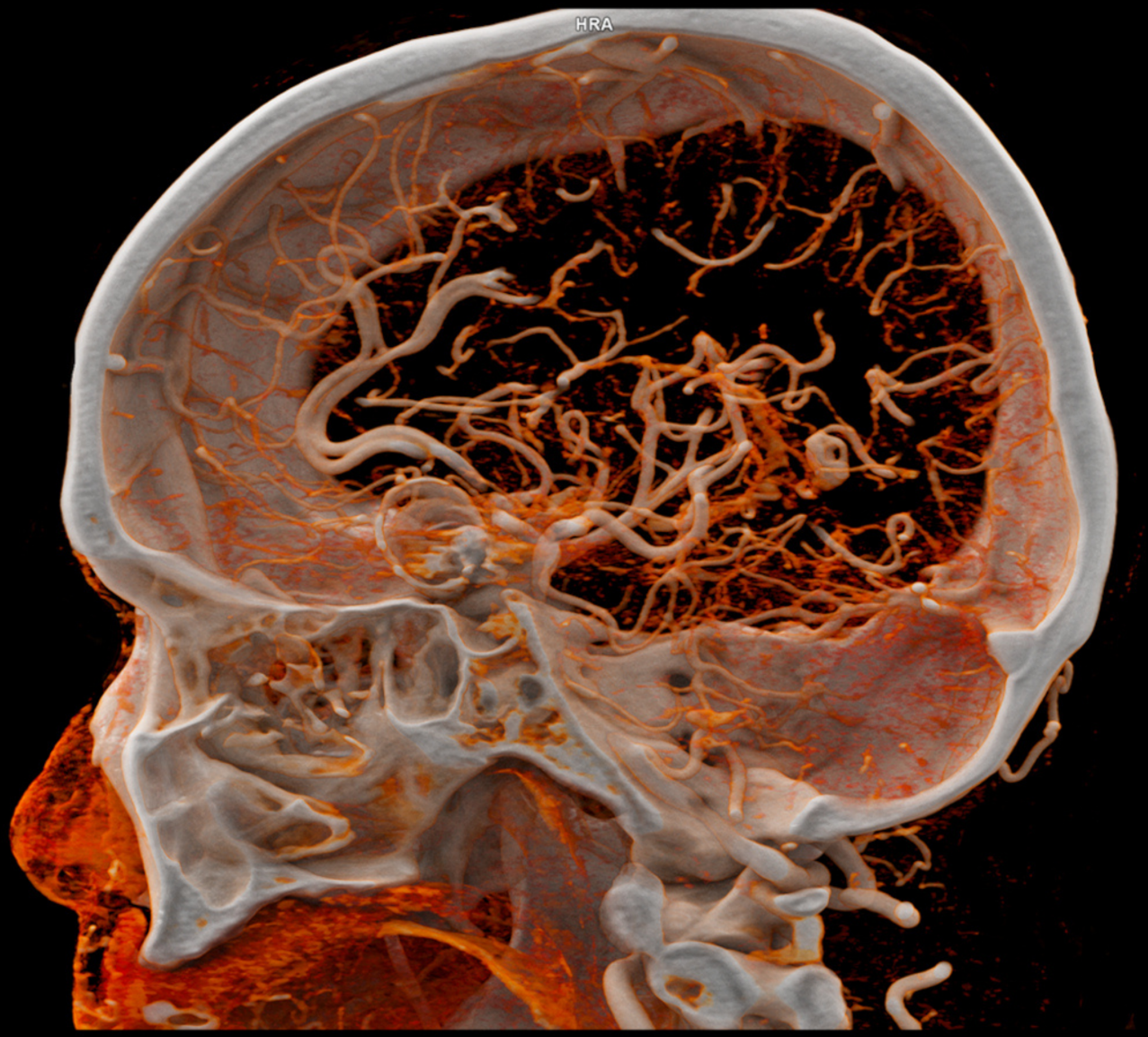
Joint Reconstruction Spectral CBCT for Proton Therapy

Daniel Moens



Delft University of Technology

TU Delft

Joint Reconstruction Spectral CBCT for Proton Therapy

by

Daniel Moens

Instructors:	M. Goorden & H. Kekkonen
Project Duration:	February, 2025 - September, 2025
Faculties:	Faculty of Applied Physics & Faculty of Applied Mathematics, Delft

Cover:	CT Angiography of the Cerebral Arteries; Photon-Counting CT by Siemens Healthineers
Style:	TU Delft Report Style, with modifications by Daan Zwaneveld

TU Delft

Abstract

By implementing cone-beam computed tomography (CBCT) into proton therapy radiation units, a predetermined treatment plan could be updated prior to each treatment fraction according to the changing anatomy of the patient for better dose distribution. However, CBCT does not produce high enough image quality compared to fan-beam CT (FBCT), which nowadays is used to build a treatment plan based on the stopping power ratio (SPR) of the objective tissue in the body. Spectral CBCT is a promising method to potentially increase the image quality of conventional CBCT. A provided joint reconstruction spectral CBCT algorithm in `MATLAB` is used to determine whether the low image quality of CBCT can be improved, as joint reconstruction algorithms have been proven to improve image quality for FBCT in practical experiments. The provided code is converted to `Python`, after which equivalent results are ensured using comparative analysis. A phantom with multiple biological materials is then implemented in this acquired `Python` code to investigate the quality of the reconstruction images. Moreover, SPR maps and a VMI are constructed from these images and their quality determined.

The results show that for 10 to 12 iterations, the used reconstruction provides the reconstructed images with the lowest mean squared error (MSE). For higher iterations, the image becomes oversmoothed and loses quality. In future research, the used joint reconstruction algorithm should be compared to non-joint reconstruction algorithms to investigate the impact of this technique on the image quality, after which it could be applied to more realistic data.

Contents

Nomenclature

Abbreviations

Abbreviation	Definition
CT	Computed Tomography
CBCT	Cone-Beam Computed Tomography
FBCT	Fan-Beam Computed Tomography
SECT	Single-Energy Computed Tomography
DECT	Dual-Energy Computed Tomography
PCCT	Photon-Counting Computed Tomography
PCD	Photon-Counting Detector
EID	Energy Integrating Detector
SPR	Stopping Power Ratio
VMI	Virtual Monochromatic Image
LASSO	Least Absolute Selection & Shrinkage Operator
TV	Total Variation
MSE	Mean Square Error
CRC	Contrast Recovery Coefficient
ROI	Region Of Interest

Symbols

Symbol	Definition	Unit
μ	Attenuation Coefficient	m^{-1}
I	Intensity	$\mathrm{J\,m^{-2}\,s^{-1}}$
ρ	Density	$\mathrm{g\,cm^{-3}}$
μ/ρ	Mass Attenuation Coefficient	$\mathrm{cm^2\,g^{-1}}$
Φ_{eff}	Effective X-Ray Spectrum	J^{-1}
Y	Measured Photon Count	-
$\hat{Y}$	Expected Photon Count	-
A_i^b	Density Line Integral	$\mathrm{g\,cm^{-2}}$
α_j^b	Density Contribution	$\mathrm{g\,cm^{-3}}$
λ_b	Regularisation Tuning Parameter	-
γ_b	Huber Tuning Parameter	-

1 Introduction

External beam radiotherapy is used to combat cancer cells within the body by directing a beam of particles toward malignant cells with the aim of destroying them. Proton therapy is a form of external beam radiotherapy with substantially better dose distribution than more conventional photon therapy [1]. It is essential to acquire detailed knowledge of the anatomy of the patient to determine a treatment plan, especially for proton therapy, since protons have a finite range and therefore release their radiation dose extremely localised [2]. The treatment plan is based on the stopping power ratio (SPR), which is the proton stopping power of a material relative to that of water. The proton stopping power is a material property that dictates the path length of the protons passing through the material [3].
Nowadays, a fan-beam computed tomography (FBCT) scanner is used to calculate SPR values to accurately determine the targeted region and surrounding organs at risk within the body, after which a treatment plan is established before starting treatment [4]. For this, it is also relevant that CT scanners have the ability to reconstruct a high distinction between soft tissue materials, both in the SPR maps and in the final reconstruction images, since tumours and most soft tissue have very similar material properties [5]. The treatment then consists of many treatment fractions over many different days. However, since the treatment plan must be determined in advance, loss of positional precision during a treatment fraction is probable due to changes in the anatomy of a patient [6]. Since FBCT scanners need a spiral around a specific part of the body to create a reconstructed image, these types of scanners are much larger compared to cone-beam CT (CBCT) scanner units. Therefore, it is not possible to integrate FBCT with radiotherapy units to update the treatment plan according to the changing anatomy of the patient directly before a treatment fraction. However, it is possible to integrate CBCT scanners into radiation therapy due to their spatial efficiency compared to FBCT scanners. Currently, CBCT scanners integrated in radiotherapy units are used to check if the patient is positioned correctly and thus aligned with the predetermined treatment plan. However, CBCT does not produce a high enough image quality to be able to correct the initial treatment plan directly according to the current patient anatomy [7]. Integration of CBCT imaging with better quality in radiotherapy units allows for pretreatment adaptation of the necessary radiation therapy, resulting in greater dose accuracy during treatment.
One way to possibly obtain higher image quality from CBCT imaging is to use spectral CBCT, which allows for integration of the energy dependence of the interaction between X-rays and matter into the reconstruction process. Spectral CT can be performed in a number of different ways. In this thesis, photon counting CT (PCCT) is considered. This type of CT uses detectors, typically made from semiconductors, that are able to determine the energy of individual X-

ray photons [8]. This is in contrast to the conventional energy-integrating detector (EID), which essentially determines the total energy of the photons that arrive on the detector. The individual photons detected by a PCD are sorted into discrete energy intervals. Usually, an image is reconstructed for each of these energy bins, after which the images are combined to form a final reconstruction image or to extract information on material properties. However, this technique does not use all available detector information in an optimal way, as it [9]. Therefore, this thesis focuses on joint reconstruction, which is the method of combining all information from the different energy bins into a reconstruction image in a single step. Joint reconstruction has been applied in practical experiments to FBCT and has shown promise in improving image quality in the reconstructed image [7]. This thesis will investigate whether joint reconstruction allows for a higher image quality in CBCT.

Mechlem *et al.* [10] provided a joint reconstruction material decomposition and reconstruction algorithm for PCCT. This algorithm was later implemented into a `MATLAB` code by Mory *et al.* [11] to work for spectral FBCT, after which it was adapted to CBCT by van Leenen [12]. The goal of this thesis is to implement this `MATLAB` code equivalently into `Python`, so that in the future, the code can be made more efficient by allowing integration with open software packages available in `Python`. After validating the new code, a phantom with multiple biological materials is built to investigate the image quality of the reconstruction images. In addition, SPR maps are built from the reconstruction images to investigate whether these are reconstructed accurately.

The structure of this thesis is as follows. In Chapter 2, a theoretical background on a number of different topics is provided in Chapter 2 to better understand the general concept of this thesis. This contains a mathematical background on inverse problems, along with the functioning of regularisation in stabilising inverse problems. Then, an explanation of the basics of CT is provided, including different detectors, spectral CT, and the interaction between X-rays and matter on which CT is based. Lastly, some basic information about CT reconstruction is provided, along with the optimisation algorithm used in this thesis for image reconstruction. In Chapter 3, the implementation of the optimisation algorithm used is explained. Chapter 4 provides the results from the code implementation from the previous chapter, after which they are discussed.

2

Theory

This chapter provides the theoretical background required to understand the techniques and methodology used in this thesis. It introduces the mathematical framework for inverse problems, the principles of computed tomography, common reconstruction techniques, and the optimisation algorithm used for image reconstruction. Together, these topics lay the foundation for the development and evaluation of the methodology of this thesis, presented in a later chapter.

2.1. Inverse Problems

An inverse problem is a problem in which certain causal factors are calculated when the physical effects of them are known [13]. This could, for example, be the problem of finding the shape of an object when the shadow it casts is known. This is opposed to a direct problem, which aims to calculate the exact opposite; the effect is calculated from a known cause. In practice, it can be quite difficult to accurately produce an estimate of a physical cause when only measurements are known. In this chapter, a basic understanding of the mathematics behind inverse problems will be explained in detail.

Consider the forward operator matrix $\mathbf{A} \in \mathbb{R}^{k \times d}$, which represents a model of the underlying physical process and the measured physical effect $\mathbf{m} \in \mathbb{R}^k$. Here, the variable k is the number of individual, discrete data points from the measurement and d is the number of unknowns that are calculated in order to reconstruct the desired physical cause. We define a certain perturbed measurement model as follows:

$$\mathbf{m} = \mathbf{A}\mathbf{u} + \mathbf{n}\,. \tag{2.1}$$

Here, $\mathbf{n} \in \mathbb{R}^k$ is the noise present in the measurements, which is an unknown quantity. The goal of the problem is to obtain an estimate of the object of interest $\mathbf{u} \in \mathbb{R}^d$, which is the desired physical cause. The measurement $\mathbf{m}$ is known to be a discrete quantity due to the nature of measurement devices. Moreover, even though the quantity $\mathbf{u}$ describes a continuous physical process, it is also considered discrete for practical computation, since computers cannot directly handle continuous functions. The matrix formula presented in Equation (2.1) can also be written as follows:

$$\begin{bmatrix} m_1 \\ \vdots \\ m_k \end{bmatrix} = \begin{bmatrix} a_{11} & \cdots & a_{1d} \\ \vdots & \ddots & \vdots \\ a_{k1} & \cdots & a_{kd} \end{bmatrix} \cdot \begin{bmatrix} u_1 \\ \vdots \\ u_d \end{bmatrix} + \begin{bmatrix} n_1 \\ \vdots \\ n_k \end{bmatrix} . \tag{2.2}$$

Unfortunately, the noise in the problem is not known. This quantity is therefore often modelled as a random variable. We take the expected norm value of the noise to be small for the following paragraph. Assuming that the noise can be modelled in this way, we can try to solve the inverse problem by reconstructing an estimate of the quantity $\mathbf{u}$. However, when the forward matrix $\mathbf{A}$ is ill-conditioned, it is not possible to take the inverse of the matrix $\mathbf{A}$ and calculate the desired physical quantity as follows: $\mathbf{u} = \mathbf{A}^{-1}(\mathbf{m} - \mathbf{n})$. Moreover, the exact values of the noise in the system cannot be estimated, and therefore it is not possible to use the quantity $\mathbf{n}$ in this way.
The measurement model in Equation (2.1) is ill-conditioned if it breaks at least one of the following conditions for well-posed problems [14]:

1. **Existence**: There is at least one solution to the problem.
2. **Uniqueness**: There is at most one solution to the problem.
3. **Stability**: The solution changes continuously with the initial conditions.

The existence of a solution in this particular case is whether it is possible to reconstruct an estimation of $\mathbf{u}$. If there are multiple solutions, there are multiple possible estimations for $\mathbf{u}$, and the actual physical effect can only be recovered when certain *a priori* knowledge is applied to the problem.

We examine three different cases for $\mathbf{A} \in \mathbb{R}^{k \times d}$ that could cause the perturbed measurement model from Equation 2.1 to become ill-conditioned [13]. In the case where $d \neq k$, $\mathbf{A}$ is either underdetermined or overdetermined. This means that there are more unknowns than equations in the problem or more equations than unknowns, respectively. In both of these cases, one of the conditions of well-posed problems is broken. In the former, where the problem is underdetermined, the presence of more unknown variables than equations results in a problem where there are multiple possible solutions for the estimation of $\mathbf{u}$. On the other hand, if the system is overdetermined, a solution for $\mathbf{u}$ could exist if the obtained overdetermined system of equations is consistent. However, in a perturbed system the chance becomes near zero that the resulting system is consistent, as the noise is modelled by a random variable. Therefore, the system has a very high chance to break the existence condition for well-posed problems. This makes the inverse problem ill-conditioned for $d \neq k$. Moreover, the matrix $\mathbf{A}$ is not invertible in both cases, as it is not a square matrix and therefore the naïve inversion will not be possible for this case.
Now, if we assume $d = k$, we know that there exists an inversion $\mathbf{A}^{-1} : \mathbb{R}^k \to \mathbb{R}^k$, in the case where $\mathbf{A}$ has full rank. The reason why $\mathbf{A}$ could result in an ill-conditioned problem for $d = k$ is found in the behaviour of the condition number. The condition number, defined as $\kappa(\mathbf{A}) = ||\mathbf{A}|| \cdot ||\mathbf{A}^{-1}||$, is a measure of numerical sensitivity; how much the output changes in response to tiny changes in the input. The norm $|| \cdot ||$ for determining the condition number of the forward matrix $\mathbf{A}$ is assumed to be the Euclidean matrix norm. By definition, an ill-conditioned problem has a large condition number, which means that the inverse problem has high numerical sensitivity and is therefore unstable.
It is possible to show why exactly a high condition number results in high numerical sensitivity. If we consider the true value $\mathbf{u}$, we can say that $\hat{\mathbf{u}} = \mathbf{A}^{-1}\mathbf{m} = \mathbf{u} + \mathbf{A}^{-1}\mathbf{n}$, where $\hat{\mathbf{u}}$ is the result from the naïve inversion. This can also be written as $\mathbf{A}\hat{\mathbf{u}} = \mathbf{m} = \mathbf{A}\mathbf{u} + \mathbf{n}$, as the measurement

$\mathbf{m}$ must remain equal for both the perturbed and unperturbed calculations. Therefore, we can write $\mathbf{A}(\hat{\mathbf{u}} - \mathbf{u}) = \mathbf{A}\Delta\mathbf{u} = \mathbf{n}$. Then, we can relate the condition number with the relative error in the output to the relative error in the input. We consider $\frac{||\mathbf{n}||}{||\mathbf{Au}||}$. This is the ratio of the amount of noise in the system to the result of a measurement without noise. This will then be a measure for the amount of noise in the system, normalised by an unperturbed measurement, and can be seen as the relative input error in the system. Furthermore, we know that $\frac{||\Delta\mathbf{u}||}{||\mathbf{u}||}$ is the scaled ratio of the difference between the true value and the perturbed value, $\boldsymbol{\Delta}\mathbf{u}$, and the true value $\mathbf{u}$. This can be seen as the relative error in the output of the unperturbed case. If we calculate the ratio between the relative input and output errors, the following result can be obtained [15]:

$$\begin{aligned}
\frac{\frac{||\Delta\mathbf{u}||}{||\mathbf{u}||}}{\frac{||\mathbf{n}||}{||\mathbf{Au}||}} &= \frac{||\Delta\mathbf{u}|| \cdot ||\mathbf{Au}||}{||\mathbf{n}|| \cdot ||\mathbf{u}||} \\
&= \frac{||\mathbf{A}^{-1}\mathbf{n}|| \cdot ||\mathbf{Au}||}{||\mathbf{n}|| \cdot ||\mathbf{u}||} \\
&\leq \frac{||\mathbf{A}^{-1}|| \cdot ||\mathbf{n}|| \cdot ||\mathbf{A}|| \cdot ||\mathbf{u}||}{||\mathbf{n}|| \cdot ||\mathbf{u}||} = ||\mathbf{A}^{-1}|| \cdot ||\mathbf{A}|| = \kappa(\mathbf{A}) \,,
\end{aligned} \tag{2.3}$$

using $||\mathbf{Au}|| \leq ||\mathbf{A}|| \cdot ||\mathbf{u}||$, $\forall\mathbf{u}$, etc. So we get the following result, in which we can clearly see the effect of the condition number on the changes in output:

$$\frac{||\Delta\mathbf{u}||}{||\mathbf{u}||} \leq \kappa(\mathbf{A}) \frac{||\mathbf{n}||}{||\mathbf{Au}||} \,. \tag{2.4}$$

By this formula, we can see that with a very large $\kappa(\mathbf{A})$, even for small fluctuations in the noise, the relative output error can be very large. This means that there is a lot of uncertainty in the possible final reconstruction of $\mathbf{u}$, which is not desirable. Therefore, the naïve inversion $\hat{\mathbf{u}} = \mathbf{A}^{-1}\mathbf{m} = \mathbf{u} + \mathbf{A}^{-1}\mathbf{n}$ will not offer an appropriate solution for the case $d = k$. Because of this, the stability condition for well-posed problems is broken. This means that for the case $d = k$ the perturbed measurement model from Equation (2.1) becomes ill-conditioned.

As we have seen, it is generally not feasible to invert the matrix $\mathbf{A}$ to obtain an accurate estimation of $\mathbf{u}$. This is especially the case when the matrix $\mathbf{A}$ is such that the inverse problem is ill-conditioned. Moreover, direct inversion also fails when the matrix $\mathbf{A}$ is singular, or if it is over- or underdetermined. The maximum likelihood estimation method is one of many approaches to estimate a quantity $\mathbf{u}$ from measurements $\mathbf{m}$. The problem encountered in this thesis is described well by a Poisson distribution, since the measurements are counts of discrete events. This will be explained in more detail in Section 2.5.1. In the case of a discrete probability distribution, the maximum likelihood estimate can be described as the value of the parameter that assigns the highest probability to the measurement $\mathbf{m}$ [16]. To be able to use the maximum likelihood method, a likelihood function must be determined. The likelihood function is obtained by multiplying the probability mass function of the Poisson distribution for each independent measurement m_i [17]:

$$\mathcal{P}(\mathbf{m} \mid \mathbf{Au}) = \prod_i \frac{\hat{m}_i(\mathbf{Au})^{m_i} e^{-\hat{m}_i(\mathbf{Au})}}{m_i!} \,. \tag{2.5}$$

Here, $\hat{m}_i(\mathbf{Au})$ is the prediction corresponding to measurement m_i and based on the current estimate $\mathbf{u}$. Optimisation is more convenient in the logarithmic domain as it simplifies the necessary calculations. Since the logarithm is a monotone function, the estimated value maximises the likelihood function if and only if it maximises the log-likelihood function [16]. Therefore, the likelihood function is usually converted into a negative log-likelihood function:

$$\mathcal{L}(\mathbf{u}) = -\log \mathcal{P}(\mathbf{m} \mid \mathbf{Au}) = \sum_i \left(\hat{m}_i(\mathbf{Au}) - m_i \log \hat{m}_i(\mathbf{Au})\right) . \tag{2.6}$$

In this equation, $\mathcal{L}(\mathbf{u})$ is the negative log-likelihood, which is the cost function to be minimised to obtain an estimate. The final maximum likelihood estimate is therefore acquired with the following:

$$\hat{\mathbf{u}} = \underset{\mathbf{u}\in\mathbb{R}^d}{\arg\min}\, \mathcal{L}(\mathbf{u}) . \tag{2.7}$$

The maximum likelihood estimation method will not provide an accurate estimate of $\mathbf{u}$ on its own. Using this method would result in unstable and non-unique solutions, especially when direct inversion of the matrix $\mathbf{A}$ is not possible. To stabilise the problem, *a priori* knowledge is needed to be able to acquire an estimate of $\mathbf{u}$. This *a priori* knowledge will be incorporated into the problem as a convex term that is added to the negative log-likelihood function, which is called the regularisation term. Different types of regularisation and their purposes are given in the following section.

2.2. Regularisation

In order to stabilise the given inverse problem, $\mathbf{m} = \mathbf{Au} + \mathbf{n}$, a regularisation term is needed. There are multiple ways to apply regularisation to an inverse problem, depending on the type of *a priori* knowledge we have for a specific case. A certain type of regularisation can include conditions for the resulting solution, such as an amount of smoothness or sparseness [18]. Three types of regularisation terms will be explained in more detail in the following subsections; Tikhonov regularisation, LASSO regularisation, and total variation regularisation.

2.2.1. Tikhonov Regularisation

Consider the perturbed measurement model of Section 2.1, $\mathbf{m} = \mathbf{Au} + \mathbf{n}$. Tikhonov regularisation provides the negative log-likelihood function with *a priori* knowledge in the form of an ℓ^2 penalty term. The negative log-likelihood function from Formula (2.6) then becomes the following convex and differentiable optimisation problem [19]:

$$\underset{\mathbf{u}\in\mathbb{R}^d}{\arg\min}\left\{ \sum_i \left(\hat{m}_i(\mathbf{Au}) - m_i \log \hat{m}_i(\mathbf{Au})\right) + \alpha||\mathbf{u}||_2^2 \right\} , \tag{2.8}$$

where the quadratic term $||\mathbf{u}||_2^2 = \sum_{i=1}^d u_i^2$ is called the regularisation term and $\alpha > 0$ is called the tuning parameter. This regularisation term is introduced to prioritise vectors $\mathbf{w} \in \mathbb{R}^d$ that have a lower value for $||\mathbf{u}||_2^2$. For high values of the quadratic term, the cost function $\mathcal{L}(\mathbf{u}) + \alpha||\mathbf{u}||_2^2$ would increase, leading to a decrease in prioritisation, because the optimisation seeks to minimise the cost function. Because of this, Tikhonov regularisation tends to suppress solutions with large values. Vectors $\mathbf{u}$ in which large values are present, for example due to a large amount of noise, are also suppressed by this ℓ^2 penalty term. In this way, the Tikhonov regularisation is used to incorporate a certain amount of smoothness into the resulting physical quantity $\mathbf{u}$ [19], depending on the size of the parameter α. Moreover, the size of α should be considered carefully. A too small value may result in too little influence in reducing the noise, while an excessively large value would result in over-smoothing of $\mathbf{u}$, and thus a loss of detail.

Adding the convex and differentiable penalty term not only results in the problem being stabilised by the suppression of large fluctuations, but due to its convex and coercive properties, it will also have a unique solution [20]. The unique solution obtained with Tikhonov regularisation can be shown, under suitable conditions, to converge to a stable approximation of the original inverse problem. While this may not coincide with the exact true solution in the presence of noise or oscillatory behaviour, in practice it provides a much more stable and interpretable result than the maximum likelihood estimation method. Another advantage is that this problem can be solved analytically because of its differentiability.

2.2.2. LASSO Regularisation

To introduce more sparseness in the solution, we look at the Least Absolute Selection and Shrinkage Operator (LASSO) regularisation. Opposed to Tikhonov regularisation, an ℓ^1 based penalty term is introduced in the form of $||\mathbf{u}||_1$. Then, the formula for the LASSO regularisation is expressed as follows:

$$\underset{\mathbf{u}\in\mathbb{R}^d}{\arg\min}\left\{ \sum_i \left(\hat{m}_i(\mathbf{Au}) - m_i \log \hat{m}_i(\mathbf{Au})\right) + \alpha||\mathbf{u}||_1 \right\}, \tag{2.9}$$

where $||\mathbf{u}||_1 = \sum_{j=1}^{d} |\mathbf{u}_j|$ and the parameter $\alpha > 0$ is again the tuning parameter. This equation can be equivalently formulated using a constraint in the following manner [21]:

$$\underset{\mathbf{u}\in\mathbb{R}^d}{\arg\min} \sum_i \left(\hat{m}_i(\mathbf{Au}) - m_i \log \hat{m}_i(\mathbf{Au})\right), \text{ subject to } ||\mathbf{u}||_1 \leq s\,. \tag{2.10}$$

Here, a new tuning parameter s is introduced, which is related to the tuning parameter α [19]. Moreover, for every $\alpha > 0$, there exists an s such that Equations (2.9) and (2.10) have the same solution [13]. This form of regression will shrink the solution of the inverse problem towards zero, which means that a certain amount of sparseness is introduced depending on the size of the tuning parameter α. Primarily irrelevant variable coefficients, which are already close to zero, would be reduced to zero, and more influential coefficients would consequently be highlighted.

To explain this effect more clearly, let us look at a geometric illustration of the effect of ℓ^1 and ℓ^2 based regularisation in the simple case $d = 2$. Notice that the Tikhonov regularisation of Equation (2.8) can also be written as the minimisation of the negative log-likelihood cost function with an ℓ^2 norm constraint, $||\mathbf{u}||_2^2 \leq s$, just as in Equation (2.10). Again, s is a new tuning parameter which is related to the tuning parameter α, equivalent to the LASSO constraint formulation.

