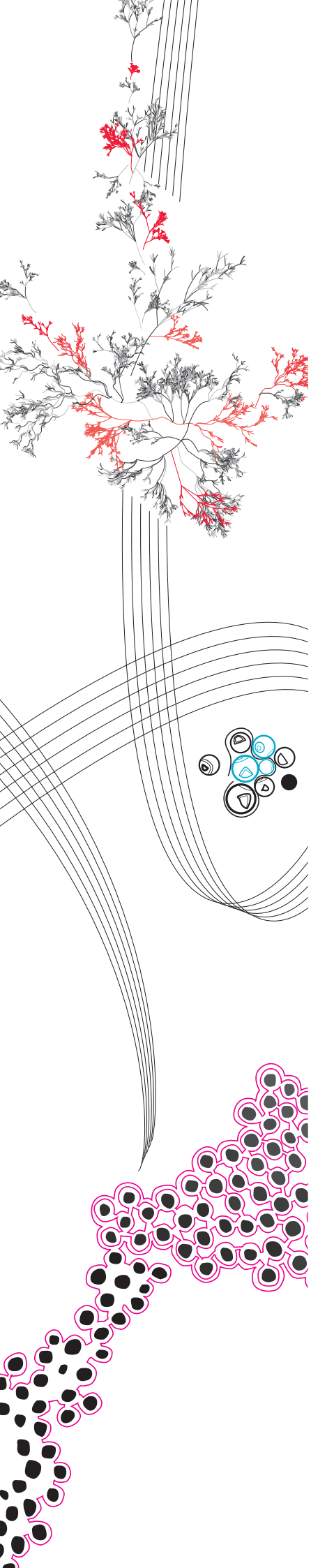


MASTER THESIS – APPLIED MATHEMATICS

On the importance of mutual coherence for image reconstruction with neural networks

P. Loohuis

Faculty of Electrical Engineering, Mathematics and Computer Science (EEMCS)

Specialization: Systems Theory, Applied Analysis and Computational Science (SACS)

Chair: Applied Analysis

Graduation committee:
Prof. Dr. C. Brune
Dr. Ir. G. Meinsma
Y.E. Boink MSc.

Daily supervisor:
Y.E. Boink MSc.

Date of presentation:
November 13, 2020

UNIVERSITY OF TWENTE.

Abstract

We consider the issue of solving inverse problems with a convolutional neural network (CNN). We introduce a left preconditioning method that lowers the mutual coherence of the preconditioned forward mapping as much as possible and examine whether this is beneficial when the inverse problem is solved with a CNN. We consider a CNN because this network contains less trainable parameters, which makes it a popular option for learning the inverse mapping. This type of network, however, can only be used when the forward mapping is a local mapping. If this is not the case, e.g. when the Radon transform is used, then preconditioners can be used in order to transform the forward mapping to one that is close to the identity matrix. These mappings can have a low mutual coherence, which can be beneficial when solving inverse problems with neural networks. Besides that, we propose a framework based on information theory that connects inverse problems, compressed sensing and preconditioning via the Bayesian perspective on inverse problems. This framework enables us to analyze the effects of our preconditioning method. We show that the mutual coherence depends on the amount of regularization applied to the inverse problem, where a low mutual coherence implies small amounts of regularization. This has a big effect on the noise present in the inverse problem, leading to less detailed reconstructions. We show that when only left preconditioners are used, lowering the mutual coherence as much as possible is therefore not beneficial. In the discussion, we explain that lowering the mutual coherence can be beneficial when right preconditioners are also taken into account.

Keywords. Inverse problems, mutual coherence, preconditioning, information theory, Bayesian view, Radon transform

Acknowledgements

With this thesis I finish five beautiful years as an Applied Mathematics student. This final academic year was the most memorable year of my life. It started with a fantastic internship in Seoul, South Korea. I really want to thank prof. dr. Wonshik Choi for giving me the opportunity to work in your department at the Korea University. Furthermore, I owe a lot to Seokchan, Hojun, Jungho, Yongwoo, Pilsang, Jaechul, Eun Kyeong, Cham and Ilayda for making sure I had the best time of my life. The time I spend in Seoul was truly amazing and I will forever cherish the memories we made together.

Secondly, I want to thank my daily supervisor Yoeri Boink for guiding me during the second part of my academic year, i.e. this final project. To be honest, this part was quite challenging since the Covid-19 pandemic forced me to work from home. Because of your great supervision, your sharp mind and the jokes, I considered the time I spend on this project as enjoyable. I'm very grateful for that.

Next, I would like to thank prof. dr. Christoph Brune. Your enthusiasm and optimism was infectious and talking to you was always pleasant. You were always there for advice and for mathematical discussions. Many thanks for the time and effort you invested in me.

Furthermore, I want to say thanks to my family, Bennie and friends from Rijssen for supporting me, motivating me and giving me great distraction from my studies. Life would be a much dimmer place when you guys are not around. I also want to thank David, Dieuw, Douwe, Frank, Ma(a)ike(l), Nico, Phteven, S(chr)ander and Sjaak for the infinite amount of 'bakkies' and great chats. Without you my time as a student would not have been the same and I will definitely miss our time together as students. It was really a joy to work with you, hang out with you and speculate with you about a possible 'frikandellen party' in 'het hok opwaarts' at the S(N)ACS floor.

Finally, I would like to thank my roommates Bauwe, Jelte and Ruben. There is never a boring day at home, the atmosphere is perfect and we always laugh together. Everyday I go home with a smile.

Contents

Chapter 1

Introduction

The emergence of neural networks pushed the boundaries of performance in many fields of application [60, 92], where development of better hardware enabled researchers to use bigger networks with more expressive power. The network architecture choice is decisive for the performance and depends on the problem at hand. One of the factors that is important for that choice is the amount of available training samples. When a limited amount is available, the preferred network architecture is one with a limited number of trainable parameters in order to reduce the chance of overfitting [3]. A popular option is the so-called convolutional neural network (CNN). This type of network extracts the spatial features and dependencies of local areas in an input layer by applying a weighted sum between the pixels of that neighbourhood and a sliding kernel. Since CNNs contain limited trainable parameters, training is less time consuming.

Solving an inverse problem with a CNN means that the network learns the inverse mapping of the forward mapping. Since CNNs extract the features of local regions, it is preferable to have a forward mapping that acts somewhat locally on the data. If not, then the inverse problem can be transformed to a different problem which is better solvable with a CNN. For discrete inverse problems preconditioning methods [9] can help to transform the forward mapping of the inverse problem to a mapping that is close to the identity matrix by means of a matrix multiplication. For ill-posed inverse problems it is still an open question what type of preconditioner is best and what parameters are able to give valuable insight in the quality of the preconditioner.

Forward mappings close to the identity matrix play an important role in the field of compressed sensing [27]. These matrices can have a low mutual coherence, which then result in better [55] and faster [75] recovery of the input by iterative solvers. It is also shown that a low mutual coherence of the forward mapping can be beneficial for the recovery of inputs when solved with neural networks [91]. In [91] the authors showed that a low mutual coherence was beneficial for the classification accuracy of a neural network.

In this thesis the effect of a low mutual coherence of the forward mapping on the reconstruction quality is investigated in case when the inverse problem is solved with a CNN and limited number of training samples. The mutual coherence of the forward mapping is lowered by the use of preconditioning. The discrete ill-posed limited-angle Radon transform is the forward mapping, where the use of a limited number of measurement angles makes the problem more challenging to solve. Research on better methods for solving problems with this forward mapping is practically relevant since measurements of an X-ray computed tomography (CT) scanner are obtained by using the Radon transform. The case of limited measurement angles is of particular interest due to the concerns on the exposure of high amounts of X-ray radiation [63].

1.1 Contribution

The contribution of this thesis consists of two parts. First, we introduce a preconditioning approach that is unique to an inverse problem with the Radon transform forward mapping and CNN as solution method. This approach aims at lowering the mutual coherence of the preconditioned forward mapping as much as possible and examines if this is beneficial for the reconstructions of the CNN. To the best of our knowledge, this is the first time the role of the mutual coherence is examined in this manner. The preconditioning method can be seen as a first reconstruction phase and its outcome will form the input for the CNN. A similar approach is done by *Jin et al.* in [52], where the authors used the filtered backprojection as their first reconstruction phase. Our approach differs from that method in the sense that our first reconstruction method is based on compressed sensing rather than the analytical solution of the Radon transform. Another difference is the fact that we give a thorough analysis on the effects of our preconditioning method, which is not done in [52].

The analysis is part of our second contribution, in which we present a framework that connects the fields of well-posed inverse problems, preconditioning and compressed sensing via information theory. This framework shows that the mutual coherence can give insight on how much information an inverse problem contains and on how preconditioning can affect this amount of information. Besides that, it is discussed that the connection does not hold for ill-posed inverse problems. It is demonstrated that the general treatment of ill-posed inverse problems, i.e. regularization, does not fit into the theory. This gives ideas for further theoretical research on connecting these fields for ill-posed problems, which then can possibly be used for the development of new preconditioning methods.

1.2 Thesis outline

First, in Chapter 2, background information is given on inverse problems, neural networks, compressed sensing and preconditioning, after which the connecting framework is presented. Chapter 3 proposes our method and starts with a general idea of the method before explaining the different parts of the method in detail. Chapter 4 provides details on the experimental setup of our thesis. It starts with a brief introduction of the Radon transform and the analytical way of reconstructing the input. Next, the details are given on the used data sets and how the different parameters are tuned. Then, two different reconstruction methods are explained, including their implementation details. Finally, the quality measures are given that are used to assess the quality of the reconstructions. Chapter 5 contains the results of the estimated parameters and the two reconstruction methods. The contribution of each part of our proposed method is subsequently examined and an analysis of our proposed method is given. Finally, in Chapter 6 a conclusion of our research is given and ideas for further research are formulated.

Chapter 2

Theoretical background

This chapter contains the necessary background information for our proposed method. First a brief introduction to inverse problems is given. These inverse problem can be solved via iterative solvers and neural networks. Background information on the latter is given in the second section. Afterwards, the topics of compressed sensing and preconditioning are introduced. Finally, the connecting framework is presented, which is used for the analysis of our proposed method.

2.1 Inverse problems

This section contains some background information on inverse problems. It starts with a brief introduction on the definition of inverse problems and some problems one may encounter when handling these problems. Afterwards, the variational view on inverse problems is discussed.

2.1.1 Introduction to inverse problems

Inverse problems is a field in applied mathematics that aims to recover an unknown object from the observed measurements that are induced by this object. These inverse problems appear in many field of applications, like geophysics, image processing, computed tomography (CT) and non-destructive testing [44]. We can mathematically describe an inverse problem as recovering unknown $x \in \mathcal{X}$ that is a solution of the equation

$$Ax = y \tag{2.1}$$

where $y \in \mathcal{Y}$ are the given measurements and $A : \mathcal{X} \to \mathcal{Y}$ is the forward mapping that maps elements from $\mathcal{X}$ to $\mathcal{Y}$, often both Banach spaces. The mapping A captures the (physical) system that replicates the measurements for a given object x. In this thesis we consider the finite dimensional case, in which the forward mapping is a matrix $A \in \mathbb{R}^{m \times n}$, $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^m$. The object that we want to recover can for instance be a vector or image, and it can be recovered by inverting the forward mapping. For many applications this inversion is not trivial due to the *ill-posedness* of the inverse problem. An inverse problem is labeled as ill-posed if it is not *well-posed*, which in Hadamard's sense is [41]:

Definition 1 (Well-posedness)**.** *Let $A : \mathcal{X} \to \mathcal{Y}$ be the forward mapping, then the inverse problem $Ax = y$ is called well-posed if holds:*

1. *There exists at least one solution.*
2. *The solution is unique.*
3. *The solution depends continuously on the data.*

In many cases the third rule of Hadamard is violated, which means that the problem at hand is unstable under slight perturbations. The perturbations are in general embedded in the problem in the form of measurement noise η, i.e.

$$Ax + \eta = y \quad \text{or} \quad Ax = y^\eta, \tag{IP}$$

where $\eta \in \mathbb{R}^m$ and $\|y^\eta - y\| \leq \eta$. The instability is due to the clustering of very small singular values of A [43], which could lead to instability in recovering the object x. By including details about the solution into the problem one might make the solution useful and stable, which is called *regularization* [34].

2.1.2 Variational view on inverse problems

A popular way of treating inverse problems is by treating an optimization problem on a functional [81], where the following functional is minimized

$$\hat{x}_\alpha = \arg\min_x \mathcal{F}(x) = \arg\min_x \mathcal{D}(Ax, y^\eta) + \alpha\mathcal{R}(x), \tag{2.2}$$

where the *data fidelity* term is denoted by $\mathcal{D}(Ax, y^\delta)$, $\mathcal{R}(x)$ is called the regularizer and α is the regularization parameter. The data fidelity term needs to ensure that $Ax \approx y^\eta$ while the regularizer adds *a-priori* information about the solution x to problem. The regularization parameter α controls to which extent the minimizer of the functional will obey the data or the a-priori information. Higher noise levels in the data require more regularization, thus a higher α. The higher regularization parameter stabilizes the solution, but can be at the cost of the accuracy of the solution.

We consider an imaging inverse problem where both x and y are two dimensional images. Choosing the right data fidelity term and regularizer for this problem can be hard. The data fidelity term is dependent on the nature of the noise whereas the regularization term is dependent on a-priori information about the solution x. If for example the additive noise is Gaussian then often the least-squares data fidelity term is chosen, i.e. $\frac{1}{2}\|Ax - y^\eta\|^2$ [7]. For the regularization term it is important to have knowledge about the solution. If it is known that the solution is smooth then a successful regularized inverse problem is the *Tikhonov regularized inverse problem*

$$\hat{x}_\alpha = \arg\min_x \|Ax - y^\eta\|_2^2 + \alpha\|x\|_2^2\,, \tag{2.3}$$

where the solution can be approximated via direct inversion with the regularized Moore-Penrose inverse:

$$\hat{x}_\alpha = \left(A^T A + \alpha I\right)^{-1} A^T y^\eta. \tag{2.4}$$

When the images x contain sharp edges, Tikhonov regularization is not ideal due to its smoothing effect. An example of a regularizer that allows sharp edges in the solution is the *total variation*

(TV) regularizer, where the regularizer is equal to [18]:

$$\mathcal{R}(x) = \mathrm{TV}(x) = \sum_{j=1}^{n} \sqrt{v_j^2 + h_j^2}, \tag{2.5}$$

where $v = L_1 x$ and $h = L_2 x$. The L_1 and L_2 matrices are the horizontal and vertical first-order finite difference matrices, respectively, defined as

$$L_1 = I_{n\times n} \otimes L, \qquad L_2 = L \otimes I_{n\times n},$$

where $\otimes$ is the Kronecker product and

$$L = \begin{bmatrix} 1 & & & \\ -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \end{bmatrix}. \tag{2.6}$$

By minimizing the optimization problem (2.2) with this regularizer, one enforces sparsity in the gradient of solution x. This leads to a solution with piecewise constant parts, thus edges-preserving. The total variation regularizer is often used for image reconstruction problems and is introduced by *Rudin et al.* [79] in their famous work on noise removal with their designed *ROF*-model. For the analysis on the TV regularizer we refer to [14] and for more background information on different types of regularizers we refer to [7, 44].

For direct inversion of regularized inverse problems, like in (2.4), it is important to select an appropriate value of the regularization parameter α. Background information on methods that can determine an appropriate value of α can be found in [6]. Direct inversion is, however, not always applicable. Many inverse problems contain large matrices, which makes direct inversion seldom practically due to its computational efficiency and/or infeasible storage [44]. Then, iterative solvers can be used in order to approximate the solution of the inverse problem. Background information on iterative solvers for inverse problems can be found in [22, 68]. Another option for solving inverse problems is the use of neural networks, which is discussed next.

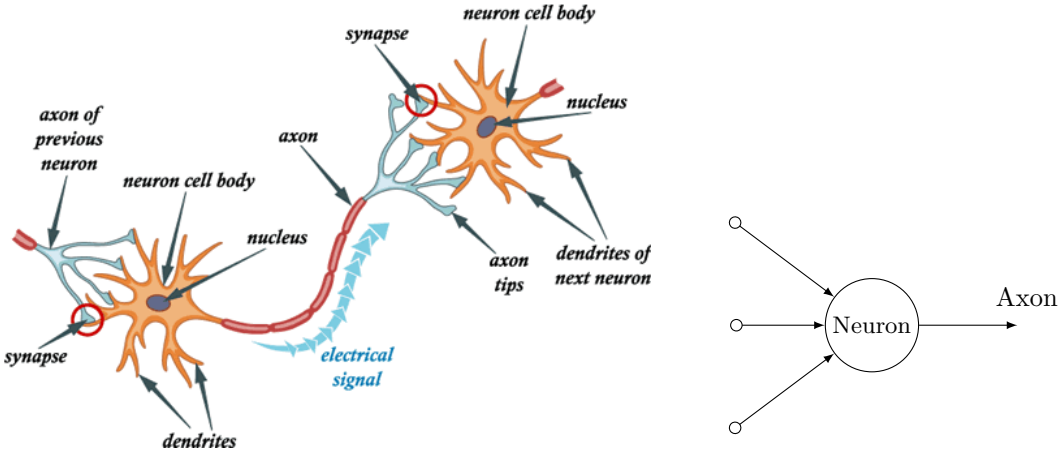
2.2 Neural networks

Instead of solving inverse problems via the traditional way, one can also use a neural network. This section contains the basics of neural networks. The section starts with a brief history and also discusses some of the basic architectures. Later the core of deep neural networks is explained and afterwards our network of choice is discussed in more detail.

2.2.1 Introduction to neural networks

Brief history

Neural networks are based on the learning process that occurs in the brain of a human being. The brain contains cells (*neurons*) and are connected to other cells by *axons* and *synapses*, which is illustrated in Figure 2.1a. This connected network of cells is artificially represented in a network of computational unit, represented by neurons. Weights connect the different neurons and represent the strength of the synaptic connections in the biological model [3]. This artificial representation of a biological neural network is illustrated in 2.1b.



(a) *Biological neural network* **(b)** *Artificial neural network*

Figure 2.1: *Illustration of connections between neurons via axons: biological versus artificial.*

The input of a neural network is formed by pairs, e.g. images and their labels or ground truth. The output values of a neuron are calculated by computing a function of the input. The output is compared to the ground truth and the corresponding error is the feedback for the network. Based on this error, weights will be adjusted such that the function can make a better prediction. The adjustment of weights is called *learning* or *training*.

One of the first neural networks that was capable of training was introduced by Rosenblatt in 1957 [77] and is the foundation of modern artificial intelligence [3, 40]. His work was the extension of an early brain model made by McCulloch and Pitts [65]. Rosenblatt's model, the *perceptron*, used inputs that where either on (1) or off (-1) and the output was a weighted sum of these inputs followed by a thresholding. This single layer perceptron caused a lot of excitement in the world of neural networks. This excitement lasted until Minsky and Papert [67] published a book on the perceptron, proving that the expressive power of single layer perceptrons was limited [3].

The interest in this field was revived when Werbos introduced the *backpropagation algorithm* [89]. It is a dynamic programming method that enables the possibility to train the weights automatically, which made training of a neural network much more efficient. At this stage, the quality of these models was still fairly confined due to the lack of data and computational power. The growth of the computational power and the emergence of GPU computing has led to a growing interest in the field of neural networks, which even brought about competitions in designing better architectures for image classification tasks [3]. Due to these better resources, researchers were able to go beyond the shallow networks and build deeper networks, which led to the area of 'deep learning'.

A milestone in the development of newer deeper networks was *LeNet-5* created by *LeCun et al.* in 1998 [56]. Their work is considered to be one of the first *convolutional neural networks* (CNN). Nowadays there are many different architectures in the field of artificial intelligence, e.g. *recurrent neural networks* (RNN), *generative adversarial network* (GAN), autoencoders, all designed for beating human performances. Still, there are many human achievements that cannot be beaten by these learned models, but some of them are outperformed, e.g. image recognition [48] and games [11]. For further reading on different neural network architectures we refer to [3, 40].

