Motion correction in X-ray tomography

submitted by

Ander Biguri

for the degree of Doctor of Philosophy

of the

University of Bath

Department of Electrical and Electronic Engineering

September 2017

Abstract

bla

Contents

Chapter 1

Introduction

Chapter 2

X-ray imaging in medicine

Chapter 3

The image reconstruction problem

This chapter tries to explain the mathematics behind CT reconstruction, the FDK algorithm and iterative reconstruction algorithms. After the formal proposition of the mathematical problem of integrating over straight lines,the FDK algorithm is introduced. Then, the alternative proposal of the iterative algebraic methods is shown, followed by a wide variety of different algorithms that can be used to solve the algebraic problem. These include gradient descend techniques, Krylov subspace methods, statistical approaches and compressed sensing techniques. Finally, a discussion of the challenges that arise from the use iterative algorithms are described.

3.1 Geometry of CBCT

3.2 FDK

3.3 Iterative reconstruction algorithms

Nowadays the FDK algorithm is the most widely used algorithm, and only until very recently [CITE] it has been the only algorithm available in any commercial medical or industrial CT device. While using FDK is advantageous in some cases, often the algorithm behaves poorly, specially when errors arise in the data, or the amount of data is limited. This is because FDK is based on an analytical approximation of straight path integrals in continuous spaces. Reality is far from straight path integrals, as due to X-ray physics photons do not behave linearly. Photons from CT machines are polychromatic, and human tissue is behaves non-linearly in respect to X-ray energy.

Additionally, Compton scattering is a common effect, where the photons get reflected in different angles dependent on their energy. Apart from photon physics related errors, electronic noise is always present in detector technology being the only feasible way of avoiding it longer exposition times, harmful to live tissue. Limited data can additionally harm the image reconstruction, as CT has generally less detector data than the amount of voxels are desired to reconstruct. All these effects make CT image reconstruction a challenging problem and have strong effect in the behaviour of FDK. As an alternative to FDK, iterative algebraic reconstruction algorithms try to minimize a functional by updating the image continuously and comparing it to the measured data. These algorithms have been shown to improve reconstruction quality when the data is noisy and/or limited. Additionally, as they are an algebraic tool, they allow careful tuning of the mathematics, enabling them to change their behaviour.

Iterative algorithms in CT generally reffer to those algorithms that, as the name says, iterate, but solve the linearized model

$$Ax = b + \tilde{e} \tag{3.1}$$

on where $x \in \mathbb{R}^{N_{voxels}}$ is a vector representing the lexicographically ordered voxels of the 3D image, $b \in \mathbb{R}^{N_{pixels}}$ a vector of the detector measured pixels. A is the linearized model matrix, a matrix that describes the behaviour of the CT system. Each row of the matrix A is describes the behaviour of the X-rays that affect each single pixel in the detector. However, this matrix is so big that in practize its explicit form is impossible to store, and the matrix product operations Ax (or projection) and $A^T b$ are implemented instead. The next chapter goes into a bit more detail on how to operate with matrix A and its limitations. Errors from measurement are inevitable in any application, and there are linearization errors, as no model is perfect. In equation 3.1, $\tilde{e}$ represents all those errors.

As an exact solution for x can not be found, the problem in equation 3.1 is minimized as

$$\hat{x} = \arg\min_x \|Ax - b\|^2 + R(x), \tag{3.2}$$

on where $R(x)$ is an optional regularization function. This minimization function has been widely studied in mathematics and there are multiple algorithms that can solve it. However not all algorithms that solve the equation can be used in CT reconstruction, due to the nature of the A matrix, as it is very big (approximately $10^8 \times 10^8$ in a standard medical image) and very sparse (approximately 0.0017% of sparsity in a standard medical image). This makes the matrix severely ill-conditioned and impossible to store in memory. Additionally, often the CT problem can be underdetermined, making the

problem ill-posed and further constraining the possible algorithms that can be used. That said, a wide variety of algorithm have been proposed to solve the CT algebraic problem and new ones are still being published. This section discusses a few of the available and most common algorithms that have been studied in this work.

3.3.1 Algebraic Reconstruction Techniques

Arguably the most well known iterative algorithm is the method known as the algebraic reconstruction technique (ART)[9], known as the Kaczmarz method outside the CT imaging field. The ART algorithm, for matrix elements $a_{ij} \in R$ is defined as

$$x^{n+1} = x^n + \frac{b_i - \langle a_i, x^n \rangle}{\|a_i\|^2} a_i^T, \tag{3.3}$$

where a_i is the i-th row of matrix A and $\langle \, , \rangle$ denotes the inner product. The ART method projects the image into the hyperplane described by the equation in row i. Generally the method includes a relaxation parameter λ_n that controls the update size, and that decreases with the iteration number. This generally avoids the cyclical convergence that the method describes when the solution is not unique (the intersection of the hyperplanes is not a single point). By relaxing the update step, the algorithm converges to a single point. Generally the algorithm is also run with some inequality constrains, the most common one being a positivity constrain for x, as negative values are not physically possible.

Studies on the convergence of the ART algorithm show[6] that randomly choosing the order of the rows in each iteration (n) increase the convergence rate, even more if the probabilities of picking rows are different than one (different methods propose different probabilities)[17][12].

However, the ART method has a major disadvantage: the image x^{n+1} needs to be updated i times each iteration. In current CT applications, and specifically in CBCT, the amount of rows in the matrix i.e. the total amount of independent pixel measurements in the detector is a massive number. Following the same definition of standard medical image size from the thesis, a 512×512 detector, with 360 projection means that the amount of rows is in the order of 10^8. In order to update the image less, the Simultaneous Iterative Reconstruction Technique (SIRT)[10] can be used, a method that is very similar to Cimmino's method, that updates the image using simultaneously (instead of sequentially) all data in the measurement b, thus each iteration is a single update. While SIRT generally solves the problem of the high amount of updates in ART, it suffers from a very slow convergence in comparison, and will generally plateau in a solution that is not as good as what ART provides. The SIRT algorithm can be

described in matrix form as

$$x^{n+1} = x^n + \lambda_n V A^T W^{-1} (b - Ax) \tag{3.4}$$

where $V = 1/\text{diag}\left(\sum_j a_{ij}\right)$ and $W = \text{diag}\left(\sum_i a_{ij}\right)$.

