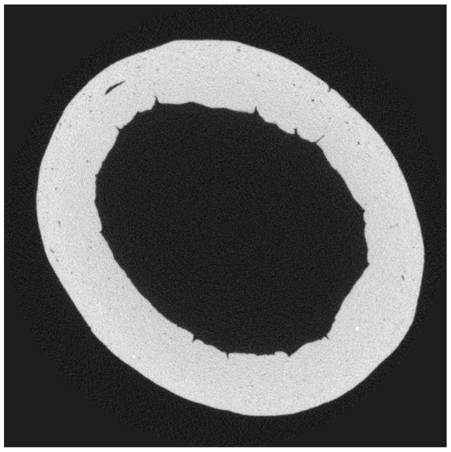
Utrecht University

A Discrete Ginzburg-Landau Functional for Regularised Tomographic Image Reconstruction



Author
Jinhan Wu
2387182

Supervisor
Dr. T. van Leeuwen
CWI & Mathematical Institute, UU
Second Reader
Dr. P. Salanevich
Mathematical Institute, UU

A thesis submitted in partial fulfilment of the degree of Master of Science in Mathematical Sciences.

Mathematical Institute, Faculty of Science,
Utrecht University, The Netherlands,
Saturday 2nd September, 2023

Abstract

Tomographic imaging has a wide range of applications including in medicine and in industry because of its non-destructive nature. Ideally, accurate images can be retrieved using high-quality fully sampled computed tomography (CT) scan data. In practice, the images are reconstructed from noisy subsampled data. One way to acquire good quality reconstructions is to encode prior information to the reconstruction algorithm.

In this project, we consider the discrete (graph) Ginzburg-Landau (GL) functional regularisation to express such prior information. We study the use of graph GL regularisation in binary image denoising and binary CT image reconstruction. For the binary image denoising problem, we examine the influence of various graph definitions on the performance of the graph GL functional. For tomographic image reconstruction problem, we compare graph GL regularisation with two other well-known methods, total variation (TV) regularisation and simultaneous iterative reconstruction technique (SIRT), in various noise levels and downsampled projection settings. Graph GL regularisation shows good performance with limited and high-noise data. Moreover, we collect our own CT scan data, and investigate how it might be applied in practical scenarios.

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Dr. Tristan van Leeuwen, for his guidance, suggestions, and support throughout this project. This thesis would not have been possible without him.

I would also like to thank my colleagues in the Computational Imaging research group at CWI for welcoming me to the group. I would like to especially thank Alex and Vlad for their suggestions and explanations, and Felix for helping me make the CT scans. Additionally, I would like to express gratitude to Dr. Yves van Gennip and Dr. Monique Laurent for their valuable input.

Finally, I would like to thank my family and friends for their support and encouragement throughout my studies.

Contents

Notation

Numbers

n	Dimension of the 2D image as a vector, $\sqrt{n} \in \mathbb{N}$
m	Dimension of the sinogram as a vector
N	Number of images in the dataset
K	Number of iterations in proximal gradient descent
L_G	Lipschitz constant for ∇G
θ	Angular increment for CT projections
α	Stepsize in proximal gradient descent, hyperparameter
ε	Regularisation parameter in graph GL functional, hyperparameter
β	Regularisation parameter in GL-3SR, hyperparameter

Functions

$F : \mathbb{R}^n \mapsto \mathbb{R}$	Objective function
$R : \mathbb{R}^n \mapsto \mathbb{R}$	Regularisation function
$W : \mathbb{R} \mapsto \mathbb{R}$	Double-well potential

Matrices/Vectors

$\mathbf{K} \in \mathbb{R}^{m \times n}$	Discrete Radon transform
$\mathbf{L} \in \mathbb{R}^{n \times n}$	Graph Laplacian matrix
$\mathbf{L}_4 \in \mathbb{R}^{n \times n}$	4-regular graph Laplacian matrix
$\mathbf{L}_p \in \mathbb{R}^{n \times n}$	Learned graph Laplacian matrix
$\mathbf{v} \in \mathbb{R}^n$	Noisy image (reshape into 2D when plotting)
$\bar{\mathbf{u}} \in \{0,1\}^n$	Ground truth (reshape into 2D when plotting)
$\hat{\mathbf{u}} \in \mathbb{R}^n$	Predicted image (reshape into 2D when plotting)
$\hat{\mathbf{u}}^b \in \{0,1\}^n$	Binary version of the predicted image (reshape into 2D when plotting)
$\mathbf{f} \in R^m$	Sinogram, also called CT scan measurements (reshape into 2D when plotting)
$\mathbf{f}^{\boldsymbol{\delta}} \in R^m$	Noisy sinogram (reshape into 2D when plotting)

1 Introduction

Computed tomography (CT) is a technique that generates images of the internal structure of the object being scanned. Due to its non-invasive nature, tomographic imaging has a wide range of applications, such as medical imaging at hospitals, luggage security screening at airports, and flaw detection in industries. In a typical CT scan, an object is placed in between an X-ray source and an array of detectors. The X-ray source and the detector array rotate around a fixed axis of the object, and detectors measure the attenuation of X-rays as they pass through the object from different angles. Parallel-beam and fan-beam are the two most prevalent types of beam geometry emitted by the X-ray source. Figure 1.1 illustrates the two geometries.

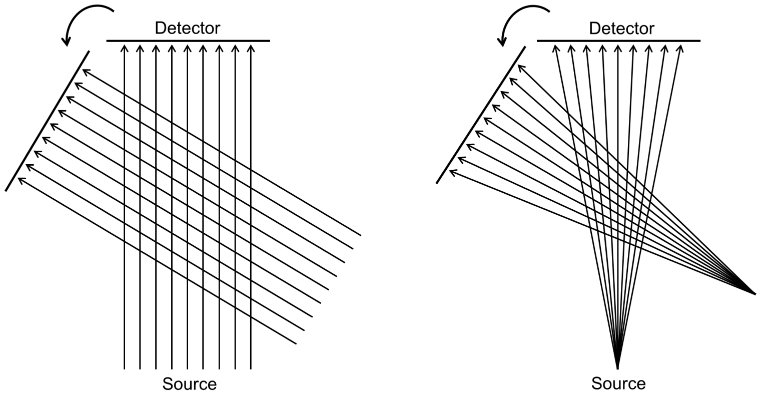


Figure 1.1: Illustration of beam geometries. Left: parallel-beam geometry. Right: fan-beam geometry with flat detector.

Following the acquisition of the CT measurements, a tomographic image is generated utilising certain reconstruction algorithm. Mathematically, the collection of the CT measurements can be described by Radon transform. Johann Radon originally attempted to reconstruct a real-valued function of two variables from its line integrals over all angles in $[0, 360°]$ (Radon, 1986). It is widely acknowledged that his findings laid the groundwork for CT.

In this project, we focus on the reconstruction of discrete binary images. Binary images consist of pixels with two possible values: 0 for black and 1 for white. Binary CT image reconstruction is useful when the primary goal is to separate two objects in an image. For example, in (Meng, Wang, & Xing, 2010), a method for detecting metal objects in CT images is developed by reconstructing images with pixels where the pixels representing metal have value 1 and the remaining pixels have value 0.

If we view obtaining the CT measurements as a forward problem, then recovering the underlying ground truth image from the measured data is its inverse problem. As a consequence of limited data and experiment deficiencies, these inverse problems are often ill-posed (Antil, Di, & Khatri, 2020). Various reconstruction approaches were developed over the years. They can be classified into three categories: analytical, algebraic and variational approaches. The analytical approach includes filtered back projection (FBP) (Ramachandran & Lakshminarayanan, 1971). FBP is fast but it needs sufficient data and does not tolerate much noise in the CT measurements. The algebraic approaches, e.g. simultaneous iterative reconstruction technique (SIRT) (Andersen & Kak, 1984), involve discretising the Radon transform and solve the resulting system iteratively. Such algebraic approaches outperform the analytical approach when dealing with limited data. However, in some circumstances, we would like to encode prior information on expected image structures. For instance, one important prior knowledge in our project is the images being reconstructed consisting 2 pixel values. Therefore, in this project we focus on CT image reconstruction with the variational approach, as it employs regularisation to reconstruct the ground truth image, allowing us to incorporate such prior knowledge.

One popular choice of regularisation terms is the total variation (TV) regularisation(Chambolle & Lions, 1997). The total variation has been extensively studied (Candès, Romberg, & Tao, 2006; Candes, Romberg, & Tao, 2006) and shows good performance on restoring images corrupted by Gaussian noise (Rudin, Osher, & Fatemi, 1992). However, the usage of this regulariser is restricted as it cannot be differentiated. In recent years, Ginzburg-Landau (GL) functional has been considered as an alternative to the TV functional in the image processing literature, since it is differentiable and it Γ-converges to the total variation functional (Modica, 1987). Furthermore, by viewing an image as a graph where each node represents a pixel, the GL functional has been extended to a graphical framework (Bertozzi & Flenner, 2012). It is proved that similar Γ-convergence result holds for the graph GL functional (van Gennip & Bertozzi, 2012). The graph GL functional consists of a graph Laplacian term and a double-well potential term, with two wells located at 0 and 1. This makes the graph GL particularly suitable for our task of reconstructing binary images, as the double-well potential pushes the pixel values to either 0 or 1. Moreover, since the graph Laplacian matrix in graph GL functional depends on how the graph is defined, it allows for additional modelling flexibility.

It is worth mentioning that the graph Laplacian term in the graph GL functional can be utilised as regularisation on its own. Despite the fact that the graph Laplacian regulariser is a relatively new prior, many empirical results demonstrate that it is effective in image denoising (Pang & Cheung, 2017). It can be shown that minimising the graph Laplacian regulariser results in a smooth graph (Shuman, Narang, Frossard, Ortega, & Vandergheynst, 2013). Additionally, it is proved analytically that the graph Laplacian regulariser converges to the anisotropic Dirichlet energy (Pang & Cheung, 2017). The graph Laplacian regularisation offers us an alternative way of interpreting the graph GL regularisation. Consider we want to recover binary images from CT measurements with the graph Laplacian regulariser. In contrast to adding constraints on the pixel values such that they have to be binary, a double-well potential term can be added to the objective function, forcing the pixels values to be in the two minima of the double-well potential. These two terms, the graph Laplacian regulariser and the double-well potential, form the graph GL functional.

In this project, we investigate the usage of the graph GL functional as a regulariser. As the graph GL depends on the choice of the graph Laplacian matrices, we compare the performance of the proposed model when using various graph Laplacian matrices. This includes the 4-regular graph Laplacian and a graph Laplacian learned from correlated image data, i.e. we employ an algorithm to decide how the graph should be defined based on the pixel values in the data. In this project, such algorithm is chosen to be GL-3SR (Humbert, Le Bars, Oudre, Kalogeratos, & Vayatis, 2021), where the learned graph Laplacian matrix is sparse. Other related graph Laplacian learning methods, which are mostly developed in the graph signal processing literature, include GL-SigRep (Dong, Thanou, Frossard, & Vandergheynst, 2016) and DeepGLR (Zeng, Pang, Sun, & Cheung, 2019).

In this project, we begin with considering a simplified model of CT image reconstruction, where we use the graph GL regularisation to solve image denoising problems. We first find a suitable minimisation method to solve such optimisation model, then investigate the model performance with different graph Laplacian matrices on image denoising with two datasets containing handwritten digits images. Then, we return to the CT image reconstruction problem, where we compare our proposed model, the graph GL regularisation, with various reconstruction approaches such as SIRT and TV regularisation. The experiment is carried out in various number of projections and noise levels with simulated CT measurements for images of the handwritten number 0. Furthermore, in order to test our model in real world, we made our own CT scan of a playdough number 0.

The thesis is structured as follows. Section 2 introduces the relevant theories that underpin our model. The topics covered include the mathematics of CT, the proposed CT image reconstruction model, i.e. variational approach with graph GL functional as regularisation, the minimisation methods used to solve the model, and graph Laplacian learning methods. Section 3 and Section 4 show the results and discussion of the experiments on image denoising and CT image reconstruction with graph GL regulariser. It is then followed by the conclusions and possible future work in Section 5.

The experiments were performed a computer running the operating system Microsoft Windows 10, with processor 11th Gen Intel(R) Core(TM) i5-11300H @ 3.10GHz and installed RAM 16.0GB. The code is written in Python and can be found on Github `https://github.com/jhwuuu/Image-Reconstruction-with-Graph-GL-Functional`.

2 Methodology

In this section we discuss the theory behind our model. Section 2.1 introduces the mathematical theory underlying computed tomography. This section is based on (Hansen, Jørgensen, & Lionheart, 2021) and is inspired by (Tatiana A. Bubba, 2019). Section 2.2 discusses CT image reconstruction via variational approach, where the image is recovered by solving a minimisation problem with an objective function consisting of a data-fidelity term and a regularisation term. The algorithms employed to solve the minimisation problem are also described. Section 2.4 presents the regulariser used in this thesis, the so called graph Ginzburg-Landau functional. Finally, Section 2.6 offers an overview of the experiments carried out in this thesis, and the assessments used to evaluate the model performance.

2.1 Mathematics of Computed Tomography (CT)

Computed tomography is a non-destructive imaging technique commonly used in healthcare and industries. It measures the intensity loss of X-ray beams as they pass through the object being scanned, then a computational algorithm is used to reconstruct the image of the object based on the X-ray measurements. For simplicity, we introduce the idea of CT using the parallel-beam geometry. It is important to note that in practice, other beam geometries are more commonly used.

2.1.1 X-ray and Lambert-Beer Law

First, we provide the formulation for the intensity decay of a single X-ray beam after passing through an object. This is described by the Lambert-Beer law.

Definition 2.1. (Lambert-Beer law) Consider an X-ray detection setup illustrated in Figure 2.1. Denote I_0 the initial intensity of the X-ray beam ℓ, and let I_1 be the beam intensity after traversing an object. Let $\mu(x)$ be the X-ray attenuation of the object at the point x on ℓ. The intensity I_1 is of the form

$$I_1 = I_0 \exp\left(-\int_\ell \mu(x)dx\right). \tag{2.1}$$

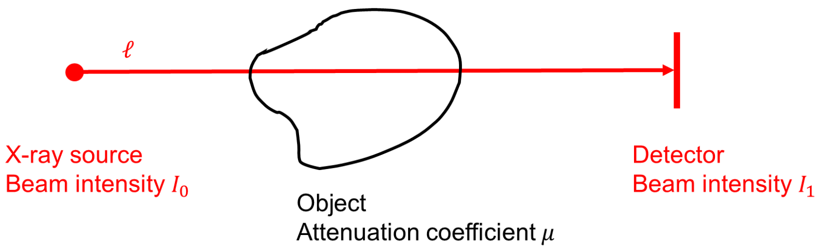


Figure 2.1: Illustration of an X-ray beam intensity detection after passing through an object.

Rearranging (2.1) into line-integral form, we obtain

$$b = -\log\frac{I_1}{I_0} = \int_\ell \mu(x)dx.$$

The quantity b is called the *absorption* or *projection*.

2.1.2 The Radon Transform

Having described the projection for a single X-ray beam, we now present how CT combines the projections from different angles to reconstruct the object. This is achieved with Radon transformation.

In the two-dimensional setting, the Radon transform is the integral transform that maps a function defined in $\mathbb{R}^2$ to a function defined on the space of lines in $\mathbb{R}^2$.

First, we show how to characterise the lines in the (x_1, x_2) Cartesian coordinate system. Given an angle φ, we define the two orthogonal vectors

$$\boldsymbol{\varphi} = \begin{pmatrix} \cos\varphi \\ \sin\varphi \end{pmatrix}, \quad \boldsymbol{\varphi}^{\perp} = \begin{pmatrix} -\sin\varphi \\ \cos\varphi \end{pmatrix}.$$

Then, the line that is orthogonal to $\boldsymbol{\theta}$ and is located distance s away from the origin, denoted by $L_{\varphi,s}$, can be expressed as

$$L_{\varphi,s} = \{\mathbf{x} \in \mathbb{R}^2 \mid \mathbf{x} \cdot \boldsymbol{\varphi} = s\}, \quad \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix},$$

where $\mathbf{x} \cdot \boldsymbol{\theta} = x_1 \cos\varphi + x_2 \sin\varphi$. Line $L_{\varphi,s}$ is depicted in Figure 2.2 (left).

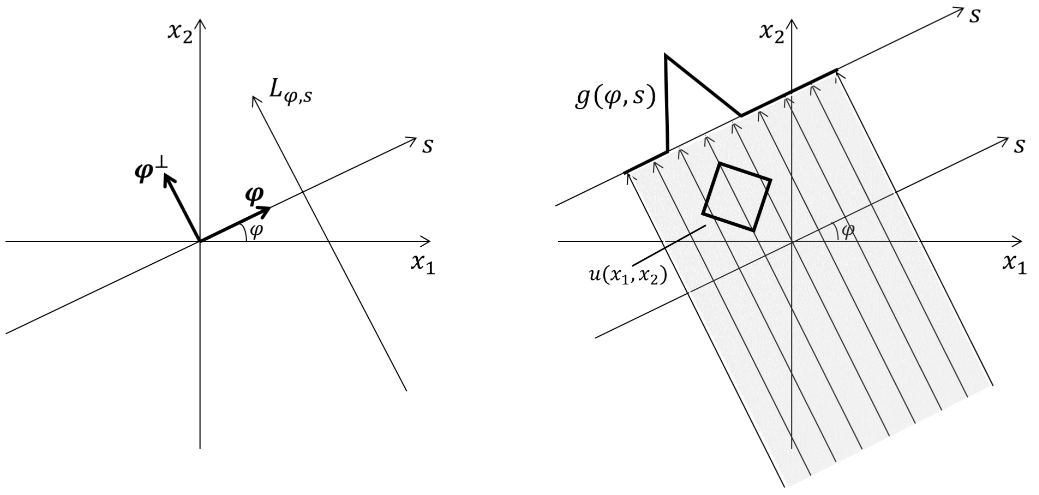


Figure 2.2: Left: Illustration of a line $L_{\varphi,s}$ in the (x_1, x_2) coordinate. Right: The projection $g(\varphi, s)$ of an object using parallel X-ray beams at the angle φ described by $u(x_1, x_2)$.

Moreover, given a fixed φ, the projection along $L_{\varphi,s}$ is of the form

$$g(\varphi, s) = \int_{L_{\varphi,s}} u(x_1, x_2) d\mathbf{x}, \qquad s \in \mathbb{R}.$$

Figure 2.2 (right) demonstrates an example of the projection $g(\varphi, s)$ of the object $u(x_1, x_2)$ at angle φ, with s restricted to some interval on $\mathbb{R}$.

Then, the Radon transform of u is defined as the projections at all angles, i.e.

$$R[u](\varphi, s) = g(\varphi, s) = \int_{L_{\varphi,s}} u(x_1, x_2) d\mathbf{x}, \qquad (\varphi, s) \in [0, 360^\circ) \times \mathbb{R}.$$

This collection of the projections, i.e. the output of the Radon transform $g(\varphi, s) = R[u](\varphi, s)$ for all angles φ, is called the *sinogram*. Figure 2.3 illustrates a simple image and its sinogram.

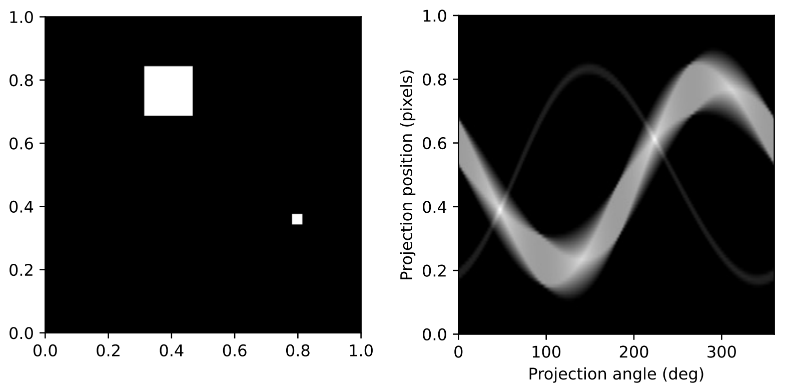


Figure 2.3: Left: A 64×64 pixel image with two blocks having pixel value 1. Other pixels are of value 0. Right: The resulting sinogram obtained by taking the Radon transform, projections made at every 1°.