Basic architectures

The perceptron

Rosenblatt's perceptron is a linear classifier that calculates a weighted sum of the input, followed by an activation function. In Figure 2.2 the architecture of the perceptron is given. In this example we consider an input with 3 nodes $x = [x_1, x_2, x_3]$, for each node a weight w_i, i.e. $w = [w_1, w_2, w_3]$; and an added bias term b. These weights are multiplied with its input and a bias is added. Subsequently it is summed before entering the activation function

$$\sigma(x) = \text{sign}(x) = \begin{cases} 1, & x \geq 0 \\ -1, & x < 0. \end{cases}$$

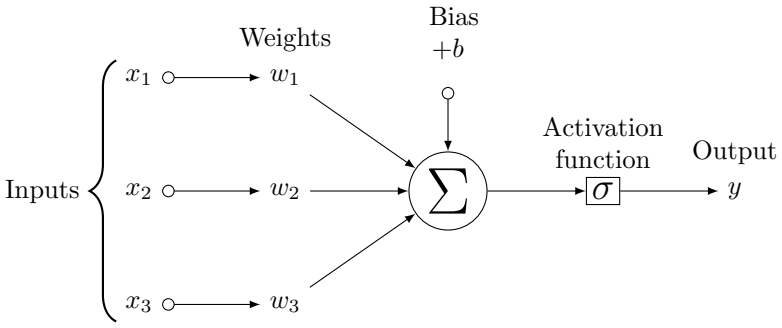


Figure 2.2: *Architecture of a perceptron with bias.*

This activation function is called the sign function. Therefore, a prediction $\hat{y}$ is defined as:

$$\hat{y} = \sigma\left(w \cdot x + b\right) = \sigma\left(\sum_{i=1}^{3} w_i x_i + b\right) = \begin{cases} 1, & \sum_{i=1}^{3} w_i x_i + b \geq 0 \\ -1, & \sum_{i=1}^{3} w_i x_i + b < 0. \end{cases} \tag{2.7}$$

Due to its binary output, one can see this perceptron as a linear classifier. The idea is to minimize the distance of misclassified points to a linear separating hyperplane. For example: if output $y_i = 1$ is misclassified, then $\sum_{i=1}^{3} w_i x_i + b < 0$ and vice versa. When trained, the weights and bias form the hyperplane that separates the two classes where vector w is perpendicular to the hyperplane. An example of the construction of the hyperplane is given in Figure 2.3. Later in this section it is explained how the parameters like weights and biases are trained.

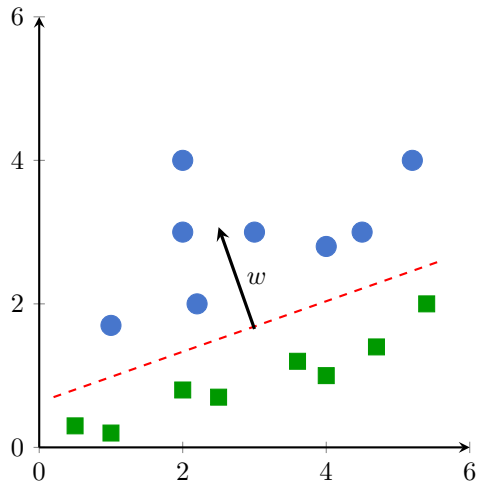


Figure 2.3: *Illustration of a linear separating hyperplane determined by a perceptron. The two different colored geometries represent the two classes to separate.*

The multilayer perceptron

It is discussed that a perceptron consists of an input layer and an output layer that is constructed by performing only one computation. By adding several computational layers before calculating the output, one designs a *multilayer perceptron* (MLP). An example of such a network is illustrated in Figure 2.4, in which two hidden layers are present. This type of network is referred to as a *feed-forward network* since every layer feeds the data towards the next layer until the output is determined. Besides that, this type of network is also called a *fully connected network* since every neuron in a specific layer is connected to all neurons in the previous and successive layer. These MLPs are characterized by its *hidden layers* and represent the layers between the input and output. They are called 'hidden' since the user of these networks only has the input and output to their disposal. In this example only one output is generated, but this is not generally the case.

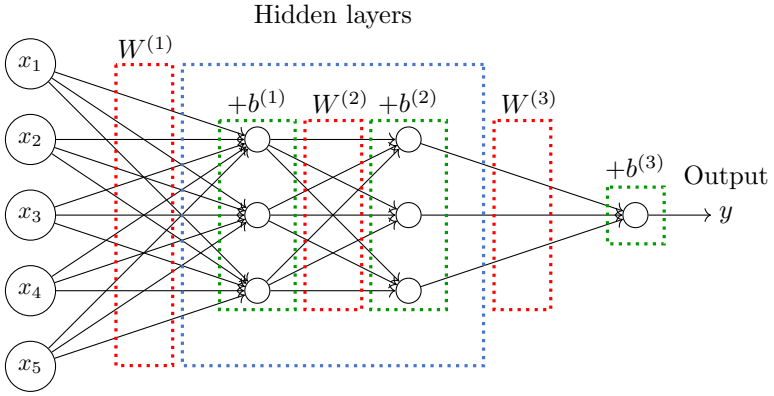


Figure 2.4: *Illustration of a fully connected multilayer perceptron with two hidden layers and one output.*

Values of each neuron can be calculated by using simple linear algebra. To demonstrate how this works, we use the network illustrated in Figure 2.4. Hidden layer 1 consists of three different neurons, i.e. $h^{(1)} = \begin{bmatrix} h_1^{(1)} \\ h_2^{(1)} \\ h_3^{(1)} \end{bmatrix}$, and each neuron in that layer is connected to all neurons from the previous layer. The value of the first neuron $h_1^{(1)}$ in layer $h^{(1)}$ is calculated by

$$\sigma\left(w_{1,1}^{(1)}x_1 + \ldots + w_{1,5}^{(1)}x_5 + b_1^{(1)}\right) \tag{2.8}$$

where $\sigma(\cdot)$ is the *nonlinear activation function* and $w_{ij}^{(1)}$ is the weight of the connection between the i-th neuron in layer $h^{(1)}$ and the j-th neuron in the input layer. This means that

$$h^{(1)} = \begin{bmatrix} h_1^{(1)} \\ h_2^{(1)} \\ h_3^{(1)} \end{bmatrix} = \sigma\left(\begin{bmatrix} w_{1,1}^{(1)} & w_{1,2}^{(1)} & w_{1,3}^{(1)} & w_{1,4}^{(1)} & w_{1,5}^{(1)} \\ w_{2,1}^{(1)} & w_{2,2}^{(1)} & w_{2,3}^{(1)} & w_{2,4}^{(1)} & w_{2,5}^{(1)} \\ w_{3,1}^{(1)} & w_{3,2}^{(1)} & w_{3,3}^{(1)} & w_{3,4}^{(1)} & w_{3,5}^{(1)} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} + \begin{bmatrix} b_1^{(1)} \\ b_2^{(1)} \\ b_3^{(1)} \end{bmatrix}\right), \tag{2.9}$$

$$\text{thus } h^{(1)} = \sigma\left(W^{(1)}x + b^{(1)}\right). \tag{2.10}$$

In a similar way it can be stated that

$$h^{(2)} = \sigma\left(W^{(2)}h^{(1)} + b^{(2)}\right) \tag{2.11}$$

$$\text{and } \hat{y} = \sigma\left(W^{(3)}h^{(2)} + b^{(3)}\right). \tag{2.12}$$

The adjustment of the weights and biases in this network is based on the error of the prediction $\hat{y}$. The training of the weights and biases is discussed later. The weights and biases form the parameters to be trained and are denoted by the set Φ. In order to give a generalization of the above example, we assume that a network has $L+1$ different layers, where $x = a^{(0)}$ and

$\hat{y} = a^{(L)}$. Each layer can be described as

$$a^{(l)} = \mathcal{M}^{(l)}_{\Phi(l)}\left(a^{(l-1)}\right) = \sigma\left(W^{(l)}a^{(l-1)} + b^{(l)}\right) \tag{2.13}$$

and the prediction as

$$\hat{y} = a^{(L)} = \mathcal{M}^{(L)}_{\Phi(L)}\left(a^{(L-1)}\right) \tag{2.14}$$

$$= \left(\mathcal{M}^{(L)}_{\Phi(L)} \circ \cdots \circ \mathcal{M}^{(1)}_{\Phi(1)}\right)(x), \tag{2.15}$$

where $\circ$ represents a composition.

The perceptron can at most linearly separate a data set, while a wider network with nonlinear activation function can do this in a nonlinear manner. It is strictly necessary to use a nonlinear activation function since linear activation functions result in an affine mapping. By increasing the width of the network one has more neurons (parameters) to construct a good prediction. More parameters lead to more ways to describe or to separate the data, which is beneficial if one wants an accurate prediction on a specific data set [3, 40].

2.2.2 Training a neural network

In order to form a meaningful prediction of the data, the network is trained and therefore the parameters are adjusted. How much a parameter is changed depends on the partial derivative with respect to that parameter. The calculation of the partial derivative is done via the so-called *backpropagation*. The actual adjustment of the weights is done via optimizers of which *stochastic gradient descent* is a popular method. Both backpropagation and optimizers are dependent on the prediction error and therefore on the *loss function*. Thus, below we will start with the explanation of the loss function before addressing the backpropagation and the optimizers. Afterwards, the activation function will be discussed and some practical issues that can occur while training a deep neural network.

The loss function

The goal of the neural network is to train the parameters Φ in such a manner that the predictions $\hat{y} \in \mathbb{R}^m$ are close to the ground truth $y \in \mathbb{R}^m$, in which the predictions are based on the input $x \in \mathbb{R}^n$. To determine how close the predictions are to the ground truth one needs to use a measure, which is called the *loss function*. A popular loss function is the so-called *mean squared error* (MSE), in which the error is the mean of the squared error of the N input samples. Then, the optimization of the parameters over all input samples, can be described as

$$\min_{\Phi} L(\Phi) = \min_{\Phi} \frac{1}{N} \sum_{j=1}^{N} \left(\hat{y}(x_j, \Phi)_j - y_j\right)^2. \tag{2.16}$$

The choice of loss function always depends on the problem at hand. For many application the MSE is used because it is differentiable and convex [94]. It is beyond the scope of this thesis to investigate different loss functions. For further reading on loss functions in general we refer to [3] and for loss functions used for image reconstruction tasks we refer to [94].

Backpropagation

The reason that the calculation of the partial derivatives is called backpropagation is the fact that the direction of computation is reversed compared to that of a feed-forward network. It basically feeds the error backwards through the network. The value of the partial derivatives with respect to the weights and biases depends on the used loss function. To illustrate this, we consider a fully connected network with $L+1$ layers of which $a^{(0)}$ is the input x and $\hat{y} = a^{(L)}$ is the output. Due to equations (2.13) and (2.14) the output of the neural network is formulated as

$$a^{(L)} = \sigma\left(z^{(L)}\right), \tag{2.17}$$

$$\text{where } z^{(L)} = W^{(L)}a^{(L-1)} + b^{(L)}, \tag{2.18}$$

and based on this prediction a loss is calculated. Since the prediction is calculated by a composition of prior layers, one can use the chain rule for differentiation for the calculation of the derivatives with respect to the weights and biases of a specific layer l, i.e. $\frac{\partial L}{\partial w^{(l)}}$ and $\frac{\partial L}{\partial b^{(l)}}$, respectively.

These derivatives give the sensitivity of the loss function when changing a certain weight or bias. If, for example, the partial derivative of the loss function with respect to $w_{1,2}$ and $w_{1,5}$ have the values 4.5 and 0.1, respectively. Then, it can be stated that the loss function is 45 times more sensitive to changes in $w_{1,2}$ rather than $w_{1,5}$. When all partial derivatives are calculated, one can optimize the network by adjusting the parameters. The optimization process is discussed in the next subsection. The adjusted neural network subsequently makes a new prediction by feeding the data through the adjusted network. Then, based on the new loss of the prediction, the partial derivatives are calculated and the different parameters are again optimized. This process repeats itself until it is converged or a certain amount of epochs are completed.

Optimizers

One can train the parameters by updating them according to a specific update rule. An easy way of updating is the usage of the gradient descent update rule:

$$\Phi := \Phi - \eta\frac{\partial L}{\partial \Phi} \tag{2.19}$$

where η is called the *learning rate* and Φ is the vector with all the trainable parameters, i.e. weights and biases. When a large data set is used for training the network, the calculation of the partial derivative in (2.19) is very slow. Furthermore, the optimization of the loss function is highly nonconvex with many local minima, which makes it very likely that the loss function ends up in a local minimum due to the decline of the gradient when it approaches the minimum.

In order to improve the speed of the derivation of the partial derivative, one can use only one randomly chosen input sample for the prediction $\hat{y}$. Based on this prediction the partial derivative is calculated. Due to the randomness the name of this method is *stochastic gradient descent*. By taking only one training sample instead of all N training samples, one approximates the derivative defined in the gradient descent update rule. Due to this approximation it is possible that the optimization of the loss function does not converge to a local minimum. A method that is more often used is the *mini batch stochastic gradient descent*, which gives better approximations of the derivative since it is based on a batch of k samples rather than one. These stochastic methods enormously speed up the process when large data sets are considered, albeit less accurate [3]. Many other update rules are extensions of the stochastic gradient descent and are therefore not discussed here. For further reading on different optimizers we refer to [78].

Practical issues in training

Exploding and vanishing gradients

So far, it is not discussed what type of activation function is used. The choice of activation function can be of great importance for the overall performance of the network. If one calculates the derivative with respect to the weights in one of the layers close to the input, then the derivative of the activation function is taken into account many times. This is due to the fact that the output is a composition of prior layers. When the derivative of the activation function is always between zero and one then the gradient decreases exponentially. This can lead to extremely small gradients of those weights. A small gradient means that the corresponding weight does not contribute much to the loss function and is therefore hardly updated, leading to bad predictions of the neural network. This anomaly is called *vanishing gradients*. On the contrary, if the derivative of the activation function is always bigger than one then the gradients will grow exponentially. This blow up of the gradient is called *exploding gradients* [3].

A solution to this problem is to only consider activation functions with gradients equal to one for positive values. Often used activation functions are the ReLU and the leaky ReLU. These can be formulated as

$$\text{ReLU: } \sigma(x) = \begin{cases} 0, & x \leq 0 \\ x, & x > 0 \end{cases} \tag{2.20}$$

$$\text{leaky ReLU: } \sigma(x) = \begin{cases} \alpha \cdot x, & x \leq 0 \\ x, & x > 0 \end{cases} \tag{2.21}$$

where $\alpha \in (0, 1)$.

Overfitting

It is stated that a neural network is trained on a specific data set, often called the *training data set*. If trained properly, the neural network is able to make good predictions on this specific data set. *Overfitting* means that it is not able to make good reconstruction on an unseen test data set, albeit accurate on the training data set. This means that the trained network is not able to *generalize* well on unseen data sets. This problem often occurs when the network is too complex and therefore has too many parameters to fit the training data. Reducing the complexity or increasing the amount of training data helps to improve the predictive power for unseen data sets. It depends on the nature of the data how complex the network needs to be in order to capture all the features of the data. A good criterion for preferable amount of training data is to use an amount that is at least two to three times the size of the parameters [3]. *Pretraining* [40] can also improve the generalization of the network. This method improves the initial guess of the weights by training a shallow network on the training data set.

2.2.3 Convolutional neural network

Convolutional neural networks (CNNs) are biological inspired feedforward networks based on the research on the visual cortex of a cat by Hubel and Wiesel [49]. Their work describes the different layers that contribute to the vision of a cat, in which each layer picks up certain information. The first neural network that mimics pattern recognition is the *neocognitron* [37], which was the building block for the famous CNN architect LeNet-5 by *LeCun et al.* [56]. The different layers in a CNN network mimic the layers present in the visual cortex of human beings. By training the network correctly, one can pick up the spatial structure of the input. In many cases the input of a CNN is an image, either colored or gray-scale. This type of network has proven to be better than human beings in image recognition tasks [48]. In this section we discuss the main building blocks of the CNN, of which the convolutional layer is the most important.

Convolutional layer

CNNs are build for discovering the spatial features on grid-structured inputs, almost always images. All layers in the network are considered to be three dimensional tensors, in which the l-th layer has a width of $W^{(l)}$, a height of $H^{(l)}$ and a depth of $D^{(l)}$. If the input layer is a colored image, then its depth is equal to 3 since a colored image has three color channels: red, green and blue (RGB). Therefore, the depth is in many cases referred to as *channels*. The CNN extracts the spatial features and dependencies of local areas in the input layer by applying a weighted sum between the pixels in that neighbourhood and a *kernel*. It is important that the kernel has the same depth as its input and that the size is kept constant in a specific layer. Furthermore, the kernel is in general square and of odd length, which is very convenient since those kernels have a clear center. Afterwards one bias per channel is added and finally an activation function is used in order to formulate the successive layer. This successive layer is often called the *feature map* since it captures the different spatial features of the input layer. By sliding the kernel along the width and height of the input layer, one can extract the local spatial features of the entire layer. The calculation of the weighed sum between the pixels and the sliding kernel is called a *convolution*, thus the convolutional layer. In Figure 2.5 it is illustrated how the feature map is constructed by using two different $3 \times 3 \times 3$ kernels. It is clear that every pixel in the feature map is a convolution of the kernel with a local 3×3 neighbourhood in the previous layer. In addition, each applied kernel produces a feature map and each one of them are stacked together to form the entire feature map. Furthermore, it is also illustrated that the size of the feature map changes when applying the convolution with the kernel. If the input size in layer l is $H^{(l)} \times W^{(l)} \times D^{(l)}$ and $N_k^{(l)}$ kernels of size $K_W^{(l)} \times K_H^{(l)} \times D^{(l)}$ are used, then the feature map has the dimension $\left(H^{(l)} - K_H^{(l)} + 1\right) \times \left(W^{(l)} - K_W^{(l)} + 1\right) \times N_k^{(l)}$. By adding *zero padding*, i.e. zeros on the boundaries along the width and height of the input, one can ensure that the feature map has the same dimension. This leads to a larger $H^{(l)}$ and $W^{(l)}$ and therefore to a larger feature map.

So far it is explained that a successive layer in a CNN is constructed by first applying convolutions with a specific amount of kernels, then adding a bias and finally applying an activation function. In order to generalize this mathematically it is assumed that the input layer $l-1$ is $a^{(l-1)}$ and is of size $H^{(l-1)} \times W^{(l-1)} \times D^{(l-1)}$. The output layer is $a^{(l)}$ and $N_K^{(l-1)}$ kernels are used of size $K_H^{(l-1)} \times K_W^{(l-1)} \times D^{(l-1)}$. Then the expression for $a^{(l)}$ is equal to:

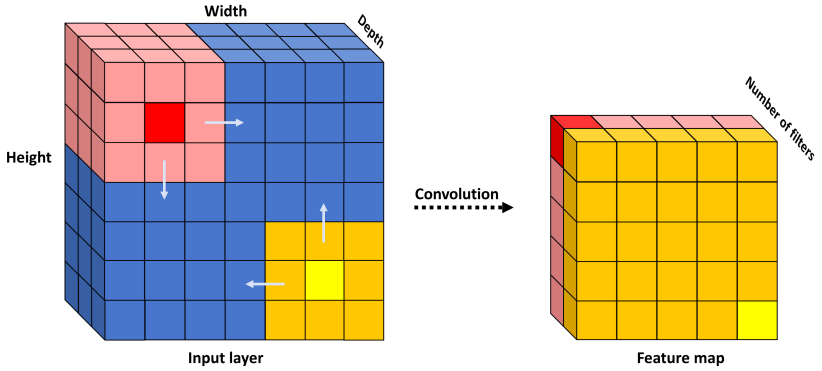


Figure 2.5: *Illustration of a convolutional operation with two* $3 \times 3 \times 3$ *kernels, which lead to two different feature maps. In this example a stride of 1 is used.*

$$a^{(l)}(u,v,z) = \sigma\left(b^{(l-1)} + \sum_{m=1}^{K_W^{(l-1)}} \sum_{n=1}^{K_H^{(l-1)}} \sum_{d=1}^{N_K^{(l-1)}} a^{(l-1)}\left(m+u-1, n+v-1, d\right) \cdot K^{(l-1)}\left(m,n,d\right)\right)$$
$$\forall u \in \{1,2,\ldots,W^{(l-1)}\} \tag{2.22}$$
$$\forall v \in \{1,2,\ldots,H^{(l-1)}\}$$

where both $a^{(l-1)}$ and $a^{(l)}$ are 3D tensors, K is the kernel, $b^{(l-1)}$ is the bias and the activation function is denoted by $\sigma(\cdot)$. In this thesis it is assumed that the step size of the kernel in both directions is equal to 1. This step size is called *stride*.

The number of trainable parameters is greatly reduced by using a CNN. This is because each pixel in an layer is only connected to a local region in the previous layer. The amount of parameters is further decreased by using *shared weights*. This means that a kernel uses the same weights for all pixels in that specific layer instead of learning weights for each entry separately [40]. Therefore, a CNN is much less demanding in terms of its computational burden and memory.

Pooling layer

Just like convolutions, *pooling layers* are one of the building blocks of CNNs. A pooling layer treats each channel separately and considers a region $p \times p$ at the time. If *max-pooling* is used then the maximum value of that $p \times p$ window is passed through to the next layer. When pooling is applied to layer l together with a stride s, then the dimension of the output layer is:

$$\Big((H^{(l)} - p)/s + 1\Big) \times \Big((W^{(l)} - p)/s + 1\Big) \times D^{(l)}. \tag{2.23}$$

Pooling is frequently used in combination with a stride 2 and a 2×2 window in order to reduce the spatial footprint of the input, which is illustrated in Figure 2.6. This reduces the size of a layer by a factor 2 and introduces a non-linearity into the network. An alternative for max-pooling is the average pooling, but is in general less used [3].