However, middle ground has also been proposed. Kak and Andersen proposed[1] the Simultaneous Algebraic Reconstruction Technique (SART) on where the image is updated using simultaneously all data from each X-ray projection, but still updating the image multiple times per iteration. Finally, the update can also be done using block-based methods, or Oriented Subsets (OS) with a variety of methods generally described as OS-SART[2] methods.

Similarly as with ART, the order of the subsets in both OS-SART and SART have influence in the convergence, with a lower impact than in ART. In this work an three methods have been implemented, a completely ordered method, a randomized ordered method with full sampling (i.e. all projections are ensured to be used once and only once per iteration) and an angular distance based one. This last one orders the subsets by selecting the next one as the subset with largest angular distance from the ones already used. The heuristic rationale is that the projections at larger angular distance update the image by a bigger step than projections angularly near. In all further result in this thesis, the default ordering is random, unless otherwise explicitly stated.

Relaxation parameter λ

As previously mentioned, changing the relaxation parameter per iteration can be of advantage, by avoiding cyclical convergence and often by increasing the general convergence rate. On of the commonly used methods for the reduction of lambda is simply reducing it by a reduction factor each iteration as

$$\lambda_{n+1} = \lambda_n * r_\lambda \tag{3.5}$$

where r_λ is some value close to one, such as $r_\lambda = 0.99$ or $r_\lambda = 0.999$. Alternatively, convergence studies for SART/OS-SART/SIRT algorithms have shown[8] that an update of the relaxation parameter in the form of

$$\lambda_{n+1} = \frac{\lambda_0}{1 + n^\alpha} \tag{3.6}$$

where λ_0 is an initial value of the relaxation parameter (usually 1) and $0 < \alpha \leq 1$. this can improve the convergence of the algorithms.

It is worth noticing that this family of algorithms is very closely related to the well

known gradient descend methods, as the gradient of equation 3.2 is proportional to $A^T(Ax-b)$, or in other words $V = I$ and $W = I$ in equation 3.4. The gradient descend methods have been widely studied in the past years[18][14], as the Neural Networks community tries to find faster converging methods to train the nets they research. Among other methods proposed, Nesterov proposed an accelerated version of the gradient descend[13], that obtains a rate of convergence of $1/n^2$. The proposed update updates the result image in each iteration by pushing it in the current update and previous update direction combined. Nesterovs Accelerated Gradient (NAG) defines

$$\lambda_{n+1} = \frac{1+\sqrt{1+4\lambda_n^2}}{2} \tag{3.7}$$

$$\gamma_n = \frac{1-\lambda_n}{\lambda_{n+1}} \tag{3.8}$$

$$y^{n+1} = x^n - \frac{1}{\beta}\nabla f(x^n) \tag{3.9}$$

$$x^{n+1} = (1-\gamma_n)y^{n+1} + \gamma y^n \tag{3.10}$$

with $\lambda_0 = 1$ and β being the Lipschitz smoothness of the function f. The line in equation 3.9 can be replaced by the SART/OS-SART/SIRT update on equation 3.4 to obtain an accelerated convergence rate. Some experimental results on the convergence of the algorithms can be found in Chapter 5.

3.3.2 Conjugate Gradient Least Squares

The conjugate gradient for the least squares problems (CGLS)

3.3.3 Statistical inversion

3.3.4 Total variation minimization with POCS

Sometimes solving a regularized problem may result in a better final image than just trying to solve the data constrain with the model. This is especially useful in more ill-conditioned problems, such as when the data is very noisy (thus the model does not fit the data with accuracy) or when few projections are available (the system is more under-determined). In these cases, regularisation can add a user constrain in the image domain that pushes the algorithm towards an specific solution among all the multiple possibilities. While a variety of regularization techniques and norms exist, the most suitable for CT imaging is the total variation (TV) norm.

The total variation norm is defined as the sum of the 2-norms of the directional gradients of the variable,

$$\|x\|_{\mathrm{TV}} = \sum_n \left\| \sum_\alpha \partial_\alpha x_n \right\|_2 . \tag{3.11}$$

Applied to CT imaging, the total variation norm is the sum of the total change occurred in the image. An image with less total variation would be an image that would have less change, or more flat, same valued regions. Regularizing with the TV norm as a minimization term will yield an image that is piecewise smooth and it happens that most of the objects imaged in CT scanners are piecewise smooth in linear attenuation, even more in medical CT imaging.

However, solving the minimization problem in equation XX is not trivial with the TV regularization. One of the first robust algorithm is the so called Adaptive Steepest Descend, Projection Onto Convex Subsets, or ASD-POCS algorithm[16]. This algorithm not only minimizes the data constrain with TV regularization but also adaptively controls the TV minimization update, in order to adapt its strength according to the data constrain update. Several adaptations and improvements of this algorithm have been proposed in the literature[CITES], all based in the same mathematical basis.

ASD-POCS

The previous algorithms discussed in this chapter where unconstrained minimization methods. While the TV minimization problem can be solved similarly (see section 3.3.5) formalizing the algorithm as a non-linear constrained minimization adds an advantage in the case where there system is under-determined. In an unconstrained problem such as in equation XX, the balance between the data constrain and the regularization constrain can be tuned via a hyperparameter, but in the case of an under-determined system, multiple solutions for the data fidelity term may exist. By reformulating it as it is shown in the rest of this section, the image with the same data fidelity 2-norm but the lowest TV norm can be chosen.