The geometric illustration of ℓ^1 and ℓ^2 based regularisation is shown in Figure 2.1. Here, the solution of the log-likelihood function minimisation is located in the darkest blue spot, and surrounded by ellipsoids that represent the residual errors of the minimisation, $\mathcal{L}(\mathbf{u}) = C$. The red and light blue regions on the axes are the constraints imposed by regularisation, which in this case is $||\mathbf{u}||_2^2 = u_1^2 + u_2^2 \leq s$ for Tikhonov and $||\mathbf{u}||_1 = |u_1| + |u_2| \leq s$ for LASSO regularisation, respectively. As described in Equation (2.10) and the constrained version of the Tikhonov regularisation, the regularised estimate is given by the first point where a residual contour intersects with the constraint region. From the figure it becomes clear exactly why LASSO regularisation encourages sparseness in the solution. As the constraint region for ℓ^1 regularisation has corners aligned with the u_1 and u_2 axes, resulting from the absolute value, the residual errors prefer to intersect the constraint region at these corners. This is in contrast

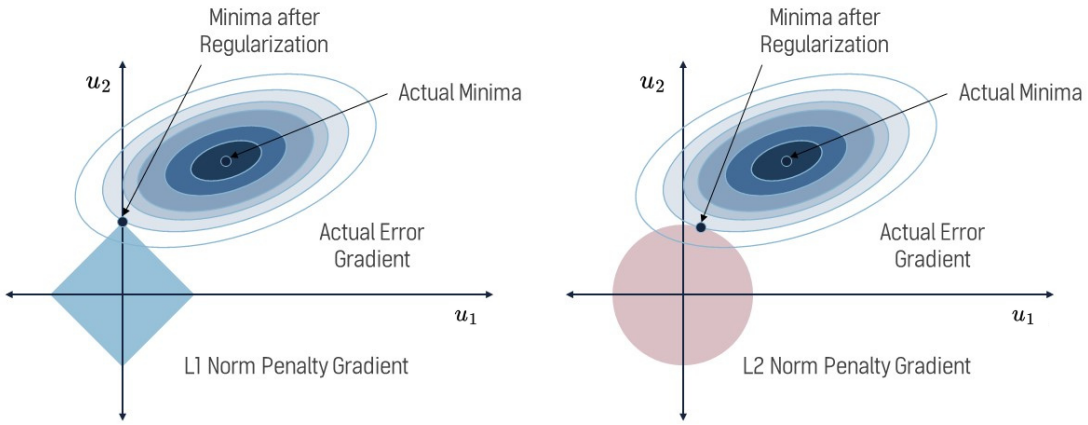


Figure 2.1: Geometrical interpretation of the effect of ℓ^1 regularisation (left) and ℓ^2 regularisation (right) on the solution of the log-likelihood function minimisation [22].

to the circular constraint region, where there is no specific preference to land in any part of the region.

2.2.3. Total Variation Regularisation

In image reconstruction, the main focus of this thesis, regularisation will mainly be used for differences between pixels in an image to restrict certain conditions on the final image. For example, it is desirable that a large difference between two pixels is preserved, as this could indicate a transition to a new part of the image. On the other hand, for small differences between pixels, it is desirable that these differences are smoothed out and pushed towards zero to encourage sparseness and, as a result, reduce the noise. Total variation (TV) regularisation is a form of ℓ^1 regularisation, which is used to introduce this type of *a priori* knowledge. We define the following equation as the general notation for ℓ^1 regularisation [13]:

$$\underset{\mathbf{u}\in\mathbb{R}^d}{\arg\min}\left\{ \sum_i \left(\hat{m}_i(\mathbf{Au}) - m_i \log \hat{m}_i(\mathbf{Au})\right) + \alpha||L\mathbf{u}||_1 \right\}, \tag{2.11}$$

where L is introduced as a discretised differential operator. As can be seen in Equation (2.9), for $L = \mathbf{I}$, the identity matrix, this equation is defined as LASSO regularisation. TV regularisation in one dimension is defined as the general formula for ℓ^1 regularisation with the following differential operator:

$$L = \frac{1}{\Delta x}\begin{bmatrix} -1 & 1 & & & & \\ & -1 & 1 & & & \\ & & \ddots & \ddots & & \\ & & & -1 & 1 & \\ & & & & -1 & 1 \\ & & & & 0 & -1 \end{bmatrix}. \tag{2.12}$$

This discretised differential operator is the approximation of a differential using the forward difference, which can also be written as $u'(x_j) \approx \big(u(x_{j+1}) - u(x_j)\big)/\Delta x$. So, essentially, instead of regularising individual values, the difference between two values is punished. In the general case, TV regularisation can be defined as follows:

$$\underset{\mathbf{u}\in\mathbb{R}^d}{\arg\min}\left\{ \sum_i \Big(\hat{m}_i(\mathbf{Au}) - m_i \log \hat{m}_i(\mathbf{Au})\Big) + \alpha||\nabla \mathbf{u}||_1 \right\}, \tag{2.13}$$

with $||\nabla \mathbf{u}||_1 = \sum_{j=1}^{d} |\nabla u_j|$. In this case, the ∇ operator is a notation for a discrete version of differentiation, since we are working with discrete quantities. The discretised differential operator from Equation (2.12) is an example of a possible discretised differentiation method and would give $\sum_{j=1}^{d} |\nabla u_j| = \sum_{j=1}^{d} |u_{j+1} - u_j|/\Delta x$. Like LASSO regularisation, TV regularisation promotes sparseness due to the ℓ^1 norm. However, by introducing the gradient in the equation, now differences between values will be pushed towards zero. This means that when two values are very close to one another, the regularisation pushes them towards each other, creating a difference of zero, while large differences between values will be less punished.

2.3. Basics of Computed Tomography

Computed tomography (CT) scanners are used to create images of the internal structure of a patient, with the aim of diagnosing an injury or a disease or planning medical treatments such as surgery or radiation therapy [23]. There are many different types of CT scanners, with differences depending on the manufacturer, purpose, and innovation. In the following sections, a detailed theoretical background is given on CT scanners and their functioning. This includes different types of X-ray detectors, the difference between fan-beam CT and cone-beam CT, and the innovative spectral CT. Furthermore, an explanation of the attenuation coefficient is given, which is the material characteristic that governs X-ray transport, determines CT data acquisition, and therefore forms the reconstructed image of the scanned sample. Lastly, a background on the functioning of proton therapy will be provided, along with information on proton stopping power and SPR maps.

A CT scanner consists of a torus-shaped machine, called a gantry, as can be seen in Figure 2.2. In the cavity of the gantry, a patient table is located on which the patient or object to scan can be placed [24]. An X-ray emitter and an X-ray detector are located inside the gantry, both of which revolve directly opposite each other around the sample on the patient table. The X-ray emitter usually consists of an X-ray tube, which converts electrons to polychromatic X-rays, which are composed of a broad range of different photon energies. Moreover, there are typically two collimators, one at the source and one at the detector, which absorb low-energy X-rays [24]. The source collimator forces the X-rays produced in the X-ray tube to a certain desired shape, typically in a fan or cone shape. The detector collimator, also known as an anti-scatter grid, ensures that the amount of X-rays scattering from the sample or other equipment that reaches the detector is reduced, resulting in better contrast in the image. However, this type of collimator is not always implemented in a CT scanner.

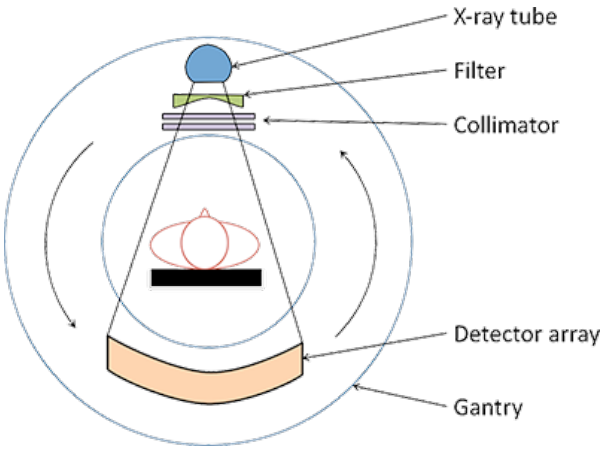


Figure 2.2: Front view schematic of the different parts of a CT scanner [25].

2.3.1. Attenuation Coefficient

The linear attenuation coefficient, denoted by μ and with SI unit cm^{-1}, is a material property that describes how X-rays are attenuated in a material. Consequently, CT scanners use this property to measure the internal structure of a sample. When an X-ray travels through a material, the intensity of that X-ray decreases depending on the composition of the material. X-rays travel much more easily through mediums such as water than through materials with higher density.

The attenuation coefficient is related to the probability per unit path length that an X-ray undergoes an interaction with the medium through which it travels [26]. The effect of X-ray absorption in a material was described by Johann Heinrich Lambert in 1760 in the following formula [27]:

$$I = I_0 \cdot e^{-\mu d} \,. \tag{2.14}$$

Here, I_0 is the initial intensity of the X-ray beam in units $\mathrm{J\,m^{-2}\,s^{-1}}$. The initial intensity decreases to I exponentially when travelling through an object with absorption coefficient μ and thickness d. The product μd is then the expected number of interactions along thickness d. However, this equation only holds when the X-rays that pass through the material are monochromatic, as higher-energy photons do not get attenuated at the same rate as lower-energy photons [28]. The equation that describes the decrease in X-ray intensity for polychromatic X-rays is given as follows [29]:

$$I(E) = I_0(E) e^{-\int \mu(E,x) dx} \,. \tag{2.15}$$

Here, the linear attenuation coefficient as a function of place and energy is given as $\mu(E, x)$. The integral in the exponent is a line integral that calculates the attenuation coefficient over the path of a specific X-ray. The value of this attenuation coefficient is therefore also dependent on the energy since we are now looking at the attenuation of polychromatic X-rays.

The mass attenuation coefficient is defined as the linear attenuation coefficient of a certain material divided by the density, denoted as μ/ρ and given in units $\mathrm{cm^2\,g^{-1}}$. Whereas the linear attenuation coefficient is density dependent, the mass attenuation coefficient is not. For example, the linear attenuation coefficient of ice is different from that of water, while the mass attenuation coefficient is equivalent for both states.

In the energy range of 20 - 150 keV, which is the broad range in which most CT scanners operate [30], two primary interaction effects govern the attenuation; Compton scattering and

the photoelectric effect [31]. From these interactions, higher-energy photons mostly undergo Compton scattering, while lower-energy photons have a higher probability of undergoing the photoelectric effect [32]. Since the linear attenuation coefficient is related to the probability of these interactions with the medium, it is possible to write it as a linear combination of the effect of Compton and photoelectric interactions [33]. This can be expressed in the following formula:

$$\mu(E) = a_p f_p(E) + a_c f_c(E) \,, \tag{2.16}$$

where $a_p > 0$ and $f_p(E)$ are the contribution and energy dependence function of the photoelectric effect, respectively, and $a_c > 0$ and $f_c(E)$ those of Compton scattering. Because these interactions are energy dependent, the attenuation coefficient of a material also varies with photon energy. As mentioned earlier, lower-energy photons have a much higher probability of interaction with the photoelectric effect compared to Compton scattering. Therefore, when an X-ray passes through a material with a high attenuation coefficient, this not only results in an X-ray beam with less intensity, but also with a higher average energy because lower-energy photons have a higher chance of being absorbed by the tissue [32]. This effect is also known as beam hardening and will be explained in more detail in Section 2.5.4.

The linear combination from Equation (2.16) makes it possible to approximate the linear attenuation coefficient of any material as a linear combination of a set of basis materials in the following way [33]:

$$\mu(E) = a_1 f_1(E) + a_2 f_2(E) + \cdots + a_n f_n(E) \,. \tag{2.17}$$

Here, a_i are constants that give the contribution of material i to the composition of the material, and $f_i(E)$ are the linear attenuation coefficients of material i as a function of energy. With this approximation, it is possible to assume that the attenuation coefficient of all tissues in the human body can be described as a linear combination of the attenuation coefficients of water and bone [34]. This linear combination will be applied in Section 2.5.1 in the joint reconstruction optimisation algorithm to reconstruct different biological materials within the body. Therefore, the attenuation coefficient of an arbitrary biological material can be approximated as follows:

$$\mu(E) \approx c_1 \cdot \mu_{\text{bone}}(E) + c_2 \cdot \mu_{\text{water}}(E) \,. \tag{2.18}$$

Here, the variables c_1 and c_2 are dimensionless relative volume fractions. These variables represent the weights of the attenuation coefficients of bone and water to describe the attenuation coefficient of any arbitrary biological material within the energy range of 20 - 150 keV.

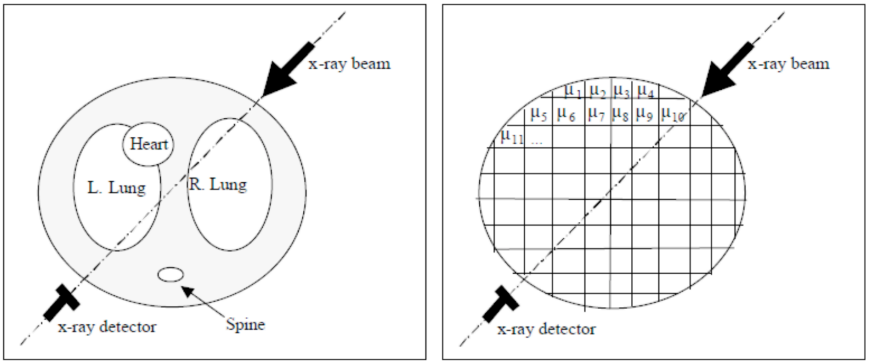
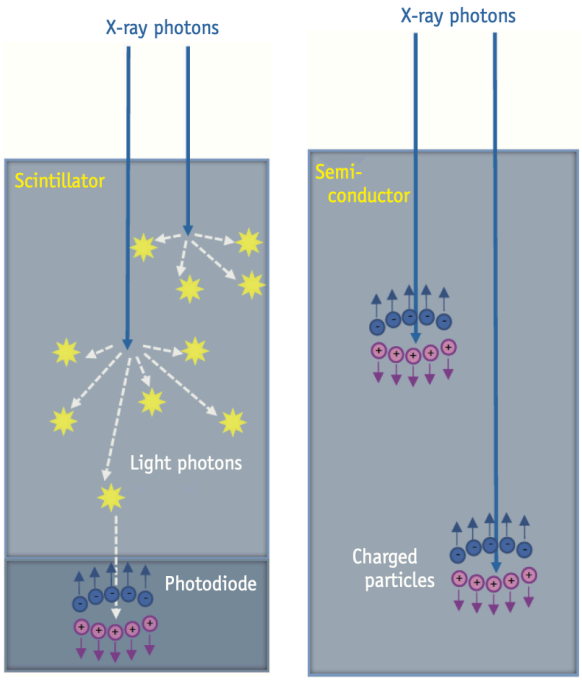


Figure 2.3: X-rays travelling through the body are detected and a material map of attenuation coefficients is made [35].

X-rays that travel through the body during a CT scan arrive at the detector with a decreased intensity compared to their initial intensity. After scanning the sample from many angles, the signals from the detector are transformed into a material map that consists of the attenuation coefficients of a single slice originating from the sample. To finally be able to form an entire reconstructed image, information from line integral data from each different angle is combined. This can be seen in Figure 2.3, where an X-ray beam passes through the body and is detected by the detector. After combining the line integral data from many different angles, an image is produced showing the attenuation coefficients along the line of travel. More details on the reconstruction of this material map are explained in Section 2.5.

2.3.2. X-ray Detectors

There are multiple ways to detect the X-rays that have passed through the patient. The most widely used X-ray detector is the energy integrating detector (EID) [36]. Depending on the type, a detector consists of hundreds of individual crystalline scintillators. Each scintillator is separated by a thin septum and connected to a photodiode below the scintillator [34]. One of these scintillator cells can be seen in Figure 2.4a. When X-ray photons enter the scintillator, they are absorbed by atoms, which then eject an electron [37]. When an electron is ejected in this manner, it is called a photoelectron. This photoelectron travels within the scintillator and unloads its energy onto the surrounding electrons in the material, after which they get excited. After some time, these electrons emit their energy in the form of visible or ultraviolet light, indicated as a yellow spark in Figure 2.4a. This scintillation light can travel relatively far within the scintillator and would spread into neighbouring detector cells if left unconfined. This optical crosstalk is prevented by the septa between the scintillator cells, effectively blocking the scintillation light and keeping it confined to the pixel where it was generated. The confined light is then detected by the photodiode, which transforms it into an electric signal proportional to the total energy of the absorbed X-ray photon. However, the signals sent by the photodiode



(a) Schematic of an EID. **(b)** Schematic of a PCD.

Figure 2.4: Schematics of the composition of an EID and a PCD [34].

do not correspond to individual X-rays. That is because the light of a specific electron from which an electric signal is created overlaps with all light in the direct vicinity of that electron, due to the very high photon flux onto the detector. Therefore, an EID essentially measures the total photon energy [8]. This means that high-energy photons contribute more to the electric signal than low-energy photons, resulting in a poor use of the information carried by low-energy photons [38]. That is one of the reasons it is desirable to know the individual photon energy of each incident X-ray.

A very new method of detecting X-rays is the use of a photon-counting detector (PCD). Instead of a crystalline scintillator, a PCD works with a semiconductor as a detector, typically cadmium telluride (CdTe). This semiconductor, shown in Figure 2.4b, converts the photons directly into an electric signal proportional to the magnitude of the X-ray energy, after which each signal can be recorded individually [8]. This happens by introducing a strong electric field in the semiconductor using a cathode and an anode. When an incident X-ray enters the PCD, a small electrical particle cloud is created, consisting of positive and negative particles, holes and electrons, respectively. The electrons are directly attracted to the cathode, creating an electrical signal [8]. When the individual photon energies are known, it is possible to sort every photon into discrete energy intervals, called energy bins. The way these energy bins are then reconstructed into an image is explained in detail in Section 2.5.1.
PCCT scanners have many advantages compared to CT scanners equipped with an EID. As follows from Figure 2.4 and the previous explanation of the functioning of both detectors, an EID creates an electric signal in two steps, while a PCD creates a signal in a single step. The conversion of X-rays into light in the scintillator of an EID gives the system more room to create signal noise. This is mainly due to the afterglow of the light emitted from the electrons in the scintillator. After an electron releases its energy as light, which is picked up by the photodiode to create a signal, an afterglow remains for a few milliseconds [8]. This afterglow causes a larger signal than is factually proportional to the energy of the X-ray, and thus heightens the thermal noise in the system. As the PCD directly converts the incident X-ray to a signal, this thermal noise is not present. In addition, PCCT scanners implement an energy threshold in their detection. Because the X-rays are measured independently, it is possible to filter out photon energies below a certain value, typically 20 keV [8]. This threshold is placed to reduce the electric noise significantly, as this tends to correspond to photon energies below the threshold level. However, such a threshold cannot be implemented for EIDs, as they indirectly measure the total photon energy. Therefore, it is not possible to filter out specific low-energy photons, which results in increased electric noise. Lastly, the septa between each scintillator element in an EID occupy a space in which no X-rays can be detected. This causes the dose efficiency to be lower, as there are photons whose energy will not be detected. The absence of these septa in a PCD results in an improved spatial resolution [8].

For the reconstruction algorithm, it is necessary to be able to calculate the amount of photons landing on the detector. Let Y denote the measured photon count, which can then be calculated by applying the polyenergetic version of Lambert-Beer's law to PCCT by introducing the effective spectrum:

$$Y = \int_0^\infty \Phi_{\text{eff}}(E)\, e^{-\int \mu(E,x)dx} dE\,. \tag{2.19}$$

The exponential in the formula describes the amount of attenuation of an X-ray according to Equation (2.14). The variable $\Phi_{\text{eff}}(E)$ is the energy dependent, effective X-ray spectrum, which is the photon count of the incident spectrum before it undergoes attenuation. This variable consists of three other variables, namely $\Phi_{\text{eff}}(E) = \Phi(E) \cdot S^s(E) \cdot D(E)$ [39]. Here, $\Phi(E)$ is

the X-ray spectrum that reaches the detector before the photons are classified into energy bins. The variable $S^s(E)$ is the response spectrum for energy bin s of the detector, included to adapt this formula to PCCT scanners. This spectrum defines how photons of energy E contribute to bin s. Furthermore, $D(E)$ is the detector efficiency, which reflects the chance that a photon of energy E is successfully registered as a count. The energy dependency originates from the fact that photons with different energies react differently with the detector. The SI unit of Φ_{eff} is J^{-1}, that is, photon count per unit energy. After multiplying by the unit J from dE, it becomes clear that Y is unitless. This matches the expectation, as it is an integer number.

Although scintillators are long believed to be too slow to detect individual photons and their energies, as a PCD does, the use of scintillators for PCCT is currently being investigated. Taguchi *et al.* [40] proved that the scintillator cerium-doped lanthanum bromide ($LaBr_3$:Ce) provided a comparable or better counting capability and superior material decomposition quality than the semiconductor CdTe in Monte Carlo simulations. The production of these scintillators is much cheaper than that of semiconductors such as CdTe. Therefore, these advancements in research are exciting developments towards a future where PCCT scanners are much more widely available.

2.3.3. Fan-beam CT vs. Cone-beam CT

In addition to the multiple possible X-ray detectors, there are also different methods by which the sample can be scanned. The most widely available CT scanning technique is fan-beam CT (FBCT) [41], which means that the polychromatic X-rays emitted from the X-ray tube are shaped in a thin fan [42]. After this X-ray fan passes through the sample, it is detected by a flat-line detector directly opposite the emitter, as can be seen in the left image of Figure 2.5. In the image, the red lines represent the paths of the emitted X-rays. To obtain a full image of the sample, the detector and emitter revolve around the sample inside the gantry. In this way, the sample is scanned from many different angles, typically at least one thousand [43], after which an image of a thin slice of the sample may be created. Finally, to scan the whole patient, the flat X-ray beam and detector spiral around the patient to obtain a three-dimensional reconstruction.

It is also possible to perform a CT scan in which the X-rays are emitted in a cone shape.

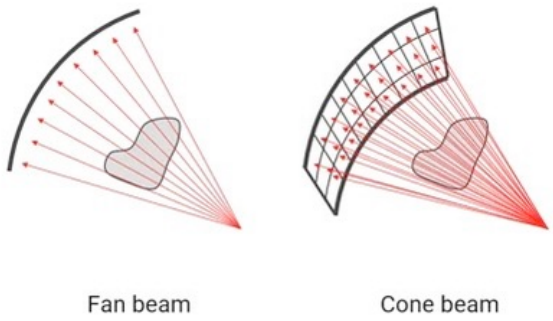


Figure 2.5: Fan-beam CT (left) vs. cone-beam CT (right) [44]. The X-ray, shown as red lines, pass through the sample to be received by their respective detectors.

This is called cone beam CT (CBCT) [42]. In the right image in Figure 2.5 it is clear that the X-rays, again displayed as red lines, are emitted in a cone shape, after which they land on a two-dimensional detector. A very important advantage of CBCT is that it is much easier to integrate in other types of setup. Since the FBCT spirals around the patient to obtain a reconstruction image, it takes up much more space than a CBCT that captures the entire

patient rotating around a smaller area. However, because of its cone-shaped beam, CBCT is also much more susceptible to scattering in the sample. Sample scatter that would not be detected by the flat beam of FBCT could be detected by CBCT because of its volumetric shape. This leads to lower contrast and a more influential presence of imaging artefacts, compared to FBCT [7].

2.3.4. Spectral CT

Conventional CT uses single-energy CT (SECT), which emits a single-energy spectrum by the X-ray tube. With SECT, the energy dependency of the attenuation coefficient is not taken into account since the technique only pinpoints a single attenuation coefficient value. Therefore, it omits the behaviour of the attenuation coefficient for different values of photon energy, resulting in reduced material distinction and increased noise [45]. Moreover, SECT can only reconstruct the average value of the linear attenuation coefficient at a certain energy. Therefore, using this SECT, it can be very difficult to distinguish similarly composed materials.
To gather more spectral information for the reconstruction, spectral CT is introduced. In conventional CT, spectral CT entails the use of more than one spectrum from the X-ray tube to adjust for the fact that the attenuation coefficient depends on photon energy. The most common implementation of spectral CT is called dual-energy CT (DECT), which uses energy spectra at two different energies to acquire data. Because of this, the energy dependency of the linear attenuation coefficient can be taken into account. These energy spectra are of lower energy and another of higher energy [32]. This makes the material distinction much clearer than using SECT. DECT can be carried out in a few different ways [8]. The first is performed by integrating two X-ray tubes into the gantry of a CT scanner that emit different spectra of low- and high-energy photons, which is called dual-source CT. Secondly, it is possible to have a single X-ray tube in the gantry, which oscillates rapidly between spectra of low- and high-energy photons. This method is called rapid kVp switching CT.
As explained in Section 2.3.2, PCDs can measure the individual energies of photons and sort them into energy bins. With these energy bins, PCCT scanners can also take spectral information into account with their detectors, even when using a single energy spectrum [8].

Introducing spectral CT, it is possible to determine two unknown quantities in the reconstruction [46]. This is, for example, very important in material decomposition algorithms, as the contribution of two different basis materials needs to be determined to create a material map, as explained in Section 2.3.1. Therefore, spectral CT can improve the material distinction between different materials, specifically soft tissues with very similar attenuation coefficients, providing better quality anatomical images and improving diagnostic capabilities [47]. There are more advantages and applications of spectral CT compared to conventional CT, such as low-dose CT, contrast imaging, or K-edge imaging [29]. However, in this thesis, the improvement of soft tissue contrast is the most important advantage.

2.4. Proton Therapy

External beam radiotherapy is used to combat cancer cells within the body by directing a beam of particles toward malignant cells with the aim of destroying them. Around 60%-70% of cancer patients receive some form of radiotherapy in the course of their treatment, the most used being photon therapy [48]. However, photon therapy has the disadvantage that it irradiates not only cancer cells, but also surrounding healthy tissue. This is because the X-rays used for photon therapy are able to penetrate completely through the body, losing energy according to the attenuation coefficient in the irradiated area. This is the same principle used for CT; however, the energy of the X-rays used in photon therapy is much higher than those used in CT. The photon energy used in photon therapy can be up to 18 MeV for deep

tissue cancer [3], compared to the range of 20 - 150 keV used during a CT scan. Since X-rays penetrate the entire body, the radiation dose is not localised in the object region, resulting in either a higher risk of adverse side effects or in a decrease in dose to spare healthy tissue.

A newer method within external beam radiotherapy is proton therapy. Instead of X-rays, a beam of protons is used to irradiate cancer cells. Protons travel only a finite range within the body, resulting in a very localised dose peak, as particles rapidly lose all their energy in the last millimetres before stopping [2]. This effect is known as the Bragg peak and can be seen in Figure 2.6. Due to this effect, the dose provided by the proton beam is much more localised than that of the photon beam, resulting in less damaged healthy tissue, which is of course beneficial to the patient.

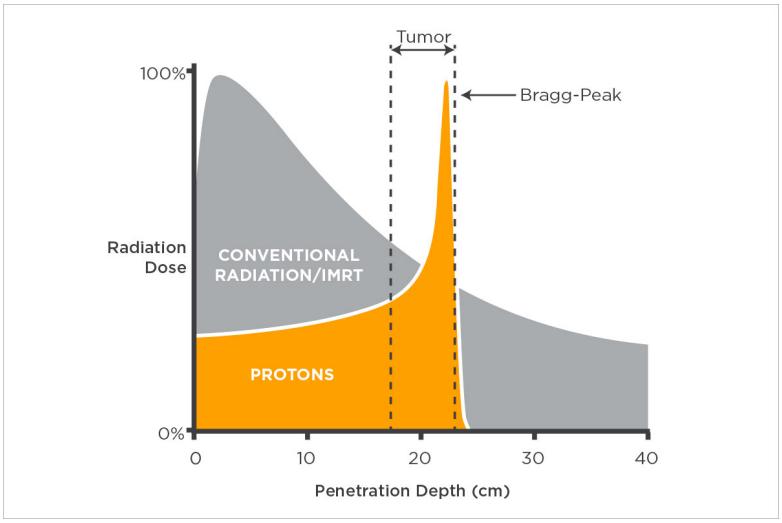


Figure 2.6: Visualisation of the radiation dose per penetration depth for protons and conventional radiation types, such as photons. The Bragg peak produces a very localised dose peak at the target tissue [49].

The proton stopping power, measured in units $\mathrm{J\,m^{-1}}$, is defined as the energy loss of a proton travelling through a material and is therefore mathematically written as the negative derivative of the energy. The complete expression for calculating the proton stopping power S for a proton with a certain energy E_p is given by Bethe's equation [50]:

$$S(E_p) = -\frac{dE_p}{dx} = \rho_e \frac{k_1}{\beta^2}\left[\frac{1}{2}\ln\left(\frac{k_2\beta^2 T_{\max}}{I_m^2(1-\beta^2)}\right) - \beta^2\right]. \tag{2.20}$$

Here, k_1 and k_2 are products of physical constants, $\beta = v_p/c$ is the speed of the proton relative to the speed of light, and $T_{\max}$ is the maximum energy transferred from the proton to an electron inside the material. Moreover, the variable ρ_e is the electron density of the medium and I_m is the mean excitation energy. The mean excitation energy is calculated using a version of Bragg's additivity rule [4]:

$$\ln I_m = \left(\sum_i \frac{\omega_i\ Z_i}{A_i}\ln I_i\right)\left(\sum_i \frac{\omega_i\ Z_i}{A_i}\right)^{-1}. \tag{2.21}$$

However, before this equation can be used, the material composition of material m is determined. This consists of a certain number of elements i. With this material composition, the intensity of a specific material I_m can be calculated. The variable ω_i is the weight fraction of element i in material m. Since the material composition is assumed to be known, the corresponding weight fractions can be directly determined from that composition. Furthermore,

the variable Z_i is the atomic number of element i, A_i is the atomic weight of element i, and I_i is the excitation energy of element i.