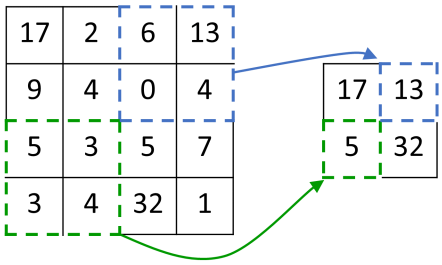


Figure 2.6: *Example of max-pooling performed on a* 4×4 *grid with a* 2×2 *window and a stride 2.*

Due to this reduction in dimension one can study the larger components present in the input. Besides that, downscaling of the dimensions reduces the computational burden of the successive layers. Furthermore, this operation does not have any trainable parameters and it increases the so-called *receptive field*, which is explained in the next subsection.

Receptive field

In the previous subsection it is explained that one pixel in the feature map is determined by a convolution between a kernel and a local region in the previous layer. If the network is larger than two layers, one can still determine which pixels in the input layer contribute to the final output layer. The region that contributes to a certain pixel in the output is called *receptive field*. To illustrate this concept we consider one input layer and two feature map, of which the second feature map is considered to be the output. The different layers are illustrated in Figure 2.7 where the colors blue, green and yellow resemble the different layers. In this example a convolution with a 3×3 kernel and no zero padding is used. This figure illustrates the receptive field of the red pixel in the output layer. Since the kernel size is 3×3 the red pixel is determined by a 3×3 region in feature map 1. Every pixel in that layer again depends on a 3×3 region in the input layer. Due to the overlap of the convolutions one can conclude that the receptive field in the input layer is a 5×5 region.

If the amount of convolutional layers gets larger then the receptive field will grow linearly. The receptive field in layer l is then defined as:

$$r^{(l)} = r^{(l-1)} + ((k^{(l)} - 1) \cdot \prod_{i}^{l-1} s_i) \tag{2.24}$$

where s_l is the stride in layer l and $k^{(l)}$ is the kernel size in layer l. The receptive field will grow faster if pooling is inserted in the network. In that case the receptive field will grow exponentially. Not all pixels in the receptive field contribute equally to an output pixel. This concept of unequal contribution is demonstrated by *Luo et al.* [61] and is called the *effective receptive field*. Since the convolutions are sliding along the dimensions of the height and width, some pixels will contribute more to the next feature map. This means that pixels closer to the center of the receptive field contribute more than pixels on the boundary. These contributions scale exponentially.

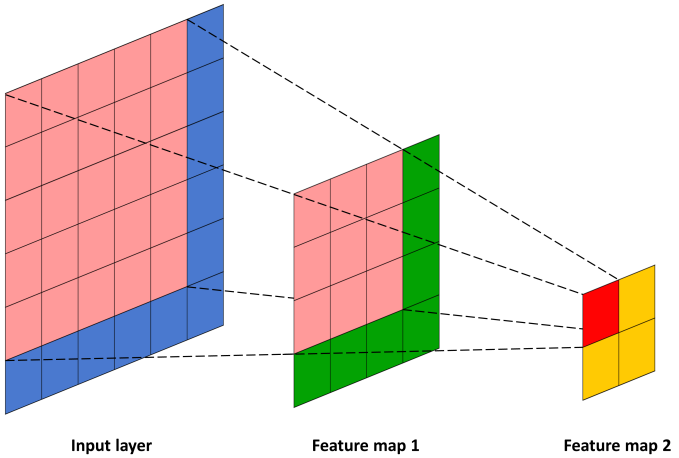


Figure 2.7: *The corresponding feature map of a pixel in the output layer.*

Skip connections

Another building block of a CNN is the *skip connection*, which adds the input of a certain layer to the output of a layer further into the network. This means that the output of the network is connected to the input by different pathways of different lengths, in which shorter pathways have higher learning capabilities [3]. When having more pathways to the output, the optimizer has the ability to choose the relevant pathway for a certain input. An example of the use of a skip connection is given in Figure 2.8.

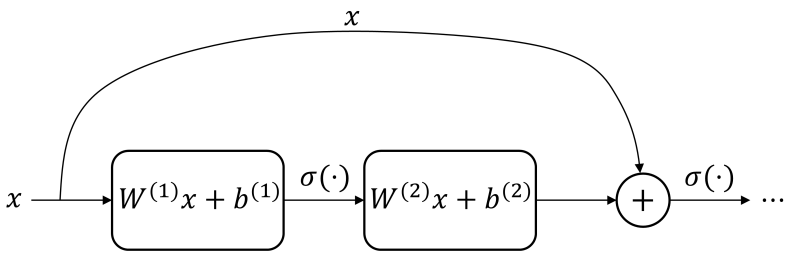


Figure 2.8: *Illustration of a skip connection.*

2.3 Compressed sensing

Below the field of compressed sensing is shortly introduced, along with its main aim and its implication for solving inverse problems with the help of machine learning.

2.3.1 Origin of compressed sensing

The mathematical description of the recovery of signals goes back to the middle of the 19^{th} century where C. Shannon published his work on the recoverability of a continuous signal by a discrete sequence of samples. His finding is summarized in the famous Nyquist-Shannon sampling theorem:

Theorem 1 (Nyquist-Shannon sampling theorem [83])**.** *If a function $x(t)$ contains no frequencies higher than B hertz, it is completely determined by giving its ordinates at a series of points spaced $1/(2B)$ seconds apart.*

For the discrete case it holds that the amount of measurements should at least be as large as the length of the signal [36, 70]. However, for many applications this leads to a large measuring time and huge data storage. The theory of *compressed sensing* is build to analyze the reconstruction quality of recovered signals when less measurements are taken. The theory is based on the idea that many high-dimensional signals have a sparse representation/basis and can be recovered even though a few measurements are taken [10, 36, 70]. It was introduced by Donoho [27] and by Tao & Candès [19]. They showed that under certain conditions a sparse input $x \in \mathbb{R}^n$ can be fully recovered from noisy measurements $y = Ax + \eta$, where $A \in \mathbb{R}^{m \times n}$, $m \leq n$ and $\eta \in \mathbb{R}^m$. These conditions are based on the properties of the forward mapping A and will be verified later. Due to its success it is widely used in fields like imaging, optics, statistical learning and machine learning. An example is the efficient compression technique JPEG, where its success is due to the fact that natural images have a sparse basis [35].

Recovering sparse signals from measurements can be summarized as the optimization problem

$$\min_x \|x\|_0 \text{ subject to } y = Ax \tag{P_0}$$

where it is clear from the definition of the ℓ_0-norm that this problem promotes sparsity. The issue is that solving equation (P_0) is an NP-hard problem [10, 19], where NP stands for "nondeterministic polynomial time". It roughly means that the amount of operations needed in order to solve it, can not by bound by a polynomial. Therefore, there is no polynomial time algorithm that solves this problem.

Researchers searched for another convex objective function that approximates the optimization problem (P_0). The ℓ_p-norms are capable of adequate approximations of the defined problem, where the ℓ_p-norm is defined as

$$\|x\|_p = \begin{cases} \left(\sum_{i=1}^{n} |x_i|^p \right)^{\frac{1}{p}} & \text{for } p \in [1, \infty) \\ \max_{i=1,\dots,n} |x_i| & \text{for } p = \infty. \end{cases} \tag{2.25}$$

Chen et al. [23] proved that the ℓ_1-norm is the closest norm to ℓ_0 and therefore optimization problem (P_0) was replaced by

$$\min_x \|x\|_1 \text{ subject to } y = Ax \tag{P_1}$$

and goes under the name *basic pursuit*. The only difference between optimization problems (P_0) and (P_1) is the use of norm. The goal is to recover sparse signals so the question is whether the ℓ_1-norm promotes sparsity. In Figure 2.9 it is illustrated that the subspace of solutions $y = Ax$ intersects the ℓ_1-ball in one of the coordinate axis, which is not the case for the ℓ_2-ball. This means that one of the coordinates of the solution is zero and therefore it promotes sparsity. Then there is one remaining question, which is the core of compressed sensing [55]: when do the solutions of (P_0) and (P_1) coincide? In other words: when is '$\ell_0 = \ell_1$'? The answer to this question will be discussed later.

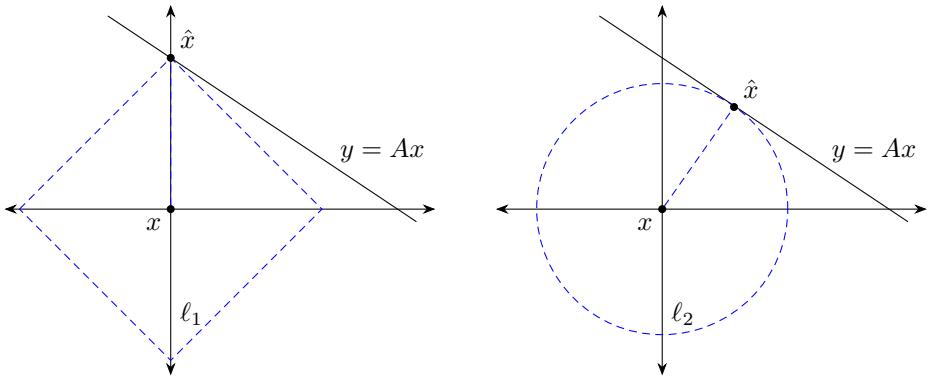


Figure 2.9: *Solution of optimization problem with two different norms: ℓ_1- and ℓ_2-norm.*

When measurements are corrupted with noise, the minimization problem (P_1) can be modified to

$$\min_x \|x\|_1 \text{ subject to } \|Ax - y\|_2^2 \leq \epsilon \tag{2.26}$$

and for a specific regularization parameter $\lambda > 0$ the solution to (2.26) is equal to the unconstrained optimization problem

$$\min_x \frac{1}{2}\|Ax - y\|_2^2 + \lambda\|x\|_1 . \tag{2.27}$$

Popular methods for solving (2.27) are *interior-point methods* and *iterative thresholding* methods [55]. These convex optimization methods tend to give the best accuracy but with a high computational cost [70]. On the other hand, greedy algorithms likes *Orthogonal Matching Pursuit* do not consider the unconstrained optimization problem (2.27) but iteratively approximate the coefficients and the basis of the input signal such that (P_1) holds. Theoretically they are as accurate as the convex optimization methods but are faster and easier to implement [55, 70].

2.3.2 '$\ell_0 = \ell_1$'

Candès and Tao [20] developed a condition on the forward mapping A that is sufficient for (P_1) to coincide with (P_0). This condition is called the *Restricted Isometry Property* (RIP) and is summarized in Theorem 2.

Theorem 2 (RIP [19]). *The forward mapping A is said to satisfy the restricted isometry property of order $s \in \mathbb{N}$ with parameter $\delta_s \in (0,1)$ if*

$$(1-\delta_s)\|x\|_{\ell_2}^2 \leq \|Ax\|_{\ell_2}^2 \leq (1+\delta_s)\|x\|_{\ell_2}^2 . \tag{2.28}$$

Lemma 1 (Restricted isometry constant (RIC) [86]). *Assume that the forward mapping $A \in \mathbb{R}^{m\times n}$ satisfies RIP of order $s \in \mathbb{N}$ with parameter $\delta_s \in (0,1)$, then the s-th restricted isometry constant δ_s is given by*

$$\delta_s = \max_{S\subset\{1,\dots,n\},|S|\leq s} \left\| A_S^T A_S - I \right\|_2 . \tag{2.29}$$

If the RIP is satisfied up to order s then all submatrices of A containing s-columns are well conditioned and therefore stable recovery of the signal x is ensured [5]. The smaller the restricted isometry constant δ_s is, the better the reconstruction capabilities are and the closer the mapping is to an isometry, i.e. a distance preserving mapping. RIP also implies that any set of s columns of A is nearly orthonormal [2] and that the Gram matrix $A_S^T A_S$ is positive definite [25] with all its eigenvalues laying in the interval $[1-\delta_s, 1+\delta_s]$ [45]. These eigenvalues lie in this interval if the diagonal elements of the Gram matrix $A_S^T A_S$ are close to one and if the off-diagonals are very small. To show this, the concept of *Geršgorin circles* is used.

Theorem 3 (Geršgorin circle theorem [85]). *The eigenvalues of an $m \times m$ matrix M all lie in the union of m discs $d_i = d_i(c_i, r_i)$, $i = 1, 2, \dots, m$, centered at $c_i = M_{ii}$ and with radius*

$$r_i = \sum_{\substack{j=1 \\ j\neq i}}^{n} |M_{ij}| .$$

Now consider a subset of column indices $S \subset \{1,\dots,n\}$ where $|S| \leq s$ and the Gram matrix $A_S^T A_S$. Suppose that for $s \geq 1$ there are positive values δ_d and δ_O such that $\delta_d + \delta_O = \delta_s \in (0,1)$ and that holds:

$$|G_{ii} - 1| < \delta_d \qquad \forall i \tag{2.30}$$

$$|G_{ij}| < \delta_O / s \qquad \forall i \neq j. \tag{2.31}$$

Then the Geršgorin circle theorem states that the center of the disc does not deviate more than δ_d from 1 and the radius of each disc is not bigger than $(s-1)\delta_O/s < \delta_O$. This means that all the eigenvalues are in the range $(1-\delta_d-\delta_O, 1+\delta_d+\delta_O) = (1-\delta_s, 1+\delta_s)$.

The off-diagonal elements of the Gram matrix are closely related to another important property of the forward mapping A: *mutual coherence*.

Definition 2 (Mutual coherence [33]). *Let $A = (a_i)_{i=1}^n$ be an $m \times n$ matrix. Then its mutual coherence $\mu(A)$ is defined as*

$$\mu(A) = \max_{i\neq j} \frac{|\langle a_i, a_j \rangle|}{\|a_i\|_2 \|a_j\|_2}. \tag{2.32}$$

This property characterizes the dependency between the columns of the forward mapping and is summarized in Definition 3.

Definition 3 (Spark [28]). *Given forward mapping $A \in \mathbb{R}^{m\times n}$, then spark(A) is the minimal number s_p such that there is a set of s_p columns of A that is linearly dependent. If A has normalized columns then*

$$spark(A) \geq 1 + \frac{1}{\mu(A)}. \tag{2.33}$$

Matrices with highly correlated columns are generally ill-conditioned [51]. The maximal mutual coherence for a matrix $A \in \mathbb{R}^{m\times n}$ is 1, which is the case when A contains mutual linear dependent columns. The minimal mutual coherence can only be zero when $m = n$ and when the columns of A form an orthogonal basis. If $m < n$ then the minimal mutual coherence can not be equal to zero since this would mean that n m-dimensional columns are orthogonal. Therefore, the lower bound of the mutual coherence of underdetermined systems is unequal to zero. This lower bound is the so-called *Welch bound* [29, 33].

Lemma 2 (Welch bound [55]). *Let A be an $m \times n$ matrix. Then we have that*

$$\mu(A) \in \left[\sqrt{\frac{n-m}{m(n-1)}}, 1\right] \tag{2.34}$$

The relationship between the off-diagonal elements of the Gram matrix and the mutual coherence can be shown easily when A has orthonormal columns, i.e. $\|a_i\|_2 = 1\ \forall i = 1, \ldots, n$. Then $\mu(A)$ is equal to the biggest off-diagonal of $G = A^T A$ in absolute sense. This is shown in Lemma 3.

Lemma 3. *Let $A \in \mathbb{R}^{m\times n}$ with ℓ_2-normalized columns then $\mu(A)$ is equal to the largest off-diagonal of the Gram matrix $G = A^T A$ in absolute value. Furthermore, the eigenvalues of the Gram matrix $G = A_S^T A_S$ lie in the interval $\big[1 - \mu(s-1), 1 + \mu(s-1)\big]$.*

Proof. Let $A = (a_i)_{i=1}^n$ be an $m \times n$ with ℓ_2-normalized columns, then the corresponding Gram matrix is defined as:

$$G = A^T A \tag{2.35}$$

$$= \begin{bmatrix} 1 & \langle a_1, a_2\rangle & \ldots & \langle a_1, a_n\rangle \\ \langle a_2, a_1\rangle & 1 & \ldots & \langle a_2, a_n\rangle \\ \vdots & \vdots & \ddots & \vdots \\ \langle a_n, a_1\rangle & \langle a_n, a_2\rangle & \ldots & 1 \end{bmatrix} \in \mathbb{R}^{n\times n}. \tag{2.36}$$

Due to the ℓ_2-normalized columns of A we know

$$\mu(A) = \max_{i\neq j} \big|\langle a_i, a_j\rangle\big| \tag{2.37}$$

we can conclude that the biggest off-diagonal element of G in absolute sense is equal to the mutual coherence of A.

Now, if one considers $S \subset \{1, \ldots, n\}$ with $|S| = s$ and $A_S \in \mathbb{R}^{m\times s}$, then the corresponding Gram matrix $G = A_S^T A_S \in \mathbb{R}^{s\times s}$. Due to the Geršgorin circle theorem we know that the eigenvalues of the Gram matrix lie in the interval

$$\left[1 - \max_j R_j, 1 + \max_j R_j\right]$$

where R_j is the radius in column $1 \leq j \leq s$. The radius is defined as

$$\begin{aligned} R_j &= \sum_{1 \leq k \leq s,\ k \neq j} \left| \langle a_k, a_j \rangle \right| \\ &\leq (s-1) \cdot \mu. \end{aligned}$$

□

Due to Lemma 3 we know that $\delta_s \leq \mu(s-1)$ and this shows the connection between mutual coherence and RIP. Therefore, for a sparse solution to meet the condition '$\ell_0 = \ell_1$', is directly related to the mutual coherence of the forward mapping A. This means that we can formulate a sufficient condition on the solution x in order to meet '$\ell_0 = \ell_1$'. This sufficient condition is summarized in the following theorem.

Theorem 4 (from [55]). *Given forward mapping $A \in \mathbb{R}^{m \times n}$ and solution $x \in \mathbb{R}^n \setminus \{0\}$ of (P_0). If*

$$\|x\|_0 < \frac{1}{2}\left(1 + \frac{1}{\mu(A)}\right)$$

then x is a unique solution of (P_0) and (P_1).

This shows that the mutual coherence of A has a direct effect on the degree of sparsity of the unique solution. The mutual coherence is of great importance since it is very hard for general matrices to check whether RIP is satisfied because of the high computational complexity [10, 64]. In these cases it is much easier to check the mutual coherence of the matrix. This property can give good insight in whether good sparse recovery is guaranteed, albeit weaker than RIP [69]. From theorem 4 it is clear that it is desirable to have a small mutual coherence [21].

2.3.3 Learned approach to compressed sensing

The standard iterative reconstruction algorithms used in compressed sensing, tend to give unwanted results when applied to medical imaging. The algorithms have the propensity to remove detailed structures in the reconstruction phase. These details can be of extreme importance for physicians in order to make a good diagnosis. Furthermore, these algorithms are time consuming and are very sensitive to changes of the regularization parameter. Combining compressed sensing with machine learning could fill this void [50]. *Calderbank et al.* [15] where among the first researchers to combine these fields. Their proposed method is called *compressed learning*. In their work they showed that a support vector machine derived from compressed data and an RIP fulfilling forward mapping, has the same accuracy as the best linear classifier that is based on the full data. Recently many researchers analyzed the effect of compressed sensing properties on the reconstruction quality of a neural network. *Xuan et al.* [91] showed that a lower mutual coherence has positive effects on the reconstruction quality and their approach demonstrated its robustness to noise. In their approach they added the mutual coherence as a penalty to the loss function of the deep neural network (DNN). Besides that, in [50] it was proposed that when adding data in the frequency domain k, the neural network was better in learning the inversion due the fact that this addition made the forward mapping satisfying RIP. Including the extra line in k-space is the same as adding supplementary information to the neural network by attaching additional nodes to the neural network [13]. Their explanation is a consequence of Cover's theorem [26]. It states that if training data is not linearly separable in a low-dimensional space, it is more likely to be separable when a higher dimensional space is

considered [13, 46]. Figure 2.10 illustrates this principle.
These examples show that certain compressed sensing properties can be beneficial for the reconstruction quality of the neural network. Later in this thesis it will be clear which property is used and how we used it.

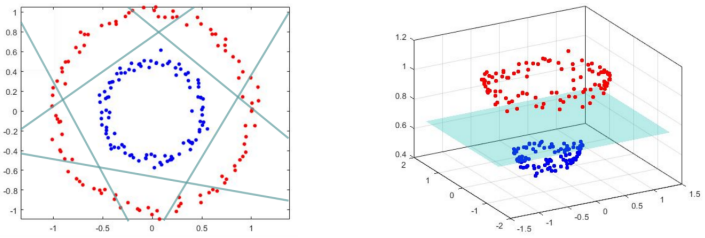


Figure 2.10: *The figure states that in 2D five linear hyperplanes are we needed to separate the data. In 3D only one linear hyperplane is needed. See [13] for more details.*

2.4 Preconditioning

This section contains background information about preconditioning. First it is explained what the traditional aim is and afterwards its connection with regularized inverse problems is given. Finally, we describe two important methods that can be used for the construction of preconditioners.