The minimization will yield an image $\vec{x}^*$ that minimizes

$$\vec{x}^* = \arg\min_x \|\vec{x}\|_{\mathrm{TV}} \tag{3.12}$$

subject to

$$\|A \cdot \vec{x} - \vec{b}\| \leq \epsilon, \tag{3.13}$$

$$\vec{x} \geq 0. \tag{3.14}$$

As previously described in this chapter, the data fidelity on equation 3.13 while

desired to be zero, it will never reach to zero, due to inconsistencies in the data, model, noise, etcetera. Thus, in this algorithm it is introduced as a inequality constrain, instead of as the minimization problem itself. This introduces the parameter ϵ in the algorithm, the maximum 2-norm allowed for the data inconsistency. The problem in hand is now non-linear, due to the constrains, but convex.

The conditions for a constrained minimization to find the optimal solution can be obtained by satisfying the Karush Kuhn-Tucker conditions (a generalization of the Lagrange multiplies for inequality constrains). First, the Lagrangian for the current problems needs to be defined, as

$$\mathbf{L} = \|\vec{x}\|_{\mathrm{TV}} + \lambda_0 \cdot (\|A \cdot \vec{x} - \vec{b}\|^2 - \epsilon^2) - \vec{\lambda} \cdot \vec{x}, \tag{3.15}$$

where $\vec{\lambda}$ is a vector of the same size as the image, but λ_0 is a single value. Two inequality constrains are imposed to the Lagrange multipliers, namely non-negativity

$$\lambda_i \geq 0, \tag{3.16}$$

and complementarity

$$h_i(\vec{x})\lambda_i = 0, \tag{3.17}$$

where $i = 0, 1, ..., N_{pixels}$, and h_i is an alternative form of the inequality constrains as

$$h_0(\vec{x}) = \|A \cdot \vec{x} - \vec{b}\|^2 - \epsilon^2 \leq 0 \tag{3.18}$$

$$h_i(\vec{x}) = -x_i \leq 0 \qquad\qquad i \in [1, N_{pixels}] \tag{3.19}$$

Thus, only when the inequalities are violated does h_i turns non-zero, and with the complementarity condition, does the corresponding λ_i turns zero. A solution can be found for $\vec{x}$ when the gradient of the Lagrangian is zero, and if the differential operator is defined as

$$\nabla_{\vec{x}} Q(\vec{x}) = \sum_i \partial_{x_i} Q(\vec{x}) \vec{\delta}_i \tag{3.20}$$

where $\vec{\delta}_i$ is the Kronecker delta. The gradient of the Lagrangian can be then written as

$$\begin{aligned} \nabla_{\vec{x}} \mathbf{L} &= \nabla_{\vec{x}} \|\vec{x}\|_{\mathrm{TV}} + \lambda_0 \nabla_{\vec{x}} h_0(\vec{x}) + \sum_{i=1}^{N_{pixels}} \lambda_i \nabla_{\vec{x}} h_i(\vec{x}) = 0 \\ &= \nabla_{\vec{x}} \|\vec{x}\|_{\mathrm{TV}} + 2\lambda_0 A^T \cdot (A \cdot \vec{x} - \vec{b}) - \vec{\lambda} = 0 \end{aligned} \tag{3.21}$$

Further simplification can be applied to equation 3.21. As the non-negativity constrains are only active in zero valued voxels, the Lagrange multipliers are zero for strictly positive voxels. Thus, by adding an indicator function

$$\vec{x}_{indic} = \begin{cases} 1 & \vec{x} \neq 0 \\ 0 & \vec{x} = 0 \end{cases} \tag{3.22}$$

the Lagrangian gradient can be shortened to

$$\nabla_{\vec{x}}\mathbf{L} = \mathrm{diag}(\vec{x}_{indic})\left(\nabla_{\vec{x}}\|\vec{x}\|_{\mathrm{TV}} + \lambda_0 \nabla_{\vec{x}} h_0(\vec{x})\right) = 0. \tag{3.23}$$

Separating this new equation into two vectors,

$$\begin{aligned} \vec{d}_{\mathrm{TV}} &= \mathrm{diag}(\vec{x}_{indic})\left(\nabla_{\vec{x}}\|\vec{x}\|_{\mathrm{TV}}\right) \\ \vec{d}_{\mathrm{data}} &= \mathrm{diag}(\vec{x}_{indic})\left(\nabla_{\vec{x}} h_0(\vec{x})\right) \end{aligned} \tag{3.24}$$

brings to the Karush Kuhn-Tucker conditions: $\vec{x}$ will be an optimal condition if $\vec{d}_{\mathrm{TV}}$ and $\vec{d}_{\mathrm{data}}$ are pointing in exactly the opposite direction. In practice the algorithm will only check if the vectors are pulling in opposite direction (by computing the dot product) and that the inequality constrains are satisfied. By checking the direction of the vectors the algorithm ensures that even if the data constrain is satisfied, only the optimal solution regarding both TV norm and data fidelity is chosen.

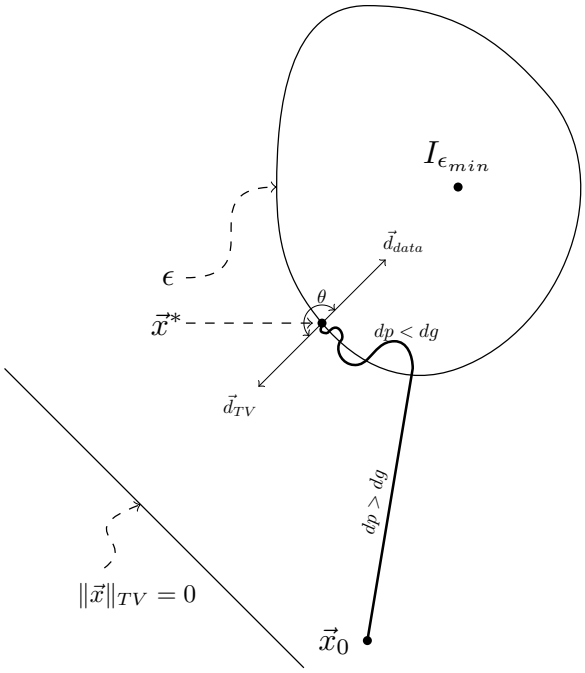

Figure 3-1: Conceptual diagram of the ASD-POCS algorithm path to the solution.

