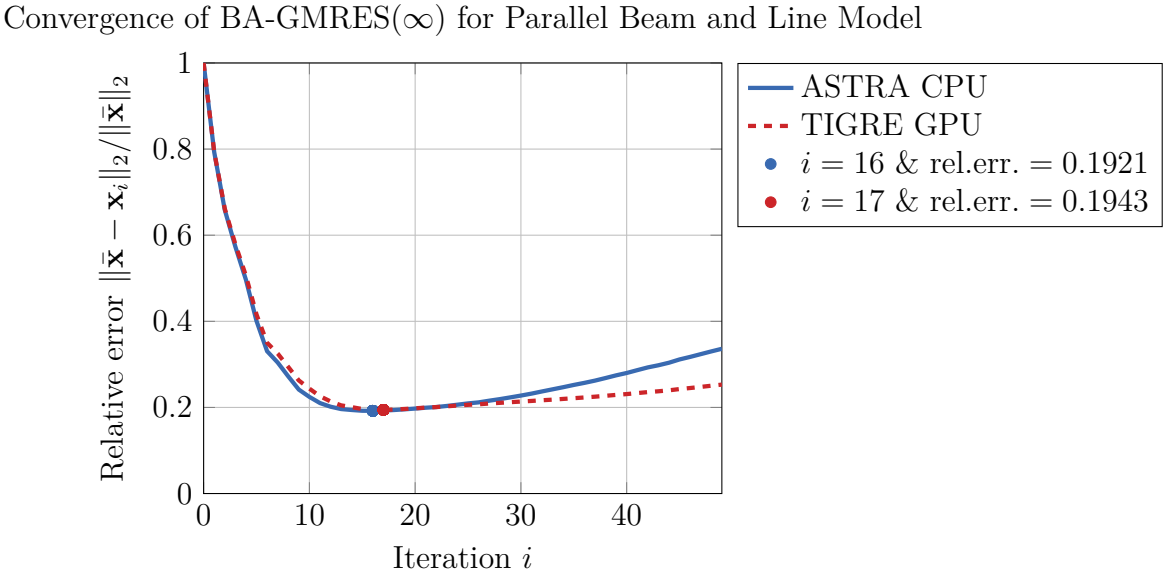


Figure 4.6: The convergence history of BA-GMRES(∞) for parallel beam geometry and with the forward projector being the line model. ASTRA CPU creates a matched projector pair while TIGRE creates an unmatched projector pair.

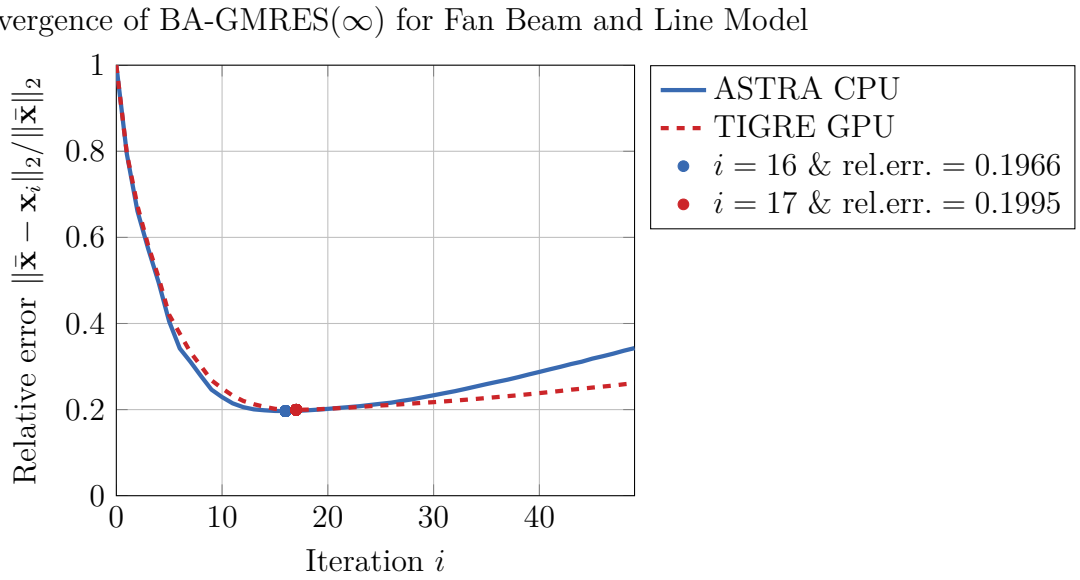


Figure 4.7: The convergence history of BA-GMRES(∞) for fan beam geometry and with the forward projector being the line model. ASTRA CPU creates a matched projector pair while TIGRE creates an unmatched projector pair.

4.4.4 The Joseph Model from ASTRA and TIGRE

We will now consider the Joseph model as the forward projector. Here we can obtain an unmatched projector pair from ASTRA GPU and, thus, compare the convergence history when both ASTRA and TIGRE provide an unmatched projector pair.

We consider the case with parallel beam geometry. The convergence history for BA-GMRES(∞) with only unmatched projector pairs is shown in Figure 4.8. We see that ASTRA GPU obtains semi-convergence and reach the optimum at iteration 17 with a relative error of 20% while TIGRE does not obtain semi-convergence. The convergence curve for TIGRE diverges. The same behavior is obtained for AB-GMRES(∞) (see Appendix B.1 Figure B.3).

Based on the comparison of the forward projectors and the convergence histories, we can conclude, that something is wrong with the implementation of Joseph's model in TIGRE. Moreover, based on our studies, we can conclude that the divergence obtained with TIGRE's Joseph's model does not depend on the geometry. We, however, do experience semi-convergence with the ABBA methods for the unmatched projector pair from ASTRA GPU.

4.4.5 Conclusion

From our studies of ASTRA and TIGRE, we can conclude that ASTRA and TIGRE produce approximately the same line forward projector and, thus, we do not gain any

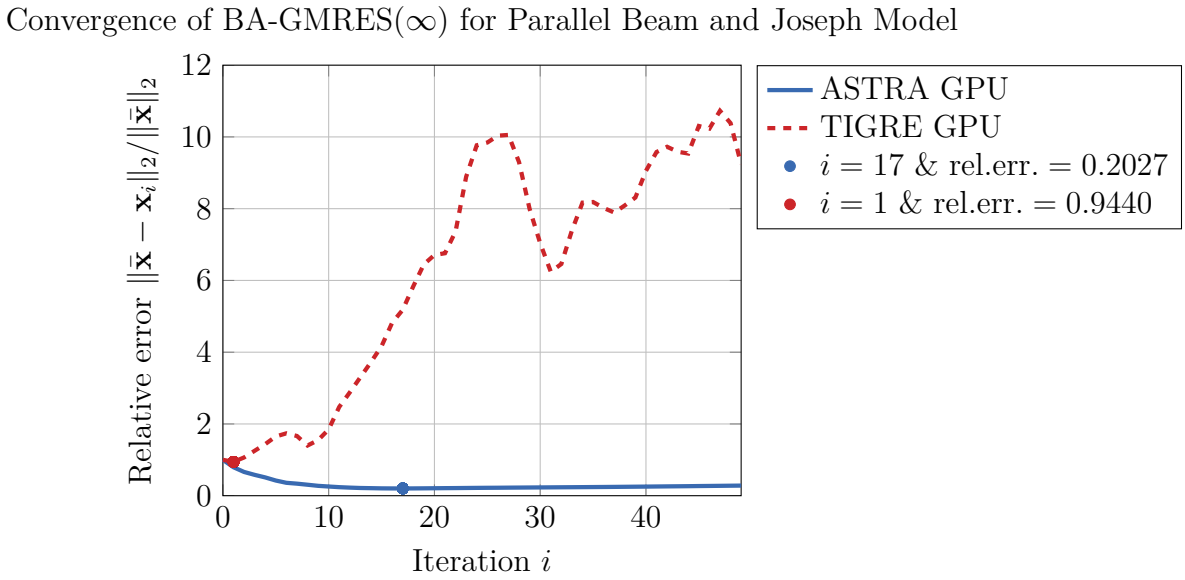


Figure 4.8: The convergence history of BA-GMRES(∞) for parallel beam geometry and with the forward projector being Joseph's model. ASTRA GPU creates an unmatched projector pair and so does TIGRE.

new information from one of the software packages. For the Joseph forward projector, we see that ASTRA obtains semi-convergence as expected while TIGRE does not. The structure of the Joseph forward projector in TIGRE does not look like in ASTRA as there are some missing parts, see Figure 4.4 and 4.5, that we think originates from a bug in TIGRE's code.

Based on the results from the analysis, we choose to disregard the TIGRE toolbox and only focus on ASTRA and the user domain. Furthermore, we note that ASTRA only includes one unmatched projector pair. It is obtained with ASTRA GPU and it is with Joseph's model as the forward projector.

4.5 Projector Pairs from User Domain

We will, in this section, consider using the ABBA methods given a user domain from Professor Emil Y. Sidky. The user domain provides some Python code that can both produce forward and back projection operators. The code is CPU-based and the software is for 2D CT problems.

The user domain provides a line and a strip model for the forward projector for both parallel and fan beam geometries. Table 4.10 gives an overview. Note that the strip model in the user domain is an approximation to the strip model discussed in Section 2.3.2.

	Line	Strip	Joseph
	Parallel beam		
Emil, CPU	✓	✓	✗
	Fan beam		
Emil, CPU	✓	✓	✗

Table 4.10: The forward projection models in the user domain.

Figure 4.9 illustrates how the code approximates the strip model. To the left, an example of the strip model is shown. Here we compute the area of the red triangle for the chosen pixel as discussed in Section 2.3.2. To the right, the approximation of the strip model given in the user domain is shown. Here we choose a number of X-rays between the two main X-rays (marked with thick lines). Adding the length of each ray within the chosen pixel will approximate the area computed in the strip model. Increasing the number of rays between main rays leads to better approximations of the area. However, this also means an increase in memory and computational cost.

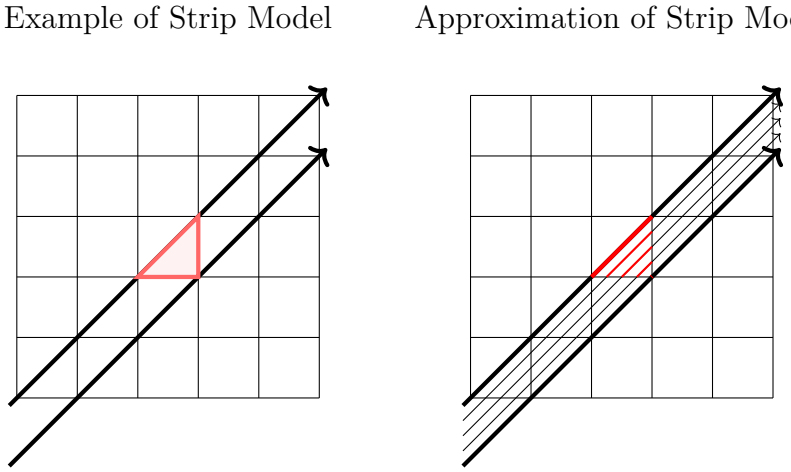


Figure 4.9: An example of the strip model to the left and an approximation of the strip model to the right.

The user domain provides two back projectors shown in Table 4.11. For parallel beam geometry, the user domain only has the matched transpose while for fan beam geometries there are both the exact transpose and an inexact transpose. The inexact transpose is the back projector described in Section 2.3.4 but for fan beam geometries (which we have no documentation for). In the CT community, the inexact transpose is also referred to as the pixel-driven back projector.

	Parallel beam	Fan beam
Emil, CPU	$\mathbf{B} = \mathbf{A}^T$	$\mathbf{B} = \mathbf{A}^T$ Pixel-driven (not documented)

Table 4.11: The back projection operators in the user domain.

4.5.1 Setup

We need to provide some information about our problem to the user domain. First, we must state the dimensionality of the output image $\mathbf{x}$ which include the length of the image sides in centimeter and the number of pixels in both directions. Furthermore, when using the forward projector model strip, we must state a number of rays called $nrays\_per\_strip$ in the user domain.

We have, so far, been working with the Shepp Logan phantom of size 128×128 which we will continue with. Thus, the number of pixels in both directions is $nx = ny = 128$. We have until now, in ASTRA, worked with the length of each pixel being 1, thus, the length of the sides will be $ximageside = yimageside = 1.28$ cm. Furthermore, we must inform the distance between the source and detector which we set to 100 cm and the radius being 50 cm. Finally, we must state the number of view angles and the number of bins. We will use the same values as previously, namely $nviews = 181$ and $nbins = 128$.

In the next sections, the convergence history for different projector pairs will be investigated. We only consider the fan beam geometry as this is the only geometry in the user domain with unmatched projector pairs.

4.5.2 The Line Model

We consider using the line model as the forward projector and the pixel-driven back projector from the user domain. Figure 4.10 shows the convergence history for BA-GMRES(∞). We have semi-convergence and reach the optimum at iteration 18 with a relative error of approximately 19%. In Appendix B.2 Figure B.4 the convergence history for AB-GMRES(∞) is shown. The convergence behavior is the same as for BA-GMRES(∞) with the optimal iteration being 16 with a relative error of $\sim$19%. In conclusion, the ABBA methods can solve the CT problem with the unmatched projector pair constructed with the line model as the forward projector and with the pixel-driven back projector from the user domain.

4.5.3 The Strip Model

Considering the strip model with 10 rays per strip for the forward projector, the convergence history of BA-GMRES(∞) looks as shown in Figure 4.11. We see semi-convergence and reach the optimum at iteration 14 with a relative error of $\sim$24% (similar behavior is seen for AB-GMRES(∞) in Appendix B.2 Figure B.5). Increasing the number of rays per strip (we tried 50 rays per strip) has shown not to affect the convergence history nor the relative error.

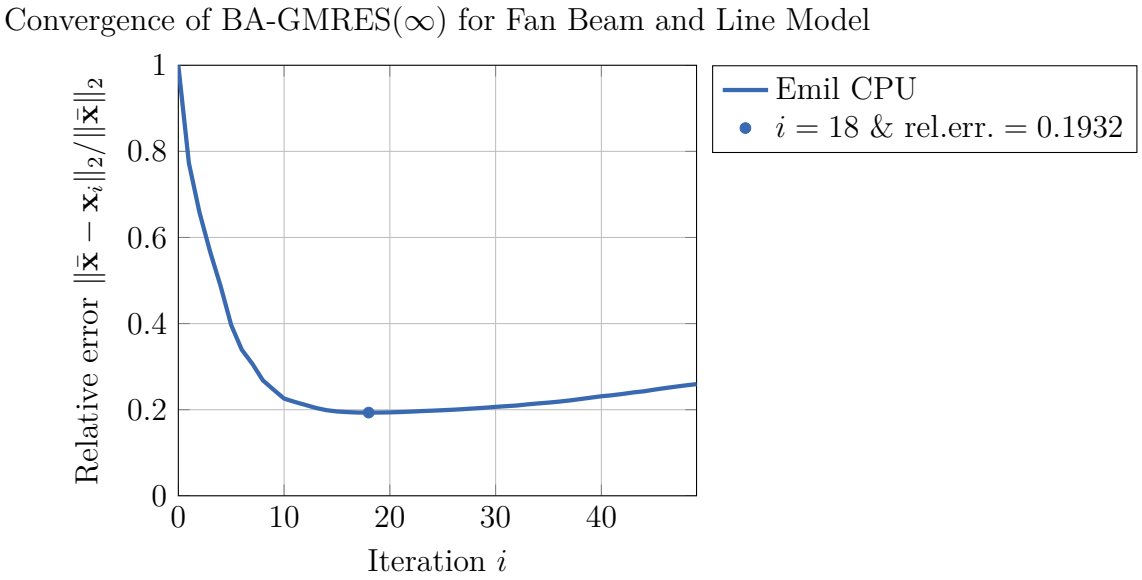


Figure 4.10: The convergence history of BA-GMRES(∞) for fan beam geometry when using the line model as the forward projector.

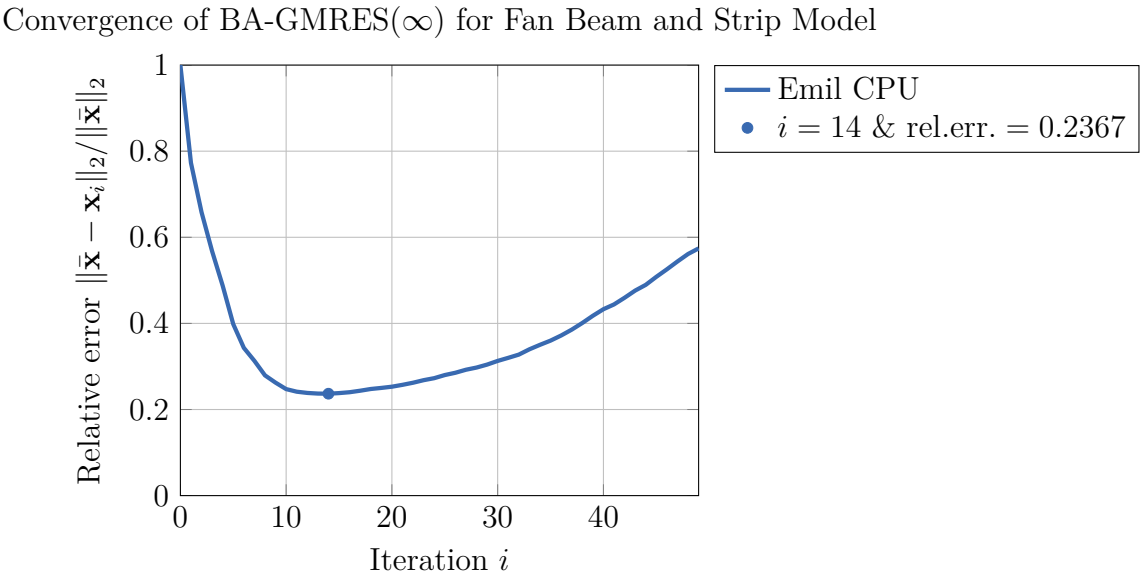


Figure 4.11: The convergence history of BA-GMRES(∞) for fan beam geometry when using the strip model (using 10 rays per strip) as the forward projector.

4.5.4 Conclusion

We can conclude that the projector pairs within the user domain result in semi-convergence for the ABBA methods. Thus, we will continue working with both the line and strip models from the user domain. For the strip model, we will use 10 rays per strip as we can conclude there was no difference in the solution. The only difference is that increasing to 50 rays per strip will increase the memory and, thereby, computation time. Furthermore, we will only look at fan-beam geometries when considering the user domain as we focus on unmatched projector pairs.

4.6 How Unmatched are the Projector Pairs?

We will in the rest of the thesis consider using the public domain from ASTRA and the user domain from Professor Emil Y. Sidky. As we are interested in how the ABBA methods work for unmatched projector pairs, we will now take a look at how unmatched our projector pairs $\mathbf{A}$ and $\mathbf{B}$ are.

4.6.1 Public Domain

With ASTRA GPU, we can compute the forward projector from the Joseph model and its associated unmatched back projector (see Section 2.3.4). We can both obtain an unmatched pair when using parallel and fan beam geometry.

Computing the relative error for parallel beam geometry yields

$$\frac{\left\| \frac{\mathbf{A}_\mathrm{p}}{\|\mathbf{A}_\mathrm{p}\|_F} - \frac{\mathbf{B}_\mathrm{p}^T}{\|\mathbf{B}_\mathrm{p}^T\|_F} \right\|_F}{\left\| \frac{\mathbf{A}_\mathrm{p}}{\|\mathbf{A}_\mathrm{p}\|_F} \right\|_F} = 0.4115, \tag{4.8}$$

and for fan beam we obtain

$$\frac{\left\| \frac{\mathbf{A}_\mathrm{f}}{\|\mathbf{A}_\mathrm{f}\|_F} - \frac{\mathbf{B}_\mathrm{f}^T}{\|\mathbf{B}_\mathrm{f}^T\|_F} \right\|_F}{\left\| \frac{\mathbf{A}_\mathrm{f}}{\|\mathbf{A}_\mathrm{f}\|_F} \right\|_F} = 0.1568. \tag{4.9}$$

The results show that the projector pair constructed for parallel beam is 41% unmatched while it is only approximately 16% when using fan beam geometry.

To illustrate that the matrix constructed from the forward and back projector is not positive definite, we compute the 10 leftmost eigenvalues of $\mathbf{A}_\mathrm{f}\mathbf{B}_\mathrm{f}$ and they are visualized in the complex plane shown in Figure 4.12. We see multiple negative eigenvalues which confirms the problem with the unmatched projector pairs.

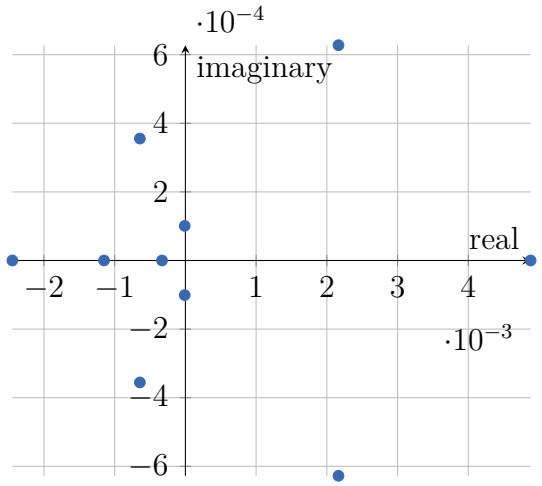


Figure 4.12: The 10 leftmost eigenvalues of **AB** when considering the 2D test problem for fan beam geometry and with the unmatched projector pair from ASTRA.

4.6.2 User Domain

The user domain only has unmatched projector pairs for the fan beam geometry. The user domain provides an unmatched projector pair when using line and strip as the forward projector.

The relative error when using the line model yields

$$\frac{\left\| \frac{\mathbf{A}_\mathrm{l}}{\|\mathbf{A}_\mathrm{l}\|_F} - \frac{\mathbf{B}_\mathrm{l}^T}{\|\mathbf{B}_\mathrm{l}^T\|_F} \right\|_F}{\left\| \frac{\mathbf{A}_\mathrm{l}}{\|\mathbf{A}_\mathrm{l}\|_F} \right\|_F} = 0.3085. \tag{4.10}$$

For the strip model with 10 rays per strip, we obtain

$$\frac{\left\| \frac{\mathbf{A}_\mathrm{s}}{\|\mathbf{A}_\mathrm{s}\|_F} - \frac{\mathbf{B}_\mathrm{s}^T}{\|\mathbf{B}_\mathrm{s}^T\|_F} \right\|_F}{\left\| \frac{\mathbf{A}_\mathrm{s}}{\|\mathbf{A}_\mathrm{s}\|_F} \right\|_F} = 0.3454. \tag{4.11}$$

Thus, the user domain produces projector pairs that are approximately 31% (for line) and 35% (for strip) unmatched.

4.7 Summary

This chapter has focused on finding suitable software packages that can provide unmatched projector pairs to the ABBA methods. We have investigated two public domains, ASTRA and TIGRE, and one user domain provided by Professor Emil Y. Sidky.

Table 4.12 summaries which forward projector models are available in the software packages. Note, a circle is used around TIGRE's Joseph model as we have experienced problems. From our study, we concluded that there is a bug in TIGRE's code for the Joseph forward projection model.

	Line	Strip	Joseph
	Parallel beam		
ASTRA, CPU	✓	✓	✓
ASTRA, GPU	✗	✗	✓
TIGRE, GPU	✓	✗	⊘
Emil, CPU	✓	✓	✗
	Fan beam		
ASTRA, CPU	✓	✓	✗
ASTRA, GPU	✗	✗	✓
TIGRE, GPU	✓	✗	⊘
Emil, CPU	✓	✓	✗

Table 4.12: An overview of the forward projection models in the toolboxes for parallel and fan beam geometries. Operators with problems are marked with a circle.

Table 4.13 summarises the available back projectors within the software packages. Due to limited documentation, the back projection operators are in some cases still a mystery.

	Parallel beam	Fan beam
ASTRA, CPU	$\mathbf{B} = \mathbf{A}^T$	$\mathbf{B} = \mathbf{A}^T$
ASTRA, GPU	From Section 2.3.4	Not documented
TIGRE, GPU	From Section 2.3.4	Not documented
Emil, CPU	$\mathbf{B} = \mathbf{A}^T$	$\mathbf{B} = \mathbf{A}^T$ and Pixel-driven (not documented)

Table 4.13: An overview of the back projection operators in the toolboxes for parallel and fan beam geometries.

Based on the analysis, we end up choosing three unmatched projector pairs for our further studies. We consider the public domain ASTRA and the user domain. The chosen projectors are shown in the gray box. The percentages illustrate how unmatched the projector pairs are.

Unmatched projector pairs within the software domains, we will use:

- ASTRA GPU: **A** is Joseph's model and **B** is from Section 2.3.4 for parallel beam ($\sim 41\%$) and for fan beam not documented ($\sim 16\%$).
- Emil CPU: **A** is the line model and **B** is pixel-driven ($\sim 31\%$).
- Emil CPU: **A** is the strip model with 10 rays per strip and **B** is pixel-driven ($\sim 35\%$).

CHAPTER 5

Applying Restart and Stopping Criteria

At this stage, we can construct unmatched projector pairs and are now able to investigate the influence of using restart and different choices of stopping criteria for our ABBA iterative solvers. We will consider both using the ASTRA toolbox and the user domain for the construction of the unmatched projector pairs (see the gray box in Section 4.7 for details). We continue using the Shepp Logan test problem as used in the previous chapters.

This chapter studies the influence of using restarted GMRES. We will investigate how this affects the convergence and compare the amount of work for AB/BA-GMRES(∞) and AB/BA-GMRES(p).

Moreover, we study the influence of the stopping criteria Discrepancy Principle (DP) and Normalized Cumulative Periodogram (NCP). We study their reliability and how good they are at finding the optimal iteration.

The structure of the chapter follows:

- Study the influence of using restarted GMRES.
- Study the stopping criteria DP and NCP.

5.1 Restarted GMRES

In Section 3.7, we introduced restarted GMRES to reduce the computational and memory complexity of large inverse problems, making them easier to solve. We will now investigate how this affects the ABBA solvers when we have unmatched projector pairs. We will look at ASTRA and the user domain software separately as they do not include the same models when considering unmatched projector pairs.

5.1.1 Public Domain: ASTRA

We start by investigating the convergence when using the unmatched projector pair from ASTRA GPU. To limit the extent of this chapter, we will only present the example with fan beam geometry here. However, ASTRA can also create an unmatched projector pair when we have parallel beam geometry. For the interested reader, we refer to Appendix B.3.2 where the results for the ABBA methods are shown. This chapter compares the non-restarted ABBA solvers (referred to as $p = \infty$) with the restarted versions with $p = 3, 5, 10$. We are interested in how applying restart affects the convergence history of the ABBA methods and what happens when changing the restart parameter p.

Figure 5.1 shows the convergence history for BA-GMRES(∞) and BA-GMRES(p) for the different choices of p when using fan beam geometry. The relative error for BA-GMRES(∞) increases drastically due to the semi-convergence compared to using restarted BA-GMRES. BA-GMRES(∞) is the first to reach the optimum at iteration 18 with a relative error of 20.74%. Then $p = 10$ reaches the optimum at iteration 23 followed by $p = 5$ at iteration 27 with the relative errors 20.69% and 20.63%, respectively. The last to reach the optimum is $p = 3$ at iteration 50 with a relative error of 20.62%.

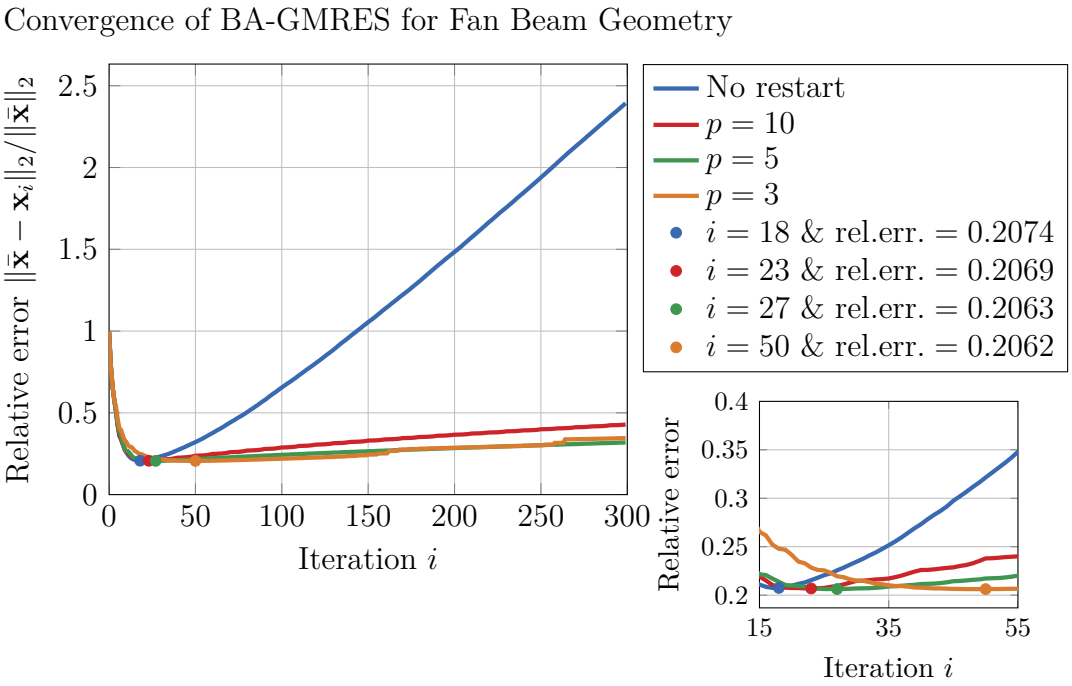


Figure 5.1: The convergence history of BA-GMRES(∞) and BA-GMRES(p) with $p = 3, 5, 10$ when having fan beam geometry.

Considering Figure 5.1, we make multiple observations. First, we see that there is a dependence between the number of iterations and the restart parameter p. As p decreases, the number of iterations increases as the convergence rate decrease. Hence, when using restarted BA-GMRES the solver is less sensitive to semi-convergence. If we

take too many steps in restarted BA-GMRES the solution is much better compared to taking too many steps in BA-GMRES(∞). The same behavior is seen for AB-GMRES (see Figure B.6 in Appendix B.3.1).

In Section 3.7, we discussed the number of FLOPs needed for BA-GMRES(∞) and BA-GMRES(p). For the 300 iterations just discussed (Figure 5.1), the relative error as a function of giga (G) FLOPs is shown in Figure 5.2 (we here compute the residuals in each iteration). The dots illustrate the number of FLOPs needed to reach the optimum. BA-GMRES(∞) was the first to converge and, thereby, also uses the fewest number of FLOPs if we stop at the optimum. If we were to run all 300 iterations then BA-GMRES(∞) is the most expensive solver. BA-GMRES(3) uses most FLOPs to reach the optimum as it needs a lot of iterations. In Appendix B.3.1 Figure B.7, the convergence of AB-GMRES when comparing the relative error with the number of FLOPs is shown. The overall behavior is similar to BA-GMRES. The only difference is that it is cheaper to reach the optimum (because the number of iterations to optimum is less than for BA-GMRES) both for AB-GMRES(∞) and AB-GMRES(p).

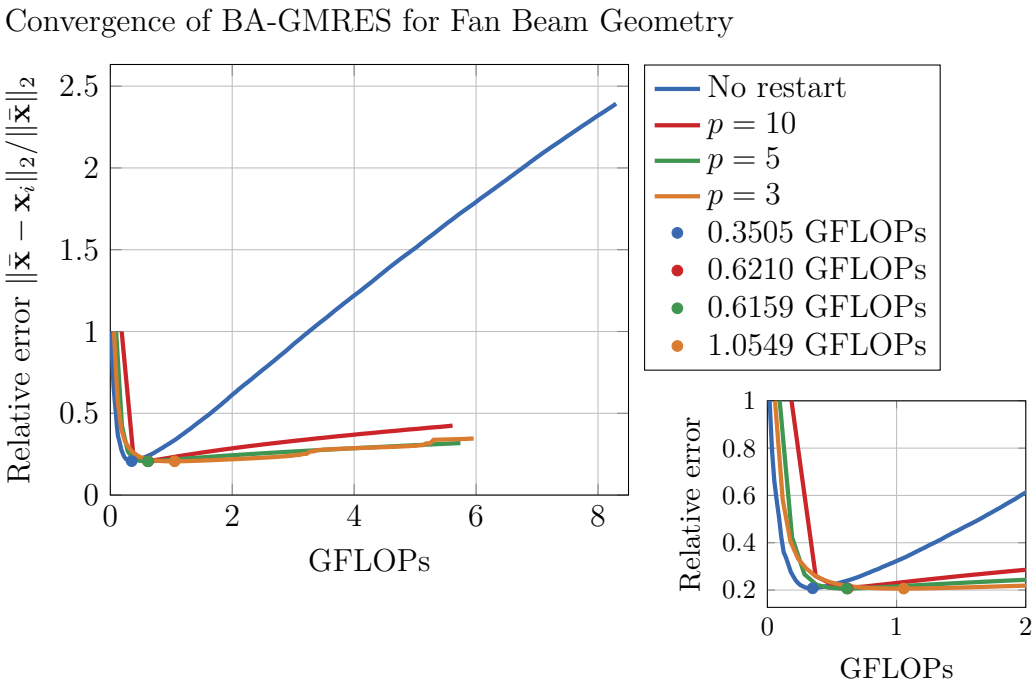


Figure 5.2: The relative error as a function of giga (G) FLOPs for 300 iterations of BA-GMRES(∞) and BA-GMRES(p) with $p = 3, 5, 10$.

Finally, we also consider the relative error as a function of the wall clock time, see Figure 5.3. It is clear, that running all 300 iterations is expensive when using BA-GMRES(∞) compared to BA-GMRES(p). Looking at the time it takes for the solvers to reach optimum, we see that BA-GMRES(5) and BA-GMRES(∞) approximately take the same time and are closely followed by BA-GMRES(10). The overall behavior is the same for

AB-GMRES (see Figure B.8 in Appendix B.3.1). The only difference is that $p = 10$ reaches the optimum the fastest followed by not using restart and $p = 5$, respectively.