2.4.1 Origin of preconditioning

Popular methods for solving problem $y = Ax$ are direct methods and iterative methods. Direct methods, based on factorizations of the forward mapping A into easy invertible matrices, are widely used because of their robustness and predictability in terms of storage and time [9]. If the forward mapping is not too large a direct solver can be used to efficiently solve the inverse problem in an appropriate amount of time. Nowadays, many problems are multidimensional or either too large (e.g. most PDE related problems [88]), causing the direct solver to solve billions or more equations. Since direct solvers scale poorly with the problem size, iterative solvers are required for solving these large-scale problems. There are situations where even iterative methods fail to converge. Then, preconditioners can be used in order to (possibly) solve the inverse problem in a reasonable amount of time [9, 80]. A good preconditioner transforms system $y = Ax$ into another system that possesses better properties for solving it iteratively [9]. The origin of the term *preconditioning* is associated with the observation that lowering the *condition number* κ guarantees faster convergence of the solver [88]. To illustrate this, we assume that the conjugate gradient method is used for solving $y = Ax$. The convergence bounds can be summarized as follows.

Theorem 5 (Convergence conjugate gradient method [75])**.** *Let $A \in \mathbb{R}^{n\times n}$ be a symmetric and positive definite matrix. The conjugate gradient method for solving $y = Ax$ converges after at most n steps. Moreover the error $\bar{x} - x^{(k)}$ ath the k-th iteration ($k < n$) is orthogonal to the search direction $p^{(j)}$ for $j = 0, \ldots, k-1$ and*

$$\frac{\left\|\bar{x} - x^{(k)}\right\|_A}{\left\|\bar{x} - x^{(0)}\right\|_A} \leq \frac{2c^k}{1 + c^{2k}}, \quad \text{with } c = \frac{\sqrt{\kappa(A)} - 1}{\sqrt{\kappa(A)} + 1}. \tag{2.38}$$

This means that a higher condition number κ gives a slow convergence, while a condition number close to one gives a fast convergence. When an Euclidean 2-norm is used, the condition number $\kappa(A)$ is equal to

$$\kappa(A) = \frac{\lambda_{max}(A)}{\lambda_{min}(A)}, \tag{2.39}$$

which means that iterative algorithms converge fast when the eigenvalues are close to one. Bringing these eigenvalues close to one is exactly the goal of using preconditioners [9, 88].

There are three different ways of using preconditioners, i.e. left, right and split [9, 75]. If a nonsingular *left preconditioner* P is used then then the linear system

$$P^{-1}Ax = P^{-1}b \tag{2.40}$$

has the same solution as system $y = Ax$. In case when P approximates A then this system might be less challenging to solve. One can also use a *right preconditioner*:

$$AP^{-1}y = b, \quad x = P^{-1}y. \tag{2.41}$$

Additionally, a *split preconditioner* can also be used, i.e,

$$P_1^{-1}AP_2^{-1}y = P_1^{-1}b, \quad x = P_2^{-1}y, \tag{2.42}$$

where the preconditioner is $P = P_1P_2$. The use of these preconditioners either depend on the PDEs that describe the physics of the problem or on the forward mapping A [9, 75]. PDE based preconditioning is often not successful since it might be necessary to have complete knowledge of the problem or it might be that the knowledge is too hard to get and to use [9]. Consequently, preconditioners have been developed that can be applied universally and are purely based on the forward mapping A. Since in our inverse problem the forward mapping is given, we only consider this type of preconditioner.

2.4.2 Preconditioning and regularized inverse problems

Due to the ill-posedness of many discretized inverse problem, a certain amount of regularization is needed in order to stabilize the process of calculating the solution [17]. In section 2.1 it is explained that Tikhonov regularization is a popular regularization method. In order to verify the connection between solving preconditioned systems and Tikhonov regularization, an explanation via the *Bayesian view* on inverse problems is very convenient. From this perspective, all the variables of inverse problem (IP) are regarded as random. Thus, the stochastic version of (IP) is formulated as

$$Y = AX + E \tag{2.43}$$

where $Y,\ E \in \mathbb{R}^m$, $X \in \mathbb{R}^n$ and it is assumed that $X \sim \mathcal{N}(x_0, C_0)$ and $E \sim \mathcal{N}(e_0, C_1)$. Since all variables are considered to be stochastic, the solution to this problem is in fact a probability density function. The probability density of X without knowing Y is called the *prior density* and is denoted by $\pi_{\text{prior}}(x)$. The *likelihood density* is the probability density of Y given that $X = x$ and stands for $\pi_{\text{likelihood}}(y|x)$. The solution to an inverse problem is always dependent on the measurement and therefore $\pi_{\text{posterior}}(x|y)$ is the probability density that belongs to the solution. This probability density is called the *posterior* [16]. Using *Bayes' Theorem*, the solution to the inverse problem (2.43) is denoted by

$$\begin{aligned}\pi_{\text{posterior}}(x|y) &= \frac{\pi_{\text{prior}}(x)\pi_{\text{likelihood}}(y|x)}{\pi(y)} \\ &\propto \pi_{\text{prior}}(x)\pi(y|x).\end{aligned} \tag{2.44}$$

Due to the assumption that X is normally distributed and the neglection of the normalization, it can be stated that that the prior density is proportional to

$$\pi_{\text{prior}}(x) \propto \exp\left(-\frac{1}{2}(x - x_0)^T C_0^{-1}(x - x_0)\right). \tag{2.45}$$

A similar expression can also be found for $\pi(y|x)$, in which x is fixed and therefore the only randomness in y is due to e. This means that

$$\begin{aligned}\pi(y|x) &= \pi_{\text{noise}}(e) \\ &\propto \exp\left(-\frac{1}{2}(e-e_0)^T C_1^{-1}(e-e_0)\right) && e = y - Ax \\ &= \exp\left(-\frac{1}{2}(y-Ax-e_0)^T C_1^{-1}(y-Ax-e_0)\right), && (2.46)\end{aligned}$$

and due to (2.44) this implies that

$$\pi(x|y) \propto \exp\left(-\frac{1}{2}(y-Ax-e_0)^T C_1^{-1}(y-Ax-e_0) - \frac{1}{2}(x-x_0)^T C_0^{-1}(x-x_0)\right). \quad (2.47)$$

Since the covariance matrices C_0 and C_1 are symmetric and positive semi-definite, their Cholesky decomposition can be used in order to find an expression for their inverse. Due to the Cholesky decomposition it is known that $C_0 = LL^T$ whereby L is a lower triangular matrix. This means that $C_0^{-1} = (L^{-1})^T L^{-1} = P_2^T P_2$. In a similar fashion we can state that $C_1^{-1} = P_1^T P_1$. This leads to a new expression for (2.47):

$$\begin{aligned}\pi(x|y) &\propto \exp\left(-\frac{1}{2}(y-Ax-e_0)^T P_1^T P_1(y-Ax-e_0) - \frac{1}{2}(x-x_0)^T P_2^T P_2(x-x_0)\right) \\ &= \exp\left(-\frac{1}{2}\left\|P_1(y-Ax-e_0)\right\|^2 - \frac{1}{2}\left\|P_2(x-x_0)\right\|^2\right). && (2.48)\end{aligned}$$

Since all variables are stochastic, the desired solution to the inverse problem (2.43) is the *most likely* solution of the conditional probability density $\pi(x|y)$. This solution, if exists, is equal to the *maximum a posteriori* (MAP) estimate:

$$\begin{aligned}x_{\text{map}} &= \arg\max_x \pi(x|y) \\ &= \arg\min_x \left\|P_1(y-Ax-e_0)\right\|^2 + \left\|P_2(x-x_0)\right\|^2. && (2.49)\end{aligned}$$

Equation (2.49) shows the direct relationship between the MAP estimator and Tikhonov regularization in case when $e_0 = 0$, $P_1 = I$ and $P_2 = L$. The solution to the Tikhonov regularized inverse problem of $y = Ax + e$ is indeed the minimizer of the functional

$$\mathcal{F}_\alpha(x) = \|Ax-b\|^2 + \alpha\left\|L(x-x_0)\right\|^2.$$

This functional seeks for a solution x that fits the data but the additional second term prohibits disproportionate gain of the solution. The connection between this problem and a preconditioned system that is solved iteratively, is not obvious at this stage. In order to demonstrate that, another random variable $W = P_2(X - x_0)$ is introduced. Due to the fact that $X \sim \mathcal{N}(x_0, C_0)$, it is clear that

$$\begin{aligned}\mathbb{E}(W) &= 0 && \text{and} \\ \mathbb{E}(WW^T) &= \mathbb{E}(P_2(X-x_0)(X-x_0)^T P_2^T) \\ &= P_2\,\mathbb{E}((X-x_0)(X-x_0)^T)P_2^T \\ &= P_2 C_0 P_2^T && C_0 = \left(P_2^T P_2\right)^{-1} = P_2^{-1}P_2^{-T} \\ &= I\end{aligned}$$

and therefore W is *Gaussian white noise*. The introduction of the random variable W changes (2.49) to

$$\begin{aligned}
&\arg\min_x \left\|P_1(y - Ax - e_0)\right\|^2 + \left\|P_2(x - x_0)\right\|^2 \\
&\quad = \arg\min_w \left\|P_1(y - AP_2^{-1}w - Ax_0 - e_0)\right\|^2 + \|w\|^2 \\
&\quad = \arg\min_w \left\|P_1(y - Ax_0 - e_0) - P_1AP_2^{-1}w\right\|^2 + \|w\|^2 \\
&\quad = \arg\min_w \left\| \begin{bmatrix} P_1(y - Ax_0 - e_0) \\ 0 \end{bmatrix} - \begin{bmatrix} P_1AP_2^{-1} \\ I \end{bmatrix} w \right\|^2 .
\end{aligned} \tag{2.50}$$

This means that x_{map} is the solution in the least-squares sense of the linear system

$$\begin{bmatrix} P_1AP_2^{-1} \\ I \end{bmatrix} w = \begin{bmatrix} P_1(y - Ax_0 - e_0) \\ 0 \end{bmatrix}, \quad w = P_2(x - x_0)$$

and therefore solving this linear system is equal to solving iteratively the preconditioned linear system

$$\begin{aligned}
&P_1AP_2^{-1}w = P_1(y - Ax_0 - e_0), \quad w = P_2(x - x_0) \qquad \text{or equivalently} \\
&P_1AP_2^{-1}w = P_1(y - Ax_0 - e_0), \quad x = P_2^{-1}w + x_0.
\end{aligned} \tag{2.51}$$

This equation is similar to the one presented in (2.42). Therefore, we know that preconditioning and a regularized inverse problem are closely related to each other. The right preconditioner P_2 in (2.51) acts on the solution and can be of great value when prior information of x is known. On the other hand, the left preconditioner acts on the data fidelity term. Since data is often noisy, a good left preconditioner has to make sure that the noise is somewhat controlled after applying it.

2.4.3 Methods of preconditioning

Two preconditioning methods that construct approximations of the inverse of the forward mapping are incomplete factorizations and (sparse) approximated inverse methods. These two methods are briefly discussed below.

(Incomplete) factorizations

A factorization method first factorizes the forward mapping A into an easier form to invert, of which Cholesky factorizations and LU factorizations are popular [9]. The part on Cholesky is omitted since it is roughly similar. The LU factorization is based on Gaussian elimination and can only be applied to a square forward mapping A, where all leading principal submatrices $A_{(1:k)\times(1:k)}$ are nonsingular for $k = 1, \ldots, n-1$ [39]. The idea is to factorize A into a product of a lower triangular matrix L and an upper triangular matrix U, i.e. $A = LU$, before constructing the inverse of the matrix. The factorization process often constructs factors L and U that are more dense than A itself due to the fill-in of entries in the Gaussian elimination process, which can lead to a large memory usage. Rejecting parts of the fill-in leads to an approximate LU factorization $A \approx \bar{L}\bar{U}$, where $\bar{L}$ and $\bar{U}$ are the incomplete factors. This method is called the

incomplete LU (ILU). Discarding the fill-in can depend on the position, value or both of them [9, 80]. Other factorization methods include the *singular value decomposition* (SVD) and the QR decomposition. The QR decomposition factorizes A into an upper triangular matrix R and an orthogonal matrix Q, i.e. $Q^TQ = QQ^T = I$. For more background information on factorizations we refer to [9, 39].

Approximated inverse

A poplar method for computing the approximate inverse of a forward mapping A is the Frobenius norm minimization method [9], which was introduced in the 1970s by Benson [8]. For constructing a left preconditioner one solves

$$\min_P \ \|I - PA\|_F \tag{2.52}$$

and for the right preconditioner

$$\min_P \ \|I - AP\|_F \,, \tag{2.53}$$

where $\|\cdot\|_F$ is the Frobenius norm. Due to the definition of the Frobenius norm, i.e. $\|A\|_F = \sqrt{\mathrm{tr}(A^TA)}$, we know that

$$\begin{aligned}
\min_P \ \|I - AP\|_F^2 &= \min_{P=[p_1,\dots,p_m]} \ \left\| \Big(Ap_1 - e_1, \dots, Ap_m - e_m \Big) \right\|_F^2 , \quad \|A\|_F = \sqrt{\mathrm{tr}(A^TA)} \\
&= \min_{P=[p_1,\dots,p_m]} \ \sum_{i=1}^{m} \|Ap - e_i\|_2^2 \,, \qquad \text{every term is independent} \\
&= \sum_{i=1}^{m} \min_p \|Ap - e_i\|_2^2 \,.
\end{aligned} \tag{2.54}$$

Thus, solving the minimization problem with the Frobenius norm is equal to solving m least-squares problems, which can be solved by using gradient-based methods [68]. Then, the solution to (2.52) is equal to $P = (p_1^*, \dots, p_m^*)$, where each p_i^* is the optimal solution to (2.54).

2.5 Connecting framework based on information theory

Traditionally, a left preconditioner P_1 is designed to transform the system (IP) such that it is faster to solve. The left preconditioner transforms the forward mapping A such that the preconditioned forward mapping P_1A is close to the identity matrix, where the eigenvalues of the preconditioned forward mapping are close to one. Solving a preconditioned system iteratively with the eigenvalues close to one has a tremendous advantage in terms of speed [9]. This means that a preconditioned forward mapping close to an orthonormal system/isometry is faster solvable by iterative methods. Section 2.3 discusses that exactly these forward mappings play a key role in compressed sensing because of their low mutual coherence. Therefore, both preconditioning and compressed sensing want to obtain a forward mapping that has a low mutual coherence.

Matrices with a low mutual coherence give valuable insight in the nature of the inverse problem. The *Bayesian view* on inverse problems and aspects from *information theory* give us the opportunity to connect the field of inverse problems with compressed sensing and preconditioning via the mutual coherence. Recall from equation (2.49) that the most likely solution of a Bayesian inverse problem, when only left preconditioners are considered, is:

$$\begin{aligned} x_{\text{map}} &= \arg\max_x \pi_{\text{posterior}}(x|y) \\ &= \arg\min_x \left\| P_1(y - Ax - e_0) \right\|^2, \end{aligned} \tag{2.55}$$

where P_1 is the left preconditioner. The optimal value for x is obtained when $\nabla_x \left\| P_1(y - Ax - e_0) \right\|^2 = 0$, meaning that

$$x_{\text{map}} = \left(A^T P_1^T P_1 A \right)^{-1} A^T P_1^T P_1 (y - e_0).$$

The Hessian of the inverse problem is defined as

$$\nabla_x^2 \left\| P_1(y - Ax - e_0) \right\|^2 = A^T P_1^T P_1 A. \tag{2.56}$$

Since the inverse of the Hessian is the covariance matrix of Gaussian random variables, the following posterior distribution is obtained:

$$\pi_{\text{posterior}}(x|y) \sim \mathcal{N}\left(x_{\text{map}}, \left(A^T P_1^T P_1 A \right)^{-1} \right), \tag{2.57}$$

which only holds under the assumption that A is non-singular. The $A^T P_1^T P_1 A$ in (2.56) is called the *Fisher information matrix*, which is defined as [62]:

$$I(x) \coloneqq -\mathbb{E}_x \left[\frac{d^2}{dx^2} \ln \pi(y|x) \right] = A^T P_1^T P_1 A. \tag{2.58}$$

This Fisher information matrix gives the curvature of the log likelihood. If there is no curvature then the measurements generated from x are not very informative, since it would be harder to pick an optimal value from the posterior distribution. High curvature, thus more information, leads to a clearer optimal value of the posterior distribution. This is why a high informative system has a posterior distribution with a low variance. Therefore, the Fisher information matrix evaluates the bulk of information about x from the measurements y. A forward mappings A with a low mutual coherence is close to an orthonormal matrix, which means that the measurements

$y = Ax+e$ are close to the unknown x if the noise is not large. This means that the measurements are already very informative, meaning that knowing y gives a lot of information about the unknown x. Therefore, the information present in the inverse problem is connected to the mutual coherence of the forward mapping and thus to compressed sensing. The connection between the information in the system and preconditioning is described in equation (2.57), where it is clear that preconditioning can affect the information in the system.

Chapter 3

Proposed combined method

In this chapter the proposed method is discussed. The general idea of our proposed method is explained in the first section of this chapter. This method consists of two different preconditioners. Each of these preconditioners is explained in detail in the following two sections.

3.1 The composition

The application of preconditioning such that the preconditioned inverse problem can be solved with a CNN, is not a novel approach. Since the preconditioner approximates the inverse of the forward mapping, its use can be seen as a first reconstruction phase. An example of its use is presented by *Jin et al.* in [52], where they use a limited-angle Radon transform as forward mapping, the *filtered backprojection* (FBP) as first reconstruction phase and afterwards the problem is solved with a CNN. Due to the computational efficiency of the FBP it is the most used type of first reconstruction phase for inverse problems with the Radon transform as forward mapping [71, 76]. Since the reconstruction quality of the FBP can be poor when noise is present and a limited amount of measurement angles are used [76], it is useful to explore what type of preconditioner can make better reconstructions in these circumstances. What parameters give valuable insight in the quality of the preconditioner is still a loose end.

A low mutual coherence of the forward mapping showed to be beneficial for the classification accuracy of the neural network in [91]. Thus, it is examined if a low mutual coherence is also useful in case when an ill-posed inverse problem is solved by a CNN with the limited-angle Radon transform as forward mapping. As far as we are aware, this is the first time this is examined for this type of problem. Our preconditioning method lowers the mutual coherence of the preconditioned forward mapping as much as possible by means of matrix multiplications. Since the preconditioners lower the mutual coherence, our method is based on compressed sensing, whereas the FBP is based on the analytical solution of the Radon transform. The preconditioners are easy to construct and their contribution to the inverse problem is easy to analyze since the method consists of matrix multiplications. Furthermore, by using matrix multiplications we can connect the fields of inverse problems, compressed sensing and preconditioning via the information theory based framework presented in section 2.5. This framework enables us to give a detailed analysis of the contribution of our preconditioning method. Analysis of the effect of the preconditioner is not given in [52].

The proposed method consists of two parts. First, the preconditioner P_α from [4] is used to create the preconditioned inverse problem $P_\alpha(Ax+\eta) = P_\alpha y$, where $P_\alpha A$ has a lower mutual coherence. Afterwards a second preconditioner is used, which is the truncated regularized pseudo-inverse P_λ. The truncated regularized pseudo-inverse is chosen for the inverse because of its speed to construct and it is as successful in lowering the mutual coherence in comparison to iteratively derived preconditioners [84]. This results in the preconditioned version of inverse problem (IP):

$$\begin{aligned} P_\lambda P_\alpha (Ax+\eta) &\approx Ix + P_\lambda P_\alpha \eta \\ &= P_\lambda P_\alpha y. \end{aligned} \tag{PIP}$$

Using two preconditioners instead of one, enables us to lower the mutual coherence even further.