Figure 3-1 shows a conceptual diagram of the ASD-POCS algorithm. There is an area around the image with minimum data constrain, $I_{\epsilon_{min}}$. The solution $\vec{x}^*$ generally lies on the boundary of the area with the user specified ϵ. From an initial image $\vec{x}_0$, the algorithm walks towards the area of acceptable ϵ more strongly than towards the area of minimum TV, as the step sizes of the vectors $\vec{d}_{\text{TV}}$ and $\vec{d}_{\text{data}}$, dp and dg respectively, are adaptively controlled to be $dp > dg$. Once the image is within acceptable 2 norm, then the step size is changed in order to have stronger $\vec{d}_{\text{TV}}$ ($dp < dg$). The optimal solution can be found when both vectors point in opposing direction, or in other words, when the angle between them is 180 degrees, or the cosine of it is -1,

$$\cos\theta = \frac{\vec{d}_{\text{TV}} \cdot \vec{d}_{\text{data}}}{\|\vec{d}_{\text{TV}}\|\|\vec{d}_{\text{data}}\|}. \tag{3.25}$$

The pseudocode for the algorithm can be seen in algorithm 1. The algorithm is essentially solving the two vector in equation 3.24, the data vector in lines [5-8] and the TV vector in lines [18-22]. Line [9] enforces the positivity constrain. In the algorithm, dtv is initialized according to α, an user specified TV hyperparameter for TV, together with dp, the step size performed by the data constrain. After the TV minimization is performed, the step size of the TV vector is rechecked. If the TV minimization step is too big (bigger than the data step size), and the desired ϵ is still not achieved, the

step size is reduced further. This method of adaptively setting the step size of the TV iteration relating to the data step size is what ensures the optimal condition is achieved. Finally, the stopping criteria relies in either achieving the desired ϵ with with a desired $\cos\theta$, or stopping due to reaching a maximum amount of iterations (β decreases with iteration number). In the original proposition of the ASD-POCS algorithm (and shown here), the data constrain is solved using SART, however any other algorithm solving the same minimization problem can be used here (e.g. CGLS or OS-SART).

The algorithm has 7 parameters that need to be set up: β and β_{red} are the initial value and reduction ratio of the SART hyperparameter, similarly α and α_{red} serve as hyperparameter and reduction ratio for the TV minimization. r_{max} controls the maximum allowed ratio of change between the data minimization and the TV minimization, in order to adapt the step sizes. The number of iterations the TV minimization performs per iteration of the data minimization is defined as n_{TV}. Finally, the allowed data error is ϵ, as described before. The initial values of the variables in the algorithm are a key factor on its convergence. Empirical tests show that wrong parametrization of the algorithm can lead to severely noisy reconstructions. An study of the sensitivity of these parameters to changes has been performed by Lohvithee *et al*[CITE VEEs PAPER, at least as "future"]. The study shows that some parameters can be safely set up to a static value regardless of the data, such as the data hyperparameters, but that ϵ, n_{TV} and α are critical parameters to tune in order to get an usable reconstruction, and they are heavily data dependant. While some algorithms have successfully replaced the initial set of α by some data based heuristics [CITE PCSD][1] to the best of the authors knowledge there is no mathematical proposal for setting these parameters. The biggest drawback of this method is that several reconstructions may be needed to find the best parameters for an specific application.

Note that this minimization approach, while used for TV minimization in the original article, can be used for a variety of different minimization functions. For example, the TV minimization step could be replaced by a prior image minimization[CITE PICS], or any other convex minimization function. Similarly, the data minimization step can be replaced by any other minimization algorithm, as long as it minimizes the problem in equation XX

[1]These algorithms, namely PCSD and Aw-PCSD, are also available in TIGRE, by Manasavee Lohvithee.

Algorithm 1 ASD-POCS

Set: $\beta, \beta_{red}, n_{TViter}, \alpha, \alpha_{red}, r_{max}$
$\vec{x} = 0$;
while Stopping criteria not met **do**
    $\vec{x}_{prev} = \vec{x}$
    for n_{angles} **do**
        $\vec{x} = \vec{x} + \beta V A^T W^{-1}(\vec{b} - A\vec{x})$ ▷ SART update
    end for
    $\beta = \beta * \beta_{red}$
    $\vec{x} = \max(0, \vec{x})$ ▷ Enforce positivity
    $\vec{x}_{out} = \vec{x}$
    $\epsilon_{now} = \|A\vec{x} - \vec{b}\|$ ▷ Current ϵ
    $dp = \|\vec{x} - \vec{x}_{prev}\|$ ▷ Change in $\vec{d}_{\text{data}}$
    if first iteration **then**
        $dtv = \alpha * dp$ ▷ Initialize TV hyperparameter
    end if
    $\vec{x}_{prev} = \vec{x}$

    for n_{TViter} **do** ▷ TV update
        $\vec{dx} = \nabla_{\vec{x}} \|\vec{x}\|_{TV}$
        $\hat{dx} = \vec{dx}/\|\vec{dx}\|$
        $\vec{x} = \vec{x} - dtv \cdot \hat{dx}$
    end for
    $dg = \|\vec{x} - \vec{x}_{prev}\|$ ▷ Change in $\vec{d}_{\text{TV}}$
    if $dg > r_{max} * dp$ **and** $\epsilon_{now} > \epsilon$ **then**
        $dtv = dtv * \alpha_{red}$
    end if
    ▷ Check stopping criteria
    $\cos\theta = \vec{dp} \cdot \vec{dg}/\|\vec{dp}\| \cdot \|\vec{dg}\|$
    if $(\cos\theta < -0.9$ **and** $\epsilon_{now} < \epsilon)$ **or** $\beta < 0.005$ **then**
        Stop
    end if
end while

The gradient of the TV norm

In order to minimize the TV norm via gradient descend, the gradient of the TV norm needs to be computed, $\nabla_{\vec{x}}||\vec{x}||_{TV}$, being $\vec{x}$ the vectorized form of a N-dimensional image.