As mentioned in Section 2.1.1, the Lambert-Beer law can be rearranged into the line-integral form. Therefore, Radon transform and the Lambert-Beer law can be linked by setting

$$u(x_1, x_2) = \mu(x_1, x_2),$$
$$g(\varphi, s) = -\log\left(\frac{I_{\varphi,s}}{I_0}\right),$$

which gives

$$I_{\varphi,s} = I_0 \exp\left(-\int_{L_{\varphi,s}} \mu(x_1, x_2) d\mathbf{x}\right). \tag{2.2}$$

Equation (2.2) allows us to characterise X-rays measurements at the detector at any angle and position.

2.1.3 The Discrete Model

As we are primarily interested in recovering images from their CT scan data, the objects being reconstructed will now be referred to as images. We have seen how to express X-ray measurements of an image from all possible angles in theory. In practice, however, we only have a finite number of angles and detectors. As a consequence, we discretise the images and the line integrals, and construct a system of equations.

The images are naturally discretised by pixels. We denote $\mathbf{u}$ the discrete image and let $\mathbf{u}_i$ be the pixel value of the image at pixel i. There are various ways to discretise the Radon transform, e.g. by interpolation. In general, suppose there are m detectors in total, the system can be written as

$$f_i = \sum_{j=1}^{n} k_{ij} u_j, \qquad i = 1, ..., m,$$

where $f_i = g(\varphi_i, s_i)$ is the beam intensity detected by detector i, u_j denotes the pixel value of the image u in the j-th pixel ($j = 1, ...n$), and k_{ij} indicates the contribution of pixel j to detector i. The matrix form of the resulting system is

$$\mathbf{Ku} = \mathbf{f},$$

where $\mathbf{K} \in \mathbb{R}^{m \times n}$ is called the *discrete Radon transform*, $\mathbf{u} \in \mathbb{R}^n$ is a size $\sqrt{n} \times \sqrt{n}$ image as a vector ($\sqrt{n} \in \mathbb{N}$), and $\mathbf{f} \in \mathbb{R}^m$ are the *measurements*. Finally, we incorporate the error sources in the discrete model, i.e.

$$\mathbf{Ku} = \mathbf{f} + \boldsymbol{\delta} = \mathbf{f}^{\boldsymbol{\delta}}, \tag{2.3}$$

where $\boldsymbol{\delta}$ follows a Gaussian distribution $\mathcal{N}(0, \sigma^2 \mathbf{I})$. Here σ is the standard deviation of the Gaussian distribution, and we call σ the *noise level*.

2.2 Regularised CT Image Reconstruction

Given the CT scan measurements $\mathbf{f}^{\boldsymbol{\delta}}$, CT image reconstruction attempts to rebuild the underlying binary image $\bar{\mathbf{u}}$ (also called the *ground truth*). However, equation (2.3) can often be an ill-posed problem, since the system is either overdetermined or underdetermined when $m \neq n$. Accurate CT image reconstruction is also hindered by the noise in $\mathbf{f}^{\boldsymbol{\delta}}$. The accuracy of the reconstructed image can be increased by adding a regularisation term to equation (2.3). Moreover, regularisation enables us to incorporate prior knowledge on expected image structures.

Consider the inverse problem of recovering an image $\mathbf{u} \in \{0, 1\}^n$ from CT measurements

$$\mathbf{f}^{\delta} \approx \mathbf{Ku},$$

where $\mathbf{u} \in \mathbb{R}^n$ represents a 2D image in vector form, $\mathbf{K} \in \mathbb{R}^{m \times n}$ is a discrete Radon transform, and $\mathbf{f}^{\delta} \in \mathbb{R}^m$ are the measurements.

To solve the inverse problem, a regularised variational approach is of the form

$$\min_{\mathbf{u}} \; F(\mathbf{u}) = G(\mathbf{u}) + R(\mathbf{u}), \tag{2.4}$$

where

$$G(\mathbf{u}) = \frac{1}{2} \|\mathbf{Ku} - \mathbf{f}^{\boldsymbol{\delta}}\|^2$$

is the data fidelity term, and $R(\mathbf{u})$ is the regularisation term.

A common method to solve the minimisation problem (2.4) is proximal gradient descent (Beck, 2017, p. 129). First, we introduce the concept of subgradient.

Definition 2.2. (subgradient) Consider the real vector space $\mathbb{R}^n$. Let $R : \mathbb{R}^n \to (-\infty, \infty)$ be a proper function and let $\mathbf{u} \in \mathrm{dom}(R)$. A vector $\mathbf{w} \in \mathbb{R}^n$ is called a subgradient of R at $\mathbf{u}$ if

$$R(\mathbf{v}) \geq R(\mathbf{u}) + \langle \mathbf{w}, \mathbf{v} - \mathbf{u} \rangle, \quad \forall \mathbf{v} \in \mathbb{R}^n.$$

The set of all subgradients of R at $\mathbf{u}$ is called the subdifferential of R at $\mathbf{u}$ and is denoted by $\partial R(\mathbf{u})$.

Definition 2.3. (stationary point) Let $G : \mathbb{R}^n \to (-\infty, \infty)$ be a proper function and let $R : \mathbb{R}^n \to (-\infty, \infty)$ be a proper convex function such that $\mathrm{dom}(R) \subseteq \mathrm{int}(\mathrm{dom}(G))$. Consider the problem

$$\min_{\mathbf{u}\in\mathbb{R}^n} F(\mathbf{u}) = G(\mathbf{u}) + R(\mathbf{u}). \tag{2.5}$$

A point $\mathbf{u}^*$ in which G is differentiable is called a stationary point of (2.5) if

$$-\nabla G(\mathbf{u}^*) \in \partial R(\mathbf{u}^*).$$

$\mathbf{u}^*$ is also called the stationary point of the function F.

The proximal gradient method utilises this relaxation of the concept of stationary point.

Definition 2.4. (proximal mapping) Given a function $R : \mathbb{R}^n \to (-\infty, \infty)$, the proximal mapping of R is the operator given by

$$\mathrm{prox}_R(\mathbf{x}) = \arg\min_{\mathbf{u}\in\mathbb{R}^n} \left\{ R(\mathbf{u}) + \frac{1}{2}\|\mathbf{u} - \mathbf{x}\|^2 \right\}, \qquad \forall \mathbf{x} \in \mathbb{R}^n.$$

Definition 2.5. (proximal gradient method) Consider the minimisation model

$$\min_{\mathbf{x}\in\mathbb{R}^n} F(\mathbf{u}) = G(\mathbf{u}) + R(\mathbf{u}). \tag{2.6}$$

Assume

i) $G : \mathbb{R}^n \to (-\infty, \infty]$ is proper and closed, $\mathrm{dom}(G)$ is convex, $\mathrm{dom}(R) \subseteq \mathrm{int}(\mathrm{dom}(G))$ and ∇G is L_G-Lipschitz over $\mathrm{int}(\mathrm{dom}(G))$.

ii) $R : \mathbb{R}^n \to (-\infty, \infty]$ is proper closed and convex.

The proximal gradient method is given by Algorithm 1. Here $\alpha^{(k)}$ denotes the stepsize at iteration k.

Algorithm 1: The proximal gradient method

```
Initialisation: choose $\mathbf{u}^{(0)} \in \mathrm{int}(\mathrm{dom}(G))$.
1 for $k = 0, 1, ...$ do
2     i) pick $\alpha^{(k)} > 0$;
3     ii) set $\mathbf{u}^{(k+1)} = \mathrm{prox}_{\alpha^{(k)}R}\left(\mathbf{u}^{(k)} - \alpha^{(k)}\nabla G(\mathbf{u}^{(k)})\right)$.
4 end
```

To get the idea behind the proximal gradient method, consider a quadratic approximation $\bar{G}$ of G near $\mathbf{u}$,

$$G(u) + \nabla G(u)^\intercal(\mathbf{v} - \mathbf{u}) + \frac{1}{2\alpha}\|\mathbf{v} - \mathbf{u}\|^2,$$

which can be considered as a second-order Taylor expansion of G at $\mathbf{u}$ but replacing $\nabla^2 G$ by $\frac{1}{\alpha}\mathbf{I}$. The motivation to get the next iteration $\mathbf{u}^\dagger$ of $\mathbf{u}$ is to minimise the approximation $\bar{G}$

instead G while keeping R unchanged, i.e.

$$\begin{aligned}
\mathbf{u}^{\dagger} &= \arg\min_{\mathbf{v}} G(\mathbf{u}) + \nabla G(\mathbf{u})^{\intercal}(\mathbf{v}-\mathbf{u}) + \frac{1}{2\alpha}\|\mathbf{v}-\mathbf{u}\|^2 + R(\mathbf{u}) \\
&= \arg\min_{\mathbf{v}} \alpha G(\mathbf{u}) + \alpha\nabla G(\mathbf{u})^{\intercal}(\mathbf{v}-\mathbf{u}) + \frac{1}{2}\|\mathbf{v}-\mathbf{u}\|^2 + \alpha R(\mathbf{u}) \quad \text{(multiply by } \alpha) \\
&= \arg\min_{\mathbf{v}} \frac{\alpha^2}{2}\|\nabla G(\mathbf{u})\|^2 + \alpha\nabla G(\mathbf{u})^{\intercal}(\mathbf{v}-\mathbf{u}) + \frac{1}{2}\|\mathbf{v}-\mathbf{u}\|^2 + \alpha R(\mathbf{u}) \quad (\pm \text{ constant}) \\
&= \arg\min_{\mathbf{v}} \frac{1}{2}\|(\mathbf{v}-\mathbf{u}) + \alpha\nabla G(\mathbf{u})\|^2 + \alpha R(\mathbf{u}) \quad \text{(complete square)} \\
&= \arg\min_{\mathbf{v}} \frac{1}{2}\|\mathbf{v}-(\mathbf{u}-\alpha\nabla G(\mathbf{u}))\|^2 + \alpha R(\mathbf{u}) \\
&= \operatorname{prox}_{\alpha R}(\mathbf{u}-\alpha\nabla G(\mathbf{u})).
\end{aligned}$$

We give the convergence result when the stepsize $\alpha^{(k)}$ is set to be constant.

Definition 2.6. (convergence of the proximal gradient method) Suppose the assumptions in Definition 2.5 hold. Denote $\{\mathbf{u}^{(k)}\}_{k\geq 0}$ the set of iterations obtained by the proximal gradient method for solving problem (2.6) with a constant stepsize $\alpha \in (0, \frac{2}{L_G})$. Then

(a) the sequence $\{(G+R)(\mathbf{u}^{(k)})\}_{k\geq 0}$ is non-increasing. Moreover, $(G+R)(\mathbf{u}^{(k+1)}) < (G+R)(\mathbf{u}^{(k)})$ if and only if $\mathbf{x}^{(k}$ is not a stationary point of (2.6);

(b) $\frac{1}{\alpha}(\mathbf{u} - \operatorname{prox}_{\alpha R}(\mathbf{u}-\alpha\nabla G(\mathbf{u}))) \to \mathbf{0}$ as $k\to\infty$;

(c) all limit points of the sequence $\{\mathbf{u}^{(k)}\}_{k\geq 0}$ are stationary points of problem (2.6).

The gradient of $G(\mathbf{u})$ is

$$\nabla G(\mathbf{u}) = \mathbf{K}^H(\mathbf{K}\mathbf{u}-\mathbf{f}), \tag{2.7}$$

and the update step of the proximal gradient method with constant stepsize is given by

$$\mathbf{u}^{(k+1)} = \operatorname{prox}_{\alpha R}\left(\mathbf{u}^{(k)} - \alpha\mathbf{K}^H(\mathbf{K}\mathbf{u}^{(k)}-\mathbf{f})\right). \tag{2.8}$$

The procedure of solving (2.4) is summarised in Algorithm 2, where we further break down equation (2.8) to a two-stage update. The range of the stepsize α in Algorithm 2 is decided as follows. Since ∇G is Lipschitz continuous, we have

$$\|\nabla G(\mathbf{u}) - \nabla G(\mathbf{v})\| \leq L_G\|\mathbf{u}-\mathbf{v}\|, \qquad \forall \mathbf{u},\mathbf{v}\in\mathbb{R}^n. \tag{2.11}$$

Substituting the gradient of G (2.7) into (2.11), we obtain

$$\|\mathbf{K}^H\mathbf{K}(\mathbf{u}-\mathbf{v})\| \leq \|\mathbf{K}^H\mathbf{K}\|\cdot\|\mathbf{u}-\mathbf{v}\|, \qquad \forall \mathbf{u},\mathbf{v}\in\mathbb{R}^n.$$

Note that (2.10) can be viewed as an image denoising problem with the same regularisation term as in (2.4). Therefore, we devote Section 3 to study the image denoising problem (2.10), to gain better understanding of the proximal operator.

Algorithm 2: The proximal gradient method with constant stepsize for solving equation (2.4)

```
Initialisation: choose $\mathbf{u}^{(0)} \in \text{int}(\text{dom}(G))$. Pick $\alpha \in (0, \frac{2}{\|\mathbf{K}^H\mathbf{K}\|})$ and $K \in \mathbb{Z}_+$.
1 for $k = 0, 1, ..., K$ do
2     i) Set
          $$\mathbf{v}^{(k)} = \mathbf{u}^{(k)} - \alpha\mathbf{K}^H\left(\mathbf{K}\mathbf{u}^{(k)} - \mathbf{f}\right). \quad (2.9)$$
3     ii) Solve
          $$\mathbf{u}^{(k+1)} = \text{prox}_{\alpha R}(\mathbf{v}^{(k)}) = \arg\min_{\mathbf{u}} \frac{1}{2\alpha}\|\mathbf{u} - \mathbf{v}^{(k)}\|^2 + R(\mathbf{u}). \quad (2.10)$$
4 end
5 return $u^{(K+1)}$
```

2.3 Minimisation Methods

Algorithm 2 exhibits the optimisation problem that needs to be solved. We describe the minimisation techniques employed in this thesis before discussing the regulariser choices. The primary method used is Newton-CG method. Cubic regularisation is also briefly discussed. We denote the iterations of the minimisation methods by $\cdot^k$, in order to distinguish them from those of the proximal gradient method $\cdot^{(k)}$.

2.3.1 Newton-CG

This section is based on (Nocedal & Wright, 1999, p. 165). Newton-CG is an algorithm designed for large-scale unconstrained optimisation problems. One reason that it is suitable for large-scale problems is because, by using a modified CG method, it can be implemented without calculating the inverse of Hessian matrix directly.

Suppose we want to solve

$$\min_{\mathbf{u}\in\mathbb{R}^n} F(\mathbf{u}),$$

where $F : \mathbb{R}^n \mapsto \mathbb{R}$ is smooth. Denote ∇F^k the gradient of F at $\mathbf{u}^k$, and $\nabla^2 F^k$ the Hessian matrix at $\mathbf{u}^k$. Suppose $\nabla^2 F^k$ is positive definite, the Newton-CG iteration is updated according to

$$\mathbf{u}^{k+1} = \mathbf{u}^k + \tau^k\mathbf{p}^k,$$

where τ^k is the step length and $\mathbf{p}^k$ is the Newton direction, i.e. $\mathbf{p}^k$ satisfies

$$\nabla^2 F^k\mathbf{p}^k = -\nabla F^k.$$

The Newton-CG finds the search direction $\mathbf{p}^k$ using *CG* method. Define the residual

$$\mathbf{r}^k = \nabla^2 F^k\mathbf{p}^k + \nabla F^k.$$

Let $\mathbf{B}^k$ represent $\nabla^2 F^k$. Denote $\mathbf{d}^j$ the search directions and $\mathbf{z}^j$ the sequence of iterates that it generates. Algorithm 3 shows the Newton-CG method for finding a minimiser of the objective function F.

Algorithm 3: The Newton-CG method for finding a local minimiser of F.

```
Initialisation: choose initial point $\mathbf{u}^0 \in \text{int}(\text{dom}(F))$.
for $k = 0, 1, 2, ...$ do
    Define tolerance $\eta^k = \min(0.5, \sqrt{\|\nabla F^k\|})\|\nabla F^k\|$;
    Set $\mathbf{z}^0 = 0, \mathbf{r}^0 = \nabla F^k, \mathbf{d}^0 = -\mathbf{r}^0$;
    for $j = 0, 1, 2, ...$ do
        if $(\mathbf{d}^j)^\intercal \mathbf{B}^k \mathbf{d}^j \leq 0$ then
            if $j = 0$ then
                return $\mathbf{p}^k = -\nabla F^k$;
            end
            else
                return $\mathbf{p}^k = \mathbf{z}^j$;
            end
        end
        Set $\tau^j = (\mathbf{r}^j)^\intercal \mathbf{r}^j / (\mathbf{d}^j)^\intercal \mathbf{B}^k \mathbf{d}^j$;
        Set $\mathbf{z}^{j+1} = \mathbf{z}^j + \tau^j \mathbf{d}^j$;
        Set $\mathbf{r}^{j+1} = \mathbf{r}^j + \tau^j \mathbf{B}^k \mathbf{d}^j$;
        if $\|\mathbf{r}^{j+1}\| < \eta^k$ then
            return $\mathbf{p}^k = \mathbf{z}^{j+1}$;
        end
        Set $\omega^{j+1} = (\mathbf{r}^{j+1})^\intercal \mathbf{r}^{j+1} / (\mathbf{r}^j)^\intercal \mathbf{r}^j$;
        Set $\mathbf{d}^{j+1} = -\mathbf{r}^{j+1} + \omega^{j+1} \mathbf{d}^j$;
    end
    Set $\mathbf{u}^{k+1} = \mathbf{u}^k + \tau^k \mathbf{p}^k$, where $\tau^k$ satisfies the Wolfe, Goldstein, or Armijo
      backtracking condition (using $\tau^k = 1$ if possible);
end
return $u^{(K+1)}$
```

It is important to note that the CG approach is intended to solve positive definite systems. This is not always the case for the Hessian matrix of the objective function F in (2.4). In Algorithm 3 the CG method is modified such that the inner iteration is terminated when a direction of negative curvature is formed. This ensures the $\mathbf{p}^k$ found by CG method is a descent direction. With the usage of the tolerance η^k, the CG iterations can end at an imprecise solution.

Definition 2.7. (convergence of the Newton-CG method) Suppose that F is twice differentiable and $\nabla^2 F(\mathbf{u})$ is continuous in a neighbourhood of a minimiser $\mathbf{u}^*$. Suppose $\nabla^2 F(\mathbf{u}^*)$ is positive definite. Consider the iteration $\mathbf{u}^{k+1} = \mathbf{u}^k + \mathbf{p}^k$ with $\mathbf{p}^k$ satisfying

$$\|\mathbf{r}^k\| \leq \eta^k \|\nabla F^k\|, \qquad 0 < \eta^k < 1, \ \forall k,$$

where the sequence $\{\eta^k\}$ is called the forcing sequence. Assume that $\eta^k < \eta$ for some constant $\eta \in [0, 1)$. Then, if the starting point $\mathbf{u}^0$ is sufficiently near $\mathbf{u}^*$, the sequence $\{\mathbf{u}^k\}$ converges to $\mathbf{u}^*$ and satisfies

$$\|\nabla^2 F(\mathbf{u}^*)(\mathbf{u}^{k+1} - \mathbf{u}^*)\| \leq \hat{\eta} \|\nabla^2 F(\mathbf{u}^*)(\mathbf{u}^k - \mathbf{u}^*)\|, \tag{2.12}$$

for some constant $\hat{\eta}$ with $\eta < \hat{\eta} < 1$.

Remark 2.1. In Algorithm 3, the choice of the forcing sequence is $\min(0.5, \sqrt{\|\nabla F^k\|})$. This ensures a superlinear convergence rate.

Remark 2.2. The drawback of the Newton-CG is that when $\nabla^2 F^k$ is nearly non-invertible, the performance of the Newton-CG can be poor, with many iterations and little reduction in the objective function value.

Remark 2.3. We use the `Newton-CG` option of the `optimizer` function implemented in Scipy package (Virtanen et al., 2020). The maximum number of CG iteration is $20 \times n$. The algorithm terminates if after max iteration the value $(\mathbf{d}^j)^\intercal \mathbf{B}^k \mathbf{d}^j > 3$ for all iterations.