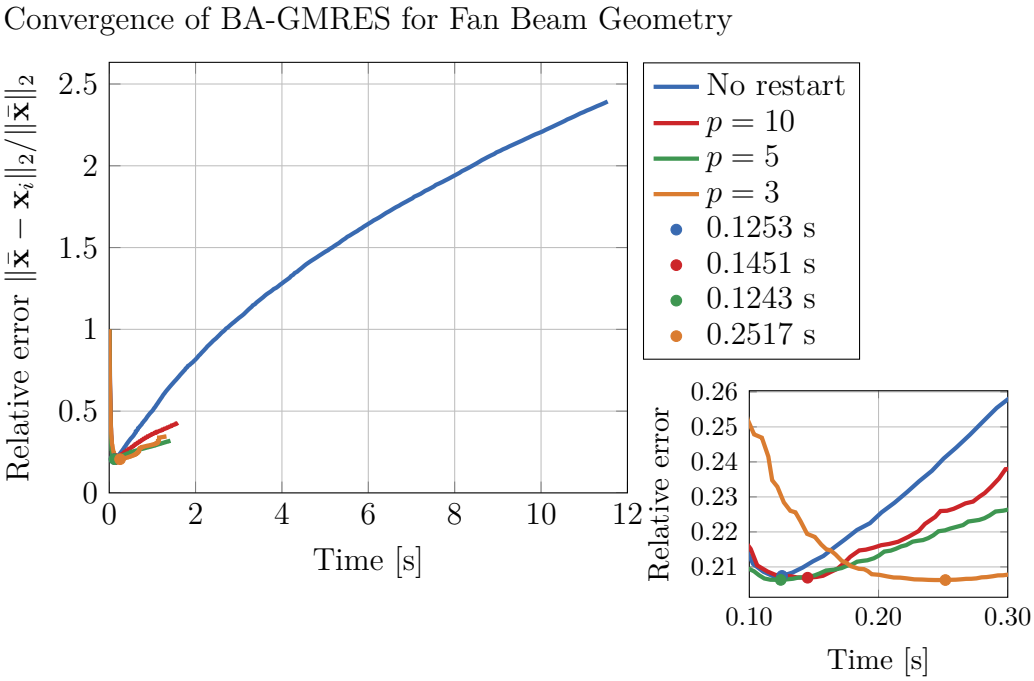


Figure 5.3: The relative error as a function of the wall-clock time for 300 iterations of BA-GMRES(∞) and BA-GMRES(p) with $p = 3, 5, 10$.

5.1.2 User Domain

From the user domain, we will consider the two unmatched projector pairs (FP = line model and FP = strip model). The user domain only has unmatched projector pairs for fan beam geometry, so that is the only geometry we consider here.

5.1.2.1 Line Model

Using the line model as the forward projector in BA-GMRES(∞) and BA-GMRES(p) leads to the result shown in Figure 5.4. We see the same overall behavior as obtained with ASTRA. The only major difference is that the relative error for BA-GMRES(∞) does not increase as much as when we considered ASTRA (keep in mind it is two different forward projectors as ASTRA uses Joseph's model). With ASTRA the relative error where approximately 2.5 at iteration 300 where it is only around 0.6 here.

Figure 5.5 show the relative error as a function of the time. The user domain is based on CPU computations; therefore, reaching the optimum takes more time than the results from ASTRA. The results show that increasing p decrease the time it takes to reach the optimal solution due to the number of iterations needed. So there is a trade-off between

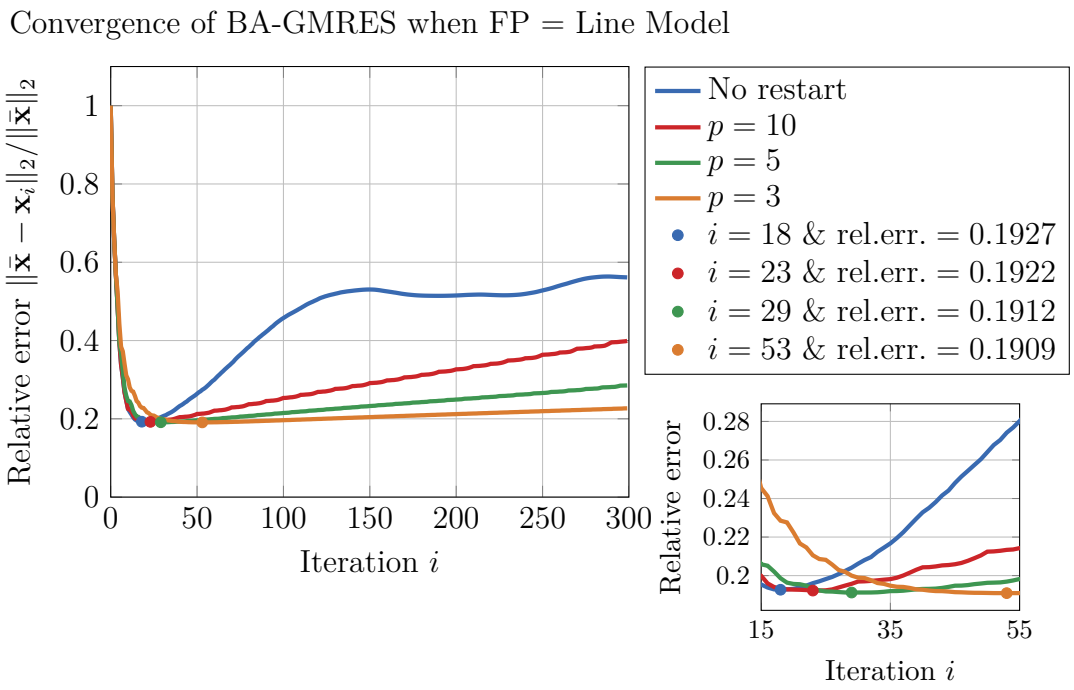


Figure 5.4: The convergence history of BA-GMRES(∞) and BA-GMRES(p) with $p = 3, 5, 10$ when having fan beam geometry. The forward projector is the line model.

the time and obtaining the smallest possible relative error. When using restarted BA-GMRES, it seems from the results that we can stop before and still obtain the same relative error as BA-GMRES(∞) and in that case, the wall-clock time would be shorter.

We have constructed the same figures with AB-GMRES which are shown in Appendix B.3.3. The number of iterations needed to reach the optimum follows the same structure as for BA-GMRES and the same behavior for the wall-clock time is obtained. The major difference is that the relative error obtained with AB-GMRES(∞) after approximately 125 iterations follows AB-GMRES(10).

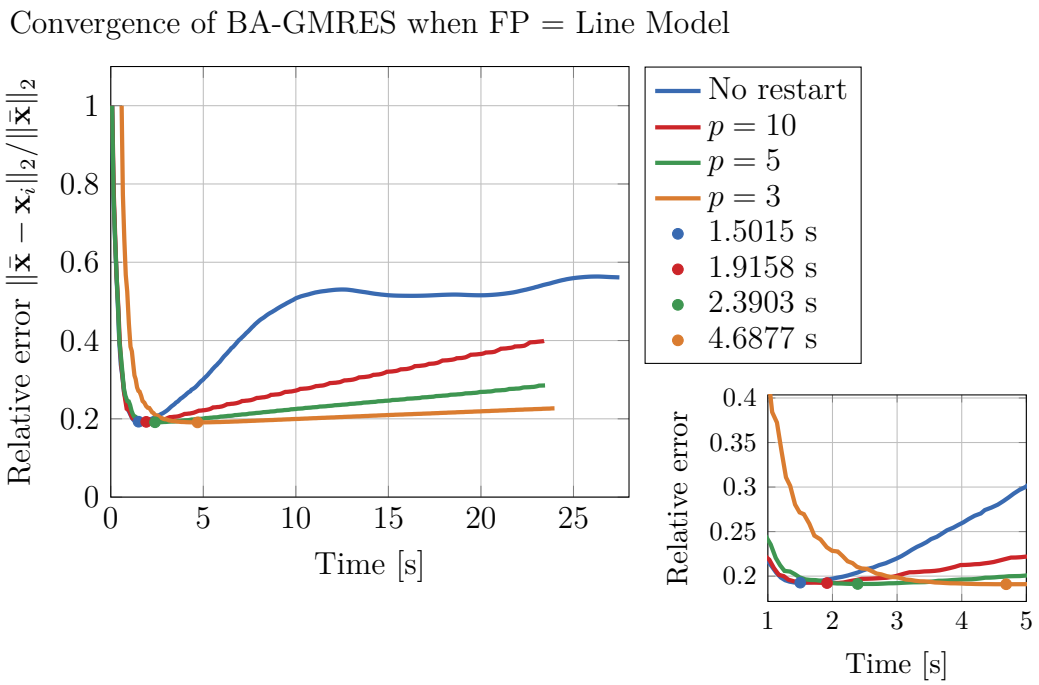


Figure 5.5: The relative error as a function of the wall-clock time for 300 iterations of BA-GMRES(∞) and BA-GMRES(p) with $p = 3, 5, 10$ and when using the line model as the forward projector.

5.1.2.2 Strip Model

Finally, we also consider using the strip model as the forward projector and, thereby, consider our last unmatched projector pair. We here use 10 rays per strip.

Figure 5.6 shows the convergence history. We see that BA-GMRES(∞) reaches the optimum at iteration 14 followed by BA-GMRES(10) at iteration 15 and BA-GMRES(5) at iteration 18. The relative error for those solutions is approximately 23% and is, therefore, higher than the previous projector pairs. Another observation is that the relative error of BA-GMRES(10) reaches a point where it exceeds the relative error of 1 which was not seen in the example with the line model. Although the convergence rate has increased for restarted BA-GMRES it does continue to be smaller than the convergence rate for BA-GMRES(∞). The increased convergence rate imply that the solution is more sensitive to selecting a wrong stopping criterion.

The wall-clock time is shown in Figure 5.7. The time it takes to reach the optimum is much larger compared to the previous example. The reason is that the strip model in the user domain is an approximation and, thus, for each strip, we have 10 rays we need to make computations with instead of just one. Thus, we extend the scale of the problem.

The results for AB-GMRES are shown in Appendix B.3.4. The only major difference is

that the relative error is below one.

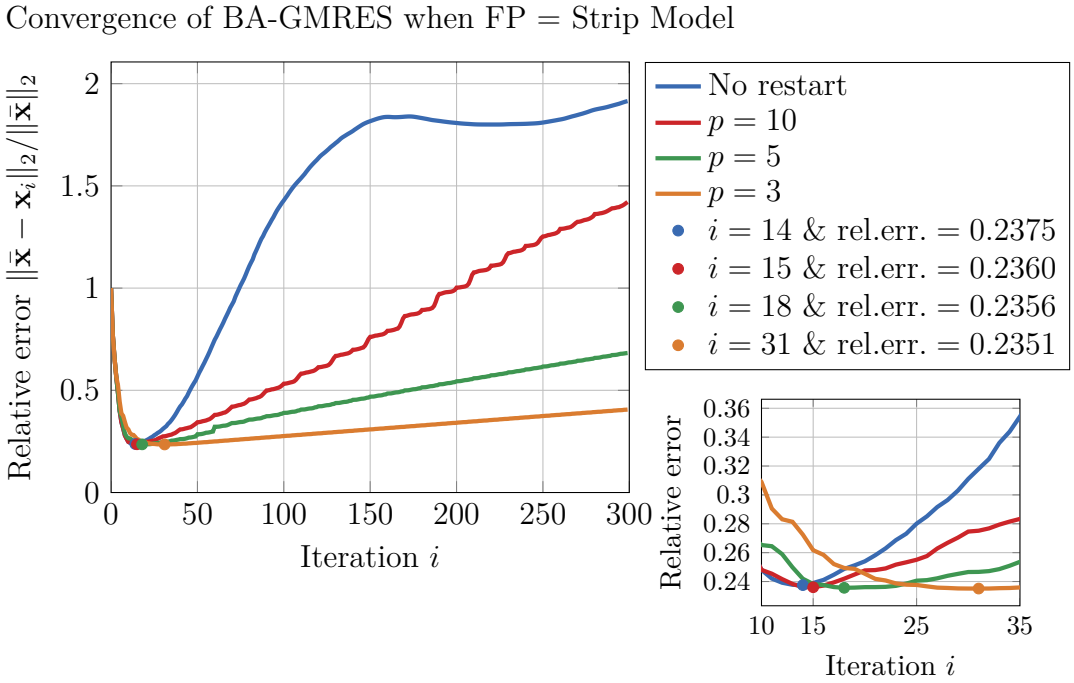


Figure 5.6: The convergence history of BA-GMRES(∞) and BA-GMRES(p) with $p = 3, 5, 10$ when having fan beam geometry. The forward projector is the strip model with 10 rays per strip.

5.1.3 Conclusion

From the results, we can conclude that using restarted GMRES decreases the convergence rate compared to not using restart. This means that restarted GMRES is less sensitive to stopping criteria that may take too many iterations compared to the regular GMRES. There is a relation between the value of the restart parameter p and the rate of convergence. A small p yields a slow convergence while increasing p yields increasing the convergence rate and makes it approach AB/BA-GMRES(∞). So besides the memory complexity, there is a trade-off between stability and computational speed.

If we need to take many iterations or we do not have a robust way of selecting the stopping criterion (if it takes too many iterations) then restarted GMRES with a low p value is preferred as the solution does not change as much from iteration to iteration. Furthermore, the memory requirement is lower compared to regular GMRES. If we, however, do not have memory issues the ABBA methods without restart compute the solution in the fewest amount of iterations.

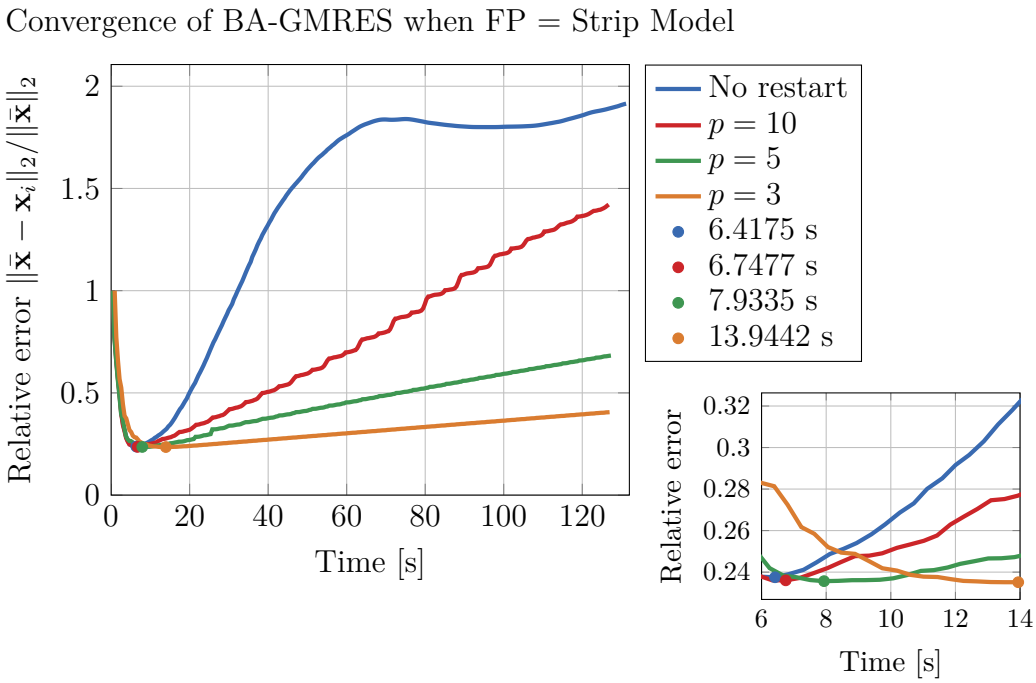


Figure 5.7: The relative error as a function of the wall-clock time for 300 iterations of BA-GMRES(∞) and BA-GMRES(p) with $p = 3, 5, 10$ and when using the strip model with 10 rays per strip as the forward projector.

5.2 Stopping Criteria

In Section 3.8, we discussed two stopping criteria, namely DP and NCP. We will now combine these stopping criteria with the ABBA methods and study how good they are at finding the optimal solution. We will consider if there is a difference depending on the unmatched projector pair.

5.2.1 Discrepancy Principle

The first stopping criterion we consider is DP. We use the safety factor $\nu_{\text{dp}} = 1.02$ and for the noise level, we use the true standard deviation of the noise, meaning $\eta = \sigma$. We introduce DP in both AB/BA-GMRES(∞) and AB/BA-GMRES(5).

5.2.1.1 Public Domain: ASTRA

For the fan beam geometry, ASTRA provides a 16% unmatched projector pair. Figure 5.8 shows the convergence history for BA-GMRES when computing $\mathbf{A}$ and $\mathbf{B}$ from ASTRA GPU with the fan beam geometry. Considering the convergence history for BA-GMRES(∞), we see that the optimal solution is at iteration 17 but DP makes the solver stop at iteration 12. So, DP stops 5 iterations too early. The relative error at iteration 12 is $\sim 23\%$ while at iteration 18 it is $\sim 21\%$. Considering BA-GMRES(5), we

see that the optimal iteration is iteration 27 while DP stops the solver at iteration 14. The relative error difference between iterations 14 and 27 is $\sim 2\%$.

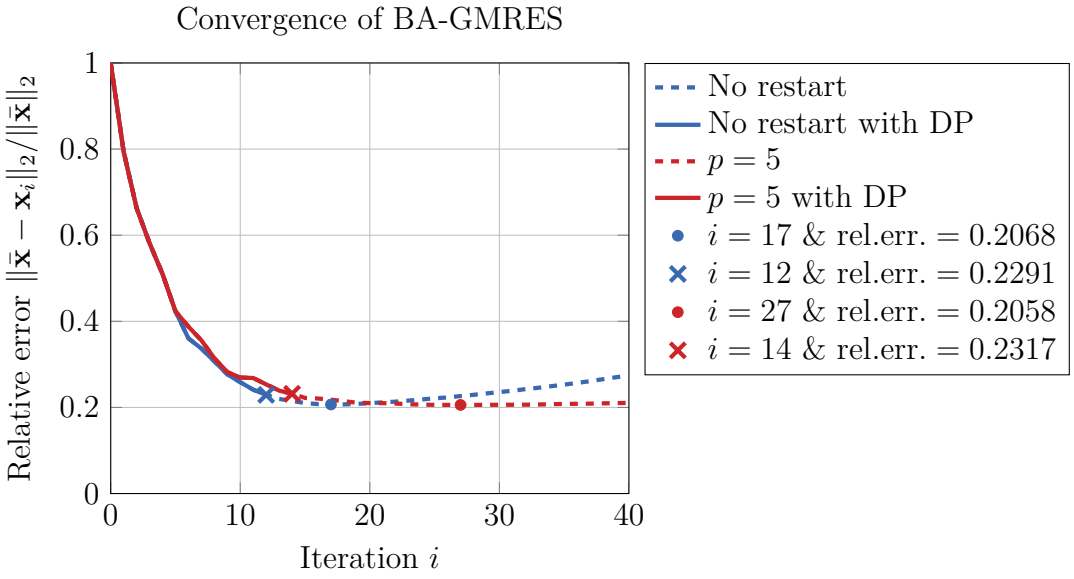


Figure 5.8: The convergence history of BA-GMRES(∞) and BA-GMRES(5). We both consider using the stopping criterion DP and not using a stopping criterion.

We can conclude that DP stops the solvers too early. However, this is only in one simulation. To get a better understanding of the behavior of DP, we will consider 100 realizations of the noise. From those, we can compute the ratio

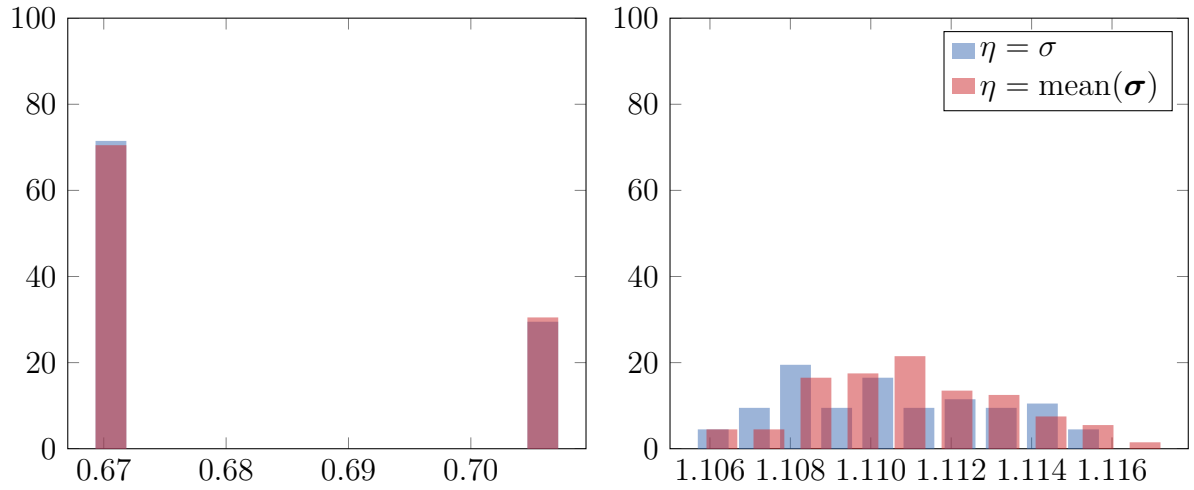
$$R_{i,\mathrm{dp}} = \frac{i_{\mathrm{dp}}}{i_{\mathrm{opt}}}. \tag{5.1}$$

We can use this ratio to determine how consistent the DP stopping criterion is and if it under or overestimates the optimal iteration number. So far, we have seen it underestimate the parameter. Furthermore, we will look at the ratio between the two solutions

$$R_{\mathrm{e,dp}} = \frac{e_{\mathrm{dp}}}{e_{\mathrm{opt}}} = \frac{\|\mathbf{x}_{\mathrm{dp}} - \bar{\mathbf{x}}\|_2}{\|\mathbf{x}_{\mathrm{opt}} - \bar{\mathbf{x}}\|_2}, \tag{5.2}$$

to see how far from each other the solutions are. For these two ratios, we will both consider using the true standard deviation ($\eta = \sigma$) and a mean of 100 realizations of the standard deviation ($\eta = \mathrm{mean}(\boldsymbol{\sigma})$) as the real world would never have the true σ.

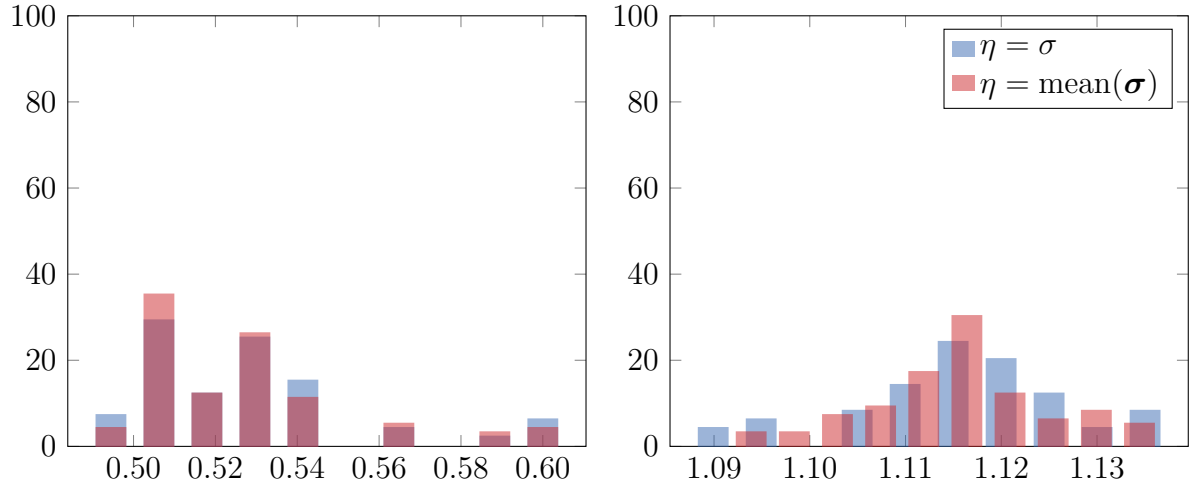
Figure 5.9 shows the ratios when using BA-GMRES(∞). We obtain the same behavior for η being the true standard deviation and when using an approximation. Considering Figure 5.9a, we see that the overall trend is that DP underestimates the optimal iteration. DP shift between stopping at two different iterations as $\sim 70\%$ of the times we obtain a ratio $R_{i,\mathrm{dp}} = 0.67$ and the last $\sim 30\%$ of the time it is at ratio 0.71. If we look at the error ratios in Figure 5.9b, we see that the difference between the optimal solution and the one obtained with DP is around 11% different.



(a) The ratios $R_{i,\text{dp}}$ for BA-GMRES(∞). **(b)** The ratios $R_{e,\text{dp}}$ for BA-GMRES(∞).

Figure 5.9: A histogram showing the ratios (5.1) to the left and (5.2) to the right when using BA-GMRES(∞) to solve the unmatched CT problem obtained from ASTRA GPU with fan beam geometry.

If we consider the case with restarted GMRES, see Figure 5.10, the results show that DP is not as consistent as for BA-GMRES(∞) as the iteration ratios are more distributed. We, however, still see that DP underestimates.



(a) The ratios $R_{i,\text{dp}}$ for BA-GMRES(5). **(b)** The ratios $R_{e,\text{dp}}$ for BA-GMRES(5).

Figure 5.10: A histogram showing the ratios (5.1) to the left and (5.2) to the right when using BA-GMRES(5) to solve the unmatched CT problem obtained from ASTRA GPU with fan beam geometry.

In Appendix B.4.1 Figure B.17 and B.18 the results for AB-GMRES(∞) and AB-GMRES(5) are shown, respectively. For AB-GMRES(5), almost 80% of the iteration ratios are the same but other than that the results for AB-GMRES and BA-GMRES look similar.

If we consider a parallel beam geometry, ASTRA GPU provides a 41% unmatched projector pair and the results for AB/BA-GMRES are shown in Appendix B.4.1 (not shown here as the difference between histograms from fan and parallel beam geometry is small). We obtain error ratios of around 1.01 to 1.02 (when using BA-GMRES) which corresponds to a difference of 1-2% between $\mathbf{x}_{\text{dp}}$ and the optimal solution. The iteration ratios are, however, still below one as we have seen for the fan beam geometry.

In conclusion, when considering parallel beam geometry, DP underestimates the number of iterations but the solution found with DP is more similar to the optimal solution compared to when considering fan beam geometry. This happens despite that the projector pair is more unmatched for parallel beam than for fan beam. To obtain better results, we could consider changing ν_{dp}, however, that comes with the risk of getting poor results if the noise level is underestimated.

5.2.1.2 User Domain

The user domain produces a 31% unmatched projector pair when using the line model as the forward projector. Figure 5.11 shows the convergence history for BA-GMRES when using the unmatched projector pair. The optimal solution for BA-GMRES(∞) is found at iteration 18 while DP stops at iteration 12. In the case with BA-GMRES(5), the optimal number of iterations is 32 while DP only takes 15. Thus, we again see that DP stops too early.

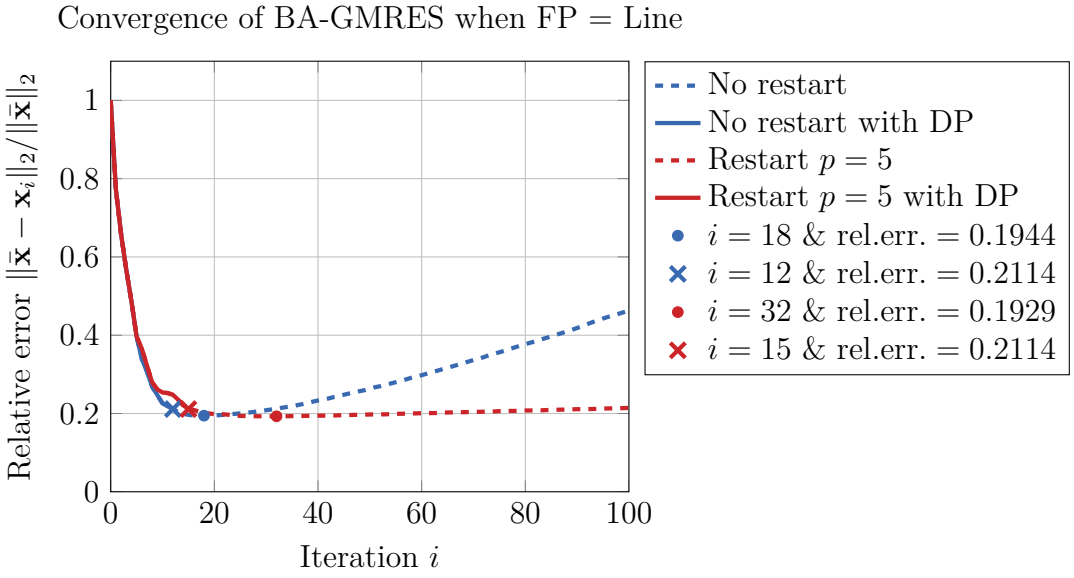
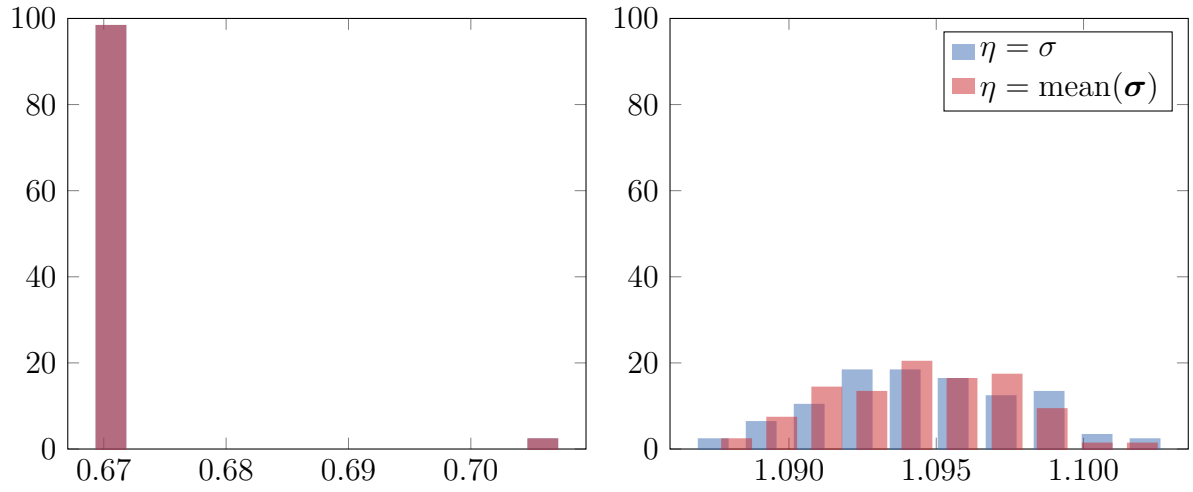


Figure 5.11: The convergence history of BA-GMRES with an unmatched projector pair from the user domain with the forward projector being the line model.

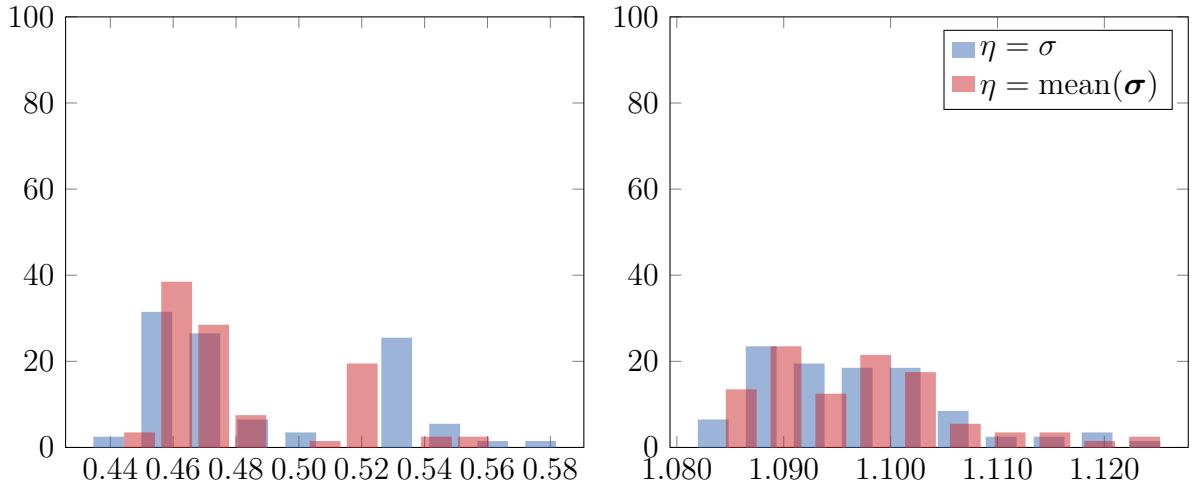
We again consider 100 realizations of the noise to see how the overall behavior is. Figure 5.12 shows the iteration and error ratios when considering BA-GMRES(∞). Almost all

100 realizations lead to the same iteration ratio (0.67) for both η values. Thus, DP is in this case consistent but stops too early leading to a result $\mathbf{x}_{\text{dp}}$ which is 9-10% different from the optimal solution.



(a) The ratios $R_{i,\text{dp}}$ for BA-GMRES(∞). **(b)** The ratios $R_{e,\text{dp}}$ for BA-GMRES(∞).