The main challenge wit the $\nabla_{\vec{x}}||\vec{x}||_{TV}$ term is that $||\vec{x}||_{TV}$ is not differentiable in the general case. However, in the CT case, $\vec{x}$ can be described as x_{ijk}, a regularly discretized mesh of directional indices i, j, k of maximum value $i_{max}, j_{max}, k_{max}$. The gradient of x has an additional Cartesian index α:

$$g^{\alpha} = (\nabla x)^{\alpha} = \partial_{\alpha} x \tag{3.26}$$

$$g^{\alpha}_{ijk} = \partial_{\alpha} x_{ijk}. \tag{3.27}$$

The TV norm can me then defined as sum of the 2-norms of the gradient of x, g, over the Cartesian coordinate, resulting in a scalar.

$$\|x\|_{\mathrm{TV}} = \sum_{ijk} \sqrt{\sum_{\alpha} \left(g^{\alpha}_{ijk}\right)^2} = \sum_{ijk} \sqrt{\sum_{\alpha} \left(\partial_{\alpha} x_{ijk}\right)^2}, \tag{3.28}$$

This is the term that the total variation regularization algorithm minimizes with a gradient descend. In order to perform this, the gradient of this term with respect to x is needed, now defined in a scalar field

$$(\nabla_{\vec{x}}||x||_{TV})_{ijk}. \tag{3.29}$$

This derivative can be expanded to a 3 component value for each x_{ijk} as:

$$\begin{aligned}
(\nabla_x \|x\|_{\mathrm{TV}})_{ijk} &= \frac{\partial}{\partial x_{ijk}} \|x\|_{\mathrm{TV}} = \partial_{x_{ijk}} \sum_{i'j'k'} \sqrt{\sum_{\alpha} \left(\partial_{\alpha} x_{i'j'k'}\right)^2} \\
&= \sum_{i'j'k'} \partial_{x_{ijk}} \sqrt{\sum_{\alpha} \left(\partial_{\alpha} x_{i'j'k'}\right)^2} \\
&= \sum_{i'j'k'} \frac{\sum_{\alpha} \left(\partial_{\alpha} x_{i'j'k'}\right) \partial_{x_{ijk}} \left(\partial_{\alpha} x_{i'j'k'}\right)}{\sqrt{\sum_{\alpha} \left(\partial_{\alpha} x_{i'j'k'}\right)^2}}.
\end{aligned} \tag{3.30}$$

This term now contains ∂_{α} derivatives, i.e. derivatives in the Cartesian coordinate system $[x, y, z]$. These are defined as

$$
\begin{aligned}
\partial_x x_{i'j'k'} &= \lim_{h\to 0} \frac{x_{i'+h,j',k'} - x_{i',j',k'}}{h} \\
\partial_y x_{i'j'k'} &= \lim_{h\to 0} \frac{x_{i',j'+h,k'} - x_{i',j',k'}}{h} \\
\partial_z x_{i'j'k'} &= \lim_{h\to 0} \frac{x_{i',j',k'+h} - x_{i',j',k'}}{h}.
\end{aligned} \tag{3.31}
$$

However, x is discrete, thus the limit definition of the derivative can not be used to numerically compute it, but an approximation of it can. By setting $h = 1$, equation 3.31 becomes the backward finite differences of the first order approximation of a derivative, a very computationally cheap operation. The derivative w.r.t. the Cartesian coordinate can be rewritten as

$$
\begin{aligned}
\partial_\alpha x_{i'j'k'} = \delta_{\alpha x}\left(x_{i',j',k'} - x_{i'-1,j',k'}\right) + \delta_{\alpha y}\left(x_{i',j',k'} - x_{i',j'-1,k'}\right) \\
+ \delta_{\alpha z}\left(x_{i',j',k'} - x_{i',j',k'-1}\right)
\end{aligned} \tag{3.32}
$$

where δ_α is a Kronecker delta for the Cartesian axis. The other partial derivative term that appears in equation 3.30 is $\partial_{x_{ijk}}\left(\partial_\alpha x_{i'j'k'}\right)$. As the derivative is w.r.t. x_{ijk}, each component of x is an independent variable, thus $\partial_{x_{ijk}}\left(\partial_\alpha x_{i'j'k'}\right)$ is zero everywhere but in indices $i = i' \wedge j = j' \wedge k = k'$, where the derivative is 1. The term then becomes

$$
\begin{aligned}
\partial_{x_{ijk}}\partial_x x_{i'j'k'} &= \partial_{x_{ijk}}\left(x_{i',j',k'} - x_{i'-1,j',k'}\right) = \delta_{i',i}\delta_{j',j}\delta_{k',k} - \delta_{i'-1,i}\delta_{j',j}\delta_{k',k} \\
&= \delta_{i',i}\delta_{j',j}\delta_{k',k} - \delta_{i',i+1}\delta_{j',j}\delta_{k',k} \\
\partial_{x_{ijk}}\partial_y x_{i'j'k'} &= \partial_{x_{ijk}}\left(x_{i',j',k'} - x_{i',j'-1,k'}\right) = \delta_{i',i}\delta_{j',j}\delta_{k',k} - \delta_{i',i}\delta_{j'-1,j}\delta_{k',k} \\
&= \delta_{i',i}\delta_{j',j}\delta_{k',k} - \delta_{i',i}\delta_{j',j+1}\delta_{k',k} \\
\partial_{x_{ijk}}\partial_z x_{i'j'k'} &= \partial_{x_{ijk}}\left(x_{i',j',k'} - f_{i',j',k'-1}\right) = \delta_{i',i}\delta_{j',j}\delta_{k',k} - \delta_{i',i}\delta_{j',j}\delta_{k'-1,k} \\
&= \delta_{i',i}\delta_{j',j}\delta_{k',k} - \delta_{i',i}\delta_{j',j}\delta_{k',k+1}.
\end{aligned} \tag{3.33}
$$