2.3.2 Cubic Regularisation

Suppose we need to solve

$$\min_{\mathbf{u} \in \mathbb{R}^n} F(\mathbf{u}),$$

where F has a Lipschitz continuous Hessian, i.e.

$$\|\nabla^2 F(\mathbf{u}) - \nabla^2 F(\mathbf{v})\| \leq L\|\mathbf{v} - \mathbf{u}\|, \qquad \forall \mathbf{u}, \mathbf{v} \in \mathbb{R}^n. \tag{2.13}$$

Then the next iteration $\mathbf{u}^{k+1}$ can be selected as a solution of the following auxiliary problem

$$\mathbf{u}^{k+1} \in \arg\min_{\mathbf{v}} \xi_{2,\mathbf{u}^k}(\mathbf{v}), \tag{2.14}$$

where argmin means to identify a global minimiser, and the auxiliary function $\xi_{2,\mathbf{u}}$ is of the form

$$\xi_{2,\mathbf{u}}(\mathbf{v}) = F(\mathbf{u}) + \nabla F(\mathbf{u})^\intercal(\mathbf{v} - \mathbf{u}) + \frac{1}{2}\nabla^2 F(\mathbf{u})(\mathbf{v} - \mathbf{u})^2 + \frac{L}{6}\|\mathbf{v} - \mathbf{u}\|^3.$$

The idea behind equation (2.14) is that the auxiliary function $\xi_{2,\mathbf{u}}(\mathbf{v})$ is an upper second-order approximation of the objective function F, i.e.

$$F(\mathbf{v}) \leq \xi_{2,\mathbf{u}}(\mathbf{v}), \qquad \forall \mathbf{v} \in \mathbb{R}^n.$$

This approach is known as the cubic regularisation of Newton's method(Nesterov & Polyak, 2006).

Let M be a positive parameter. Define

$$\bar{F}_M(\mathbf{u}) = \min_{\mathbf{v}} \left[F(\mathbf{u}) + \nabla F(\mathbf{u})^\intercal(\mathbf{v} - \mathbf{u}) + \frac{1}{2}\nabla^2 F(\mathbf{u})(\mathbf{v} - \mathbf{u})^2 + \frac{M}{6}\|\mathbf{v} - \mathbf{u}\|^3 \right].$$

Let $T_M(\mathbf{u})$ be such that

$$T_M(\mathbf{u}) \in \arg\min_{\mathbf{v}} \left[F(\mathbf{u}) + \nabla F(\mathbf{u})^\intercal(\mathbf{v} - \mathbf{u}) + \frac{1}{2}\nabla^2 F(\mathbf{u})(\mathbf{v} - \mathbf{u})^2 + \frac{M}{6}\|\mathbf{v} - \mathbf{u}\|^3 \right].$$

Algorithm 4 describes the cubic regularisation of Newton's method. Due to the complexity, we refer to the original work (Nesterov & Polyak, 2006) for the convergence result.

Algorithm 4: The cubic regularisation of Newton's method.

```
Initialisation: choose $\mathbf{u}^0 \in \mathrm{int}(\mathrm{dom}(G))$. Pick $K \in \mathbb{Z}_+$.
1 for $k = 0, 1, ..., K$ do
2     i) Find $M_k \in [L_0, 2L]$, where $L_0 \in [0, L]$ is a parameter, such that
          $F(T_{M_k}(\mathbf{u}^k)) \leq \bar{F}_{M_k}(\mathbf{u}^k)$.
3     ii) Set
          $\mathbf{u}^{k+1} = T_{M_k}(\mathbf{u}^k)$.
4 end
5 return $u^{K+1}$
```

2.4 The Graph Ginzburg-Landau (GL) functional

We investigate the use of the graph Ginzburg-Landau (GL) functional

$$R'_{gl}(\mathbf{u}) = \frac{1}{2}\sum_{i,j} w_{ij}(u_i - u_j)^2 + \frac{1}{\varepsilon}\sum_i W(u_i), \tag{2.15}$$

where $W(x) = x^2(x-1)^2, x \in \mathbb{R}$ is a double-well potential, as a regulariser in this thesis. In order to accomplish this, we interpret images as undirected, no self-loop graphs $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where the vertex set $\mathcal{V}$ consists of all image pixels, and $\mathcal{E} \in \mathcal{V} \times \mathcal{V}$ is the edge set. The pixel values of the images are viewed as graph signals residing on the graph.

Remark 2.4. The expression of the graph GL functional shows that it is not convex, therefore does not satisfy the assumption on R in Definition 2.5. This means that we may not have guarantee on the convergence of proximal gradient method and Newton-CG. There are many variations on the proximal gradient method for non-convex R such as (Li & Lin, 2015), but in this thesis we stick to the standard version described by Algorithm 2.

Remark 2.5. The objective function (2.4) with R'_{gl}, i.e. $F = G + R'_{gl}$, satisfies the assumption (2.13) in cubic regularisation method when we restrict F on a convex domain, e.g. $[-3, 3]^n$. Therefore, the solution found by cubic regularisation is the global minimum.

2.4.1 Derivation of the Graph GL Functional

The graph Ginzburg-Landau functional is extended from its continuous version

$$\mathcal{F}_\varepsilon(u) = \frac{1}{2}\int_\Omega |\nabla u(x)|^2 dx + \frac{1}{\varepsilon}\int_\Omega W(u(x))dx,$$

which was commonly used in physics where it models phase separation and in the image processing (van Gennip, 2019). To see how we derive the graph GL functional, we first introduce some notations related to graphs and the non-local derivative.

Definition 2.8. (graph Laplacian) Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, the (combinatorial) graph Laplacian matrix of the graph is defined as

$$\mathbf{L} = \mathbf{D} - \mathbf{W},$$

where $\mathbf{W}$ is the weight matrix of $\mathcal{G}$,

$$\mathbf{W}_{ij} = \begin{cases} w_{ij} > 0, & \text{if } (i,j) \in \mathcal{E}, \\ 0, & \text{if } (i,j) \notin \mathcal{E}, \end{cases}$$

and $\mathbf{D}$ is the diagonal degree matrix,

$$\mathbf{D}_{ij} = \begin{cases} \sum_j \mathbf{W}_{ij}, & \text{if } i = j, \\ 0, & \text{if } i \neq j. \end{cases}$$

As $\mathcal{G}$ is undirected, $\mathbf{L}$ is symmetric and positive semi-definite. This means that all eigenvalues of $\mathbf{L}$ are non-negative. Also, 0 is always an eigenvalue of $\mathbf{L}$, with corresponding eigenvector $[1, ..., 1]^\intercal$. The two common normalisations of $\mathbf{L}$ are the normalised graph Laplacian

$$\mathbf{L}_n = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{W}\mathbf{D}^{-\frac{1}{2}},$$

and the random-walk graph Laplacian

$$\mathbf{L}_r = \mathbf{I} - \mathbf{D}^{-1}\mathbf{W}.$$

Note that the first term of R'_{gl} in (2.15) can be written as

$$\frac{1}{2}\sum_{i,j} w_{ij}(u_i - u_j)^2 = \mathbf{u}^\intercal\mathbf{L}\mathbf{u}. \tag{2.16}$$

This means that using the graph Laplacian term $\mathbf{u}^\intercal\mathbf{L}\mathbf{u}$ in the regularisation imposes smoothness to the pixel values—as we can see from the expression, since w_{ij} are positive, $\mathbf{u}^\intercal\mathbf{L}\mathbf{u}$ would only be small when u_i and u_j with $(i,j) \in \mathcal{E}$ have similar values.

Definition 2.9. (non-local derivative) Consider $\Omega \in \mathbb{R}^n$. Let $u : \Omega \to \mathbb{R}$ be a function. Define the non-local derivative

$$\frac{\partial u}{\partial y}(x) = \frac{u(y) - u(x)}{d(x,y)}, \qquad x, y \in \Omega, \tag{2.17}$$

where d satisfies $0 < d(x,y) \leq \infty,\ \forall x, y \in \Omega$.

We can now show how the graph GL functional is derived. The derivation is based on (Merkurjev, Kostic, & Bertozzi, 2013). It is straightforward to see that the term $\int_\Omega W(u(x))dx$ can be discretised as $\sum_i W(u_i)$. Define the weight function as

$$w(x,y) = \frac{1}{d(x,y)^2},$$

then the non-local derivative (2.17) can be written as

$$\frac{\partial u}{\partial y}(x) = \sqrt{w(x,y)}\,(u(y) - u(x)), \tag{2.18}$$

and the non-local gradient $\nabla_w u(x) : \Omega \to \Omega \times \Omega$ is of the form

$$(\nabla_w u)(x,y) = (u(y) - u(x))\sqrt{w(x,y)}, \qquad x, y \in \Omega. \tag{2.19}$$

Taking the square on both side of (2.19) and taking the integral, we obtain

$$\int_{\Omega} |\nabla u|^2 dx = \int_{\Omega\times\Omega} (u(y) - u(x))^2 \, w(x,y) dxdy. \tag{2.20}$$

Hence, we have derived the discretised GL functional, i.e.

$$R'_{gl}(\mathbf{u}) = \frac{1}{2}\sum_{i,j} w_{ij}(u_i - u_j)^2 + \frac{1}{\varepsilon}\sum_i W(u_i).$$

Moreover, R'_{gl} can be rewritten in a more compact form by using equation (2.16). For convenience, we rescale equation (2.15) and redefine ε and will use

$$R_{gl}(\mathbf{u}) = \frac{1}{2}\mathbf{u}^{\intercal}\mathbf{L}\mathbf{u} + \frac{1}{\varepsilon}\sum_i W(u_i),$$

as the definition of the graph GL functional in the rest of the thesis.

2.4.2 Choices of Graph Laplacian Matrix

Using the graph GL functional as regularisation has the benefit of allowing for more modelling flexibility as the graph Laplacian matrix can be customised by the user. Naturally, the choice of the graph Laplacian matrix $\mathbf{L}$, i.e. how we decide the connectivity of the graph and the edge weights, will influence the performance of the graph GL functional R_{gl} as a regulariser. Here we introduce two graph Laplacian matrices that is used in our experiments.

4-Regular Graph Laplacian

In graph theory, a 4-regular graph means that all nodes in the graph have a degree of 4. Here we slightly abuse the notation, and view a graph that forms a square mesh as a 4-regular graph. This means that the nodes on the border of the mesh only have degree 2 or 3.

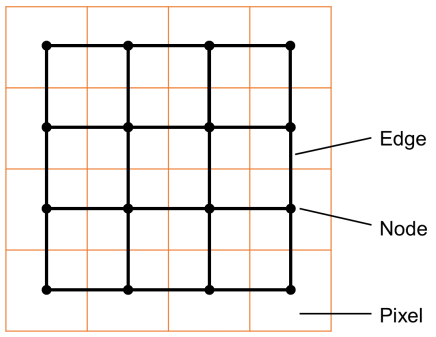


Figure 2.4: Illustration of a 4-regular graph resides on a 4×4 pixel image.

The corresponding 4-regular graph Laplacian matrix is obtained by view the images as 4-regular graphs. Figure 2.4 demonstrates how the graph is defined in this case using a 4×4 image. We set the weights to 1 for all edges. This graph Laplacian is denoted by $\mathbf{L}_4$.

Remark 2.6. Research has shown that a normalised graph Laplacian matrix exhibits a better experiment result than the unnormalised ones (Bertozzi & Flenner, 2012). However, in this thesis we use the unnormalised version for the 4-regular graph Laplacian $\mathbf{L}_4$, as in our experiment the unnormalised $\mathbf{L}_4$ is numerically more stable when solving the model using Newton-CG.

Learned Graph Laplacian

Having seen how we can define the graph Laplacian from a fixed graph structure, we introduce a different way, where we use an algorithm to learn the graph structure and edge weights from a set of correlated images. As discussed in Introduction, such algorithms can be found in many graph signal processing literature. We employ the GL-3SR method (Humbert et al., 2021), which is a model that learns the graph Laplacian with a sparse graph structure and smooth graph signals. The method was initially introduced in graphs signal literature, and had not been applied to data consists of images. We achieve this by treating image pixel values as graph signals.

The model learns the graph Laplacian $\mathbf{L}$ by learning the eigendecomposition of $\mathbf{L}$, i.e.

$$\mathbf{L} = \mathbf{X}\boldsymbol{\Lambda}\mathbf{X}^\intercal,$$

where each column of $\mathbf{X} \in \mathbb{R}^{n\times n}$ is an eigenvector of $\mathbf{L}$, and $\boldsymbol{\Lambda} \in \mathbb{R}^{n\times n}$ is a diagonal matrix where the diagonal elements are the eigenvalues of $\mathbf{L}$. Denote $\mathbf{H}$ the spectral representation of the images $\mathbf{Y}$, i.e.

$$\mathbf{H} = \mathbf{X}^\intercal\mathbf{Y},$$

where $\mathbf{Y} \in \mathbb{R}^{n\times N}$ and N is the number of images we use to learn $\mathbf{L}$. Given the images data $\mathbf{Y}$, the model fit a graph Laplacian matrix by solving

$$\begin{aligned}
\min_{\mathbf{H},\mathbf{X},\boldsymbol{\Lambda}} \; & \|\mathbf{Y} - \mathbf{X}\mathbf{H}\|_F^2 + \gamma\|\boldsymbol{\Lambda}^{1/2}\mathbf{H}\|_F^2 + \beta\|\mathbf{H}\|_S, && (2.21)\\
\text{s.t.} \quad & \text{(a)} \quad \mathbf{X}^\intercal\mathbf{X} = \mathbf{I}_n, \mathbf{x}_1 = \frac{1}{\sqrt{n}}\mathbf{1}_n, \\
& \text{(b)} \quad (\mathbf{X}\boldsymbol{\Lambda}\mathbf{X}^\intercal)_{k,\ell} \le 0, \quad k \neq \ell, \\
& \text{(c)} \quad \boldsymbol{\Lambda} = \operatorname{diag}(0, \lambda_2, ..., \lambda_n) \succeq 0, \\
& \text{(d)} \quad \operatorname{tr}(\boldsymbol{\Lambda}) = n \in \mathbb{R}_*^+,
\end{aligned}$$

where $\|\cdot\|_F$ is the Frobenius norm, and $\|\cdot\|_S$ should be a norm that induces sparsity. The paper proposed to use either $\ell_{2,1}$ or $\ell_{2,0}$ for S.

There are three terms in the objective function (2.21). The first is a data-fidelity term, where the spectral representation of the images $\mathbf{Y}$ is learned. The second term is a frequently used smoothness regularisation in graph learning (Humbert et al., 2021; Kalofolias, 2016). The third term is a sparsity regularisation that results in a solution $\mathbf{H}$ that is row-sparse. The second and third term indirectly forces smoothness and sparsity on the learned $\mathbf{L}$, respectively.

The restrictions (a)-(d) are the conditions that a graph Laplacian matrix's eigenvectors and eigenvalues should satisfy. Constraints (a) requires that the eigenvectors are of unit length and the eigenvector corresponding to eigenvalue 0 is $\frac{1}{\sqrt{n}}[1, ..., 1]^\intercal$. Constraints (b) demands that the non-diagonal elements in the learned $\mathbf{L}$ are negative. Constraints (c) requires that the learned $\mathbf{L}$ is positive semi-definite, and 0 is an eigenvalue of $\mathbf{L}$. Constraints (d) is proposed in (Dong et al., 2016) to avoid the trivial solution where the predicted eigenvalues are all zeros. By putting constraints on $\mathbf{X}$ and $\mathbf{\Lambda}$, the model ensures that the learned matrix $\mathbf{L}$ is indeed a graph Laplacian matrix.

There are two hyperparameters, γ and β in the model. However, in the original paper it is stated that the choice of γ does not influence on the accuracy. Therefore, we choose a fixed value $\gamma = 0.001$ and leave it out of the hyperparameter tuning process.

We denote the learned graph Laplacian matrix $\mathbf{L}_p$. Note that $\mathbf{L}_p$ obtained using the GL-3SR is normalised.

2.5 Other CT Image Reconstruction Approaches

We present a brief summary of the two CT image reconstruction methods used in our thesis, TV regularisation and SIRT. We will compare our GL regularisation model to these two methods on binary CT image reconstruction in Section 4. Details of the two methods can be found in the references for interested readers.

Total Variation (TV) Regularisation

The reconstruction model with anisotropic TV regularisation (Condat, 2017) is of the form

$$\min_{\mathbf{u}} \frac{1}{2}\|\mathbf{Ku} - \mathbf{f}^{\boldsymbol{\delta}}\|^2 + \frac{1}{\mu} R_{tv}(\mathbf{u}), \tag{2.22}$$

where

$$R_{tv}(\mathbf{u}) = \frac{1}{2}\sum_{i,j\in\mathcal{V}} w_{ij}|u_i - u_j|.$$

The model can be solved by the Split Bregman method (Goldstein & Osher, 2009).

SIRT

SIRT is an algebraic iterative method (Andersen & Kak, 1984), where the iteration is of the form

$$\mathbf{u}^{(k+1)} = \mathbf{u}^{(k)} + \mathbf{M}_1\mathbf{K}^H\mathbf{M}_2\left(\mathbf{f}^{\boldsymbol{\delta}} - \mathbf{Ku}^{(k)}\right),$$

where $\mathbf{M}_1 \in \mathbb{R}^{n\times n}$ and $\mathbf{M}_2 \in \mathbb{R}^{m\times m}$ are the diagonal matrices whose diagonal elements are the inverse of sums of the columns and rows of the discrete Radon transform $\mathbf{K}$, respectively.

2.6 Experiments Overview

Now that we have established our model for CT image reconstruction, i.e.

$$\min_{\mathbf{u}} \frac{1}{2}\|\mathbf{Ku} - \mathbf{f}\|^2 + \frac{1}{2}\mathbf{u}^\intercal\mathbf{Lu} + \frac{1}{\varepsilon}\sum_i W(u_i), \tag{2.23}$$

this section seeks to provide a summary of the experiments performed in the thesis.

The experiments are divided into two categories, with each category containing two experiments. As we have seen in Algorithm 2, an image denoising problem needs to be solved in each iteration. Therefore, the first part of the experiments is related to image denoising using graph GL regularisation. We consider the model

$$\min_{\mathbf{u}} \frac{1}{2\alpha}\|\mathbf{u}-\mathbf{v}\|^2 + \frac{1}{2}\mathbf{u}^\intercal \mathbf{L}\mathbf{u} + \frac{1}{\varepsilon}\sum_i W(u_i), \tag{2.24}$$

where $\mathbf{v}$ is the noisy image. The parameter α is the stepsize from Algorithm 2, but here we interpret it as a penalty parameter. We begin by justifying the use of Newton-CG to solve (2.24), where we can also learn about the graph GL functional's performance on image denoising. This experiment is run with simulated 64×64 pixel blob data. Then, we experiment with the influence of graph Laplacian matrix in (2.24), comparing the model performance in different noise levels with two graph Laplacian $\mathbf{L}_4$ and $\mathbf{L}_p$. This experiment is carried out with two datasets on handwritten digits, namely MNIST and USPS.

For the second part of the experiments, we return to the binary CT image reconstruction model (2.23). We investigate the performance of our model with $\mathbf{L}_4$ and $\mathbf{L}_p$ in various noise level and projection downsampling settings, using MNIST dataset. We also compare the reconstruction results from our model to those of TV regularisation and SIRT. Finally, we scan a number 0 made of playdough. This is a 3D object so we only take the central slice from the scan. The central slice can be seen as a binary image (playdough and air). We then apply our model with $\mathbf{L}_4$ to reconstruct the binary image, using the measurements of the central slice as the input data. We also compare the results from various projection downsampling settings.

Data Preparation

We require binary images to serve as the ground truth, hence we need to convert the images from MNIST and USPS datasets to binary images. Let $\mathbf{u}$ be a $\sqrt{n}\times\sqrt{n}$ pixel colourful image where its pixels $u_i \in \{0, ...P\}$. The binary image $\mathbf{u}^b$ is obtained by setting a threshold t and let

$$u_i^b = \mathbb{1}_{\{u_i > t\}}, \quad i = 1, ..., n, \quad t = \frac{P}{2}. \tag{2.25}$$

For image denoising problem (2.24), given the ground truth image $\bar{\mathbf{u}}$, the noisy image $\mathbf{v}$ is obtained by

$$\mathbf{v} = \bar{\mathbf{u}} + \boldsymbol{\delta}, \tag{2.26}$$

where $\boldsymbol{\delta} \sim \mathcal{N}(0, \sigma^2\mathbf{I})$.

For binary CT image reconstruction, given the ground truth image $\bar{\mathbf{u}}$, the noisy measurements $f^{\boldsymbol{\delta}}$ are obtained by

$$\mathbf{f}^{\boldsymbol{\delta}} = \mathbf{f} + \boldsymbol{\delta}, \tag{2.27}$$

where $\mathbf{f} = \mathbf{K}\bar{\mathbf{u}}$ are the simulated measurements.