The stopping power ratio (SPR) is defined as the stopping power of a material relative to the stopping power of water and is therefore a unitless quantity. Using Bethe's equation from Formula (2.20), we obtain the following expression for the SPR:

$$\mathrm{SPR} = \frac{\rho_{\mathrm{e,\ mat}}}{\rho_{\mathrm{e,\ water}}} \cdot \frac{\ln\left(\frac{2m_e c^2\beta^2}{I_{\mathrm{mat}}\cdot(1-\beta^2)}\right) - \beta^2}{\ln\left(\frac{2m_e c^2\beta^2}{I_{\mathrm{water}}\cdot(1-\beta^2)}\right) - \beta^2}\,. \tag{2.22}$$

This equation is used to determine the required proton dose distribution for proton therapy [3]. This is done by determining the water equivalent path length [51], which is used to determine the entire treatment plan needed to irradiate specific tissues within the body. This plan includes the angles in which the tissue should be irradiated, or the intensity of the proton beam, to name a few.

As mentioned in the Introduction, the integration of a CBCT scanner with a proton therapy machine is a very interesting topic in CT applications. As proton therapy is a very directed and specific type of radiation therapy, it is very important to know exactly where the beam has to irradiate the tissue. In practice, SPR maps are determined before treatment to construct a specific treatment plan for the patient. However, since FBCT machines are typically large due to the spiralling required to capture specific areas of the body, it is not possible to integrate these types of scanners with radiotherapy units to provide immediate updates to the treatment plan according to the changing anatomy of the body. Therefore, it would be desirable to integrate the more spatially efficient CBCT scanners into radiation therapy units to be able to update the treatment plan directly.

2.5. Basics of Image Reconstruction

In the context of X-ray computed tomography reconstruction, an inverse problem considers the reconstruction of the inner structure of a patient from X-ray projection images [13]. This is opposed to the related direct problem, which considers the exact opposite problem of reconstructing the measured data, in the form of a projection, out of a known internal structure. From the data gathered during a CT scan, a reconstruction is required to draw any important conclusions about the patient's health. It is obvious that the inner structure of a patient is not a discrete map of attenuation coefficients; however, it is treated in this way in the inverse problem to acquire an estimate of the mapping of the attenuation coefficient within the body, which would be $\mathbf{u}$ as given in the perturbed measurement model (2.1). In conventional CT, this inverse problem is often resolved using a filtered back projection (FBP) algorithm [52]. This algorithm applies a high-pass convolution filter to the measured data before projecting them to form a reconstruction image. As this algorithm is not applicable to joint reconstruction algorithms, it will not be explained in detail. The most prominent way to approach this problem from the raw data acquired in a PCCT scanner is to sort every photon energy into discrete energy intervals, called energy bins.

2.5.1. The Joint Reconstruction Algorithm

As mentioned in Section 2.3.2, a PCD categorises photons into discrete energy bins to prepare for reconstruction. The information collected by the PCCT in the form of energy bins is then used to create a reconstruction of each bin individually. The result is a number of reconstructions equal to the number of bins, with each reconstruction pertaining to a certain photon energy range. To obtain a final reconstruction, these separate images are combined into one.

However, when reconstructing each energy bin separately, the photon count per reconstruction image will be smaller, resulting in images with inferior image quality [9]. Therefore, this method introduces a lot of noise into the system, partly due to the inferior photon count per reconstruction, but also because the separate reconstruction ignores relations between the energy bins. Joint reconstruction combines the information from all energy bins into a single inverse problem with the goal of directly obtaining a final reconstruction image. Joint reconstruction has only been applied to FBCT in theory, but not yet in practice. It has been proven that this technique results in higher image qualities [7], therefore, it is interesting to consider whether this will be the same for CBCT.

In the upcoming section, the optimisation algorithm used in this thesis is explained in detail. This algorithm is a joint reconstruction decomposition and reconstruction algorithm for PCCT, provided by Mechlem *et al.* [10]. The initial algorithm was built for spectral FBCT. It was implemented by Mory *et al.* [11] in `MATLAB`, after which it was adapted to CBCT by van Leenen [12].

When the approximation from Equation (2.17) is made with the set of basis materials, here taken as bone and water, it is possible to write the attenuation coefficient as follows:

$$\int \mu_i(E, x)dx = \sum_{b=1}^{2} A_i^b f^b(E) \, . \tag{2.23}$$

This equation follows the same principles of Equation (2.17); however, instead of calculating the attenuation coefficient in a single voxel, the path-integrated attenuation is taken along a certain path between the source and a detector pixel $i = \{1, 2, \ldots, P\}$. Since the variable i describes a certain path between the source and a detector pixel, this variable consists of a combination between a certain projection angle and a certain detector pixel. Due to the line integral over the attenuation coefficient, the expression on the left-hand side is unitless. Furthermore, A_i^b is the line integral of the density of basis material b along a certain X-ray path i, and therefore describes the amount of basis material b present along that path. The unit of the line integral A_i^b is $\mathrm{g\,cm^{-2}}$. Lastly, the variable $f^b(E)$ gives the values of the mass attenuation coefficient μ/ρ of each basis material b at certain energies E, given in units of $\mathrm{cm^2\,g^{-1}}$. Therefore, this is the value of the attenuation coefficient for a specific material, in this case bone and water, over the range of energies 20 - 150 keV. In cases where a contrast agent is introduced in the patient's body to highlight specific parts in the final reconstructed image, the optimisation algorithm can be extended to include a third material. However, this extension is not considered in this thesis.
As we are looking to reconstruct a certain measured projection on a discrete voxel grid, we must discretise the line integral over the density A_i^b in Equation (2.23) to obtain the following expression for the attenuation coefficient:

$$\int \mu_i(E, x)dx = \sum_{b=1}^{2} f^b(E) \sum_{j=1}^{N} a_{ij}\alpha_j^b \, . \tag{2.24}$$

The variable A^b is discretised by summing the product of the variables a_{ij} and α_j^b over all voxels $j = \{1, 2, \ldots, N\}$ in the reconstruction. The variable a_{ij} represents the elements of the forward projection matrix $\mathbf{A} \in \mathbb{R}^{P \times N}$ at a certain detector pixel and projection angle i. In this context, the variable gives the distance an X-ray travels through voxel j to along a certain path i. It is given in unit cm. This variable also acts as the forward projection matrix in the perturbed measurement model from Equation (2.1). The variable α_j^b is the contribution of basis material b to the density in voxel j along the X-ray path, in units of $\mathrm{g\,cm^{-3}}$. This variable

essentially contains the information needed to acquire an estimate of the reconstruction image. The contribution of each of the basis materials to the biological material that exists in voxel j can be used to determine the attenuation coefficient in that voxel. Dividing this variable by the density of the respective basis material gives us the relative volume fractions given in Equation (2.18). Looking back at Equation (2.1), we can say that this is the variable $\mathbf{u}$ in this problem and therefore α_j^b is the variable optimised in this algorithm.

To be able to determine the expected photon counts $\hat{Y}_i$ that land on a specific detector pixel i, Equations (2.19) and (2.24) are combined to generate the following expression:

$$\hat{Y}_i = \int_0^\infty \Phi_{\text{eff},i}(E)\, e^{\sum_{b=1}^{2} f^b(E) \sum_{j=1}^{N} a_{ij}\alpha_j^b} dE\,. \tag{2.25}$$

The variable $\Phi_{\text{eff},i}(E)$ is the energy dependent, effective X-ray spectrum for a specific ray i that originates from the source and lands on a detector pixel, and more details on this variable were explained in Section 2.3.2. A simulated spectrum and detector response are used in our implementation of the algorithm, and therefore the variable $\Phi_{\text{eff},i}(E)$ is known.

As discussed in Section 2.3.2, PCCT scanners minimise the influence of thermal and other electronic noise by implementing energy thresholds. Because the influence of external noise is minimised, reconstruction is assumed to be limited solely by the noise exhibited by X-ray detectors [53]. As incident X-rays are detected by the PCD in a discrete manner, the Poisson distribution can be introduced to simulate the noise [50]. This is because Poisson statistics model the probabilities of discrete, independent events occurring in a fixed interval. This is also the case for detecting photons in photon-counting CT scanners. Therefore, we can conclude that $Y_i \sim \text{Poisson}(\hat{Y}_i)$, where Y_i is the measured photon count and $\hat{Y}_i$ the expected photon count on a certain detector pixel i from a certain angle. Following the equations explained in Section 2.1, we can obtain the following likelihood function:

$$\mathcal{P}\left(\mathbf{Y}|\hat{\mathbf{Y}}(\vec{\alpha})\right) = \prod_{i=1}^{P} \frac{\hat{Y}_i(\vec{\alpha})^{Y_i}}{Y_i!} e^{-\hat{Y}_i(\vec{\alpha})}\,. \tag{2.26}$$

The measurements $\mathbf{m}$ from Equation (2.1) are the photon counts measured by each path i from the source to a detector pixel. Moreover, the expected photon counts depend on the optimisation variable $\vec{\alpha} = (\alpha_1^1, \ldots, \alpha_N^1, \alpha_1^2, \ldots, \alpha_N^2)^T$. From the likelihood function, the negative log-likelihood function is calculated such that the optimisation parameter $\vec{\alpha}$ can be minimised:

$$-\mathcal{L}(\vec{\alpha}) = \sum_{i=1}^{P} \hat{Y}_i - Y_i \cdot \log(\hat{Y}_i) = \sum_{i=1}^{P} h_i(\hat{Y}_i(\vec{\alpha}))\,, \tag{2.27}$$

Here, we used the abbreviation $h_i(\hat{Y}_i(\vec{\alpha}))$, so that further calculations can be presented more clearly. Moreover, due to the discrete nature of the voxel grid, the expected photon counts $\hat{Y}_i(\vec{\alpha})$ are described by the discretised version of the polychromatic Lambert-Beer's law:

$$\hat{Y}_i(\vec{\alpha}) = \sum_{s=1}^{S} N_i^s \exp\left[-\sum_{b=1}^{B} f^b(E^s) \sum_{j=1}^{N} a_{ij}\alpha_j^b\right]\,. \tag{2.28}$$

To discretise this equation, the integral over energy is written as a sum over all energy bins $s = 1, \ldots, S$. Furthermore, the variable N_i^s is introduced to represent the amount of photons detected by detector pixel and projection angle i in energy bin s if there is no object present in the scanner.

The negative log-likelihood does not give an accurate minimisation of $\vec{\alpha}$ on its own, as this optimisation problem is ill-conditioned. As explained in Section 2.2, to achieve stability in the problem and obtain a solution to this optimisation, we introduce the regularisation term $\mathcal{R}(\vec{\alpha})$. By including this function in the cost function, additional noise is suppressed by incorporating prior knowledge about the attenuation coefficients of the materials in the object. The regularisation function used in this paper is taken from Mechlem *et al.* [10]:

$$\mathcal{R}(\vec{\alpha}) = \sum_{b=1}^{2} \sum_{j=1}^{N} \sum_{k \in N_j} \lambda_b \phi^{\text{Huber}}(\alpha_j^b - \alpha_k^b, \gamma_b) \,, \tag{2.29}$$

$$\phi^{\text{Huber}}(\Delta, \gamma_b) = \begin{cases} \Delta^2 & \text{if } \ |\Delta| < \gamma_b \\ 2\gamma_b|\Delta| - \gamma_b^2 & \text{if } \ |\Delta| \geq \gamma_b \ . \end{cases} \tag{2.30}$$

In the regularisation term, N_j represents a geometric neighbourhood of voxel j, which in this reconstruction consists of the 26 immediate neighbours of voxel k. Furthermore, the variable λ_b is the tuning parameter for basis material b, which determines the influence of regularisation on the reconstruction of basis material b. Furthermore, the regularisation term contains the twice differentiable Huber loss function, which is shown in Equation (2.30), and is a combination of a Tikhonov and a TV regularisation term, as explained in Section 2.2. In this formula, Δ is the difference between the value of α_j and its neighbouring voxels $k \in N_j$ and γ_b is a tuning parameter for basis material b. The ℓ^2 based part of the penalty function is represented by a quadratic function, which applies to $|\Delta| < \gamma_b$. This is done to ensure smoothness in areas of reconstruction in which there are small voxel differences and to balance out small changes. However, when the difference between two voxel values is greater than a predetermined parameter γ_b, an ℓ^1 penalty is introduced. This is to ensure that larger differences between neighbouring voxels, which are more indicative of image structures and boundaries, are penalised using a linear function to preserve sharp edges without excessive smoothing. The function given in Equation (2.29) was specifically chosen in this optimisation algorithm to provide *a priori* knowledge adjusted to reflect the characteristics of the attenuation coefficient within the human body. Inside the body, the attenuation coefficient is often subject to change. However, very small changes have a high probability of being noise, as the data is most likely gathered from the same tissue. For larger voxel differences, a change in tissue or an abnormality within the body is likely, and is therefore important to preserve in the reconstructed image.
Combining the negative log-likelihood and the regularisation term, the final optimised value of $\vec{\alpha}$ is as follows:

$$\begin{aligned} \vec{\alpha}^{(\text{opt})} &= \arg\min_{\vec{\alpha}} \theta(\vec{\alpha}) = \arg\min_{\vec{\alpha}} \left(-\mathcal{L}(\vec{\alpha}) + \mathcal{R}(\vec{\alpha})\right) \\ &= \arg\min_{\vec{\alpha}} \left(\sum_{i=1}^{P} h_i(\hat{Y}_i(\vec{\alpha})) \ + \ \sum_{b=1}^{2} \sum_{j=1}^{N} \sum_{k \in N_j} \lambda_b \phi^{\text{Huber}}(\alpha_j^b - \alpha_k^b, \gamma_b) \right) , \end{aligned} \tag{2.31}$$

where the cost function, that is, $\theta(\vec{\alpha}) = -\mathcal{L}(\vec{\alpha}) + \mathcal{R}(\vec{\alpha})$, is defined as the function to be minimised.

Instead of directly minimising the cost function in the optimisation algorithm, quadratic surrogate functions $Q(\vec{\alpha};\vec{\alpha}^{(n)})$ of the cost function $\theta(\vec{\alpha})$ are used. These accelerate the convergence of the optimisation algorithm and reduce the reconstruction time of the image [10]. Surrogate functions are derived separately for the negative log-likelihood and regularisation term in each iteration step n. The functions are manipulated to be quadratic in the variable $\vec{\alpha}^{(n)}$ and separable for each voxel. Then, the objective function can be optimised analytically much more easily and can be optimised in parallel for different voxels. We denote the surrogate function of the log-likelihood estimator by $Q_{\mathcal{L}}(\vec{\alpha};\vec{\alpha}^{(n)})$ and of the regularisation term by $Q_{\mathcal{R}}(\vec{\alpha};\vec{\alpha}^{(n)})$. Similarly, the log-likelihood segment of the cost function is denoted as $\theta_{\mathcal{L}}(\vec{\alpha})$ and that of the regularisation $\theta_{\mathcal{R}}(\vec{\alpha})$. If the surrogate functions satisfy the following restrictions, minimising the surrogate function is equivalent to minimising the cost function $\theta(\vec{\alpha})$:

$$Q(\vec{\alpha}^{(n)};\vec{\alpha}^{(n)}) = \theta(\vec{\alpha}^{(n)}) \tag{2.32a}$$

$$\left.\frac{\partial Q(\vec{\alpha};\vec{\alpha}^{(n)})}{\partial \alpha_j^b}\right|_{\vec{\alpha}=\vec{\alpha}^{(n)}} = \left.\frac{\partial \theta(\vec{\alpha})}{\partial \alpha_j^b}\right|_{\vec{\alpha}=\vec{\alpha}^{(n)}} \tag{2.32b}$$

$$Q(\vec{\alpha};\vec{\alpha}^{(n)}) \geq \theta(\vec{\alpha})\,. \tag{2.32c}$$

These conditions ensure that the cost function and the calculated surrogate function have the same position and tangent at the desired optimisation point $\vec{\alpha}^{(n)}$ for iteration step n, meaning that at that point both functions are approximately equal.

In the following part, the surrogate function $Q_{\mathcal{L}}(\vec{\alpha};\vec{\alpha}^{(n)})$ is determined for the log-likelihood $-\mathcal{L}(\vec{\alpha})$. The following abbreviations are used to simplify the mathematical notation:

$$\begin{aligned} -\mathcal{L}(\vec{\alpha}) &= \sum_{i=1}^{P} h_i(\hat{Y}_i), \quad h_i(\hat{Y}_i) = \hat{Y}_i \;-\; Y_i \log(\hat{Y}_i) \\ l_i^{sb}(\vec{\alpha}^b) &= \sum_{j=1}^{N} a_{ij}\alpha_j^b f^b(E^s), \quad l_i^s(\vec{\alpha}^b) = \sum_{b=1}^{2} l_i^{sb}(\vec{\alpha}^b) \\ t_i^s(\vec{\alpha}) &= e^{-l_i^s(\vec{\alpha})}, \quad \beta_i^{s,(n)} = \frac{\hat{Y}_i(\vec{\alpha}^{(n)})}{t_i^s(\vec{\alpha}^{(n)})}\,. \end{aligned} \tag{2.33}$$

First, we rewrite the forward model in Equation (2.28) with the abbreviations from Equation (2.33), to obtain the following equation:

$$\hat{Y}_i = \sum_{s=1}^{S} N_i^s t_i^s(\vec{\alpha}) = \sum_{s=1}^{S} \frac{N_i^s}{\beta_i^{s,(n)}} t_i^s(\vec{\alpha})\beta_i^{s,(n)}\,. \tag{2.34}$$

The term $\beta_i^{s,(n)}$ was added so that the following simplification can be performed:

$$\sum_{s=1}^{S} \frac{N_i^s}{\beta_i^{s,(n)}} = \frac{1}{\hat{Y}_i(\vec{\alpha}^{(n)})} \sum_{s=1}^{S} N_i^s * e^{-l_i^s(\vec{\alpha}^{(n)})} = \frac{1}{\hat{Y}_i(\vec{\alpha}^{(n)})} \cdot \hat{Y}_i(\vec{\alpha}^{(n)}) = 1\,, \tag{2.35}$$

where we used the abbreviation of $\beta_i^{s,(n)}$ found in Equation (2.33) and the expression in Equation (2.28) to obtain the result. Then, by the result from Equation (2.35), Jensen's inequality

can be applied to the function $h_i(\hat{Y}_i)$, with $\hat{Y}_i$ equal to the expression in Equation (2.34). The conditions for Jensen's inequality can be found in Appendix A.1. The condition that $\sum_{i=1}^{n} p_i = 1$ is satisfied by Equation (2.35). Moreover, we conclude that

$$\begin{aligned} \hat{Y}_i(\vec{\alpha}^{(n)}) \geq N_i^s e^{-l_i^s(\vec{\alpha}^{(n)})} &\implies \frac{N_i^s}{\hat{Y}_i(\vec{\alpha}^{(n)})} * e^{-l_i^s(\vec{\alpha}^{(n)})} \leq 1 \\ &\implies \frac{N_i^s}{\beta_i^{s,(n)}} \in [0,1] \text{ for } s = 1, \ldots, S \ . \end{aligned} \tag{2.36}$$

Here, $\hat{Y}_i(\vec{\alpha}^{(n)}) \geq N_i^s e^{-l_i^s(\vec{\alpha}^{(n)})}$ is used because of the absence of the summation over the variable s in Equation (2.28). Only one condition is left to prove to be able to use Jensen's inequality, namely that $h_i(\hat{Y}_i)$ is a convex function. This is verified as follows:

$$\frac{\partial^2 h_i(\hat{Y}_i)}{\partial(\hat{Y}_i)^2} = \frac{Y_i}{(\hat{Y}_i)^2} > 0 \,, \tag{2.37}$$

as Y_i, $(\hat{Y}_i)^2 > 0$. Now that we know that the function $h_i(\hat{Y}_i)$ is convex, Jensen's inequality can be used. The equation that follows from the application of Jensen's inequality to $h_i(\hat{Y}_i)$ is defined as the first surrogate function:

$$\begin{aligned} -\mathcal{L}(\vec{\alpha}) = \sum_{i=1}^{P} h_i(\hat{Y}_i) &= \sum_{i=1}^{P} h_i \left(\sum_{s=1}^{S} \frac{N_i^s}{\beta_i^{s,(n)}} t_i^s(\vec{\alpha}) \beta_i^{s,(n)} \right) \\ &\leq \sum_{i=1}^{P} \sum_{s=1}^{S} \frac{N_i^s}{\beta_i^{s,(n)}} h_i(t_i^s(\vec{\alpha}) \beta_i^{s,(n)}) \equiv Q_1(\vec{\alpha}; \vec{\alpha}^{(n)}) \,. \end{aligned} \tag{2.38}$$

In Appendix A.2 a proof is given that conditions (2.32a) - (2.32c) hold for surrogate function $Q_1(\vec{\alpha}, \vec{\alpha}^{(n)})$. The first surrogate function is not yet quadratic with respect to the line integral $l_i^{sb}(\vec{\alpha}^{b,(n)})$, and therefore not in the variable $\vec{\alpha}^{(n)}$. This means that it does not meet the prerequisites given earlier to accelerate the optimisation process. Since we have moved the summation over s out of the function $h_i(\hat{Y}_i)$, it is possible to make it a quadratic function. Therefore, another surrogate function is derived from $Q_1(\vec{\alpha}, \vec{\alpha}^{(n)})$ using a second-order Taylor expansion of $g_i^{(n)}(l_i^{sb}) = h_i(t_i^s(\vec{\alpha}) \beta_i^{s,(n)})$ around the line integral $l_i^{sb}(\vec{\alpha}^{b,(n)})$:

$$\begin{aligned} g_i^{(n)}(l_i^{s1}, l_i^{s2}) \approx q_i^{(n)}(l_i^{s1}, l_i^{s2}) = g_i^{(n)}(l_i^{s1,(n)}, l_i^{s2,(n)}) &+ \sum_{b=1}^{2} \left. \frac{\partial g_i^{(n)}(l_i^{s1}, l_i^{s2})}{\partial l_i^{sb}} \right|_{l_i^{sb} = l_i^{sb,(n)}} \left(l_i^{sb} - l_i^{sb,(n)} \right) \\ &+ \frac{1}{2} \sum_{k=1}^{2} \sum_{m=1}^{2} T_i^{s,(n)} \left(l_i^{sk} - l_i^{sk,(n)} \right) \left(l_i^{sm} - l_i^{sm,(n)} \right) , \end{aligned} \tag{2.39}$$

where $l_i^{sb,(n)}$ is an abbreviation of $l_i^{sb}(\vec{\alpha}^{(n)})$. Furthermore, the variable $T_i^{s,(n)}$ is a variable introduced to approximate the second derivative of the variable $g_i^{(n)}(l_i^{sb})$. After combining Equations (2.38) and (2.39), the following expression is defined for the second surrogate function:

$$Q_2(\vec{\alpha}, \vec{\alpha}^{(n)}) = \sum_{i=1}^{P} \sum_{s=1}^{S} \frac{N_i^s}{\beta_i^{s,(n)}} q_i^{(n)}(l_i^{s1}, l_i^{s2}) \ , \tag{2.40}$$

with $q_i^{(n)}(l_i^{s1}, l_i^{s2})$ as given in formula (2.39). Similarly to $Q_1(\vec{\alpha}, \vec{\alpha}^{(n)})$, conditions (2.32a) - (2.32c) can be shown to hold for $Q_2(\vec{\alpha}, \vec{\alpha}^{(n)})$ and are presented in Appendix A.2. We have now obtained a surrogate function that is quadratic with respect to $l_i^{sb}(\vec{\alpha}^{b,(n)})$. However, this surrogate function still does not suffice for our analytical calculations. The desired surrogate function is separable with respect to the image voxels j, which $Q_2(\vec{\alpha}, \vec{\alpha}^{(n)})$ is not. To derive this desired function, we rewrite the line integral as follows:

$$\begin{aligned} l_i^{sb}(\vec{\alpha}^b) = \sum_{j=1}^{N} a_{ij}\alpha_j^b f^b(E^s) &= \sum_{j=1}^{N} w_{ij}\left(\frac{a_{ij}f^b(E^s)}{w_{ij}}(\alpha_j^b - \alpha_j^{b,(n)}) \;+\; l_i^{sb,(n)}\right) \\ &\equiv \sum_{j=1}^{N} w_{ij}\zeta_{ij}^{sb}(\alpha_j^b - \alpha_j^{b,(n)}) \,, \end{aligned} \tag{2.41}$$

with:

$$w_{ij} = \frac{a_{ij}}{\sum_{j=1}^{N} a_{ij}} \in [0,1], \qquad \sum_{j=1}^{N} w_{ij} = 1 \,. \tag{2.42}$$

After substituting this expression for the line integral $l_i^{sb}(\vec{\alpha}^b)$ into the second quadratic function, Jensen's inequality can be used. The use of Jensen's inequality is justified by conditions (2.42) and the fact that $q_i^{(n)}(l_i^{s1}, l_i^{s2})$ is a positive quadratic function and therefore convex. The third and final surrogate function is defined as follows:

$$\begin{aligned} Q_2(\vec{\alpha}, \vec{\alpha}^{(n)}) &= \sum_{i=1}^{P}\sum_{s=1}^{S} \frac{N_i^s}{\beta_i^{s,(n)}} q_i^{(n)}(l_i^{s1}, l_i^{s2}) \\ &= \sum_{i=1}^{P}\sum_{s=1}^{S} \frac{N_i^s}{\beta_i^{s,(n)}} q_i^{(n)}\Big(\sum_{j=1}^{N} w_{ij}\zeta_{ij}^{s1}(\alpha_j^1 - \alpha_j^{1,(n)}), \sum_{j=1}^{N} w_{ij}\zeta_{ij}^{s2}(\alpha_j^2 - \alpha_j^{2,(n)})\Big) \\ &\leq \sum_{i=1}^{P}\sum_{s=1}^{S}\sum_{j=1}^{N} w_{ij}\frac{N_i^s}{\beta_i^{s,(n)}} q_i^{(n)}\Big(\zeta_{ij}^{s1}(\alpha_j^1 - \alpha_j^{1,(n)}), \zeta_{ij}^{s2}(\alpha_j^2 - \alpha_j^{2,(n)})\Big) \\ &\equiv Q_{\mathcal{L}}(\vec{\alpha}, \vec{\alpha}^{(n)}) \,. \end{aligned} \tag{2.43}$$

This surrogate function now consists of a sum over all voxels N, and therefore satisfies the given prerequisite of separability.

The surrogate function for the regularisation term $\theta_{\mathcal{R}}(\vec{\alpha})$, as given in Equation (2.29), is already separable in the voxels j, which satisfies one of the conditions for a surrogate. Now, the function only needs to be quadratic in $\vec{\alpha}^{(n)}$. Although $\phi^{\text{Huber}}(\alpha_j^b - \alpha_k^b, \gamma_b)$ is quadratic in the region $|\Delta| < \gamma_b$, this does not mean that it is quadratic for $\vec{\alpha}^{(n)}$. To obtain this, we rewrite $\phi^{\text{Huber}}(\alpha_j^b - \alpha_k^b, \gamma_b)$ as follows [54]:

$$\begin{aligned} \phi^{\text{Huber}}(\alpha_j^b - \alpha_k^b, \gamma_b) &= \phi^{\text{Huber}}\Big[\frac{1}{2}\big(2\alpha_j^b - \alpha_j^{b,(n)} - \alpha_k^{b,(n)}\big) + \frac{1}{2}\big(-2\alpha_k^b + \alpha_j^{b,(n)} + \alpha_j^{b,(n)}\big)\Big] \\ &\leq \frac{1}{2}\phi^{\text{Huber}}\big(2\alpha_j^b - \alpha_j^{b,(n)} - \alpha_k^{b,(n)}\big) + \frac{1}{2}\phi^{\text{Huber}}\big(2\alpha_k^b - \alpha_j^{b,(n)} - \alpha_k^{b,(n)}\big) \,. \end{aligned} \tag{2.44}$$

The inequality arises from the use of Jensen's inequality, as ϕ^{Huber} is a convex function and because the sum of the weights of the rewritten Huber function is $\frac{1}{2} + \frac{1}{2} = 1$. The function is now quadratic in $\vec{\alpha}^{(n)}$ because, assuming that the difference $\alpha_j^{b,(n)} - \alpha_k^{b,(n)}$ is small, inserting $\vec{\alpha}^{(n)}$ in the function provides a value very close to zero. In this region, the function is quadratic by definition. After substituting this formulation of the Huber function, we obtain the surrogate function for the regularisation term, which satisfies the conditions:

$$\begin{aligned}
\mathcal{R}(\vec{\alpha}) &= \sum_{b=1}^{2} \sum_{j=1}^{N} \sum_{k \in N_j} \lambda_b \phi^{\text{Huber}}(\alpha_j^b - \alpha_k^b, \gamma_b) \\
&= \sum_{b=1}^{2} \sum_{j=1}^{N} \sum_{k \in N_j} \lambda_b \Big[\frac{1}{2} \phi^{\text{Huber}}(2\alpha_j^b - \alpha_j^{b,(n)} - \alpha_k^{b,(n)}, \gamma_b) + \frac{1}{2} \phi^{\text{Huber}}(2\alpha_k^b - \alpha_j^{b,(n)} - \alpha_k^{b,(n)}, \gamma_b) \Big] \\
&\equiv Q_{\mathcal{R}}(\vec{\alpha}; \vec{\alpha}^{(n)})
\end{aligned} \tag{2.45}$$

In Appendix A.3 it is shown that this function abides by conditions (2.32a) - (2.32c) for surrogate functions.