These terms are practically a selecting function for i', j', k' ,matching only in the indices $i, i+1, j, j+1, k, k+1$ in the sum of the right hand side of equation 3.30. However the indices are limited to $i' \in [1, i_{max}]$, $j' \in [1, j_{max}]$ and $k' \in [1, k_{max}]$. As boundary conditions, Neumann boundary conditions are set to zero. To enforce that, a Kronecker deltas can be introduced for each index, $(1 - \delta_{i,i_{max}})$, with the same approach with the other indices.

Finally, substituting in equation 3.30, the gradient of the TV norm can be described as

$$
\begin{aligned}
(\nabla_x \|x\|_{\mathrm{TV}})_{ijk} &= \sum_{i'j'k'} \frac{\sum_\alpha \left(\partial_\alpha x_{i'j'k'}\right) \partial_{x_{ijk}} \left(\partial_\alpha x_{i'j'k'}\right)}{\sqrt{\sum_\alpha \left(\partial_\alpha x_{i'j'k'}\right)^2}} \\
&= \sum_{i'j'k'} \frac{\partial_x x_{i'j'k'} \partial_{x_{ijk}} \left(\partial_x x_{i'j'k'}\right) + \partial_y x_{i'j'k'} \partial_{x_{ijk}} \left(\partial_y x_{i'j'k'}\right) + \partial_z x_{i'j'k'} \partial_{x_{ijk}} \left(\partial_z x_{i'j'k'}\right)}{\sqrt{\sum_\alpha \left(\partial_\alpha x_{i'j'k'}\right)^2}} \\
&= \frac{\partial_x x_{i,j,k} + \partial_y x_{i,j,k} + \partial_z x_{i,j,k}}{\sqrt{\sum_\alpha \left(\partial_\alpha x_{i,j,k}\right)^2}} \\
&- \frac{\left(1-\delta_{i,i_{max}}\right) \partial_z x_{i+1,j,k}}{\sqrt{\sum_\alpha \left(\partial_\alpha x_{i+1,j,k}\right)^2}} - \frac{\left(1-\delta_{j,j_{max}}\right) \partial_y x_{i,j+1,k}}{\sqrt{\sum_\alpha \left(\partial_\alpha x_{i,j+1,k}\right)^2}} - \frac{\left(1-\delta_{k,k_{max}}\right) \partial_z x_{i,j,k+1}}{\sqrt{\sum_\alpha \left(\partial_\alpha x_{i,j,k+1}\right)^2}}.
\end{aligned} \tag{3.34}
$$

Equation 3.34 is the numerical approximation of the gradient of the total variation norm, and describes scalar field of the same size of the image. The same approach can be used with central and forward differences to obtain a similar equation, however central differences may not correctly minimize the TV norm of the image. As central differences do not take into account the value of the current voxel ijk, a chequerboard pattern would have zero TV norm, and this is the opposite of the purpose of the algorithm, therefore only numerical approximations of derivatives that take immediately adjacent pixel values into account can be used (such as forward or backward finite differences).

3.3.5 Total variation regularization via Rudin-Osher-Fatemi model

A different minimization approach to POCS is the approach proposed by Jia *et al*[7], that uses the Rudin-Osher-Fatemi (ROF) model for total variation minimization, widely used in the denoising literature[15][5][19]. By starting from the same minimization problem, namely

$$
\hat{x} = \arg\min_x \|Ax - b\|^2 + \lambda \|x\|_{TV}, \tag{3.35}
$$

and a forward-backward splitting algorithm[4] is used to split the minimization into two alternating steps, the TV and the data steps. If the optimality condition is considered to be

$$
\frac{\partial}{\partial x_\alpha} \|Ax - b\|^2 + \lambda \frac{\partial}{\partial x_\alpha} \|x\|_{TV} = 0, \tag{3.36}
$$

being α the set of Cartesian coordinates, then the problem can be split into the following equations, where g is a auxiliary function and $\mu > 0$:

$$\lambda \frac{\partial}{\partial x_\alpha} \|x\|_{TV} = \mu(x - g) \tag{3.37}$$

$$\frac{\partial}{\partial x_\alpha} \|Ax - b\|^2 = -\mu(x - g). \tag{3.38}$$

By solving for g, the simplified version of the algorithm can be seen in 2.

Algorithm 2 TV minimization with ROF model

1: Solve: $g = x - \frac{\lambda}{\mu} \frac{\partial}{\partial x_\alpha} \|Ax - b\|^2$ ▷ SART
2: Minimize: $x = \arg\min_x \|x\|_{TV} + 0.5 * \mu \|x - g\|^2$
3: Enforce positivity: $x = \max(0, x)$

The first line of the algorithm its essentially a gradient descend iteration, which is essentially a SART iteration. Note that the SART iteration can be replaced by other data-minimization algorithms such as CGLS. The second line is the ROF model, widely researched in image denoising. The ROF model tries to find the image x with minimum total variation subject to having the minimal deviation from its original value g. By changing the value of he hyperparameter μ, the strength of this regularization is controlled. A high μ will ensure that the image is very similar to its original value, while a small μ will be more lax. The advantage of this approach compared to the ASD-POCS algorithm is that it requires no extra projection or backprojection operations. Additionally, minimizing the ROF model is a very well studied problem in the image processing field, and it has lead to finding computationally efficient methods.