Model Assessments

We use two error metrics to evaluate the performance of the model: Dice score and root-mean-square error (RMSE). Denote $\hat{\mathbf{u}}$ the optimal solution returned by either (2.24) and (2.23). We first convert $\hat{\mathbf{u}}$ into a binary image $\hat{\mathbf{u}}^b$ according to (2.25), where we set $t = 0.5$. The Dice score is then given by

$$\text{Dice}(\hat{\mathbf{u}}^b, \bar{\mathbf{u}}) = \frac{2|\hat{\mathbf{u}}^b \cap \bar{\mathbf{u}}|}{|\hat{\mathbf{u}}^b| + |\bar{\mathbf{u}}|}. \tag{2.28}$$

The RMSE is given by

$$\mathrm{RMSE}(\hat{\mathbf{u}}, \bar{\mathbf{u}}) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{u_i} - \bar{u}_i)^2}. \tag{2.29}$$

3 Results and Discussion on Image Denoising

In this section, we focus on the image denoising model (2.24), which also appears in solving the CT reconstruction problem in Algorithm 2. In section 3.1, we start by examining the performance of Newton-CG in solving equation (2.24). In particular, we look into the best option for the initial guess for Newton-CG. We compare Newton-CG with cubic regularisation to determine whether the Newton-CG solution is the global minimum of equation (2.24). Moreover, we provide the denoising results with different noise levels. In section 3.2, we investigate the influence of replacing the graph Laplacian matrix in (2.24). We test two distinct graph Laplacian matrices on two datasets, both featuring handwritten digits.

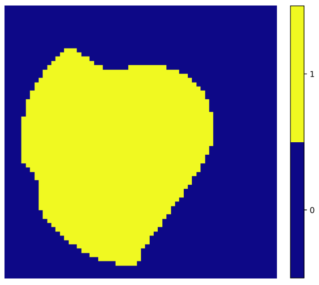
3.1 Blob Images

In this section, we simulate 64×64 pixel blob images to investigate the use of Newton-CG to solve the denoising model (2.24). Two initialisation vectors for Newton-CG are assessed. As mentioned in Section 2, the objective function in (2.24) is not convex. Therefore, we may not have guarantee on the convergence of Newton-CG method. To establish whether the Newton-CG method's result is the global minimum, we also apply cubic regularisation to solve (2.24) and compare the performance of the two methods.

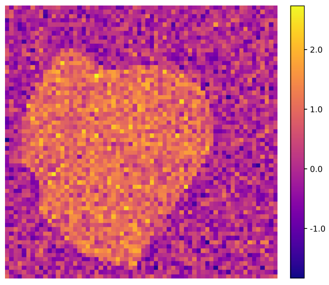
3.1.1 Initialisation Vectors

An initial guess is required as an input for the Newton-CG method. In this section we test what would be a good choice for our model. The two initialisation vectors considered in this project are the noisy image $\mathbf{v}$ itself and the zero vector $\mathbf{0}$. We use the 4-regular graph Laplacian matrix in equation (2.10), and set the maximum number of iterations for the Newton-CG method to 200. The experiment carried out is as follows:

1. Generate 40 blob images of size 64×64 pixels as ground truth $\{\bar{\mathbf{u}}_i\}_{i=1}^{40}$ and add Gaussian noise according to (2.26), with $\sigma = 0.5$, to them to obtain the noisy images $\{\mathbf{v}_i\}_{i=1}^{40}$. Randomly split the data $\{(\mathbf{v}_i, \bar{\mathbf{u}}_i)\}_{i=1}^{40}$ into two sets of equal size $\{(\mathbf{v}_i, \bar{\mathbf{u}}_i)_{train}\}_{i=1}^{20}$ and $\{(\mathbf{v}_i, \bar{\mathbf{u}}_i)_{test}\}_{i=1}^{20}$. Figure 3.1 illustrates an example of a 64×64 pixel blob image $\bar{\mathbf{u}}$ and its corresponding $\mathbf{v}$ when $\sigma = 0.5$.



(a) Ground truth $\bar{\mathbf{u}}$



(b) Corresponding noisy image $\mathbf{v}$ with $\sigma = 0.5$

Figure 3.1: A 64×64 pixel blob image $\bar{\mathbf{u}}$ and its noisy version $\mathbf{v}$ with noise level $\sigma = 0.5$.

2. Perform hyperparameter tuning for (α, ε) using the training set. This process is done for two initial guesses of Newton-CG, the noisy image itself $\mathbf{v}_{i,train}$ and the zero vector $\mathbf{0}$, respectively. Denote $\hat{\mathbf{u}}_i^{\mathbf{v}}$ as the solution obtained by using $\mathbf{v}_{i,train}$ as the initial guess, and

$\hat{\mathbf{u}}_i^{\mathbf{0}}$ as the solution by using $\mathbf{0}$ as the initial guess. The solutions are binarised according to (2.25), where we set $t = 0.5$. The best combination is selected based on the average Dice score between the denoised image $\hat{\mathbf{u}}_i^{\mathbf{v}}$ (and $\hat{\mathbf{u}}_i^{\mathbf{0}}$) and the ground truth $\bar{\mathbf{u}}_i$, i.e.

$$\frac{1}{20}\sum_{i=1}^{20} \text{Dice}((\hat{\mathbf{u}}_i^{\mathbf{v}})^b, \bar{\mathbf{u}}_i), \qquad \text{and} \qquad \frac{1}{20}\sum_{i=1}^{20} \text{Dice}((\hat{\mathbf{u}}_i^{\mathbf{0}})^b, \bar{\mathbf{u}}_i).$$

If there are multiple combinations of (α, ε) returning the same Dice score, we further compare which pair of combinations has the lowest average RMSE. The parameters are selected from $\alpha \in \{0.2, 0.4, ..., 2\}$ and $\varepsilon \in \{0.02, 0.04, ..., 0.2\}$. Since we only consider equation (2.24) as a model on its own, there is no constraint on the value of α.

3. Test the model with the selected $(\hat{\alpha}, \hat{\varepsilon})$ using the test set $\{(\mathbf{v}_i, \bar{\mathbf{u}}_i)_{test}\}_{i=1}^{20}$ for both initial guesses. The Dice score and RMSE between the denoised image $\hat{\mathbf{u}}_i^{\mathbf{v}}$ (or $\hat{\mathbf{u}}_i^{\mathbf{0}}$) and the ground truth $\bar{\mathbf{u}}_i$ are measured. Additionally, we record for each $\mathbf{v}_i$ whether the Newton-CG is converged when solving with two initial guesses.

Both initial guesses selected $(\hat{\alpha}, \hat{\varepsilon}) = (2, 0.2)$. Table 3.1 demonstrates the test result for noise level $\sigma = 0.5$. Overall, using the noisy image $\mathbf{v}$ produces noticeably better results out of the two initial guesses. The average Dice score and average RMSE of $\mathbf{0}$ are worse than those of $\mathbf{v}$. Additionally, only 10 out of 20 solutions are converged when using $\mathbf{0}$, indicating unstable performance. It is worth noting that, after discarding the unconverged results, the performance of $\mathbf{0}$ is comparable to using the noisy image $\mathbf{v}$ as the initial guess. However, $\mathbf{0}$ requires more iterations to run, whether or not the unconverged cases are included.

Table 3.1: Performance comparison for two initial guesses, tested on $N_2 = 20$ blob images with noise level $\sigma = 0.5$. The parameters $(\hat{\alpha}, \hat{\varepsilon}) = (2, 0.2)$ are used for model (2.24).

	Initial guess	
Measurement	**Zero vector 0**	**Noisy image v**
Number of successful convergence	10	19
Average number of iterations	71	40
Average number of iterations (excl. unconverged)	94 ± 26	41 ± 11
Dice socre	0.9655 ± 0.0583	0.9919 ± 0.0019
Dice score (excl. unconv.)	0.9916 ± 0.0033	0.9920 ± 0.0019
RMSE	0.1333 ± 0.0862	0.0816 ± 0.0083
RMSE (excl. unconverged)	0.0873 ± 0.0119	0.0812 ± 0.0084

Figure 3.2 illustrates the error metrics, RMSE and Dice score, for the test data using the two initial guesses. It is straightforward to see that the unconverged case when using $\mathbf{v}$ as initial guess still yields a high Dice score and a low RMSE, which is not necessarily the case for $\mathbf{0}$.

Figure 3.3 shows the denoising of the noisy image $\mathbf{v}$ shown in Figure 3.1 with model (2.24) with $(\hat{\alpha}, \hat{\varepsilon}) = (2, 0.2)$, using $\mathbf{v}$ itself as the initial guess. Figure 3.3a and 3.3b show the denoised image $\hat{\mathbf{u}}$ and its binary version $\hat{\mathbf{u}}^b$. Figure 3.3c demonstrates the difference between $\hat{\mathbf{u}}^b$ and the ground truth $\bar{\mathbf{u}}$. It is clear in this figure that the error is from the edge of the blob. This

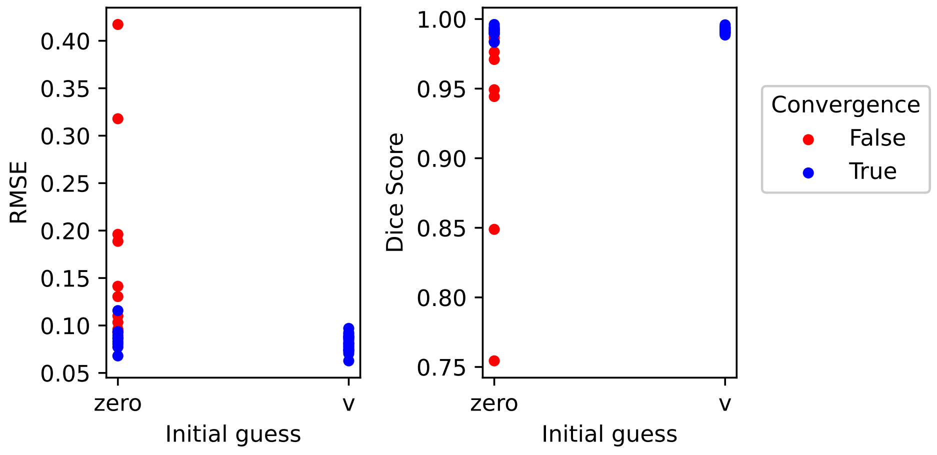
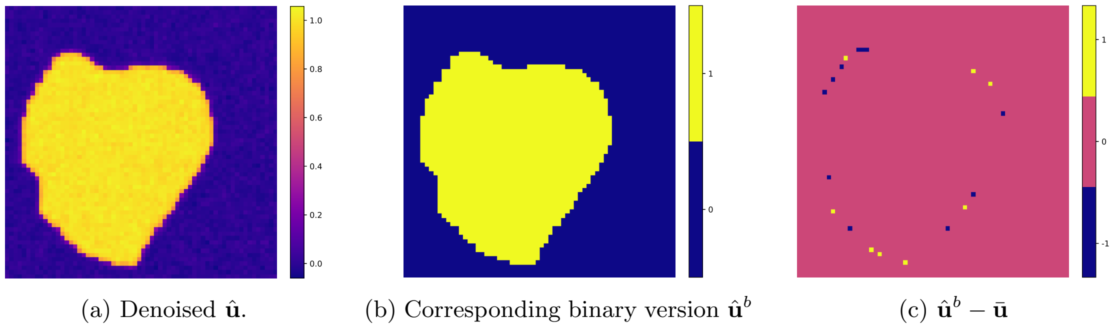


Figure 3.2: Error metrics with two initial guesses $\mathbf{0}$ and $\mathbf{v}$ on the $N_2 = 20$ test data. The parameters $(\hat{\alpha}, \hat{\varepsilon}) = (2, 0.2)$ are used for model (2.24).

can be explained by the graph Laplacian term attempting to avoid sharp transitions, i.e. from 0 to 1, in the neighbour pixel values. As a result, rather than having values of 0 or 1, the pixels on the edge of the blob would have values closer to 0.5 in the solution $\hat{\mathbf{u}}$. Therefore, it is difficult for the model to correctly predict the values of pixels on the edge of the blob. Figure 3.4 demonstrates the corresponding log objective value and log norm of the gradient of the objective function in each iteration while using Newton-CG to denoise the noisy image $\mathbf{v}$ in Figure 3.1. Newton-CG converged successfully in this example, and the objective values decrease monotonically. The log norm of the gradient decreases to -10, meaning the solution found is indeed a stationary point of the objective function.



(a) Denoised $\hat{\mathbf{u}}$. (b) Corresponding binary version $\hat{\mathbf{u}}^b$ (c) $\hat{\mathbf{u}}^b - \bar{\mathbf{u}}$

Figure 3.3: The denoising result of $\mathbf{v}$ in Figure 3.1, obtained using model (2.24) with $(\hat{\alpha}, \hat{\varepsilon}) = (2, 0.2)$. The initial guess $\mathbf{v}$ is used for Newton-CG.

3.1.2 Methods Comparison

In order to check if Newton-CG produces the global minimum, we also employ cubic regularisation[1] to solve model (2.24) and compare the solutions between the two minimisation methods.

[1]The function is implemented by https://github.com/cjones6/cubic_reg

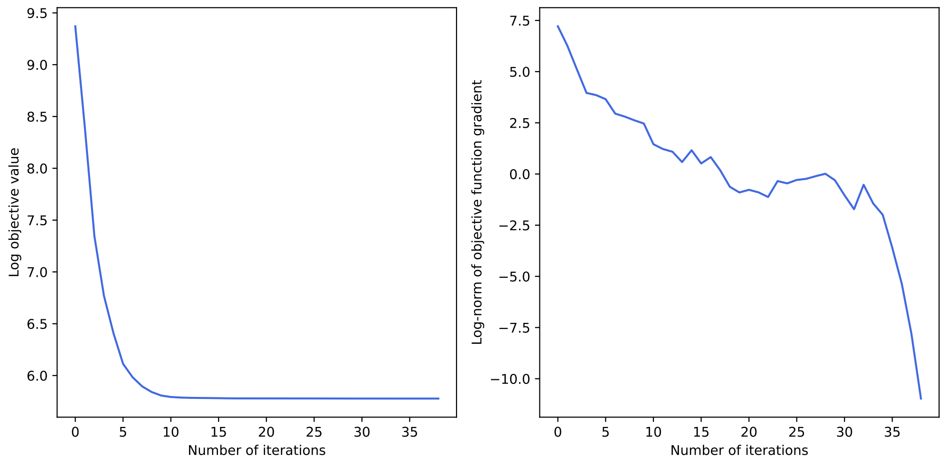


Figure 3.4: Left: log objective value against iteration. Right: log norm of the objective function gradient against iteration. Values are obtained from the denoising result of $\mathbf{v}$ in Figure 3.1, solved by model (2.24) with $(\hat{\alpha}, \hat{\varepsilon}) = (2, 0.2)$. The initial guess $\mathbf{v}$ is used for Newton-CG.

The data used are a number of 64×64 pixel blob images and their $\sigma = 0.5$ noisy version. The model's hyperparameters are set to be $(\hat{\alpha}, \hat{\varepsilon}) = (2, 0.2)$, which were determined in the previous section. We use $\mathbf{v}$ as initial guess for both Newton-CG and cubic regularisation and analyse the results in the converged and unconverged cases.

When Newton-CG successfully converges, we discover that Newton-CG and cubic regularisation return almost identical solutions. In this case we conclude that Newton-CG finds the global minimum of the objective function (2.24).

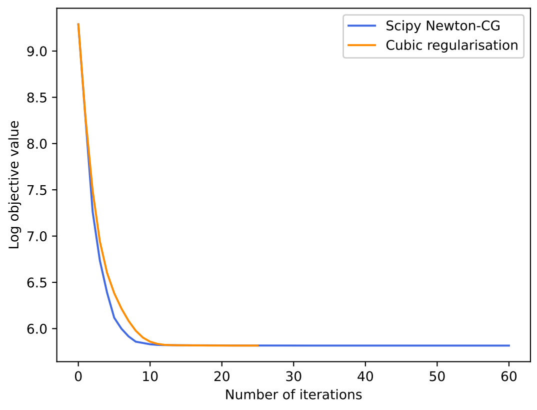
On the other hand, we collect 10 image pairs $(\bar{\mathbf{u}}, \mathbf{v})$ where Newton-CG failed to converge. Table 3.2 lists several statistics we summarised from the solutions given by Newton-CG $\hat{\mathbf{u}}^N$ and cubic regularisation $\hat{\mathbf{u}}^C$ in this case. All 10 solutions returned by cubic regularisation $\hat{\mathbf{u}}^C$ are successfully converged. We can see that in the unconverged situation, the solutions returned by Newton-CG are not stationary points, as the norm of the gradient at $\hat{\mathbf{u}}^N$ are not closed to 0. However, the Dice score of $\hat{\mathbf{u}}^N$ are considerably close to those of the cubic regularisation solutions $\hat{\mathbf{u}}^C$.

Table 3.2: Performance comparison of Newton-CG and cubic regularisation when Newton-CG did not converge. Statistics are obtained based on 10 unconverged solutions.

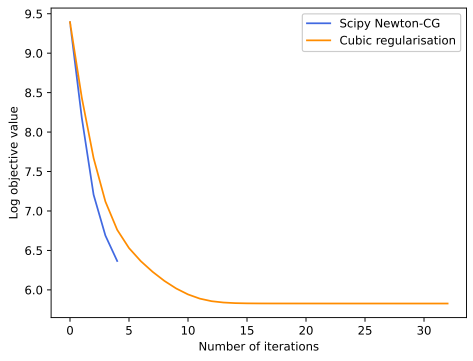
Statistic	**Value**
Average norm of the gradient at $\hat{\mathbf{u}}^N$	15.4038
Average difference in objective value	71.3516
Average difference in RMSE	0.0368
Average difference in Dice score	0.0146

Figure 3.5 exhibit three examples of the objective values in log scale against iterations when solving (2.24) using both methods. Figure 3.5a shows a case where Newton-CG converges. Figure 3.5b and 3.5c demonstrate two typical situations where Newton-CG did not converge. In Figure 3.5b Newton-CG stopped after 5 iterations due to non-positive definite Hessian matrix. This leads to large value of the norm of the gradient at $\hat{\mathbf{u}}^N$, and large difference in

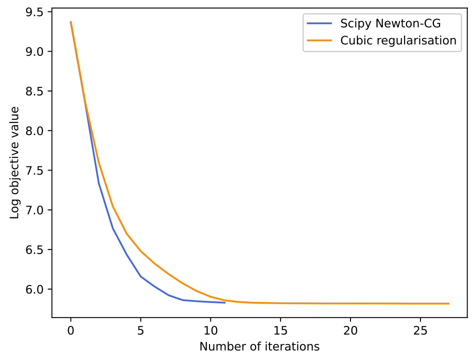
objective values between $\hat{\mathbf{u}}^N$ and $\hat{\mathbf{u}}^C$. In Figure 3.5c Newton-CG runs more iterations before the Hessian matrix becoming non-positive definite. Despite the solution $\hat{\mathbf{u}}^N$ is not a stationary point, the norm of the gradient at $\hat{\mathbf{u}}^N$ is close to 0. The difference in objective values between $\hat{\mathbf{u}}^N$ and $\hat{\mathbf{u}}^C$ is small in this situation.



(a) Newton-CG iterations converge. The solutions found by Newton-CG and cubic regularisation are nearly identical.



(b) Newton-CG iterations did not converge. The algorithm stopped after 5 iterations.



(c) Newton-CG iterations did not converge. The algorithm stopped after 11 iterations.

Figure 3.5: Plots of the log objective values of the two methods against iterations. In (a) the solution found by Newton-CG nearly coincides with that of the cubic regularisation. (b) and (c) are two cases where Newton-CG did not converge, and the solutions were not stationary points.

In the unconverged case, we conclude that the solution Newton-CG returns is not the global minimum. However, it can still be useful as we have seen that the Dice score of these solutions are close to that of the global minimum solutions $\hat{\mathbf{u}}^C$.

The time needed for both methods to find a solution is listed in Table 3.3. The long hours needed for solving with 128×128 pixel images means that cubic regularisation does not scale well with high dimensional data.

To summarise, Newton-CG is fast and therefore offers more freedom for us in selecting the image size. Moreover, the solution of Newton-CG provides high Dice score even in the un-

Table 3.3: Time needed for both methods to return a solution for various image sizes.