3.2 Preconditioner P_α

A method that lowers the mutual coherence is presented by *Alemazkoor et al.* in [4]. By designing a proper minimization problem for the construction of the preconditioner for the inverse problem $y = Ax + \eta$, they can ensure that the mutual coherence is lowered. In section 2.3 it is explained that the minimal possible mutual coherence is bounded by the Welch bound. Furthermore, it is discussed that the mutual coherence is the biggest off-diagonal (in absolute sense) of the corresponding Gram matrix in case when the forward mapping has unit norm columns. Due to the normalization this Gram matrix has ones on the diagonal. By enforcing the Gram matrix of the preconditioned forward mapping $P_\alpha A$ to be close to a matrix with ones on the diagonal and off-diagonals smaller (in absolute sense) than the Welch bound, one can ensure that the mutual coherence is lowered when P_α is multiplied with A. A matrix with ones on the diagonal and off-diagonals that are not bigger (in absolute sense) than the Welch bound, is called an equiangular tight frame (ETF) matrix and belongs to the space [4]

$$\mathcal{H}_{\mu_E} = \left\{ G \in \mathbb{R}^{m\times m} \;:\; G = G^T,\; G_{ii} = 1,\; \max_{i\neq j} |G_{ij}| < \mu_E \right\}, \tag{3.1}$$

where μ_E is equal to the Welch bound given by

$$\mu_E = \sqrt{\frac{n-m}{m(n-1)}}.$$

Furthermore, by enforcing the preconditioner to be close to the identity matrix, one can control the extent to which the noise is amplified or reduced. Assuring that the preconditioner is close to these two matrices gives the following minimization problem for the construction of the preconditioner P_α:

$$\min_{P_\alpha\in\mathbb{R}^{m\times m},\, G\in\mathcal{H}_{\mu_E}} f(G, P_\alpha) = \min_{P_\alpha\in\mathbb{R}^{m\times m},\, G\in\mathcal{H}_{\mu_E}} \left\| G - A^T P_\alpha^T P_\alpha A \right\|_F^2 + \alpha \|I - P_\alpha\|_F^2, \tag{3.2}$$

where the parameter $\alpha \geq 0$ controls how close the preconditioner is to the identity matrix. The optimal solution to (3.2) can be found by using a gradient-based method, e.g. conjugate

gradient method. The gradient of the function in (3.2) with respect to P_α is defined as

$$\begin{aligned}\frac{\partial}{\partial P_\alpha} f(G, P_\alpha) =& \frac{\partial}{\partial P_\alpha}\left[\left\|G - A^T P_\alpha^T P_\alpha A\right\|_F^2 + \alpha\|I - P_\alpha\|_F^2\right] \\ \overset{\text{def}\,\|\cdot\|_F}{=}& \frac{\partial}{\partial P_\alpha}\left[\operatorname{tr}\left\{\left(G - A^T P_\alpha^T P_\alpha A\right)^T\left(G - A^T P_\alpha^T P_\alpha A\right)\right\} + \alpha\operatorname{tr}\left\{(I - P_\alpha)^T(I - P_\alpha)\right\}\right] \\ =& \; 4P_\alpha A\left(A^T P_\alpha^T P_\alpha A - G\right)A^T + 2\alpha\left(P_\alpha - I\right). \end{aligned} \tag{3.3}$$

For the details of the calculation of this partial derivative we refer to Appendix A. The algorithm for creating the preconditioner is given in Algorithm 1.

Algorithm 1 Alternating derivation of the solution to minimization problem (3.2) [4].

1: Input: random matrix P_α^0, identity matrix G_0, threshold δ_{thr} and Welch bound μ_E
2: **while** $\delta_t < \delta_{\text{thr}}$ **do**
3: Form $D = P_\alpha^{t-1} A$
4: Normalize the columns of D to derive $\tilde{D}$
5: Form $\tilde{G} = \tilde{D}^T\tilde{D}$
6: Form $G_t(i,j) = \begin{cases} 1 & i = j, \\ \tilde{G}_{ij} & i \neq j, \left|\tilde{G}_{ij}\right| \leq \mu_E \\ \operatorname{sign}\left(\tilde{G}_{ij}\right)\cdot\mu_E & i \neq j, \left|\tilde{G}_{ij}\right| > \mu_E \end{cases}$
7: Solve $P_\alpha^t = \underset{P_\alpha \in \mathbb{R}^{m\times m}}{\arg\min}\; f(G_t, P_\alpha^{t-1})$ using conjugate gradient method and derivative (3.3)
8: Calculate $\delta_t = \left[f(G_t, P_\alpha^t) - f(G_{t-1}, P_\alpha^{t-1})\right] / f(G_{t-1}, P_\alpha^{t-1})$
9: **end while**

This algorithm indicates that P_α is adjusted every iteration, which leads to changes in $\tilde{G}$. By iteratively adjusting the entries of $\tilde{G}$ that are bigger than the Welch, one gets the adjusted Gram matrix G. This matrix G will be used for minimizing $f(\cdot)$, which forces the updated P_α to be of such a form that the Gram matrix $G = A^T P_\alpha^T P_\alpha A$ is close to an ETF matrix. The objective function is solved by using the conjugate gradient method and the optimization toolbox *MinFunc* [82]. This toolbox is specifically build for optimization problems with matrices as input.

3.3 Preconditioner P_λ

In section 2.4.3 two different preconditioning methods are discussed, where it is explained that the factorization method is based on Gaussian elimination. The LU factorization can only be applied to a square forward mapping A when all leading principal submatrices are nonsingular, whereas the ILU factorization is originally made for M-matrices and diagonally dominant matrices [66]. A matrix A is an M-matrix if $a_{ij} \leq 0$ for $i \neq j$, A nonsingular and all entries of A^{-1} are bigger or equal to zero. Discretizations of one variable elliptical PDEs are characterized by this type of matrix [80]. A diagonally dominant matrix A satisfies

$$|a_{ii}| \geq \sum_{i\neq j}|a_{ij}| \quad \forall i.$$

For many inverse problems the forward mapping A is not even close to one of these discussed matrices, leading to a forward mapping A that often contains zero pivots. This causes the Gaussian elimination process to break down due to the division by zero. One can add a constant ϵ to the diagonal of the forward mapping but this gives no guarantee that the factorization is accurate [9]. Break down also takes place when a QR decomposition is made from a rank deficient forward mapping A [39]. A method that is unaffected by these breakdowns, is the approximated inverse preconditioning method and will therefore be the preconditioning method for inversion of the forward mapping. A simple method that approximates the inverse of a matrix is the application of the pseudo-inverse as a preconditioner. Examples of its use as preconditioner can be found in [31, 59, 84]. The pseudo-inverse is the minimizer to:

$$P = \arg\min_P \|I - PA\|_F \tag{3.4}$$

and the closed form solution is summarized in the following proposition.

Proposition 1 (from [84])**.** *Given $A \in \mathbb{R}^{m\times n}$, let $U\Sigma V^T$ be the singular value decomposition of A and $\sigma_1, \ldots, \sigma_r$ the nonzero singular value of A in descending order. Then the minimizer of (3.4) is given by*

$$P = \begin{cases} A^T \left(AA^T\right)^{-1} & r = m \\ \left(A^T A\right)^{-1} A^T & r = n \\ V diag\left(\left[\underbrace{1/\sigma_1, \ldots, 1/\sigma_r}_{r}, \underbrace{0, \ldots, 0}_{n-r}\right]\right) U^T & r < \min\{m, n\}. \end{cases} \tag{3.5}$$

The use of preconditioners like the pseudo-inverse can severely amplify the noise [4]. Therefore, the minimization problem (3.4) needs to be regularized and leads to a regularized pseudo-inverse that minimizes

$$P_\lambda = \arg\min_P \|I - PA\|_F + \lambda \|P\|_F \tag{3.6}$$

where $\lambda > 0$. The closed form solution is equal to:

$$P_\lambda = \begin{cases} A^T \left(AA^T + \lambda I\right)^{-1} & r = m \\ \left(A^T A + \lambda\right)^{-1} A^T & r = n \\ V_{\hat{r}} \Psi_{\hat{r}} U_{\hat{r}}^T & r < \min\{m, n\}, \end{cases} \tag{3.7}$$

where $\hat{r} \leq r = \text{rank}(A)$, $A = U\Sigma V^T$, $V_{\hat{r}}$ consists of the first $\hat{r}$ columns of V, $U_{\hat{r}}$ consists of the first $\hat{r}$ columns of U, and $\Psi_{\hat{r}} = \text{diag}\left(\frac{\sigma_1}{\sigma_1^2+\lambda}, \ldots, \frac{\sigma_{\hat{r}}}{\sigma_{\hat{r}}^2+\lambda}\right)$. The derivation of the regularized pseudo-inverse is given in Appendix B.

Chapter 4

Experimental setup

This chapter explains the details of the experimental setup of this thesis and starts with a brief background of the application. In this part the corresponding forward mapping is examined and its reconstruction method. Subsequently it is explained to which data sets the forward mapping is applied in order to create the data and how the parameters of our proposed method are optimized. Afterwards, the different reconstruction methods are discussed, including their implementation details. Finally, the quality measures are discussed that are used to assess the quality of the reconstructions.

4.1 Radon transform

In the section a brief introduction is given on the Radon transform and its application to medical imaging. Afterwards an analytical reconstruction method of the Radon transform is discussed and is denoted by the filtered backprojection.

4.1.1 The forward mapping

The discovery of X-rays was a milestone in the development of the field of medical imaging. For a specific projection angle θ, the X-ray beams travel along a line $x\cos\theta + y\sin\theta = t$ whereby t varies for each beam. The measured intensity $M(t,\theta)$ at the receiver, the attenuated initial X-ray intensity, is the line integral along line $x\cos\theta + y\sin\theta = t$, denoted by [54]:

$$M(t,\theta) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} O(x,y)\delta(x\cos\theta + y\sin\theta - t)\,dx\,dy, \tag{4.1}$$

where the measured intensity depends on the structure inside the object and $\delta(\cdot)$ is the Dirac delta function. The expression (4.1) is referred to as the *Radon transform* of the object $O(x,y)$. The complete measured signal for a specific θ, is composed by a sequence of line integrals.

Due to the use of X-rays in *computerized tomography* (CT), the study on inverting the Radon transform is very important. A CT scanner obtains X-ray images of the object O inside the scanner while the X-ray transmitter and receiver rotate around the object. The received X-ray images are referred to as *measurements* or *projections* and are obtained for a certain amount of projection angles θ. An illustration of a CT setup is given in Figure 4.1. How the Radon transform can be inverted, is explained in the next section. For more background information on CT, we refer to [76]. For the creation of data a discretized version of the Radon transform [58] is used and will be the forward mapping A in (IP).

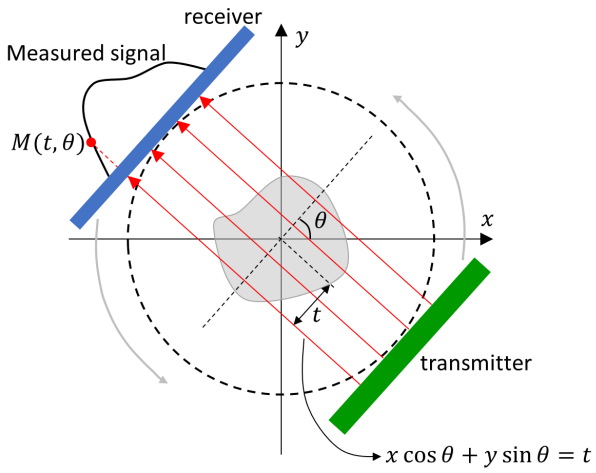


Figure 4.1: *Illustration of 2D CT, where a rotating transmitter irradiates the object with X-rays and the receiver collects the attenuated signals along the travel pathways of the X-rays.*

4.1.2 Reconstructing the object

Since the measured intensities $M(t,\theta)$ are line integrals, one way of reconstructing the intensity profile of the object is by smearing back those intensities over that specific line, which is called *backprojection*. By smearing back the intensities for all different lines and projection angles, a blurry reconstruction of the object O is obtained. In order to sharpen the reconstruction a *high-pass filter* is applied before backprojecting the measured intensities, in which low frequencies are suppressed in comparison to high frequencies [54]. The filtering and backprojection together is called the *filtered backprojection* (FBP) and is considered to be the standard method for reconstructing the object inside the scanner. Since the filter is applied on the frequencies of the measured signal, the measurements for a specific projection angle θ first need to be transformed to the Fourier domain:

$$F(w,\theta) = \int_{-\infty}^{\infty} M(t,\theta) e^{-i2\pi wt} dt. \tag{4.2}$$

Then, the filtering $|w|$ is applied and afterwards the inverse Fourier transform, which leads to:

$$Q(t,\theta) = \int_{-\infty}^{\infty} F(w,\theta)|w|\, e^{i2\pi wt} dw. \tag{4.3}$$

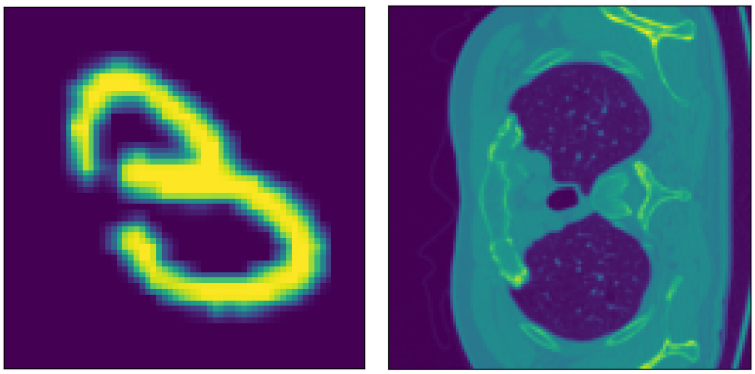
By backprojecting the filtered projections for every projection angle θ along the line $x\cos\theta + y\sin\theta = t$, one completes the FBP and obtains the reconstruction of the object:

$$\begin{aligned} O(x,y) &= \int_0^{\pi} Q(t,\theta)\, d\theta \\ \text{thus } O(x,y) &= \int_0^{\pi} Q(x\cos\theta + y\sin\theta, \theta)\, d\theta. \end{aligned} \tag{4.4}$$

The choice of filter can also play an important role when one recovers the object. The used filter $|w|$ in (4.3) is called the *Ram-Lak filter*. For more details on the FBP and other filters, we refer to [54, 72, 76].

4.2 Data

The data is generated by applying the Radon transform to two different data sets, of which the MNIST data set [56] is the first. These images are normalized, scaled to 64×64 pixels and for the Radon transform only 4 measurement angles are used. Each angular view is discretized into $\sqrt{2} \cdot 64 \approx 91$ components such that each angle has the same resolution. To the measurements a Gaussian noise with noise level $\delta = 0.05$ is added, where the average maximum value of the measurements is 2.1. The second data set is the Low-Dose Parallel Beam (LoDoPaB) CT data set [57]. The images are scaled to 128×128 pixels and for the Radon transform only 36 angles are used. All angular views are discretized into $1.5 \cdot 128 = 192$ components such that they have the same resolution. To each measurement Gaussian noise is added such that the noisy image has a signal-to-noise ratio (SNR) of 40 dB. Examples of both data sets are given in Figure 4.2.



(a) *Example MNIST.* **(b)** *Example LoDoPaB.*

Figure 4.2: *Example of a sample of each data set.*

4.3 Parameter estimation

Our proposed method consists of a product of two matrices, both with one parameter, where λ is the parameter for the pseudo-inverse and α for the other preconditioner. We aim to lower the mutual coherence of the preconditioned forward mapping as much as possible. For the optimization of these parameters we use a dataset of three figures of 48×48 pixels that mimic certain features from the MNIST and LoDoPaB data sets and are illustrated in Figure 4.3. These figures contain sharp edges, smooth parts or a combination of both. The forward mapping A is the Radon transform with 24 measurement angles. It is applied to all images and subsequently Gaussian noise is added such that the image has an SNR of 40 dB. Afterwards our proposed method is applied to these noisy measurements. For each α the parameter λ is chosen that minimizes the average mean squared error (MSE) of the images in Figure 4.3 and their preconditioned counterpart. Then, the combination of parameters (α, λ) is chosen that gives the lowest mutual coherence of the preconditioned forward mapping $P_\lambda P_\alpha A$. If different parameter combinations give the same mutual coherence then the combination with the lowest average MSE is chosen.

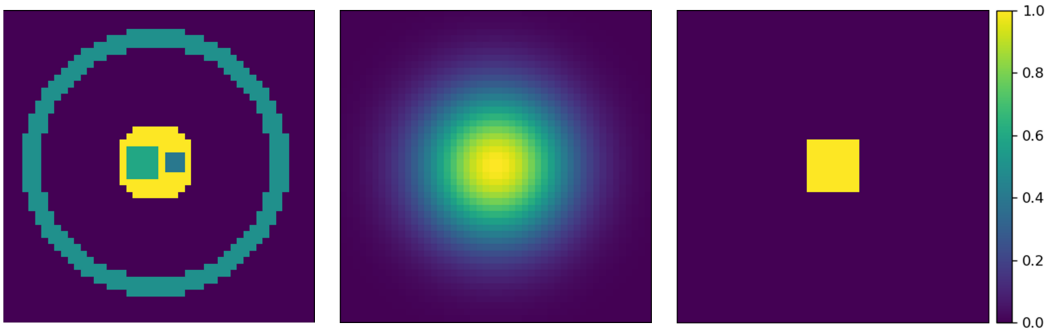


Figure 4.3: *The three used figures for parameter estimation of our proposed method.*

The application of the preconditioner will be the first part of the total reconstruction process. Due to the fact that a limited angle Radon transform is used, severe artifacts will be present after the first reconstruction phase. The general treatment is then the addition of a second reconstruction method that can eliminate the artifacts. This is discussed in the next section.

4.4 Reconstruction methods

For the second reconstruction phase, two different reconstruction methods are applied to the results of our first reconstruction phase, i.e. the preconditioned data. For comparison purposes, these two second phase reconstruction methods are also applied to the standard reconstruction/preconditioning method of the Radon transform, i.e. the FBP with Ram-Lak filter. The quality of our proposed method will be judged based on the outcomes of the two second phase reconstruction methods.

4.4.1 Reconstruction method 1: residual network

In order to give a valid comparison between the FBP and our proposed network, two different CNNs are used. For image reconstruction problems the U-Net is the standard reconstruction method. This network is so powerful that it can be hard to judge how much the first reconstruction phase contributes to the total reconstruction process. Therefore, also a smaller less powerful network is used as second phase reconstruction method. This small-scale neural network is illustrated in Figure 4.16 and due to the skip connection this network will be referred to as *residual network*. In this network the skip connection connects the input with the output. This means that the network learns the residual between the input and groundtruth, which is proven to be easier to learn [42]. Only 6 convolutional layers without max pooling are considered since this leads to a network with a limited receptive field. Together with the ground truth, the preconditioned data will form the input pairs for the neural network.

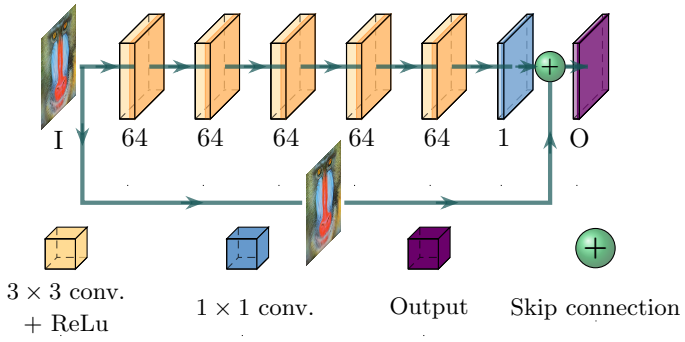


Figure 4.16: *Illustration of a small residual network, where the numbers indicate how many filters are used, where I is the input and O is the output.*

4.4.2 Reconstruction method 2: U-Net

The U-Net is known for its expressive power, which is due to its large receptive field. A large receptive field means that many pixels in the input layer contribute to the a pixel in the output layer. Therefore, more dependencies between pixels can be explored which makes the network more powerful. This big receptive field is due to the use of multiple max pooling layers and many convolutional layers. Due to the many convolutional layers, it is prone to vanishing gradients, which is prevented by the use of the skip connections. Also in this network a skip connection is used that connects the input with the output. The U-Net is illustrated in Figure 4.52.

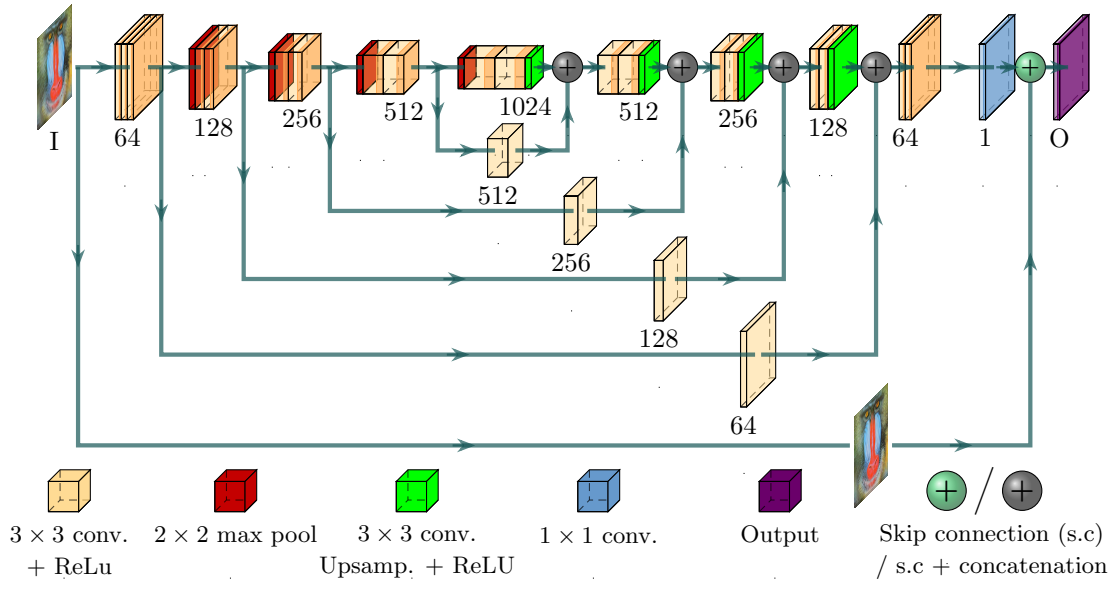


Figure 4.52: *Illustration of a U-Net, where the numbers underneath the layers represent the used filters per convolutional layer, where I is the input and O is the output.*

4.5 Implementation

The small-scale residual network only uses convolutional layers and one skip connection. The U-Net is the same as presented in [52], except that batch normalization is omitted. Both neural networks make use of biases, the ℓ^2 loss function (MSE) and the leaky ReLU activation function with parameter $\gamma = 0.1$. The networks make use of the uniform Xavier initializer, in which the weights in each layer $W_{ij} \sim U\left[-\frac{\sqrt{6}}{\sqrt{n_{in}+n_{out}}}, \frac{\sqrt{6}}{\sqrt{n_{in}+n_{out}}}\right]$, where n_{in} and n_{out} represent the size of the previous and next layer, respectively. The neural networks are trained in Tensorflow with only 100 training samples. In order to obtain a good validation, more validation samples are considered. This also gives the opportunity to examine the case where more training samples are given. The networks are applied to images in the test set. Based on the reconstructions of the test data, we can determine the quality of the first reconstruction phase. For our proposed method we make use of the discretized version of the Radon transform, while for the FBP the Radon transform from the 'scikit-image' toolbox in Python is used. Due to the large number of parameters in the U-Net network, the stopping criterion is based on the *moving average*. Every epoch the average validation loss of the last 5 epochs is calculated and the training is stopped when this moving average is increasing for three consecutive epochs. The optimization is done with the ADAM optimizer with an exponential weight decay of the learning rate. Other necessary information is given in Table 4.1.