The final quadratic surrogate function $Q(\vec{\alpha}; \vec{\alpha}^{(n)})$ that is used to analytically calculate the optimisation algorithm then consists of the derived surrogate functions of $Q_{\mathcal{L}}(\vec{\alpha}; \vec{\alpha}^{(n)})$ and $Q_{\mathcal{R}}(\vec{\alpha}; \vec{\alpha}^{(n)})$ as follows:

$$Q(\vec{\alpha}; \vec{\alpha}^{(n)}) = Q_{\mathcal{L}}(\vec{\alpha}; \vec{\alpha}^{(n)}) + Q_{\mathcal{R}}(\vec{\alpha}; \vec{\alpha}^{(n)}) \,. \tag{2.46}$$

As both surrogate functions satisfy conditions (2.32a) - (2.32c) separately, so does $Q(\vec{\alpha}; \vec{\alpha}^{(n)})$ due to linearity.

After having calculated the surrogate functions with which the cost function can be solved analytically, the final algorithm is used to calculate the variable $\vec{\alpha}$:

$$\vec{\alpha}^{(n+1)} = \vec{\alpha}^{(n)} - \big(H_Q^{(n)}\big)^{-1} \cdot \nabla Q(\vec{\alpha}; \vec{\alpha}^{(n)}) \Big|_{\vec{\alpha} = \vec{\alpha}^{(n)}} \,, \tag{2.47}$$

with

$$\big(H_Q^{(n)}\big)^{-1} = -\big(H_{Q_L}^{(n)} + H_{Q_{\mathcal{R}}}^{(n)}\big)^{-1} \,, \tag{2.48}$$

$$\nabla Q(\vec{\alpha}; \vec{\alpha}^{(n)}) = \nabla \big(Q_{\mathcal{L}}(\vec{\alpha}; \vec{\alpha}^{(n)}) + Q_{\mathcal{R}}(\vec{\alpha}; \vec{\alpha}^{(n)}) \big) = \nabla \theta(\vec{\alpha}^{(n)}) \,, \tag{2.49}$$

where $H_Q^{(n)}$, $H_{Q_L}^{(n)}$ and $H_{Q_{\mathcal{R}}}^{(n)}$ are the Hessian matrices of $Q(\vec{\alpha}; \vec{\alpha}^{(n)})$, $Q_{\mathcal{L}}(\vec{\alpha}; \vec{\alpha}^{(n)})$ and $Q_{\mathcal{R}}(\vec{\alpha}; \vec{\alpha}^{(n)})$, respectively.

With this final iterative algorithm, the values of $\vec{\alpha} = (\alpha_1^1, \ldots, \alpha_N^1, \alpha_1^2, \ldots, \alpha_N^2)^T$ can be calculated within the reconstruction image. This results in a material map per basis material in which the relative volume fraction is given in fractions from zero to one.

2.5.2. Ordered Subsets Algorithm and Nesterov Acceleration

The ordered subsets algorithm and Nesterov acceleration are used to accelerate the convergence of the optimisation algorithm. According to the ordered subsets algorithm, the projection angles are divided into M subsets $S_1, \ldots, S_M$ [10]. Then, an approximation of the gradient of the cost function is calculated using only subset S_m, which will be denoted as $\nabla_{S_m}\theta(\vec{\alpha})$. This is done to reduce the computational cost and in that way increase the convergence speed [55]. This algorithm will be combined with Nesterov acceleration, which also accelerates convergence of the optimisation algorithm by using the calculated gradient of the cost function $\nabla_{S_m}\theta(\vec{\alpha})$ as a momentum term [10]. Mechlem *et al.* were able to show that using ordered subsets and Nesterov acceleration in the optimisation algorithm given in Section 2.5.1, significantly increases the convergence speed [10]. The full ordered subsets algorithm, combined with Nesterov acceleration, is denoted mathematically as follows:

$$
\begin{aligned}
&\text{Initialize} \quad \vec{z}^{(0)} = \vec{v}^{(0)} = \vec{\alpha}^{(0)},\ t_0 = 1 \\
&\text{For} \quad n = 0, 1, \ldots \\
&\quad \text{For} \quad m = 0, 1, \ldots, M-1 \\
&\qquad k = nM + m \\
&\qquad t_{k+1} = \frac{1}{2}\left(1 + \sqrt{1 + 4t_k^2}\right) \\
&\qquad \vec{\alpha}^{(k+1)} = \vec{z}^{(k)} - (H_\phi)^{-1} \cdot \nabla_{S_m}\theta(\vec{z}^{(k)}) \\
&\qquad \vec{v}^{(k+1)} = \vec{z}^{(0)} - \sum_{l=0}^{nM+m} t_l (H_\phi)^{-1} \cdot \nabla_{S_m}\theta(\vec{z}^{(l)}) \\
&\qquad \vec{z}^{(k+1)} = \vec{\alpha}^{(k+1)} + \frac{t_{k+1}}{\sum_{l=0}^{k+1} t_l}\left(\vec{v}^{(k+1)} - \vec{\alpha}^{(k+1)}\right).
\end{aligned}
\tag{2.50}
$$

The algorithm works to minimise the variable $\vec{\alpha}$ and uses variables $\vec{z}$ and $\vec{v}$ as auxiliary variables for different iteration steps.

2.5.3. Virtual Monochromatic Image

In clinical practice, a virtual monochromatic image (VMI) is used extensively in PCCT. They are images that consist of an attenuation coefficient map of the reconstructions at a specific energy level. This is carried out directly using an adapted version of the material decomposition formula of Equation (2.17) with the reconstructed images of the basis materials as weights:

$$
\mu_{\text{VMI}}(E) = x_{\text{water}} \cdot \mu_{\text{water}}(E) + x_{\text{bone}} \cdot \mu_{\text{bone}}(E). \tag{2.51}
$$

Here, the attenuation coefficient mapped in the VMI for a specific energy E, $\mu_{\text{VMI}}(E)$, is equal to the linear combination between the bone and water attenuation coefficients at the same energy, with the bone and water reconstructed images as weights, x_{bone} and x_{water}, respectively. The bone and water images are thus multiplied by their respective mass attenuation coefficients at a certain energy and then added to produce the VMI. VMIs are used to detect abnormalities in the image and to determine the diagnoses of patients [56]. Therefore, improving soft tissue contrast is very important for this application.

2.5.4. Image Artefacts

The term artefact refers to any systematic deviation between the reconstruction of a CT scan and the true attenuation coefficients. There are two important artefacts that can be encountered in modern-day CT scanning. The first artefact, known as beam hardening, arises from the physical properties of X-rays. As mentioned earlier, X-ray beams contain photons of various energy levels. When passing through an object, lower-energy photons are absorbed more effectively by the object than higher-energy photons [32]. As lower-energy photons are absorbed, the average photon energy of the beam increases while passing through the object, that is, the beam 'hardens' [57]. Consequently, the attenuation coefficient will depend on the path length of the X-ray beam, resulting in an apparent decrease in attenuation in a uniform object. This is called the cupping effect.

Another artefact that is common in CT is the partial volume effect. This effect usually occurs when a certain voxel encompasses more than one different substance [58]. The voxel is then assigned the weighted average of the attenuation coefficients of these materials, depending on the relative volume fractions of each material in the voxel. This causes the reconstructed attenuation coefficient value to be skewed from the true value in that voxel. In this thesis, an object is scanned in the reconstruction that has assigned materials per different voxel, meaning that there is no overlap between different materials. However, due to the limited spatial resolution in CT reconstruction, bordering voxels with different materials are often blurred [58]. For example, two bordering voxels with high contrast in attenuation are smoothed out in the reconstruction, even though each pixel has a single assigned material.

3

Method

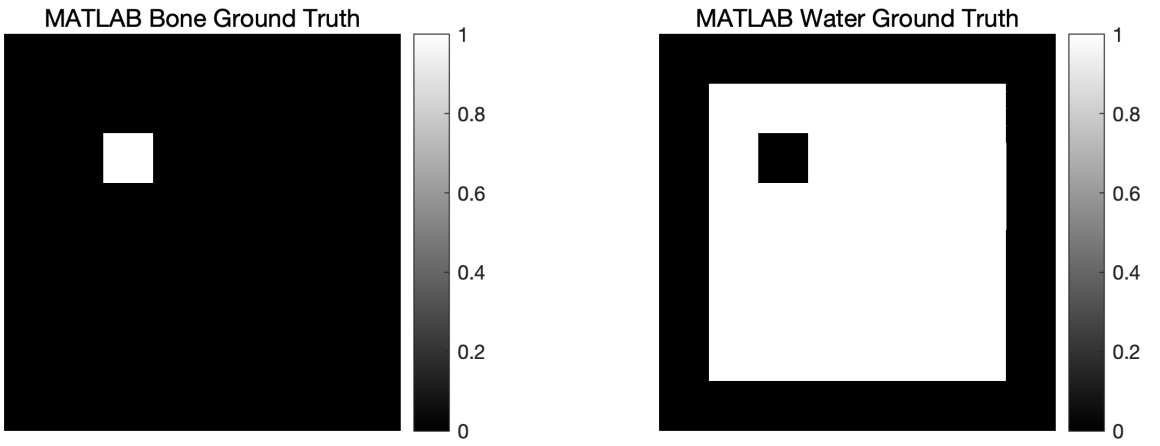
In the following chapter, the methods used in this thesis are explained in detail. First, the process of converting the provided `MATLAB` code to `Python` is laid out along with the encountered difficulties. Then, a more complex phantom is built with multiple biological materials that can be implemented in the `Python` code. Lastly, this phantom is used to build reconstruction images with different variables, after which they are used to reconstruct other types of images, such as SPR maps and VMIs. The image quality is determined using a number of different methods for each implementation of the reconstructed images.
The variable mapped in the reconstructed image is the relative volume fraction. This variable is introduced in Equation (2.18) to represent the weights to describe any biological material in the energy range of 20 - 150 keV as a linear combination between bone and water. These values are calculated by dividing the optimised variable α_j^b by the density of the respective basis material b. Dividing α_j^b by the density results in a unitless quantity that represents the relative volume fraction of basis material b in voxel j. In other words, this quantity reflects the fraction of the voxel that is occupied by basis material b, and therefore the values range from zero to one. The equations to calculate the SPR maps and VMI from these relative volume fractions are given in their corresponding sections.

3.1. Code Process and Explanation

As part of my thesis, I was assigned to convert a `MATLAB` code implementation to `Python`. The `MATLAB` implementation is designed to perform CBCT image reconstruction using the algorithm explained in Section 2.5.1. The original code was developed by Mory *et al.* [11], which was initially intended to be used for FBCT. As part of his Master's thesis, Jacco van Leenen adapted this code to work for CBCT in `MATLAB` [12]. This implementation consists of twelve different files, containing the calculations based on the algorithm by Mechlem *et al.* [10]. The reason for converting this code to `Python` is to make it more efficient by allowing integration with open software packages available in `Python`.

3.1.1. The `MATLAB` Phantom

To be able to run the reconstruction, a phantom is built to represent a sample for which the reconstruction algorithm creates a material map. A phantom is a volumetric object that is used in the code as a digital object sample, representing an object that can be used in a CT scanner. This virtual sample consists of a number of voxels decided by the length of each edge of the volume. In this thesis, each basis material is represented by a separate phantom component,



(a) Plotted slice of the bone part of the original 32x32x32 voxel phantom.

(b) Plotted matrix of the water part of the original 32x32x32 voxel phantom.

Figure 3.1: Slices of the original 32x32x32 voxel phantoms from the `MATLAB` code.

which is constructed from the relative volume fraction that indicates the location of a specific material within the phantom. Arbitrary slices of the phantom components used in the original `MATLAB` algorithm can be seen in Figure 3.1. In this case, the phantom is a simplified version of reality containing only bone and water. Therefore, each phantom component slice has a value of one in the voxels where the material of their respective component is located. For example, the bone component of the phantom has a value of one for each voxel in which bone is located and zero if there is water present. On the other hand, the water component has a value of one if there is water in that voxel and zero if there is bone. The region of the components that has a value of zero in all basis materials is strictly vacuum and therefore omitted in the reconstruction.

This phantom is of course a very simplified version of real data from a CT scanner; however, it is a starting point to observe how an object is reconstructed using this algorithm. In a later section, a more complex phantom is implemented in the same reconstruction algorithm. Here, more biological materials, such as blood and muscle, are added to the phantom.

3.1.2. `MATLAB` to `Python` Code Conversion

Most of the conversion was done using ChatGPT. The smaller files and functions were relatively easy to convert with AI; however, ChatGPT made many mistakes in the larger and more complicated files. Therefore, these errors needed to be fixed to create a working implementation of the `MATLAB` code in `Python`. To validate that the errors were corrected, equivalent phantoms and variable values are implemented in both codes. In addition, the random variables that are introduced in the codes to reflect a more realistic scenario are turned off. The `Python` and `MATLAB` codes were then required to provide exactly equivalent outputs at all points in the codes.

The errors were mainly based on the fact that `Python` is a zero-based programming language, while `MATLAB` is one-based. This means that, for example, the first element in an array in `Python` is called using the following command: `array[0]`. This is opposed to `MATLAB`, where the first element in an array is called using `array(1)`. There were many small tweaks that were necessary for the code to work in `Python`, such as shifting the `range` function in the for-loops and array indexing elements by one.

The file in which most of the problems occurred was `conebeamtomo3.py`. This was also the longest function in the `MATLAB` implementation, and therefore ChatGPT had many difficulties

converting this code. The purpose of this file is to create the forward projection matrix **A** by calculating the coordinates of each projection angle through the object and then calculating the number of voxels in the object that are hit by the X-rays. This variable corresponds to the line integral A_i^b as shown in Equation 2.23, for each detector pixel and projection angle i and basis material b. There were many problems that arose from the fact that `Python` is zero-based. As most of the elements of **A** are zero, the results of the final forward projection matrix are stored in a sparse matrix to reduce memory.

After the forward projection matrix is correctly made and has equivalent values in both `Python` and `MATLAB`, the optimisation algorithm explained in Section 2.5.1 is used to determine each voxel value of the reconstructed image. As `MATLAB` and `Python` have different ways of formulating the dimensions of matrices, certain changes had to be made to be able to multiply them to acquire an accurate result. As the matrices contain a very large number of values, these are not easy to compare with their counterparts in `MATLAB`. The best way to accurately ensure that the calculations were equivalent is to download the `MATLAB` files of a certain matrix, load it into `Python`, and then compare it with the equivalent matrix using the following example prompt: `print(np.max(np.abs(array_Python - array_MATLAB)))`. This is essentially calculating the maximum absolute difference of two arrays. If this results in a value of zero or a value so close to zero that it is negligible, we can conclude that these arrays are equal. Going through every calculation in `Python`, and ensuring that every outcome of these calculations results in equal matrices in both `MATLAB` and `Python`, was the only way to ensure that both implementations are exactly equivalent.

To verify that the `MATLAB` and `Python` codes produced equivalent results, the mean absolute difference is taken between the final reconstructed images. The reconstruction variables are equal in both codes: `objectSize = 32`, `nIter = 10`, and `nProj = 100`. The remaining variables have values as given in Table 3.1.

3.1.3. The More Complex Phantom

After obtaining equivalent results in both `MATLAB` and `Python`, a more complex phantom is implemented in the `Python` algorithm. To provide an easy visual representation, the density map of this phantom is shown in Figure 3.2. In the image, a slice of the phantom is shown where the location of each material is indicated with their densities. For bone, the same density value is used as in the original 32x32x32 `MATLAB` phantom, which is 1.92 $\mathrm{g\,cm^{-3}}$. The bone is

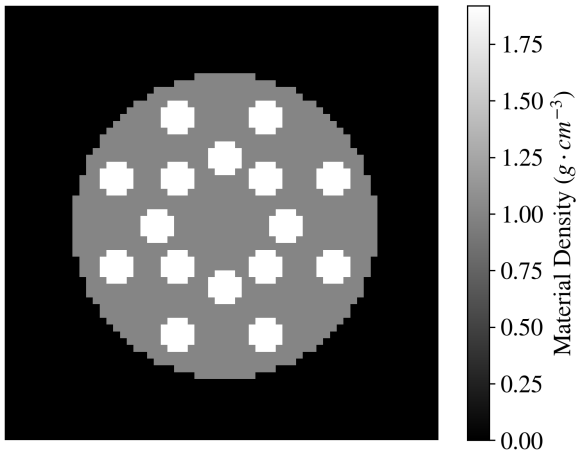


Figure 3.2: Density map of the more realistic phantom used in the `Python` code. The radius of the phantom is taken to be 22.4 mm.

then located in the small white circles, and the surrounding grey area is water with density 1.0 $\mathrm{g\,cm^{-3}}$. In the reconstruction programme, the pixel pitch was set to one, which means that the size of a pixel in the phantom is equal to 1 mm. This more complex phantom was given a size of 64x64x64 voxels, which means that the phantom represents a cubed object with sides of 6.4 cm. The radius of the phantom was chosen to be 22.4 mm.

In order to be able to add other materials to the phantom, the contribution of the basis materials to other biological materials has to be determined in the form of a linear combination. As explained in Section 2.3.1, it has been proven that the attenuation coefficient of any biological material can be written as a linear combination of the attenuation coefficients of a set of basis materials. This linear combination is given in Equation (2.18). However, before this equation can be used, the attenuation coefficients of bone and water must be calculated for each energy in the energy range, denoted as $\mu_{\text{water}}(E)$ and $\mu_{\text{bone}}(E)$, respectively. The mass attenuation coefficient as a function of energy is found using the `Python` library `xraydb`, which contains functions that store these values. The function that returns the variable μ/ρ in SI units $\mathrm{cm^2\,g^{-1}}$ is called `xraydb.mu_elam`. As mentioned in Section 2.5.1, the typical range in which CT scanners operate is 20 - 150 keV. The variable μ/ρ is calculated for each energy in this range and then multiplied by their density in $\mathrm{g\,cm^{-3}}$ to obtain their respective attenuation coefficients. However, the function `xraydb.mu_elam` only provides the mass attenuation coefficient μ/ρ for elements of the periodic table. This is solved by using the composition of bone and water from their elements, calculating the attenuation coefficient, and adding these together as such:

$$\frac{\mu}{\rho} = \sum_i w_i \Big(\frac{\mu}{\rho}\Big)_i \,, \tag{3.1}$$

where w_i is the fraction of the weight of a certain element i and $(\mu/\rho)_i$ the mass attenuation coefficient that corresponds to that element. The compositions of all biological materials used were obtained from a NIST database [59]. After entering the compositions of the basis materials into the code, μ/ρ was calculated using the function `xraydb.mu_elam` to obtain the mass attenuation coefficient in SI units $\mathrm{cm^2\,g^{-1}}$. The approximated attenuation coefficients of water, blood, and muscle match the factual values very precisely, all of which have a mean relative error of 0.01% for eight known data points provided by NIST [60]. The mean relative error for the approximation of the attenuation coefficient for bone with the same calculations is 2.05%.

Now that the attenuation coefficients of bone and water are known, we can begin to approximate the attenuation coefficient of other biological materials in the body, in this case chosen as muscle and blood. These are both soft tissue materials and therefore very related to water in terms of density and attenuation coefficient. To calculate the linear combination weights c_1 and c_2 from Equation (2.18), the compositions of muscle and blood are obtained from the same NIST database used for bone and water. Using the same formula as given in Equation (3.1), the attenuation coefficient of blood and muscle is calculated. After this, they are fitted to the function in Equation (2.18) using a least-squares fit in `Python`. The results of this fit are the linear combination weights of Equation (2.18) which best describe the attenuation coefficients of muscle and blood. To implement these values in the phantom, Equation (3.1) is used. After determining the exact location of each material, the following equation can be used to determine the phantom as a density map:

$$\text{Phantom} = \sum_b \left(\sum_i w_{i,b} \cdot \text{Fraction}_i \right) \cdot \rho_b \,. \tag{3.2}$$

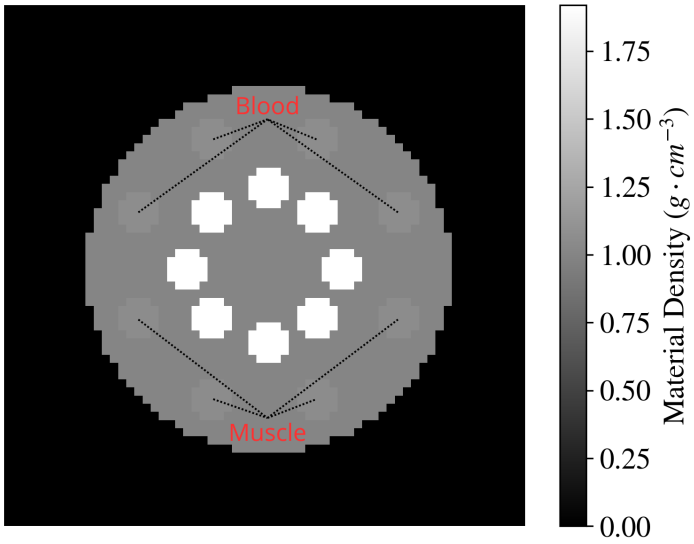


Figure 3.3: Detailed 64x64 phantom including blood and muscle. The upper half of the smaller circles contain blood and the lower half contain muscle. The colorbar represents the densities of the materials present in the phantom.

Here, the outer sum runs over the basis materials b. Each fraction of material i, Fraction_i, is assigned to basis material b by multiplying by a weight $w_{i,b}$, which specifies how much material i contributes to basis material b. Again, a density map of the phantom with more materials is shown in Figure 3.3.

Reconstructed images with iterations 5, 10, 12 and 25 are obtained from the algorithm, after which their image quality is calculated with the methods above. The reconstruction variables are equal in both codes: `objectSize = 32`, `nIter = 10`, and `nProj = 100`. The remaining variables have values as given in Table 3.1.

3.1.4. Problems with the Complex Phantom

In a previous implementation of the complex phantom, the radius was taken to be 25.6 mm instead of 22.4 mm. However, the reconstruction produced images that exhibited a substantial amount of noise, especially in the background of the images. After investigating, we noticed that the phantom was not within view of the detector. The detector has a size of 6.4 cm in its length, and 4.4 cm in its width and therefore the projection of the phantom onto the image should be within these limits. In the Appendix B.1, figures are given that show the reconstructed images for the larger phantom. Moreover, projections of the larger phantom are given, along with an explanation on how this problem was solved. Therefore, the phantom was adapted to the one shown in Figure 3.3.

3.2. Code Implementation

Now that the phantom is built to fully be within the field of view of the detector, different variables are investigated, and their optimised values are determined to achieve the highest image quality. The code contains a number of variables that can be changed to adjust the reconstructed image. These variables can also determine the quality of the image. The variables used in the code of the optimisation algorithm are displayed in Table 3.1. Here, each variable is given with their initial value originating from the `MATLAB` code, except for the `objectSize` variable, which has been increased to 64 to obtain the more complex phantom as explained in Section 3.1.3.

Table 3.1: Variables used in the reconstruction algorithm with their initial `MATLAB` value and a brief description.

Variable	Value / Dimensions	Description
`objectSize`	64 pixels	Size of the phantom, phantom is built as an `objectSize` x `objectSize` x `objectSize` matrix.
`dSource`	2300 mm	Distance of the phantom to the source of the X-rays.
`dDetector`	900 mm	Distance of the phantom to the detector.
`nProj`	300	Amount of projections performed by the scanner, which is equal to the number of different angles.
`pixelPitch`	1 mm	Distance between two detector pixels.
`nPixelsY`	64	Amount of pixels in the y-direction of the detector.
`nPixelsZ`	44	Amount of pixels in the z-direction of the detector.
`nSubset`	4	Amount of subsets used in the Ordered Subsets algorithm in the reconstruction.
`nIter`	10	Amount of iterations through the algorithm.
`lambda_`	[100, 50]	Tuning parameter of the regularisation term, which determines the quantity of the effect of the regularisation of bone and water, respectively.
`delta_huber`	[0.01, 0.05]	Huber regularisation parameter of bone and water, respectively.

The variables of greatest interest are `nIter`, `lambda_`, and `delta_huber` because they have the greatest influence on the final reconstructed image. The descriptions of these quantities are shown in Table 3.1. Both `delta_huber` and `lambda_` directly affect the influence of regularisation in the image. Note that the `delta_huber` variable in the code is denoted as γ_b in Subsection 2.5.1 and is henceforth referenced as such. This also applies to the variable `lambda_`, which is denoted as λ_b. The values incorporated in the code are the values that were given in the original publication of the algorithm.

3.2.1. Reconstructed Image Quality and Contrast

An easy way to get an indication on the quality of a reconstructed image is to calculate the mean squared error (MSE) in the image. The MSE can be defined as follows:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(x_{\text{image}} - x_{\text{ground truth}})^2 \,. \tag{3.3}$$

Here, x_{image} represents the reconstructed image or a specific region within the image, and $x_{\text{ground truth}}$ the ground truth image. The MSE is calculated only for a certain region of interest (ROI) to exclude unnecessary information and noise. The used ROI depends on the regions in which the MSE is calculated. This ROI is isolated in the reconstructed image using a mask in `Python` to isolate certain regions in the images. To use Equation (3.3), the ROI must be the same in the ground truth and reconstructed image.

Another way to determine the quality of the reconstructed image is to calculate the contrast of an image. Specific ROIs are isolated with the use of masks. These include the regions with water, bone, blood, and muscle, from which the mean value of the relative volume fraction of each material is calculated independently. With these mean values, the contrast can be calculated between specific regions using the formula for the Michelson contrast:

$$\text{Contrast} = \left|\frac{\mu_{\text{region, 1}} - \mu_{\text{region, 2}}}{\mu_{\text{region, 1}} + \mu_{\text{region, 2}}}\right| . \tag{3.4}$$

In the equation, $\mu_{\text{region, i}}$ represents the average value of the volume fraction within a certain ROI i. These regions could represent, for example, the object and the background, or two distinct materials within the object, such as bone and water. The results of this formula range from zero to one. In this range, zero indicates that the regions are indistinguishable from each other. On the other hand, a contrast value of one indicates maximum contrast, where one of the attenuation coefficients in the numerator equals zero in the limit. Ideally, the contrast of a reconstructed image is equal to the contrast in the ground truth image for the same regions.

The implementation of the more complex phantom shown into the now correct `Python` code is performed using the following number of iterations: 5, 8, 10, 12, 15, 18, 20, 25. For all images, a total of 300 projections are used and all other variables in Table 3.1 remain the same. To obtain an idea of the quality of the reconstructed images, the MSE is then calculated with Equation (3.3) for the bone and water regions using the masks as described above. Moreover, the mean values of the relative volume fraction of the water, bone, blood, and muscle regions are calculated. With these mean values, the contrast between water and bone, between water and blood, and between water and muscle is calculated with Equation (3.4).

3.2.2. Regularisation

It is also interesting to consider the role of the regularisation function and the regularisation parameters used in the reconstruction. As explained in Section 2.2, the regularisation term is needed to stabilise the cost function to obtain an accurate estimation of the parameter $\vec{\alpha}$. It is possible to vary the variables γ_b and λ_b from Equation (2.29) to determine the role of this regularisation in the reconstruction.

In Table 3.1, the original values for the tuning parameter λ_b and the Huber tuning parameter γ_b are given for the basis materials water and bone. To draw conclusions on the influence of these tuning parameters on the reconstruction, these values are altered and their reconstructed images observed. First, these values are all decreased to zero to observe how the reconstruction

behaves without the presence of a regularisation term. This is done by implementing the tuning parameters $\lambda_b = [0, 0]$ and $\gamma_b = [0, 0]$. However, as explained in Section 2.2, the cost function becomes unstable when regularisation is absent. Consequently, the code will produce singular matrices if the tuning parameters are given as $\lambda_b = [0, 0]$ and $\gamma_b = [0, 0]$. Therefore, instead of inputting zero, negligible values are implemented for the tuning parameters. These are arbitrarily chosen as $\lambda_b = [0.001, 0.001]$ and $\gamma_b = [0.00001, 0.00001]$. With these values, the regularisation provides a stable solution to the problem, without having a large influence on the reconstructed images. To confirm that the regularisation term influences the final reconstructed image only with negligible values, the contribution of the regularisation function to the total cost function is plotted. Moreover, to study the effect of regularisation on the reconstruction, the originally provided tuning parameters are first halved, after which they are also doubled. Therefore, the reconstruction is performed first with the tuning parameters $\lambda_b = [50, 25]$ and $\gamma_b = [0.005, 0.025]$ and then with $\lambda_b = [200, 100]$ and $\gamma_b = [0.02, 0.1]$. These values were chosen to study the effect on the reconstruction when regularisation plays a smaller and larger role.