	Image size		
Method	32×32	64×64	128×128
Cubic regularisation	~ 20s	~ 7mins	~ 5 hrs
Newton-CG (converged)	< 0.1s	< 0.1s	< 1s
Newton-CG (unconverged)	~ 2s	~ 15s	~ 6min

converged case. Therefore, for subsequent experiments we stick to Newton-CG with the noisy image itself as the initial guess.

3.1.3 Denoising with Various Noise Levels

In this section, we evaluate the performance of the proposed model (2.24) when denoising blob images with various noise levels. We apply the model to blob images with three noise level $\sigma \in \{0.15, 0.5, 0.7\}$. For each noise level, the experiment is carried out based on the approach laid out in Section 3.1.1, with the only difference being that we only use the noisy image $\mathbf{v}$ as the initial guess.

Table 3.4: Selected hyperparameters $(\hat{\alpha}, \hat{\varepsilon})$ for $\sigma \in \{0.15, 0.5, 0.7\}$.

Noise level σ	$(\hat{\alpha}, \hat{\varepsilon})$
0.15	(1,0.1)
0.5	(2,0.2)
0.7	(1.6, 0.2)

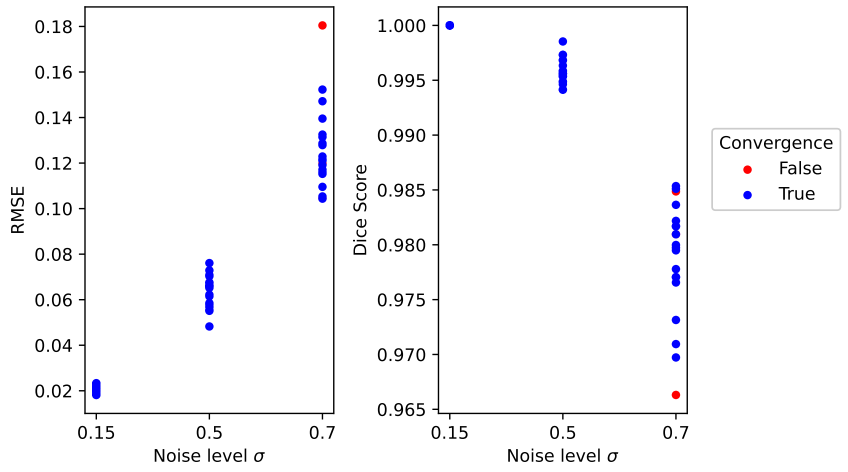


Figure 3.6: Left: scatter plot of RMSE and Dice score of the denoised images with three noise levels $\sigma \in \{0.15, 0.5, 0.7\}$, obtained using Newton-CG with $\mathbf{v}$ as the initial guess. There are 20 test images used for each noise level.

Table 3.4 shows the hyperparameter tuning results for three noise levels. Figure 3.6 displays the image denoising results for three noise levels using the selected hyperparameter values in Table 3.4. We see that the Dice score decreases as σ increases, and the RMSE grows as σ increases. This is to be expected, given that the more corrupted an image is, the more challenging it is to recover all of the pixel values. We also notice that for large noise level, there is higher variance in Dice scores and RMSEs. Moreover, the number of unconverged solutions grows as σ increases. When $\sigma = 0.15$, all 20 solutions are converged, whereas there are two unconverged cases when $\sigma = 0.7$. This indicates that it is difficult for Newton-CG to converge when σ is large.

3.2 Digit Images

Having analysed the performance of the minimisation method Newton-CG and the model behaviour with various noisy levels, we now look into how the graph Laplacian matrix would affect model (2.24). We study on the benefit of using two different graph Laplacian matrices. In particular, we try the graph Laplacian matrix where the connectivity of the nodes and the edge weights in the graph are not predefined by us, but learned from datasets using GL-3SR.

The denoising model (2.24) is applied to two digit image datasets: MNIST and USPS. The two digit image datasets were chosen because in each dataset the images of the same digit have the same label but these images are not identical. This makes them a good choice for learning the underlying graph structure. Their small image sizes are also advantageous. This is because the GL-3SR algorithm takes more time to train for a large number of nodes. Moreover, high dimensionality means that the Newton-CG method is less likely to converge.

3.2.1 MNIST Experiment Description

There are various versions for MNIST with different image sizes. In this section we use the version with 8×8 pixel images, where each pixel has an integer value in $\{0, ..., 16\}$.

The following steps are taken in this experiment.

0. Select image samples of a digit:
   MNIST dataset contains 178 images of the digit 0 in total. We randomly sample $N = 170$ images of the digit 0 from the dataset.

1. Obtain the ground truth images:
   We convert the images of the digit 0 into binary images $\bar{\mathbf{u}}_i, i \in \{0, .., N\}$ according to equation (2.25), where the threshold t is chosen to be 8. $X^b = \{\bar{\mathbf{u}}_i\}_{i=1}^{N=170}$ will be used as the ground truth. Figure 3.7 displays four images of digit 0 in the MNIST dataset and the corresponding binary counterpart.

2. Obtain the noisy images:
   We obtain the noisy image set $X = \{\mathbf{v}_i\}_{i=1}^{N=170}$ by adding Gaussian noise to the ground truth according to equation (2.26). The noisy images are generated with $\sigma \in \{0.15, 0.25, 0.3, 0.35\}$.

3. Tune the hyperparameters:
   The noisy images and their ground truth (X, X^b) are split into a training set of size $N_1 = 150$ and a test set of size $N_2 = 20$. The hyperparameters that returns the best mean dice score are selected. If there are multiple hyperparameter combinations that yield the same Dice score, we then seek which combination has the lowest RMSE. The hyperparameter tuning process for the denoising model (2.24) with different Laplacian matrices are conducted as follows.

   a) Learned graph Laplacian $\mathbf{L}_p$:
   There are three hyperparameters to be tuned: β from GL-3SR algorithm (2.21), α and ε in

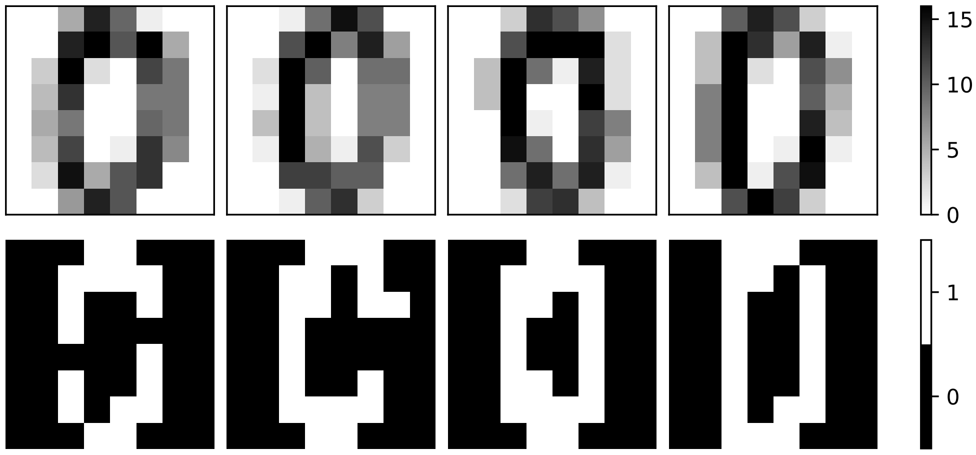


Figure 3.7: Top: four examples of digit 0 images in the MNIST dataset. Bottom: their corresponding binary images.

the objective function of (2.24). 5-fold cross validation is used to tune the hyperparameters due to the relatively small data size. We split the training data into 5 folds of the same size. Each time we use three folds of X^b to learn the graph Laplacian matrix $\mathbf{L}_p$, and apply the denoising model (2.24) to the remaining two folds of X with various hyperparameter values. After the best combination of the hyperparameters is selected by cross validation, we learn the graph Laplacian $\mathbf{L}_p$ on the whole $N_1 = 150$ of X^b and test the performance of the selected hyperparameters on the test data. Note that $\mathbf{L}_p$ is learned using only the ground truth data, not the noisy images. The process is illustrated in Figure 3.8. The hyperparameters are selected in the ranges $\alpha \in \{0.02, 0.04, 0.06, 0.08\} \cup \{0.1, 0.2, ..., 1.9, 2\}$, $\varepsilon \in \{0.01, ..., 0.09\} \cup \{0.1, 0.2, ..., 1\}$, and $\beta \in \{1, ..., 15\}$.

b) 4-regular graph Laplacian $\mathbf{L}_4$:
There are two hyperparameters to be tuned: α and ε in the objective function (2.24). We tune the hyperparameters on the $N_1 = 150$ training data and then test them on the test data. The ranges of α and ε are the same as in a).

4. Test the models:
We test the models with selected hyperparameters $(\hat{\beta}, \hat{\alpha}, \hat{\varepsilon})_{\mathbf{L}_p}$ and $(\hat{\alpha}, \hat{\varepsilon})_{\mathbf{L}_4}$ for noise levels $\sigma \in \{0.15, 0.25, 0.3, 0.35\}$ on the $N_2 = 20$ test data. The mean Dice score and mean RMSE are calculated for the chosen hyperparameter combination.

3.2.2 MNIST Experiment Results and Discussion

Figure 3.9 illustrates the unit length eigenvector corresponding to the second smallest eigenvalue of the learned graph Laplacian matrix using all $N_1 = 150$ training data. This figure shows the type of images that minimises the graph Laplacian term $\frac{1}{2}\hat{\mathbf{u}}\mathbf{L}_p\hat{\mathbf{u}}$ in the objective function (2.24).

The parameters selected and the testing result are displayed in Table 3.5. In general, the

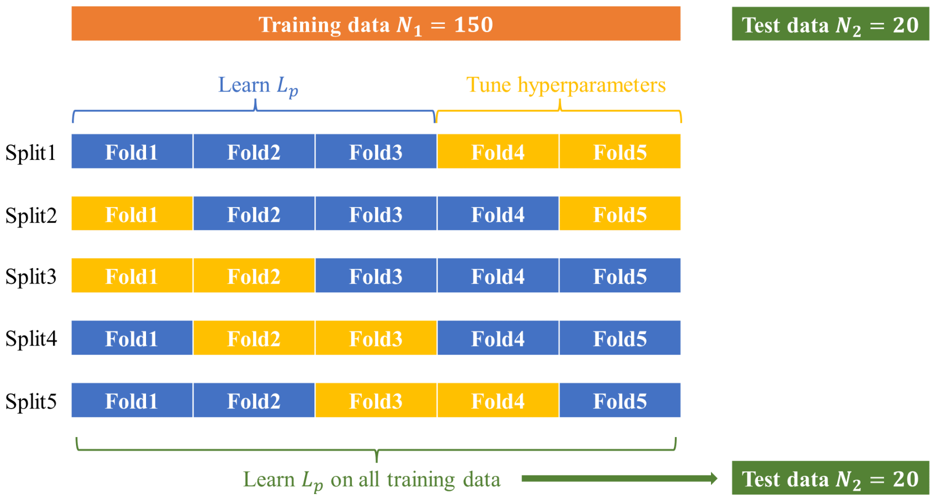


Figure 3.8: 5-fold cross validation for MNIST digit 0.

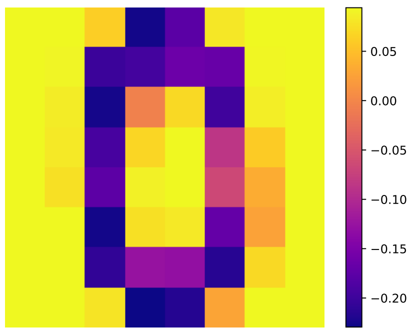


Figure 3.9: The unit length eigenvector corresponding to the second small eigenvalue of the learned graph Laplacian matrix using $N_1 = 150$ images of MNIST digit 0.

model with both graph Laplacian matrices can reduce the noise in the noisy images and recover most of the pixels in binary images. As shown in Table 3.5, model (2.24) with learned graph Laplacian matrix $\mathbf{L}_p$ yields better denoising results than that of the model with 4-regular graph Laplacian matrix $\mathbf{L}_4$. For $\sigma = 0.15$, both models can fully recover the ground truth binary image. This is not a surprising result, as the average Dice score between the noisy images $X_{\sigma=0.15}$ and their ground truth $X^b_{\sigma=0.15}$ is 1. This means that the noisy image can be directly binarised into the ground truth images. In this case, the $\mathbf{L}_4$ model gives a lower RMSE value, although both RMSE values are small. For the $\sigma = 0.25$, model $\mathbf{L}_4$ does not improve much on the Dice score, comparing the Dice score from model $\mathbf{L}_4$ 0.9789 with the reference Dice score between $X_{\sigma=0.25}$ and $X^b_{\sigma=0.25}$ 0.9773, whereas model $\mathbf{L}_p$ still recovers most of the pixels correctly, with a Dice score of 0.9930. For $\sigma \in \{0.25, 0.3, 0.35\}$, model $\mathbf{L}_p$ yields better performance since it returns a higher Dice score and a lower RMSE. For all σ, model $\mathbf{L}_p$ is able to reduce RMSE values by more than a half. To sum up, model $\mathbf{L}_5$ is more

Table 3.5: Best combination of parameters and performance comparison of 2 different graph Laplacian matrices. Tested on $N_2 = 20$ iamges of the digit 0. For reference, the mean RMSE and Dice score between X and X^b of corresponding σ are shown in the parentheses.

	$\sigma = 0.15$ (0.1496/1)		$\sigma = 0.25$ (0.2520/0.9773)		$\sigma = 0.3$ (0.3031/0.9469)		$\sigma = 0.35$ (0.3496/0.9336)	
	$\mathbf{L_4}$	$\mathbf{L_p}$	$\mathbf{L_4}$	$\mathbf{L_p}$	$\mathbf{L_4}$	$\mathbf{L_p}$	$\mathbf{L_4}$	$\mathbf{L_p}$
$\hat{\beta}$	N.A.	1	N.A.	10	N.A.	3	N.A.	6
$\hat{\alpha}$	0.04	0.2	0.08	0.6	0.1	1.6	0.2	1.3
$\hat{\varepsilon}$	0.03	0.2	1	0.4	0.2	0.3	0.2	0.7
Dice score	1	1	0.9781	**0.9875**	0.9656	**0.9852**	0.9469	**0.9727**
(std)	(0)	(0)	(0.0239)	(0.0161)	(0.0175)	(0.0135)	(0.0234)	(0.0241)
RMSE	**0.0487**	0.0515	0.1982	**0.1098**	0.2016	**0.1063**	0.2174	**0.1578**
(std)	(0.0053)	(0.0060)	(0.0206)	(0.0413)	(0.0244)	(0.0500)	(0.0276)	(0.0426)
Successful conv. ratio	1	1	1	1	1	1	1	1

able to reduce the noise in the images, and can recover the binary image with higher accuracy.

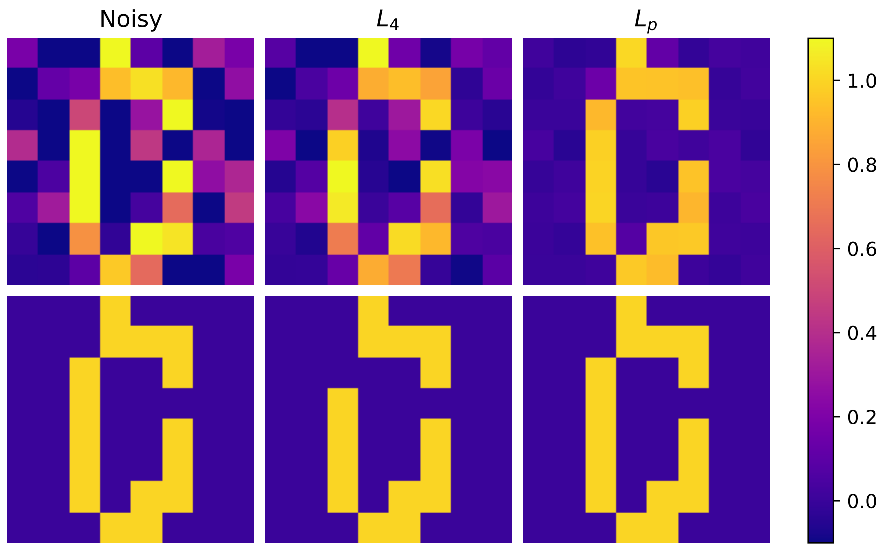


Figure 3.10: Left: the noisy image $\mathbf{v}$ with $\sigma = 0.3$ and its binary image, i.e. the ground truth $\bar{\mathbf{u}}$. Middle: the denoised image $\hat{\mathbf{u}}_{\mathbf{L}_4}$ obtained using the proposed model with $\mathbf{L}_4$ and its binary version. Right: the denoised image $\hat{\mathbf{u}}_{\mathbf{L}_p}$ obtained using the proposed model with $\mathbf{L}_p$ and its binary version.

Figure 3.10 shows an example of a noisy image $\mathbf{v}$ with $\sigma = 0.3$ and the denoised image $\hat{\mathbf{u}}_{\mathbf{L}_4}$ and $\hat{\mathbf{u}}_{\mathbf{L}_p}$ using the proposed model (2.24) with $\mathbf{L}_4$ and $\mathbf{L}_p$, respectively. We can see that model $\mathbf{L}_p$ successfully recovered all the pixel values, and it significantly reduced the noise, by comparing $\hat{\mathbf{v}}_{\mathbf{L}_p}$ with the ground truth $\bar{\mathbf{u}}$. Model with $\mathbf{L}_4$ failed to bring down the noise in $\mathbf{v}$, and predicted one pixel value wrong in the binarised denoised image.

3.2.3 USPS Experiment Results and Discussion

However, model $\mathbf{L}_p$ does not always has such good performance. To test how well it performs with higher resolution images and dataset with more variance, we apply model (2.24) on the USPS dataset.

Created by the U.S. Postal Service, the USPS dataset consists of 16×16 pixel images of handwritten digits on the envelops. Each pixel is assigned a value in $[-1, 1]$. There are around 1000 images in the dataset for each digit. The experimentation is similar to what was carried out with the MNIST dataset, except that the threshold for obtaining binary images is now set to $t = 0$. The other difference is that we do not apply cross validation on the USPS dataset for model $\mathbf{L}_p$, as there are a rather large number of images the digit 0. We randomly select $N = 800$ image samples for digit 0, which is split it into 3 sets of size $N_1 = 500, N_2 = 200$, and $N_3 = 100$.

Figure 3.11 displays four images of digit 0 in the USPS dataset and the corresponding binary counterpart. It can be seen that there are more variations in terms of the position where the digit resides in the USPS dataset than in the MNIST dataset.

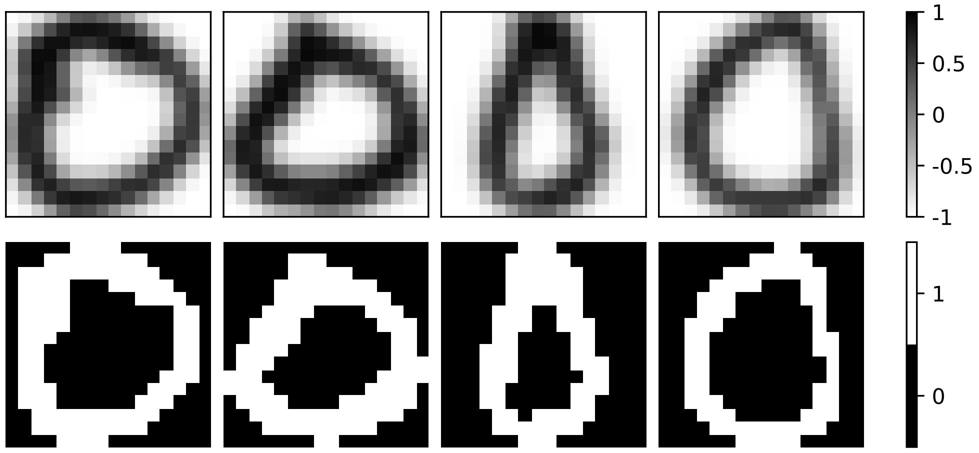


Figure 3.11: Top: four examples of 16×16 pixel images of the digit 0 in the USPS dataset. Bottom: their corresponding binary images.