Table 4.1: *Description of the parameters for each second phase reconstruction method (RM).*

Parameters	**RM 1**	**RM 2**
# conv. layers	6	20
# training samples	100	100
# validation samples	250	250
# test samples	250	250
# epochs	500	moving average
# kernels per conv. layer	64	varies from 64 to 1024
kernel size	5×5	5×5
batch size	8	8
activation function	$\text{lReLU}_{\gamma=0.1}$	$\text{lReLU}_{\gamma=0.1}$
start learning rate	10^{-3}	$5 \cdot 10^{-4}$
final learning rate	10^{-4}	10^{-4}

4.6 Quality measures

Besides visual interpretation, quantitative measures for examining the reconstruction quality are considered. In this thesis we consider the *peak signal-to-noise ratio* (PSNR) and the *structural similarity index measure* (SSIM). The PSNR is defined as

$$\text{PSNR} = 10 \cdot \log_{10}\left(\frac{MAX_{GT}}{MSE}\right), \tag{4.5}$$

where MAX_{GT} is the maximum value of the ground truth and MSE is the mean squared error between the ground truth and its reconstruction. The PSNR is a ratio between the maximal power of the signal and the corruption present in the reconstruction. From (4.5) it is clear that

in general the higher the PSNR, the better the reconstruction is. Since this is just an absolute error, it does not take into account the structural similarities between the reconstruction and the ground truth. Structural similarities can give good insight in the quality of the reconstruction (RS), which is why the SSIM is used. The SSIM is defined as [87]:

$$\text{SSIM} = \frac{(2\mu_{RS} \cdot \mu_{GT} + c_1)(2C_{RS,GT} + c_2)}{(\mu_{RS}^2 + \mu_{GT}^2 + c_1)(\sigma_{RS}^2 + \sigma_{GT}^2 + c_2)}, \tag{4.6}$$

where the constants c stabilize the fraction, μ_x is the average of x, σ_x^2 is the variance of x and $C_{x,y}$ is the covariance of x and y.

Chapter 5

Results

This chapter summarizes the results of both reconstruction phases and the analysis on the results. First, the results are given of the first reconstruction phase. Secondly, the reconstructions are given of the residual network and U-Net. Afterwards the individual contribution of P_α and P_λ is discussed and finally an analysis on the effects of our proposed method is given.

5.1 Parameter estimation

For each α the corresponding λ is chosen that minimizes the average MSE of the reconstructions. Then, the parameter pair (α, λ) is picked that minimizes the mutual coherence of $P_\lambda P_\alpha A$. If different pairs result in the same mutual coherence, then the one with the smallest average MSE is selected. Due to the ill-posedness of A, some regularization is needed. The minimal value for λ is considered to be 0.0001, since lower values of λ resulted in low quality reconstructions of other data sets. Figure 5.1 summarizes for different α the resulting λ, average MSE and mutual coherence, which states that $(\alpha, \lambda) = (5, 0.0001)$ is the optimized parameter pair.

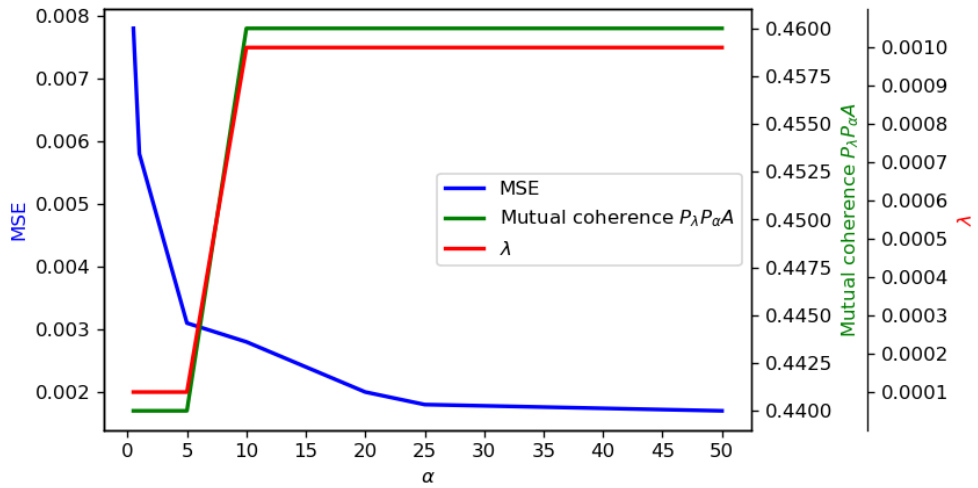


Figure 5.1: *Mutual coherence of $P_\lambda P_\alpha A$ and average MSE for different parameter pairs (α, λ).*

The reconstructions of the figures in Figure 4.3 are given in Figure 5.2 and 5.3. They show that there is almost no difference between these reconstructions.

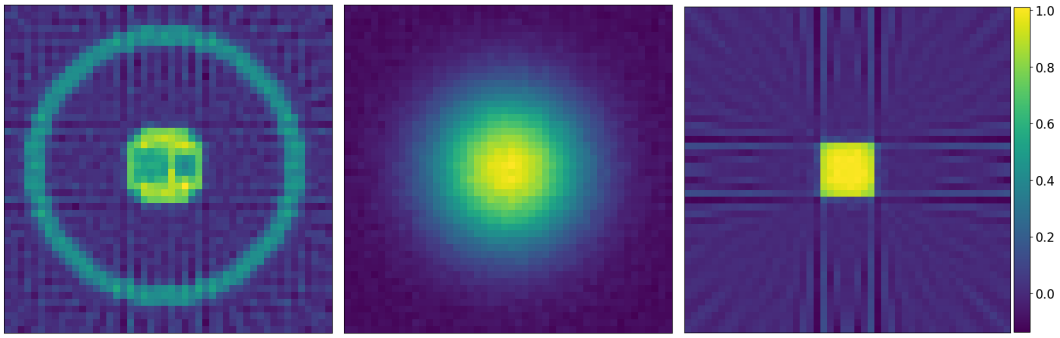


Figure 5.2: *Reconstructions of the images in Figure 4.3 using FBP.*

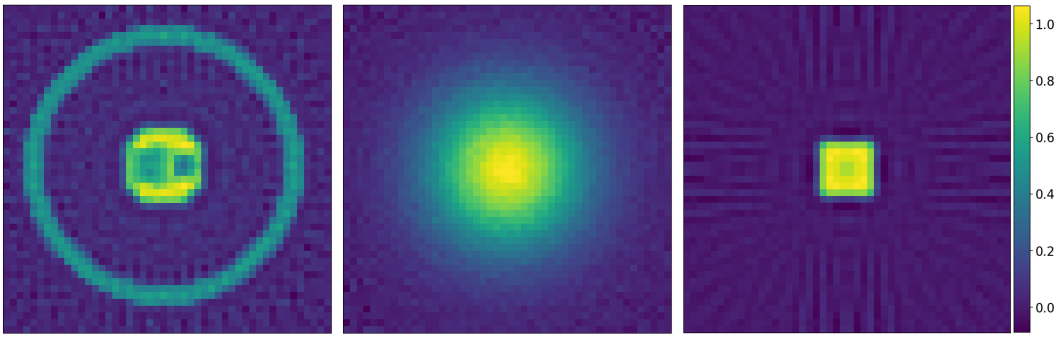


Figure 5.3: *Reconstructions of the images (preconditioned images) in Figure 4.3 using the proposed method.*

The difference is more clear when we consider horizontal cross-sections from the middle of the reconstructions. The cross-sections of the reconstructions in Figure 5.2 and 5.3 are given in Figure 5.4. Figure 5.4 shows that our proposed method leads to more overshooting near the edges and to bigger oscillations in flat regions. This leads to a bigger error of the reconstructions. An explanation for this behavior is given in section 5.5.

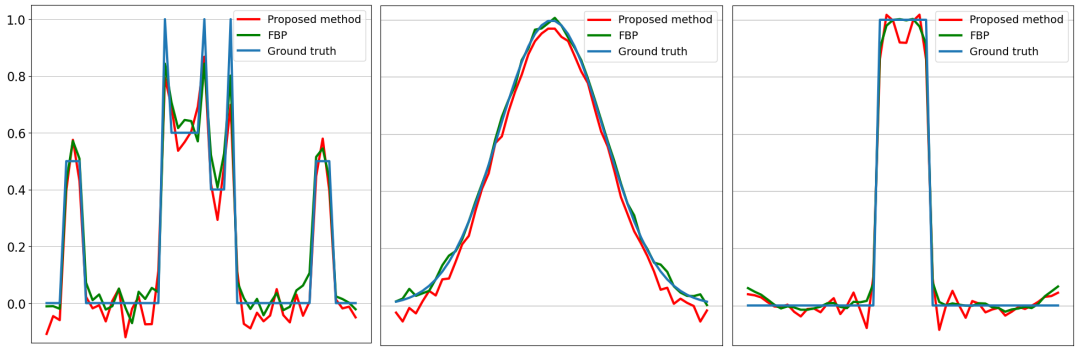


Figure 5.4: *Horizontal cross-sections from the reconstructions of the images in Figure 4.3.*

The optimized parameter pair (α, λ) is used to construct the preconditioners P_α and P_λ. These preconditioners are used to precondition the inverse problem $y = Ax + \eta$, which leads to the preconditioned inverse problem

$$P_\lambda P_\alpha (Ax + \eta) = P_\lambda P_\alpha y. \tag{5.1}$$

Our proposed preconditioning method lowers the mutual coherence from 0.732 to 0.576 in case when A is applied to the MNIST data set. For the LoDOPaB a bigger matrix A is used, where our preconditioning method lowers the mutual coherence from 0.908 to 0.60.

For both the FBP and our proposed method and for both data sets, examples of the results of the first reconstruction phase are given in Figure 5.5 and Figure 5.6.

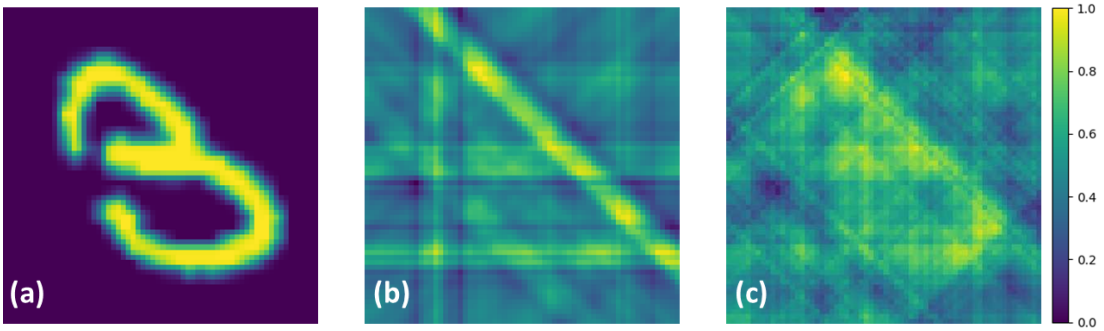


Figure 5.5: *MNIST inputs after first reconstruction phase, with in* ***(a)*** *the ground truth,* ***(b)*** *FBP and* ***(c)*** *proposed method.*

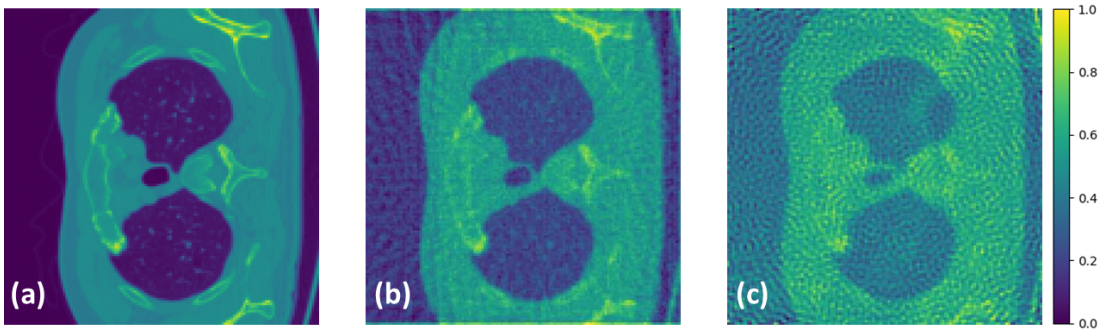


Figure 5.6: *LoDoPaB inputs after first reconstruction phase, with in* ***(a)*** *the ground truth,* ***(b)*** *FBP and* ***(c)*** *proposed method.*

The visual quality of the first reconstruction phase is similar for the FBP and our proposed method if the MNIST data set is considered. The number of used angles is so small that both methods struggle to reconstruct the data, leading to outcomes that are not close to the ground truth. When the LoDoPaB data set is considered, the outcome belonging to our proposed method contains less details. This is either due to the error of the approximation of the inverse of A or due to the magnification of noise. This is discussed in section 5.5.

5.2 Reconstruction method 1: residual network

Together with the ground truth, the outcome of the first reconstruction phase will form the input pair for the neural networks. Both the residual network and U-Net are considered to be a second phase reconstruction method. The outcomes of the residual network are given in Figure 5.7 and Figure 5.8, for MNIST and LoDoPaB, respectively. In the Appendices C and D more results of the residual network are given. From the results it can be concluded that this network has difficulty in reconstructing the MNIST data set, which is due to the use of very few measurement angles. However, the outcome of the LoDoPaB data set in Figure 5.8 gives more clarity on the difference in quality between FBP and our proposed method. When the FBP is used as first reconstruction phase, the residual network is able to give a more detailed reconstruction. This is probably due to the fact that the input of the residual network, generated by the FBP, contains more details.

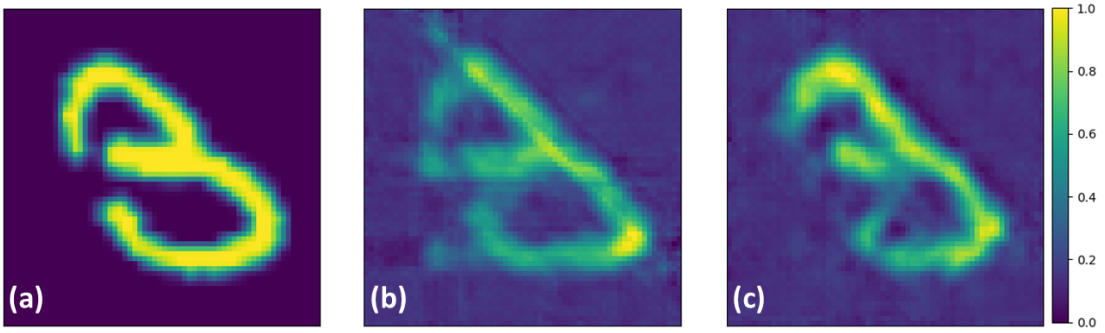


Figure 5.7: *Reconstruction by the residual network of digit 3, with in* ***(a)*** *the ground truth,* ***(b)*** *FBP and* ***(c)*** *proposed method.*

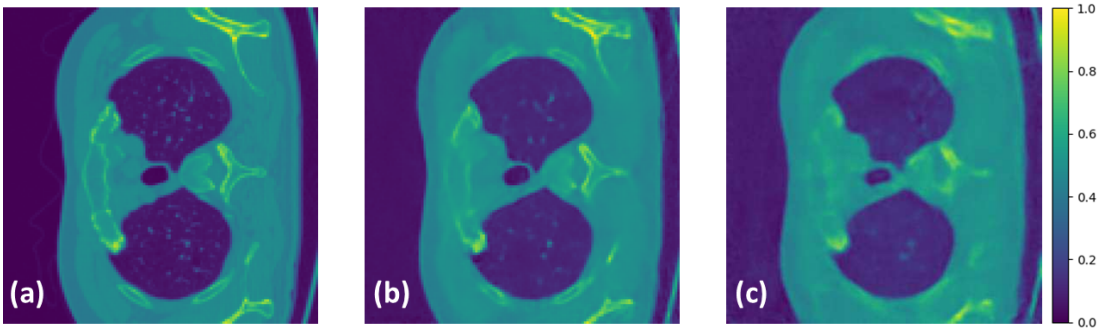


Figure 5.8: *Reconstruction by the residual network of an LoDoPaB image, with in* ***(a)*** *the ground truth,* ***(b)*** *FBP and* ***(c)*** *proposed method.*

Since less parameters are present in this network, it is less prone to overfitting. Therefore a fixed number of epochs is used rather than a moving average. In case when the LoDoPaB data set is considered, Figure 5.9 gives the train and validation loss for both the FBP and our proposed method. This figure clearly validates the observation that the FBP results in a better reconstruction of the residual network. It shows that the reconstruction loss is significantly lower when the FBP is used instead of our proposed method.

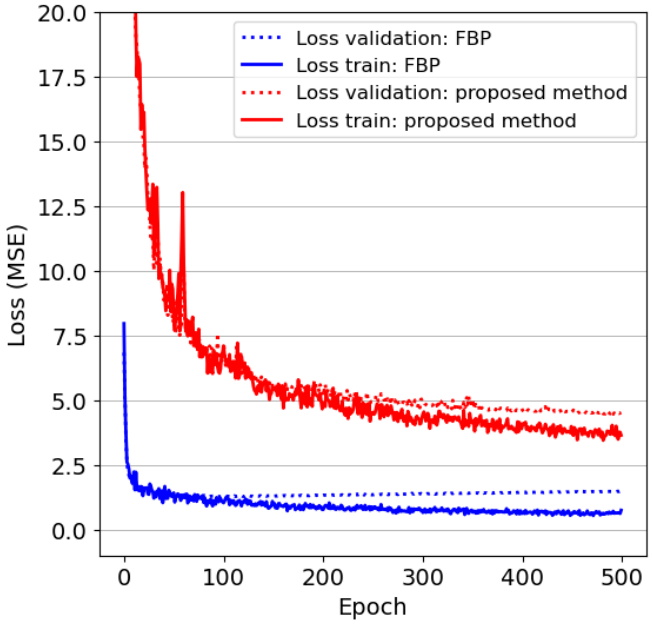


Figure 5.9: *Train and validation loss for the residual network applied to the LoDoPaB data set.*

In Table 5.1 the average and standard deviation of both the PSNR and SSIM are given. These values are based on the reconstructions of the images in the test set. The table shows that the reconstructions of the MNIST data set for the FBP and our proposed method are similar in quality, which is due to the fact that the inputs for the residual network do not differ a lot in

terms of their visual quality. There is a clear difference in quality, though, when the LoDoPaB data set is reconstructed. From section 5.1 it is clear that the FBP results in a more detailed output of the first reconstruction phase. The extra details lead to a much better reconstruction of the residual network in terms of visual quality, PSNR and SSIM.

Table 5.1: *Average and standard deviation of PSNR and SSIM for the reconstructions of the residual network.*

Data set	First reconstruction method	PSNR	SSIM
MNIST	FBP	18.8 ± 1.76	0.63 ± 0.06
MNIST	Proposed method	18.53 ± 2.19	0.66 ± 0.06
LoDoPaB	FBP	39.39 ± 3.34	$0.97 \pm 8.8 \cdot 10^{-3}$
LoDoPaB	Proposed method	35.27 ± 3.36	$0.93 \pm 14.8 \cdot 10^{-3}$

5.3 Reconstruction method 2: U-Net

The reconstructions of the MNIST data set by the U-Net are given in Figure 5.10 and in the Appendices C and D. These reconstructions show that the U-Net gives more detailed reconstructions compared to the residual network. It makes sense that the U-Net is able to give better reconstructions since the U-Net has more expressive power, due to the use of more trainable parameters and the use of max pooling. Despite its expressive power, the network is not able to exactly recover the test images of the MNIST data set. This is partly due to the fact that the inputs of the U-Net are not close to the ground truth, meaning that the network needs to learn a difficult mapping from input to ground truth. Another explanation is that the use of only limited number of training samples makes it harder to make good reconstructions of unseen data, i.e. the test data.