Figure 5.12: A histogram showing the ratios (5.1) to the left and (5.2) to the right when using BA-GMRES(∞) with the forward projector being the line model and for fan beam geometry



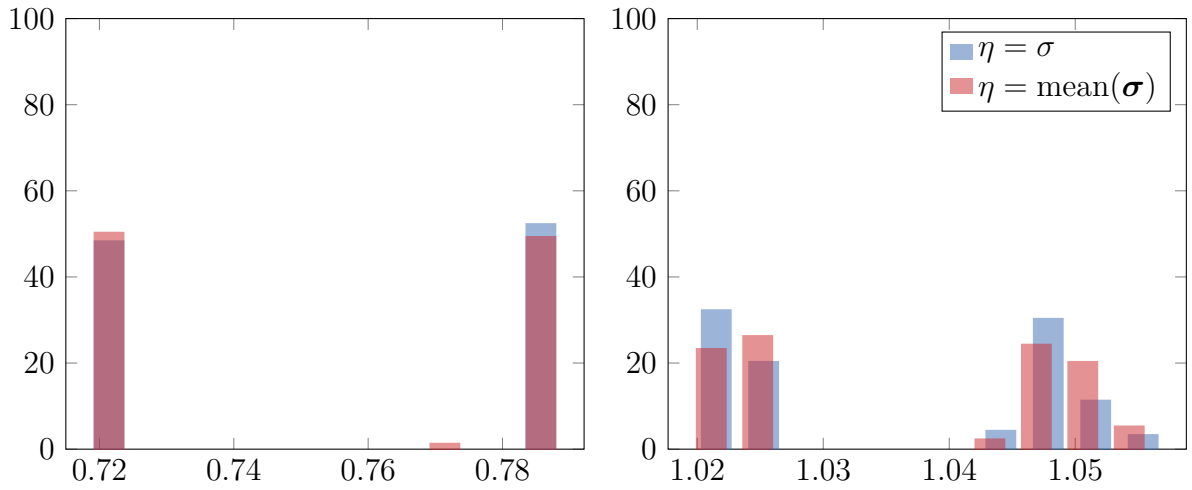
(a) The ratios $R_{i,\text{dp}}$ for BA-GMRES(5). **(b)** The ratios $R_{e,\text{dp}}$ for BA-GMRES(5).

Figure 5.13: A histogram showing the ratios (5.1) to the left and (5.2) to the right when using BA-GMRES(5) with the forward projector being the line model and for fan beam geometry

Looking at the ratios for BA-GMRES(5), see Figure 5.13, we see a more distributed result. Thus, when using restart in this case DP is not consistent leading to error ratios between 8-12%. The error ratios are not much larger even though the iteration ratios are smaller than for BA-GMRES(∞). This is due to the result not changing much in each iteration when using restarted GMRES compared to not using restart.

The results for AB-GMRES(∞) are shown in Appendix B.4.2. Here we see similar behavior. When considering AB-GMRES(∞), DP underestimates and is consistent. When looking at AB-GMRES(5), DP underestimates and is no longer consistent. AB-GMRES obtain error ratios between 5 and 6% for AB-GMRES(∞) and between 7 and 11% for AB-GMRES(5).

We will also take a quick look at how the results look if we instead of using the line model as the forward projector use the strip model with 10 rays per strip. Thus, we will use a projector pair that is 35% unmatched. Figure 5.14 shows the results for BA-GMRES(∞) and Figure 5.15 shows the results for BA-GMRES(5) (for AB-GMRES results see Figure B.25 and B.26 in Appendix B.4.2). The major difference is that the error ratios now are between 1% and 5% instead of 5% and 12%.



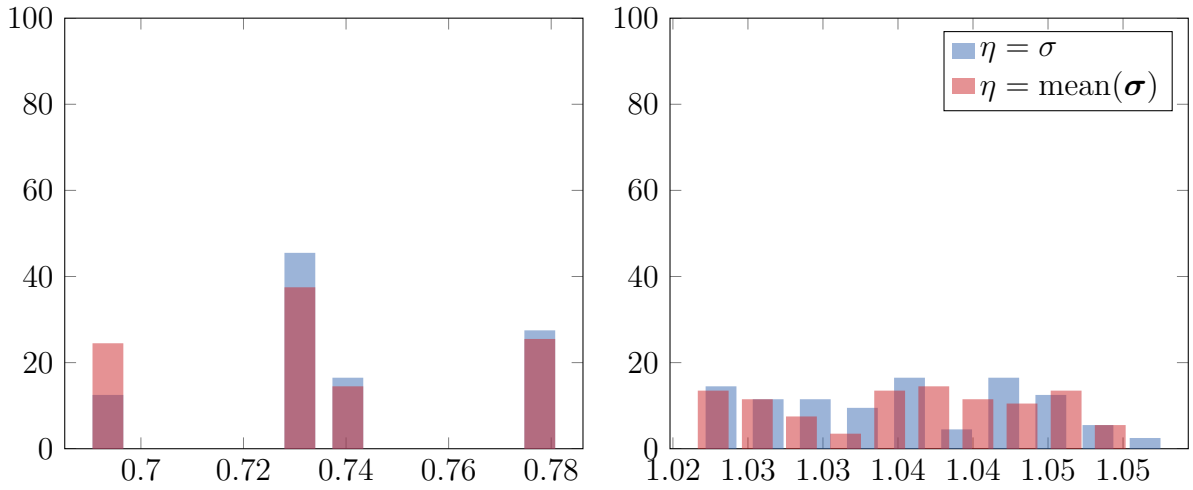
(a) The ratios $R_{i,\text{dp}}$ for BA-GMRES(∞). **(b)** The ratios $R_{e,\text{dp}}$ for BA-GMRES(∞).

Figure 5.14: A histogram showing the ratios (5.1) to the left and (5.2) to the right when using BA-GMRES(∞) with the forward projector being the strip model (10 rays per strip) and for fan beam geometry.

5.2.1.3 Conclusion

We briefly summarize the relevant results to base a conclusion on DP. A good stopping criterion should consistently provide solutions close to the optimal solution. Therefore, we have sought to investigate if DP is consistently good. We say that a method is consistent if it mostly provides a similar result given different realizations of the same noise model. Moreover, we have seen that $\eta = \sigma$ gives approximately the same result as $\eta = \text{mean}(\boldsymbol{\sigma})$ and, thus, we do not differentiate between them.

Table 5.1 gives an overview of the consistency for all possible unmatched projector pairs when using AB/BA-GMRES with and without restart. Not using restart, yields DP being consistent for all possible projector pairs. Using restarted GMRES entails DP not being consistent.



(a) The ratios $R_{i,\text{dp}}$ for BA-GMRES(5). **(b)** The ratios $R_{e,\text{dp}}$ for BA-GMRES(5).

Figure 5.15: A histogram showing the ratios (5.1) to the left and (5.2) to the right when using BA-GMRES(5) with the forward projector being the strip model (10 rays per strip) and for fan beam geometry.

		AB(∞)	AB(5)	BA(∞)	BA(5)
ASTRA GPU	Parallel beam	✓	✗	✓	✓
	Fan beam	✓	✗	✓	✗
User domain	Line	✓	✗	✓	✗
	Strip	✓	✗	✓	✗

Table 5.1: Overview of the consistency for DP.

Table 5.2 show the percentage difference between $\mathbf{x}_{\text{dp}}$ and $\mathbf{x}_{\text{opt}}$ obtained from the error ratios. Using the parallel beam projector pair from ASTRA results in $\mathbf{x}_{\text{dp}}$ being very close to the optimal solution when considering BA-GMRES with and without restart. Furthermore, using the strip model from the user domain results in the solution found with DP being close to the optimal. With DP, we have obtained solutions that are at most 14% different from the optimal solution. In general, DP underestimates the

		AB(∞)	AB(5)	BA(∞)	BA(5)
ASTRA GPU	Parallel beam	6.9% - 8.1%	6% - 12%	1% - 2%	1.5% - 2.3%
	Fan beam	7% - 9%	7% - 12%	10% - 12%	9% - 14%
User domain	Line	9% - 10%	8% - 12%	5% - 6%	7% - 11%
	Strip	2% - 5%	2% - 5%	1% - 2%	2% - 5%

Table 5.2: Overview of the difference between $\mathbf{x}_{\text{dp}}$ and $\mathbf{x}_{\text{opt}}$.

optimal number of iterations leading to solutions that are between 1% and 14% different from the optimal solution. The overall trend is that DP is consistent when using AB/BA-GMRES(∞) and not consistent when using AB/BA-GMRES(5).

5.2.2 Normalized Cumulative Periodogram

The second stopping criterion we consider is NCP, which has the advantage of not needing to set any parameters. We will again consider the same unmatched projector pairs and look at their ratios

$$R_{i,\text{ncp}} = \frac{i_{\text{ncp}}}{i_{\text{opt}}}, \tag{5.3}$$

and

$$R_{\text{e,ncp}} = \frac{e_{\text{ncp}}}{e_{\text{opt}}} = \frac{\|\mathbf{x}_{\text{ncp}} - \bar{\mathbf{x}}\|_2}{\|\mathbf{x}_{\text{opt}} - \bar{\mathbf{x}}\|_2}. \tag{5.4}$$

5.2.2.1 Public Domain: ASTRA

Figure 5.16 shows the convergence history for BA-GMRES when creating $\mathbf{A}$ and $\mathbf{B}$ from ASTRA GPU for both $p = 5$ and $p = \infty$. Considering $p = \infty$, we achieve the optimal solution at iteration 17 but NCP stops at iteration 8 and for $p = 5$ the optimal solution is at iteration 27 but NCP stops at iteration 7. Thus, NCP seems to stop too early and this leads to relative errors of 30-35% instead of 20%.

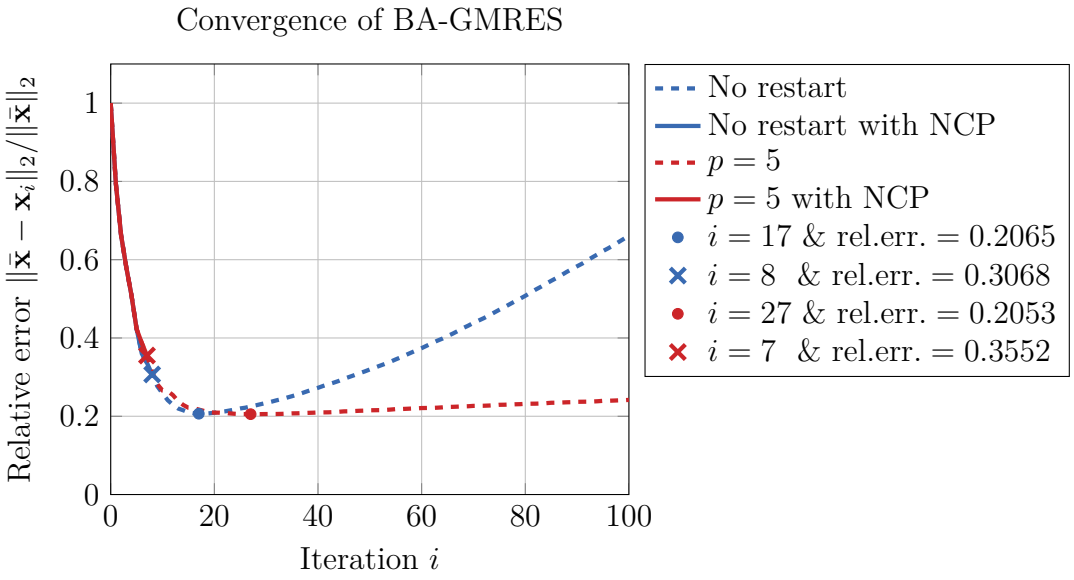
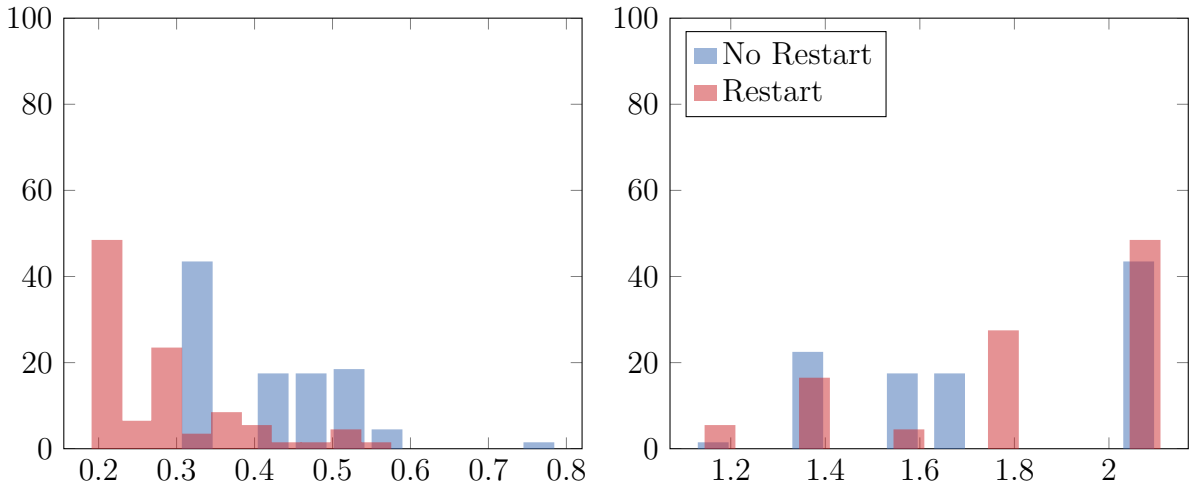


Figure 5.16: The convergence history of BA-GMRES(∞) and BA-GMRES(5). We both consider using the NCP stopping criterion and not using a stopping criterion.

Considering 100 realizations of the noise and solving the inverse CT problem, we obtain the results shown in Figure 5.17. The difference between using GMRES with and without restart is very little. In both cases, NCP severally underestimates the optimal iteration and it is not consistent. Looking at the error ratios, we see that the relative error is also very high for the found solutions $\mathbf{x}_{\text{ncp}}$ compared to the optimal solution. The results are similar to AB-GMRES shown in Appendix B.5.1 Figure B.27. Moreover, the results for parallel beam are also shown in Appendix B.5.1.



(a) The ratios $R_{i,\text{ncp}}$ for BA-GMRES. **(b)** The ratios $R_{e,\text{ncp}}$ for BA-GMRES.

Figure 5.17: A histogram showing the ratios (5.3) to the left and (5.4) to the right when using BA-GMRES to solve the unmatched CT problem obtained from ASTRA GPU for fan beam geometry. We both consider no restart $p = \infty$ and restart $p = 5$.

5.2.2.2 User Domain

Considering the user domain with the line model as the forward projector, we obtain the convergence history shown in Figure 5.18. Again, we see that NCP stops too early.

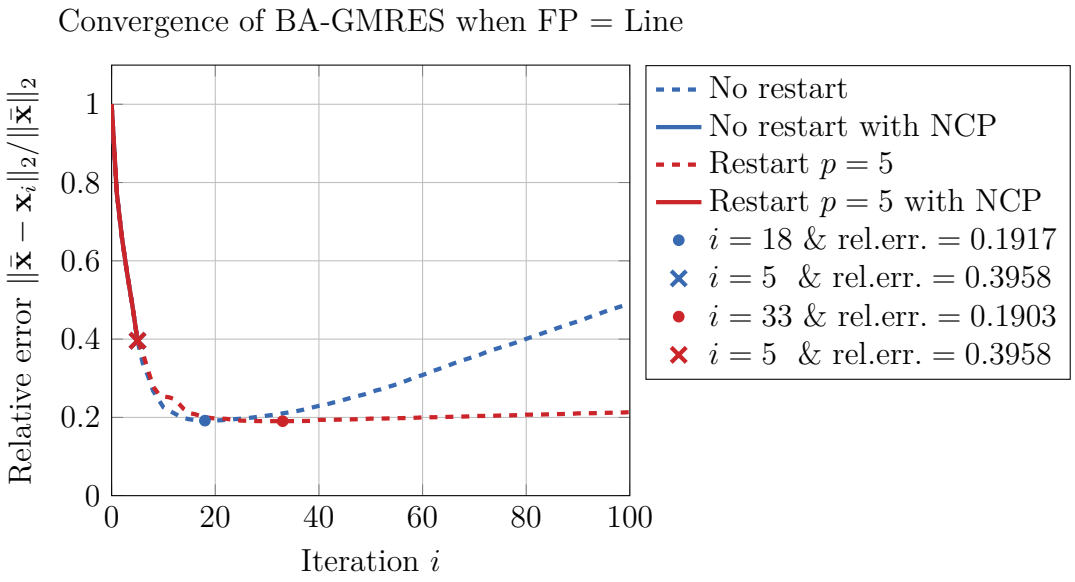
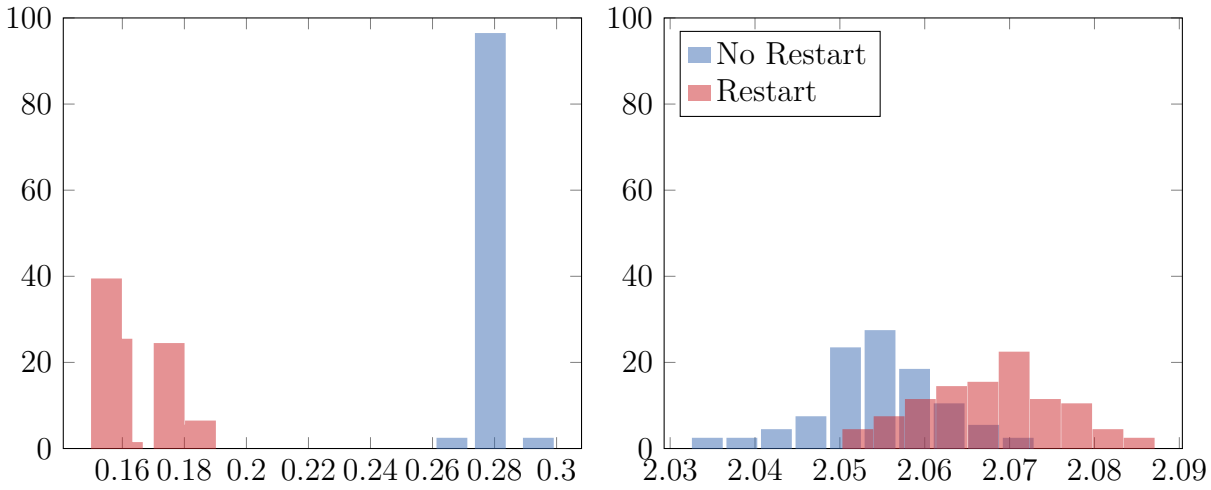


Figure 5.18: The convergence history of BA-GMRES. The projector pair is constructed from the user domain and the forward projector is the line model.

We consider 100 realizations of the noise and when using the line model as the forward projector we obtain the results shown in Figure 5.19. We do not see a tail here as we did

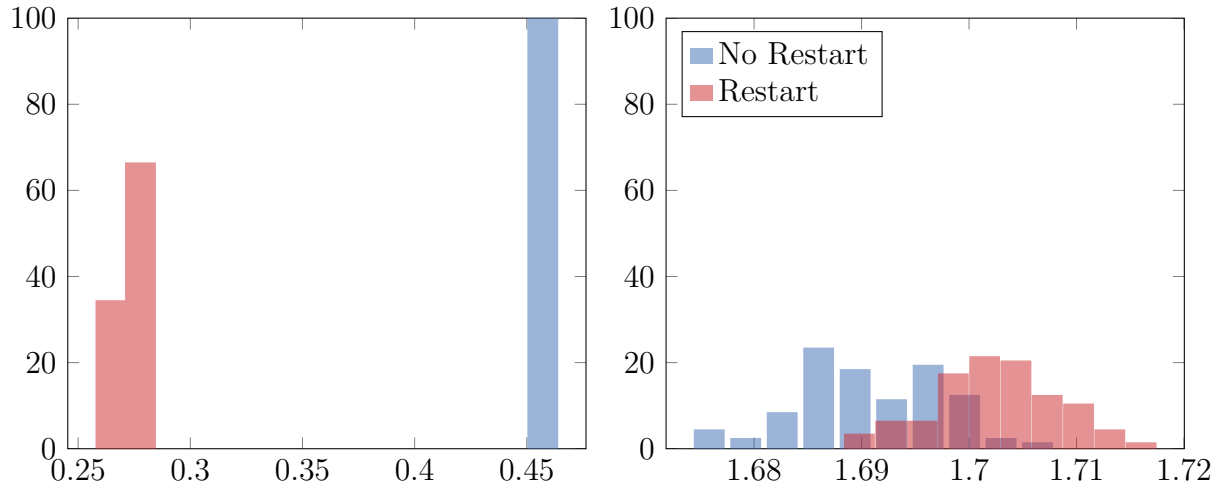
in the results obtained with ASTRA. Thus, we see that NCP is more consistent compared to using ASTRA's projector pair. Not using restarted GMRES results in almost 100% having an iteration ratio $R_{\text{i,ncp}} = 0.28$ while using restarted GMRES results in all $R_{\text{i,ncp}}$ being between 0.16 and 0.18. The error ratios show that the relative error obtained for $\mathbf{x}_{\text{ncp}}$ are twice as large as for $\mathbf{x}_{\text{opt}}$. The results for AB-GMRES are shown in Appendix B.5.2 Figure B.30. For $p = \infty$, 100% of the iteration ratios are at 0.55, thus, NCP is here consistent. For $p = 5$, the iteration ratios are distributed between 0.2 to 0.3. In both cases, $R_{\text{e,ncp}}$ is around 1.4 (for $p = \infty$) and 1.55 (for $p = 5$).



(a) The ratios $R_{i,\text{ncp}}$ for BA-GMRES. **(b)** The ratios $R_{e,\text{ncp}}$ for BA-GMRES.

Figure 5.19: A histogram showing the ratios (5.3) to the left and (5.4) to the right when using BA-GMRES to solve the unmatched CT problem obtained from the user domain with the forward projector being the line model and for fan beam geometry. We both consider $p = \infty$ (referred to as no restart) and $p = 5$ (referred to as restart).

If we use the strip model instead of the line model we obtain the results for BA-GMRES shown in Figure 5.20. We see that NCP is consistent and underestimate less compared to using the line model (Figure 5.19), however, it is still not satisfying results as NCP stops too early. The results for AB-GMRES are shown in Appendix B.5.2 Figure B.31 and it looks similar to the BA-GMRES results. The only difference is that NCP underestimates less as $R_{\text{i,ncp}} \approx 0.69$ for $p = \infty$ and for $p = 5$ $R_{\text{i,ncp}}$ is between 0.35 and 0.45.



(a) The ratios $R_{i,\text{ncp}}$ for BA-GMRES.

(b) The ratios $R_{e,\text{ncp}}$ for BA-GMRES.

Figure 5.20: A histogram showing the ratios (5.3) to the left and (5.4) to the right when using BA-GMRES to solve the unmatched CT problem obtained from the user domain with the forward projector being the strip model with 10 rays per strip and for fan beam geometry. We both consider $p = \infty$ (referred to as no restart) and $p = 5$ (referred to as restart).

5.2.2.3 Conclusion

We have summarized the results in the Tables below. Table 5.3 shows the consistency of NCP using the different projector pairs in the ABBA methods with and without restart. Citation signs are used around results that have slight variations but still overall looks consistent. We can conclude that NCP is consistent when using the projector pairs from the user domain. Moreover, NCP is only consistent for AB-GMRES when using the projector pair from ASTRA with parallel beam geometry.

		AB(∞)	AB(5)	BA(∞)	BA(5)
ASTRA GPU	Parallel beam	✓	"✓"	✗	✗
	Fan beam	✗	✗	✗	✗
User domain	Line	✓	"✓"	✓	"✓"
	Strip	✓	"✓"	✓	✓

Table 5.3: Overview of the consistency for NCP.

In general, NCP severely underestimates the optimal number of iterations leading to bad reconstructions, see Table 5.4. All solutions are between 15-120% different from the optimal solution. Thus, we can conclude that NCP is not a good stopping criterion for the ABBA methods.

		AB(∞)	AB(5)	BA(∞)	BA(5)
ASTRA GPU	Parallel beam	45%	60%	15% - 110%	15% - 110%
	Fan beam	20% - 100%	20% - 100%	20% - 120%	20% - 120%
User domain	Line	40%	55%	103% - 107%	105% - 109%
	Strip	20% - 22%	30% - 32%	67% - 71%	69% - 72%

Table 5.4: Overview of the difference between $\mathbf{x}_{\text{ncp}}$ and $\mathbf{x}_{\text{opt}}$.

5.3 Summary

In this chapter, we have studied the influence on the convergence rate when using restarted GMRES. Moreover, we have studied the two stopping criteria DP and NCP.

Applying restarted GMRES reduced memory limitations. Furthermore, we saw that restarted GMRES results in slower convergence compared to using regular GMRES. Hence, using restarted GMRES makes the ABBA methods more stable to a stopping criterion which overestimates the optimal number of iterations. Moreover, we found that there is a trade-off between memory and speed as using a low restart parameter can be more time-consuming compared to not using restarted GMRES if few iterations are needed to reach the optimum.

From the analysis of the stopping criteria, we found that both DP and NCP stop too early and, thereby, underestimate the optimal number of iterations. For DP, we did not see a significant difference between using the true standard deviation $\eta = \sigma$ and the estimated $\eta = \text{mean}(\boldsymbol{\sigma})$. Thereby, DP turned out to be fairly stable, even if we only had a good estimate of the error. When using regular GMRES, DP is consistent but when applying restart DP becomes inconsistent. NCP roughly underestimated the optimal number of iterations. Using projector pairs from the user domain resulted in NCP being consistent while the projector pairs from ASTRA resulted in NCP being inconsistent. The results are summarized in Table 5.5 and it is clear that NCP is not well suited for the ABBA methods.

	Consistent	Error
DP	✓*	1% - 14%
NCP	✗	15% - 120%

Table 5.5: An overview of the stopping criteria considered. Illustrating if they are consistent and how close to the optimal solution they came. * indicates that DP was only consistent for GMRES(∞).

CHAPTER 6

Real Computed Tomography Data

So far, we have used simulated CT data to construct a CT problem (more specifically, the Shepp Logan phantom) but now we will use our ABBA iterative methods on real data. When using real data, it is not possible to compare our reconstruction from the ABBA methods with the ground truth. We will, therefore, start by considering a data set that includes a filtered back projection (FBP) reconstruction $\mathbf{x}_{\text{fbp}}$ and use that as our ground truth. The FBP reconstruction is not necessarily the exact solution $\bar{\mathbf{x}}$, however, we assume it is close and can give a hint that the ABBA methods are close to the optimal solution.

We will consider using the regular and restarted ABBA methods. For both cases, we will also consider using the stopping criteria DP and NCP. Moreover, this chapter uses the ASTRA toolbox to construct an unmatched projector pair as it is the fastest (uses the GPU), and based on the results from the simulated data, we also did not see a big difference between using the different projector pairs.

Chapter 1 introduced a data set that caused CGLS to not converge when using an unmatched projector pair. We would like to demonstrate the advantages of using the ABBA methods and, thus, we will at the end of this chapter study the reconstructions from the ABBA methods on the same data. We demonstrate how increasing the number of iterations affects the reconstructions in the same fashion as shown in Figure 1.1 from Chapter 1.

The structure of the chapter follows:

- Study the reconstructions from the ABBA methods using regular GMRES, restarted GMRES, and the stopping criteria DP and NCP.
- Investigate the ABBA methods on an example where CGLS does not converge.

6.1 Real Data with a FBP Reconstruction

The Finnish Inverse Problems Society held the Helsinki Tomography Challenge 2022. In connection with this, real CT data has been published and can be found at [19]. We have chosen to work with the `htc2022_td` data which is collected with a fan beam geometry. They provide a sinogram $\mathbf{b}$ and a filtered back projection (FBP) reconstruction $\mathbf{x}_{\text{fbp}}$. We will solve the CT problem with the ABBA methods and compare the reconstructions with the provided FBP reconstruction.

To construct the forward and back projectors, we use ASTRA GPU meaning the forward projector is Joseph's model. Figure 6.1 show the convergence history for AB/BA-GMRES(∞). The ABBA methods achieve the solution closest to $\mathbf{x}_{\text{fbp}}$ at iteration 21 for AB-GMRES(∞) and at iteration 24 for BA-GMRES(∞). The relative error for AB-GMRES(∞) is 8.17% and for BA-GMRES(∞) it is 8.10%.

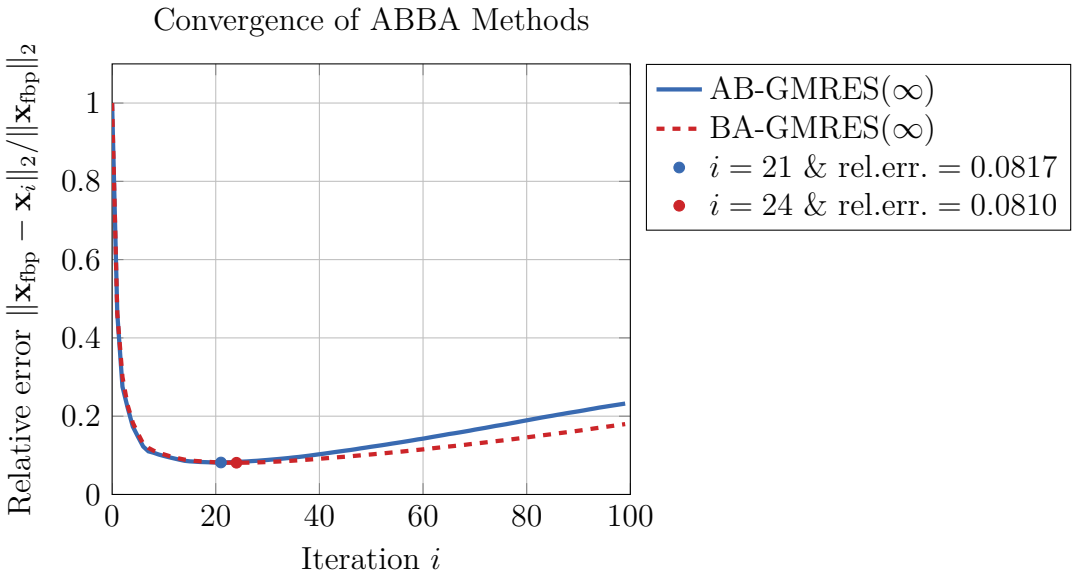


Figure 6.1: The convergence history of AB-GMRES(∞) and BA-GMRES(∞) when comparing the reconstructions from the ABBA methods with $\mathbf{x}_{\text{fbp}}$.

The reconstructions at the optimal solution are shown in Figure 6.2 beside the FBP reconstruction. From the figure, we see that the FBP reconstruction does not have a constant "background" as the corners are light (pixel values greater than 0) which we do not have in the reconstructions from the ABBA methods. This may influence the relative error and if we were to only consider the object of interest then the relative error may be smaller. Furthermore, it looks like the reconstructions include some beam hardening artifacts as the middle part of the image looks a little darker but not to a severe extent and we assume that the error comes from the assumption that our X-rays are monochromatic.

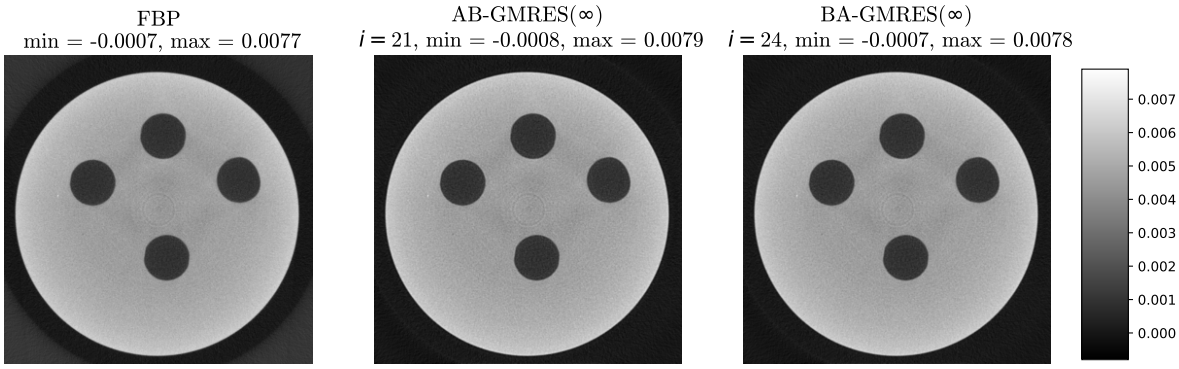


Figure 6.2: To the left, the FBP reconstruction from Helsinki Tomography Challenge 2022 is shown. In the middle, the best reconstruction from AB-GMRES(∞) is shown and to the right, the best reconstruction from BA-GMRES(∞) is shown.

6.1.1 Using Restarted GMRES

We consider applying restarted GMRES. Figure 6.3 shows the convergence history for the ABBA methods when $p = 5$. Both methods use twice as many iterations as AB/BA-GMRES(∞) and obtain a very small decrease in the relative error compared to AB/BA-GMRES(∞). However, the effects of semi-convergence are hardly visible compared to AB/BA-GMRES(∞). We have also tried with $p = 3$ and $p = 10$ (see Appendix B.6), and the results show the same behavior but when $p = 3$ the optimal solution appears very late (iteration 83 and 96), and for $p = 10$ it happens already at iteration 27 and 34. The relative error is approximately the same as seen in Figure 6.3.

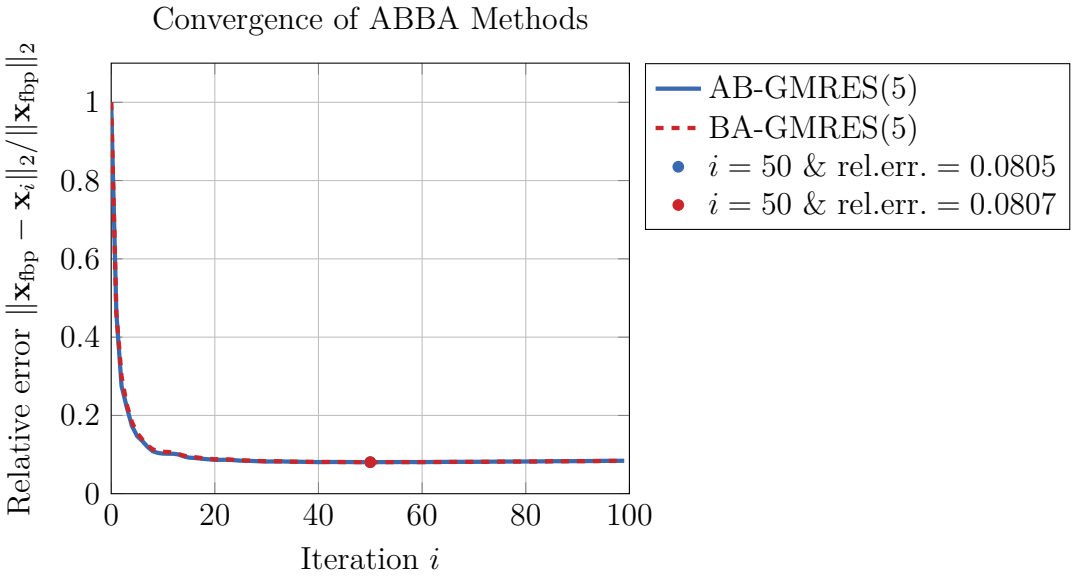


Figure 6.3: The convergence history of AB-GMRES(5) and BA-GMRES(5) when comparing the reconstructions from the ABBA methods with $\mathbf{x}_{\text{fbp}}$.