For the model with learned graph Laplacian $\mathbf{L}_p$, we first fit the graph Laplacian matrix $\mathbf{L}_p$ using GL-3SR with the $N_1 = 500$ learning set, then tune the hyperparameters $(\beta, \alpha, \varepsilon)$ using the $N_2 = 200$ tuning set. After obtaining the best combination of the hyperparameters, we test it on the test set of size $N_3 = 100$, with the $\mathbf{L}_p$ learned on $N_1 = 500$ learning set. For model with $\mathbf{L}_4$, we perform hyperparameter tuning for (α, ε) on the $N_2 = 200$ tuning set, and test it on the $N_3 = 100$ test set.

The unit length eigenvector corresponding to the second smallest eigenvalue of the learned graph Laplacian matrix $\mathbf{L}_p$ is demonstrated in Figure 3.12. This shows the shape of 0 that $\mathbf{L}_p$ favours when minimising the graph Laplacian term in model (2.24).

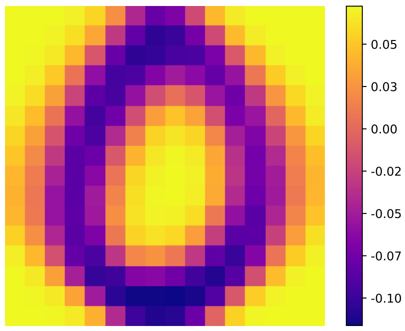


Figure 3.12: The unit length eigenvector corresponding to the second small eigenvalue of the learned Laplacian matrix using $N_1 = 500$ USPS images of the digit 0.

Table 3.6: Performance comparison of 2 different approaches. Tested on $N_3 = 100$ images of digit 0.

	$\sigma = 0.15$ (0.1502/0.9997)		$\sigma = 0.25$ (0.2515/0.9760)		$\sigma = 0.3$ (0.2987/0.9504)		$\sigma = 0.35$ (0.3497/0.9229)	
	$\mathbf{L_4}$	$\mathbf{L_p}$	$\mathbf{L_4}$	$\mathbf{L_p}$	$\mathbf{L_4}$	$\mathbf{L_p}$	$\mathbf{L_4}$	$\mathbf{L_p}$
$\hat{\beta}$	N.A.	8	N.A.	12	N.A.	10	N.A.	8
$\hat{\alpha}$	0.5	0.2	0.5	0.6	0.4	1	0.4	1.5
$\hat{\varepsilon}$	0.06	0.2	0.1	0.4	0.1	0.4	0.2	0.4
RMSE	**0.0320**	0.0637	**0.0914**	0.1286	**0.1215**	0.1650	**0.1560**	0.1980
(st.d)	(0.0062)	(0.0053)	(0.0259)	(0.0184)	(0.0257)	(0.0268)	(0.0207)	(0.0279)
Dice score	**0.9999**	0.9998	**0.9916**	0.9854	**0.9825**	0.9685	**0.9741**	0.9524
(st.d)	(0.0005)	(0.0007)	(0.0064)	(0.0084)	(0.0086)	(0.0123)	(0.0118)	(0.0148)
Successful conv. ratio	0.99	**1**	0.95	**1**	**0.97**	0.95	**1**	0.96

Table 3.6 displays hyperparameter values selected and the denoising results with noisy levels $\sigma \in \{0.15, 0.25, 0.3, 0.35\}$. We notice that model (2.24) does not always converge, due to the larger image size. For each σ, model $\mathbf{L}_4$ always has better performance than model $\mathbf{L}_p$, with lower average RMSEs and higher average Dice scores. For $\sigma = 0.15$ and $\sigma = 0.25$, model $\mathbf{L}_p$ is more stable, with convergent rate 1 in both case. However, this is not the case for $\sigma = 0.3$ and $\sigma = 0.35$.

Figure 3.13 demonstrates an example of denoising a noisy image image $\mathbf{v}$ with $\sigma = 0.3$ in the USPS dataset using the proposed $\mathbf{L}_4$ and $\mathbf{L}_p$ models. In this case, model $\mathbf{L}_4$ has a better performance. The noise in $\mathbf{v}$ has significantly reduced in the solution $\hat{\mathbf{u}}_{\mathbf{L}_4}$, and the binarised denoised image $\hat{\mathbf{u}}^b_{\mathbf{L}_4}$ has only two pixels different than the ground truth $\bar{\mathbf{u}}$. Model $\mathbf{L}_p$ noticeably decreased some noise in $\mathbf{v}$, but failed to bring smoothness to the denoised image $\hat{\mathbf{u}}_{\mathbf{L}_p}$. We can see in the binarised image $\hat{\mathbf{u}}^b_{\mathbf{L}_p}$ that there are some “holes” in the recovered 0.

From the experiments on the MNIST and USPS datasets, we find that it is not always beneficial to learn the graph Laplacian matrix from the dataset. When the images in the dataset exhibit high variance, the 4-regular graph Laplacian matrix can be a better choice for model (2.24).

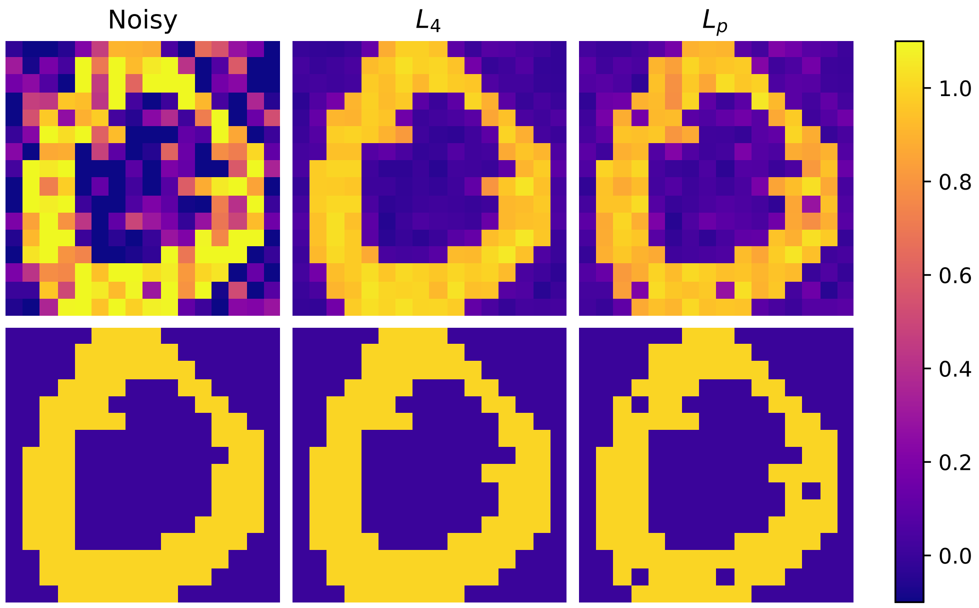


Figure 3.13: Left: the noisy image $\mathbf{v}$ with $\sigma = 0.3$ and its corresponding ground truth $\bar{\mathbf{u}}$. Middle: the denoised image obtained using the proposed model with $\mathbf{L}_4$ and its binary version. Right: the denoised image obtained using the proposed model with $\mathbf{L}_p$ and its binary version. Hyperparameters are the same as shown in Table 3.6.

4 Results and Discussion on CT Image Reconstruction

Having discussed the image denoising problem (2.24), we return to the full model (2.23) solved by Algorithm 2. In Section 4.1 we work with simulated measurements data. We compare the reconstruction accuracy of GL regularisation,TV regularisation and SIRT in various noise and projection angle settings. In Section 4.2, we put our suggested graph GL regularisation to the test using real measurements from the CT scan of a number made of playdough.

4.1 CT Image Reconstruction on MNIST Digits

4.1.1 Experiment Description

In this section, we opt for images of the digit 0 from the 28×28 pixel MNIST dataset, where each pixel is assigned an integer value from the set $\{0, ..., 255\}$. The ground truth images $\bar{\mathbf{u}}$ are obtained using (2.25), with $t = 127$. Figure 4.1 depicts four examples of the images of 0 in the MNIST dataset and their corresponding binary images.

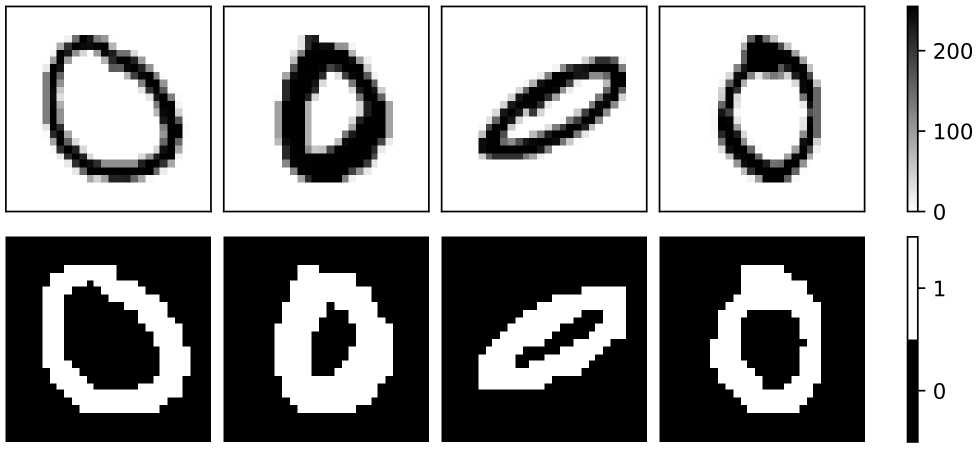


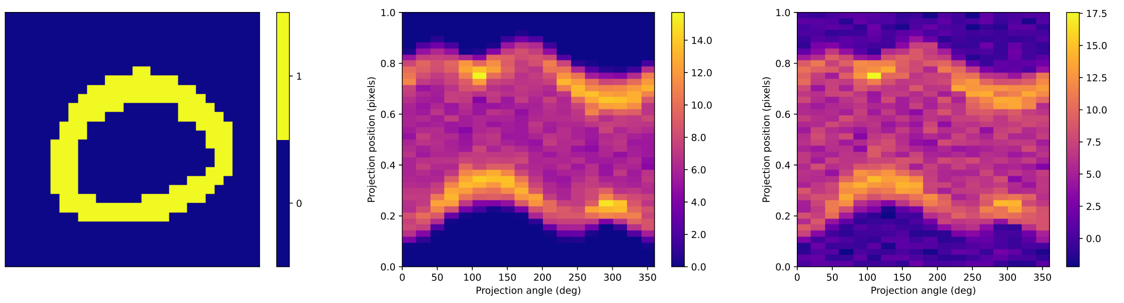
Figure 4.1: Top: four 28×28 pixel images of the digit 0 in the MNIST dataset. Bottom: their corresponding binary images.

The discrete Radon transform $\mathbf{K}$ and the measurements $\mathbf{f}$ of $\bar{\mathbf{u}}$ are generated using ASTRA Toolbox (Van Aarle et al., 2016). Table 4.1 lists all projection geometries used to obtain the discrete Radon transform matrix $\mathbf{K}$ and the measurements (sinograms) $\mathbf{f}$ of the binary images. As shown in Table 4.1, there are seven parameters needed to define a projection geometry. In our experiment, we use fan-beam geometry with flat detector, as shown in Figure 1.1 (right). The number of detectors is set to be 1.5 times the number of pixels on each row/column of the image. We take the centre of the image as the origin. The distance between the origin and X-ray source is 100 and the distance between the origin and the detector is 50. The projection angle is the angle at which we make a projection. The projection angles are varied in increments of θ from $0°$ to $360°$. For example, when $\theta = 5°$, projections are made at 72 equally spaced angles of $5°, 10°, ...360°$. The projection angle decides the row dimension of $\mathbf{K}$.

Table 4.1: The projection geometries used to obtain sinograms.

Parameter	**Setting**
beam	flat fan beam
discretisation	strip
number of detectors	$\lfloor 1.5 \times \sqrt{n} \rfloor = 42$
detector width	1
projection angle	every θ, 2θ,... $\in (0°, 360°]$
origin to source dist.	100
origin to detector dist.	50

After the measurements $\mathbf{f}$ are acquired, we add noise to it according to (2.27). Figure 4.2 illustrates an example of a binary image $\bar{\mathbf{u}}$ of the digit 0, its corresponding sinogram $\mathbf{f}$ using the settings listed in Table 4.1 with $\theta = 20°$, and the noisy version of $\mathbf{f}$, $\mathbf{f}^{\boldsymbol{\delta}}$ with $\sigma = 1$.



(a) Ground truth $\bar{\mathbf{u}}$ (b) Corresponding sinogram $\mathbf{f}$ with $\theta = 20°$ (c) Noisy sinogram $\mathbf{f}^{\delta}$ with $\sigma = 1$

Figure 4.2: An example of a binary image of 0 in the MNIST, its corresponding sinogram, obtained using $\theta = 20°$, and the noisy sinogram with noise level $\sigma = 1$.

Given the discrete Radon transform $\mathbf{K}$, we take $\mathbf{f}^{\boldsymbol{\delta}}$ as an input and reconstruct the underlying binary image of the digit 0. We try out the model in several combinations of projection angles and noise levels, more specifically, the following 9 scenarios: $(\theta, \sigma) \in \{5°, 10°, 20°\} \times \{0.3, 0.07, 1\}$. We then compare the performance of the four reconstruction models: SIRT, TV regularisation, and our proposed model, i.e. the graph GL regularisation with $\mathbf{L}_4$ and $\mathbf{L}_p$. The settings of each model are described below.

I SIRT :
   We do not perform hyperparameter tuning on this method. The number of iterations is set to be 30. The computation is done using the built-in SIRT algorithm in ASTRA Toolbox.

II Total variation:
   We use the built-in `splitbregman` algorithm in PyLops (Ravasi & Vasconcelos, 2020) to solve the model (2.22). We use the default setting in `splitbregman` for other parameter values, but treat the regularisation parameter μ as a hyprparameter. We tune μ for $\mu \in \{0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 1.5\}$. Then the best parameter is chosen according to the average Dice score (2.28) between the reconstructed images and their corresponding ground truth.

III Graph GL regularisation:
   The model is solved by Algorithm 2. We choose $K = 2000$ for the number of iterations. For

the model with $\mathbf{L}_4$, there are two hyperparameters α and ε. As shown in Section 2.2, the range of α depends on the discrete Radon transform $\mathbf{K}$, whereas $\mathbf{K}$ depends on θ. Table 4.2 displays the dimensions of $\mathbf{K}$ used in the experiment, the corresponding range of α, and a selection of values of α chosen to tune the hyperparameters. Therefore, we search for the best combination of $(\alpha, \varepsilon) \in T_\theta \times \{0.1, 0.05, 0.01, 0.005\}$, for $\theta \in \{5°, 10°, 20°\}$, where T_θ is defined in Table 4.2. For the model with $\mathbf{L}_p$, there is one extra parameter β. Therefore, we search for the best combination in $(\beta, \alpha, \varepsilon) \in \{5, 10, 15\} \times T_\theta \times \{0.1, 0.05, 0.01, 0.005\}$. Again, the combination of the hyperparameters with the highest average Dice score is selected.

Table 4.2: The dimension of $\mathbf{K}$ in three θ values, the corresponding α range and the corresponding values of α selected to perform hyperparameter tuning.

$\boldsymbol{\theta}$	dim($\mathbf{K}$)	**α range**	**selected values of α for hyperparameter tuning**
5°	3024×784	$(0, 4.27 \times 10^{-4})$	$T_{5°} = \{4 \times 10^{-4}, 1 \times 10^{-4}, 5 \times 10^{-5}\}$
10°	1512×784	$(0, 8.44 \times 10^{-4})$	$T_{10°} = \{8 \times 10^{-4}, 4 \times 10^{-4}, 1 \times 10^{-4}\}$
20°	756×784	$(0, 1.62 \times 10^{-3})$	$T_{20°} = \{1 \times 10^{-3}, 5 \times 10^{-4}, 1 \times 10^{-4}\}$

We sampled $N = 900$ images of the digit 0 from MNIST dataset. The 900 images are split into three sets: $N_1 = 500$ images are used to learn the graph Laplacian matrix $\mathbf{L}_p$, $N_2 = 300$ images are used to tune the hyperparameters for TV and graph GL regularisation models, and the rest $N_3 = 100$ images are used during testing.

4.1.2 Results and Discussion

Table 4.3 lists the hyperparameter tuning result. These are the parameter values used during testing. The best performance for $\mathbf{L}_p$ model is achieved with $\beta = 10$, though the outcomes for all three β values are quite similar with the same (α, ε).

Table 4.3: Parameters selected by the hyperparameter tuning. These are the parameter values used for MNIST digit 0 CT image reconstruction.

Setting	**TV:** $\hat{\mu}$	$\mathbf{L}_4$: $(\hat{\alpha}, \hat{\varepsilon})$	$\mathbf{L}_p$: $(\hat{\beta}, \hat{\alpha}, \hat{\varepsilon})$
$\boldsymbol{\theta = 5°, \sigma = 0.3}$	1	(0.0004, 0.005)	(10, 0.0001, 0.005)
$\boldsymbol{\theta = 5°, \sigma = 0.7}$	1	(0.0004, 0.005)	(10, 0.0001, 0.005)
$\boldsymbol{\theta = 5°, \sigma = 1}$	0.5	(0.0004, 0.01)	(10, 0.0001, 0.01)
$\boldsymbol{\theta = 10°, \sigma = 0.3}$	1	(0.0001, 0.01)	(10, 0.0008, 0.01)
$\boldsymbol{\theta = 10°, \sigma = 0.7}$	1	(0.0008, 0.05)	(10, 0.0001, 0.01)
$\boldsymbol{\theta = 10°, \sigma = 1}$	0.5	(0.0004, 0.01)	(10, 0.0004, 0.01)
$\boldsymbol{\theta = 20°, \sigma = 0.3}$	1.5	(0.001, 0.05)	(10, 0.0005, 0.05)
$\boldsymbol{\theta = 20°, \sigma = 0.7}$	1	(0.001, 0.05)	(10, 0.0005, 0.01)
$\boldsymbol{\theta = 20°, \sigma = 1}$	0.5	(0.0005, 0.01)	(10, 0.0005, 0.01)

Table 4.4 and 4.5 exhibit the Dice score and RMSE of the 4 models, respectively. From Table 4.4 we can see that TV, $\mathbf{L}_4$ and $\mathbf{L}_p$ models have compatible performance, whereas the Dice score of SIRT is the least satisfactory. When $\theta = 5°$, TV, $\mathbf{L}_4$ and $\mathbf{L}_p$ can predict all pixel values correctly. When $\theta = 10°$, the Dice score for TV decreases as σ increases. Both $\mathbf{L}_4$ and $\mathbf{L}_p$ can still predict all pixel values correctly on average when $\sigma = 0.4$ and $\sigma = 0.7$. When $\theta = 20°$, $\mathbf{L}_4$ and $\mathbf{L}_p$ have Dice score 1 when $\sigma = 0.3$. As σ further increases, the performance of TV, $\mathbf{L}_4$ and $\mathbf{L}_p$ methods are similar, with $\mathbf{L}_4$ having slightly higher Dice scores than the other two.