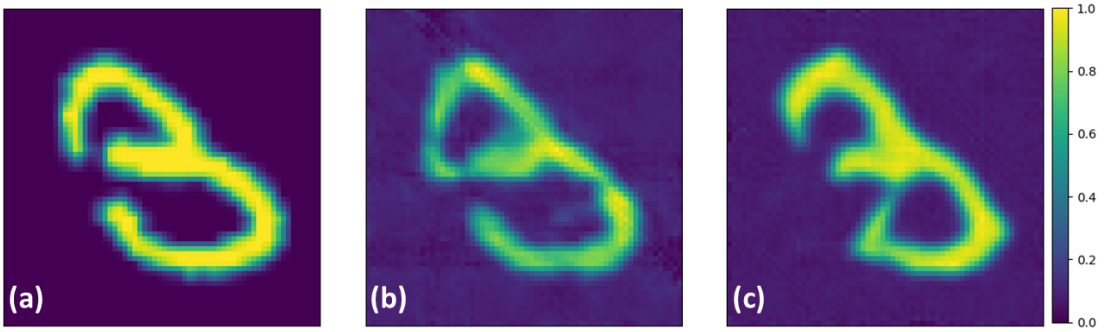


Figure 5.10: *Reconstruction by U-Net of digit 3, with in* ***(a)*** *the ground truth,* ***(b)*** *FBP and* ***(c)*** *proposed method.*

The difference between the FBP and our proposed method is more profound when the LoDoPaB data set is considered. The main reason for this difference is the fact that the input generated by the FBP contains more details than the one generated by our proposed method. This means that the U-Net needs to learn an easier mapping from input to ground truth when the FBP is used as first reconstruction phase.

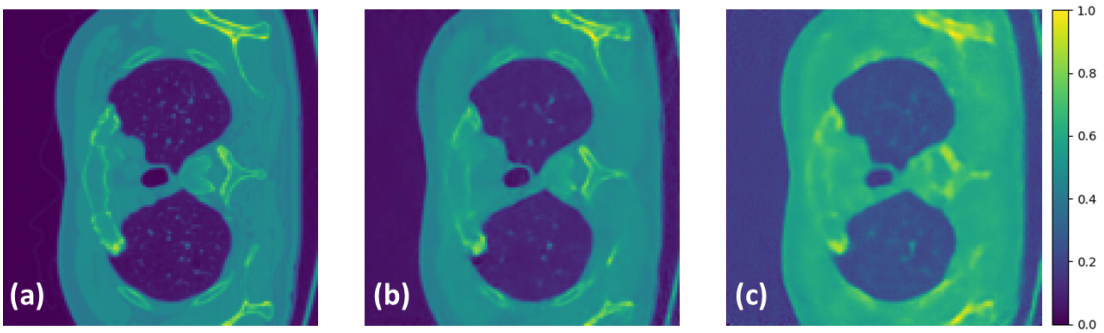


Figure 5.11: *Reconstruction by U-Net of an LoDoPaB image, with in* ***(a)*** *the ground truth,* ***(b)*** *FBP and* ***(c)*** *proposed method.*

For the U-Net the number of epochs is determined by the moving average, due to the chance of overfitting. Figure 5.12 details the train and validation loss per epoch for the LoDoPaB data set. The figure validates the claim that the reconstruction of the U-Net is better when the FBP is used as first reconstruction phase.

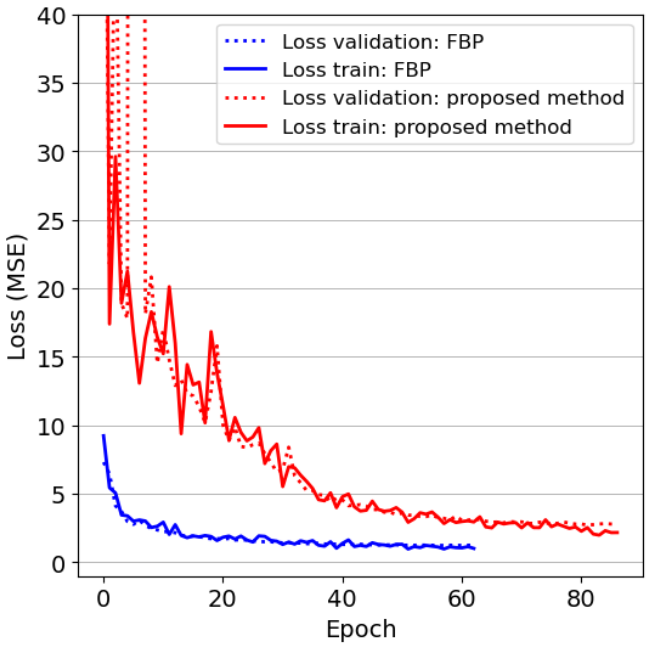


Figure 5.12: *Train and validation loss for the U-Net applied to the LoDoPaB data set.*

Table 5.2 summarizes the average and standard deviation of both the PSNR and SSIM. These values are derived from the reconstructions of the test set by the U-Net. Interestingly, in case when the MNIST data set is considered, switching from the residual network to the U-Net gives the biggest gain in SSIM rather than PSNR. It makes sense that the PSNR roughly stays the same since both the residual network and the U-Net struggle with the reconstructions of the digits. These errors contribute to the MSE, leading to a lower PSNR for both networks. On the other hand, switching to the U-Net gives a clearer digit as a result, leading to a more structured digit and therefore leading to a big gain in SSIM. Another interesting finding is the fact that switching from the residual network to the U-Net gives the biggest gain in PSNR when our proposed method is considered. An explanation for this finding is given in section 5.5. Besides that, the PSNR and SSIM show that the FBP is the superior first reconstruction phase in case of the LoDoPaB data set, which is expected because of the more detailed input for the networks. Finally, it shows that the reconstructions of the MNIST data set are roughly similar for the FBP and our proposed method.

Table 5.2: *Average and standard deviation of PSNR and SSIM for the reconstructions of the U-Net.*

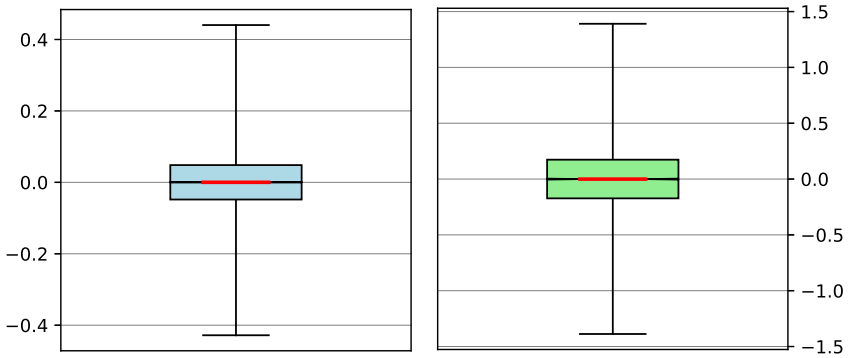
Data set	First reconstruction method	PSNR	SSIM
MNIST	FBP	20.4 ± 2.63	0.8 ± 0.05
MNIST	Proposed method	19.68 ± 2.72	0.83 ± 0.06
LoDoPaB	FBP	40.61 ± 3.36	0.98 ± 0.01
LoDoPaB	Proposed method	37.21 ± 3.23	0.95 ± 0.01

5.4 Contribution P_α and P_λ

In section 3.1 it is explained that the use of the two preconditioners P_α and P_λ can lower the mutual coherence more than when only P_λ is used. The effect of the second preconditioner P_α on the mutual coherence is, however, rather small. Depending on α and λ, the difference between the mutual coherence of $P_\lambda A$ and $P_\lambda P_\alpha A$ was at most roughly 0.02, in favor of $P_\lambda P_\alpha A$. The question is whether this small decrease in mutual coherence makes a difference for the quality of the preconditioner. The mutual coherence should have a bound when more than one preconditioner is used. If not, an infinite number of preconditioners could be used in order to reduce the mutual coherence to zero, which is not possible since the preconditioners are non-singular and the forward mapping is ill-posed. But, it is not clear what that theoretical bound is when more than one preconditioner is used. Thus, it is questionable whether it is useful to use more than one preconditioner. This small decrease in mutual coherence suggests that the mutual coherence can not be lowered significantly by a second preconditioner. More research on this topic is needed.

5.5 Analysis proposed method

From section 5.1 it is clear that, compared to the FBP, our proposed method result in different inputs for the CNNs. From Figure 5.6 it can be concluded that the input belonging to our proposed method contains less details than that of the FBP. From (*PIP*) it is clear that the use of left preconditioners affects the noise present in the system. The effect on the noise is clear from the box plots in Figure 5.13, where it is shown that the variance of the noise substantially increases after preconditioning with $P_\lambda P_\alpha$.



(a) *Noise e* **(b)** *Preconditioned noise* $P_\lambda P_\alpha e$

Figure 5.13: *Box plots of the added noise in* (*IP*) *and* (*PIP*)*, with in* ***(a)*** *noise* e *and in* ***(b)*** *noise* $P_\lambda P_\alpha e$*, respectively. Note the difference in axis.*

From an information theory perspective, the noise can be detrimental for the information in the system. This can be shown via the Bayesian view on inverse problem, which assumes that all variables of the inverse problem (*IP*) are random. When no preconditioning is used and $E \sim \mathcal{N}(e_0, \sigma^2 I)$, then the Fisher information matrix is defined as [62]:

$$I(x) = \frac{1}{\sigma^2} A^T A. \tag{5.2}$$

From (5.2) it is clear that a lower variance of the noise is beneficial for the information present in the system, which intuitively makes sense. As explained in section 2.5, the inverse of the Fisher information matrix is the covariance matrix of the posterior, which means that a lower variance of the noise implies a smaller variance of the posterior $\pi_{\text{posterior}}(x|y)$. Since our proposed method magnifies the noise substantially, there is less information present in the system, which leads to a bigger variance of the posterior.

The magnification of the noise is due to the choice of lowering the mutual coherence as much as possible. This choice implies that the regularization parameter λ can not be big, resulting in less regularization of the noise. In Figure 5.14 it is illustrated how the mutual coherence of the preconditioned forward mapping $P_\lambda P_\alpha A$ changes by an increasing regularization parameter λ. It shows that regularization of the noise implies a growth in the mutual coherence. The low regularization also explains why the cross-sections of our proposed method in Figure 5.4, contain bigger oscillations and tend to overshoot near edges.

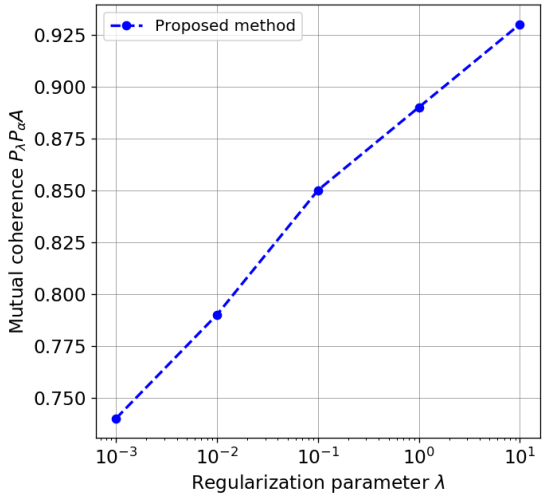


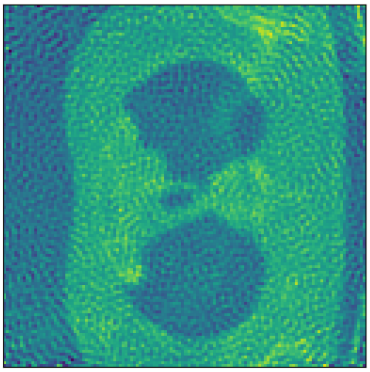
Figure 5.14: *Effect of regularization on the mutual coherence of the preconditioned forward mapping $P_\lambda P_\alpha A$.*

This effect of regularization is algebraically verified for 2×2 full rank matrices in section E.1 of Appendix E. There, the following proposition is proved:

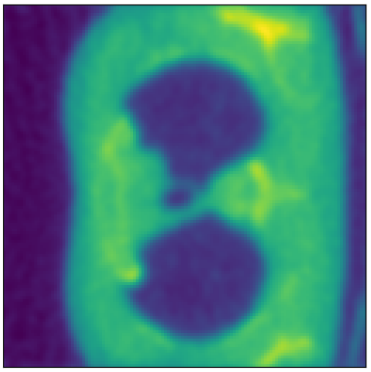
Proposition 2. *Given a 2×2 full rank matrix $A = U\Sigma V^T$, $\sigma_1 > \sigma_2$ and the regularized pseudo-inverse P_λ defined in (3.7), then the mutual coherence of the preconditioned forward mapping $P_\lambda A$ increases when λ goes up.*

For larger matrices the derivation is too involved. Therefore, a geometrical perspective on the effect of regularization of 2×2 matrices is given in section E.3, where it is shown that regularization decreases the angle between the column vectors. When column vectors get more aligned, the mutual coherence increases. For bigger $n \times n$ or $m \times n$ matrices the column vectors of $P_\lambda A$ are n-dimensional. Due to linearity it is expected that the effect of regularization is the same as in the case of a 2×2 matrix. For a rank deficient A, where $r < \min\{m, n\}$, $P_\lambda A$ gives an r-dimensional subspace. Due to linearity it is expected that the regularization has the same effect on the column vectors as in the case when A is full rank. This means that also in this case the mutual coherence increases when more regularization is applied.

This observation contradicts the conclusions made in section 2.5, where it is stated that forward mappings with a lower mutual coherence indicate that the inverse problem is more informative. One would suggest that regularization makes the inverse problem more informative due to the regularization of noise. An example is given in Figure 5.15. This figure shows that the use of more regularization leads to images with clearer details, where the regularization parameter is increased to $\lambda = 2.5$. Important is that this is not the optimal value for λ.
But, as the above result indicates, higher regularization leads to a higher mutual coherence of the preconditioned forward mapping and therefore to a less informative system. This contradiction is due to the ill-posedness of the forward mapping. For well-posed problems the pseudo-inverse preconditioner can almost perfectly approximate its inverse, leading to a preconditioned forward mapping that is close to the identity matrix. Then, when applying regularization, the mutual coherence of the preconditioned forward mapping increases since it is less close to the identity matrix. Clearly more understanding on regularization, mutual coherence, ill-posedness and informativeness is needed.



(a) *Outcome proposed method.*



(b) *Outcome proposed method with more regularization.*

Figure 5.15: *Outcome differences of the first reconstruction when the amount of regularization changes.*

Besides their effect on the information in the inverse problem, the preconditioners also affect the type of noise present in the inverse problem. Before preconditioning Gaussian noise $E \sim \mathcal{N}(e_0, \sigma^2 I)$ is present, which has the corresponding correlation matrix [53]:

$$\text{corr}(E) = \left(\text{diag}\left(\sigma^2 I\right)\right)^{-\frac{1}{2}} \sigma^2 I \left(\text{diag}\left(\sigma^2 I\right)\right)^{-\frac{1}{2}} = I, \tag{5.3}$$

where $\text{diag}(Z)$ is a diagonal matrix with on the diagonal the diagonal elements of Z. Since the off-diagonal elements of the correlation matrix are zero, this type of noise is regarded as uncorrelated. By using left preconditioners, i.e. left multiplying the noise with the invertible matrices P_λ and P_α, the covariance matrix of the preconditioned noise is $\sigma^2 P_\lambda P_\alpha P_\alpha^T P_\lambda^T$. This gives the following correlation matrix:

$$\text{corr}(P_\lambda P_\alpha E) = \left(\text{diag}\left(\sigma^2 P_\lambda P_\alpha P_\alpha^T P_\lambda^T\right)\right)^{-\frac{1}{2}} \sigma^2 P_\lambda P_\alpha P_\alpha^T P_\lambda^T \left(\text{diag}\left(\sigma^2 P_\lambda P_\alpha P_\alpha^T P_\lambda^T\right)\right)^{-\frac{1}{2}}. \tag{5.4}$$

Since the off-diagonal elements of this correlation matrix are not equal to zeros, the preconditioned inverse problem contains correlated noise. A box plot of these off-diagonals is given in Figure 5.16, where it is clear that some of the off-diagonal elements are bigger than zero. Furthermore, considering the fact that the off-diagonals of (5.4) are always between -1 and 1, we can conclude that some entries are moderately high. Roughly 0.12% of the entries have an absolute value above 0.2, which concerns more than 321000 entries. It is stated that correlated noise is harder to remove [30, 93]. In [30] the authors use a 3-layer neural network that is a mixture of a CNN and a fully connected network, while in [93] a 12-layer CNN is used for the removal of the noise. The nature of the preconditioned noise and the bigger receptive field of the U-Net might explain why switching from the residual network to the U-Net gives the biggest gain in PSNR when our preconditioning method is used.

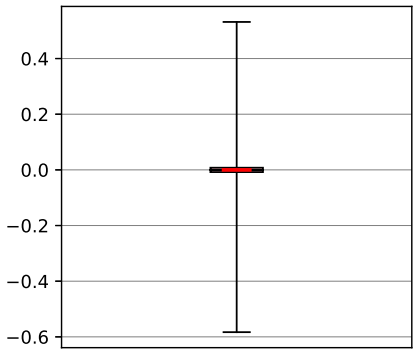


Figure 5.16: *Box plot of the off-diagonal elements of* (5.4).

Chapter 6

Conclusion and Outlook

6.1 Conclusion

It was shown that the LoDoPaB data set was informative for the comparison between the filtered backprojection and our proposed method. The outcomes of the first reconstruction phase showed that our proposed method lost more details compared to the filtered backprojection. It was shown that this was due to the magnification of noise, which was detrimental for the informativeness of the system. Furthermore, we showed that the mutual coherence was highly linked to the amount of regularization applied to our proposed method. We proved for small 2×2 matrices that the mutual coherence increased when regularization expanded. A geometrical perspective on regularization was given that gave insight into why this also holds for larger rank deficient matrices, which were of particular interest in our thesis. We showed that a low mutual coherence was only obtained when little regularization was applied. Furthermore, we demonstrated that for ill-posed problems the connection between the fields of inverse problems, preconditioning and compressed sensing was not trivial. It was concluded that more regularization suppresses the noise but leads to a less informative system.

The final conclusion is that a lower mutual coherence is not beneficial when only left preconditioners are considered. Lowering the mutual coherence as much as possible excludes the use of high amounts of regularization, leading to magnification of noise. Since right preconditioners were not considered in this research, it is still too early to conclude that a lower mutual coherence is not beneficial.

6.2 Outlook

In section 5.5 it is discussed that for ill-posed problems regularization is leading to a less informative system. It might be expected that more regularization can give a more informative system since it suppresses noise and it is beneficial for the showing the details of the image. The latter is shown in Figure 5.15 in section 5.5. It would be interesting to build an information theory based framework that incorporates regularization for ill-posed inverse problems. It might be that the mutual coherence is not suitable for this framework, since it is the worst-case scenario of the coherences between the columns of the forward mapping. One could instead use the average coherence [32]. The new developed framework could provide ideas for the development of new preconditioners.

From the results it is clear that a low mutual coherence of the forward mapping is not a good predictor for success when only left preconditioners are considered. It might be beneficial to lower the mutual coherence as much as possible when also a right preconditioner P_2 is taken into account. The effect of adding a right preconditioner is explained in Appendix F. Due to the positive semi-definiteness of $P_2^T P_2$, it is clear that the addition of P_2 lowers the variance of the posterior. But, when P_2 adds information that is not valid, it might happen that the correct solution of the inverse problem is not contained in the posterior distribution $\mathcal{N}\left(x_2^*, \left(A^T P_1^T P_1 A + P_2^T P_2\right)^{-1}\right)$. From the Bayesian view it is not clear which preconditioner P_2 is valid. It is interesting to research which type of preconditioners can add valuable information to the inverse problem. Since edges need to be preserved and the images in both data sets contain smooth parts, priors like total generalized variation [12] or Perona-Malik [74] could be good options for P_2.

Instead of using the mutual coherence to optimize the preconditioner, one could also use the mutual coherence to improve the reconstruction quality of the CNN. A low coherence between the filters in the CNN proved to be beneficial for classification problems [38] and feature learning [47], i.e. representation of the necessary features of an image. Low coherence is guaranteed when the CNN uses non-learned Gaussian random filters, i.e. filters with entries that are i.i.d Gaussian random variables. It could be interesting to examine whether it is possible to build a CNN that simultaneously learns the inverse mapping and the features of the images. These features can be added to layers deep in the CNN, i.e. layers close to the output, which possibly can have a regularizing effect. How to build a CNN that learns the features of an image is given in [47].