Figure 6.4 shows the reconstructions when using $p = 5$ alongside the FBP reconstruction.

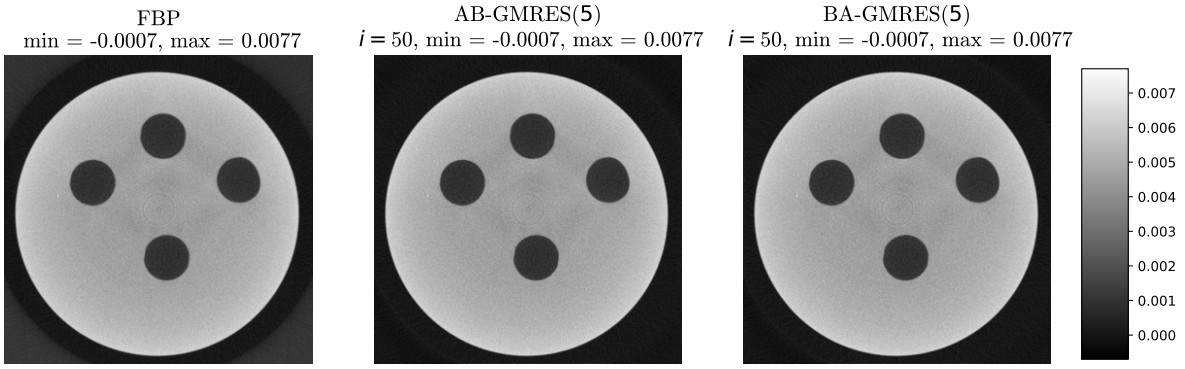


Figure 6.4: To the left, the FBP reconstruction from Helsinki Tomography Challenge 2022 is shown. In the middle, the best reconstruction from AB-GMRES(5) is shown and to the right, the best reconstruction from BA-GMRES(5) is shown.

6.1.2 Using Stopping Criteria

Using DP as a stopping criterion entails that we need to determine the noise level. We do not have any information about the noise level and, therefore, we need to do an approximation. This is done by taking out the first 20 columns in the sinogram which we by a visual inspection can see does not hit the object and, therefore, should be zero if no noise were present. Then we take the standard deviation of the extracted data and end up with $\eta \approx 0.00455$. DP is very sensitive to the choice of η and is very likely to take an extensive amount of iterations if the safety factor is not high enough. We choose $\nu_{\text{dp}} = 3$ as $\nu_{\text{dp}} = 1.02$ and $\nu_{\text{dp}} = 2$ make the ABBA methods take the maximum number of iterations.

Figure 6.5 and 6.6 show the reconstruction $\mathbf{x}_{\text{dp}}$ when having AB/BA-GMRES(∞) and when using the restarted ($p = 5$) ABBA methods, respectively. In both cases, we see that DP stops too early and that the reconstructions are more blurry than $\mathbf{x}_{\text{fbp}}$.

Considering the NCP stopping criterion, we obtain the results shown in Figure 6.7 for AB/BA-GMRES(∞) and Figure 6.8 for AB/BA-GMRES(5). Using regular GMRES leads to the ABBA methods stopping at iterations 9 and 7 with the optimum being at iterations 21 and 24, respectively. Using restarted GMRES, NCP stops at iteration 6 while the optimal iteration is 50. Thus, NCP severely underestimates the number of iterations required. Inspecting the reconstructions, we see that their quality is much worse than $\mathbf{x}_{\text{fbp}}$, and many details are blurred.

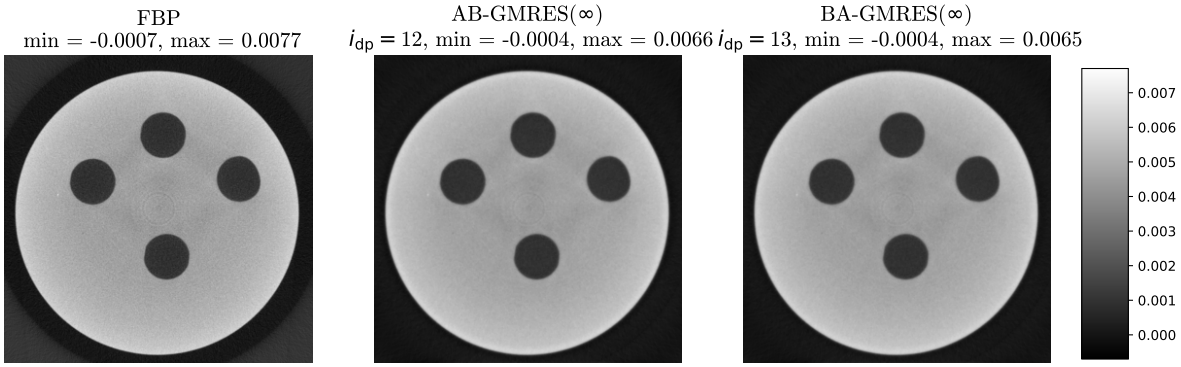


Figure 6.5: To the left, the FBP reconstruction from Helsinki Tomography Challenge 2022 is shown. In the middle, the reconstruction from AB-GMRES(∞) when using DP is shown and to the right, the reconstruction when using DP in BA-GMRES(∞) is shown.

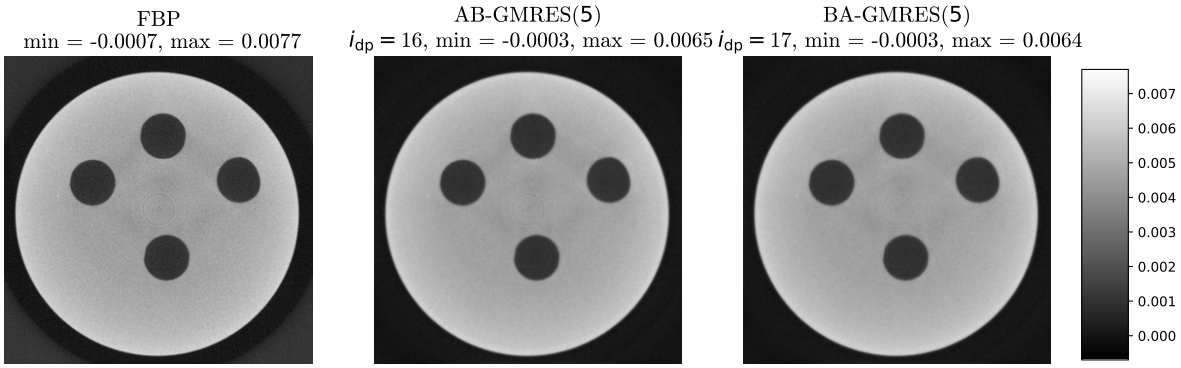


Figure 6.6: To the left, the FBP reconstruction from Helsinki Tomography Challenge 2022 is shown. In the middle, the reconstruction from AB-GMRES(5) when using DP is shown and to the right, the reconstruction when using DP in BA-GMRES(5) is shown.

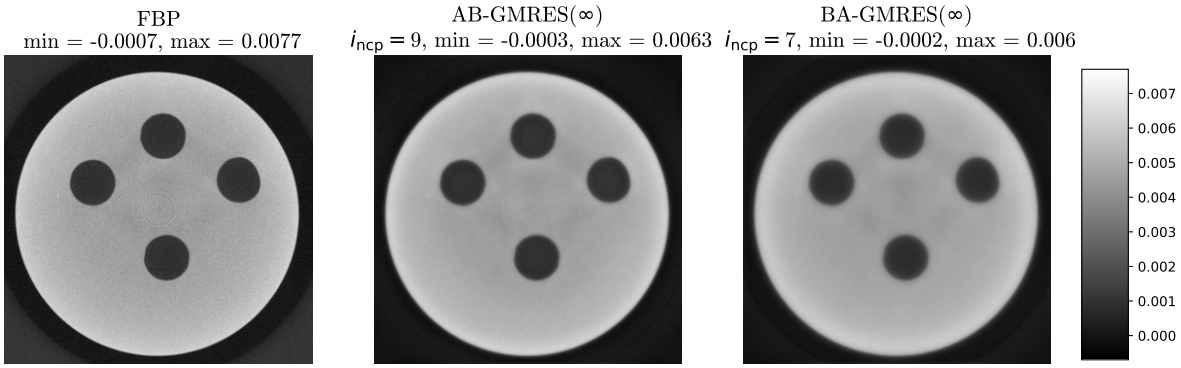


Figure 6.7: To the left, the FBP reconstruction from Helsinki Tomography Challenge 2022 is shown. In the middle, the reconstruction from AB-GMRES(∞) when using NCP is shown and to the right, the reconstruction when using NCP in BA-GMRES(∞) is shown.

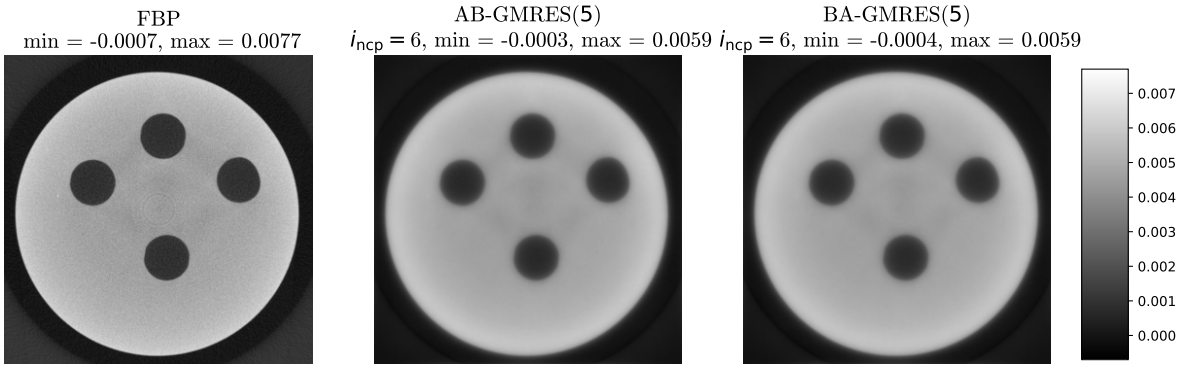


Figure 6.8: To the left, the FBP reconstruction from Helsinki Tomography Challenge 2022 is shown. In the middle, the reconstruction from AB-GMRES(5) when using NCP is shown and to the right, the reconstruction when using NCP in BA-GMRES(5) is shown.

6.2 Lego Gandalf Data

In Chapter 1, we looked at a cross-section of a Lego Gandalf figure (see Figure 1.1). The reconstructions were made with CGLS and we saw that it did not converge as CGLS is not guaranteed to converge when having unmatched projector pairs. In this section, we will use the ABBA methods to reconstruct the interior of the Lego Gandalf.

We will use the same approach as used in Figure 1.1, meaning, we will look at the solution for different numbers of iterations. We will not consider using stopping criteria as we have already concluded they do not estimate the optimal number of iterations well. Furthermore, we will not look at using restarted GMRES for this example, as restart is a tool to lower the memory cost which is not needed here.

CGLS and the ABBA methods do not converge with the same speed, so we will consider a different number of iterations than shown in Figure 1.1. We start by looking at the reconstructions for iterations 5, 10, 20, and 40 as previous results have shown fast convergence. In the end, we will also consider looking at the reconstructions when using 80 iterations to see how the reconstruction looks when taking many iterations (just as is done for the CGLS example).

Figure 6.9 show the reconstructions from AB-GMRES(∞) and Figure 6.10 show the reconstructions from BA-GMRES(∞). At first, the reconstructions are blurry but as the number of iterations increases, the reconstruction becomes sharper. Figure 6.11 shows the reconstructions at iteration 80 for the ABBA methods, and we see that the minimum and maximum values have increased in magnitude compared to the smaller iterations (5-40). Contrary to CGLS, the results show that the ABBA methods do converge to a proper solution before beginning to semi-converge.

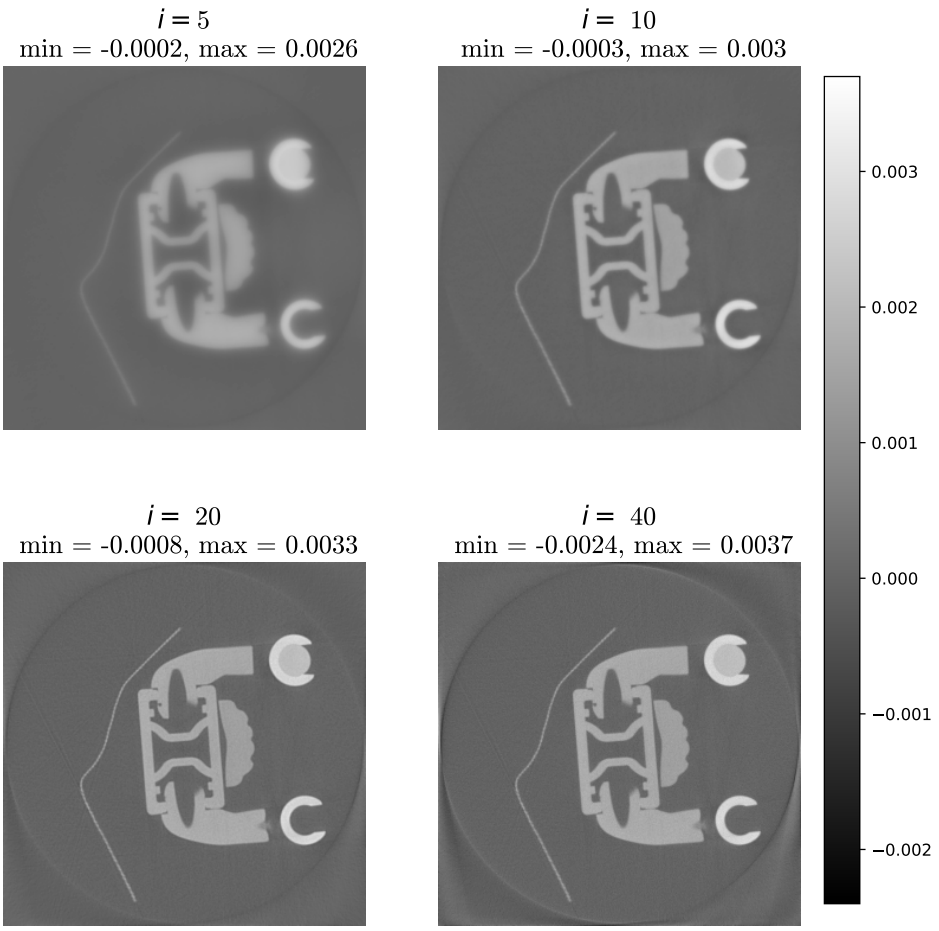


Figure 6.9: The AB-GMRES(∞) reconstructions when using the unmatched projector pair from ASTRA for iterations 5, 10, 20, and 40.

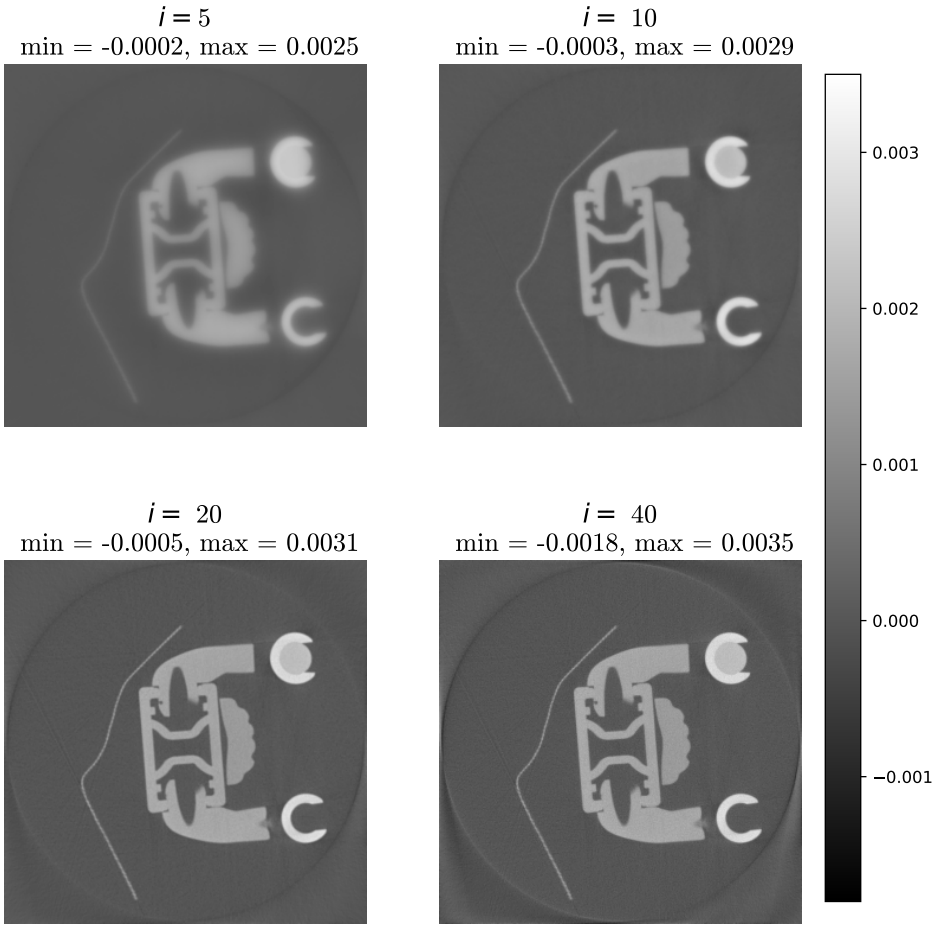


Figure 6.10: The BA-GMRES(∞) reconstructions when using the unmatched projector pair from ASTRA for iterations 5, 10, 20, and 40.

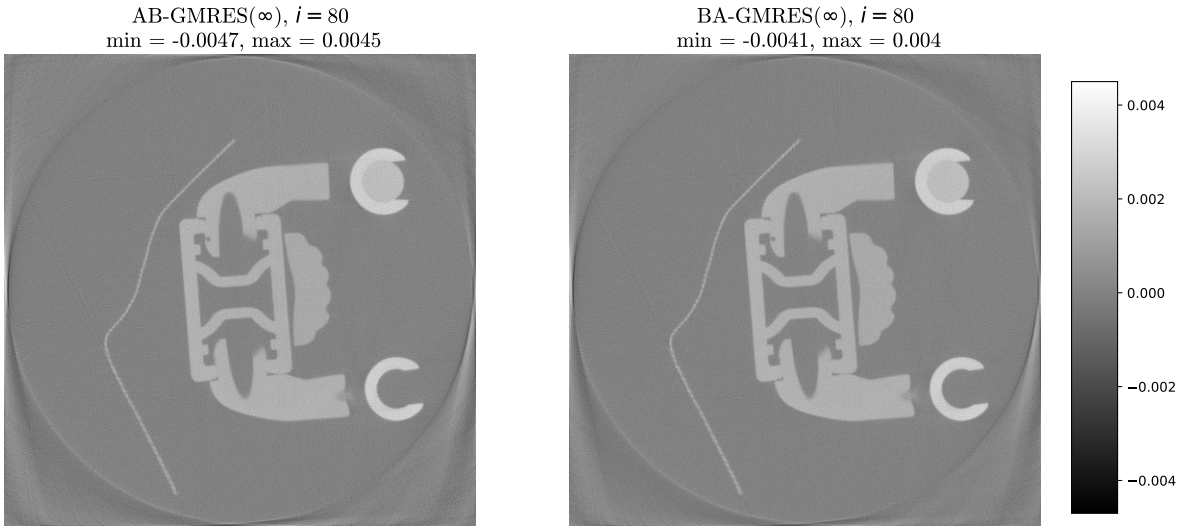


Figure 6.11: The ABBA reconstructions for 80 iterations.

6.3 Summary

Up until this chapter, we have only considered simulated CT problems. In this chapter, we have considered two real-world data sets. The major difference between real data and simulated is that we do not have the true solution for comparison and do not have a good estimate of the noise within the problem.

For one of the data sets, we were able to compare the reconstructions from the ABBA methods with a FBP reconstruction which resulted in the possibility of looking at convergence curves and determining an optimum. Looking at the reconstructions for regular and restarted GMRES, they were very close to the FBP reconstruction. Restarted GMRES almost removed the consequence of semi-convergence. Using DP and NCP, the reconstructions were blurry due to the stopping criteria underestimating the optimal number of iterations.

Finally, we considered the CT problem stated in the Introduction (see Figure 1.1) in which the CGLS method failed to converge and the results showed convergence of the ABBA methods.

CHAPTER 7

Discussion

The foundation of this thesis is based on iterative methods failing to solve CT problems with an unmatched projector pair. Such problems arise when wanting to increase the performance efficiency as this involves matrix-free implementations of the forward and back projectors resulting in different discretization schemes/model approximations.

The ABBA iterative methods are well-suited for matrix-free implementations and are guaranteed to converge. This thesis has investigated the convergence of the ABBA methods when noise is present in the measurements. CT problems are usually large-scale problems and to reduce the memory requirement, we have considered restarting the ABBA methods and investigated the effect on the convergence. Moreover, we have considered two stopping criteria, namely DP and NCP. This chapter will discuss the results obtained from the study.

7.1 The Unmatched Projector Pair

The purpose of this thesis has been to examine the ABBA iterative methods when having unmatched projector pairs. To create unmatched projector pairs we have utilized existing Python implementations of forward and back projectors from public toolboxes as well as code provided by Professor Emil Y. Sidky (referred to as the user domain). The thesis has been challenged by a lack of documentation for the implementation of projectors. This applies both to mathematical descriptions of back projectors for fan beam geometry and also to the limited documentation for the public toolboxes.

In the process of constructing unmatched projector pairs, we experienced some issues with one of the forward projectors in TIGRE (Joseph's model or in TIGRE called interpolated). Comparing the corresponding forward projector from ASTRA, we found that parts of the X-rays were missing in the forward projection from TIGRE. Finding the root of the problem can be time-consuming and since the purpose of the project is to investigate the ABBA methods, we chose not to work further with TIGRE. Thus, the public unmatched projector pairs considered in this thesis originated from ASTRA's GPU based implementation of Joseph's model as the forward projector. We have no mathematical description of the back projector but the main programmer of ASTRA, Dr. Willem Jan Palenstijn, referred to the back projector as being "(mostly) exactly the

same model as for the parallel beam case" described in Section 2.3.4.

ASTRA includes an unmatched projector pair for parallel and fan beam geometry. To determine how unmatched the forward ($\mathbf{A}$) and back ($\mathbf{B}$) projectors are, we have computed the relative difference between $\mathbf{A}$ and $\mathbf{B}^T$. Having parallel beam geometry results in the projector pair having a relative difference of 41% and having fan beam geometry leads to a relative difference of 16%. The user domain included two projector pairs for fan beam geometries. The first is the line model as the forward projector with a relative difference of 31% to the back projector and the second is an approximation of the strip model as the projector pair has a 35% relative difference. Again, the back projector is in principle the same as in Section 2.3.4 but for fan beam geometry instead of parallel beam.

Lastly, using ASTRA's projectors means that the core operations are performed on a GPU while using the user domain yields performing our operations on the CPU.

7.2 Performance of the ABBA Iterative Methods

From our studies, we found that the ABBA iterative methods obtain semi-convergence when noise is present in the measurements. To reduce the memory requirement of the ABBA solvers, we introduced restarted GMRES where the restart parameter p determines how many iterations the ABBA methods will take before restarting. We also here obtained semi-convergence, however, applying restart entails a decreased convergence rate which depends on the value of the restart parameter p; a decreasing p-value yields a decreasing convergence rate. So, there is a trade-off between stability and computational speed as a decreasing p entails increasing the optimal number of iterations. The relative error at the optimum for restarted GMRES was slightly smaller than for the regular GMRES, however, in the order of 10^{-3}.

Decreasing the restart parameter p yields increasing the number of FLOPs needed to reach the optimum. However, this is not reflected in time as regular GMRES reached the optimum at approximately the same time as restarted GMRES with $p = 5, 10$ (looking at the GPU based operators). Computing the maximum number of iterations leads to restarted GMRES being the fastest in time and computing the fewest FLOPs.

Applying the stopping criterion DP with the safety factor $\nu_{\text{dp}} = 1.02$, we experienced stopping too early (which is also concluded in [17, Ch. 11, Fig. 11.15]) both for the regular and restarted ABBA methods. We considered two cases, one where we used the true noise level and one where we from 100 realizations of the noise computed an approximation of the noise level. DP obtained similar behavior for both cases. Furthermore,

we found that DP is consistent when using AB/BA-GMRES(∞) but inconsistent using AB/BA-GMRES(5). Generally, the solutions obtained from DP, $\mathbf{x}_{\text{dp}}$, was 1% to 14% different from the optimal solution.

Considering the NCP stopping criterion, we found that it severely underestimates the optimal number of iterations leading to $\mathbf{x}_{\text{ncp}}$ being 15% to 120% different from the optimal solution. Moreover, it was problem dependent if NCP were consistent or not. We obtained consistency for AB/BA-GMRES(∞) and AB/BA-GMRES(5) when using the projector pairs from the user domain. Using ASTRA, we only had consistency when using regular and restarted AB-GMRES for parallel beam geometry. In all other cases with ASTRA, NCP was inconsistent.

7.3 Reconstruction with Real Data

The thesis considered using the ABBA methods with all its extensions on real 2D data. Thus, we do not have ground truth or information about the noise level.

The data from Helsinki Tomography Challenge 2022 included a FBP reconstruction that we used as our ground truth. Thus, we were able to make convergence curves and compare the number of iterations taken by DP and NCP to the optimal number. We obtained reconstructions with the ABBA methods which were 92% similar to the FBP reconstruction both with the regular and restarted ABBA methods.

To use DP as a stopping criterion, we needed to estimate the noise level η. It was done by computing the standard deviation in a snip of the sinogram which we knew only included background noise. As DP is very sensitive to the choice of η, we needed to adjust the safety factor and chose $\nu_{\text{dp}} = 3$. Using DP on the regular ABBA methods resulted in the number of iterations being halved compared to the optimum. For the restarted ABBA methods, DP underestimated even more. To make DP stop later, we could tweak the safety factor, however, it ruins the idea of using DP. We would like DP to be stable and not need to tweak any parameters otherwise we gain nothing by using it compared to just using a maximum number of iterations.

Using NCP resulted in severe underestimations of the optimal parameter (more than DP). Thus, the reconstruction quality was much worse compared to the FBP reconstruction as many details were blurred.

Lastly, we considered a CT problem with an unmatched projector pair that resulted in CGLS failing to converge (see Figure 1.1 from the Introduction). We demonstrated that the ABBA methods were able to solve the problem as we saw semi-convergence.

7.4 Future Work

The search for a robust stop criterion is, in general, a focus area, however, sometimes in some applications some stop criteria work well. Based on the results from this thesis, we have not been able to find a robust stopping criterion and, thus, it is a topic of future research to find or develop a robust stopping criterion for the ABBA methods.

Moreover, we could consider other back projector operators to accelerate the computations. An example could be using the filtered back projection which is discussed in [27]. A filtered back projection contributes to a more sharp reconstruction and an increased convergence rate.

Finally, this thesis has focused on 2D CT problems which did not require all available memory. The next step would be to increase the problem size and do numerical experiments on 3D problems for simulated and real data. Furthermore, we would need to study how the different stopping criteria respond to such problems. To solve 3D problems, we would need to modify the GitHub code provided. We would need to change the setup for TIGRE as it is hard-coded to be of two dimensions and we would also need to change the ASTRA setup as more specifications are needed (see Documentation ➔ 3D Geometries in [4]).

CHAPTER 8

Conclusion

In this thesis, we performed an exploratory study of the AB-GMRES and BA-GMRES (ABBA) iterative methods. We investigated the convergence when having noisy measurements for miscellaneous unmatched projector pairs with different degrees of diversity. To reduce the memory requirement, we studied restarted GMRES and its effect on the convergence. Furthermore, we explored the two stopping criteria DP and NCP.

Based on our study, we found that the ABBA methods obtained semi-convergence when having noisy measurements. The use of different unmatched projector pairs from ASTRA and the user domain resulted in different convergence rates. However, we did not see a relation between the degree of diversity and the convergence rate.

From the results, we can conclude that AB/BA-GMRES(∞) converges the fastest and, therefore, reaches the optimum first, where the convergence rate of AB/BA-GMRES(p) converges slower resulting in needing more iterations to reach the optimum. The computational time of AB/BA-GMRES(∞) in FLOPs and seconds are smaller or the same as for AB/BA-GMRES(p) if we can stop at the optimum. However, the advantage of restarted GMRES appears if we do not have a robust stopping criterion, need more iterations to reach the optimum than needed in the examples covered in this thesis, or if we run out of memory. In those cases, we prefer to consider the restarted ABBA methods. The advantages of using restarted GMRES are:

- Reduced memory requirement (depend on the restart parameter p).
- The ABBA methods are less sensitive to stopping criteria that overestimate the optimal number of iterations.
- The Krylov subspace is kept ”small” which lowers the cost of evaluating it.

On the contrary, the disadvantage of using restarted GMRES is that the ABBA methods need more iterations to reach the optimum. Thus, we can conclude that choosing the best number of iterations before restart (p) depends on the memory available and the behavior of the stopping criteria.

Lastly, we can conclude that both NCP and DP stopped too early. DP was consistent when using the ABBA methods without restart while it became inconsistent when we applied restart. The stability of NCP depended on the projector pair and not if we used restarted GMRES or not. We obtained stable results for the projectors from the user domain while we obtained unstable results for the public domain. Thus, we did not find a robust stopping criterion.

Based on the exploratory study, we can conclude that the ABBA iterative methods can solve CT problems with unmatched projector pairs.

APPENDIX A
Additional Theory

A.1 Mathematically Equivalent: GMRES and BA-GMRES

We aim to prove that GMRES and BA-GMRES are mathematically equivalent. GMRES is given in Algorithm 1 and BA-GMRES is given in Algorithm 3. To show that the algorithms are equivalent we consider the two first iterations $k = 1, 2$ for both methods and look at the minimization problem for $\mathbf{y}_k$. If the methods are equivalent, then $\mathbf{y}_k$ for both methods should be equivalent.

GMRES needs to solve the unmatched normal equations (3.26) which leads to $\mathbf{M} = \mathbf{BA}$ and $\mathbf{d} = \mathbf{Bb}$. It applies to both methods that $\mathbf{x}_0 = \mathbf{0}$. Thus, we have that for both methods the initial variables are

$$\mathbf{r}_0 = \mathbf{Bb} - \mathbf{BAx}_0 = \mathbf{Bb}, \tag{A.1}$$

and

$$\mathbf{w}_1 = \frac{\mathbf{r}_0}{\|\mathbf{r}_0\|_2} = \frac{\mathbf{Bb}}{\|\mathbf{Bb}\|_2}. \tag{A.2}$$

A.1.1 GMRES

We will consider two iterations ($k = 2$) of the GMRES method.

Iteration $j = 1$:

$$\mathbf{q}_1 = \mathbf{BAw}_1 \tag{A.3}$$

Iteration $i = 1$:

$$h_{1,1} = (\mathbf{q}_1, \mathbf{w}_1) = (\mathbf{BAw}_1, \mathbf{w}_1) = (\mathbf{BAw}_1)^T \mathbf{w}_1, \tag{A.4}$$

$$\mathbf{q}_1 = \mathbf{q_1} - h_{1,1}\mathbf{w}_1 = \mathbf{BAw}_1 - h_{1,1}\mathbf{w}_1 = (\mathbf{BA} - h_{1,1}\mathbf{I})\mathbf{w}_1. \tag{A.5}$$

We are done with iterating over i and continue in the for-loop for j. We then obtain

$$h_{2,1} = \|\mathbf{q}_1\|_2 = \|(\mathbf{BA} - h_{1,1}\mathbf{I})\mathbf{w}_1\|_2 = \|(\mathbf{BA} - (\mathbf{BAw}_1)^T \mathbf{w}_1\mathbf{I})\mathbf{w}_1\|_2, \tag{A.6}$$

$$\mathbf{w}_2 = \frac{\mathbf{q}_1}{h_{2,1}} = \frac{(\mathbf{BA} - h_{1,1}\mathbf{I})\mathbf{w}_1}{\|(\mathbf{BA} - h_{1,1}\mathbf{I})\mathbf{w}_1\|_2}, \tag{A.7}$$

and the solution $\mathbf{y}_1$ yields

$$\mathbf{y}_1 = \underset{\mathbf{y}}{\operatorname{argmin}} \left\| \begin{pmatrix} \|\mathbf{r}_0\|_2 \\ 0 \end{pmatrix} - \begin{pmatrix} (\mathbf{BAw}_1)^T\mathbf{w}_1 \\ \|(\mathbf{BA} - (\mathbf{BAw}_1)^T\mathbf{w}_1\mathbf{I})\mathbf{w}_1\|_2 \end{pmatrix} \mathbf{y} \right\|_2. \tag{A.8}$$

Iteration $j = 2$:

$$\mathbf{q}_2 = \mathbf{BAw}_2. \tag{A.9}$$

Iteration $i = 1$:

$$h_{1,2} = (\mathbf{q}_2, \mathbf{w}_1) = (\mathbf{BAw}_2, \mathbf{w}_1) = (\mathbf{BAw}_2)^T\mathbf{w}_1, \tag{A.10}$$

$$\mathbf{q}_2 = \mathbf{q_2} - h_{1,2}\mathbf{w}_1 = \mathbf{BAw}_2 - h_{1,2}\mathbf{w}_1. \tag{A.11}$$

Iteration $i = 2$:

$$h_{2,2} = (\mathbf{q}_2, \mathbf{w}_2) = (\mathbf{BAw}_2, \mathbf{w}_2) = (\mathbf{BAw}_2)^T\mathbf{w}_2, \tag{A.12}$$

$$\mathbf{q}_2 = \mathbf{q_2} - h_{2,2}\mathbf{w}_2 = \mathbf{BAw}_2 - h_{1,2}\mathbf{w}_1 - h_{2,2}\mathbf{w}_2. \tag{A.13}$$

We have reached $i = j$ and continue in the for-loop for j where we obtain

$$\begin{aligned} h_{3,2} &= \|\mathbf{q}_2\|_2 = \|\mathbf{BAw}_2 - h_{1,2}\mathbf{w}_1 - h_{2,2}\mathbf{w}_2\|_2, \\ &= \|\mathbf{BAw}_2 - (\mathbf{BAw}_2)^T\mathbf{w}_1\mathbf{w}_1 - (\mathbf{BAw}_2)^T\mathbf{w}_2\mathbf{w}_2\|_2. \end{aligned} \tag{A.14}$$

The solution for $\mathbf{y}_2$ yields

$$\mathbf{y}_2 = \underset{\mathbf{y}}{\operatorname{argmin}} \left\| \begin{pmatrix} \|\mathbf{r}_0\|_2 \\ 0 \\ 0 \end{pmatrix} - \mathbf{H}_2\mathbf{y} \right\|_2, \tag{A.15}$$

where

$$\mathbf{H}_2 = \begin{pmatrix} (\mathbf{BAw}_1)^T\mathbf{w}_1 & (\mathbf{BAw}_2)^T\mathbf{w}_1 \\ \|(\mathbf{BA} - (\mathbf{BAw}_1)^T\mathbf{w}_1\mathbf{I})\mathbf{w}_1\|_2 & (\mathbf{BAw}_2)^T\mathbf{w}_2 \\ 0 & \|\mathbf{BAw}_2 - (\mathbf{BAw}_2)^T\mathbf{w}_1\mathbf{w}_1 - (\mathbf{BAw}_2)^T\mathbf{w}_2\mathbf{w}_2\|_2 \end{pmatrix}.$$

A.1.2 BA-GMRES

We will again for BA-GMRES consider two iterations such that we can compare our results with GMRES.