In the article by Jia *et al*, they solve the ROF model via gradient descend and controlling its step size with Armijo's rule. In this work a different approach is taken, based on the image processing literature.

Primal Dual formulation of the ROF model

As previously shown in line 2 of algorithm 2, the ROF model can be formulated as

$$\hat{x}_{ROF} = \arg\min_x \|x\|_{TV} + \frac{\mu}{2} \|x - g\|^2. \tag{3.39}$$

A solution of this problem using a primal-dual (PDU) approach has been proposed in literature[20], by changing the minimization equation to a saddle point optimization problem. While a wide variety of methods have been proposed to minimize the ROF model[15][19][3], the PDU method has the advantage of being very parallelizable, thus

a perfect fit for GPU computing. The dual variable can be proposed by using the TV definition of $\|x\|_{TV} = \|\nabla x\|$ and observing the following consequence of the Cauchy-Schwartz inequality

$$\|\nabla x\| = \arg\max_{\|\mathbf{p}\|\leq 1} \|\mathbf{p}\nabla x\|, \tag{3.40}$$

where $\mathbf{p} = (p^1, p^2, p^3)^T$ (for the 3D case) is the said dual variable. Note that each p^i is the size of the image x. Equation 3.39 can be then rewritten as

$$\hat{x}_{ROF} = \arg\min_{x} \arg\max_{\|\mathbf{p}\|\leq 1} \|\mathbf{p}\nabla x\| + \frac{\mu}{2}\|x - g\|^2. \tag{3.41}$$

The primal and dual updates can be both obtained from this equation. For the primal update, differentiating the equation according to x results in

$$-\nabla \cdot \mathbf{p} + \mu(x - g) = 0, \tag{3.42}$$

and one can solve it for x by performing a gradient descend update as

$$x^{n+1} = x^n(1 + \tau_P^n) + \tau_P^n \left(g + \frac{1}{\mu}\nabla \cdot \mathbf{p} \right), \tag{3.43}$$

where τ_P is the primal step size. The dual update can be computed similarly, by differentiating equation 3.41 according to $\mathbf{p}$, the following equation si obtained:

$$\nabla x + \mathbf{p}\alpha = 0, \tag{3.44}$$

where α is a Lagrange multiplier for the inequality constrain $\|\mathbf{p}\| \leq 1$. This equation can be maximized with a gradient ascend method as

$$\mathbf{p}^{n+1} = \Pi_{B_0} \left(\mathbf{p}^n + \tau_D^n \nabla x \right), \tag{3.45}$$

where $\Pi_{B_0}(\mathbf{p}) = \frac{\mathbf{p}}{\max\{1, \|\mathbf{p}\|\}}$ is a projection onto the unit ball centred in the origin.

The PDU algorithm consists in updating $\mathbf{p}$ and x iteratively, by alternating the updates. In [20][11] a step size update is proposed for the primal and dual step sizes:

$$\begin{aligned} \tau_D^n &= 0.3 + 0.02n \\ \tau_P^n &= \frac{1}{\tau_D^n}\left(\frac{1}{6} - \frac{5}{15 + n} \right). \end{aligned} \tag{3.46}$$

The same update is used in this work, as the images in their work are structurally

similar to CT images and empirical test showed satisfactory results. The algorithm can be shown to converge as it is shown in [21] that the primal-dual gap decreases with each update of x^n, and the gap is suggested as a control variable for the stopping criteria. In this work the algorithm has been implemented without the stopping criteria check, and an user specified parameter for the number of iterations is passed as an input, with a default value of 50, as it empirically showed good results.

The discretization of the divergence and gradient operators are a key factor when numerically computing the PDU algorithm, as they need to be consequent with each other. Thus, the gradient can be approximated using forward differences, but as the divergence is the adjoint of the gradient, it must be approximated with backwards differences.

3.4 Discussion

Chapter 4

Experiments and Applications

In the previous two chapters the mathematical and computational challenges of image reconstruction for CT have been discussed. In chapter 3, a detailed description of a variety of different algorithms has been presented, including the ART family of algorithms, CGLS, MLEM and few TV approaches for smooth reconstruction, as well as the classic FDK reconstruction. Additionally in chapter 4 the computational aspect of CT is discussed, on where the problems computing the exact adjoint of the projection operation and mainly the computational burden of some of the operations have been mentioned. Considering the variety of available methods and the specifics of the implementation of the software developed, the TIGRE toolbox, experiments on how these algorithms compare and behave are due. Further than that, the performance of these algorithm in different experimental datasets is also an important analysis.

This chapter shows experimental analysis on both of the topics. First a variety of convergence analysis with different algorithms using synthetic data are performed, showing the differences not only between algorithms, but also between option on parameter selections. The section tries to illustrate and perhaps help build intuition on all the different parameters and options that each of these algorithms has, both within the algorithms themselves and among the different ones. Additionally some highlights on the practical challenges that the use of the algorithms entail in real applications are given.

In the second section of this chapter, a few examples of some of the algorithms are shown in different CT applications, both cone and parallel beam. Data from various different applications, from medicine to science has been tested using the TIGRE toolbox. While quantitative analysis is not possible with these datasets because the truth is not known, some insight in how the algorithms behave in each case is discussed.