3.2.3. SPR Maps

SPR maps of the reconstructed images are built using a provided `Python` code. This code calculates the SPR values of the imported reconstructed images and also builds an SPR map of these values. This is done by calculating the necessary parameters that are used in Bethe's equation in Formula (2.20). These parameters include the electron density, the mean excitation energy, and the relative velocity of the proton. All parameters are calculated at an arbitrary energy of 200 MeV, as SPR is assumed to be constant at different energies in clinical practice [50]. The electron density is calculated using a dictionary with the mass fractions for each element of a basis material and their mass densities. Then, the mean excitation energy is calculated using the Bragg additivity rule as given in Equation (2.21). Lastly, the relative velocity of the proton is calculated using its kinetic energy, previously mentioned to be 200 MeV. The reconstructed image is imported into the code that was built with `nIter = 10` and `nProj = 300`, after which an SPR map is made. Following the attenuation coefficient model explained in Section 2.3.1, the SPR is calculated for each basis material separately and combined using a version of Bragg's additivity rule to obtain the complete SPR map [51]:

$$\mathrm{SPR}_{\mathrm{tissue}} = \sum_b \omega_b \cdot \mathrm{SPR}_b \,. \tag{3.5}$$

Here, the SPR of a certain tissue is equal to the sum of the SPR values of all basis materials multiplied by their weight fraction b. This equation is useful for determining SPR maps from a CT reconstructed image, where the weight fractions are the reconstructed images of bone and water, and therefore the relative volume fraction of basis material b. This equation is also used for the ground truth of the phantom to obtain a ground truth SPR map, which is compared with the reconstructed SPR map to obtain an idea of the image quality. The ground truth SPR map is calculated again using Equation (3.5). For both the reconstructed image and the ground truth SPR maps, the colour scale is forced to be equal for a better visual comparison.

To determine the quality of the SPR map, the MSE is calculated. This is because, as explained in Section 2.4, SPR maps are desired to have values very close to reality to be used in proton therapy treatment plans. Therefore, it is interesting to investigate the error between the reconstructed image and the ground truth SPR maps to obtain an idea of the quality. However, the calculation of the error is done differently than in Section 3.2.1. Once again, the MSE is calculated because of its more accurate quality estimation. The MSE for each biological material in the phantom is calculated separately using a mask. These masks are eroded for the MSE calculations of the SPR maps, which means that each border pixel from a certain region

is deleted. This is done to reduce the influence of border artefacts, such as the partial volume effect, in the calculations.

To obtain a different measure of the quality of the reconstructed SPR map compared to ground truth, a profile is plotted in `Python`. A profile is made by choosing a line in an image, after which all values on this line are plotted in a line diagram. This is an effective way to compare the reconstruction with the ground truth, because the fluctuations in values due to imaging artefacts can be seen clearly. The profile is taken from a horizontal line through the middle of the SPR map. Therefore, it also passes through two small circles with bone to capture how the SPR values resulting from the reconstruction change compared to the ground truth SPR map. The profiles are made for both the ground truth SPR map and the reconstructed SPR maps with 10 and 25 iterations.

3.2.4. Virtual Monochromatic Image

The specific energy for the VMI is chosen arbitrarily as 80 keV and constructed from a reconstruction of ten iterations and 300 angles using the formula provided in Equation (2.51). The attenuation coefficients are calculated for the energy of 80 keV in the same way as in Section 3.1.3 with the `xraydb` library.

As VMIs are used to detect abnormalities in the body to diagnose certain diseases, it is important to have a high contrast between areas with different materials. Therefore, to determine the quality of the VMI, the contrast is calculated. However, this is done differently than in Section 3.2.1. The ratio is calculated between the contrast of the reconstructed VMI and the 'true' contrast of the ground truth VMI. This measure is called the contrast recovery coefficient (CRC). Both of these contrasts are obtained using Equation (3.4), after which the ratio is calculated as follows:

$$\text{CRC} = \frac{\text{Contrast}}{\text{True Contrast}} \,. \tag{3.6}$$

Here, the variable Contrast is the contrast between two specific regions inside the reconstructed VMI and the variable True Contrast the contrast between the same regions in the ground truth SPR map. A CRC of one is desirable, as in that case the contrast of the reconstructed image would be equal to the contrast of the ground truth. When the CRC surpasses one, it would mean that the reconstruction caused an overestimation of the VMI values, while a ratio smaller than one signifies an underestimation.

The CRC is calculated separately between the water and bone regions, the water and muscle regions, and the water and blood regions. The contrast is then calculated from both the reconstructed image VMI and the ground truth VMI using eroded masks that isolate a specific area before using Equation (3.4). Just as with the MSE calculations, this is done for the following number of iterations: 5, 8, 10, 12, 15, 18, 20, 22, 25.

4

Results

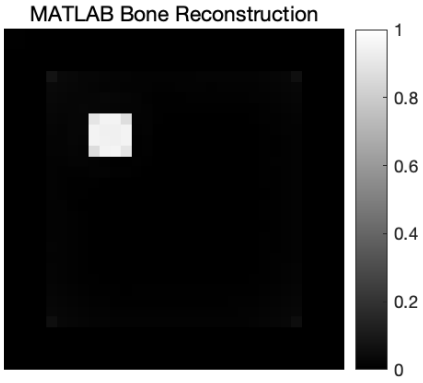
In the following chapter, the results of the conversion of `MATLAB` to `Python` are given, after which the results of different applications of the converted code are shown and explained.

4.1. `MATLAB` to `Python` conversion results

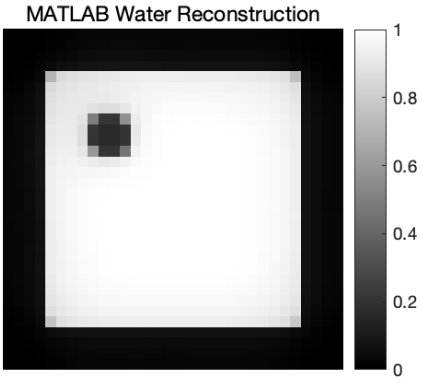
As explained in Section 3.1.2, the optimisation algorithm was transferred from a provided `MATLAB` code to `Python`. Here we validate if we obtain equivalent results in both `MATLAB` and `Python`. The results of the provided `MATLAB` algorithm are shown in Figures 4.1a and 4.1b, which are the bone and water reconstructed images, respectively. The results of the `Python` reconstructed images of the same phantom are given in Figures 4.1c and 4.1d. However, where the `Python` code needs 38.5 seconds to provide a reconstructed image, `MATLAB` takes 16.9 seconds, which is a 127% increase in time for the same phantom and variables. The reason behind this is that the current code is not yet an optimised implementation for `Python`.
The mean absolute error between the relative volume fraction values is calculated when there is no Poisson noise present. We find an mean absolute error value of 0.0003 in the bone reconstructed image and 0.0007 in the water reconstructed image. Since the range of values is [0,1], we can conclude that both codes provide very similar results. Moreover, during the conversion process, the mean absolute error between more different matrices was calculated. The mean absolute error values always resulted in negligibly small values, which means that the matrices in both codes are equal.

4.2. `Python` Reconstruction Results

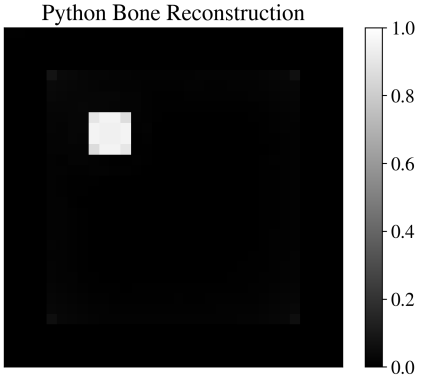
The results of the reconstruction at different iterations are shown in Figure 4.2.
It is clear that for more iterations the reconstructed images seem to have a greater resemblance to the ground truth value of the phantom. The bone and water reconstructed images for 5 iterations do not seem to resemble the ground truth well, which can be seen in Figures 4.2(a) and 4.2(b). There is a high presence of the cupping effect in the middle of the water region, and the values of the bone region have not yet converged to the right value.
It is obvious that the reconstructed images of water improve significantly for a higher number of iterations. In Figure 4.2(f), it is clear that the artefacts present in the images (b) and (d) have decreased substantially, although they are still present in the form of partial volume artefacts around the bone regions. Furthermore, the presence of artefacts in the background is also greatly reduced in image (h) compared to the images with 5 and 10 iterations. However, the contrast between the bone and water regions is reduced compared to the water image



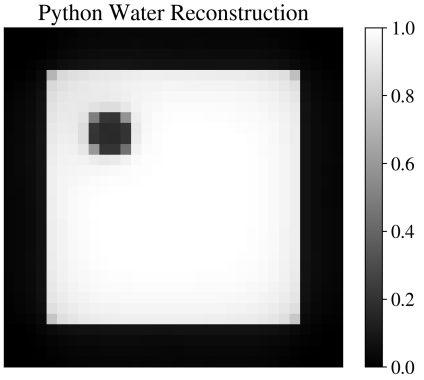
(a) MATLAB bone reconstructed image of the original 32x32 phantom.



(b) MATLAB water reconstructed image of the original 32x32 phantom.



(c) Python bone reconstructed image of the original 32x32 phantom.



(d) Python water reconstructed image of the original 32x32 phantom.

Figure 4.1: Reconstruction of the original 32x32 pixel phantom from MATLAB and from Python.

with 10 iterations. This can be seen in Table 4.1, where the mean values of the relative volume fractions of the water and bone regions are given for each image, as well as the contrast between the water and bone regions. From the table, it is clear that the mean fraction value for water approaches one for a higher number of iterations, which is the desired result. However, the mean fraction value for the bone, which should go down to zero, increases for 25 iterations. This is because the bone contribution becomes overestimated for higher iterations. This is also clear from the contrast between the water and bone regions in the table. The highest contrast is given for 12 iterations, after which this value decreases when the bone contribution becomes overestimated.

Water Reconstruction	Mean Water Region	Mean Bone Region	Contrast Water–Bone
5 iterations	0.672	1.274	0.309
10 iterations	0.875	0.123	0.752
12 iterations	0.907	0.072	0.853
25 iterations	0.924	0.334	0.469

Table 4.1: Mean of the relative volume fraction values for water and bone, and their contrast, calculated from the water reconstructed images in Figure 4.2.

The bone reconstructed image with 5 iterations in Figure 4.2(a) does not seem to represent the ground truth image well. The contrast between the reconstructed bone regions and the background appears to be low, meaning that the values of those pixels have not yet converged

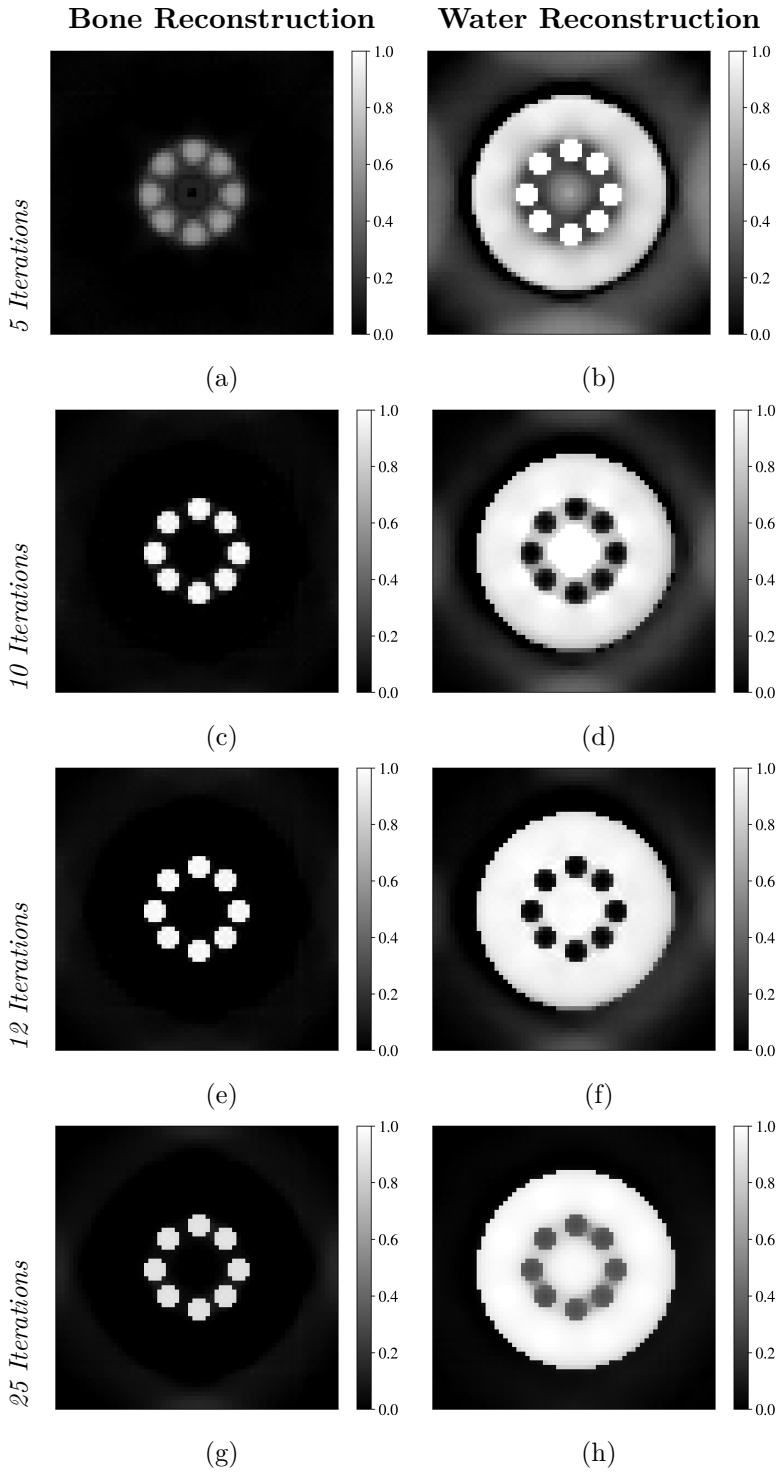


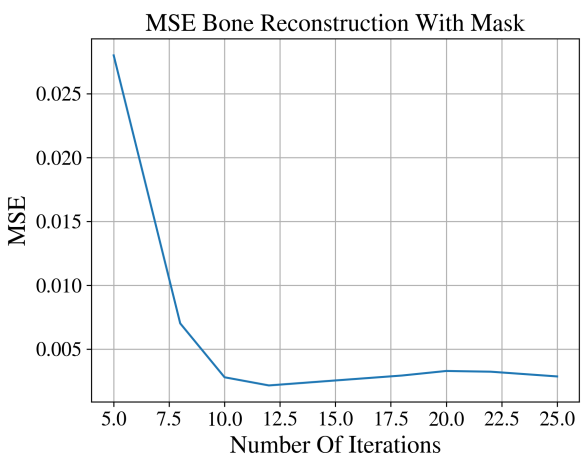
Figure 4.2: Reconstructed images of bone and water phantoms for different iteration counts. The first column shows the bone reconstructions for iterations 5, 10, 12 and 25, while the second column shows the water reconstructions for the same iterations.

to their true values due to the low iteration count. Table 4.2 gives the mean values for the water and bone regions, as well as the contrast between these regions for each bone image in Figure 4.2. The mean bone value is again closest to its desired value for 12 iterations, as now the bone contribution becomes underestimated for higher iterations. The mean water values

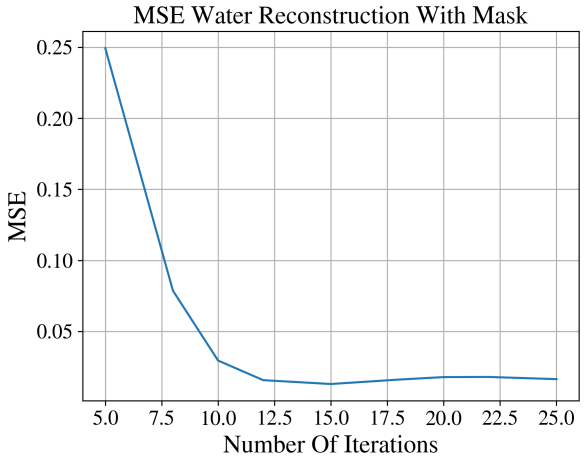
Bone Reconstruction	Mean Water Region	Mean Bone Region	Contrast Water–Bone
5 iterations	0.060	0.523	0.795
10 iterations	-0.002	0.938	1.004
12 iterations	-0.010	0.960	1.021
25 iterations	-0.002	0.886	1.004

Table 4.2: Mean of the relative volume fraction values for water and bone, and their contrast, calculated from the bone reconstructed images in Figure 4.2.

are negative, which is not physically feasible. These are likely caused by artefacts around the phantom.



(a) MSE over different iterations for the masked bone reconstructed image.



(b) MSE over different iterations for the masked bone reconstructed image.

Figure 4.3: MSE over different iterations for the masked bone and water reconstructed images.

Using the procedure and equations provided in Section 3.2.1, the mean squared error will be calculated for different iterations. The results are given in Figure 4.3a for the MSE of the masked bone reconstructed image and in Figure 4.3b for the MSE of the masked water reconstructed image. As expected, the MSE of the images with a lower number of iterations is much higher than that of a higher number of iterations. This is, of course, because the reconstructed image values with fewer iterations have not yet converged to their actual value. Moreover, it is clear that the MSE for both images converges to a certain value and either stabilises or increases after that convergence point. This convergence point is 12 iterations for the reconstructed image of bone and 15 for the reconstructed image of water.

4.3. Regularisation Results

The bone and water images in which the tuning parameters are taken as $\lambda_b = [0.001, 0.001]$ and $\gamma_b = [0.00001, 0.00001]$ are shown in Figures 4.4 (a) and 4.4 (b), respectively. All other algorithm variables have values as given in Table 3.1. Moreover, a graph that provides the regularisation cost for every iteration in the reconstruction is given in Figure B.3a in the Appendix, along with the regularisation cost of a reconstruction with the original tuning parameters. From here, we can conclude that the regularisation only provides negligible values to the final reconstructed images. Before looking at the figures, it is notable that the time it takes to reconstruct the images without regularisation is around 246.8 seconds, opposed to the 193.9 seconds it takes to reconstruct the images with regularisation for 10 iterations. This is

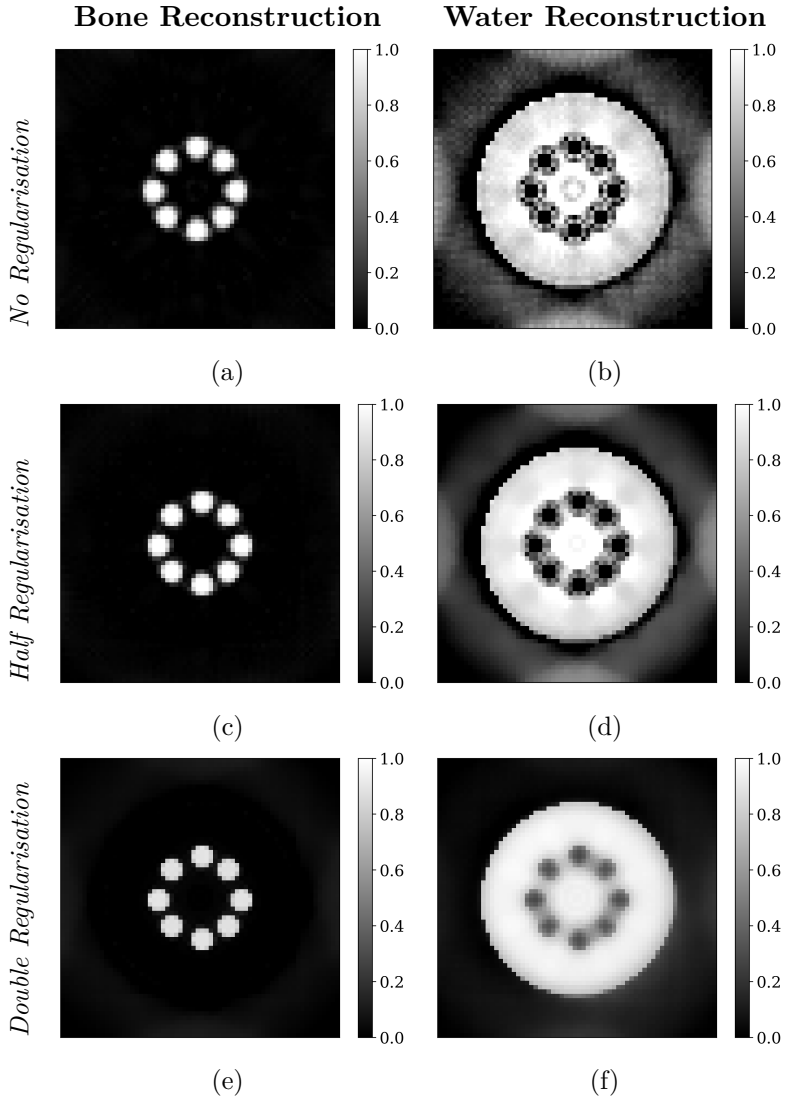


Figure 4.4: Bone and water reconstructed images with 10 iterations with different tuning parameters.

almost a 30% increase in the duration of the reconstruction. It is clear that the water image appears to have a lower quality compared to its counterpart with regularisation, shown in Figure 4.2(d). This can be seen in the large amount of noise in the background, as well as large fluctuations around the borders of the phantom and between the water and bone regions. This is, of course, as expected. The bone reconstructed image without regularisation again shows a higher presence of the partial volume effect compared to Figure 4.2(c), with larger fluctuations around the borders of the bone regions. The effect of regularisation can directly be analysed from this image. In both images, the values are much less smooth in comparison to the images with regularisation. As explained in Section 2.2, the absence of a regularisation term means that large fluctuations are not suppressed and therefore are present in the final reconstructed images. Moreover, the large presence of noise in the background, especially in Figure 4.4 (b), can be attributed to the fact that the absence of regularisation results in a decrease in sparseness. This means that the small values in the background are not pushed down to zero. To give a more quantifiable result that the image quality has decreased, the mean squared error is given for both the bone and the water images. This is calculated in the same way as in Section 4.2. The MSE for the bone image is equal to 0.007087, which is a 153% increase compared to

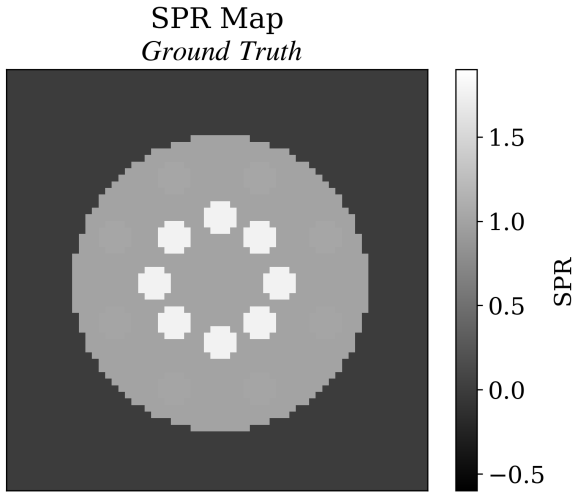
the MSE of the bone image with regularisation, which is 0.002797. Moreover, the MSE of the water image experiences an increase of 284%, from 0.029449 with regularisation to 0.112964 without. Therefore, for both reconstructed images, the MSE increases significantly; however, the lack of regularisation has a much larger effect on the water image than on the bone image.

Now, the reconstruction algorithm is implemented with half the value of the original tuning parameters: $\lambda_b = [50, 25]$ and $\gamma_b = [0.005, 0.025]$. The results are shown in Figures 4.4 (c) and 4.4 (d). It is clear that the artefacts that appeared in Figures 4.4 (a) and 4.4 (b) are less present. This is, of course, the direct effect of regularisation as explained in Section 2.2; it ensures that for certain voxel differences there is a quadratic punishment, smoothing out these differences. To quantify, the MSE for the bone image is 0.004837, which is a 73%, while the MSE for the water image is 0.073079, a 148% increase compared to the images with regularisation. It is clear that using half the values of the regularisation tuning parameters still results in images that have a significantly lower quality. This can, of course, also be seen visually, when comparing the bone and water images in Figures 4.4 (c) and 4.4 (d) with images 4.2(c) and 4.2(d). However, the small blood and muscle circles in the outer ring of the phantom are more visible than in the image with the original amount of regularisation. This must be taken into account, as one of the objectives in improving this reconstruction algorithm is to increase the soft tissue contrast.

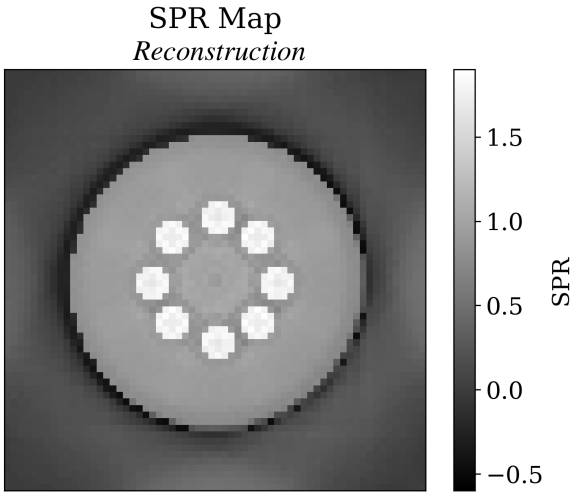
To analyse the quality of the images for a higher amount of regularisation than is used in the original code, the tuning parameters are doubled: $\lambda_b = [200, 100]$ and $\gamma_b = [0.02, 0.1]$. As in the previous cases, the MSE is calculated as in Section 4.2 and all other variables are as displayed in Table 3.1. The results of the bone and water reconstructed images are presented in Figures 4.4 (e) and 4.4 (f), respectively. At first glance, it is clear that this image is smoothed out in both the background and the actual phantom. This results in a well-reconstructed background. However, this is not as important as the information inside the phantom. It is clear that the smaller bone circles in the water reconstructed image are oversmoothed. Moreover, the blood and muscle circles that were visible in Figure 4.4 (d), are also smoothed and therefore not visible. The MSE of the bone image is now 0.004245, a 51% increase compared to the bone reconstructed image with the original regularisation tuning parameters. In addition, the water image experiences a 35% increase, up to 0.039710. This means that the image quality of the water image is much higher compared to Figure 4.4 (d), while the quality of the bone image is similar.
We can conclude that the original tuning parameters were well chosen. However, the image quality was only tested for large deviations from the original tuning parameters, and therefore more tests are needed to conclude that the tuning parameters used in this algorithm are optimal.

4.4. SPR Maps Results

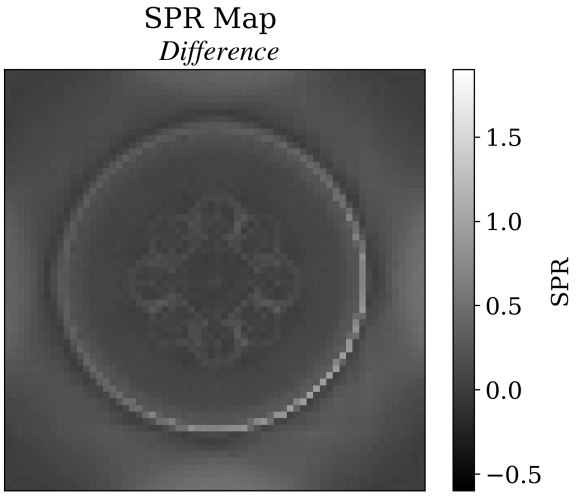
The SPR maps for the ground truth and the reconstructed image provided by the `Python` code can be seen in Figures 4.5a and 4.5b, respectively. These figures are obtained using Equation (3.5). In addition, the difference between the ground truth and the SPR map built from the reconstructed images is shown in Figure 4.5c, which shows the regions that have the largest errors in the reconstructed image compared to the ground truth. The first thing we notice is that the background of the reconstructed image SPR map exhibits a large amount of noise. Moreover, around the border of the phantom, the values become negative, which is not physically possible for the SPR. This could be a consequence of noise amplification in the reconstruction, but should be investigated further. To provide a qualitative measure of the SPR reconstructed image, the mean SPR value in the bone regions is 1.766, calculated



(a) SPR map of the ground truth.



(b) SPR map of the reconstructed images.



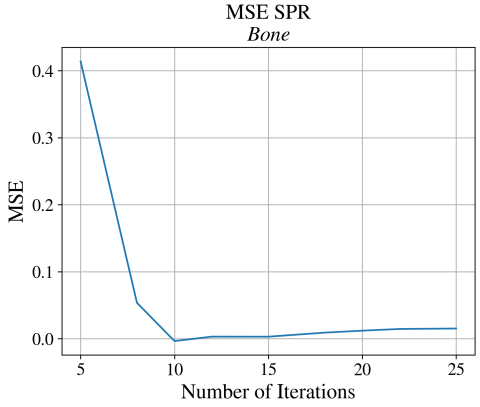
(c) SPR map of the difference between the reconstructed image and ground truth image.