Iteration $k = 1$:

$$\mathbf{q}_1 = \mathbf{BAw}_1 \tag{A.16}$$

Iteration $i = 1$:

$$h_{1,1} = \mathbf{q}_1^T\mathbf{w}_1 = (\mathbf{BAw}_1)^T\mathbf{w}_1 \tag{A.17}$$

$$\mathbf{q}_1 = \mathbf{q}_1 - h_{1,1}\mathbf{w}_1 = \mathbf{BAw}_1 - h_{1,1}\mathbf{w}_1 = (\mathbf{BA} - h_{1,1}\mathbf{I})\mathbf{w}_1 \tag{A.18}$$

From the results obtained in the for-loop for i, we can compute

$$h_{2,1} = \|\mathbf{q}_1\|_2 = \|(\mathbf{BA} - h_{1,1}\mathbf{I})\mathbf{w}_1\|_2 = \|(\mathbf{BA} - (\mathbf{BAw}_1)^T\mathbf{w}_1\mathbf{I})\mathbf{w}_1\|_2, \tag{A.19}$$

$$\mathbf{w}_2 = \frac{\mathbf{q}_1}{h_{2,1}} = \frac{(\mathbf{BA} - h_{1,1}\mathbf{I})\mathbf{w}_1}{\|(\mathbf{BA} - h_{1,1}\mathbf{I})\mathbf{w}_1\|_2}, \tag{A.20}$$

and the solution $\mathbf{y}_1$ yields

$$\mathbf{y}_1 = \underset{\mathbf{y}}{\operatorname{argmin}} \left\| \begin{pmatrix} \|\mathbf{r}_0\|_2 \\ 0 \end{pmatrix} - \begin{pmatrix} (\mathbf{BAw}_1)^T\mathbf{w}_1 \\ \|(\mathbf{BA} - (\mathbf{BAw}_1)^T\mathbf{w}_1\mathbf{I})\mathbf{w}_1\|_2 \end{pmatrix} \mathbf{y} \right\|_2. \tag{A.21}$$

We have obtained the same minimization problem as for GMRES (A.8).

Iteration $k = 2$:

$$\mathbf{q}_2 = \mathbf{BAw}_2. \tag{A.22}$$

Iteration $i = 1$:

$$h_{1,2} = \mathbf{q}_2^T\mathbf{w}_1 = (\mathbf{BAw}_2)^T\mathbf{w}_1, \tag{A.23}$$

$$\mathbf{q}_2 = \mathbf{q}_2 - h_{1,2}\mathbf{w}_1 = \mathbf{BAw}_2 - h_{1,2}\mathbf{w}_1. \tag{A.24}$$

Iteration $i = 2$:

$$h_{2,2} = \mathbf{q}_2^T\mathbf{w}_2 = (\mathbf{BAw}_2)^T\mathbf{w}_2, \tag{A.25}$$

$$\mathbf{q}_2 = \mathbf{q}_2 - h_{2,2}\mathbf{w}_2 = \mathbf{BAw}_2 - h_{1,2}\mathbf{w}_1 - h_{2,2}\mathbf{w}_2. \tag{A.26}$$

We are done iterating in i and we obtain the final variable

$$\begin{aligned} h_{3,2} &= \|\mathbf{q}_2\|_2 = \|\mathbf{BAw}_2 - h_{1,2}\mathbf{w}_1 - h_{2,2}\mathbf{w}_2\|_2, \\ &= \|\mathbf{BAw}_2 - (\mathbf{BAw}_2)^T\mathbf{w}_1\mathbf{w}_1 - (\mathbf{BAw}_2)^T\mathbf{w}_2\mathbf{w}_2\|_2. \end{aligned} \tag{A.27}$$

Thus, the solution $\mathbf{y}_k$ yields

$$\mathbf{y}_2 = \underset{\mathbf{y}}{\operatorname{argmin}} \left\| \begin{pmatrix} \|\mathbf{r}_0\|_2 \\ 0 \\ 0 \end{pmatrix} - \mathbf{H}_2\mathbf{y} \right\|_2, \tag{A.28}$$

where

$$\mathbf{H}_2 = \begin{pmatrix} (\mathbf{BAw}_1)^T\mathbf{w}_1 & (\mathbf{BAw}_2)^T\mathbf{w}_1 \\ \|(\mathbf{BA} - (\mathbf{BAw}_1)^T\mathbf{w}_1\mathbf{I})\mathbf{w}_1\|_2 & (\mathbf{BAw}_2)^T\mathbf{w}_2 \\ 0 & \|\mathbf{BAw}_2 - (\mathbf{BAw}_2)^T\mathbf{w}_1\mathbf{w}_1 - (\mathbf{BAw}_2)^T\mathbf{w}_2\mathbf{w}_2\|_2 \end{pmatrix}.$$

This is the same result as we obtained with GMRES (A.15). Note that $\mathbf{w}_2$ is the same for BA-GMRES (A.20) and GMRES (A.7) as $h_{1,1}$ is the same in both algorithms (we already know $\mathbf{w}_1$ is the same due to $\mathbf{x}_0 = \mathbf{0}$).

We have now shown that BA-GMRES and GMRES are mathematically equivalent. The same can be done for AB-GMRES and GMRES which we will not show here.

A.2 Matrix-Vector Products: Overview

A.2.1 AB/BA-GMRES(∞)

Here we give an overview of the matrix-vector (MV) products needed to run the algorithms AB-GMRES(∞) and BA-GMRES(∞) (Algorithm 6 and 7, respectively). Note, we do not need to make a MV product in line 13 in AB-GMRES if we just reuse $\mathbf{Bw}_k$ from line 5 as $\mathbf{B}\begin{bmatrix}\mathbf{w}_1 & \mathbf{w}_2 & \cdots & \mathbf{w}_k\end{bmatrix} = \begin{bmatrix}\mathbf{Bw}_1 & \mathbf{Bw}_2 & \cdots & \mathbf{Bw}_k\end{bmatrix}$. If we do not need to compute the residual for each iteration k, we can avoid the one MV product in line 14. The total number of MV products for each algorithm is shown at the top with the name. The first number of MV products illustrates the total number when computing the residual for each iteration and the second is if we choose not to compute the residuals.

Algorithm 6 AB-GMRES(∞)
(3K + 1) MV or (2K + 1) MV

1: Choose initial $\mathbf{x}_0$
2: $\mathbf{r}_0 = \mathbf{b} - \mathbf{Ax}_0$ 1 MV
3: $\mathbf{w}_1 = \mathbf{r}_0/\|\mathbf{r}_0\|_2$
4: **for** $k = 1, 2, \ldots, K$ **do**
5: $\quad \mathbf{q}_k = \mathbf{ABw}_k$ 2 MV
6: $\quad$ **for** $i = 1, 2, \ldots, k$ **do**
7: $\quad\quad h_{i,k} = \mathbf{q}_k^T \mathbf{w}_i$
8: $\quad\quad \mathbf{q}_k = \mathbf{q}_k - h_{i,k}\mathbf{w}_i$
9: $\quad$ **end for**
10: $\quad h_{k+1,k} = \|\mathbf{q}_k\|_2$
11: $\quad \mathbf{w}_{k+1} = \mathbf{q}_k/h_{k+1,k}$
12: $\quad \mathbf{y}_k = \text{argmin}_{\mathbf{y}} \|\|\mathbf{r_0}\|_2\mathbf{e}_1 - \mathbf{H}_k\mathbf{y}\|_2$
13: $\quad \mathbf{x}_k = \mathbf{x}_0 + \mathbf{B}\begin{bmatrix}\mathbf{w}_1 & \mathbf{w}_2 & \cdots & \mathbf{w}_k\end{bmatrix}\mathbf{y}_k$
14: $\quad \mathbf{r}_k = \mathbf{b} - \mathbf{Ax}_k$ 1 MV
15: $\quad$ Stopping rule goes here
16: **end for**

Algorithm 7 BA-GMRES(∞)
(3K + 2) MV or (2K + 2) MV

1: Choose initial $\mathbf{x}_0$
2: $\mathbf{r}_0 = \mathbf{B}(\mathbf{b} - \mathbf{Ax}_0)$ 2 MV
3: $\mathbf{w}_1 = \mathbf{r}_0/\|\mathbf{r}_0\|_2$
4: **for** $k = 1, 2, \ldots, K$ **do**
5: $\quad \mathbf{q}_k = \mathbf{BAw}_k$ 2 MV
6: $\quad$ **for** $i = 1, 2, \ldots, k$ **do**
7: $\quad\quad h_{i,k} = \mathbf{q}_k^T \mathbf{w}_i$
8: $\quad\quad \mathbf{q}_k = \mathbf{q}_k - h_{i,k}\mathbf{w}_i$
9: $\quad$ **end for**
10: $\quad h_{k+1,k} = \|\mathbf{q}_k\|_2$
11: $\quad \mathbf{w}_{k+1} = \mathbf{q}_k/h_{k+1,k}$
12: $\quad \mathbf{y}_k = \text{argmin}_{\mathbf{y}} \|\|\mathbf{r_0}\|_2\mathbf{e}_1 - \mathbf{H}_k\mathbf{y}\|_2$
13: $\quad \mathbf{x}_k = \mathbf{x}_0 + \begin{bmatrix}\mathbf{w}_1 & \mathbf{w}_2 & \cdots & \mathbf{w}_k\end{bmatrix}\mathbf{y}_k$
14: $\quad \mathbf{r}_k = \mathbf{b} - \mathbf{Ax}_k$ 1 MV
15: $\quad$ Stopping rule goes here
16: **end for**

A.2.2 AB/BA-GMRES(p)

Algorithm 8 and 9 show the number of MV products in AB-GMRES(p) and BA-GMRES(p), respectively. Here we cannot avoid computing $\mathbf{r}_k$ in all iterations, as we need it when restarting $k = p$. Thus, for each period l we need to make at least one MV product when computing $\mathbf{r}_{k=p}$ (illustrated in the second term in the header). If we choose to compute $\mathbf{r}_k$ in each iteration, then the number of MV products is given in the first term in the header of the algorithms.

Algorithm 8 AB-GMRES(p)
(3pL + 1) MV or (2pL + L + 1) MV

Choose initial $\mathbf{x}_0$
$\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$ 1 MV
for $l = 1, 2, \ldots, L$ **do**
  $\mathbf{w}_1 = \mathbf{r}_0 / \|\mathbf{r}_0\|_2$
  for $k = 1, 2, \ldots, p$ **do**
    $\mathbf{q}_k = \mathbf{A}\mathbf{B}\mathbf{w}_k$ 2 MV
    for $i = 1, 2, \ldots, k$ **do**
      $h_{i,k} = \mathbf{q}_k^T \mathbf{w}_i$
      $\mathbf{q}_k = \mathbf{q}_k - h_{i,k}\mathbf{w}_i$
    end for
    $h_{k+1,k} = \|\mathbf{q}_k\|_2$
    $\mathbf{w}_{k+1} = \mathbf{q}_k / h_{k+1,k}$
    $\mathbf{y}_k = \text{argmin}_{\mathbf{y}} \|\|\mathbf{r_0}\|_2 \mathbf{e}_1 - \mathbf{H}_k\mathbf{y}\|_2$
    $\mathbf{x}_k = \mathbf{x}_0 + \mathbf{B}\left[\mathbf{w}_1, \mathbf{w}_2, \cdots, \mathbf{w}_k\right] \mathbf{y}_k$
    $\mathbf{r}_k = \mathbf{b} - \mathbf{A}\mathbf{x}_k$ 1 MV
    Stopping rule goes here
  end for
  $\mathbf{x}_0 = \mathbf{x}_k$
  $\mathbf{r}_0 = \mathbf{r}_k$
end for

Algorithm 9 BA-GMRES(p)
(3pL + L + 1) MV or (2pL + 2L + 1) MV

Choose initial $\mathbf{x}_0$
$\mathbf{res} = \mathbf{b} - \mathbf{A}\mathbf{x}_0$ 1 MV
for $l = 1, 2, \ldots, L$ **do**
  $\mathbf{r}_0 = \mathbf{B}(\mathbf{res})$ 1 MV
  $\mathbf{w}_1 = \mathbf{r}_0 / \|\mathbf{r}_0\|_2$
  for $k = 1, 2, \ldots, p$ **do**
    $\mathbf{q}_k = \mathbf{B}\mathbf{A}\mathbf{w}_k$ 2 MV
    for $i = 1, 2, \ldots, k$ **do**
      $h_{i,k} = \mathbf{q}_k^T \mathbf{w}_i$
      $\mathbf{q}_k = \mathbf{q}_k - h_{i,k}\mathbf{w}_i$
    end for
    $h_{k+1,k} = \|\mathbf{q}_k\|_2$
    $\mathbf{w}_{k+1} = \mathbf{q}_k / h_{k+1,k}$
    $\mathbf{y}_k = \text{argmin}_{\mathbf{y}} \|\|\mathbf{r_0}\|_2 \mathbf{e}_1 - \mathbf{H}_k\mathbf{y}\|_2$
    $\mathbf{x}_k = \mathbf{x}_0 + \left[\mathbf{w}_1, \mathbf{w}_2, \cdots, \mathbf{w}_k\right] \mathbf{y}_k$
    $\mathbf{r}_k = \mathbf{b} - \mathbf{A}\mathbf{x}_k$ 1 MV
    Stopping rule goes here
  end for
  $\mathbf{x}_0 = \mathbf{x}_k$
  $\mathbf{res} = \mathbf{r}_k$
end for

A.3 Setup for CT

Figure A.1 shows the CT setup with a fan beam geometry. To the left, an X-ray source is illustrated. X-rays are emitted which pass through the object of interest (illustrated with a blue square) and the attenuated X-rays are picked up by the detector to the right. It is different how measurements/lengths must be specified from software package to software package. ASTRA needs |CD| and |SC| while TIGRE needs |SD| and |SC|.

With the setups shown in Table 4.6, the distance from the center of the object to the detector is zero, both in the setups for TIGRE and ASTRA. Physically, this is not possible, but mathematically we simply have a virtual detector. Considering Figure A.2, we see that the X-rays hit the detector in the same spots when $|\text{CD}| > 0$ as for $|\text{CD}| = 0$. The only difference is a scaling factor as the size of the detector element increases as |CD| increases.

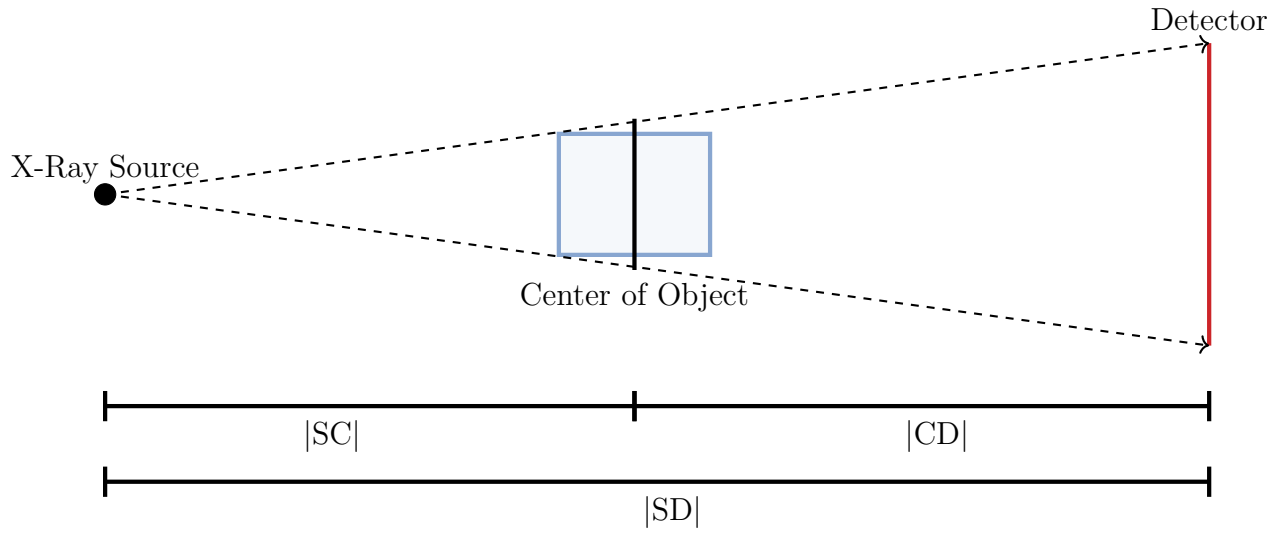


Figure A.1: An illustration of the CT setup. To the left, we see the X-ray source and to the right, we have the detector. In the middle, the object of interest is illustrated by a blue square, and the center of the object is marked with a vertical line. Different distances needed in the software packages are illustrated at the bottom of the figure.

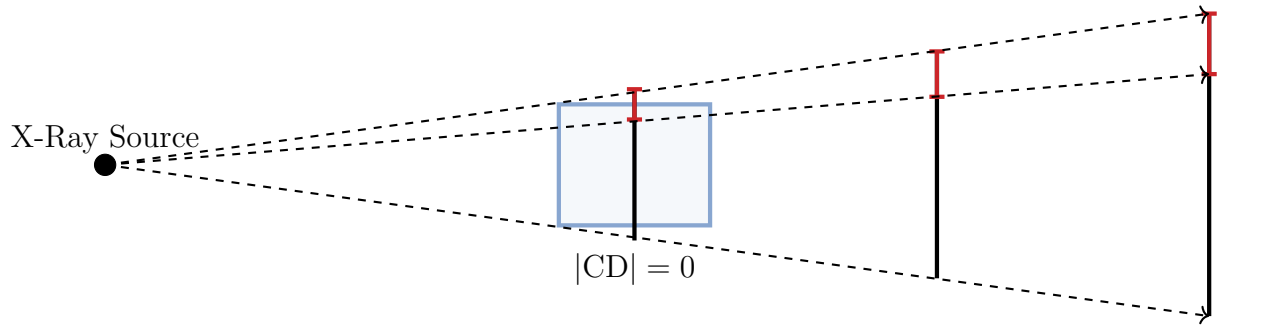


Figure A.2: Here we illustrate the connection between the distance |CD| and the detector size. When $|CD| = 0$, we have a virtual detector (as this is not physically possible) and the detector element has a certain size. Increasing |CD| means an increase in the size of the detector element as illustrated here.

APPENDIX B

Results

B.1 Comparing Public Domains

We here document the results for AB-GMRES(∞). They have not been included in the main report as the behavior looks similar to BA-GMRES(∞). The main difference is that AB-GMRES(∞) reaches the optimum a few iterations before BA-GMRES(∞).

Figure B.1 shows the convergence history for AB-GMRES(∞) when we use the forward projection model line and use parallel beam geometry and Figure B.2 is when the geometry is fan beam instead of parallel beam. We obtain semi-convergence in both cases. The convergence histories look very similar with the same number of iterations to reach the optimum and with approximately the same relative error at the optimum. Regardless of the geometry, the AB-GMRES(∞) solver provides approximately the same result.

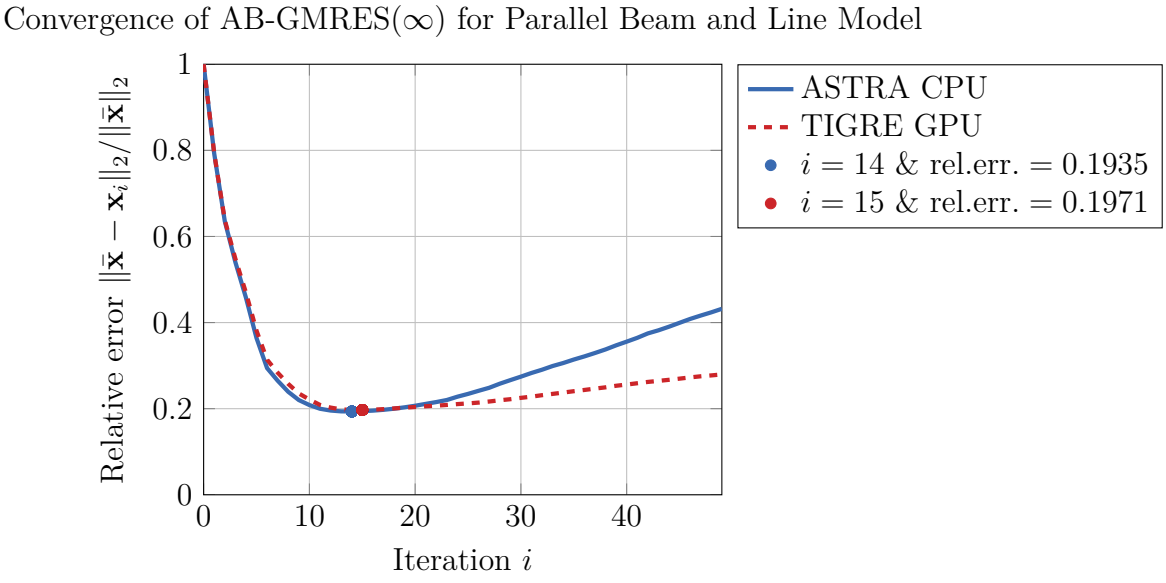


Figure B.1: The convergence history of AB-GMRES(∞) for parallel beam geometry and with the forward projector being the line model. ASTRA CPU creates a matched projector pair while TIGRE creates an unmatched projector pair.

Figure B.3 shows the convergence history of AB-GMRES(∞) when using Joseph's model as the forward projector and with parallel beam geometry. It is clear that something is

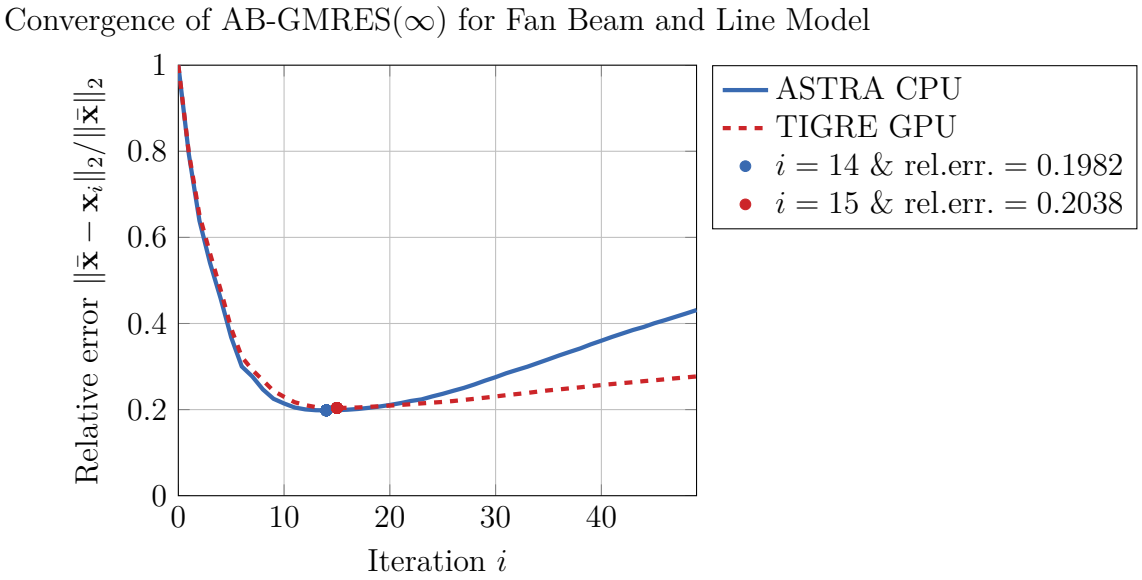


Figure B.2: The convergence history of AB-GMRES(∞) for fan beam geometry and with the forward projector being the line model. ASTRA CPU creates a matched projector pair while TIGRE creates an unmatched projector pair.

wrong with TIGRE's convergence history and the same behavior has been seen for the fan beam geometry (which is not included here).

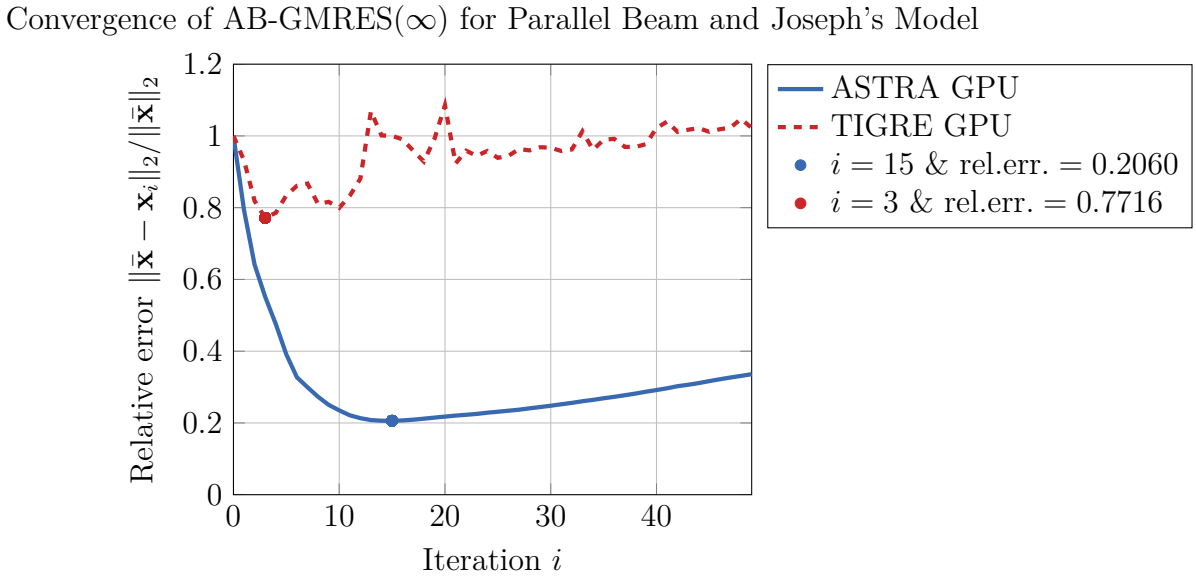


Figure B.3: The convergence history of AB-GMRES(∞) for parallel beam geometry and with the forward projector being Joseph's model. ASTRA GPU creates an unmatched projector pair and so does TIGRE.

In conclusion, AB-GMRES(∞) works well for ASTRA with the forward projector being

the line model (matched projector pair) and for Joseph's model (unmatched projector pair). The relative error at the optimum is approximately 20%. For TIGRE, we see semi-convergence with the line model but not with Joseph's model.

B.2 User Domain Projectors

The results for AB-GMRES(∞) when using the user domain software to obtain the projector pairs are shown here. Figure B.4 shows the convergence history when the forward projector is the line model and Figure B.5 is when we use the strip model with 10 rays per strip. Both obtain semi-convergence and the relative error at the optimum is approximately 19% and 24%, respectively. Adding more rays per strip does not decrease the relative error at the optimum (we tried 50 rays per strip). Thus, AB-GMRES(∞) using the strip model as the forward projector does not give as good results as the line model.

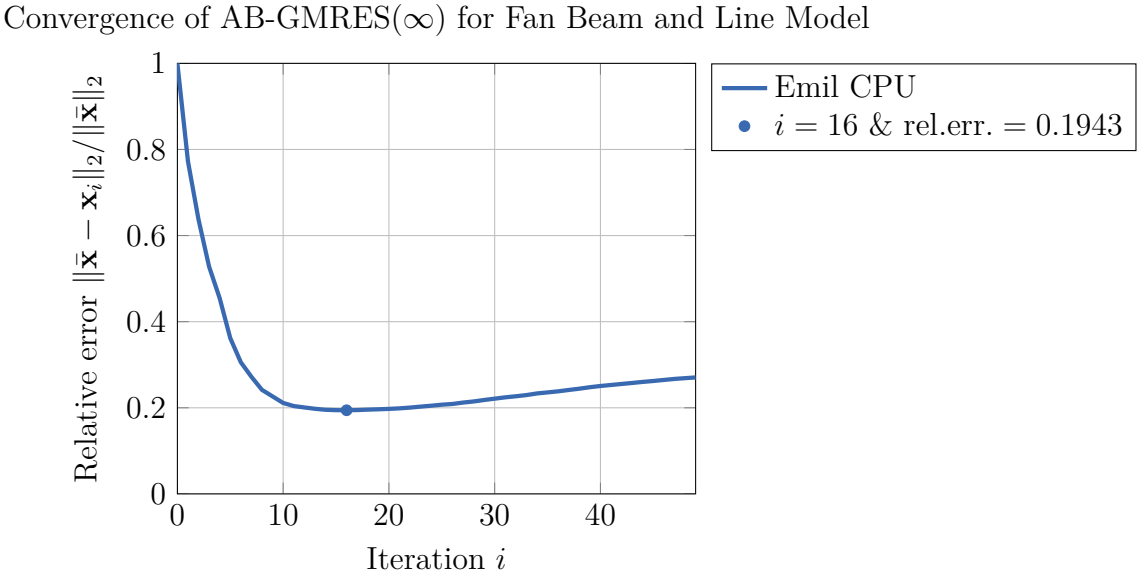


Figure B.4: The convergence history of AB-GMRES(∞) for fan beam geometry with the user domain's line model.

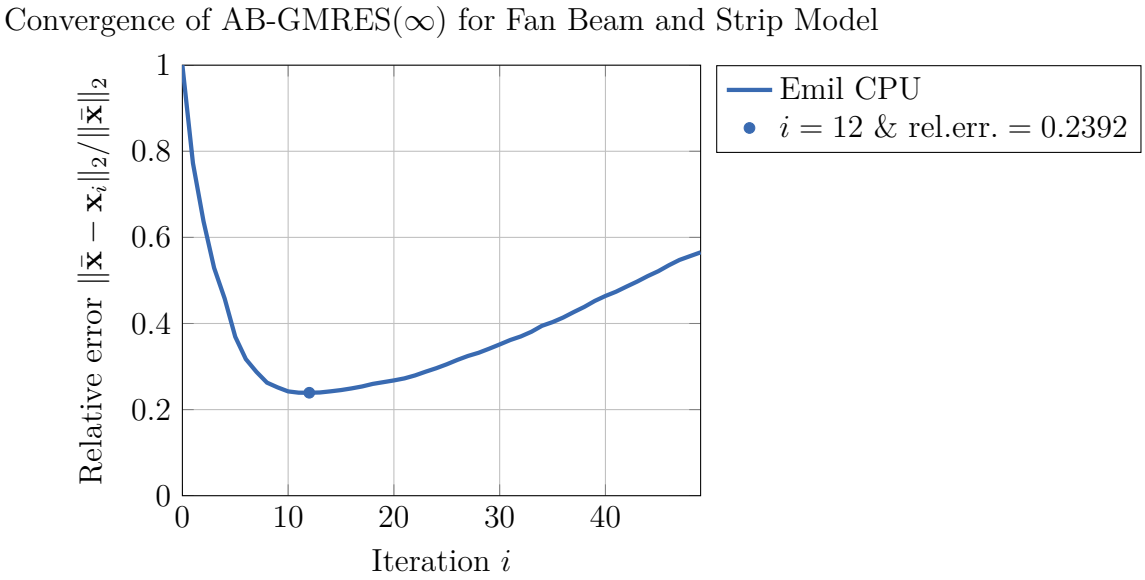


Figure B.5: The convergence history of AB-GMRES(∞) for fan beam geometry with the user domain's strip model with 10 rays per strip.

B.3 Restart

The main report includes the results for BA-GMRES when having fan beam geometry. For the interested reader, this section includes the AB-GMRES results and the results for AB/BA-GMRES when considering parallel beam geometry in ASTRA. The results are overall very similar to what is shown in the main report for BA-GMRES.

B.3.1 ASTRA: Fan Beam Geometry

Here we show the convergence history for AB-GMRES. We both consider the relative error as a function of iterations (Figure B.6), of FLOPs (Figure B.7), and of wall-clock time (Figure B.8).

The results from Figure B.6 illustrate that AB-GMRES(∞) is the first to reach the optimum at iteration 15 followed by AB-GMRES(10) at iteration 17. Having $p = 5$ and $p = 3$, we need to take more iterations before reaching the optimum. Restarted GMRES obtain slightly better solutions as the relative error at the optimum is smaller (however, it is a small difference). We see a relation between the restart parameter p and the convergence rate. When p is small so is the convergence rate and vice versa.