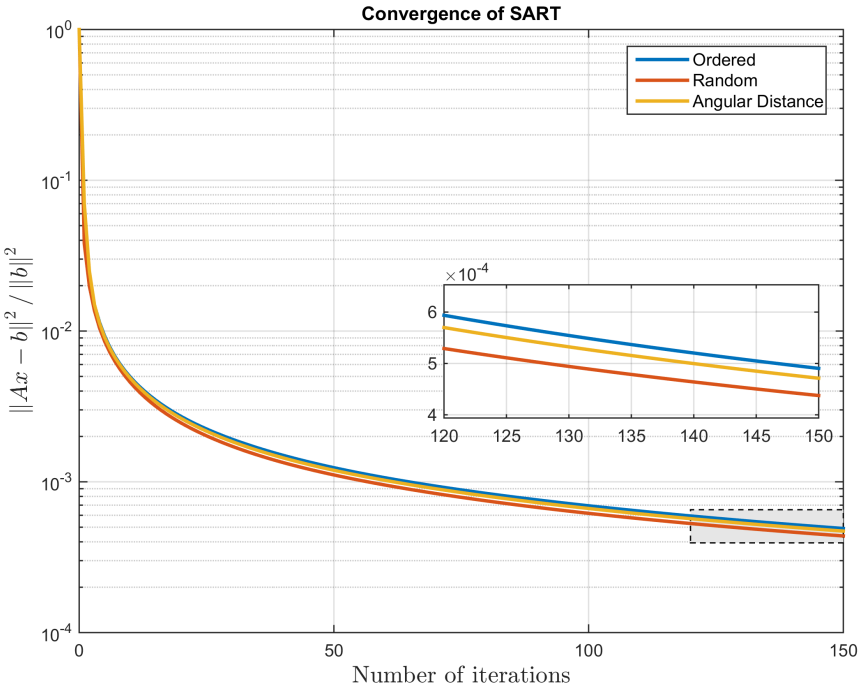

Figure 4-1: qweqwe.

4.1 Algorithm Experiments

4.2 Iterative Algorithms in Different CT Applications

4.2.1 Medical Head CBCT from The Christie Hospital

4.2.2 Cryo Soft X-Ray Tomography at Diamond Light Source

Chapter 5

Tolerance analysis of motion correction

Chapter 6

Conclusions and Future work

Bibliography

[1] A.H. Andersen and A.C. Kak. Simultaneous algebraic reconstruction technique (SART): a superior implementation of the ART algorithm. *Ultrasonic imaging*, 6(1):81–94, 1984.

[2] Y. Censor and T. Elfving. Block-iterative algorithms with diagonally scaled oblique projections for the linear feasibility problem. *SIAM Journal on Matrix Analysis and Applications*, 24(1):40–58, 2002.

[3] Antonin Chambolle. An algorithm for total variation minimization and applications. *Journal of Mathematical imaging and vision*, 20(1):89–97, 2004.

[4] Patrick L Combettes and Valérie R Wajs. Signal recovery by proximal forward-backward splitting. *Multiscale Modeling & Simulation*, 4(4):1168–1200, 2005.

[5] Joan Duran, Bartomeu Coll, and Catalina Sbert. Chambolle's projection algorithm for total variation denoising. *Image processing on Line*, 2013:311–331, 2013.

[6] G. T. Herman and L. B. Meyer. Algebraic reconstruction techniques can be made computationally efficient [positron emission tomography application]. *IEEE Transactions on Medical Imaging*, 12(3):600–609, Sep 1993.

[7] Xun Jia, Yifei Lou, John Lewis, Ruijiang Li, Xuejun Gu, Chunhua Men, William Y Song, and Steve B Jiang. Gpu-based fast low-dose cone beam ct reconstruction via total variation. *Journal of X-ray science and technology*, 19(2):139–154, 2011.

[8] Ming Jiang and Ge Wang. Convergence studies on iterative algorithms for image reconstruction. *IEEE Transactions on Medical Imaging*, 22(5):569–579, 2003.

[9] S. Kaczmarz. Angenäherte Auflösung von Systemen linearer Gleichungen. *Bulletin International de l'Académie Polonaise des Sciences et des Lettres*, 35:355–357, 1937.

[10] Avinash C. Kak, Malcolm Slaney, IEEE Engineering in Medicine, and Biology Society. *Principles of computerized tomographic imaging*. IEEE Press, New York, 1988. Published under the sponsorship of the IEEE Engineering in Medicine and Biology Society.

[11] Florian Knoll, Markus Unger, Clemens Diwoky, Christian Clason, Thomas Pock, and Rudolf Stollberger. Fast reduction of undersampling artifacts in radial MR angiography with 3D total variation on graphics hardware. *Magnetic resonance materials in physics, biology and medicine*, 23(2):103–114, 2010.

[12] Ji Liu and Stephen Wright. An accelerated randomized kaczmarz algorithm. *Mathematics of Computation*, 85(297):153–178, 2016.

[13] Yurii Nesterov. A method of solving a convex programming problem with convergence rate o (1/k2). In *Soviet Mathematics Doklady*, volume 27, pages 372–376, 1983.

[14] Sebastian Ruder. An overview of gradient descent optimization algorithms. *CoRR*, abs/1609.04747, 2016.

[15] Leonid I. Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena*, 60(1):259 – 268, 1992.

[16] E.Y. Sidky and X. Pan. Image reconstruction in circular cone-beam computed tomography by constrained, total-variation minimization. *Physics in Medicine and Biology*, 53(17):4777, 2008.

[17] Thomas Strohmer and Roman Vershynin. A randomized kaczmarz algorithm with exponential convergence. *Journal of Fourier Analysis and Applications*, 15(2):262–278, 2009.

[18] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147, 2013.

[19] Curtis R Vogel and Mary E Oman. Iterative methods for total variation denoising. *SIAM Journal on Scientific Computing*, 17(1):227–238, 1996.

[20] Mingqiang Zhu and Tony Chan. An efficient primal-dual hybrid gradient algorithm for total variation image restoration.

[21] Mingqiang Zhu, Stephen J Wright, and Tony F Chan. Duality-based algorithms for total-variation-regularized image restoration. *Computational Optimization and Applications*, 47(3):377–400, 2010.