Figure 4.5: SPR maps of the ground truth and reconstructed images.

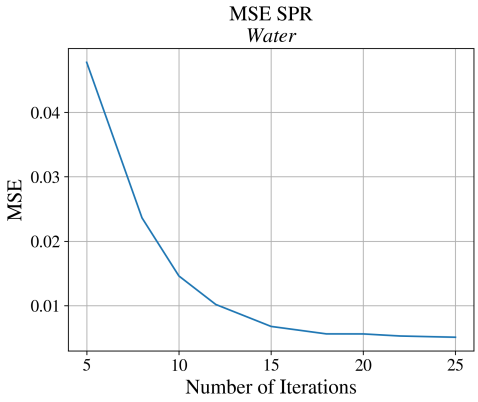
with an eroded mask over the bone region. This is only a 0.17% decrease compared to the real SPR value of bone, which is approximately 1.769. However, the SPR value of the water region experiences a 9.5% decrease compared to the real SPR value, from 1.0 to 0.905. This decrease is, of course, much larger than the decrease from the SPR value for bone and could be attributed to the large border that the water region has with all other materials in the phantom. The eroded mask deletes each first border pixel in the phantom; however, the large fluctuations in values around the border still have a large effect on the mean SPR value of water in the reconstruction. The SPR value of blood in the reconstruction undergoes approximately a 6.5% decrease from 1.023 to 0.957 and the SPR value of muscle an 8.5% decrease from 1.015 to 0.929, again calculated with eroded masks.

The course of the MSE in different material regions plotted against different iterations is shown in Figure 4.6. As expected, the MSE for the SPR value of bone decreases rapidly up to its lowest point, at 10 iterations, after which it increases slightly and stabilises there. This is different compared to the MSE for water, blood, and muscle, where the MSE decreases with the number of iterations. For blood and muscle, this value oscillates more, which is likely

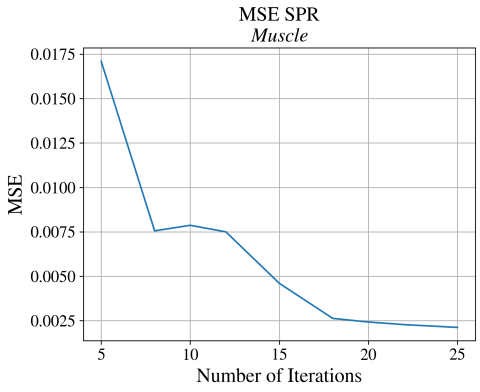
caused by the fact that there are considerably fewer pixels in these regions than in the water region, which results in less reliable data. It appears that the bone SPR values from the reconstruction deviates more from the actual value for higher iterations, while this does not happen for the SPR values of water, blood, and muscle. A possible explanation is that bone is more sensitive to model mismatch and noise amplification in higher iterations, since bone has the highest SPR value in the phantom. For higher iterations, these effects can accumulate more strongly in bone regions than in soft tissues, which results in overestimation of bone values.



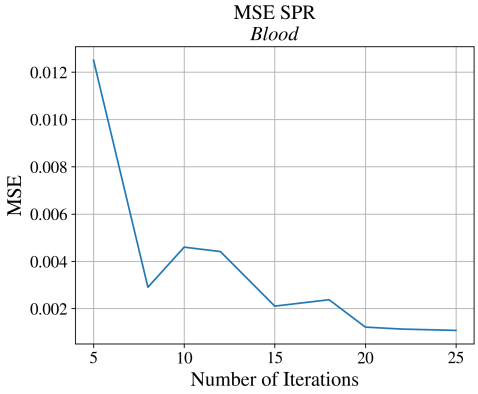
(a) CRC between the water and bone regions plotted against number of iterations.



(b) CRC between the water and muscle regions plotted against number of iterations.



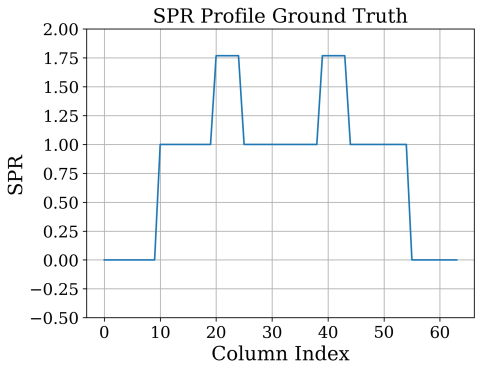
(c) CRC between the water and blood regions plotted against number of iterations.



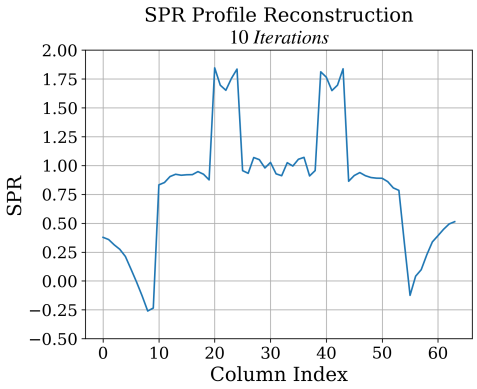
(d) CRC between the water and blood regions plotted against number of iterations.

Figure 4.6: CRC between different regions plotted against number of iterations.

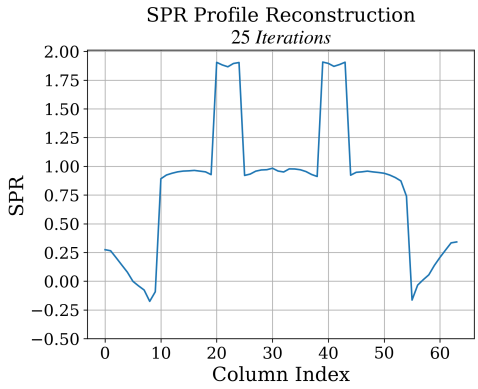
The profile plots of the SPR ground truth and SPR reconstructed image for 10 and 25 iterations are shown in Figure 4.7. The SPR is plotted against the column pixel index of the 64x64 slice of the SPR reconstruction and the ground truth shown in Figure 4.5. In Figure 4.7a, the profile of the ground truth SPR map, the SPR is clearly plotted for the background, the water region and the bone regions. In Figures 4.7b and 4.7c, the image artefacts present in the reconstructed image in Figure 4.5b are clearly visible. For example, in Figure 4.7b the cupping effect in the bone regions is clearly visible by the decrease in SPR in the centre of the region. Moreover, in both the 10 and 25 iterations profiles, the SPR values in the background have a non-zero positive value, after which they decrease to a negative number at the border of the phantom. Here, the profile of the reconstruction with 25 iterations exhibits a smaller fluctuation in values compared to the 10 iterations reconstruction. In the phantom, the values also fluctuate for both iterations. This happens mainly in each border, for example, when a



(a) Profile of the ground truth SPR map of a horizontal line through the middle of the image.



(b) Profile of the reconstruction SPR map of a horizontal line through the middle of the image.



(c) Profile of the reconstruction SPR map of a horizontal line through the middle of the image.

Figure 4.7: Profiles of the ground truth and reconstruction SPR maps.

transition occurs from the water region into the bone region. It is clear that the profile from the 25 iterations reconstruction exhibits fewer fluctuations in the SPR values compared to the 10 iterations. This is, of course, due to the increased number of iterations. The SPR value of water is also reconstructed more accurately in the 25 iterations than in the 10 iterations. However, note that the SPR value of bone in Figure 4.7c is higher than the ground truth SPR value of bone. This is a result from an overestimation due to the higher number of iterations. It is interesting to note that this effect is also clearly visible in Figures 4.6a and 4.6b. The bone SPR is overestimated for higher iterations, resulting in an increase in the MSE for the SPR for higher iterations, which can be seen in Figure 4.6a. However, the SPR value for water appears to contain less fluctuations, resulting in a lower MSE for higher iterations in Figure 4.6b.

4.5. Virtual Monochromatic Image Results

The VMI reconstructed at an energy of 80 keV can be seen in Figure 4.8. It is clear that the VMI exhibits noise in the background, as is usual in all reconstructed images. However, at first sight, they do not appear to be as present as, for example, in the SPR map reconstruction in Figure 4.5b.

To give an impression of the quality of the image, the attenuation coefficient of bone in this image is equal to 0.426 m^{-1}, calculated with an eroded mask over the bone region. This is approximately a 0.5% decrease compared to the real linear attenuation coefficient value of

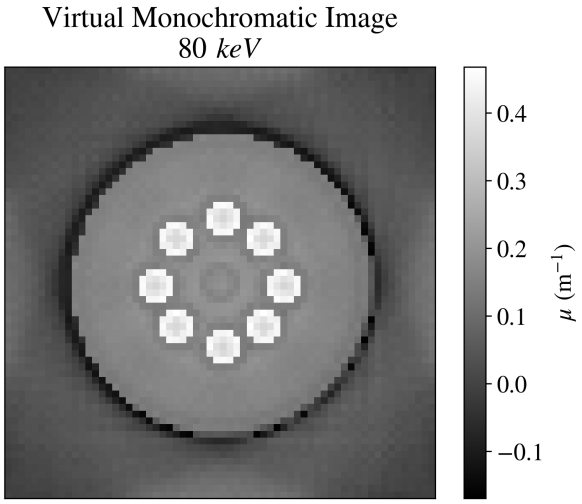
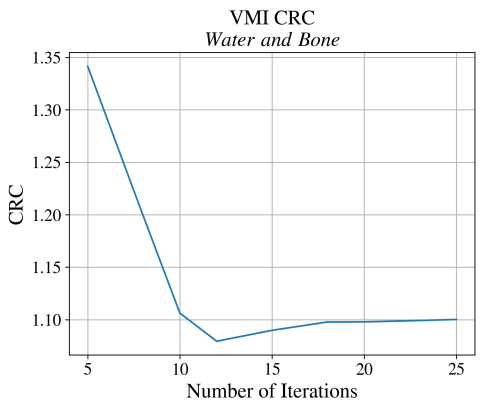
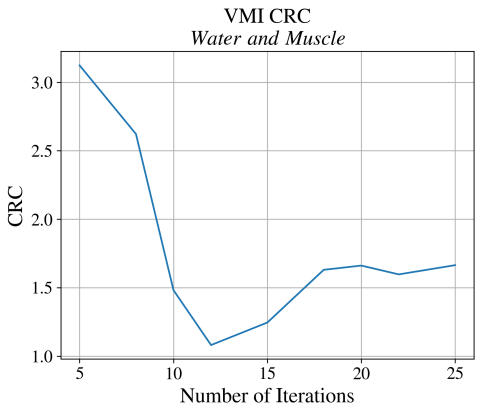


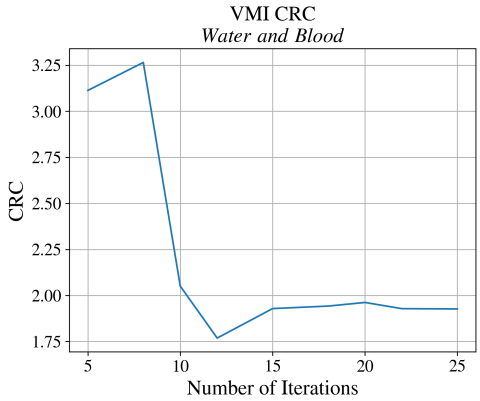
Figure 4.8: Virtual monochromatic image determined at 80 keV.



(a) CRC between the water and bone regions plotted against number of iterations.



(b) CRC between the water and muscle regions plotted against number of iterations.



(c) CRC between the water and blood regions plotted against number of iterations.

Figure 4.9: CRC between different regions plotted against number of iterations.

bone at 80 keV of 0.428 m^{-1}. This means that the VMI of the reconstructed image reflects the attenuation coefficient of bone well. However, the values of the attenuation coefficient for water, blood, and muscle exhibit larger errors. In the VMI, the linear attenuation coefficient

of water decreases with 10.3%, from 0.184 m^{-1} to 0.165 m^{-1}. Similarly, the blood attenuation coefficient decreases with 9.8% and the muscle attenuation coefficient with 11.0%. These larger errors could arise from the fact that the attenuation coefficient values of the three materials are similar. As a result, the reconstruction has difficulty distinguishing them, making their estimated values more sensitive to noise and artefacts. Specifically for the water region, there is a large amount of partial volume artefacts, and even after each first border pixel is deleted with the eroded mask, the effect is still present.

Again, just as in the SPR map, the VMI exhibits negative values directly around the phantom border. This is, of course, not physically feasible and should be investigated further.

The course of the CRC of the VMI for different iterations between different regions is shown in Figure 4.9. The CRC for each combination of materials is as expected. For low iterations, the CRC is high, indicating an image with poor contrast. It then descends rapidly onto a point of lowest CRC, after which it stabilises on a higher number. Just as with the MSE in the reconstructed images, the CRC is closest to one for 12 iterations. The lowest CRC is close to one between the regions of water and bone and between the regions of water and muscle, approximately 1.08 for both. This indicates that the contrast in the image is close to the contrast in the ground truth VMI, which is the desired result. However, the CRC between the water and blood regions at its lowest is approximately 1.77. This is a much larger value than the CRC between the other regions. It is remarkable that this large value arises for blood and not for muscle, as the properties of blood and muscle are similar. However, the cause of this increase in CRC for blood is currently unknown.

5

Discussion

It is important to continue research with the code provided in this thesis to make it applicable to more realistic data. As mentioned in Section 4.1, the `Python` code experiences a 127% increase in time compared to the equivalent `MATLAB` code for the same task. In further research, it is recommended to adapt the current code to a method that is more effective in `Python`, for example, with libraries that use faster languages such as `C++`. This could result in a much faster reconstruction programme, after which it could become easier to increase the inputted data by creating a larger phantom. Moreover, it was observed that both the `MATLAB` and `Python` codes have the opportunity to experience the presence of a singular matrix during the reconstruction process, which is then interrupted. This happens because in the `weidingerforwardmodel.py` file, the exponent of Equation (2.19) becomes positive and grows exponentially until it reaches infinite values. The variable z_k as defined in the Nesterov acceleration is used as the phantom in this file, and upon updating the variable for each iteration in the `Mechlem2017.py` file, it can become negative. This makes the exponent positive. However, this only happens with the 32x32x32 voxel phantom and not with the 64x64x64 voxel phantom. The reason behind this is unclear and should be investigated further to increase the stability of the reconstruction software.

To calculate the MSE in the SPR maps and the contrast ratio in the VMIs, eroded masks were used to perform the calculations to exclude pixels with higher probability of artefacts. However, this results in a high loss of information, as the bone, blood, and muscle regions lose almost 60% of their data points. This has a negative effect on the reliability of the calculations. In the future, border pixels also need to be reconstructed more accurately. Therefore, in future research, the border pixels should be taken into account to reflect the larger errors they exhibit.

The `Python` implementation of the provided `MATLAB` joint reconstruction optimisation algorithm still has room for improvement in producing exactly equivalent results. The MSE between the codes is equal to 0.0003 for the bone reconstructed image and 0.0007 for the water reconstructed image. These errors may still be attributed to systematic noise or rounding errors in the programming languages; however, this should be investigated.

To simplify the algorithm, the scatter that occurs in the detector is not taken into account in the calculations. However, in a future in which this algorithm is used for more realistic data, this additional noise should be incorporated into the algorithm. Another suggestion for future research entails comparing the results from the used joint reconstruction optimisation algorithm to results obtained from a non-joint reconstruction algorithm. This could indicate

the influence of the joint reconstruction technique much more clearly, which could lead to conclusions whether this method produces images with a higher quality than non-joint reconstruction algorithms. Moreover, the influence of the use of a PCCT can be investigated by applying this algorithm to a system with an EID.

6

Conclusion

Using the CBCT scanner integrated in proton radiation therapy units to update treatment plans directly before a treatment fraction would allow higher dose accuracy during treatment. However, this is currently limited by the low image quality of CBCT scanners. In this thesis, a spectral CBCT joint reconstruction algorithm is implemented to determine the quality of reconstructed images and their SPR maps. This algorithm was provided in a `MATLAB` code and converted to `Python`. The results of the algorithm show that the reconstructed images had the lowest MSE for 10-12 iterations. Larger iterations caused over-smoothing in the images, resulting in a lower quality. In addition, the CRC for the VMI is closest to one for 12 iterations. Lastly, the MSE for the SPR maps decreased for higher iterations in the water, blood, and muscle regions, while it increased for the bone region after its lowest point at 10 iterations. The converted `Python` algorithm can be improved by implementing libraries that use faster coding languages, such as `C++`. In this way, the algorithm may be able to withhold larger amounts of data. Momentarily, the `Python` implementation of the same algorithm provided similar images, but the reconstruction time increased significantly, which could be improved to allow for more realistic data. Moreover, to obtain a better understanding on the image quality of the produced reconstructed images, SPR maps, and VMI, the results should be compared to non-joint reconstruction algorithms. Additionally, this algorithm could be applied to work for EIDs, to compare whether the photon counting technique influences the image quality.

References

[1] Z. Chen et al. "Proton versus photon radiation therapy: A clinical review". In: *Frontiers in Oncology* 13 (2023). DOI: 10.3389/fonc.2023.1133909.

[2] W. Levin, H. Kooy, J. Loeffler, et al. "Proton Beam Therapy". In: *British Journal of Cancer* 93 (2005), pp. 849–854. DOI: 10.1038/sj.bjc.6602754.

[3] R. Mohan. "A Review of Proton Therapy – Current Status and Future Directions". In: *Precision Radiation Oncology* 6.2 (June 2022), pp. 164–176. DOI: 10.1002/pro6.1149.

[4] D. Leibold, D. R. Schaart, and M. C. Goorden. "Optimising proton stopping power ratio prediction with spectral cone-beam CT". In: *Physics in Medicine & Biology* 70.14 (2025), p. 145023. DOI: 10.1088/1361-6560/adebd6.

[5] Philipp Wohlfahrt and Christian Richter. "Status and innovations in pre-treatment CT imaging for proton therapy". In: *British Journal of Radiology* 93.1107 (Mar. 2020), p. 20190590. DOI: 10.1259/bjr.20190590.

[6] A. Bolsi et al. "Practice patterns of image guided particle therapy in Europe: A 2016 survey of the European Particle Therapy Network (EPTN)". In: *Radiotherapy and Oncology* 128.1 (2018), pp. 4–8. DOI: 10.1016/j.radonc.2018.03.017.

[7] D. Leibold et al. "Initial Results for Joint Reconstruction/Decomposition of Spectral Cone-Beam Computed Tomography Data". In: (2024).

[8] G. V. Toia et al. "Approaches, advantages, and challenges to photon counting detector and multi-energy CT". In: *Abdominal Radiology* 49 (2024), pp. 3251–3260. DOI: 10.1007/s00261-024-04357-x.

[9] P. Niu et al. "Improved Image Reconstruction Using Multi-Energy Information in Spectral Photon-Counting CT". In: *IEEE Access* 9 (May 2021), pp. 86102–86116. DOI: 10.1109/ACCESS.2021.3083505.

[10] K. Mechlem et al. "Joint Statistical Iterative Material Image Reconstruction for Spectral Computed Tomography Using a Semi-Empirical Forward Model". In: *IEEE Transactions on Medical Imaging* 37.1 (Jan. 2018), pp. 68–81. DOI: 10.1109/TMI.2017.2726687.

[11] C. Mory et al. "Comparison of five one-step reconstruction algorithms for spectral CT". In: *Physics in Medicine & Biology* 63.23 (Nov. 2018), p. 235001. DOI: 10.1088/1361-6560/aaeaf2.

[12] J. van Leenen. "Adapting a One-Step Reconstruction and Decomposition Algorithm for Cone-Beam CT". Supervisors: Marlies Goorden and David Leibold. Bachelor's Thesis. Delft University of Technology, Oct. 2023.

[13] H. Kekkonen. *Inverse Problems*. Lecture notes, Delft University of Technology. Accessed April 2025. 2025.

[14] J. Hadamard. *Sur les problèmes aux dérivées partielles et leur signification physique*. Princeton, NJ: Princeton University Bulletin, 1902.

[15] C. Hansen. *Truncated Singular Value Decomposition Solutions to Discrete Ill-Posed Problems with Ill-Determined Numerical Rank*. CAM Report 87-3. Department of Mathematics, University of California, Los Angeles, 1987. URL: https://ww3.math.ucla.edu/camreport/cam87-03.pdf.

[16] F. Bijma, M. Jonker, and A. van der Vaart. *An Introduction to Mathematical Statistics*. Amsterdam University Press, 2017. ISBN: 9789462985100.

[17] Johnathan M. Bardsley and John Goldes. "Regularization parameter selection methods for ill-posed Poisson maximum likelihood estimation". In: *Inverse Problems* 25.9 (2009), p. 095005. DOI: 10.1088/0266-5611/25/9/095005.

[18] H. Ohlsson. "Regularization for Sparseness and Smoothness: Applications in System Identification and Signal Processing". Ph.D. thesis. Linköping, Sweden: Linköping University, 2010. URL: https://www.diva-portal.org/smash/get/diva2:360033/FULLTEXT02.pdf.

[19] M. Schmidt. *Least Squares Optimization with L1-Norm Regularization*. Tech. rep. University of British Columbia, 2005.

[20] B. Alleche, V. D. Rădulescu, and M. Sebaoui. "The Tikhonov Regularization for Equilibrium Problems and Applications to Quasi-Hemivariational Inequalities". In: *Optimization Letters* 9.3 (July 2015), pp. 483–503. DOI: 10.1007/s11590-014-0765-3.

[21] R. Tibshirani. "Regression Shrinkage and Selection Via the Lasso". In: *Journal of the Royal Statistical Society: Series B (Methodological)* 58.1 (1996), pp. 267–288. DOI: 10.1111/j.2517-6161.1996.tb02080.x.

[22] H. Song. *L1 and L2 Regularization*. https://hayoonsong.github.io/study/2022-07-05-L1_L2/. Accessed: 2025-09-12. 2022.

[23] Mayo Clinic Staff. *CT Scan*. https://www.mayoclinic.org/tests-procedures/ct-scan/about/pac-20393675. Accessed: April 21, 2025. 2024.

[24] S. Hermena and M. Young. *CT-scan Image Production Procedures*. National Library of Medicine (US), National Center for Biotechnology Information. Last updated: August 8, 2023. 2023. URL: https://www.ncbi.nlm.nih.gov/books/NBK574548/.

[25] S. Abdulla. *CT Equipment — FRCR Physics Notes*. Last updated: 13 June 2021. Radiology Cafe. 2021. URL: https://www.radiologycafe.com/frcr-physics-notes/ct-imaging/ct-equipment/ (visited on 05/30/2024).

[26] J. T. Bushberg et al. *The Essential Physics of Medical Imaging*. 3rd. Philadelphia, PA: Lippincott Williams & Wilkins, 2011. ISBN: 978-0781780575.

[27] I. Oshina and J. Spigulis. "Beer–Lambert law for optical tissue diagnostics: current state of the art and the main limitations". In: *Journal of Biomedical Optics* 26.10 (2021), p. 100901. DOI: 10.1117/1.JBO.26.10.100901.

[28] M. Baur et al. "Correction of beam hardening in X-ray radiograms". In: *Review of Scientific Instruments* 90.2 (Feb. 2019), p. 025108. DOI: 10.1063/1.5080540.

[29] Y. Zhang et al. "Spectral CT Reconstruction with Image Sparsity and Spectral Mean". In: *IEEE Transactions on Computational Imaging* 2.4 (Dec. 2016), pp. 510–523. DOI: 10.1109/TCI.2016.2609414.

[30] S. C. van der Bie et al. "Photon-counting CT: Review of initial clinical results". In: *European Journal of Radiology* 157 (2023), p. 110495. DOI: 10.1016/j.ejrad.2022.110495.

[31] D. F. Jackson and D. J. Hawkes. "X-ray attenuation coefficients of elements and mixtures". In: *Physics Reports* 70.3 (1981), pp. 169–233. DOI: 10.1016/0370-1573(81)90014-4.

[32] D. Jumanazarov. "Spectrally Joint Reconstruction and Material Classification from Multi-Energy CT". PhD Thesis. Department of Physics, DTU Physics: Technical University of Denmark, Oct. 2021.

[33] R. E. Alvarez and A. Macovski. "Energy-selective Reconstructions in X-ray Computerized Tomography". In: *Physics in Medicine and Biology* 21.5 (1976), pp. 733–744. DOI: 10.1088/0031-9155/21/5/002.

[34] A. So and S. Nicolaou. "Spectral Computed Tomography: Fundamental Principles and Recent Developments". In: *Korean Journal of Radiology* 22.1 (2021), pp. 86–96. DOI: 10.3348/kjr.2020.0144.

[35] *Physics in Medicine Lectures Week 2 CT Imaging*. Jan. 2012. URL: https://koukalaka.wordpress.com/tag/attenuation-coefficients/ (visited on 06/01/2025).

[36] Nature Portfolio. *The future of CT scans is clearer thanks to new tech*. 2023. URL: https://www.nature.com/articles/d42473-023-00450-9 (visited on 05/30/2024).

[37] R. H. Pratt, A. Ron, and H. K. Tseng. "Atomic Photoelectric Effect Above 10 keV". In: *Reviews of Modern Physics* 45.2 (Apr. 1973), pp. 273–325. DOI: 10.1103/RevModPhys.45.273.

[38] G. V. Toia et al. "Dual-Energy Computed Tomography in Body Imaging". In: *Seminars in Roentgenology* 53.2 (2018), pp. 132–146. DOI: 10.1053/j.ro.2018.02.004.

[39] T. Weidinger et al. "Polychromatic Iterative Statistical Material Image Reconstruction for Photon-Counting Computed Tomography". In: *International Journal of Biomedical Imaging* 2016 (2016), p. 5871604. DOI: 10.1155/2016/5871604.

[40] K. Taguchi et al. "Imaging Performance of a $LaBr_3$:Ce Scintillation Detector for Photon Counting X-ray Computed Tomography: Simulation Study". In: *Medical Physics* 52.1 (Jan. 2025), pp. 158–170. DOI: 10.1002/mp.17436.

[41] H. Jung. "Basic Physical Principles and Clinical Applications of Computed Tomography". In: *Progress in Medical Physics* 32.1 (2021), pp. 1–17. DOI: 10.14316/pmp.2021.32.1.1.

[42] L. Lechuga and G. A. Weidlich. "Cone Beam CT vs. Fan Beam CT: A Comparison of Image Quality and Dose Delivered Between Two Differing CT Imaging Modalities". In: *Cureus* 8.9 (2016), e778. DOI: 10.7759/cureus.778.

[43] A. M. Salyapongse et al. "Spatial Resolution Fidelity Comparison Between Energy Integrating and Deep Silicon Photon Counting CT: Implications for Pulmonary Imaging". In: *Journal of Thoracic Imaging* 39.6 (May 2024), pp. 344–350. DOI: 10.1097/RTI.0000000000000788.

[44] A. Takase. *How Does CT Reconstruction Work?* Rigaku Corporation. 2023. URL: https://rigaku.com/products/imaging-ndt/x-ray-ct/learning/blog/how-does-ct-reconstruction-work (visited on 05/30/2024).

[45] J. Toivanen et al. "Joint reconstruction in low dose multi-energy CT". In: *Inverse Problems and Imaging* 14.4 (2020), pp. 607–629. DOI: 10.3934/ipi.2020028.

[46] J. Greffier et al. "Spectral CT imaging: Technical principles of dual-energy CT and multi-energy photon-counting CT". In: *Diagnostic and Interventional Imaging* 104.4 (Apr. 2023), pp. 167–177. DOI: 10.1016/j.diii.2022.11.003.

[47] S. Z. Adam et al. “Spectral CT of the Abdomen: Where Are We Now?” In: *Insights into Imaging* 12.1 (2021), p. 138. DOI: 10.1186/s13244-021-01082-7.

[48] T. Yuan, Z. Zhan, and C. Qian. “New Frontiers in Proton Therapy: Applications in Cancers”. In: *Cancer Communications* 39.1 (Dec. 2019), pp. 1–7. DOI: 10.1186/s40880-019-0407-3.

[49] Joint Institute for Nuclear Research. *JINR scientists invented effective way of medical proton beam focusing*. Mar. 2022. URL: https://www.jinr.ru/posts/jinr-scientists-invented-effective-way-of-medical-proton-beam-focusing/ (visited on 09/01/2025).

[50] S. Zhang et al. “Impact of joint statistical dual-energy CT reconstruction of proton stopping power images: Comparison to image- and sinogram-domain material decomposition approaches”. In: *Medical Physics* 45.5 (2018), pp. 2129–2142. DOI: 10.1002/mp.12875.

[51] N. Pischom et al. “Stopping power ratio databases for proton therapy dose calculation”. In: *Journal of Physics: Conference Series* 1505.1 (2020), p. 012012. DOI: 10.1088/1742-6596/1505/1/012012.

[52] A. Campos. *Filtered Back Projection*. https://radiopaedia.org/articles/filtered-back-projection-1. Last revised on 21 May 2024, Accessed on 2 September 2025. 2024.