Figure B.7 shows the number of FLOPs when running 300 iterations. The dots illustrate the optimum. From the results, it seems that AB-GMRES(∞) is the cheapest. However, restarted GMRES also takes many iterations to obtain a slightly better result which will result in more FLOPs. If we compute all 300 iterations, we see that restarted GMRES

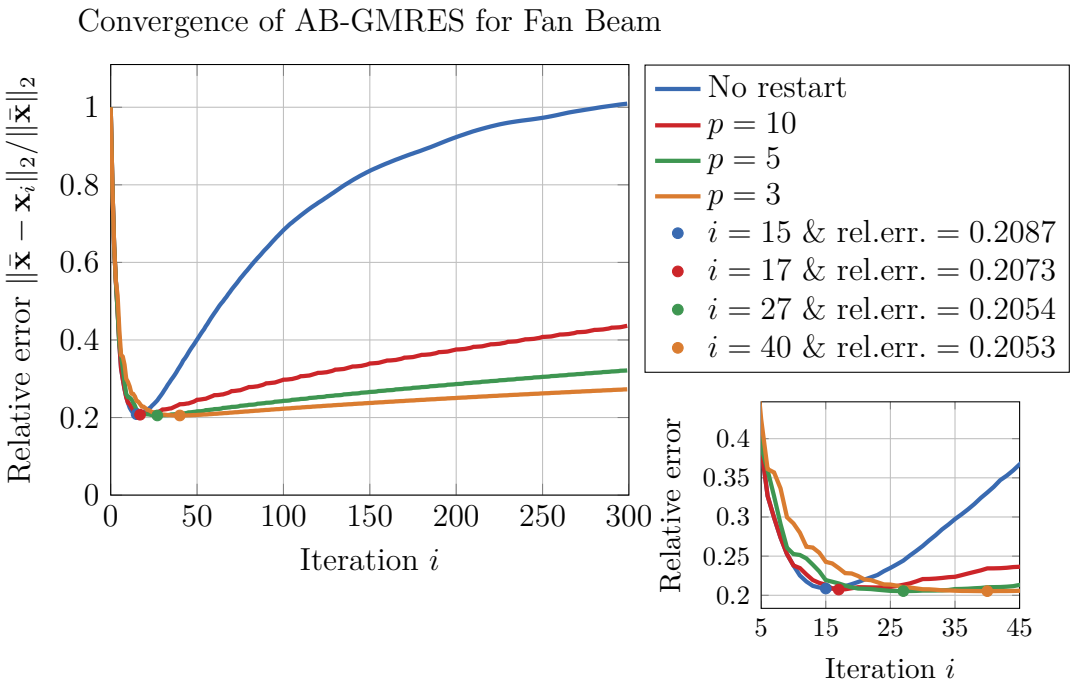


Figure B.6: The convergence history of AB-GMRES(∞) and AB-GMRES(p) with $p = 3, 5, 10$ when having fan beam geometry.

compute fewer FLOPs compared to AB-GMRES(∞).

Finally, we also consider the relative error as a function of the wall-clock time, see Figure B.8. We again look at 300 iterations, and if we compute all 300 iterations we see that restarted GMRES is time-saving. If we, however, can stop at the optimum (iteration 15 for AB-GMRES(∞)) then we see that AB-GMRES(10) is the fastest followed by AB-GMRES(∞) and AB-GMRES(5). We see that $p = 3$ is the slowest to reach the optimum which is because it first reaches the optimum at iteration 40.

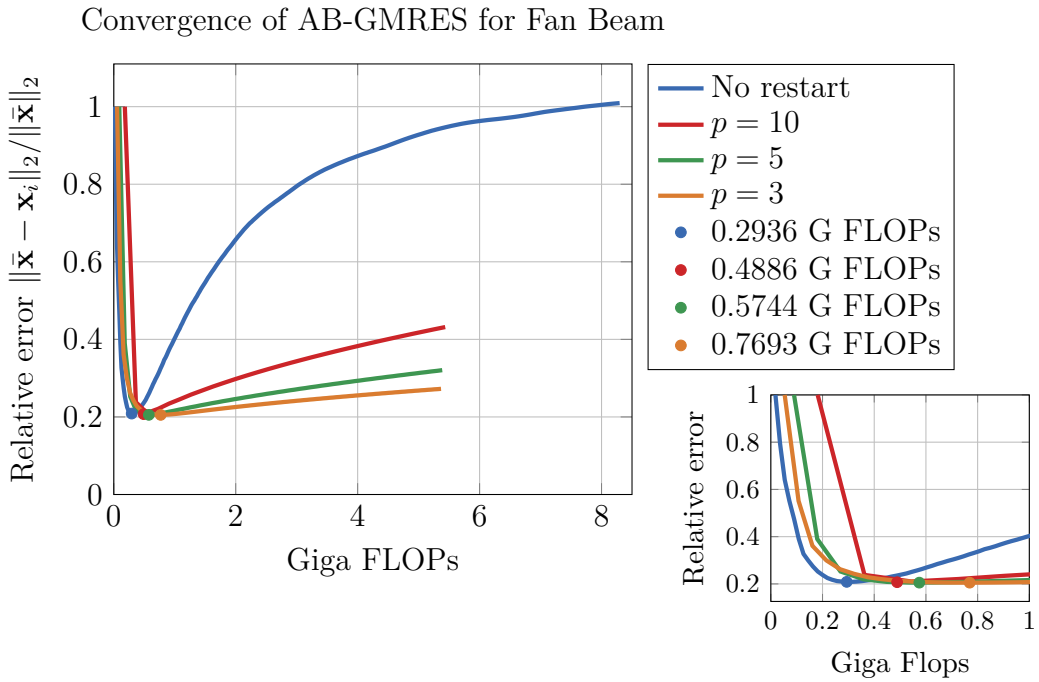


Figure B.7: The relative error as a function of giga FLOPs for 300 iterations of AB-GMRES(∞) and AB-GMRES(p) with $p = 3, 5, 10$.

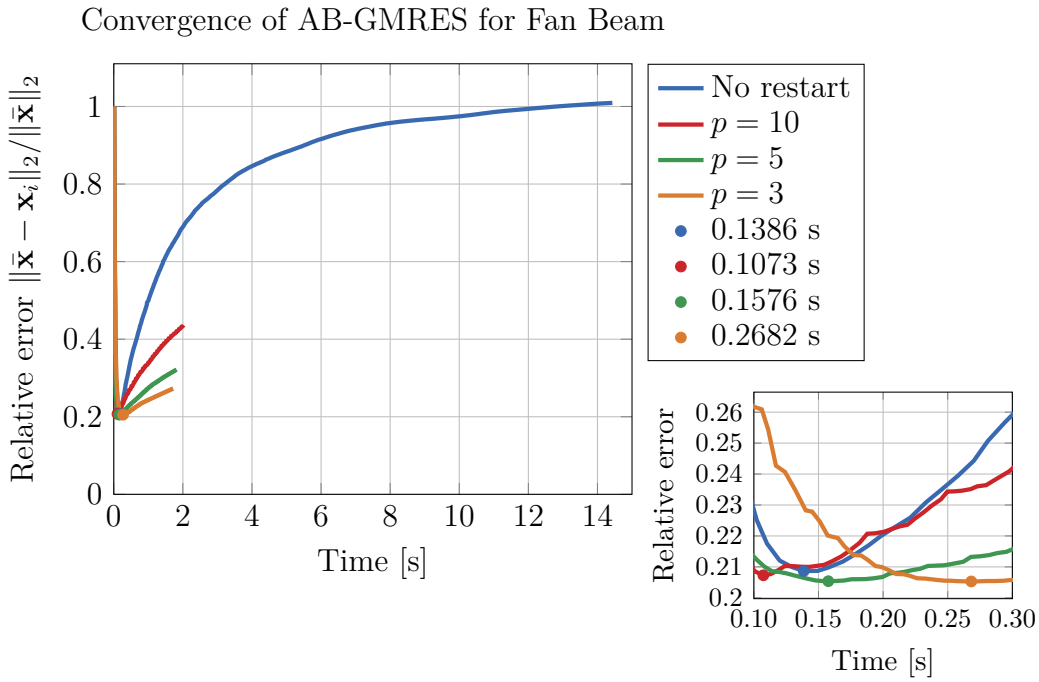


Figure B.8: The relative error as a function of the wall-clock time for 300 iterations of AB-GMRES(∞) and AB-GMRES(p) with $p = 3, 5, 10$.

B.3.2 ASTRA: Parallel Beam Geometry

Here we show the results for both AB- and BA-GMRES when we have parallel beam geometry. Overall, we see the same results as for the fan beam geometry.

Figure B.9 shows the convergence of BA-GMRES and Figure B.10 shows the convergence of AB-GMRES. In both cases, not using restart results in reaching the optimum the fastest but also leads to high relative errors right after. Using restarted GMRES, yields a decreased convergence rate, and the optimal number of iterations for AB/BA-GMRES(10) is very close to AB/BA-GMRES(∞) (only two iterations separate them). So the advantage of using restart, besides the memory complexity, is that the relative error of AB/BA-GMRES(10) after reaching the optimum does not explode as it does for AB/BA-GMRES(∞) which makes it more stable in regards to stopping criteria.

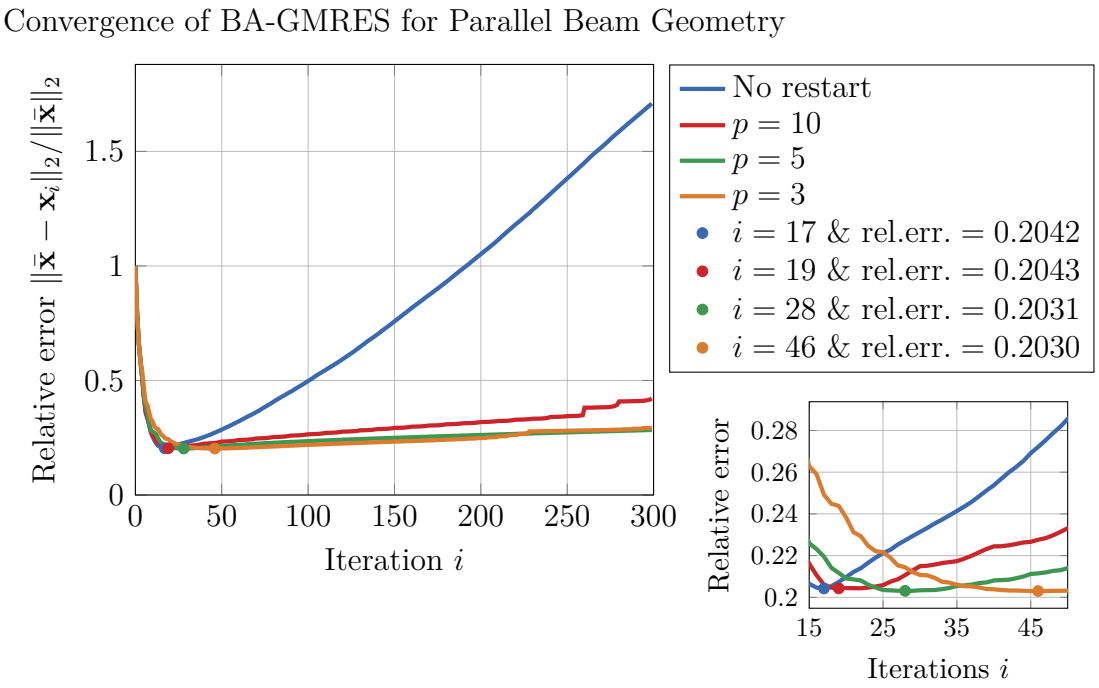


Figure B.9: The convergence history of BA-GMRES(∞) and BA-GMRES(p) with $p = 3, 5, 10$ when having parallel beam geometry.

The relative error as a function of the wall-clock time is shown in Figure B.11 for BA-GMRES and in Figure B.12 for AB-GMRES. Restarted GMRES with $p = 10$ reaches the optimum the fastest when considering the time instead of iterations. It is followed by $p = 5$ and not using restart. It is clear from the results, that it is much faster to compute 300 iterations with restarted GMRES compared to not using restart.

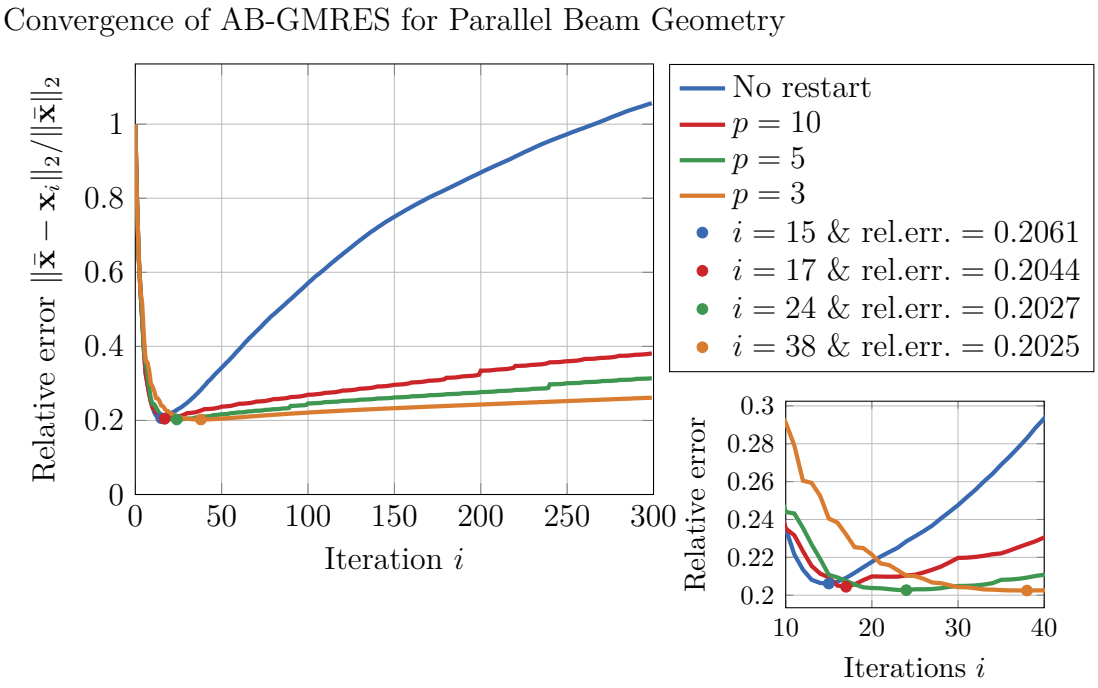


Figure B.10: The convergence history of AB-GMRES(∞) and AB-GMRES(p) with $p = 3, 5, 10$ when having parallel beam geometry.

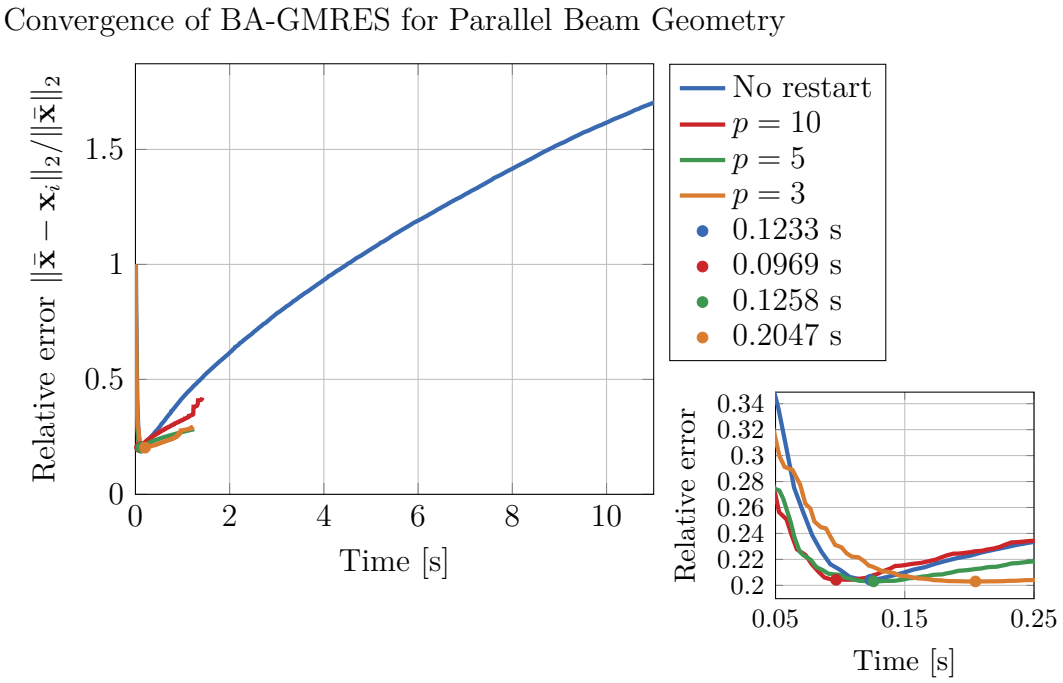


Figure B.11: The relative error as a function of the wall-clock time for 300 iterations of BA-GMRES(∞) and BA-GMRES(p) with $p = 3, 5, 10$.

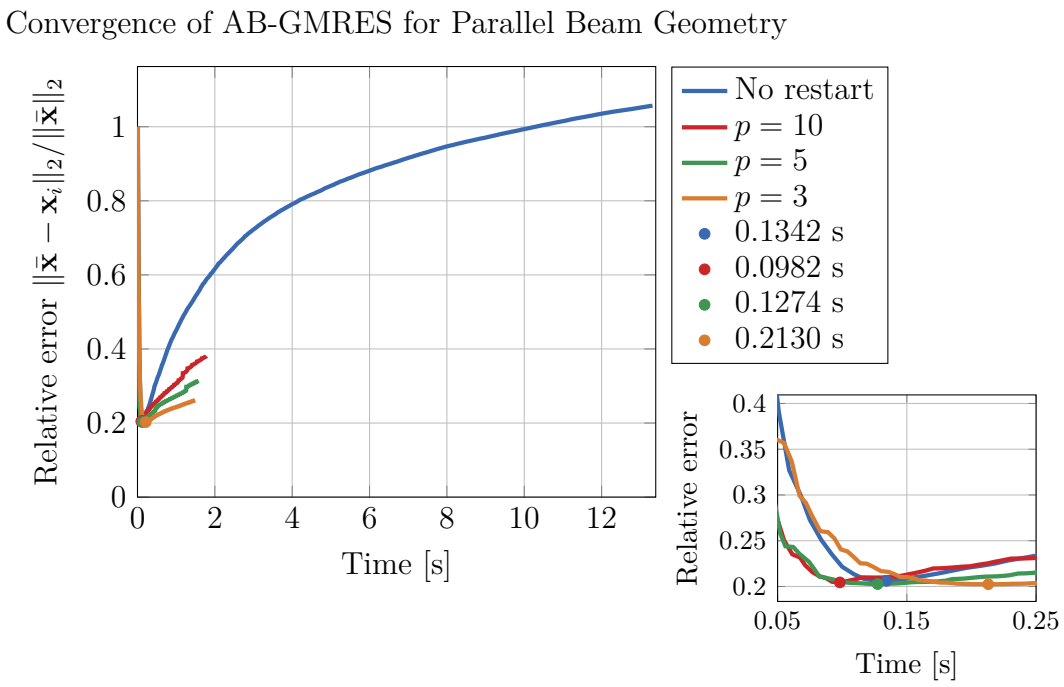


Figure B.12: The relative error as a function of the wall-clock time for 300 iterations of AB-GMRES(∞) and AB-GMRES(p) with $p = 3, 5, 10$.

B.3.3 User Domain: Line Model

This section includes the results for AB-GMRES when using the user domain to construct the unmatched projector pair. Here we consider the line model as the forward projector.

Figure B.13 shows the convergence curve for AB-GMRES(∞) and AB-GMRES(p) for $p = 3, 5, 10$ when taking 300 iterations. In previous results, the relative error for AB-GMRES(∞) has increased significantly after the optimum corresponding to relative errors of one or higher. Here, AB-GMRES(∞) increases after reaching the optimum but stabilizes around a relative error of 0.3. The solutions at iteration 300 are approximately the same for all solvers which have not been the case when using the projector pair from ASTRA.

Figure B.14 shows the relative error as a function of wall-clock time. The user domain is CPU based and, therefore, it takes longer to reach the optimum. AB-GMRES(∞) and AB-GMRES(10) are the first to reach the optimal number of iterations followed by AB-GMRES(5) and AB-GMRES(3), respectively.

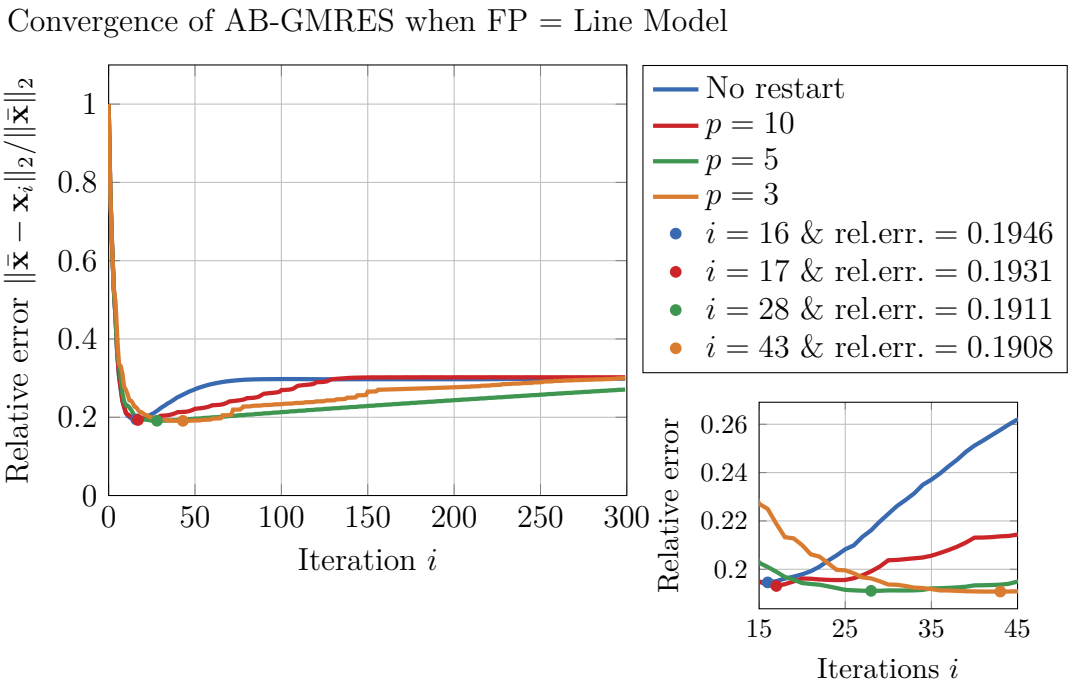


Figure B.13: The convergence history of AB-GMRES(∞) and AB-GMRES(p) with $p = 3, 5, 10$ when having fan beam geometry. The forward projector is the line model.

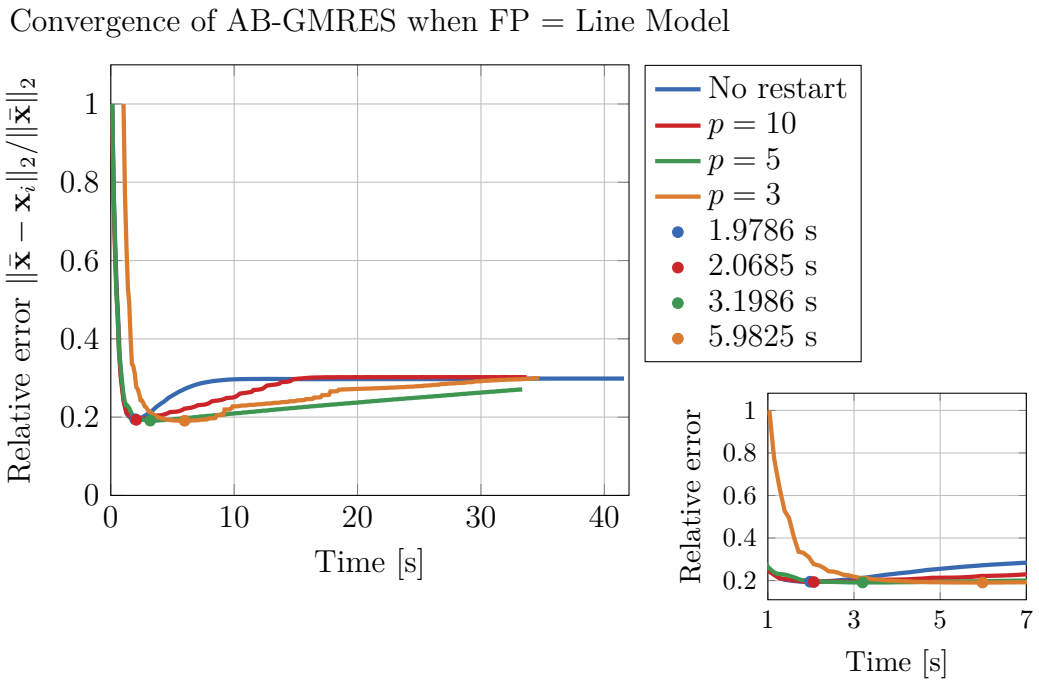


Figure B.14: The relative error as a function of the wall-clock time for 300 iterations of AB-GMRES(∞) and AB-GMRES(p) with $p = 3, 5, 10$ and when using the line model as the forward projector.

B.3.4 User Domain: Strip Model

Here we consider the results from AB-GMRES when using the user domain to construct the forward projector with the strip model with 10 rays per strip and the unmatched back projector. Figure B.15 shows the convergence history and Figure B.16 shows the relative error as a function of the wall-clock time. The appearance of the results is similar to when considering the line model as the forward projector, but the relative error is higher and the results at iteration 300 are not as similar.

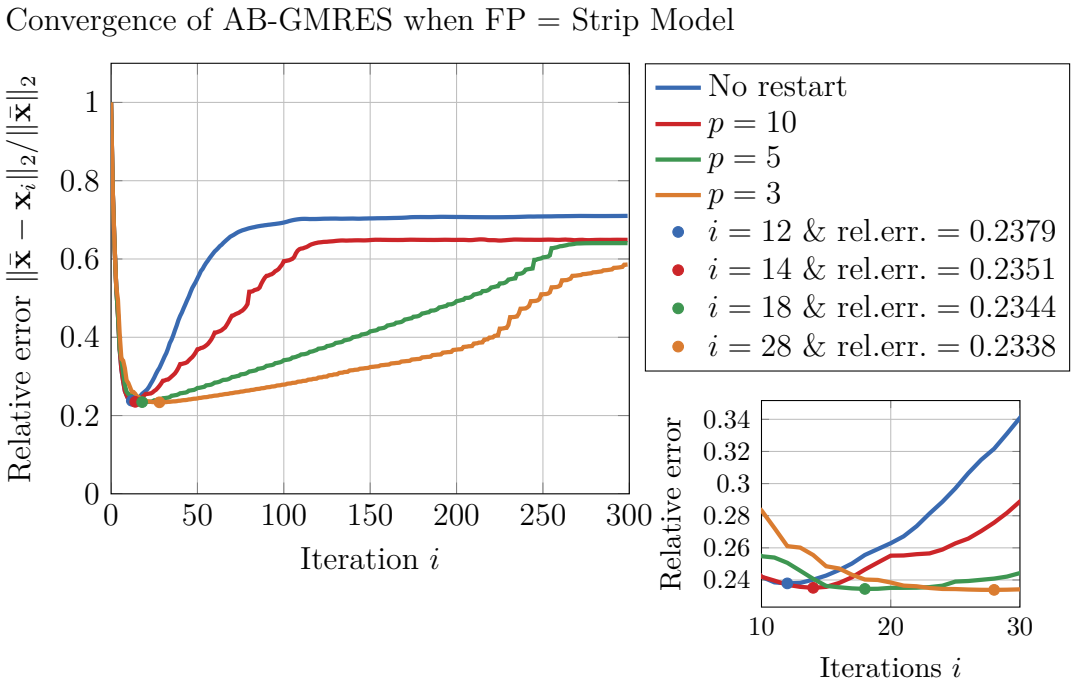


Figure B.15: The convergence history of AB-GMRES(∞) and AB-GMRES(p) with $p = 3, 5, 10$ when having fan beam geometry. The forward projector is the strip model with 10 rays per strip.

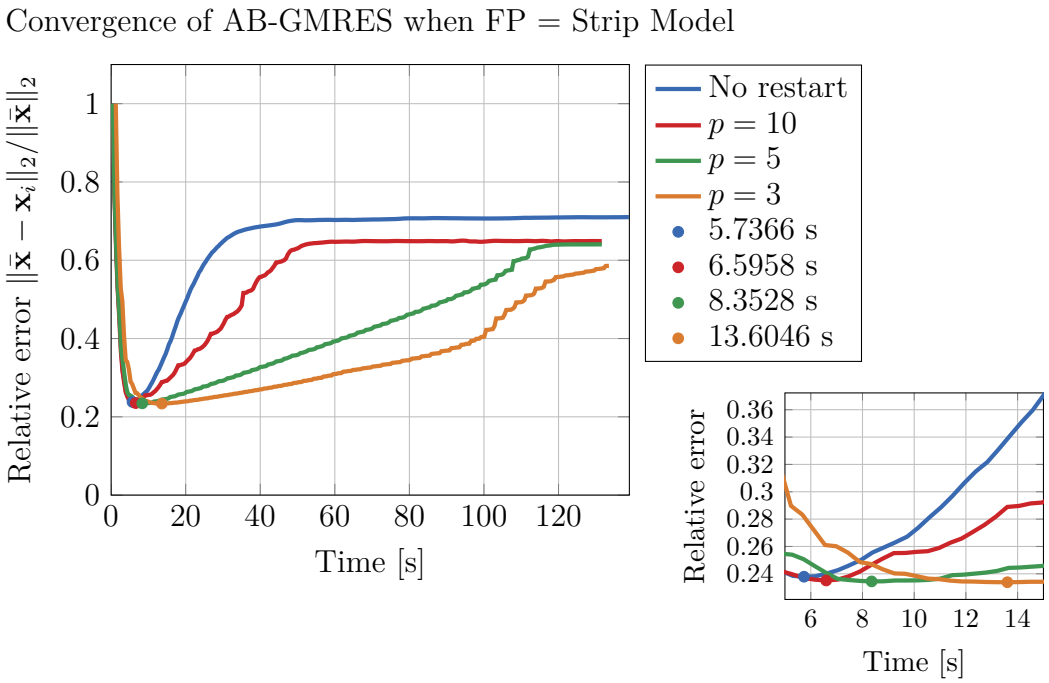


Figure B.16: The relative error as a function of the wall-clock time for 300 iterations of AB-GMRES(∞) and AB-GMRES(p) with $p = 3, 5, 10$ and when using the strip model with 10 rays per strip as the forward projector.

B.4 DP

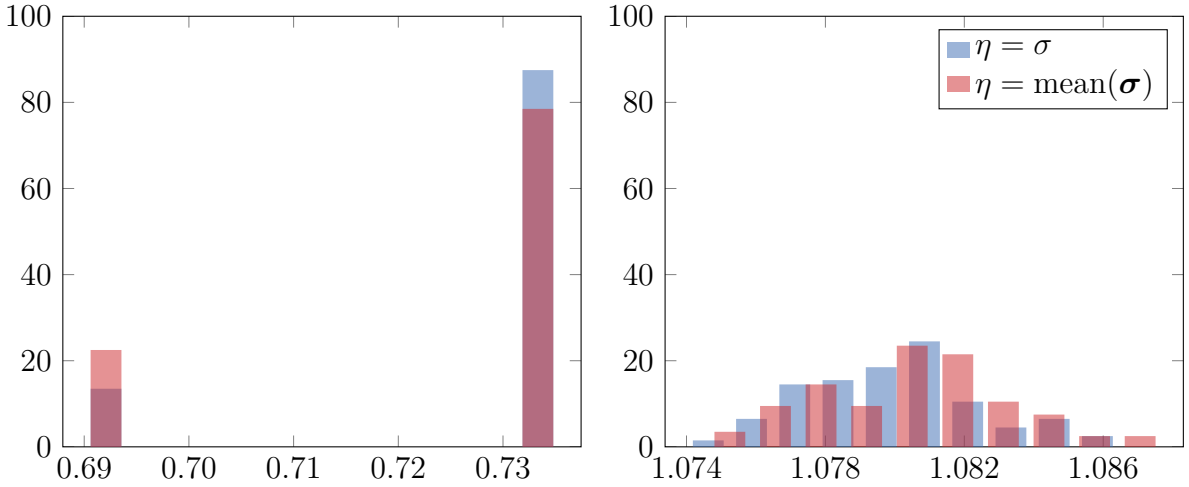
This section shows the results for AB-GMRES when considering DP as the stopping criterion. We consider the iteration ratios from (5.1) and the error ratios from (5.2) for AB-GMRES(∞) and AB-GMRES(5) with the true noise level $\eta = \sigma$ and with the approximation $\eta = \text{mean}(\boldsymbol{\sigma})$. In the main report, we have disregarded the results for parallel beam geometry as the results are similar to other results but these results have been included here.

B.4.1 Public Domain: ASTRA

ASTRA provides an unmatched projector pair both for parallel and fan beam geometries. The main report includes the fan beam results for BA-GMRES and here we show AB-GMRES for the fan beam geometry and both AB/BA-GMRES for the parallel beam geometry.

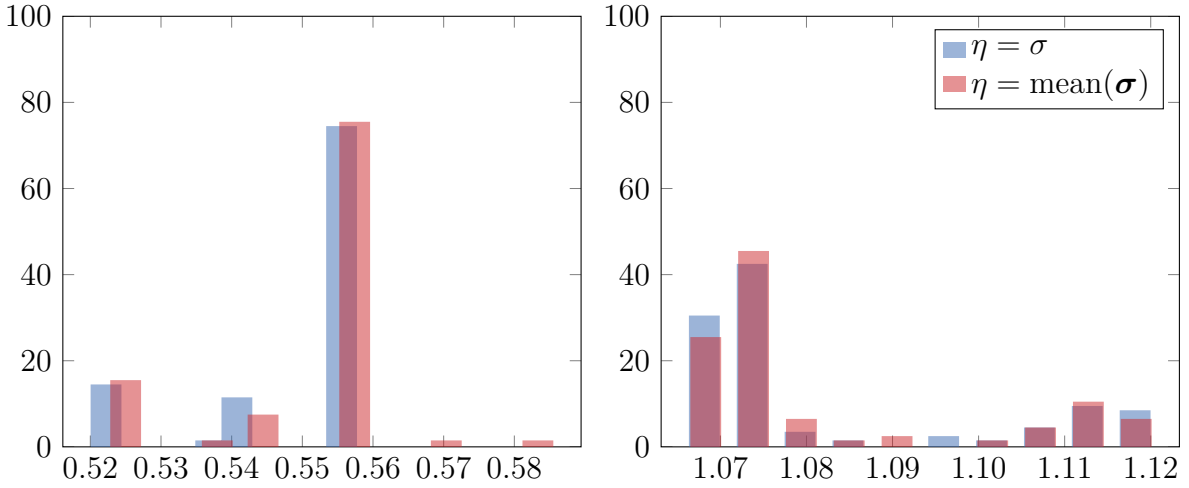
Fan beam:
Figure B.17 shows the iteration and error ratios for AB-GMRES(∞) and Figure B.18 shows the ratios for AB-GMRES(5). DP stops at the two same iterations for 100 realizations when considering AB-GMRES(∞) while i_{dp} obtain different values for AB-GMRES(5). This leads to the error ratios being between 7-12% for AB-GMRES(5) and

7-9% for AB-GMRES(∞).