Table 4.4: Dice score comparison of four different models with $(\theta, \sigma) \in \{5°, 10°, 20°\} \times \{0.3, 0.07, 1\}$. Tested on $N_3 = 100$ MNIST digit 0 CT images. Parameter values used for each model is listed in Table 4.3

(a) $\boldsymbol{\theta = 5°}$

Method	$\boldsymbol{\sigma = 0.3}$	$\boldsymbol{\sigma = 0.7}$	$\boldsymbol{\sigma = 1}$
SIRT	0.9980 ± 0.0018	0.9971 ± 0.0022	0.9963 ± 0.0024
TV	$\mathbf{1.0} \pm 0.0$	$\mathbf{1.0} \pm 0.0$	1.0 ± 0.0002
$\boldsymbol{L_4}$	$\mathbf{1.0} \pm 0.0$	$\mathbf{1.0} \pm 0.0001$	$\mathbf{1.0} \pm 0.0$
$\boldsymbol{L_p}$	$\mathbf{1.0} \pm 0.0$	$\mathbf{1.0} \pm 0.0$	$\mathbf{1.0} \pm 0.0$

(b) $\boldsymbol{\theta = 10°}$

Method	$\boldsymbol{\sigma = 0.3}$	$\boldsymbol{\sigma = 0.7}$	$\boldsymbol{\sigma = 1}$
SIRT	0.9979 ± 0.0019	0.9960 ± 0.0027	0.9935 ± 0.0029
TV	$\mathbf{1.0} \pm 0.0$	0.9999 ± 0.0003	0.9992 ± 0.0012
$\boldsymbol{L_4}$	$\mathbf{1.0} \pm 0.0$	$\mathbf{1.0} \pm 0.0001$	0.9996 ± 0.0006
$\boldsymbol{L_p}$	$\mathbf{1.0} \pm 0.0$	$\mathbf{1.0} \pm 0.0001$	$\mathbf{0.9997} \pm 0.0006$

(c) $\boldsymbol{\theta = 20°}$

Method	$\boldsymbol{\sigma = 0.3}$	$\boldsymbol{\sigma = 0.7}$	$\boldsymbol{\sigma = 1}$
SIRT	0.9959 ± 0.0034	0.9926 ± 0.0046	0.9881 ± 0.0049
TV	0.9997 ± 0.0009	0.9982 ± 0.0019	0.9949 ± 0.0033
$\boldsymbol{L_4}$	$\mathbf{1.0} \pm 0.0$	$\mathbf{0.9997} \pm 0.0006$	$\mathbf{0.9953} \pm 0.0029$
$\boldsymbol{L_p}$	$\mathbf{1.0} \pm 0.0$	0.9972 ± 0.0024	0.9952 ± 0.0029

Table 4.5: RMSE comparison of four different models with $(\theta, \sigma) \in \{5°, 10°, 20°\} \times \{0.3, 0.07, 1\}$. Tested on $N_3 = 100$ MNIST digit 0 CT images. Parameter values used for each model is listed in Table 4.3

(a) $\boldsymbol{\theta = 5°}$

Method	$\boldsymbol{\sigma = 0.3}$	$\boldsymbol{\sigma = 0.7}$	$\boldsymbol{\sigma = 1}$
SIRT	0.1263 ± 0.0076	0.1333 ± 0.0072	0.1419 ± 0.0072
TV	0.0358 ± 0.0013	0.0761 ± 0.0024	0.0891 ± 0.0036
$\boldsymbol{L_4}$	0.0051 ± 0.0001	0.0117 ± 0.0019	$\mathbf{0.0281} \pm 0.0008$
$\boldsymbol{L_p}$	$\mathbf{0.0049} \pm 0.0001$	$\mathbf{0.0115} \pm 0.0003$	$\mathbf{0.0281} \pm 0.0008$

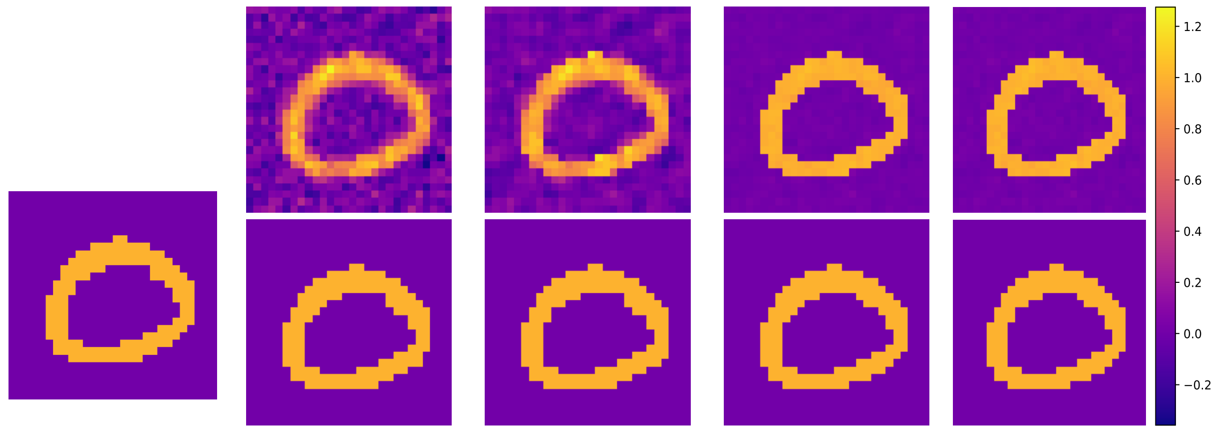
(b) $\boldsymbol{\theta = 10°}$

Method	$\boldsymbol{\sigma = 0.3}$	$\boldsymbol{\sigma = 0.7}$	$\boldsymbol{\sigma = 1}$
SIRT	0.9979 ± 0.0019	0.1444 ± 0.0077	0.1595 ± 0.007
TV	0.0459 ± 0.0033	0.0850 ± 0.0034	0.1000 ± 0.0058
$\boldsymbol{L_4}$	0.0075 ± 0.0002	0.0484 ± 0.0017	0.0274 ± 0.0076
$\boldsymbol{L_p}$	$\mathbf{0.0068} \pm 0.0002$	$\mathbf{0.0161} \pm 0.0018$	$\mathbf{0.0272} \pm 0.0073$

(c) $\boldsymbol{\theta = 20°}$

Method	$\boldsymbol{\sigma = 0.3}$	$\boldsymbol{\sigma = 0.7}$	$\boldsymbol{\sigma = 1}$
SIRT	0.1470 ± 0.0105	0.1662 ± 0.0098	0.1872 ± 0.0099
TV	0.0690 ± 0.0106	0.1054 ± 0.0106	0.1264 ± 0.011
$\boldsymbol{L_4}$	0.0221 ± 0.0008	$\mathbf{0.0444} \pm 0.0042$	$\mathbf{0.0630} \pm 0.0191$
$\boldsymbol{L_p}$	$\mathbf{0.0175} \pm 0.0005$	0.0460 ± 0.0204	0.0644 ± 0.0197

As for RMSE, it can be seen from Table 4.5 that model $\mathbf{L}_4$ and $\mathbf{L}_p$ have significantly lower RMSE values than the other two methods in all (θ, σ) settings. Between model $\mathbf{L}_4$ and $\mathbf{L}_p$, model $\mathbf{L}_p$ usually has the lower mean RMSE values, with the exception when $(\theta, \sigma) = (20^\circ, 0.7)$ and $(20^\circ, 1)$.



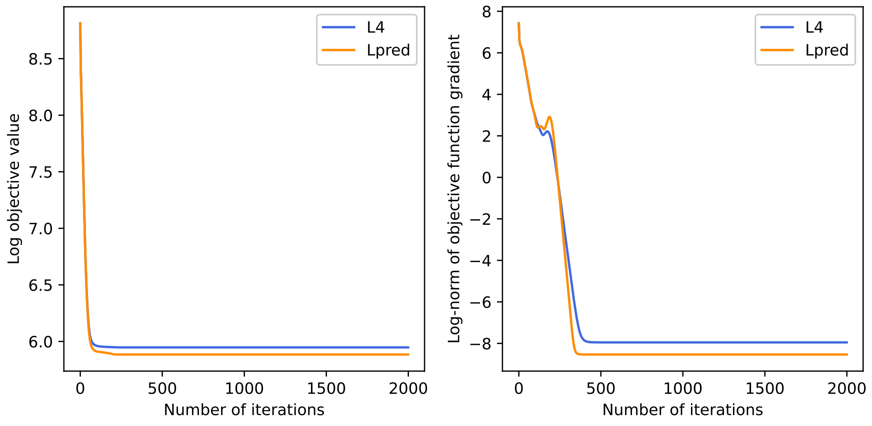
(a) Ground truth $\bar{\mathbf{u}}$ (b) SIRT, $d = .9962$ (c) TV, $d = .9962$ (d) $\mathbf{L}_4$, $d = .9962$ (e) $\mathbf{L}_p$, $d = .9974$

Figure 4.3: An example of CT image reconstruction of a digit 0 image in the MNIST. The top row consists of the predicted images using each method. The bottom row is the corresponding binary versions.

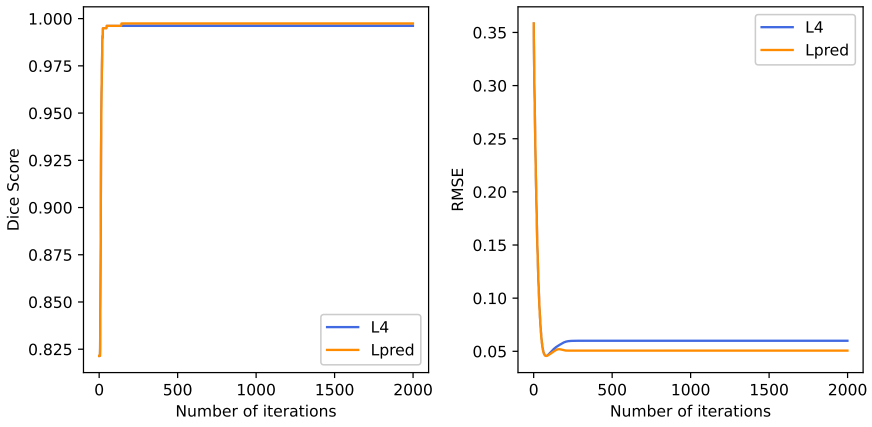
Figure 4.3 shows the reconstruction of the ground truth illustrated in Figure 4.2a, with Figure 4.2c, $\mathbf{f}^{\boldsymbol{\delta}}$, as our input data. Figure 4.3b and 4.3c indicate that the noise in the solutions from SIRT and TV are quite large, whereas the solutions given by $\mathbf{L}_4$ and $\mathbf{L}_p$, shown in Figure 4.3d and 4.3e are almost identical to their binary counterpart. This result agrees with the test results shown in Table 4.5.

Figure 4.4 demonstrates the log objective values, log norm of objective gradient, Dice score and RMSE against proximal gradient descent iterations when solving the reconstruction problem of Figure 4.2. Note that despite we plotted the log objective values for both $\mathbf{L}_4$ and $\mathbf{L}_p$ in Figure 4.4a (left), there is no point in comparing these two. This is because that the two models use different graph Laplacian matrices, therefore the objective functions are different. Figure 4.4a (right) suggests that it takes circa 500 iterations for GL regularisation to converge. It is worth noting that in the RMSEs against iteration plot in Figure 4.4b, the RMSE for both model increase after around 30 iterations. This seems to indicate a possible overfitting.

Our results show that graph GL regularisation model (2.23) works well in reconstructing binary CT images. When σ is low and there are adequate projection data, i.e. when θ is small, our model has comparable performance as TV regularisation. When σ is large and there are only limited projections, the graph GL regularisation outperforms the TV regularisation. Comparing between models with $\mathbf{L}_4$ and $\mathbf{L}_p$, we find that there is little benefit to use $\mathbf{L}_p$, as the performance of these two models are nearly identical while it takes a few hours to learn the $\mathbf{L}_p$ matrix. Furthermore, it is worth mentioning that since a large number of iterations are required for the graph GL algorithm, the time needed for graph GL is significantly longer than TV and SIRT. For models $\mathbf{L}_4$ and $\mathbf{L}_p$, running 2000 iterations to reconstruct a 28×28 pixel image takes around 20 seconds. For TV and SIRT, using the same $\mathbf{K}$ and $\mathbf{f}^{\boldsymbol{\delta}}$, reconstructing an image of the same size take less than a second.



(a) Objective values and norm of the gradient at $\mathbf{u}^{(k)}$



(b) Error metrics, Dice score and RMSE, at $\mathbf{u}^{(k)}$

Figure 4.4: Top: objective values and norm of the gradient at $\mathbf{u}^{(k)}$ for $k \in \{1, ...2000\}, k \in \mathbb{Z}_+$ when using $\mathbf{L}_4$ and $\mathbf{L}_p$ in the model. The stepsize α is 0.0005. Both plots are plotted in log scale. Bottom: Dice score and RMSE for $\mathbf{u}^{(k)}$ at each iteration.

4.2 A Real Number (Made of Playdough)

To test how the model performance in practice, we use the micro-CT scanner at CWI to scan a number 0 made of playdough. The playdough 0 is of the size 10cm×8cm×1.5cm. The projections are made at every 0.2°. As our model is designed for reconstructing 2D images, we use only the data of the central slice from the CT scan. We refer to the data of the central slice with projections made at every 0.2° as the full-sampled data. Figure 4.5 illustrates the central slice, which is 951×951 pixels in size and is reconstructed by the micro-CT scanner using the full-sample data. We consider Figure 4.5 as our reference. We observe a big air bubble in the upper left of the playdough 0. There are also some small air bubbles distributed throughout the playdough. We want to see if the GL regularisation can recreate the playdough 0 from subsampled measurements.

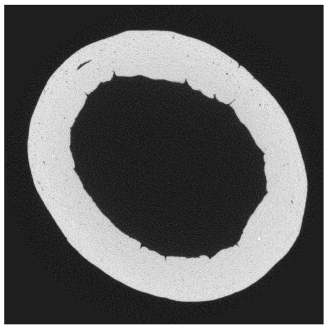


Figure 4.5: The central slice of the playdough 0 CT scan, with size 951×951 pixels. This image is reconstructed using the full-sampled data.

Data Processing and Experiment Description

A pre-experiment shows that the reconstructed image using the raw data has pixel values in the range $[-0.03, 0.027]$. However, the local minima of the double-well potential in the graph GL functional is at 0 and 1. Therefore, in order to get the most out of graph GL functional we first pre-process the raw data. This is done using flexCALC package[2].

1. Calculate the scaling factor ψ. We scale the raw data such that the reconstructed image will have pixel values around 0 and 1. We get the OTSU threshold based on the FBP reconstruction $\hat{\mathbf{u}}_{all}$ using the full sampled data, where $\hat{\mathbf{u}}_{all}$ is obtained using ASTRA Toolbox's built-in function. Then we segment the FBP reconstruction using the OTSU threshold to get a binary image $\hat{\mathbf{u}}^b_{all}$. The scaling factor ψ is then set to be

$$\psi = \frac{1}{\sum_i \hat{\mathbf{u}}^b_{all,i}} \sum_i \hat{\mathbf{u}}_{all,i} \cdot \mathbb{1}_{\{\hat{\mathbf{u}}^b_{all,i}=1\}}.$$

Dividing the raw measurements with ψ, we obtain the scaled measurements.

2. To make the dimension size easier for our algorithm to handle, we downsample the image resolution to 119×119 pixels by binning 8 pixels into 1 pixel. We perform angular sampling

[2]Implementation retrieved from https://github.com/cicwi/flexCALC

with the projection angles set at every 1° and 3.2°, respectively. In other words, we downsample the number of projections by a factor of 5 and 16, respectively, w.r.t. the original 0.2° increment. The discrete Radon transform $\mathbf{K}$ is then generated accordingly. Figure 4.6 demonstrates an example of the scaled, downsampled sinogram $\mathbf{f}^\delta$, with projections are made at 1° (In other words, we downsample the number of projections by a factor of 5).

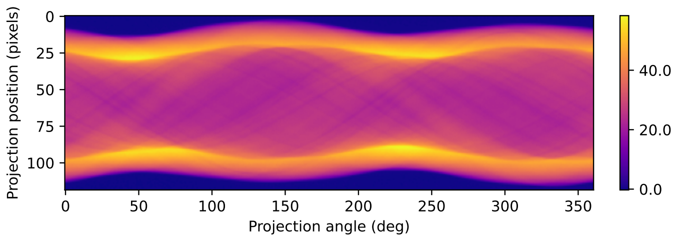


Figure 4.6: The scaled and subsampled sinogram of the central slice of the playdough 0, where the original sinogram is acquired using the micro-CT scanner.

Only the $\mathbf{L}_4$ model is used in this experiment, as we cannot learn the graph Laplacian matrix from images due to the lack of training data. There is also no hyperparameter tuning. We run Algorithm 2 for $K = 9000$ iterations. The range of α and values of (α, ε) used are listed in Table 4.6.

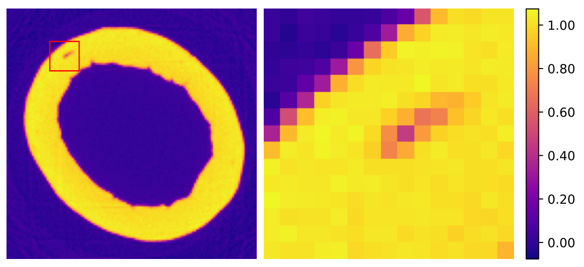
Table 4.6: The dimension of $\mathbf{K}$ in the two settings, the corresponding α range, and values of (α, ε) used.

Angular increment θ	dim($\mathbf{K}$)	α **range**	(α, ε) **value**
$\mathbf{1^\circ}$	42959×14161	$(0, 5.19 \times 10^{-5})$	$(5 \times 10^{-5}, 0.03)$
$\mathbf{3.2^\circ}$	13447×14161	$(0, 1.57 \times 10^{-4})$	$(1 \times 10^{-4}, 0.01)$

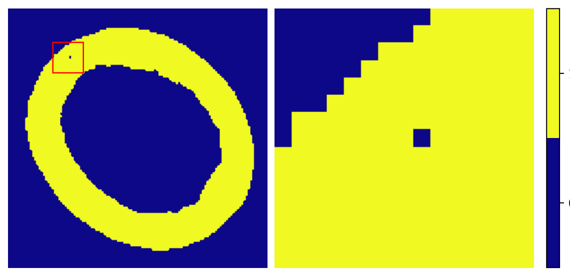
Results and Discussion

Figure 4.7 illustrate the result of using graph GL regularisation in the two settings. It is clear from the figure that the playdough 0 is recovered nicely. We can see that in both cases the air bubble on the top left corner is preserved. The size of the air bubble is much smaller than that of the original reconstruction in Figure 4.5. This is to be expected as the image size of our reconstruction is smaller. Comparing Figure 4.7a and 4.7c we find that the solution in 4.7a is much smoother. This is due to the fact that we have more projections data when $\theta = 1^\circ$. The resulting binary images Figure 4.7b and 4.7d are pretty similar, with 4.7b having a slightly more smooth edge. Figure 4.8 shows the FBP reconstruction on the binned data with no downsampling on the number of projections (i.e. using projections made on every 0.2° to reconstruct a 119×119 pixel image). In this case there is no air bubble left in the segmentation of its reconstruction. Comparing Figure 4.8a and Figure 4.7a, we find that the zoomed-in image of the reconstruction from our $\mathbf{L}_4$ model is similar to that of the FBP model. This indicates that our model can reconstruct the image with relatively limited data.

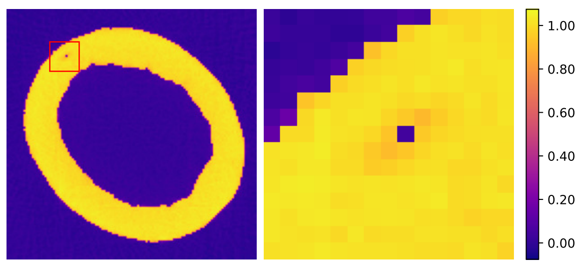
The number of iterations K required for our model to converge is much larger in this experiment, comparing to the 2000 iterations used in the experiment in Section 4.1. This can be explained by the small value of the stepsize α. The large dimension of $\mathbf{K}$ also means that our



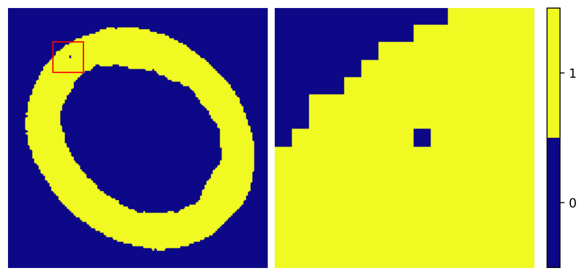
(a) $\hat{\mathbf{u}}$ and the zoomed-in image of the red box. Angular sampling at every 1°.



(b) $\hat{\mathbf{u}}^b$ and the zoomed-in image of the red box. Angular sampling at every 1°.

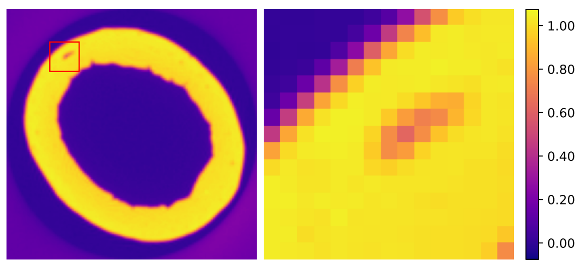


(c) $\hat{\mathbf{u}}$ and the zoomed-in image of the red box. Angular sampling at every 3.2°.

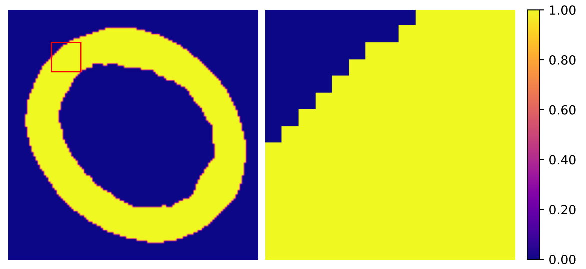


(d) $\hat{\mathbf{u}}^b$ and the zoomed-in image of the red box. Angular sampling at every 3.2°.