[53] S. Ehn et al. “Basis Material Decomposition in Spectral CT Using a Semi-Empirical, Polychromatic Adaption of the Beer–Lambert Model”. In: *Physics in Medicine & Biology* 62.1 (Jan. 2017), N1–N17. DOI: 10.1088/1361-6560/aa4e5c.

[54] H. Erdogan and J. A. Fessler. “Ordered subsets algorithms for transmission tomography”. In: *Physics in Medicine and Biology* 44.11 (1999), pp. 2835–2851. DOI: 10.1088/0031-9155/44/11/311.

[55] D. Kim, S. Ramani, and J. A. Fessler. “Combining Ordered Subsets and Momentum for Accelerated X-Ray CT Image Reconstruction”. In: *IEEE Transactions on Medical Imaging* 34.1 (2015), pp. 167–178. DOI: 10.1109/TMI.2014.2350962.

[56] L. Yu et al. “Virtual monochromatic imaging in dual-source dual-energy CT: radiation dose and image quality”. In: *Medical Physics* 38.12 (Dec. 2011), pp. 6371–6379. DOI: 10.1118/1.3658568.

[57] A. Takase. *What Is Beam Hardening in CT?* Accessed: 2025-09-14. 2022. URL: https://www.rigaku.com/en/resources/blog/what-is-beam-hardening-in-ct.

[58] University of Texas Computed Tomography (UTCT) Laboratory. *Artifacts and Partial-Volume Effects*. Accessed: 2025-09-14. 2025. URL: https://www.ctlab.geo.utexas.edu/about-ct/artifacts-and-partial-volume-effects/.

[59] National Institute of Standards and Technology (NIST). *X-Ray Mass Attenuation Coefficients: NIST Standard Reference Database*. https://physics.nist.gov/PhysRefData/XrayMassCoef/. 2024. (Visited on 05/30/2024).

[60] National Institute of Standards and Technology (NIST). *X-Ray Mass Attenuation Coefficients – Table 4*. https://physics.nist.gov/PhysRefData/XrayMassCoef/tab4.html. Accessed: April 21, 2025. 2025.

[61] H. R. Moradi et al. “Around Jensen’s Inequality for Strongly Convex Functions”. In: *Aequationes Mathematicae* 92.1 (2018), pp. 25–37. DOI: 10.1007/s00010-017-0485-8.

A

Mathematical derivations

A.1. Jensen's inequality

A very effective way to remove a summation term outside of a function is to use Jensen's inequality. If a function f is convex in the interval $[m, M]$, Jensen's inequality gives the following [61]:

$$f\left(\sum_{i=1}^{n} p_i x_i\right) \leq \sum_{i=1}^{n} p_i f(x_i)\,, \tag{A.1}$$

for all $x_i \in [m, M]$ when all $p_i \in [0, 1]$ for $i = 1, \ldots, n$ and $\sum_{i=1}^{n} p_i = 1$.

A.2. Proof of Surrogate Conditions for the Surrogate Functions of the Log-likelihood Estimator

In this section, a proof is given that the surrogate functions of the log-likelihood estimator calculated in Section 2.5.1 abide by the following prerequisites:

$$Q(\vec{\alpha}^{(n)}; \vec{\alpha}^{(n)}) = \theta(\vec{\alpha}^{(n)}) \tag{A.2a}$$

$$\left.\frac{\partial Q(\vec{\alpha}; \vec{\alpha}^{(n)})}{\partial \alpha_j^b}\right|_{\vec{\alpha}=\vec{\alpha}^{(n)}} = \left.\frac{\partial \theta(\vec{\alpha})}{\partial \alpha_j^b}\right|_{\vec{\alpha}=\vec{\alpha}^{(n)}} \tag{A.2b}$$

$$Q(\vec{\alpha}; \vec{\alpha}^{(n)}) \geq \theta(\vec{\alpha})\,. \tag{A.2c}$$

As we calculate the surrogate functions of the log-likelihood function, the log-likelihood part of the cost function is defined as follows:

$$\theta_{\mathcal{L}}(\vec{\alpha}) = \sum_{i=1}^{P} \hat{Y}_i - Y_i \cdot \log(\hat{Y}_i) = \sum_{i=1}^{P} h_i(\hat{Y}_i(\vec{\alpha}))\,. \tag{A.3}$$

First, to be able to calculate the derivative of the cost function, the derivative of the function $\hat{Y}_i(\vec{\alpha})$ is calculated as follows:

$$\begin{aligned}\frac{\partial \hat{Y}_i(\vec{\alpha})}{\partial \alpha_j^b} &= \frac{\partial}{\partial \alpha_j^b}\Big[\sum_{s=1}^{S} N_i^s e^{-\sum_{b=1}^{B} f^b(E)\sum_{j=1}^{N} a_{ij}\alpha_j^b}\Big] = \sum_{s=1}^{S} N_i^s \frac{\partial}{\partial \alpha_j^b}\Big[e^{-\sum_{b=1}^{B} f^b(E)\sum_{j=1}^{N} a_{ij}\alpha_j^b}\Big] \\ &= \sum_{s=1}^{S} N_i^s e^{-\sum_{b=1}^{B} f^b(E)\sum_{j=1}^{N} a_{ij}\alpha_j^b} \frac{\partial}{\partial \alpha_j^b}\Big[-\sum_{b'=1}^{B} f^{b'}(E)\sum_{j'=1}^{N} a_{ij'}\alpha_{j'}^{b'}\Big] \\ &= -\sum_{s=1}^{S} N_i^s f^b(E) a_{ij} \cdot e^{-\sum_{b=1}^{B} f^b(E)\sum_{j=1}^{N} a_{ij}\alpha_j^b} = -\hat{Y}_i(\vec{\alpha}) f^b(E) a_{ij} \,.\end{aligned}$$

In the last equality, we used that all variables in the summations over $b' = 1, 2$ and $j' = 1, \ldots, N$ are constants, except for $b = b'$ and $j = j'$, due to the partial derivative with respect to α_j^b. Then, it is possible to calculate the derivative of the cost function $\theta(\vec{\alpha})$ as follows:

$$\begin{aligned}\frac{\partial \theta_{\mathcal{L}}(\vec{\alpha})}{\partial \alpha_j^b} &= \frac{\partial}{\partial \alpha_j^b}\Big[\sum_{i=1}^{P} h_i(\hat{Y}_i(\vec{\alpha}))\Big] = \sum_{i=1}^{P} \frac{\partial}{\partial \alpha_j^b}\Big[\hat{Y}_i(\vec{\alpha}) - Y_i \log(\hat{Y}_i(\vec{\alpha}))\Big] \\ &= \sum_{i=1}^{P}\Big(1 - \frac{Y_i}{\hat{Y}_i(\vec{\alpha})}\Big)\cdot \frac{\partial \hat{Y}_i(\vec{\alpha})}{\partial \alpha_j^b} = -\sum_{i=1}^{P}\Big(\hat{Y}_i(\vec{\alpha}) - Y_i\Big) f^b(E) a_{ij} \,, \qquad \text{(A.4)}\end{aligned}$$

where we used the result from the derivative of $\hat{Y}_i(\vec{\alpha})$ from Equation (A.4). After substituting the variable $\vec{\alpha}^{(n)}$ into this expression, we obtain the following:

$$\left.\frac{\partial \theta_{\mathcal{L}}(\vec{\alpha})}{\partial \alpha_j^b}\right|_{\vec{\alpha}=\vec{\alpha}^{(n)}} = -\sum_{i=1}^{P}\Big(\hat{Y}_i(\vec{\alpha}^{(n)}) - Y_i\Big) f^b(E) a_{ij} \,. \qquad \text{(A.5)}$$

The first surrogate function for the log-likelihood estimator is given as follows:

$$Q_1(\vec{\alpha}; \vec{\alpha}^{(n)}) \equiv \sum_{i=1}^{P}\sum_{s=1}^{S} \frac{N_i^s}{\beta_i^{s,(n)}} h_i(t_i^s(\vec{\alpha})\beta_i^{s,(n)}) \,. \qquad \text{(A.6)}$$

Conditions (A.2a) - (A.2c) will be checked for the first surrogate function.

1. The first condition is satisfied by the following:

$$\begin{aligned}Q_1(\vec{\alpha}^{(n)}; \vec{\alpha}^{(n)}) &= \sum_{i=1}^{P}\sum_{s=1}^{S} \frac{N_i^s}{\beta_i^{s,(n)}} h_i(t_i^s(\vec{\alpha}^{(n)})\beta_i^{s,(n)}) = \sum_{i=1}^{P}\sum_{s=1}^{S} \frac{N_i^s}{\beta_i^{s,(n)}} h_i(t_i^s(\vec{\alpha})\frac{\hat{Y}_i(\vec{\alpha}^{(n)})}{t_i^s(\vec{\alpha}^{(n)})}) \\ &= \sum_{i=1}^{P}\Big(\sum_{s=1}^{S} \frac{N_i^s}{\beta_i^{s,(n)}}\Big) h_i\big(\hat{Y}_i(\vec{\alpha}^{(n)})\big) = \sum_{i=1}^{P} h_i(\hat{Y}_i(\vec{\alpha}^{(n)})) \equiv \theta(\vec{\alpha}^{(n)}) \,.\end{aligned}$$

   In this derivation, we see that $\sum_{s=1}^{S} \frac{N_i^s}{\beta_i^{s,(n)}} = 1$, as derived in the text. It was possible to take this summation separately and equal it to one, as $h_i\big(\hat{Y}_i(\vec{\alpha}^{(n)})\big)$ lost its s dependency.

2. The second condition is confirmed as follows:

$$\begin{aligned}\frac{\partial Q_1}{\partial \alpha_j^b} &= \frac{\partial}{\partial \alpha_j^b}\Big[\sum_{i=1}^{P}\sum_{s=1}^{S} \frac{N_i^s}{\beta_i^{s,(n)}} h_i\big(t_i^s(\vec{\alpha})\beta_i^{s,(n)}\big)\Big] = \sum_{i=1}^{P}\Big(\sum_{s=1}^{S} \frac{N_i^s}{\beta_i^{s,(n)}}\Big)\frac{\partial}{\partial \alpha_j^b}\Big[h_i\big(\hat{Y}_i(\vec{\alpha})\big)\Big] \\ &= \sum_{i=1}^{P} \frac{\partial}{\partial \alpha_j^b}\Big[\hat{Y}_i(\vec{\alpha}) - Y_i \log(\hat{Y}_i(\vec{\alpha}))\Big] = -\sum_{i=1}^{P}\Big(\hat{Y}_i(\vec{\alpha}) - Y_i\Big) f^b(E) a_{ij}\end{aligned}$$

Note that $\beta_i^{s,(n)} = \frac{\hat{Y}_i(\vec{\alpha}^{(n)})}{t_i^s(\vec{\alpha}^{(n)})}$ is independent of the differentiation variable α_j^b, since $\vec{\alpha}^{(n)}$ is the iteration variable. Moreover, we used the same trick as in the derivation of the first surrogate condition that the sum over s can be taken separately, which can then be equalled to one. To finalise the proof, we need to fill in the variable $\vec{\alpha} = \vec{\alpha}^{(n)}$, to obtain

$$\left.\frac{\partial Q(\vec{\alpha})}{\partial \alpha_j^b}\right|_{\vec{\alpha}=\vec{\alpha}^{(n)}} = -\sum_{i=1}^{P}\left(\hat{Y}_i(\vec{\alpha}^{(n)}) - Y_i\right) f^b(E) a_{ij} = \left.\frac{\partial \theta(\vec{\alpha})}{\partial \alpha_j^b}\right|_{\vec{\alpha}=\vec{\alpha}^{(n)}}$$

3. The third condition is satisfied by Jensen's inequality, as is explained in Section 2.5.1 by the following equation:

$$\begin{aligned}\theta(\vec{\alpha}) &= \sum_{i=1}^{P} h_i(\hat{Y}_i) = \sum_{i=1}^{P} h_i\left(\sum_{s=1}^{S} \frac{N_i^s}{\beta_i^{s,(n)}} t_i^s(\vec{\alpha})\beta_i^{s,(n)}\right) \\ &\leq \sum_{i=1}^{P}\sum_{s=1}^{S} \frac{N_i^s}{\beta_i^{s,(n)}} h_i(t_i^s(\vec{\alpha})\beta_i^{s,(n)}) \equiv Q_1(\vec{\alpha};\vec{\alpha}^{(n)})\,.\end{aligned}$$

The inequality could be used by the properties of $\frac{N_i^s}{\beta_i^{s,(n)}}$ as given in the main text.

This concludes the proof that the calculated surrogate function satisfies the prerequisites given in Equations (A.2a) through (A.2c). This makes the function $Q_1(\vec{\alpha};\vec{\alpha}^{(n)})$ a satisfactory surrogate function for the log-likelihood estimator.

Let us continue to the inspection of whether the second surrogate function for the log-likelihood part of the cost function satisfies conditions (A.2a) - (A.2c). As derived in the main text, the second surrogate function is given as:

$$Q_2(\vec{\alpha},\vec{\alpha}^{(n)}) = \sum_{i=1}^{P}\sum_{s=1}^{S} \frac{N_i^s}{\beta_i^{s,(n)}} q_i^{(n)}(l_i^{s1}, l_i^{s2})\ , \tag{A.7}$$

with

$$\begin{aligned}q_i^{(n)}(l_i^{s1}, l_i^{s2}) = g_i^{(n)}(l_i^{s1,(n)}, l_i^{s2,(n)}) &+ \sum_{b=1}^{2} \left.\frac{\partial g_i^{(n)}(l_i^{s1}, l_i^{s2})}{\partial l_i^{sb}}\right|_{l_i^{sb}=l_i^{sb,(n)}} \left(l_i^{sb} - l_i^{sb,(n)}\right) \\ &+ \frac{1}{2}\sum_{k=1}^{2}\sum_{m=1}^{2} T_i^{s,(n)}\left(l_i^{sk} - l_i^{sk,(n)}\right)\left(l_i^{sm} - l_i^{sm,(n)}\right)\ ,\end{aligned} \tag{A.8}$$

where $g_i^{(n)}(l_i^{s1}, l_i^{s2}) = h_i^s(e^{-\sum_{b=1}^{2} l_i^{sb}\beta_i^{s,(n)}})$. Conditions (A.2a) - (A.2c) will be checked for the second surrogate function.

1. The first condition will be proved by substituting the variable $\vec{\alpha}^{(n)}$ into the second surrogate function as given in Equation (A.7):

$$
\begin{aligned}
&Q_2(\vec{\alpha}^{(n)}, \vec{\alpha}^{(n)}) \equiv \sum_{i=1}^{P}\sum_{s=1}^{S} \frac{N_i^s}{\beta_i^{s,(n)}} q_i^{(n)}(l_i^{s1,(n)}, l_i^{s2,(n)}) \\
&= \sum_{i=1}^{P}\sum_{s=1}^{S} \frac{N_i^s}{\beta_i^{s,(n)}} \Bigg[g_i^{(n)}(l_i^{s1,(n)}, l_i^{s2,(n)}) + \sum_{b=1}^{2} \left.\frac{\partial g_i^{(n)}(l_i^{s1,(n)}, l_i^{s2,(n)})}{\partial l_i^{sb}}\right|_{l_i^{sb}=l_i^{sb,(n)}} \left(l_i^{sb,(n)} - l_i^{sb,(n)}\right) \\
&\qquad + \frac{1}{2}\sum_{k=1}^{2}\sum_{m=1}^{2} T_i^{s,(n)} \left(l_i^{sk,(n)} - l_i^{sk,(n)}\right)\left(l_i^{sm,(n)} - l_i^{sm,(n)}\right) \Bigg] \\
&= \sum_{i=1}^{P}\sum_{s=1}^{S} \frac{N_i^s}{\beta_i^{s,(n)}} g_i^{(n)}(l_i^{s1,(n)}, l_i^{s2,(n)}) = \sum_{i=1}^{P}\sum_{s=1}^{S} \frac{N_i^s}{\beta_i^{s,(n)}} h_i\left(t_i^s(\vec{\alpha}^{(n)})\beta_i^{s,(n)}\right) \\
&= \sum_{i=1}^{P}\sum_{s=1}^{S} \frac{N_i^s}{\beta_i^{s,(n)}} h_i\left(\hat{Y}_i(\vec{\alpha}^{(n)})\right) \equiv \theta_{\mathcal{L}}(\vec{\alpha}^{(n)}) \,,
\end{aligned} \tag{A.9}
$$

   where it is used that $t_i^s(\vec{\alpha}^{(n)})\beta_i^{s,(n)} = \hat{Y}_i(\vec{\alpha}^{(n)})$. This proofs the first condition.

2. As taking the derivative of the entire function $q_i^{(n)}(l_i^{s1,(n)}, l_i^{s2,(n)})$ is very complicated, let us break it down into smaller problems. We begin by taking the derivative of the function $l_i^{sb}(\vec{\alpha}^b) = \sum_{j=1}^{N} a_{ij}\alpha_j^b f^b(E)$:

$$
\frac{\partial l_i^{sb}(\vec{\alpha}^b)}{\partial \alpha_j^b} = \frac{\partial}{\partial \alpha_j^b}\left(\sum_{j=1}^{N} a_{ij}\alpha_j^b f^b(E)\right) = a_{ij} f^b(E) \,. \tag{A.10}
$$

   Here, the sum is taken out because we are differentiating over α_j^b for a specific j. Then, we can see that there are a few terms in the function $q_i^{(n)}(l_i^{s1,(n)}, l_i^{s2,(n)})$ that disappear when the variable $\vec{\alpha}^{(n)}$ is filled in for $\vec{\alpha}$ after having taken the derivative. The third term of the function in Equation (A.8) can be completely omitted by the product rule. After taking the derivative of this term, at least one of the functions $(l_i^{sk} - l_i^{sk,(n)})$ will remain. Therefore, when $\vec{\alpha}^{(n)}$ is substituted, all terms of the derivative will become zero. The same happens for one of the terms after taking the product rule of the second term in Equation (A.8). Additionally, the first term is independent of the variable α_j^b, which then also becomes zero after taking the derivative. We are left with the following:

$$
\begin{aligned}
\left.\frac{\partial q_i^{(n)}(l_i^{s1}, l_i^{s2})}{\partial \alpha_j^b}\right|_{\vec{\alpha}=\vec{\alpha}^{(n)}} &= \frac{\partial}{\partial \alpha_j^b}\Bigg[g_i^{(n)}(l_i^{s1,(n)}, l_i^{s2,(n)}) + \sum_{b=1}^{2} \left.\frac{\partial g_i^{(n)}(l_i^{s1}, l_i^{s2})}{\partial l_i^{sb}}\right|_{l_i^{sb}=l_i^{sb,(n)}} \left(l_i^{sb} - l_i^{sb,(n)}\right) \\
&\qquad + \frac{1}{2}\sum_{k=1}^{2}\sum_{m=1}^{2} T_i^{s,(n)} \left(l_i^{sk} - l_i^{sk,(n)}\right)\left(l_i^{sm} - l_i^{sm,(n)}\right) \Bigg]\Bigg|_{\vec{\alpha}=\vec{\alpha}^{(n)}} \\
&= \left[\sum_{b=1}^{2} \left.\frac{\partial g_i^{(n)}(l_i^{s1}, l_i^{s2})}{\partial l_i^{sb}}\right|_{l_i^{sb}=l_i^{sb,(n)}} \frac{\partial l_i^{sb}}{\partial \alpha_j^b}\right]\Bigg|_{\vec{\alpha}=\vec{\alpha}^{(n)}} \,,
\end{aligned} \tag{A.11}
$$

   where it was used that the variable $l_i^{sb,(n)}$ is independent of the variable α_j^b and therefore vanishes. Before continuing with the entire expression, we calculate the following part

separately:

$$\begin{aligned}
&\sum_{b=1}^{2} \frac{\partial g_i^{(n)}(l_i^{s1}, l_i^{s2})}{\partial l_i^{sb}}\bigg|_{l_i^{sb}=l_i^{sb,(n)}} = \sum_{b=1}^{2} \frac{\partial}{\partial l_i^{sb}} \left[h_i^s\Big(e^{-\sum_{b=1}^{2} l_i^{sb}\beta_i^{s,(n)}}\Big)\right]\bigg|_{l_i^{sb}=l_i^{sb,(n)}} \\
&= \sum_{b=1}^{2} \frac{\partial}{\partial l_i^{sb}} \left[e^{-\sum_{b=1}^{2} l_i^{sb}\beta_i^{s,(n)}} - Y_i \log\Big(e^{-\sum_{b=1}^{2} l_i^{sb}\beta_i^{s,(n)}}\Big)\right]\bigg|_{l_i^{sb}=l_i^{sb,(n)}} \\
&= \sum_{b=1}^{2} \frac{\partial}{\partial l_i^{sb}} \left[e^{-\sum_{b=1}^{2} l_i^{sb}\beta_i^{s,(n)}} - Y_i\Big(-\sum_{b=1}^{2} l_i^{sb} + \log\beta_i^{s,(n)}\Big)\right]\bigg|_{l_i^{sb}=l_i^{sb,(n)}} \\
&\Big(-e^{-\sum_{b=1}^{2} l_i^{sb}\beta_i^{s,(n)}} + Y_i\Big)\bigg|_{l_i^{sb}=l_i^{sb,(n)}} = Y_i - \hat{Y}_i(\vec{\alpha}^{(n)})\,,
\end{aligned} \tag{A.12}$$

where, again, the sum over the variable b disappears as we are differentiating α_j^b for a specific b. Substituting this expression into the derivative of $q_i^{(n)}(l_i^{s1}, l_i^{s2})$, we obtain the following expression:

$$\begin{aligned}
\frac{\partial q_i^{(n)}(l_i^{s1}, l_i^{s2})}{\partial \alpha_j^b}\bigg|_{\vec{\alpha}=\vec{\alpha}^{(n)}} &= \left[\sum_{b=1}^{2} \frac{\partial g_i^{(n)}(l_i^{s1}, l_i^{s2})}{\partial l_i^{sb}}\bigg|_{l_i^{sb}=l_i^{sb,(n)}} \frac{\partial l_i^{sb}}{\partial \alpha_j^b}\right]\bigg|_{\vec{\alpha}=\vec{\alpha}^{(n)}} \\
&= \Big[\big(Y_i - \hat{Y}_i(\vec{\alpha}^{(n)})\big) a_{ij} f^b(E)\Big]\bigg|_{\vec{\alpha}=\vec{\alpha}^{(n)}} = \big(Y_i - \hat{Y}_i(\vec{\alpha}^{(n)})\big) a_{ij} f^b(E)\,.
\end{aligned} \tag{A.13}$$

Substituting this expression back into the derivative of $Q_2(\vec{\alpha}^{(n)}, \vec{\alpha}^{(n)})$ we obtain the following result:

$$\begin{aligned}
\frac{\partial Q_2(\vec{\alpha}; \vec{\alpha}^{(n)})}{\partial \alpha_j^b}\bigg|_{\vec{\alpha}=\vec{\alpha}^{(n)}} &= \sum_{i=1}^{P}\sum_{s=1}^{S} \frac{N_i^s}{\beta_i^{s,(n)}} \frac{\partial q_i^{(n)}(l_i^{s1}, l_i^{s2})}{\partial \alpha_j^b}\bigg|_{\vec{\alpha}=\vec{\alpha}^{(n)}} \\
&= \sum_{i=1}^{P} \left(\sum_{s=1}^{S} \frac{N_i^s}{\beta_i^{s,(n)}}\right) \big(Y_i - \hat{Y}_i(\vec{\alpha}^{(n)})\big) a_{ij} f^b(E) \\
&= \sum_{i=1}^{P} \big(Y_i - \hat{Y}_i(\vec{\alpha}^{(n)})\big) a_{ij} f^b(E) \equiv \frac{\partial \theta_{\mathcal{L}}(\vec{\alpha})}{\partial \alpha_j^b}\bigg|_{\vec{\alpha}=\vec{\alpha}^{(n)}}\,.
\end{aligned} \tag{A.14}$$

The final expression proves the second surrogate condition given in (A.2b) for the function $Q_2(\vec{\alpha}^{(n)}, \vec{\alpha}^{(n)})$.

3. To prove the third condition, the convexity of $h_i(\hat{Y}(\vec{\alpha}))$ is used. As $Y_i > 0$, it is known that $h_i(\hat{Y}(\vec{\alpha}))$ is a convex function because of the following:

$$\frac{\partial^2 h_i(\hat{Y}_i)}{\partial(\hat{Y}_i)^2} = \frac{Y_i}{(\hat{Y}_i)^2} > 0\,. \tag{A.15}$$

With this result, we can say that the Taylor polynomial $q_i^{(n)}(l_i^{s1}, l_i^{s2})$ is a global under estimator for the function $g_i^{(n)}(l_i^{s1}, l_i^{s2})$. Then, we can say that $q_i^{(n)}(l_i^{s1}, l_i^{s2}) \geq g_i^{(n)}(l_i^{s1}, l_i^{s2})$.

Substituting this expression into the function $Q_2(\vec{\alpha}, \vec{\alpha}^{(n)})$ gives the following:

$$\begin{aligned} Q_2(\vec{\alpha}, \vec{\alpha}^{(n)}) &\equiv \sum_{i=1}^{P}\sum_{s=1}^{S} \frac{N_i^s}{\beta_i^{s,(n)}} q_i^{(n)}(l_i^{s1}, l_i^{s2}) \geq \sum_{i=1}^{P}\sum_{s=1}^{S} \frac{N_i^s}{\beta_i^{s,(n)}} g_i^{(n)}(l_i^{s1}, l_i^{s2}) \\ &= \sum_{i=1}^{P}\sum_{s=1}^{S} \frac{N_i^s}{\beta_i^{s,(n)}} h_i(t_i^s(\vec{\alpha})\beta_i^{s,(n)}) = Q_1(\vec{\alpha}; \vec{\alpha}^{(n)}) \geq \theta(\vec{\alpha}) \,, \end{aligned} \tag{A.16}$$

where we used the results from earlier that $Q_1(\vec{\alpha}; \vec{\alpha}^{(n)}) \geq \theta(\vec{\alpha})$.

Lastly, conditions (A.2a) - (A.2c) will be applied to the final surrogate function of the log-likelihood function. As derived in the text, the final surrogate function is given as follows:

$$Q_{\mathcal{L}}(\vec{\alpha}, \vec{\alpha}^{(n)}) \equiv \sum_{i=1}^{P}\sum_{s=1}^{S}\sum_{j=1}^{N} w_{ij} \frac{N_i^s}{\beta_i^{s,(n)}} q_i^{(n)}\Big(\zeta_{ij}^{s1}(\alpha_j^1 - \alpha_j^{1,(n)}), \zeta_{ij}^{s2}(\alpha_j^2 - \alpha_j^{2,(n)})\Big) \,, \tag{A.17}$$

with

$$\begin{aligned} w_{ij} &= \frac{a_{ij}}{\sum_{j=1}^{N} a_{ij}}, \qquad \sum_{j=1}^{N} w_{ij} = 1 \,, \\ \zeta_{ij}^{sb}(\alpha_j^b - \alpha_j^{b,(n)}) &= \frac{a_{ij} f^b(E)}{w_{ij}}(\alpha_j^b - \alpha_j^{b,(n)}) \; + \; l_i^{sb,(n)} \,. \end{aligned} \tag{A.18}$$

1. Condition (A.2a) will be applied by substituting the variable $\vec{\alpha}^{(n)}$ into the final surrogate function. Using the properties given in (A.18), the following is obtained:

$$\begin{aligned} Q_{\mathcal{L}}(\vec{\alpha}^{(n)}, \vec{\alpha}^{(n)}) &= \sum_{i=1}^{P}\sum_{s=1}^{S}\sum_{j=1}^{N} w_{ij} \frac{N_i^s}{\beta_i^{s,(n)}} q_i^{(n)}\Big(\zeta_{ij}^{s1}(\alpha_j^{1,(n)} - \alpha_j^{1,(n)}), \zeta_{ij}^{s2}(\alpha_j^{2,(n)} - \alpha_j^{2,(n)})\Big) \\ &= \sum_{i=1}^{P}\sum_{s=1}^{S}\sum_{j=1}^{N} w_{ij} \frac{N_i^s}{\beta_i^{s,(n)}} q_i^{(n)}\Big(l_i^{s1,(n)}, l_i^{s2,(n)}\Big) = \sum_{i=1}^{P}\sum_{s=1}^{S} \frac{N_i^s}{\beta_i^{s,(n)}} q_i^{(n)}\Big(l_i^{s1,(n)}, l_i^{s2,(n)}\Big) \\ &= Q_2(\vec{\alpha}^{(n)}, \vec{\alpha}^{(n)}) \equiv \theta_{\mathcal{L}}(\vec{\alpha}^{(n)}) \,, \end{aligned} \tag{A.19}$$

   where the result from the derivation in A.9 is used. Moreover, in the second equality, $\sum_{j=1}^{N} w_{ij} = 1$ is used since there are no further dependencies on the variable j. This concludes the proof that surrogate condition A.2a is satisfied for the final surrogate function.