(a) The ratios $R_{i,\text{dp}}$ for AB-GMRES(∞). **(b)** The ratios $R_{e,\text{dp}}$ for AB-GMRES(∞).

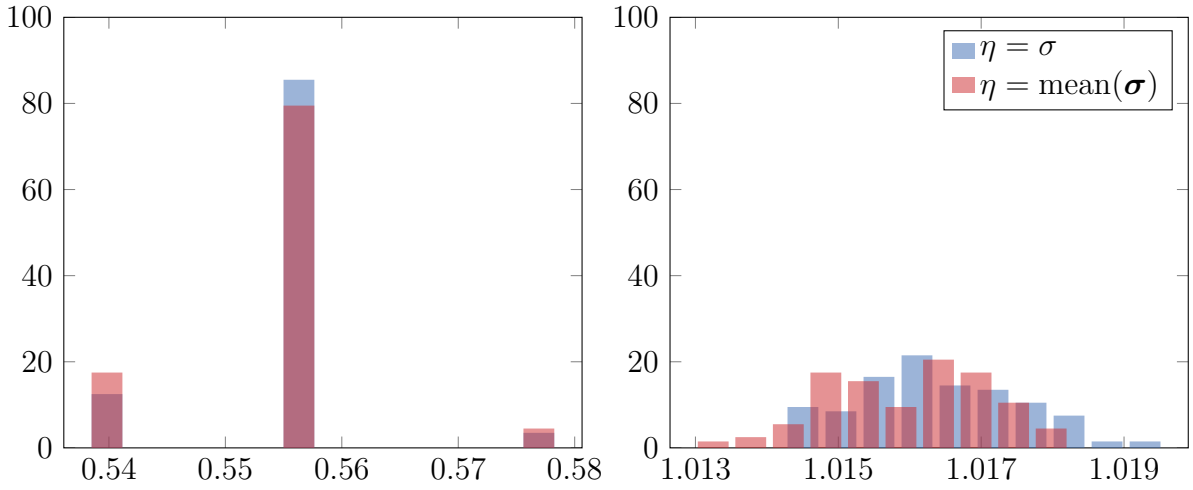
Figure B.17: A histogram showing the ratios (5.1) to the left and (5.2) to the right when using AB-GMRES(∞) to solve the unmatched CT problem obtained from ASTRA GPU with fan beam geometry.



(a) The ratios $R_{i,\text{dp}}$ for AB-GMRES(5). **(b)** The ratios $R_{e,\text{dp}}$ for AB-GMRES(5).

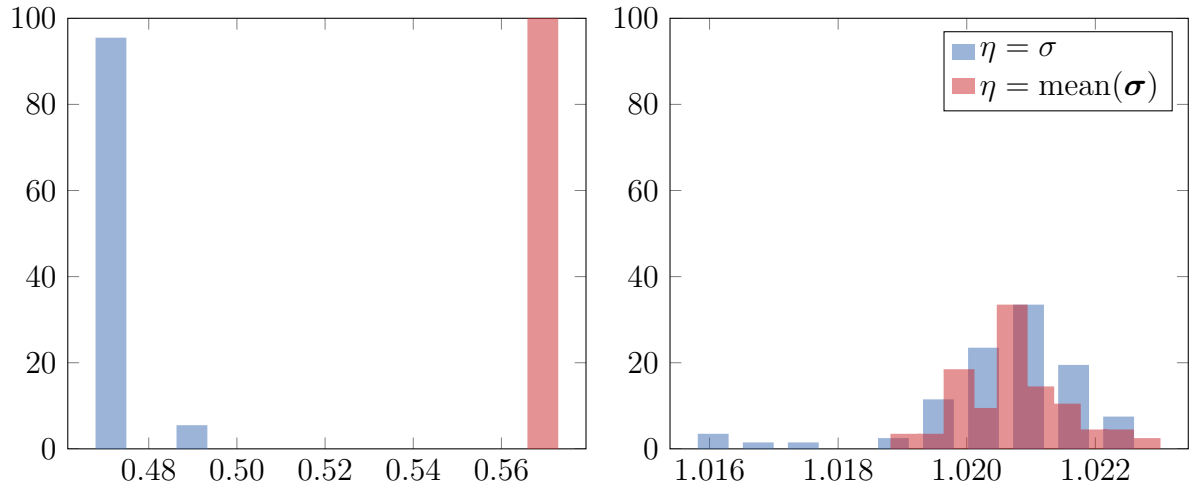
Figure B.18: A histogram showing the ratios (5.1) to the left and (5.2) to the right when using AB-GMRES(5) to solve the unmatched CT problem obtained from ASTRA GPU with fan beam geometry.

Parallel beam:
Figure B.19 shows the ratios results for BA-GMRES(∞) and Figure B.20 shows the results for BA-GMRES(5). Both results show that DP underestimates but has consistency as almost all iteration ratios lie between one or two values. The error ratios are very close to one, meaning the solution $\mathbf{x}_{\text{dp}}$ is close to $\mathbf{x}_{\text{opt}}$ though DP underestimates.



(a) The ratios $R_{i,\text{dp}}$ for BA-GMRES(∞). **(b)** The ratios $R_{e,\text{dp}}$ for BA-GMRES(∞).

Figure B.19: A histogram showing the ratios (5.1) to the left and (5.2) to the right when using BA-GMRES(∞) to solve the unmatched CT problem obtained from ASTRA GPU with parallel beam geometry.

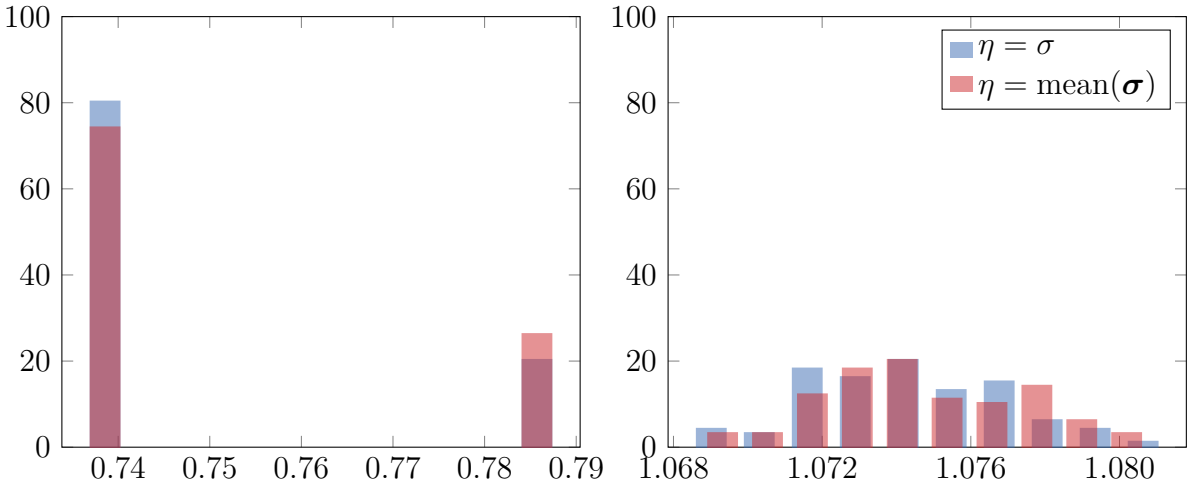


(a) The ratios $R_{i,\text{dp}}$ for BA-GMRES(5). **(b)** The ratios $R_{e,\text{dp}}$ for BA-GMRES(5).

Figure B.20: A histogram showing the ratios (5.1) to the left and (5.2) to the right when using BA-GMRES(5) to solve the unmatched CT problem obtained from ASTRA GPU with parallel beam geometry.

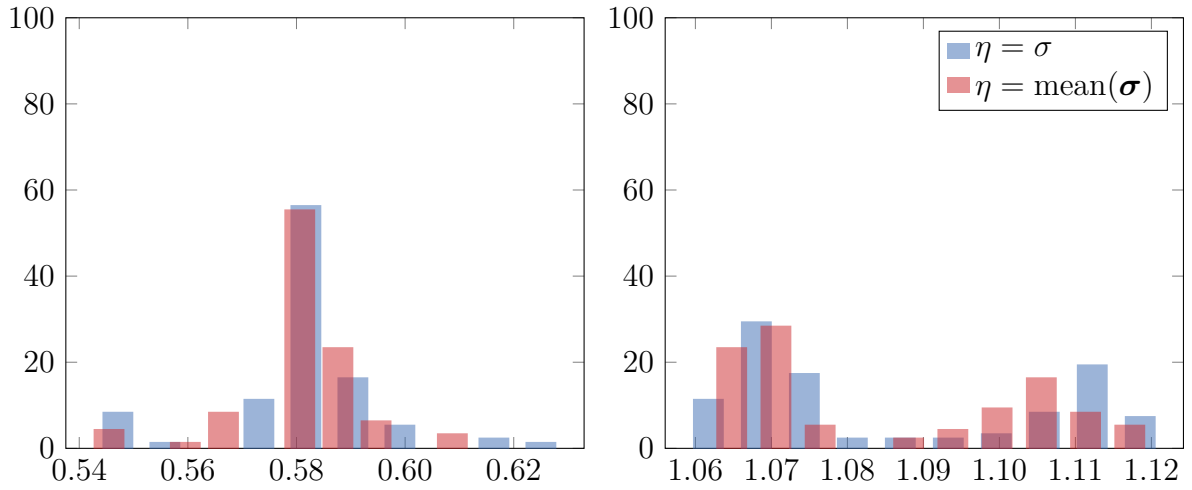
The AB-GMRES results are shown in Figure B.21 and B.22. DP is consistent for AB-GMRES(∞) with error ratios between 6.8-8% and not consistent for AB-GMRES(5)

with error ratios between 6-12%.



(a) The ratios $R_{i,\text{dp}}$ for AB-GMRES(∞). **(b)** The ratios $R_{e,\text{dp}}$ for AB-GMRES(∞).

Figure B.21: A histogram showing the ratios (5.1) to the left and (5.2) to the right when using AB-GMRES(∞) to solve the unmatched CT problem obtained from ASTRA GPU with parallel beam geometry.



(a) The ratios $R_{i,\text{dp}}$ for AB-GMRES(5). **(b)** The ratios $R_{e,\text{dp}}$ for AB-GMRES(5).

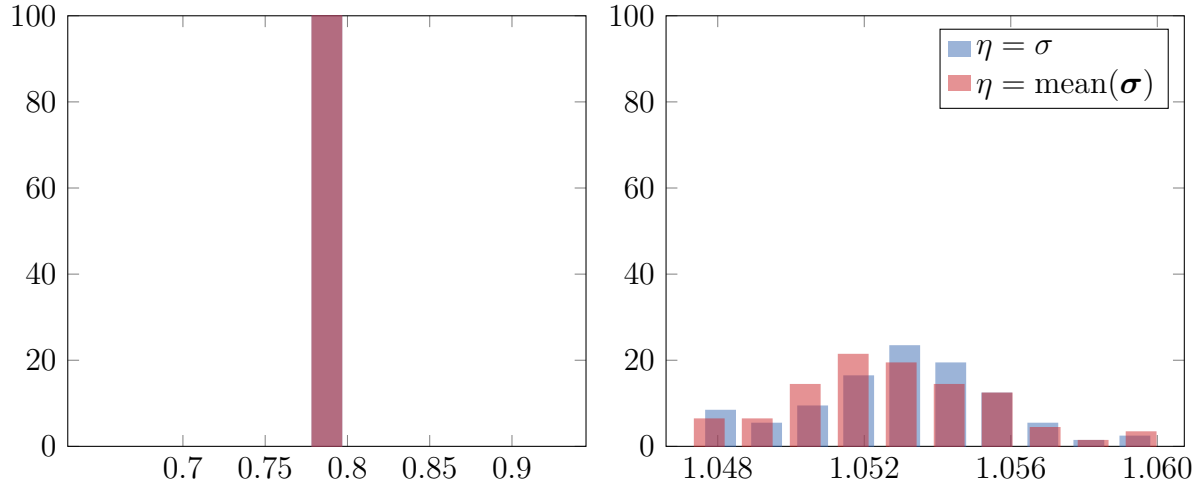
Figure B.22: A histogram showing the ratios (5.1) to the left and (5.2) to the right when using AB-GMRES(5) to solve the unmatched CT problem obtained from ASTRA GPU with parallel beam geometry.

B.4.2 User Domain

Here we show the results for AB-GMRES when using the user domain.

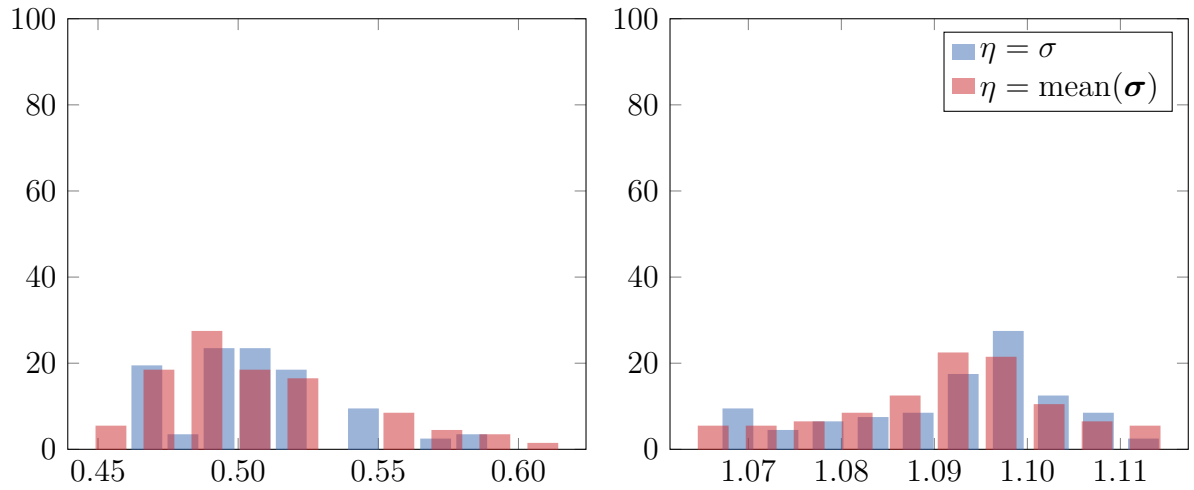
The Line Model:
When using the line model as the forward projector, the ratios when using AB-GMRES(∞)

looks like in Figure B.23 and for AB-GMRES(5) it looks as in Figure B.24. We see that DP is consistent for AB-GMRES(∞) but not for AB-GMRES(5).



(a) The ratios $R_{e,\text{dp}}$ for AB-GMRES(∞). **(b)** The ratios $R_{e,\text{dp}}$ for AB-GMRES(∞).

Figure B.23: A histogram showing the ratios (5.1) to the left and (5.2) to the right when using AB-GMRES(∞) to solve the unmatched CT problem obtained from the user domain with the forward projector being the line model and for fan beam geometry.

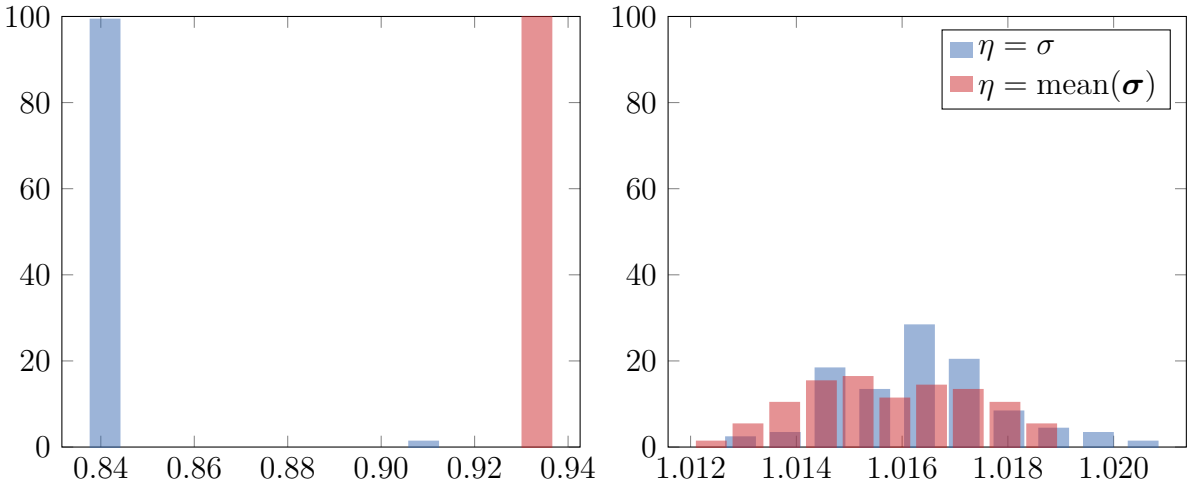


(a) The ratios $R_{e,\text{dp}}$ for AB-GMRES(5). **(b)** The ratios $R_{e,\text{dp}}$ for AB-GMRES(5).

Figure B.24: A histogram showing the ratios (5.1) to the left and (5.2) to the right when using AB-GMRES(5) to solve the unmatched CT problem obtained from the user domain with the forward projector being the line model and for fan beam geometry.

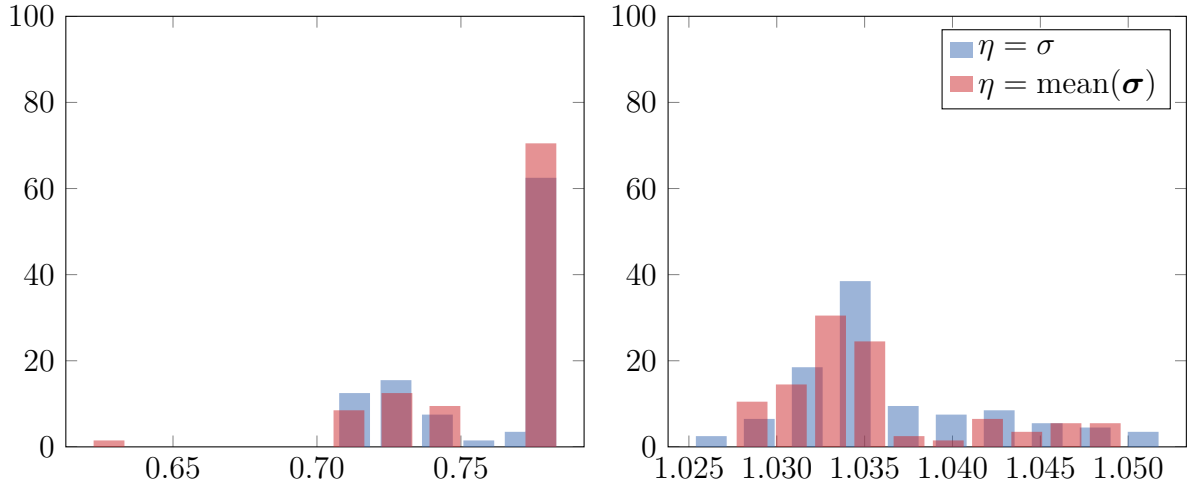
The Strip Model:

When using the strip model with 10 rays per strip as the forward projector, the ratios when using AB-GMRES(∞) looks like in Figure B.25 and for AB-GMRES(5) it looks like in Figure B.26. We, also here, see that DP is consistent for AB-GMRES(∞) but not for AB-GMRES(5).



(a) The ratios $R_{i,\text{dp}}$ for AB-GMRES(∞). **(b)** The ratios $R_{e,\text{dp}}$ for AB-GMRES(∞).

Figure B.25: A histogram showing the ratios (5.1) to the left and (5.2) to the right when using AB-GMRES(∞) to solve the unmatched CT problem obtained from the user domain with the forward projector being the strip model (10 rays per strip) and for fan beam geometry.



(a) The ratios $R_{i,\text{dp}}$ for AB-GMRES(5). **(b)** The ratios $R_{e,\text{dp}}$ for AB-GMRES(5).

Figure B.26: A histogram showing the ratios (5.1) to the left and (5.2) to the right when using AB-GMRES(5) to solve the unmatched CT problem obtained from the user domain with the forward projector being the strip model (10 rays per strip) and for fan beam geometry.

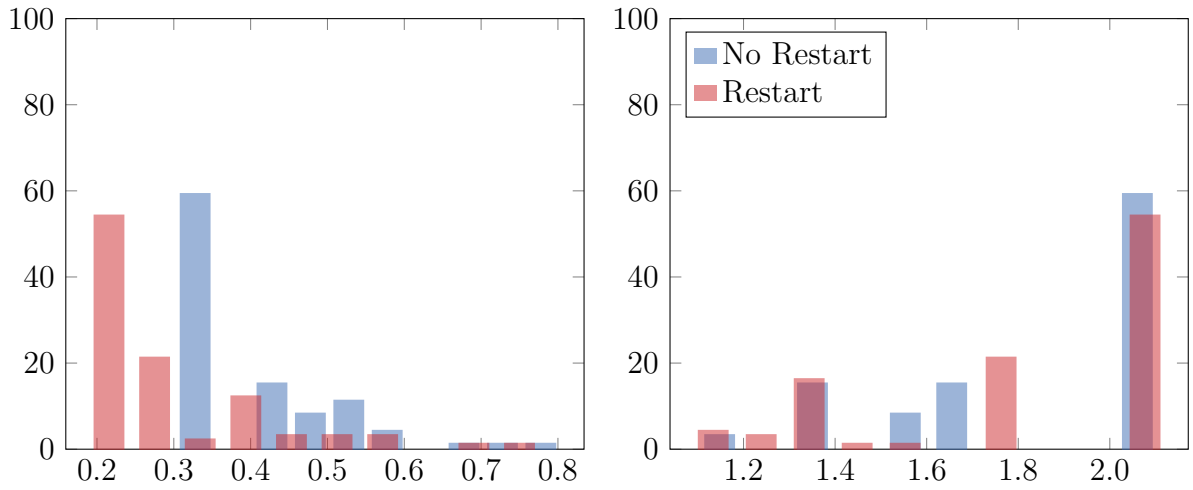
B.5 NCP

This section shows the results for AB-GMRES when considering NCP as the stopping criterion. We consider the iteration ratios from (5.3) and the error ratios from (5.4) for AB-GMRES(∞) and AB-GMRES(5). In the main report, we have disregarded the results for parallel beam geometry as the results are similar to other results but these results have been included here.

B.5.1 Public Domain: ASTRA

Here we both consider AB-GMRES for fan beam geometry and AB/BA-GMRES for parallel beam geometry.

Fan beam:
Figure B.27 shows the iteration ratios to the left and the error ratios to the right for both AB-GMRES(∞) and AB-GMRES(5). We see that NCP underestimates the optimal iteration and is not consistent. The error ratios are very large.

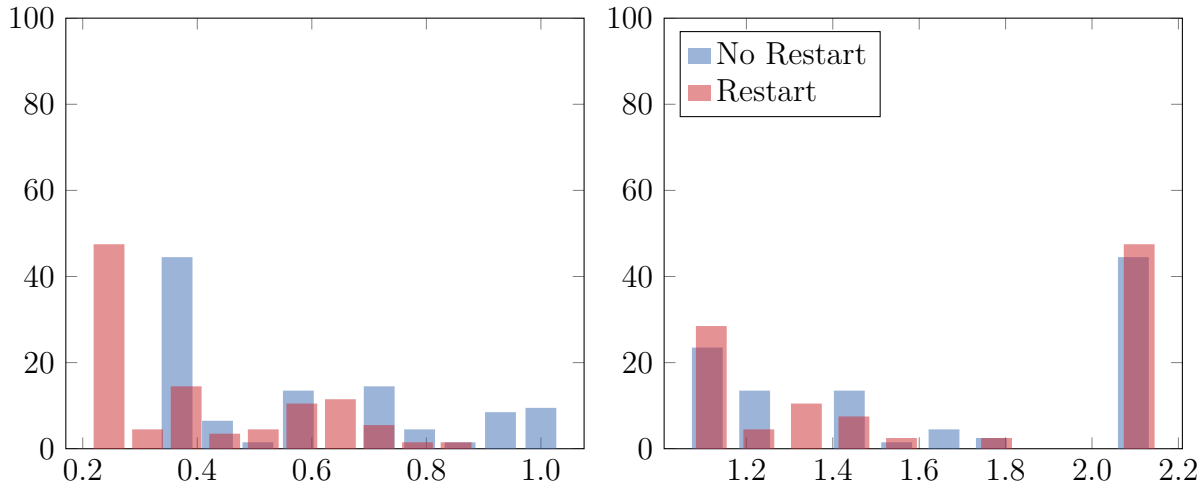


(a) The ratios $R_{i,\text{ncp}}$ for AB-GMRES.

(b) The ratios $R_{e,\text{ncp}}$ for AB-GMRES.

Figure B.27: A histogram showing the ratios (5.3) to the left and (5.4) to the right when using AB-GMRES to solve the unmatched CT problem obtained from ASTRA GPU for fan beam geometry. We both consider AB-GMRES(∞) (no restart) and AB-GMRES(5) (restart).

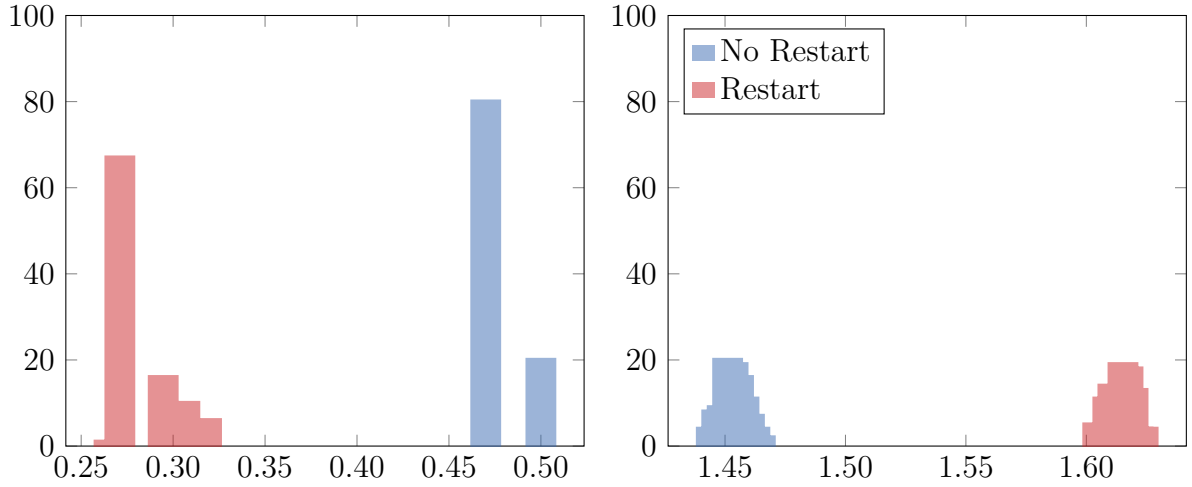
Parallel beam:
Figure B.28 shows the results for BA-GMRES and Figure B.29 shows the results for AB-GMRES. We see that NCP does not obtain consistent results for either BA-GMRES(∞) or BA-GMRES(5). On a few occasions for BA-GMRES(∞) NCP obtains approximately the optimal iteration but all other times it underestimates leading to some high error ratios.



(a) The ratios $R_{i,\text{ncp}}$ for BA-GMRES. **(b)** The ratios $R_{e,\text{ncp}}$ for BA-GMRES.

Figure B.28: A histogram showing the ratios (5.3) to the left and (5.4) to the right when using BA-GMRES to solve the unmatched CT problem obtained from ASTRA GPU for parallel beam geometry. We both consider BA-GMRES(∞) (no restart) and BA-GMRES(5) (restart).

The results for AB-GMRES, shown in Figure B.29, are more stable. NCP underestimates, however, the ratios are approximately the same and, thereby, it looks consistent. The error ratios are still high, around 45-60%.

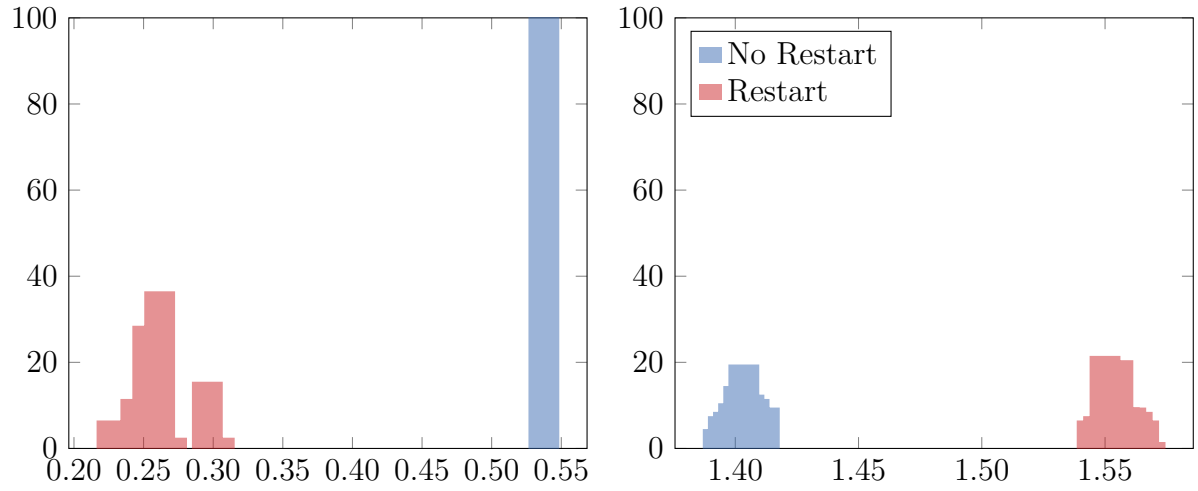


(a) The ratios $R_{i,\text{ncp}}$ for AB-GMRES. **(b)** The ratios $R_{e,\text{ncp}}$ for AB-GMRES.

Figure B.29: A histogram showing the ratios (5.3) to the left and (5.4) to the right when using AB-GMRES to solve the unmatched CT problem obtained from ASTRA GPU for parallel beam geometry. We both consider AB-GMRES(∞) (no restart) and AB-GMRES(5) (restart).

B.5.2 User Domain

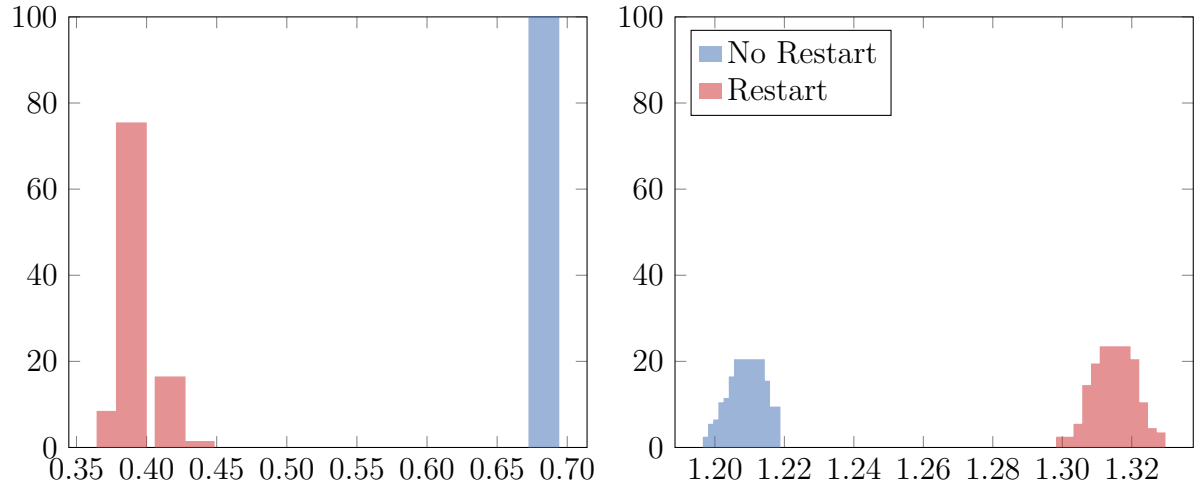
Using the line model as the forward projector yields the results shown in Figure B.30 for AB-GMRES, and using the strip model yields the results shown in Figure B.31. For AB-GMRES(∞), NCP underestimates the optimal iteration but is consistent. For AB-GMRES(5), NCP also underestimates and has more than one to two pillars, however, the iteration ratios are only 0.1 separated leading to almost the same error ratios.



(a) The ratios $R_{i,\text{ncp}}$ for AB-GMRES.

(b) The ratios $R_{e,\text{ncp}}$ for AB-GMRES.

Figure B.30: A histogram showing the ratios (5.3) to the left and (5.4) to the right when using AB-GMRES with the line model as the forward projector and for fan beam geometry. We both consider AB-GMRES(∞) (no restart) and AB-GMRES(5) (restart).



(a) The ratios $R_{i,\text{ncp}}$ for AB-GMRES.

(b) The ratios $R_{e,\text{ncp}}$ for AB-GMRES.

Figure B.31: A histogram showing the ratios (5.3) to the left and (5.4) to the right when using AB-GMRES with the strip model (10 rays per strip) as the forward projector and for fan beam geometry. We both consider AB-GMRES(∞) (no restart) and AB-GMRES(5) (restart).

B.6 Real Data

For the interested reader, we here provide the restarted ABBA solutions for $p = 3$ (Figure B.32) and $p = 10$ (Figure B.33) for the Helsinki Tomography example. Using $p = 3$, we reach the optimum very late compared to using $p = 10$. We do obtain the same relative error at the optimum for all solvers.