Figure 4.7: The reconstructed 119×119 pixel images suing $\mathbf{L}_4$ model and their binary counterparts in the two angular sampling cases. Parameter values used in the two cases are listed in Table 4.6.



(a) $\hat{\mathbf{u}}$ and the zoomed-in image of the red box. No angular sampling.



(b) $\hat{\mathbf{u}}^b$ and the zoomed-in image of the red box. No angular sampling.

Figure 4.8: The reconstructed 119×119 pixel images and their binary counterpart using FBP with no angular sampling.

model takes substantially longer time to produce a solution than the FBP reconstruction. It takes a few minutes for our model (2.23) to return a solution for both values of θ during our experiment.

5 Conclusions and Future Work

In this thesis, we examined the use of the graph Ginzburg-Landau functional as regularisation when reconstructing binary images from their CT scans. The experiments were carried out in two parts.

The first part was to study a reduced CT image reconstruction problem, where the graph GL regularisation was used in an image denoising problem. We established the feasibility of using Newton-CG as the minimiser to solve our model. Restricted to the non-convex objective function, the model were only tested on datasets with small image sizes. Next, the performance of our model when denoising images of handwritten digit 0 was tested, while we also studied the benefit of changing the graph Laplacian matrix in the graph GL functional. We found that the model works well for image denoising tasks. However, the benefit of using learned graph Laplacian matrices depends on the datasets. More specifically, when the dataset exhibits large variance in the images, the 4-regular graph Laplacian is a better choice for the graph GL regularisation.

The second part of the experiments was to investigate the performance of our model on binary CT image reconstruction, where we compare our model with SIRT and TV regularisation. For the MNIST dataset, our model outperforms the SIRT in every scenario where we subsampled projections and added different noise levels. Moreover, our model performs better than TV regularisation when using high noise data and is comparable to TV regularisation when using low and medium noise data. These results demonstrate that our model works well with noisy and subsampled data. However, we should note that our model takes much longer to run than TV and SIRT. Finally, we developed a way to apply our model to the real-world data through rescaling in the playdough experiment. The graph GL regularisation successfully preserved the finer details of the playdough 0 when reconstructing with subsampled data.

Possible future work are listed below.

- Apply minimisation algorithm that does not require convexity of the objective function to the GL regularisation. In our experiments, Newton-CG works relatively well with low-resolution images, even without the guarantee of convexity. However, a minimisation on non-convex models may bring more accurate results.
- Apply the proposed model on high-dimensional data. We have only tested the model with low-resolution images, due to the fact that the difficulty for Newton-CG to find a global minimum of the objective function increases as the dimension of the images grows. Hence, using an alternative minimisation algorithm might enable the model to be used with high-resolution images. This will broaden the usage of our model, since CT images are often of high-resolution.
- Develop a method to modify the GL functional such that the model can be extended to reconstruct non-binary images as well. Non-binary images are advantageous because they provide us the opportunity to extract additional information from their pixel values. This would also broaden the functionality of the graph GL regularisation.
- Use self-defined connectivity and edge weights of the graph to decide the graph Laplacian matrix in the graph GL functional. This can be done by e.g. extracting features from the images using some feature maps. This would allow us to incorporate more prior information on the images to the model.

References

Andersen, A. H., & Kak, A. C. (1984). Simultaneous algebraic reconstruction technique (sart): a superior implementation of the art algorithm. *Ultrasonic imaging*, *6*(1), 81–94.

Antil, H., Di, Z. W., & Khatri, R. (2020). Bilevel optimization, deep learning and fractional laplacian regularization with applications in tomography. *Inverse Problems*, *36*(6), 064001.

Beck, A. (2017). *First-order methods in optimization.* SIAM.

Bertozzi, A. L., & Flenner, A. (2012). Diffuse interface models on graphs for classification of high dimensional data. *Multiscale Modeling & Simulation*, *10*(3), 1090–1118.

Candès, E. J., Romberg, J., & Tao, T. (2006). Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on information theory*, *52*(2), 489–509.

Candes, E. J., Romberg, J. K., & Tao, T. (2006). Stable signal recovery from incomplete and inaccurate measurements. *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, *59*(8), 1207–1223.

Chambolle, A., & Lions, P.-L. (1997). Image recovery via total variation minimization and related problems. *Numerische Mathematik*, *76*, 167–188.

Condat, L. (2017). Discrete total variation: New definition and minimization. *SIAM Journal on Imaging Sciences*, *10*(3), 1258–1290.

Dong, X., Thanou, D., Frossard, P., & Vandergheynst, P. (2016). Learning laplacian matrix in smooth graph signal representations. *IEEE Transactions on Signal Processing*, *64*(23), 6160–6173.

Goldstein, T., & Osher, S. (2009). The split bregman method for l1-regularized problems. *SIAM journal on imaging sciences*, *2*(2), 323–343.

Hansen, P. C., Jørgensen, J., & Lionheart, W. R. (2021). *Computed tomography: algorithms, insight, and just enough theory.* SIAM.

Humbert, P., Le Bars, B., Oudre, L., Kalogeratos, A., & Vayatis, N. (2021). Learning laplacian matrix from graph signals with sparse spectral representation. *The Journal of Machine Learning Research*, *22*(1), 8766–8812.

Kalofolias, V. (2016). How to learn a graph from smooth signals. In *Artificial intelligence and statistics* (pp. 920–929).

Li, H., & Lin, Z. (2015). Accelerated proximal gradient methods for nonconvex programming. *Advances in neural information processing systems*, *28*.

Meng, B., Wang, J., & Xing, L. (2010). Sinogram preprocessing and binary reconstruction for determination of the shape and location of metal objects in computed tomography (ct). *Medical physics*, *37*(11), 5867–5875.

Merkurjev, E., Kostic, T., & Bertozzi, A. L. (2013). An mbo scheme on graphs for classification and image processing. *SIAM Journal on Imaging Sciences*, *6*(4), 1903–1930.

Modica, L. (1987). The gradient theory of phase transitions and the minimal interface criterion. *Archive for Rational Mechanics and Analysis*, *98*, 123–142.

Nesterov, Y., & Polyak, B. T. (2006). Cubic regularization of newton method and its global performance. *Mathematical Programming*, *108*(1), 177–205.

Nocedal, J., & Wright, S. J. (1999). *Numerical optimization.* Springer.

Pang, J., & Cheung, G. (2017). Graph laplacian regularization for image denoising: Analysis in the continuous domain. *IEEE Transactions on Image Processing*, *26*(4), 1770–1785.

Radon, J. (1986). On the determination of functions from their integral values along certain manifolds. *IEEE transactions on medical imaging*, *5*(4), 170–176.

Ramachandran, G., & Lakshminarayanan, A. (1971). Three-dimensional reconstruction from radiographs and electron micrographs: application of convolutions instead of fourier transforms. *Proceedings of the National Academy of Sciences*, *68*(9), 2236–2240.

Ravasi, M., & Vasconcelos, I. (2020). Pylops—a linear-operator python library for scalable algebra and optimization. *SoftwareX*, *11*, 100361.

Rudin, L. I., Osher, S., & Fatemi, E. (1992). Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena*, *60*(1-4), 259–268.

Shuman, D. I., Narang, S. K., Frossard, P., Ortega, A., & Vandergheynst, P. (2013). The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE signal processing magazine*, *30*(3), 83–98.

Tatiana A. Bubba. (2019). *The mathematics of x-ray tomography [slides].* Retrieved 22-07-2023, from https://www.fips.fi/slides/Bubba_SummerSchoolVFIP2019_1.pdf (Summer School on Very Finnish Inverse Problems)

Van Aarle, W., Palenstijn, W. J., Cant, J., Janssens, E., Bleichrodt, F., Dabravolski, A., ... Sijbers, J. (2016). Fast and flexible x-ray tomography using the astra toolbox. *Optics express*, *24*(22), 25129–25147.

van Gennip, Y. (2019). Graph ginzburg–landau: discrete dynamics, continuum limits, and applications. an overview. In *Proceedings of 44th sapporo symposium on partial differential equations* (Vol. 1).

van Gennip, Y., & Bertozzi, A. L. (2012). γ-convergence of graph ginzburg-landau functionals. *Adv. Differential Equations*, *17*(11-12), 1115–1180.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., ... SciPy 1.0 Contributors (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, *17*, 261–272. doi: 10.1038/s41592-019-0686-2

Zeng, J., Pang, J., Sun, W., & Cheung, G. (2019). Deep graph laplacian regularization for robust denoising of real images. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition workshops.*

Appendix A Model Robustness

In this section, we list some additional experiment results that did not make it into the main thesis, but could potentially demonstrate that the proposed model is robust.

A.1 Image Denoising on Handwritten Digit 6 Images

This section presents the outcome of the same image denoising experiment carried out in Section 3.2 using both MNIST and USPS datasets, but with handwritten digit 6 images. Digit 6, unlike digit 0, is not symmetric. Therefore, it could potentially be harder for the model to denoise. Table A.1 displays the test result on $N_2 = 20$ digit 6 MNIST images. Table A.2 displays the test result on $N_2 = 20$ digit 6 MNIST images. For MNIST dataset, model with $\mathbf{L}_p$ has better accuracies when $\sigma \in \{0.25, 0.3, 0.35\}$. However, for USPS dataset, model with $\mathbf{L}_4$ has better accuracies for all 4 values of σ. These results are identical to those of the digit 0 images. Hence, we come to the conclusion that our model is stable in terms of denoising various digits.

Table A.1: Performance comparison of two models with $\mathbf{L}_4$ and $\mathbf{L}_p$. Tested on $N_2 = 20$ MNIST the digit 6 images. For reference, the mean RMSE and Dice score between the noisy images $\hat{X}$ and their ground truth X^b of corresponding σ are shown in the parentheses.

	$\sigma = 0.15$		$\sigma = 0.25$		$\sigma = 0.3$		$\sigma = 0.35$	
	(0.1509/1)		(0.2376/0.9781)		(0.3036/0.9539)		(0.3421/0.9336)	
	$\mathbf{L_4}$	$\mathbf{L_p}$	$\mathbf{L_4}$	$\mathbf{L_p}$	$\mathbf{L_4}$	$\mathbf{L_p}$	$\mathbf{L_4}$	$\mathbf{L_p}$
β	N.A.	2	N.A.	3	N.A.	6	N.A.	3
α	0.04	0.2	0.2	1.1	0.2	0.9	0.2	1.6
ε	0.03	0.2	0.09	0.4	0.2	0.5	0.2	0.4
RMSE	**0.0474**	0.0586	0.1066	**0.0802**	0.1789	**0.1296**	0.1972	**0.1365**
(st.d)	0.0054	0.0068	0.0427	0.0324	0.0333	0.0508	0.0272	0.0455
Dice score	1	1	0.9875	**0.9953**	0.9641	**0.9820**	0.9570	**0.9773**
(st.d)	0	0	0.0161	0.0112	0.0232	0.0248	0.0214	0.0167
Successful conv. ratio	1	1	1	1	1	1	1	0.95

Table A.2: Performance comparison of two models with $\mathbf{L}_4$ and $\mathbf{L}_p$. Tested on $N_3 = 100$ USPS images of the digit 6. For reference, the mean RMSE and Dice score between the noisy images $\hat{X}$ and their ground truth X^b of corresponding σ are shown in the parentheses.

	$\sigma = 0.15$		$\sigma = 0.25$		$\sigma = 0.3$		$\sigma = 0.35$	
	(0.1504/0.9997)		(0.2513/0.9771)		(0.2991/0.9525)		(0.3510/0.9243)	
	$\mathbf{L_4}$	$\mathbf{L_p}$	$\mathbf{L_4}$	$\mathbf{L_p}$	$\mathbf{L_4}$	$\mathbf{L_p}$	$\mathbf{L_4}$	$\mathbf{L_p}$
β	N.A.	6	N.A.	12	N.A.	9	N.A.	12
α	0.9	1.1	0.2	1	0.4	0.9	0.4	1.2
ε	0.06	0.2	0.2	0.3	0.1	0.4	0.2	0.4
RMSE	**0.0314**	0.0404	0.1223	0.1157	**0.1153**	0.1554	0.1537	0.1903
(st.d)	(0.0051)	(0.0088)	(0.0101)	(0.0228)	(0.0289)	(0.0271)	(0.0242)	(0.0300)
Dice score	**0.9999**	0.9999	**0.9932**	0.9859	**0.9843**	0.9720	0.9748	0.9553
(st.d)	(0.0006)	(0.0087)	(0.0043)	(0.0070)	(0.0091)	(0.0124)	(0.0134)	(0.0164)
Successful conv. ratio	1	0.97	1	1	0.95	0.95	**1**	0.96

A.2 CT Image Reconstruction with Uniform Noise

This section exhibits the results of the same CT image reconstruction experiment carried out in Section 3.1 using MNIST dataset, but with uniform noise. This means that the noisy sinograms are of the form

$$f^{\boldsymbol{\delta}} = f + \sigma\boldsymbol{\delta}, \qquad \boldsymbol{\delta} \sim \mathcal{U}[0,1]^n.$$

Table A.3 and A.4 show the Dice score and RMSE of the 4 models when reconstructing digit 0 images in various projection angle and noise level settings. The projection angles are varied in increments of θ from 0° to 360°, where $\theta \in \{5°, 10°, 20°, 30°\}$. The noise levels are chosen to be $\sigma \in \{1, 2, 5\}$.

We can see from Table A.3 that TV regularisation yields a good Dice score when $\sigma = 1$ and $\sigma = 1$. However, as σ increases to 5, GL regularisations, both with $\mathbf{L}_4$ and $\mathbf{L}_p$, give the best performance. This holds for all θ values in our experiment. We can see from Table A.3 that GL regularisations output smaller RMSE values when $\sigma = 1$ and $\sigma = 2$. When $\sigma = 5$, however, it's usually TV regularisation that has the lowest RMSE values.

With the above analysis, we conclude that our model is stable when changing the Gaussian noise to uniform noise. Moreover, it outperforms TV regularisation in terms of Dice score when the noise level is large.

Table A.3: Dice score comparison of four different models. Tested on $N_3 = 200$ sinograms of MNIST digit 0 images.

(a) $\boldsymbol{\theta = 5°}$

Method	$\boldsymbol{\sigma = 1}$	$\boldsymbol{\sigma = 2}$	$\boldsymbol{\sigma = 5}$
SIRT	0.9978 ± 0.0018	0.9967 ± 0.0024	0.9879 ± 0.0042
TV	$\mathbf{1} \pm 0$	$\mathbf{1} \pm 0$	0.9974 ± 0.0023
$\boldsymbol{L_4}$	$\mathbf{1} \pm 0$	$\mathbf{1} \pm 0$	$\mathbf{0.9990} \pm 0.0010$
$\boldsymbol{L_p}$	$\mathbf{1} \pm 0$	$\mathbf{1} \pm 0$	$\mathbf{0.9990} \pm 0.0010$

(b) $\boldsymbol{\theta = 10°}$

Method	$\boldsymbol{\sigma = 1}$	$\boldsymbol{\sigma = 2}$	$\boldsymbol{\sigma = 5}$
SIRT	0.9978 ± 0.0018	0.9961 ± 0.0025	0.9743 ± 0.0065
TV	$\mathbf{1} \pm 0$	$\mathbf{1} \pm 0.0001$	0.9907 ± 0.0041
$\boldsymbol{L_4}$	$\mathbf{1} \pm 0$	$\mathbf{1} \pm 0.0002$	0.9956 ± 0.0023
$\boldsymbol{L_p}$	$\mathbf{1} \pm 0$	$\mathbf{1} \pm 0.0002$	$\mathbf{0.9957} \pm 0.0022$

(c) $\boldsymbol{\theta = 20°}$

Method	$\boldsymbol{\sigma = 1}$	$\boldsymbol{\sigma = 2}$	$\boldsymbol{\sigma = 5}$
SIRT	0.9959 ± 0.0033	0.9932 ± 0.0038	0.9476 ± 0.0104
TV	$\mathbf{0.9997} \pm 0.0008$	$\mathbf{0.9990} \pm 0.0012$	0.9832 ± 0.0051
$\boldsymbol{L_4}$	0.9994 ± 0.0011	0.9988 ± 0.0012	$\mathbf{0.9892} \pm 0.0049$
$\boldsymbol{L_p}$	0.9996 ± 0.0009	$\mathbf{0.9990} \pm 0.0011$	$\mathbf{0.9892} \pm 0.0049$

(d) $\boldsymbol{\theta = 30°}$

Method	$\boldsymbol{\sigma = 1}$	$\boldsymbol{\sigma = 2}$	$\boldsymbol{\sigma = 5}$
SIRT	0.9874 ± 0.0072	0.9840 ± 0.0074	0.9262 ± 0.0113
TV	$\mathbf{0.9951} \pm 0.0056$	$\mathbf{0.9926} \pm 0.0057$	0.9709 ± 0.0081
$\boldsymbol{L_4}$	0.9947 ± 0.0053	0.9923 ± 0.0056	0.9787 ± 0.0070
$\boldsymbol{L_p}$	0.9948 ± 0.0054	$\mathbf{0.9926} \pm 0.0056$	$\mathbf{0.9789} \pm 0.0068$

Table A.4: RMSE comparison of four different models. Tested on $N_3 = 200$ sinograms of MNIST digit 0 images.

(a) $\boldsymbol{\theta = 5^\circ}$

Method	$\boldsymbol{\sigma = 1}$	$\boldsymbol{\sigma = 2}$	$\boldsymbol{\sigma = 5}$
SIRT	0.1285 ± 0.0074	0.1393 ± 0.0072	0.1987 ± 0.0063
TV	0.0418 ± 0.001	0.0719 ± 0.0020	$\mathbf{0.1498} \pm 0.0065$
$\boldsymbol{L_4}$	0.0214 ± 0.0006	0.0703 ± 0.0023	0.1629 ± 0.0138
$\boldsymbol{L_p}$	$\mathbf{0.0208} \pm 0.0005$	$\mathbf{0.0691} \pm 0.0021$	0.1629 ± 0.0138

(b) $\boldsymbol{\theta = 10^\circ}$

Method	$\boldsymbol{\sigma = 1}$	$\boldsymbol{\sigma = 2}$	$\boldsymbol{\sigma = 5}$
SIRT	0.1330 ± 0.0076	0.1473 ± 0.0075	0.2245 ± 0.0075
TV	0.0500 ± 0.0030	0.0919 ± 0.0028	$\mathbf{0.1726} \pm 0.0102$
$\boldsymbol{L_4}$	0.0444 ± 0.0028	0.0789 ± 0.0025	0.1856 ± 0.0241
$\boldsymbol{L_p}$	$\mathbf{0.0403} \pm 0.0023$	$\mathbf{0.0670} \pm 0.0022$	0.1851 ± 0.0237

(c) $\boldsymbol{\theta = 20^\circ}$

Method	$\boldsymbol{\sigma = 1}$	$\boldsymbol{\sigma = 2}$	$\boldsymbol{\sigma = 5}$
SIRT	0.1488 ± 0.0106	0.1672 ± 0.0096	0.2632 ± 0.0123
TV	0.0717 ± 0.0104	0.1064 ± 0.0077	$\mathbf{0.1935} \pm 0.0088$
$\boldsymbol{L_4}$	0.0755 ± 0.0081	0.0947 ± 0.0158	0.2141 ± 0.0224
$\boldsymbol{L_p}$	$\mathbf{0.0713} \pm 0.0081$	$\mathbf{0.0921} \pm 0.0161$	0.2138 ± 0.0222

(d) $\boldsymbol{\theta = 30^\circ}$

Method	$\boldsymbol{\sigma = 1}$	$\boldsymbol{\sigma = 2}$	$\boldsymbol{\sigma = 5}$
SIRT	0.1846 ± 0.0118	0.1999 ± 0.0111	0.2849 ± 0.0109
TV	0.1275 ± 0.0192	0.1470 ± 0.0173	0.2181 ± 0.0141
$\boldsymbol{L_4}$	0.1185 ± 0.0148	0.1339 ± 0.0179	$\mathbf{0.1463} \pm 0.0203$
$\boldsymbol{L_p}$	$\mathbf{0.1158} \pm 0.0150$	$\mathbf{0.1330} \pm 0.0194$	0.1810 ± 0.0253