2. Taking the derivative of the function $q_i^{(n)}\Big(\zeta_{ij}^{s1}(\alpha_j^1 - \alpha_j^{1,(n)}), \zeta_{ij}^{s2}(\alpha_j^2 - \alpha_j^{2,(n)})\Big)$ is a very complicated task without first searching for terms that vanish due to the derivative or the substitution of α_j^b after that. The first term of the function is inherently independent of the variable of differentiation α_j^b due to the substitution of each variable at iteration step n. The third term of the function $q_i^{(n)}$ vanishes entirely. This is due to the functions $(l_i^{sk} - l_i^{sk,(n)})$ of which at least one remains in each term of the product rule, after which they vanish when substituting α_j^b. Moreover, the derivative of the second term which is evaluated in $l_i^{sb} = l_i^{sb,(n)}$, will now be evaluated in $\zeta_{ij}^{sb} = \zeta_{ij}^{sb,(n)}$. This is equivalent to evaluating the derivative in $\alpha_j^b = \alpha_j^{b,(n)}$ due to the definition of the function $\zeta_{ij}^{sb}(\alpha_j^b - \alpha_j^{b,(n)})$.

Then, using the fact that most terms in the derivative vanish, we obtain the following expression for the function $q_i^{(n)}$:

$$\frac{\partial q_i^{(n)}\Big(\zeta_{ij}^{s1}(\alpha_j^1-\alpha_j^{1,(n)}),\zeta_{ij}^{s2}(\alpha_j^2-\alpha_j^{2,(n)})\Big)}{\partial \alpha_j^b}\bigg|_{\vec{\alpha}=\vec{\alpha}^{(n)}}$$
$$=\left[\sum_{b=1}^{2}\frac{\partial g_i^{(n)}\Big(\zeta_{ij}^{s1}(\alpha_j^1-\alpha_j^{1,(n)}),\zeta_{ij}^{s2}(\alpha_j^2-\alpha_j^{2,(n)})\Big)}{\partial \zeta_{ij}^{sb}(\alpha_j^b-\alpha_j^{b,(n)})}\bigg|_{\alpha_j^b=\alpha_j^{b,(n)}} w_{ij}\frac{\partial \zeta_{ij}^{sb}(\alpha_j^b-\alpha_j^{b,(n)})}{\partial \alpha_j^b}\right]\bigg|_{\vec{\alpha}=\vec{\alpha}^{(n)}}, \tag{A.20}$$

where the sum $\sum_{j=1}^{N} w_{ij}$ arises from the substitution of Equation (**??**) in the function $(l_i^{sb}-l_i^{sb,(n)})$. The two derivatives in this expression will be calculated separately to simplify notations. We begin by calculating the following derivative:

$$\sum_{b=1}^{2}\frac{\partial g_i^{(n)}\Big(\zeta_{ij}^{s1}(\alpha_j^1-\alpha_j^{1,(n)}),\zeta_{ij}^{s2}(\alpha_j^2-\alpha_j^{2,(n)})\Big)}{\partial \zeta_{ij}^{sb}(\alpha_j^b-\alpha_j^{b,(n)})}\bigg|_{\alpha_j^b=\alpha_j^{b,(n)}}$$
$$=\sum_{b=1}^{2}\frac{\partial}{\partial \zeta_{ij}^{sb}(\alpha_j^b-\alpha_j^{b,(n)})}\left[h_i^s\Big(e^{-\sum_{b=1}^{2}\zeta_{ij}^{sb}(\alpha_j^b-\alpha_j^{b,(n)})}\beta_i^{s,(n)}\Big)\right]\bigg|_{\alpha_j^b=\alpha_j^{b,(n)}}$$
$$=\sum_{b=1}^{2}\frac{\partial}{\partial \zeta_{ij}^{sb}(\alpha_j^b-\alpha_j^{b,(n)})}\left[e^{-\sum_{b=1}^{2}\zeta_{ij}^{sb}(\alpha_j^b-\alpha_j^{b,(n)})}\beta_i^{s,(n)}-Y_i\log\Big(e^{-\sum_{b=1}^{2}\zeta_{ij}^{sb}(\alpha_j^b-\alpha_j^{b,(n)})}\beta_i^{s,(n)}\Big)\right]\bigg|_{\alpha_j^b=\alpha_j^{b,(n)}}$$
$$=\sum_{b=1}^{2}\frac{\partial}{\partial \zeta_{ij}^{sb}(\alpha_j^b-\alpha_j^{b,(n)})}\left[e^{-\sum_{b=1}^{2}\zeta_{ij}^{sb}(\alpha_j^b-\alpha_j^{b,(n)})}\beta_i^{s,(n)}-Y_i\Big(-\sum_{b=1}^{2}\zeta_{ij}^{sb}(\alpha_j^b-\alpha_j^{b,(n)})+\log\beta_i^{s,(n)}\Big)\right]\bigg|_{\alpha_j^b=\alpha_j^{b,(n)}}$$
$$=\Big(-e^{-\sum_{b=1}^{2}\zeta_{ij}^{sb}(\alpha_j^b-\alpha_j^{b,(n)})}\beta_i^{s,(n)}+Y_i\Big)\bigg|_{\alpha_j^b=\alpha_j^{b,(n)}}=-e^{-\sum_{b=1}^{2}l_i^{sb,(n)}}\beta_i^{s,(n)}+Y_i$$
$$=Y_i-\hat{Y}_i(\vec{\alpha}^{(n)})\,. \tag{A.21}$$

Then, the second term in Equation (A.20) is evaluated as follows:

$$\frac{\partial \zeta_{ij}^{sb}(\alpha_j^b-\alpha_j^{b,(n)})}{\partial \alpha_j^b}=\frac{\partial}{\partial \alpha_j^b}\left[\frac{a_{ij}f^b(E)}{w_{ij}}(\alpha_j^b-\alpha_j^{b,(n)})\;+\;l_i^{sb,(n)}\right]=\frac{a_{ij}f^b(E)}{w_{ij}}\,. \tag{A.22}$$

Substituting these formulas in the derivative for the function $q_i^{(n)}\Big(\zeta_{ij}^{s1}(\alpha_j^1-\alpha_j^{1,(n)}),\zeta_{ij}^{s2}(\alpha_j^2-\alpha_j^{2,(n)})\Big)$ results in the following expression:

$$\begin{aligned}\frac{\partial q_i^{(n)}\Big(\zeta_{ij}^{s1}(\alpha_j^1-\alpha_j^{1,(n)}),\zeta_{ij}^{s2}(\alpha_j^2-\alpha_j^{2,(n)})\Big)}{\partial \alpha_j^b}\bigg|_{\vec{\alpha}=\vec{\alpha}^{(n)}}&=\big(Y_i-\hat{Y}_i(\vec{\alpha}^{(n)})\big)w_{ij}\frac{a_{ij}f^b(E)}{w_{ij}}\\&=\big(Y_i-\hat{Y}_i(\vec{\alpha}^{(n)})\big)a_{ij}f^b(E)\,.\end{aligned} \tag{A.23}$$

The final step is to substitute this expression into the derivative of the final surrogate

function $Q_{\mathcal{L}}(\vec{\alpha};\vec{\alpha}^{(n)})$:

$$\begin{aligned}\left.\frac{\partial Q_{\mathcal{L}}(\vec{\alpha};\vec{\alpha}^{(n)})}{\partial \alpha_j^b}\right|_{\vec{\alpha}=\vec{\alpha}^{(n)}} &= \sum_{i=1}^{P}\sum_{s=1}^{S}\sum_{j=1}^{N} w_{ij}\frac{N_i^s}{\beta_i^{s,(n)}}\left.\frac{\partial q_i^{(n)}\Big(\zeta_{ij}^{s1}(\alpha_j^1-\alpha_j^{1,(n)}),\zeta_{ij}^{s2}(\alpha_j^2-\alpha_j^{2,(n)})\Big)}{\partial \alpha_j^b}\right|_{\vec{\alpha}=\vec{\alpha}^{(n)}}\\ &= \sum_{i=1}^{P}\sum_{s=1}^{S}\sum_{j=1}^{N} w_{ij}\frac{N_i^s}{\beta_i^{s,(n)}}\big(Y_i-\hat{Y}_i(\vec{\alpha}^{(n)})\big)a_{ij}f^b(E)\\ &= \sum_{i=1}^{P}\Big(\sum_{s=1}^{S}\frac{N_i^s}{\beta_i^{s,(n)}}\Big)\Big(\sum_{j=1}^{N} w_{ij}\Big)\big(Y_i-\hat{Y}_i(\vec{\alpha}^{(n)})\big)a_{ij}f^b(E)\\ &= \sum_{i=1}^{P}\big(Y_i-\hat{Y}_i(\vec{\alpha}^{(n)})\big)a_{ij}f^b(E) \equiv \left.\frac{\partial\theta_{\mathcal{L}}(\vec{\alpha})}{\partial\alpha_j^b}\right|_{\vec{\alpha}=\vec{\alpha}^{(n)}}. \end{aligned} \tag{A.24}$$

This concludes the proof that the second surrogate condition holds for the final surrogate function.

3. The last condition will be proven for the final surrogate function $Q_{\mathcal{L}}(\vec{\alpha};\vec{\alpha}^{(n)})$. Jensen's inequality will be applied to the variable w_{ij}, since it satisfies all conditions by the properties of w_{ij} in Equation (A.18:

$$\begin{aligned}Q_{\mathcal{L}}(\vec{\alpha},\vec{\alpha}^{(n)}) &\equiv \sum_{i=1}^{P}\sum_{s=1}^{S}\sum_{j=1}^{N} w_{ij}\frac{N_i^s}{\beta_i^{s,(n)}}q_i^{(n)}\Big(\zeta_{ij}^{s1}(\alpha_j^1-\alpha_j^{1,(n)}),\zeta_{ij}^{s2}(\alpha_j^2-\alpha_j^{2,(n)})\Big)\\ &\geq \sum_{i=1}^{P}\sum_{s=1}^{S}\frac{N_i^s}{\beta_i^{s,(n)}}q_i^{(n)}\Big(\sum_{j=1}^{N} w_{ij}\zeta_{ij}^{s1}(\alpha_j^1-\alpha_j^{1,(n)}),\sum_{j=1}^{N} w_{ij}\zeta_{ij}^{s2}(\alpha_j^2-\alpha_j^{2,(n)})\Big)\\ &= \sum_{i=1}^{P}\sum_{s=1}^{S}\frac{N_i^s}{\beta_i^{s,(n)}}q_i^{(n)}\Big(l_i^{s1},l_i^{s2}\Big) \equiv Q_2(\vec{\alpha},\vec{\alpha}^{(n)}) \geq \theta_{\mathcal{L}}(\vec{\alpha})\,. \end{aligned} \tag{A.25}$$

This proves that condition 2.32c holds for the final surrogate function.

A.3. Proof of Surrogate Conditions for the Surrogate Function of the Regularisation Term

Similarly to the proof given in Section A.2, we will show that the surrogate conditions are satisfied with the surrogate function of the regularisation term. The regularisation part of the cost function is given as follows:

$$\theta_{\mathcal{R}}(\vec{\alpha}) = \sum_{b=1}^{2}\sum_{j=1}^{N}\sum_{k\in N_j}\lambda_b\phi^{\text{Huber}}(\alpha_j^b-\alpha_k^b,\gamma_b)\,, \tag{A.26}$$

with $\phi^{\text{Huber}}(\alpha_j^b-\alpha_k^b,\gamma_b)$ as defined in Equation (2.30) in Section 2.5.1. Then, the derivative of this function with respect to the variable α_j^b, is:

$$\begin{aligned}\frac{\partial\theta_{\mathcal{R}}(\vec{\alpha})}{\partial\alpha_j^b} &= \frac{\partial}{\partial\alpha_j^b}\left[\sum_{b=1}^{2}\sum_{j=1}^{N}\sum_{k\in N_j}\lambda_b\phi^{\text{Huber}}(\alpha_j^b-\alpha_k^b,\gamma_b)\right]\\ &= \sum_{b=1}^{2}\sum_{j=1}^{N}\sum_{k\in N_j}\lambda_b\frac{\partial\phi^{\text{Huber}}(\alpha_j^b-\alpha_k^b,\gamma_b)}{\partial\alpha_j^b}\,, \end{aligned} \tag{A.27}$$

where the derivative of $\phi^{\text{Huber}}(\alpha_j^b - \alpha_k^b, \gamma_b)$ is given as:

$$\frac{\partial \phi^{\text{Huber}}(\alpha_j^b - \alpha_k^b, \gamma_b)}{\partial \alpha_j^b} = \begin{cases} 2(\alpha_j^b - \alpha_k^b) & \text{if } |\alpha_j^b - \alpha_k^b| < \gamma_b \\ 2\gamma_b \cdot \text{sgn}(\alpha_j^b - \alpha_k^b) & \text{if } |\alpha_j^b - \alpha_k^b| \geq \gamma_b \ , \end{cases} \tag{A.28}$$

with

$$\text{sgn}(x) = \begin{cases} -1 & \text{if } x < 0 \\ 0 & \text{if } x = 0 \\ 1 & \text{if } x > 0 \, . \end{cases} \tag{A.29}$$

As proven in Section 2.5.1, the surrogate function for the regularisation term is as follows:

$$Q_{\mathcal{R}}(\vec{\alpha}; \vec{\alpha}^{(n)}) \equiv \sum_{b=1}^{2} \sum_{j=1}^{N} \sum_{k \in N_j} \lambda_b \Big[\frac{1}{2} \phi^{\text{Huber}}(2\alpha_j^b - \alpha_j^{b,(n)} - \alpha_k^{b,(n)}, \gamma_b) + \frac{1}{2} \phi^{\text{Huber}}(2\alpha_k^b - \alpha_j^{b,(n)} - \alpha_k^{b,(n)}, \gamma_b) \Big] \tag{A.30}$$

Conditions (A.2a) - (A.2c) will be checked for the surrogate function of the regularisation term.

1. Substituting the variable $\vec{\alpha}^{(n)}$ into surrogate function (A.30) gives us the following:

$$\begin{aligned} & Q_{\mathcal{R}}(\vec{\alpha}^{(n)}; \vec{\alpha}^{(n)}) \\ & \equiv \sum_{b=1}^{2} \sum_{j=1}^{N} \sum_{k \in N_j} \lambda_b \Big[\frac{1}{2} \phi^{\text{Huber}}(2\alpha_j^{b,(n)} - \alpha_j^{b,(n)} - \alpha_k^{b,(n)}, \gamma_b) + \frac{1}{2} \phi^{\text{Huber}}(2\alpha_k^{b,(n)} - \alpha_j^{b,(n)} - \alpha_k^{b,(n)}, \gamma_b) \Big] \\ & = \sum_{b=1}^{2} \sum_{j=1}^{N} \sum_{k \in N_j} \lambda_b \Big[\frac{1}{2} \phi^{\text{Huber}}(\alpha_j^{b,(n)} - \alpha_k^{b,(n)}, \gamma_b) + \frac{1}{2} \phi^{\text{Huber}}(\alpha_j^{b,(n)} - \alpha_k^{b,(n)}, \gamma_b) \Big] \\ & = \sum_{b=1}^{2} \sum_{j=1}^{N} \sum_{k \in N_j} \lambda_b \phi^{\text{Huber}}(\alpha_j^{b,(n)} - \alpha_k^{b,(n)}, \gamma_b) \equiv \theta_{\mathcal{R}}(\vec{\alpha}^{(n)}) \, , \end{aligned} \tag{A.31}$$

   which concludes the proof of the first surrogate condition.

2. The derivative of the surrogate function is given as:

$$\begin{aligned} & \frac{\partial Q_{\mathcal{R}}(\vec{\alpha}; \vec{\alpha}^{(n)})}{\partial \alpha_j^b} \\ & = \frac{\partial}{\partial \alpha_j^b} \Big[\sum_{b=1}^{2} \sum_{j=1}^{N} \sum_{k \in N_j} \lambda_b \Big(\frac{1}{2} \phi^{\text{Huber}}(2\alpha_j^b - \alpha_j^{b,(n)} - \alpha_k^{b,(n)}, \gamma_b) + \frac{1}{2} \phi^{\text{Huber}}(2\alpha_k^b - \alpha_j^{b,(n)} - \alpha_k^{b,(n)}, \gamma_b) \Big) \Big] \\ & = \sum_{b=1}^{2} \sum_{j=1}^{N} \sum_{k \in N_j} \lambda_b \left(\frac{1}{2} \frac{\partial \phi^{\text{Huber}}(2\alpha_j^b - \alpha_j^{b,(n)} - \alpha_k^{b,(n)}, \gamma_b)}{\partial \alpha_j^b} + \frac{1}{2} \frac{\partial \phi^{\text{Huber}}(2\alpha_k^b - \alpha_j^{b,(n)} - \alpha_k^{b,(n)}, \gamma_b)}{\partial \alpha_j^b} \right) . \end{aligned} \tag{A.32}$$

After substituting the variable $\vec{\alpha} = \vec{\alpha}^{(n)}$ into the derivative, the following is obtained:

$$\begin{aligned}
&\left.\frac{\partial Q_{\mathcal{R}}(\vec{\alpha};\vec{\alpha}^{(n)})}{\partial \alpha_j^b}\right|_{\vec{\alpha}=\vec{\alpha}^{(n)}} \\
&= \sum_{b=1}^{2}\sum_{j=1}^{N}\sum_{k\in N_j} \lambda_b \left(\frac{1}{2}\frac{\partial \phi^{\text{Huber}}(2\alpha_j^{b,(n)} - \alpha_j^{b,(n)} - \alpha_k^{b,(n)}, \gamma_b)}{\partial \alpha_j^b} + \frac{1}{2}\frac{\partial \phi^{\text{Huber}}(2\alpha_k^{b,(n)} - \alpha_j^{b,(n)} - \alpha_k^{b,(n)}, \gamma_b)}{\partial \alpha_j^b} \right) \\
&= \sum_{b=1}^{2}\sum_{j=1}^{N}\sum_{k\in N_j} \lambda_b \frac{\partial \phi^{\text{Huber}}(\alpha_j^{b,(n)} - \alpha_k^{b,(n)}, \gamma_b)}{\partial \alpha_j^b} \equiv \left.\frac{\partial \theta_{\mathcal{R}}(\vec{\alpha})}{\partial \alpha_j^b}\right|_{\vec{\alpha}=\vec{\alpha}^{(n)}} . \qquad \text{(A.33)}
\end{aligned}$$

Here, the derivative of the Huber function is given as in Equation (A.28). This proofs the second surrogate condition.

3. Lastly, the third surrogate condition is again proved using Jensen's inequality:

$$\begin{aligned}
\theta_{\mathcal{R}}(\vec{\alpha}) &= \sum_{b=1}^{2}\sum_{j=1}^{N}\sum_{k\in N_j} \lambda_b \phi^{\text{Huber}}(\alpha_j^b - \alpha_k^b, \gamma_b) \\
&= \sum_{b=1}^{2}\sum_{j=1}^{N}\sum_{k\in N_j} \lambda_b \phi^{\text{Huber}}\Big[\frac{1}{2}\big(2\alpha_j^b - \alpha_j^{b,(n)} - \alpha_k^{b,(n)}\big) + \frac{1}{2}\big(-2\alpha_k^b + \alpha_j^{b,(n)} + \alpha_j^{b,(n)}\big)\Big] \\
&\leq \sum_{b=1}^{2}\sum_{j=1}^{N}\sum_{k\in N_j} \lambda_b \frac{1}{2}\phi^{\text{Huber}}\big(2\alpha_j^b - \alpha_j^{b,(n)} - \alpha_k^{b,(n)}\big) + \frac{1}{2}\phi^{\text{Huber}}\big(2\alpha_k^b - \alpha_j^{b,(n)} - \alpha_k^{b,(n)}\big) \\
&\equiv Q_{\mathcal{R}}(\vec{\alpha};\vec{\alpha}^{(n)}) . \qquad \text{(A.34)}
\end{aligned}$$

Jensen's inequality can be used due to the convexity of the function $\phi^{\text{Huber}}(\alpha_j^b - \alpha_k^b, \gamma_b)$, and because the weights of the terms inside the Huber function add up to one ($\frac{1}{2}+\frac{1}{2}=1$). The term $(-2\alpha_k^b + \alpha_j^{b,(n)} + \alpha_j^{b,(n)})$ changes sign after the use of the inequality due to the fact that the Huber function is symmetric.

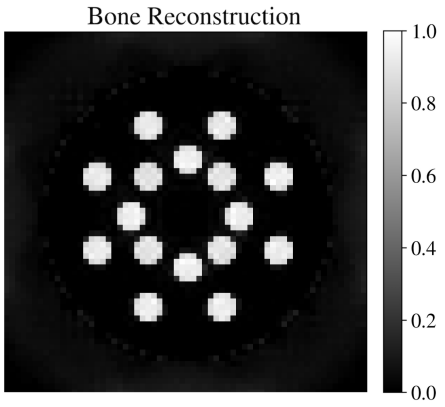
That concludes the proof, so we know that $Q_{\mathcal{R}}(\vec{\alpha};\vec{\alpha}^{(n)})$ is a satisfactory surrogate function of $\theta_{\mathcal{R}}(\vec{\alpha})$.

B

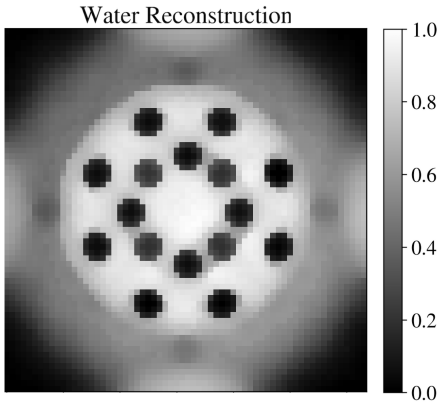
Extra Images and Results

B.1. Problems with the Complex Phantom

The implementation of the complex phantom from Figure 3.2 into the `Python` code revealed significant imaging artefacts, indicating that an error occurred during reconstruction. This can be seen in Figure B.1. To find out whether the phantom was in view of the detector, a projection of the phantom was plotted onto the detector in `Python`. This projection can be seen in Figure B.2a, where an arbitrary projection is displayed for an arbitrary energy bin. It is desirable that the phantom is small enough to have some room on the edges to ensure it is in view of the detector. However, this is clearly not the case for the 64x64x64 pixel phantom, as the phantom is pressed against the borders of the detector. This could mean that the phantom surpasses the field of view of the detector. There are two steps needed to solve this problem. First, the size of the phantom was limited in the z-direction. The number of pixels in the z-direction of the detector is 44, and therefore the depth of the phantom should not exceed this number. A new variable `objectDepth` is introduced that determines the amount of pixels with which the phantom is stacked in the z-direction. Previously, this was the same amount as the size of the phantom. With this limitation, the phantom still has a size of `objectSize` x `objectSize` x `objectSize`, however, it is stacked with `objectDepth` pixels in the centre of this phantom, while the rest of the slices in the z-direction contain only zeros. Secondly, the size of the outer circle of the phantom is adjusted to the detector, which would limit the projection of the phantom onto the detector in the y-direction. The projection of the smaller phantom is seen in Figure B.2b, for the same arbitrary projection and energy bin. It is clear that the projection of the phantom now fits better inside the detector.

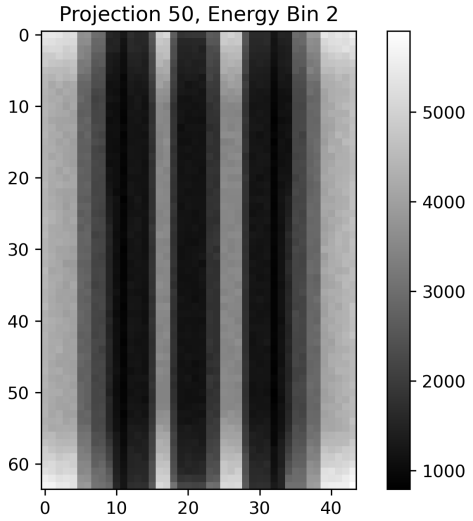


(a) Bone reconstructed image of the 64x64x64 pixel phantom of size 25.6 mm.

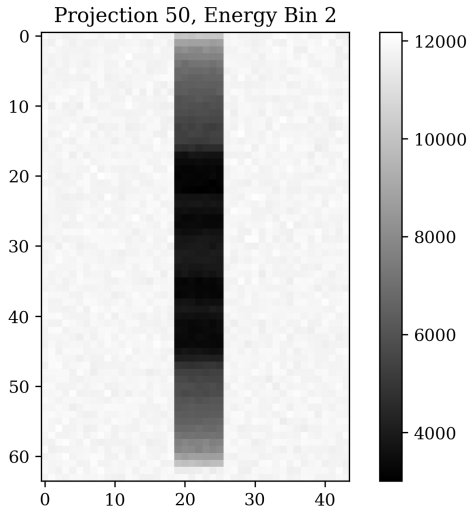


(b) Water reconstructed image of the 64x64x64 pixel phantom of size 25.6 mm.

Figure B.1: The 64x64x64 pixel reconstructed images without size correction with 100 projections, 10 iterations and 4 subsets. It is clear that especially the water image exhibits a substantial amount of artefacts.



(a) Projection of the phantom for arbitrary projection and energy bin without size correction.



(b) Projection of the phantom for arbitrary projection and energy bin with size correction.

Figure B.2: Projection of the phantom for arbitrary projection and energy bin without size correction.

B.2. Extra Results Reconstruction Quality

Water Reconstruction	Mean Muscle Region	Mean Blood Region	Contrast W–M	Contrast W–B
5 iterations	0.891	0.912	0.140	0.151
10 iterations	0.938	0.960	0.034	0.046
12 iterations	0.934	0.957	0.014	0.027
25 iterations	1.000	0.989	0.034	0.039

Table B.1: Mean values for water and bone, and their contrast, calculated from the bone reconstructions in Figure 4.2.

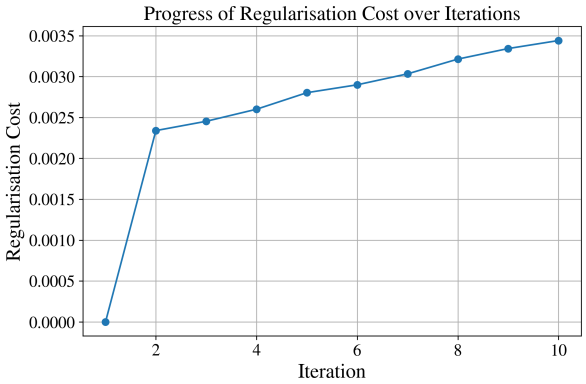
Possible text for the tables above: The mean value for the muscle and blood regions, along with the contrast between water and muscle (W – M) and between water and blood (W – B), are given in Table B.1. The desired mean value of both blood and muscle is equal to one,

Bone Reconstruction	Mean Muscle Region	Mean Blood Region	Contrast W–M	Contrast W–B
5 iterations	-0.006	-0.004	1.222	1.135
10 iterations	-0.007	-0.004	0.595	0.331
12 iterations	-0.005	-0.002	0.363	0.738
25 iterations	-0.011	-0.006	0.703	0.523

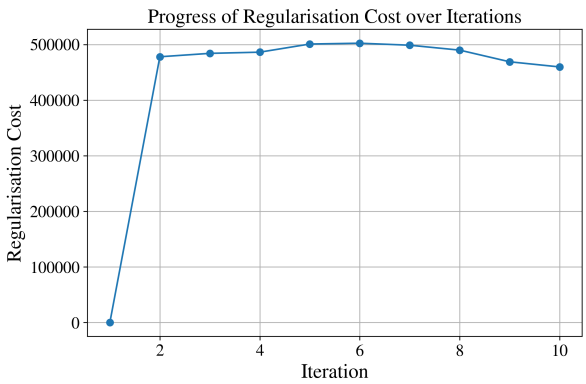
Table B.2: Mean values for water and bone, and their contrast, calculated from the bone reconstructions in Figure 4.2.

and therefore it is possible to conclude that both biological materials are constructed better for higher iterations. However, it appears that the contrast between water and both other materials decreases with higher iterations. This is likely due to inaccurate calculations for the water contribution in reconstructions with fewer iterations. As can be seen in Figure 4.3b, the MSE for water is high for low iterations. The blood and muscle filled regions are reconstructed much better for small iterations than water. According to Table 4.1, the mean contribution value in the water region is equal to 0.672, compared to 0.891 and 0.912 for muscle and blood, respectively.

B.3. Regularisation Cost Function over Iterations



(a) Regularisation cost over iterations for a reconstruction with tuning parameters $\lambda_b = [0.001, 0.001]$ and $\gamma_b = [0.000010.00001]$.



(b) Regularisation cost over iterations for a reconstruction with tuning parameters $\lambda_b = [100, 50]$ and $\gamma_b = [0.01, 0.05]$.

Figure B.3: Regularisation cost over iterations for reconstructions with different tuning parameters. It is clear that the regularisation cost from the reconstruction without regularisation (left) is negligible compared to the regularisation cost from the reconstruction with the original tuning parameters. Therefore, we can conclude that the regularisation does not contribute to the reconstruction with the negligible tuning parameters.