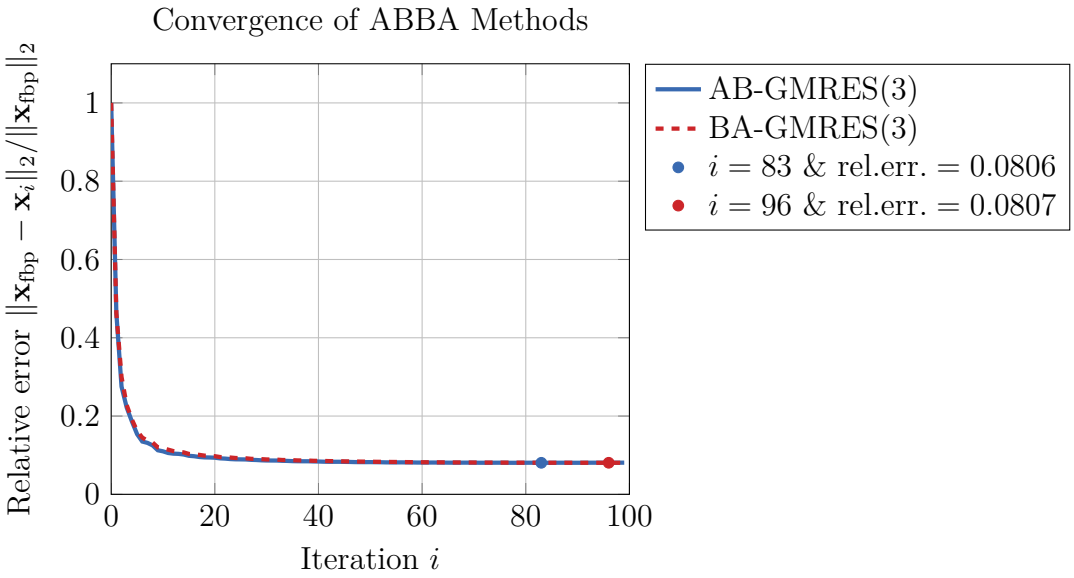


Figure B.32: The convergence history of AB-GMRES(3) and BA-GMRES(3) when comparing the reconstructions from the ABBA methods with $\mathbf{x}_{\text{fpb}}$.

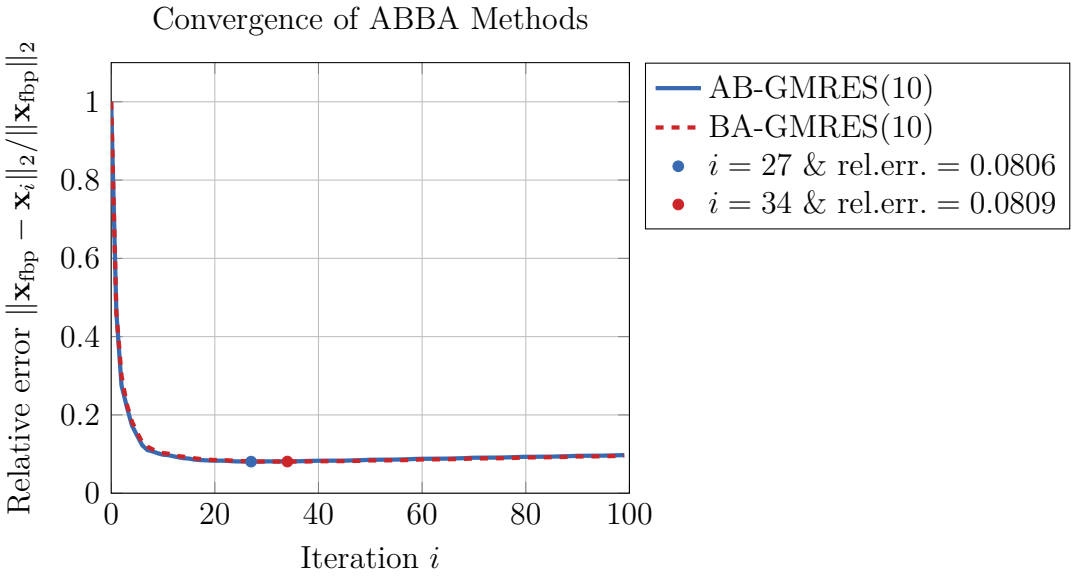


Figure B.33: The convergence history of AB-GMRES(10) and BA-GMRES(10) when comparing the reconstructions from the ABBA methods with $\mathbf{x}_{\text{fpb}}$.

APPENDIX C

ABBA-GMRES Toolbox

This thesis has resulted in a public ABBA-GMRES toolbox [23]. The toolbox includes the iterative Krylov subspace solvers AB- and BA-GMRES with all the extensions discussed in the report.

The toolbox has three main files which are found in the `src` folder. The first two files, `ct_setup.py` and `projector_setup.py` are used when using ASTRA and TIGRE to create the forward and back projectors. The third file, `solvers.py`, include the functions AB-GMRES and BA-GMRES with the associated stop criteria. We will in the following sections go into more detail and we will end this chapter with some examples which have been provided in the `examples` folder.

C.1 Built in Operators

The toolbox makes it possible to use the two open-source packages ASTRA and TIGRE to generate the projectors $\mathbf{A}$ (forward projector) and $\mathbf{B}$ (back projector). To use those, we need to import the two files `ct_setup.py` and `projector_setup.py`.

C.1.1 CT Setup

The `ct_setup.py` include the two classes `ct_tigre` and `ct_astra`. We will briefly state the required input parameters.

ct_tigre

The class ct_tigre setup the necessary parameters for a 2D X-ray CT problem in TIGRE.

```
class ct_tigre:
    '''Setup for TIGRE toolbox'''
    def __init__(self, num_pixels, num_angles, num_dets, angles, fp_model,
        bp_model, proj_geom, source_origin, source_det, det_width):
```

The input parameters:

- `num_pixels`: int
  The number of pixels in the reconstruction image $\mathbf{x}$. The image $\mathbf{x}$ must be square, thus, the dimension is num_pixels $\times$ num_pixels.

- `num_angles`: int
  The number of view angles.

- `num_dets`: int
  The number of detector elements in the 1D detector.

- `angles`: array of int or floats
  The view angles used to obtain the measurements (sinogram). The angles ***must*** be in radians.

- `fp_model`: string
  The forward projection model $\mathbf{A}$. TIGRE provides two forward projectors:
  - 'Siddon' (known from the report as the line model)
  - 'interpolated' (known from the report as Joseph's model)

- `bp_model`: string
  The back projection model $\mathbf{B}$. TIGRE provides two back projectors:
  - 'matched' which is the exact transpose of the forward projector $\mathbf{B} = \mathbf{A}^T$ (resulting in a matched projector pair).
  - 'FDK' which is the inexact transpose of the forward projector $\mathbf{B} \neq \mathbf{A}^T$ (resulting in an unmatched projector pair).

- `proj_geom`: string
  The projection geometry:
  - 'parallel' when having parallel beam geometry.
  - 'cone' when having fan beam geometry. Note, cone refers to having a 3D CT problem but we have written the code to match a 2D setup.

- `source_origin`: int or float
  The distance between the source and the origin/center is in mm.

- `source_det`: int or float
  The distance between the source and the detector is in mm.

- `det_width`: int or float
  The detector width.

ct_astra

The class ct_astra setup the necessary parameters for a 2D X-ray CT problem in ASTRA.

```
class ct_astra:
    '''Setup for ASTRA toolbox'''
    def __init__(self, num_pixels, num_angles, num_dets, angles,
        proj_model, proj_geom, source_origin, origin_det, det_width, GPU =
        True):
```

The input parameters:

- `num_pixels`: int
  The number of pixels in the reconstruction image $\mathbf{x}$. The image $\mathbf{x}$ must be square, thus, the dimension is num_pixels $\times$ num_pixels.
- `num_angles`: int
  The number of view angles.
- `num_dets`: int
  The number of detector elements in the 1D detector.
- `angles`: array of int or floats
  The view angles used to obtain the measurements (sinogram). The angles ***must*** be in radians.
- `proj_model`: string
  The forward projection model $\mathbf{A}$. ASTRA provides different forward projectors depending on the projection geometry and if we use GPU or CPU:
  - Parallel beam geometry yields:
    * When using CPU, we can choose between: 'line', 'strip', and 'linear' (known as Joseph's model in this report)
    * When using GPU, we can only choose 'linear' which is default no matter what is stated.
  - Fan beam geometry yields:
    * When using CPU, we can choose between: 'line' and 'strip'.
    * When using GPU, we can only choose 'linear' which is default no matter what is stated.
- `proj_geom`: string
  The projection geometry:

  - 'parallel' when having parallel beam geometry.
  - 'fanflat' when having fan beam geometry.

- `source_origin`: int or float
  The distance between the source and the center of rotation (the unit is: the number of pixel side lengths for reconstruction pixels).

- `origin_det`: int or float
  The distance between the center of rotation and the detector array (the unit is: the number of pixel side lengths for reconstruction pixels).

- `det_width`: int or float
  The detector width.

- `GPU`: True or False

  - GPU = True, ASTRA uses the GPU to construct the projector pair (unmatched). *(Default.)*
  - GPU = False, ASTRA uses the CPU to construct the projector pair (matched).

C.1.2 Projector Setup

TIGRE Projectors

The classes `fp_tigre` and `bp_tigre` setup the forward and back projectors, respectively, from TIGRE. They need only one input parameter which is the CT setup `ct_tigre` from `ct_setup.py`.

```
class fp_tigre:
    '''TIGRE forward projector
    Forward projection models: 'Siddon' (Line) or 'interpolated' (Joseph)
    '''
    def __init__(self,ct_tigre):
```

```
class bp_tigre:
    '''TIGRE back projector
    Back projection models: 'matched' or 'FDK'
    '''
    def __init__(self,ct_tigre):
```

ASTRA Projectors

The classes `fp_astra` and `bp_astra` setup the forward and back projectors, respectively, from ASTRA. They need only one input parameter which is the CT setup `ct_astra` from `ct_setup.py`.

```
class fp_astra:
    '''ASTRA forward projector
    Forward projection models: 'line', 'strip', or 'linear' (Joseph)
    '''
    def __init__(self,ct_astra):
```

```
class bp_astra:
    '''ASTRA back projector'''
    def __init__(self,ct_astra):
```

C.2 Custom Operators

The toolbox makes it possible for the user to provide their own forward and back projectors. It requires that the user defines a custom class where the `__matmul__` function computes a matrix-vector product of the projector given an input vector x. See the example below:

```
class custom_projector:
    # Compute matrix-vector product
    #   y = A*x
    def __matmul__(self,x):
        # Your code goes here
        # y = ...
        return y
```

C.3 Solvers

The ABBA methods take the same input parameters and return the same output parameters. Thus, the documentation for AB-GMRES and BA-GMRES has been merged.

```
AB_GMRES(A, B, b, iter, m, n, num_angles, p = 0, stop_rule = 'NO', eta =
    0, tau = 1.02, x0 = 0):
```

```
BA_GMRES(A, B, b, iter, m, n, num_angles, p = 0, stop_rule = 'NO', eta =
    0, tau = 1.02, x0 = 0):
```

Input parameters:

- `A`: matrix of size $m \times n$ or abstract matrix with `__matmul__` defined
  The forward projection operator.
- `B`: matrix of size $n \times m$ or abstract matrix with `__matmul__` defined
  The backward projection operator.
- `b`: vector of size m
  The measurements/sinogram.
- `iter`: int
  The maximum number of iterations.
- `m`: int
  The total number of pixels in the sinogram (length of `b`).
- `n`: int
  The total number of pixels in the image (length of `x0`).
- `num_angles`: class
  The total number of view angles in the CT setup.
- `p`: int
  The restart parameter. Note, if `p=iter` or `p=0`, restart is not used. *(Default is 0.)*
- `stop_rule`: string
  The stopping rules are:
  - 'DP' for using the Discrepancy Principle.
  - 'NCP' for using the Normalized Cumulative Periodogram.
  - 'NO' or any other string results in not using a stopping rule. *(Default is 'NO'.)*
- `eta`: int or float
  The noise level used in the stopping rule DP. *(Default is 0.)*
- `tau`: int or float
  The safety factor in the stopping rule DP. *(Default is 1.02.)*
- `x0`: vector of size n
  The initial guess of the solution. *(Default is a zero vector.)*

Output parameters:

- `X`: Solution matrix, kth column is the solution to the kth iteration.
- `R`: Residual matrix, kth column corresponds to the residuals for the kth iteration.

C.4 Examples within the Toolbox

To demonstrate the use of the classes and functions within the toolbox, some examples have been made. This section will briefly describe each example. An overview of the examples is given below:

- Example 1: How to construct a CT problem with ASTRA and TIGRE.
- Example 2: How to use the ABBA methods.
- Example 3: How to use restart in the ABBA methods.
- Example 4: How to use the stopping rules in the ABBA methods.

C.4.1 Example 1

Example 1 introduces the use of the `ct_setup.py` classes. The example shows how to construct a CT problem with ASTRA and TIGRE. We start by loading data of the exact image of the interior (the Shepp Logan phantom) and Figure C.1 shows the exact image $\bar{\mathbf{x}}$.

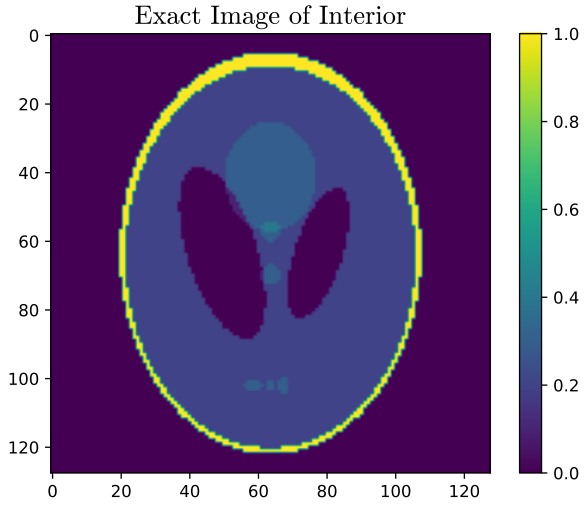
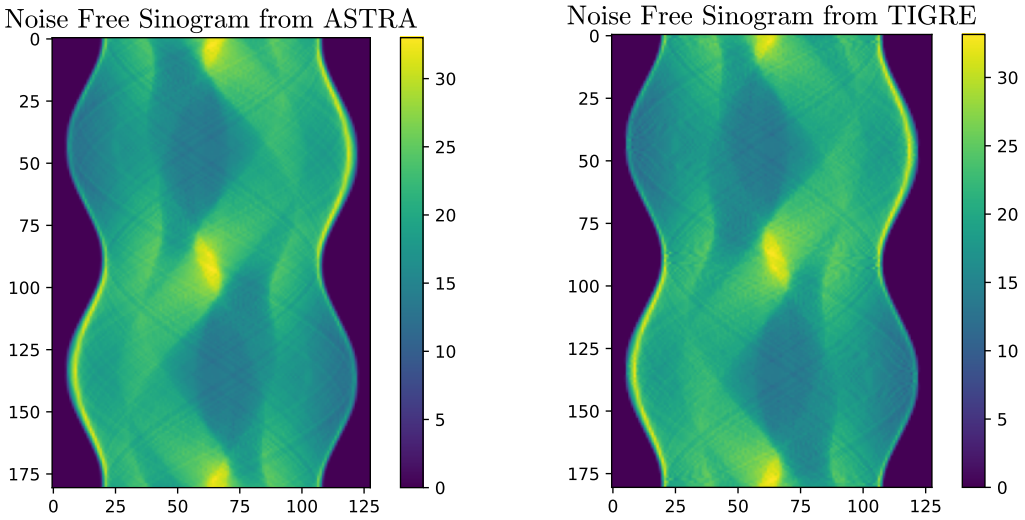


Figure C.1: The exact image of the Shepp Logan phantom.

We then setup the CT specifications in ASTRA and TIGRE and use the toolboxes to create the exact sinograms $\bar{\mathbf{b}}$ that are shown in Figure C.2.



(a) ASTRA, using fan beam geometry with Joseph's model as the forward projector and an unmatched back projector.

(b) TIGRE, using fan beam geometry with the line model as the forward projector and FDK as the back projector.

Figure C.2: A noise-free sinogram created with ASTRA and TIGRE, respectively.

C.4.2 Example 2

Example 2 demonstrates how to use the ABBA methods. We add noise to the sinogram from ASTRA (see Example 1) and create the abstract matrices $\mathbf{A}$ and $\mathbf{B}$ using `projector_setup.py`. The convergence history for AB- and BA-GMRES(∞) are shown in Figure C.3 and the optimal reconstructions are shown in Figure C.4.

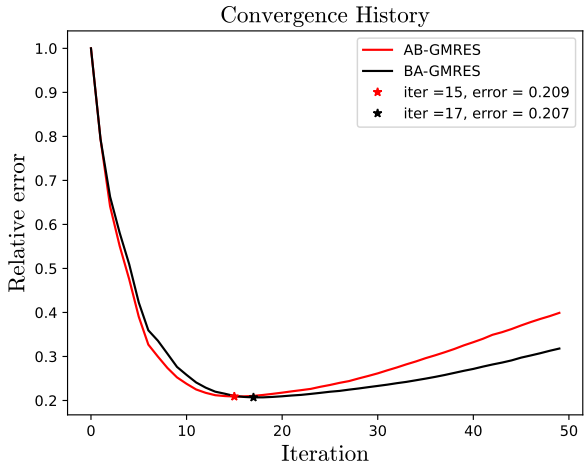


Figure C.3: The convergence history for the ABBA methods when using the CT setup from ASTRA.

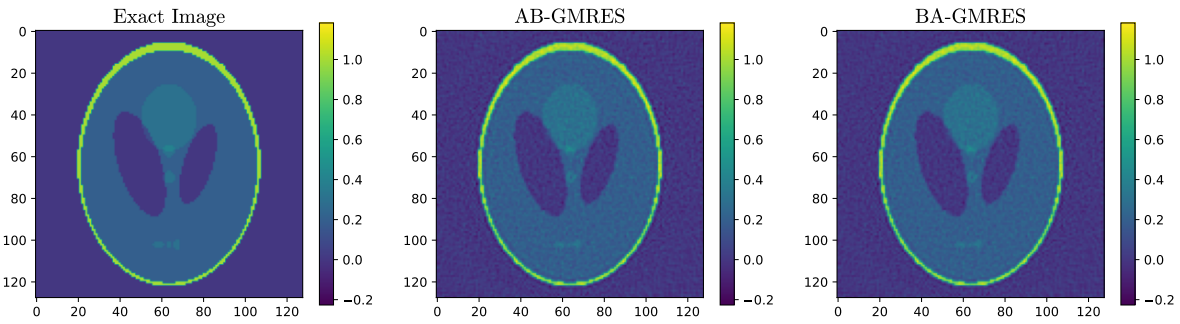


Figure C.4: The optimal reconstructions from the ABBA methods when using the CT setup from ASTRA.

C.4.3 Example 3

Example 3 demonstrates the use of restarted GMRES. We consider BA-GMRES(5) and BA-GMRES(6) with the maximum number of iterations being 100. Thus, we show that the ABBA methods can handle values of p which is not a divisor of the maximum number of iterations. Figure C.5 shows the convergence history and Figure C.6 shows the optimal reconstructions.

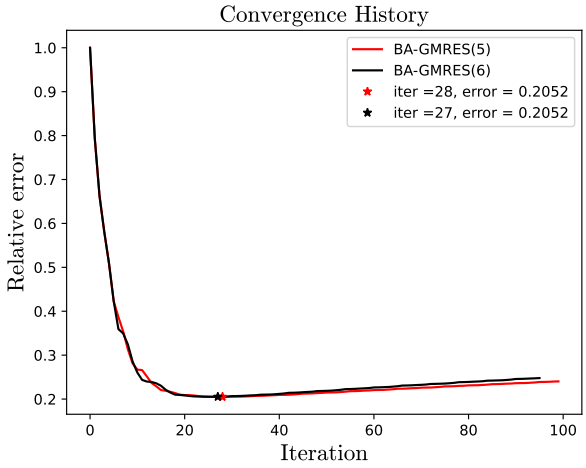


Figure C.5: The convergence history for BA-GMRES when $p = 5$ and $p = 6$.

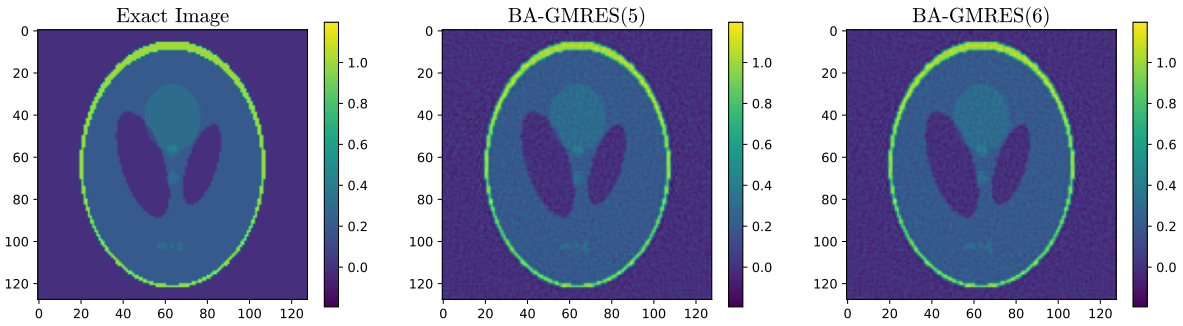


Figure C.6: The optimal reconstructions for BA-GMRES(5) and BA-GMRES(6).

C.4.4 Example 4

Example 4 illustrates how to use the stopping rules (SR) in the ABBA methods. Figure C.7 shows the convergence history for different choices of stopping rules and Figure C.8 shows the reconstructions found based on which choice of stopping rule we choose.

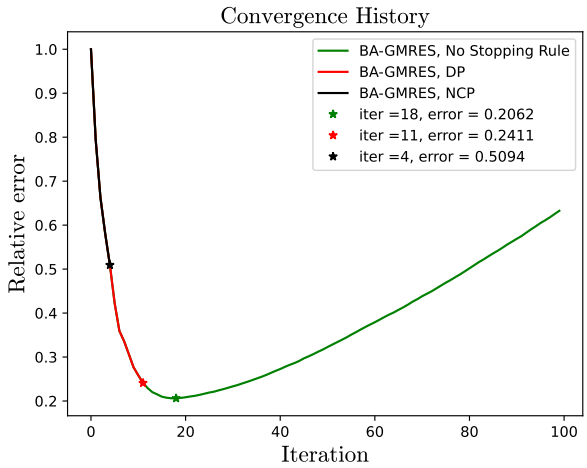


Figure C.7: The convergence history for BA-GMRES(inf) when having no stopping rule, using DP as the stopping rule, and using NCP as the stopping rule.

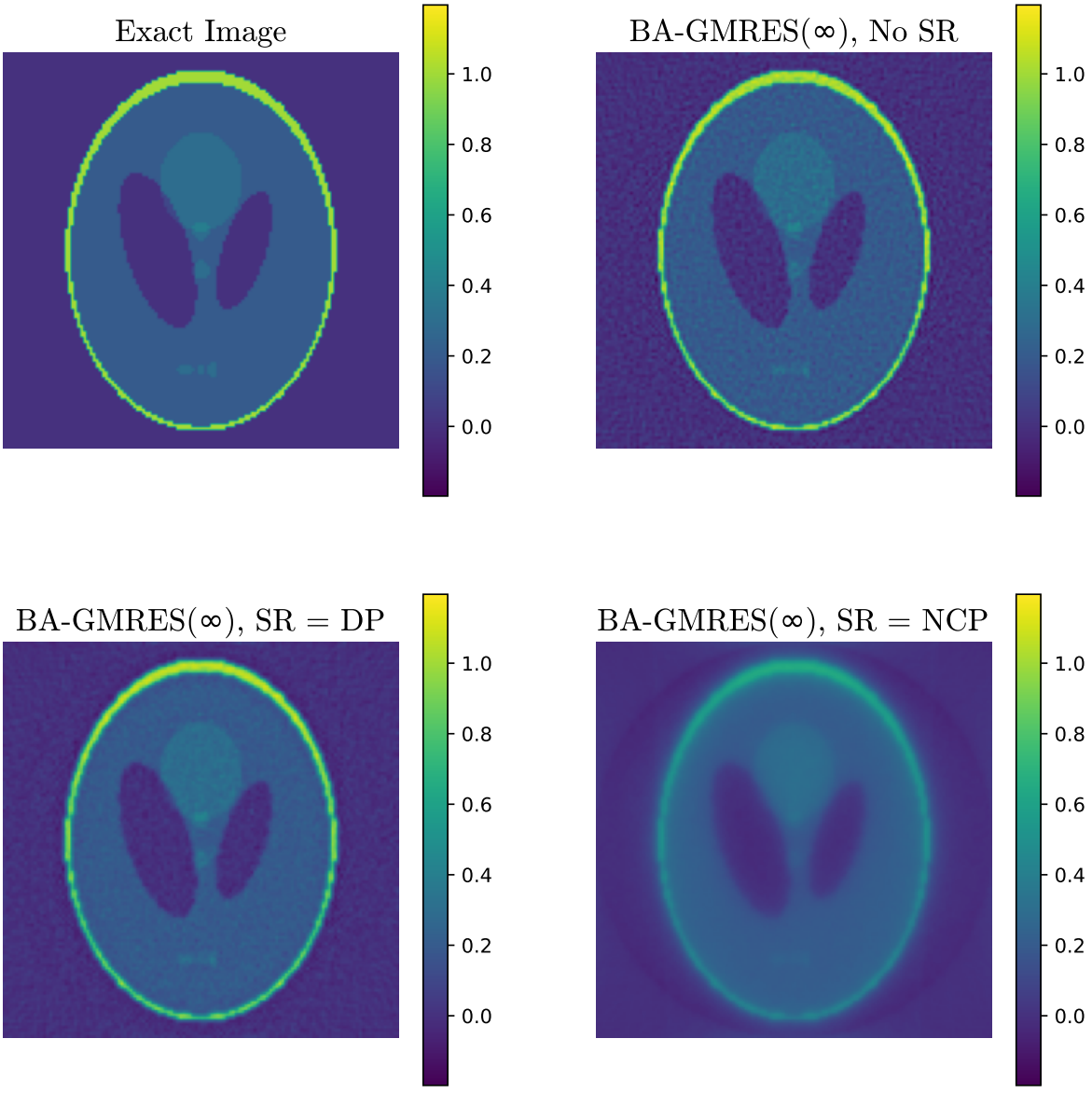


Figure C.8: The top left image shows the exact image. The top right image shows the optimal reconstruction. The bottom left image shows the solution found with the stopping rule (SR) DP, and the bottom right image shows the solution found with the NCP stopping rule.

APPENDIX D

Project Plan Evaluation

The primary plan for the project is shown in Figure D.1 with the blue boxes in the programming area and green boxes in the report area. The red boxes illustrate how it went.

The first 8 weeks followed the plan. Then we had some challenges with TIGRE's Joseph model which made us consider restart and DP instead of looking at real data while we would figure out what to do with TIGRE. Thus, a change of plans. The documentation was limited for the software packages so we also contacted the people behind ASTRA and TIGRE to gain more information. Moreover, Professor Emil Y. Sidky had some things he needed to do before he could send the user domain, therefore, working with the user domain was postponed.

In week 44, I got sick for two weeks which was also not a part of the plan. However, when I came back I continued working with the stopping criterion NCP and started looking at the user domain. At the end of the project period, we considered real data and chose to build a public GitHub repository called the ABBA-GMRES toolbox [23].

As discussed, the structure of the plan changed during the thesis as complications appeared but we achieved everything that was planned.

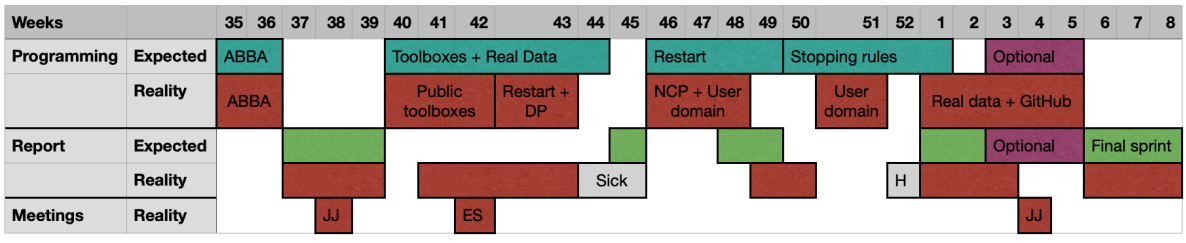


Figure D.1: An overview of the project. The blue boxes in the programming area and the green boxes in the report area illustrate the expected project plan. The red boxes illustrate how it went. The grey boxes illustrate weeks away from the project where "H" denotes the Christmas holiday. Moreover, the initials JJ is Jakob Sauer Jørgensen and ES is Emil Y. Sidky.

Bibliography

[1] Cyril Allignol et al. "Constraint programming for air traffic management: A survey." In: *The Knowledge Engineering Review* 27 (September 2012), pages 361–392.

[2] W. E. Arnoldi. "The principle of minimized iterations in the solution of the matrix eigenvalue problem." eng. In: *Quarterly of Applied Mathematics* 9 (1951), pages 17–29.

[3] *ASTRA - Tomographic Reconstruction toolbox.* `https://github.com/astra-toolbox/astra-toolbox`. Accessed: 2022-10-13. 2015.

[4] *ASTRA Documentation.* `https://www.astra-toolbox.com/`. Accessed: 2022-10-13.

[5] A. Biguri. "Iterative Reconstruction and Motion compensation in Computed Tomography on GPUs." eng. In: (2018).

[6] Ander Biguri et al. "TIGRE: A MATLAB-GPU toolbox for CBCT image reconstruction." eng. In: *Biomedical Physics and Engineering Express* 2 (2016), page 055010.

[7] Åke Björck. *Numerical methods for least squares problems.* eng. SIAM Soc. for Industrial and Applied Mathematics, 1996.

[8] Chun Yueh Chiang and Matthew M. Lin. "The eigenvalue shift technique and its eigenstructure analysis of a matrix." eng. In: *Journal of Computational and Applied Mathematics* 253 (2013), pages 235–248.

[9] Yiqiu Dong et al. "Fixing Nonconvergence of Algebraic Iterative Reconstruction with an Unmatched Backprojector." und. In: (2019).

[10] William Ford. *Numerical Linear Algebra with Applications: Using MATLAB.* eng. Elsevier Inc., 2014, pages 1–602.

[11] GP Han, ZR Liang, and JS You. "A fast ray-tracing technique for TCT and ECT studies." eng. In: *1999 Ieee Nuclear Science Symposium - Conference Record, Vols 1-3* 3 (1999), pages 1515–1518.

[12] Per Christian Hansen. *Convergence & Non-Convergence of Algebraic Iterative Reconstruction Methods.* `http://people.compute.dtu.dk/pcha/mytalks.html`. Presented at the UC Irvine Inverse Problems Seminar (May 19, 2022).

[13] Per Christian Hansen. *Discrete Inverse Problems.* Society for Industrial and Applied Mathematics, 2010.

[14] Per Christian Hansen. *Implementation of the NCP method for CT problems.* `http://people.compute.dtu.dk/pcha/ABBA/index.html`. Used in P. C. Hansen, K. Hayami, and K. Morikuni, GMRES methods for tomogarphic reconstruction with an unmatched back projector, J. Comp. Appl. Math., 413 (2022).

[15] Per Christian Hansen. "REGULARIZATION TOOLS: A Matlab package for analysis and solution of discrete ill-posed problems." In: *Numerical Algorithms* 6 (March 1994), pages 1–35.

[16] Per Christian Hansen, Ken Hayami, and Keiichi Morikuni. "GMRES methods for tomographic reconstruction with an unmatched back projector." en. In: *Journal of Computational and Applied Mathematics* 413 (October 2022), page 114352.

[17] Per Christian Hansen, Jakob Sauer Jørgensen, and William R.B. Lionheart, editors. *Computed Tomography: Algorithms, Insight, and Just Enough Theory.* English. Society for Industrial and Applied Mathematics, 2021.

[18] Ken Hayami, Jun Feng Yin, and Tokushi Ito. "GMRES methods for least squares problems." eng. In: *Nii Technical Reports* 2007 (2007), pages 1–28.

[19] *Helsinki Tomography Challenge 2022 Data.* `https://www.fips.fi/HTCdata.php#anchor1`. Accessed: 2023-02-10.

[20] K. Joost Batenburg and Jan Sijbers. "Experiences with Cell-BE and GPU for tomography." eng. In: (2015).

[21] Jakob Sauer et. al. Jørgensen. "Core Imaging Library - Part I: a versatile Python framework for tomographic imaging." eng. In: *Philosophical Transactions. Series A, Mathematical, Physical, and Engineering Sciences* (2021).

[22] P. M. Joseph. "An improved algorithm for reprojecting rays through pixel images." eng. In: *Ieee Transactions on Medical Imaging* MI-1 (1982), pages 192–196.

[23] Maria Knudsen. *ABBA-GMRES Toolbox.* `https://github.com/maria120123/ABBA-GMRES.git`. Accessed: 2023-02-26.

[24] CC Paige and MA Saunders. "LSQR - An Algorithm For Sparse Linear-Equations and Sparse Least-Squares." eng. In: *Acm Transactions on Mathematical Software* 8 (1982), pages 43–71.

[25] Evangelos et. al. Papoutsellis. "Core Imaging Library - Part II: multichannel reconstruction for dynamic and spectral tomography." eng. In: *Philosophical Transactions. Series A, Mathematical, Physical, and Engineering Sciences* (2021).

[26] Youcef Saad and Martin H. Schultz. "GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems." en. In: *SIAM Journal on Scientific and Statistical Computing* 7 (July 1986), pages 856–869.

[27] Emil Y. Sidky et al. "Iterative image reconstruction for CT with unmatched projection matrices using the generalized minimal residual algorithm." und. In: (2022).

[28] *TIGRE - Tomographic Reconstruction toolbox.* `https://github.com/CERN/TIGRE`. Accessed: 2022-10-13.

[29] Wim Van Aarle et al. “Fast and flexible X-ray tomography using the ASTRA toolbox.” eng. In: *Optics Express* 24 (2016), pages 25129–25147.

{2.7182818284}ϑωερτυθιοπσδφγηξκλ