Bibliography

[1] Singular value decomposition (svd) of a 22 matrix. https://lucidar.me/en/mathematics/singular-value-decomposition-of-a-2x2-matrix/. Accessed: 2020-09-10.

[2] V. Abolghasemi, S. Ferdowsi, B. Makkiabadi, and S. Sanei. On optimization of the measurement matrix for compressive sensing. In *2010 18th European Signal Processing Conference*, pages 427–431. IEEE, 2010.

[3] C.C. Aggarwal. *Neural networks and deep learning*. Springer, 2018.

[4] N. Alemazkoor and H. Meidani. A preconditioning approach for improved estimation of sparse polynomial chaos expansions. *Computer Methods in Applied Mechanics and Engineering*, 342:474–489, 2018.

[5] H. Arguello and G.R. Arce. Restricted isometry property in coded aperture compressive spectral imaging. In *2012 IEEE Statistical Signal Processing Workshop (SSP)*, pages 716–719. IEEE, 2012.

[6] R.C. Aster, B. Borchers, and C.H. Thurber. *Parameter estimation and inverse problems*. Elsevier, 2018.

[7] M. Benning and M. Burger. Modern regularization methods for inverse problems. *Acta Numerica*, 27:1–111, 2018.

[8] M.W. Benson. *Iterative solution of large scale linear systems*. PhD thesis, 1973.

[9] M. Benzi. Preconditioning techniques for large linear systems: a survey. *Journal of computational Physics*, 182(2):418–477, 2002.

[10] H. Boche, R. Calderbank, G. Kutyniok, J. Vybiral, et al. *Compressed sensing and its applications*. Springer, 2015.

[11] S. Borowiec. Alphago seals 4-1 victory over go grandmaster lee sedol.

[12] K. Bredies, K. Kunisch, and T. Pock. Total generalized variation. *SIAM Journal on Imaging Sciences*, 3(3):492–526, 2010.

[13] K. Bukweon, L.S. Min, and S.J. Keun. Improving learnability of neural networks: adding supplementary axes to disentangle data representation. *arXiv preprint arXiv:1902.04205*, 2019.

[14] M. Burger and S. Osher. A guide to the tv zoo. In *Level set and PDE based reconstruction methods in imaging*, pages 1–70. Springer, 2013.

[15] R. Calderbank, S. Jafarpour, and R. Schapire. Compressed learning: Universal sparse dimensionality reduction and learning in the measurement domain. *preprint*, 2009.

[16] D. Calvetti. Preconditioned iterative methods for linear discrete ill-posed problems from a bayesian inversion perspective. *Journal of Computational and Applied Mathematics*, 198(2):378 – 395, 2007.

[17] D. Calvetti and E. Somersalo. Priorconditioners for linear systems. *Inverse Problems*, 21(4):1397–1418, 2005.

[18] D. Calvetti and E. Somersalo. Hypermodels in the bayesian imaging framework. *Inverse Problems*, 24(3):034013, 2008.

[19] E.J. Candes and T. Tao. Decoding by linear programming. *IEEE transactions on information theory*, 51(12):4203–4215, 2005.

[20] E.J. Candes and T. Tao. Near-optimal signal recovery from random projections: Universal encoding strategies? *IEEE Transactions on Information Theory*, 52(12):5406–5425, 2006.

[21] E.J. Candès and M.B. Wakin. An introduction to compressive sampling. *IEEE signal processing magazine*, 25(2):21–30, 2008.

[22] A. Chambolle and T. Pock. An introduction to continuous optimization for imaging. *Acta Numerica*, 25:161–319, 2016.

[23] S.S. Chen, D.L. Donoho, and M.A. Saunders. Atomic decomposition by basis pursuit. *SIAM review*, 43(1):129–159, 2001.

[24] J. Chung, M. Chung, and D.P. O'Leary. Optimal regularized low rank inverse approximation. *Linear Algebra and its Applications*, 468:260–269, 2015.

[25] A. Cohen, W. Dahmen, and R. DeVore. Compressed sensing and best k-term approximation. *Journal of the American mathematical society*, 22(1):211–231, 2009.

[26] T.M. Cover. Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. *IEEE transactions on electronic computers*, (3):326–334, 1965.

[27] D.L Donoho. Compressed sensing. *IEEE Transactions on information theory*, 52(4):1289–1306, 2006.

[28] D.L. Donoho and M. Elad. Optimally sparse representation in general (nonorthogonal) dictionaries via 1 minimization. *Proceedings of the National Academy of Sciences*, 100(5):2197–2202, 2003.

[29] M.F. Duarte and Y.C. Eldar. Structured compressed sensing: From theory to applications. *IEEE Transactions on signal processing*, 59(9):4053–4085, 2011.

[30] D. Eigen, D. Krishnan, and R. Fergus. Restoring an image taken through a window covered with dirt or rain. In *In Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 633–640, 2013.

[31] M. Elad. Optimized projections for compressed sensing. *IEEE Transactions on Signal Processing*, 55(12):5695–5702, 2007.

[32] M. Elad. Optimized projections for compressed sensing. *IEEE Transactions on Signal Processing*, 55(12):5695–5702, 2007.

[33] Y.C. Eldar and G. Kutyniok. *Compressed sensing: theory and applications*. Cambridge university press, 2012.

[34] H.W. Engl, M. Hanke, and A. Neubauer. *Regularization of inverse problems*, volume 375. Springer Science & Business Media, 1996.

[35] M. Fornasier and H. Rauhut. *Compressive Sensing*, pages 1–48. Springer Berlin Heidelberg, Berlin, Heidelberg, 2014.

[36] S. Foucart and H. Rauhut. A mathematical introduction to compressive sensing. *Bull. Am. Math*, 54:151–165, 2017.

[37] K. Fukushima and S. Miyake. Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In *Competition and cooperation in neural nets*, pages 267–285. Springer, 1982.

[38] R. Giryes, G. Sapiro, and A.M. Bronstein. Deep neural networks with random gaussian weights: A universal classification strategy? *IEEE Transactions on Signal Processing*, 64(13):3444–3457, 2016.

[39] G.H. Golub and C.F. Van Loan. *Matrix computations*, volume 3. JHU press, 2012.

[40] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[41] J. Hadamard. *Lectures on Cauchy's problem in linear partial differential equations*. Yale University Press, 1923.

[42] Y.S. Han, J. Yoo, and J.C. Ye. Deep residual learning for compressed sensing ct reconstruction via persistent homology analysis. *arXiv preprint arXiv:1611.06391*, 2016.

[43] P.C. Hansen. *Rank-deficient and discrete ill-posed problems: numerical aspects of linear inversion*. SIAM, 1998.

[44] P.C. Hansen. *Discrete inverse problems: insight and algorithms*. SIAM, 2010.

[45] J. Haupt, W.U. Bajwa, G. Raz, and R. Nowak. Toeplitz compressed sensing matrices with applications to sparse channel estimation. *IEEE transactions on information theory*, 56(11):5862–5875, 2010.

[46] S.S. Haykin et al. Neural networks and learning machines/simon haykin., 2009.

[47] K. He, Y. Wang, and J. Hopcroft. A powerful generative model using random weights for the deep image representation. In *Advances in Neural Information Processing Systems*, pages 631–639, 2016.

[48] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[49] D.H. Hubel and T.N. Wiesel. Receptive fields of single neurones in the cat's striate cortex. *The Journal of physiology*, 148(3):574, 1959.

[50] C.M. Hyun, S.H Baek, M. Lee, S.M Lee, and J.K. Seo. Deep learning-based solvability of underdetermined inverse problems in medical imaging. *arXiv preprint arXiv:2001.01432*, 2020.

[51] J. Jia and K. Rohe. Preconditioning to comply with the irrepresentable condition. *arXiv preprint arXiv:1208.5584*, 2012.

[52] K.H. Jin, M.T. McCann, E. Froustey, and M. Unser. Deep convolutional neural network for inverse problems in imaging. *IEEE Transactions on Image Processing*, 26(9):4509–4522, 2017.

[53] R.A. Johnson, D.W. Wichern, et al. *Applied multivariate statistical analysis*, volume 6. Pearson, 2007.

[54] A.C. Kak and M. Slaney. Principles of computerized tomographic imaging. *New York*, 1988.

[55] G. Kutyniok. Theory and applications of compressed sensing. *GAMM-Mitteilungen*, 36(1):79–101, 2013.

[56] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[57] J. Leuschner, M. Schmidt, D.O. Baguer, and P. Maaß. The lodopab-ct dataset: A benchmark dataset for low-dose ct reconstruction methods. *arXiv preprint arXiv:1910.01113*, 2019.

[58] R.M. Lewitt. Multidimensional digital image representations using generalized kaiser–bessel window functions. *JOSA A*, 7(10):1834–1846, 1990.

[59] X. Liao, H. Li, and L. Carin. Generalized alternating projection for weighted-2,1 minimization with applications to model-based compressive sensing. *SIAM Journal on Imaging Sciences*, 7(2):797–823, 2014.

[60] Z.C. Lipton, J. Berkowitz, and C. Elkan. A critical review of recurrent neural networks for sequence learning. *arXiv preprint arXiv:1506.00019*, 2015.

[61] W. Luo, Y. Li, R. Urtasun, and R. Zemel. Understanding the effective receptive field in deep convolutional neural networks. In *Advances in neural information processing systems*, pages 4898–4906, 2016.

[62] A. Ly, M. Marsman, J. Verhagen, R.P.P.P Grasman, and E.J. Wagenmakers. A tutorial on fisher information. *Journal of Mathematical Psychology*, 80:40–55, 2017.

[63] C. Marant-Micallef, K.D. Shield, J. Vignat, E. Cléro, A. Kesminiene, C. Hill, A. Rogel, B. Vacquier, F. Bray, D. Laurier, et al. The risk of cancer attributable to diagnostic medical radiation: Estimation for france in 2015. *International journal of cancer*, 144(12):2954–2963, 2019.

[64] A. Massa, N. Anselmi, G. Oliveri, and M. Salucci. Advanced microwave imaging with compressive processing-concepts, methods, and applications. In *2019 IEEE International Conference on Microwaves, Antennas, Communications and Electronic Systems (COMCAS)*, pages 1–4. IEEE, 2019.

[65] W.S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[66] J. Meijerink and H.A. Van Der Vorst. An iterative solution method for linear systems of which the coefficient matrix is a symmetric m-matrix. *Mathematics of computation*, 31(137):148–162, 1977.

[67] M. Minsky and S. Papert. An introduction to computational geometry. *Cambridge tiass., HIT*, 1969.

[68] J. Nocedal and S. Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

[69] R. Obermeier and J.A. Martinez-Lorenzo. Sensing matrix design via mutual coherence minimization for electromagnetic compressive imaging applications. *IEEE Transactions on Computational Imaging*, 3(2):217–229, 2017.

[70] I. Orović, V. Papić, C. Ioana, X. Li, and S. Stanković. Compressive sensing in signal processing: algorithms and transform domain formulations. *Mathematical Problems in Engineering*, 2016, 2016.

[71] X. Pan, E.Y. Sidky, and M. Vannier. Why do commercial ct scanners still employ traditional, filtered back-projection for image reconstruction? *Inverse problems*, 25(12):123009, 2009.

[72] D.M. Pelt and K.J. Batenburg. Improving filtered backprojection reconstruction by data-dependent filtering. *IEEE Transactions on Image Processing*, 23(11):4750–4762, 2014.

[73] K.B. Petersen, M.S. Pedersen, et al. The matrix cookbook, vol. 7. *Technical University of Denmark*, 15, 2008.

[74] P.Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on pattern analysis and machine intelligence*, 12(7):629–639, 1990.

[75] A. Quarteroni, R. Sacco, and F. Saleri. *Numerical mathematics*, volume 37. Springer Science & Business Media, 2010.

[76] R.M. Rangayyan. *Biomedical image analysis*. CRC press, 2004.

[77] F. Rosenblatt. *The perceptron, a perceiving and recognizing automaton Project Para*. Cornell Aeronautical Laboratory, 1957.

[78] S. Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

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

[80] Y. Saad. *Iterative methods for sparse linear systems*, volume 82. siam, 2003.

[81] O. Scherzer, M. Grasmair, H. Grossauer, M. Haltmeier, and F. Lenzen. *Variational methods in imaging*. Springer, 2009.

[82] M. Schmidt. minfunc: unconstrained differentiable multivariate optimization in matlab. http://www.cs.ubc.ca/~schmidtm/Software/minFunc.html, 2005.

[83] C.E. Shannon. Communication in the presence of noise. *Proceedings of the IRE*, 37(1):10–21, 1949.

[84] Z. Tong, J. Wang, and S. Han. Preconditioned ghost imaging via sparsity constraint. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1484–1488. IEEE, 2020.

[85] L.N. Trefethen and D. Bau III. *Numerical linear algebra*, volume 50. Siam, 1997.

[86] R. Vershynin. Introduction to the non-asymptotic analysis of random matrices. *arXiv preprint arXiv:1011.3027*, 2010.

[87] Z. Wang, A.C. Bovik, H.R. Sheikh, and R.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[88] A.J. Wathen. Preconditioning. *Acta Numerica*, 24:329–376, 2015.

[89] P. Werbos. Beyond regression:" new tools for prediction and analysis in the behavioral sciences. *Ph. D. dissertation, Harvard University*, 1974.

[90] Wikipedia contributors. Matrix calculus — Wikipedia, the free encyclopedia. https://en.wikipedia.org/wiki/Matrix_calculus, 2020. [Online; accessed 02-July-2020].

[91] V.N. Xuan and O. Loffeld. A deep learning framework for compressed learning and signal reconstruction. In *5th International Workshop on Compressed Sensing applied to Radar, Multimodal Sensing, and Imaging (CoSeRa)*, pages 1–5, 2018.

[92] R. Yamashita, M. Nishio, R.K.G. Do, and K. Togashi. Convolutional neural networks: an overview and application in radiology. *Insights into imaging*, 9(4):611–629, 2018.

[93] K. Zhang, W. Zuo, and L. Zhang. Ffdnet: Toward a fast and flexible solution for cnn-based image denoising. *IEEE Transactions on Image Processing*, 27(9):4608–4622, 2018.

[94] H. Zhao, O. Gallo, I. Frosio, and J. Kautz. Loss functions for image restoration with neural networks. *IEEE Transactions on computational imaging*, 3(1):47–57, 2016.

Appendix A

Derivation partial derivative w.r.t preconditioner P_α

This chapter contains the detailed calculation of the partial derivative given in (3.3). The derivation of the solution is based on derivatives with respect to matrices and trace operators. For further reading on how to calculate the derivatives with respect to matrices and trace operators, we refer to [73, 90].

The task at hand is the following partial derivative:

$$\frac{\partial}{\partial P_\alpha} f(G, P_\alpha) = \frac{\partial}{\partial P_\alpha} \left[\operatorname{tr}\left\{ \left(G - A^T P_\alpha^T P_\alpha A\right)^T \left(G - A^T P_\alpha^T P_\alpha A\right) \right\} + \alpha \operatorname{tr}\left\{ \left(I - P_\alpha\right)^T \left(I - P_\alpha\right) \right\} \right],$$

where $G \in \mathcal{H}_{\mu_E}$ with

$$\mathcal{H}_{\mu_E} = \left\{ G \in \mathbb{R}^{m \times m} \;\; : \;\; G = G^T, \;\; G_{ii} = 1, \;\; \max_{i \neq j} \left| G_{ij} \right| < \mu_E \right\}. \tag{A.1}$$

Then the partial derivative can be rewritten to

$$\begin{aligned}
\frac{\partial}{\partial P_\alpha} f(G, P_\alpha) = \frac{\partial}{\partial P_\alpha} \Bigg[& \operatorname{tr}\left\{ \left(G^T G - A^T P_\alpha^T P_\alpha A G - G^T A^T P_\alpha^T P_\alpha A + A^T P_\alpha^T P_\alpha A A^T P_\alpha^T P_\alpha A\right) \right\} \\
& + \alpha \operatorname{tr}\left\{ \left(I - P_\alpha\right)^T \left(I - P_\alpha\right) \right\} \Bigg] \\
= \frac{\partial}{\partial P_\alpha} \Big[& \operatorname{tr}\left\{ G^T G \right\} - \operatorname{tr}\left\{ A^T P_\alpha^T P_\alpha A G \right\} - \operatorname{tr}\left\{ G^T A^T P_\alpha^T P_\alpha A \right\} + \operatorname{tr}\left\{ A^T P_\alpha^T P_\alpha A A^T P_\alpha^T P_\alpha A \right\} \\
& + \alpha \operatorname{tr}\left\{ \left(I - P_\alpha\right)^T \left(I - P_\alpha\right) \right\} \Big].
\end{aligned}$$

Below it is only demonstrated how to calculate $\frac{\partial}{\partial P_\alpha} \operatorname{tr}\left\{ A^T P_\alpha^T P_\alpha A G \right\}$, since the other terms are calculated in the same manner. For the derivation we follow the same procedure presented

in [90]. The following two equations are used in the calculation of the derivative of the trace operator with respect to matrix P_α:

$$\operatorname{tr}(A) = \operatorname{tr}\left(A^T\right) \tag{A.2}$$

$$\operatorname{tr}(ABCD) = \operatorname{tr}(BCDA) = \operatorname{tr}(CDAB) = \operatorname{tr}(DABC). \tag{A.3}$$

First we define

$$\begin{aligned}
d\operatorname{tr}\left\{A^T P_\alpha^T P_\alpha A G\right\} &= \operatorname{tr}\left\{d\left(A_\alpha^T P^T P_\alpha A G\right)\right\} && \text{due to (A.3)}\\
&= \operatorname{tr}\left\{d\left(AGA^T P_\alpha^T P_\alpha\right)\right\}\\
&= \operatorname{tr}\left\{d\left(AGA^T P_\alpha^T\right) P_\alpha\right\} + \operatorname{tr}\left\{AGA^T P_\alpha^T d(P_\alpha)\right\}\\
&= \operatorname{tr}\left\{AGA^T d\left(P_\alpha^T\right) P_\alpha\right\} + \operatorname{tr}\left\{AGA^T P_\alpha^T d(P_\alpha)\right\}\\
&= \operatorname{tr}\left\{AGA^T d(P_\alpha)^T P_\alpha\right\} + \operatorname{tr}\left\{AGA^T P_\alpha^T d(P_\alpha)\right\} && \text{due to (A.3)}\\
&= \operatorname{tr}\left\{P_\alpha AGA^T d(P_\alpha)^T\right\} + \operatorname{tr}\left\{AGA^T P_\alpha^T d(P_\alpha)\right\} && \text{due to (A.2)}\\
&= \operatorname{tr}\left\{\left(P_\alpha AGA^T d(P_\alpha)^T\right)^T\right\} + \operatorname{tr}\left\{AGA^T P_\alpha^T d(P_\alpha)\right\}\\
&= \operatorname{tr}\left\{d(P_\alpha) AG^T A^T P_\alpha^T\right\} + \operatorname{tr}\left\{AGA^T P_\alpha^T d(P_\alpha)\right\} && \text{due to (A.3)}\\
&= \operatorname{tr}\left\{AG^T A^T P_\alpha^T d(P_\alpha)\right\} + \operatorname{tr}\left\{AGA^T P_\alpha^T d(P_\alpha)\right\}\\
&= \operatorname{tr}\left\{\left(AG^T A^T P_\alpha^T + AGA^T P_\alpha^T\right) d(P_\alpha)\right\} && \text{due to (A.1)}\\
&= \operatorname{tr}\left\{\left(2AG^T A^T P_\alpha^T\right) d(P_\alpha)\right\} && \text{due to (A.2)}\\
&= \operatorname{tr}\left\{\left(2P_\alpha AGA^T\right) d(P_\alpha)\right\}.
\end{aligned}$$

Therefore,

$$\frac{\partial \operatorname{tr}\left\{A^T P_\alpha^T P_\alpha A G\right\}}{\partial P_\alpha} = 2P_\alpha AGA^T.$$

In a similar fashion we can conclude that

$$\frac{\partial \operatorname{tr}\left\{G^T A^T P_\alpha^T P_\alpha A\right\}}{\partial P_\alpha} = 2P_\alpha AGA^T$$

$$\text{and } \frac{\partial \operatorname{tr}\left\{A^T P_\alpha^T P_\alpha A A^T P_\alpha^T P_\alpha A\right\}}{\partial P_\alpha} = 4P_\alpha AA^T P_\alpha^T P_\alpha AA^T$$

$$\text{and } \frac{\partial \operatorname{tr}\left\{(I - P_\alpha)^T (I - P_\alpha)\right\}}{\partial P_\alpha} = 2P_\alpha - 2I.$$

To conclude:

$$\begin{aligned}
\frac{\partial}{\partial P_\alpha} f(G, P_\alpha) &= 4P_\alpha AGA^T + 4P_\alpha AA^T P_\alpha^T P_\alpha AA^T + \alpha(2P_\alpha - 2I)\\
&= 4P_\alpha A\left(A^T P_\alpha^T P_\alpha A - G\right) A^T + 2\alpha(P_\alpha - I).
\end{aligned}$$

Appendix B

Derivation regularized pseudo-inverse P_λ

In proposition 1 the rank dependent pseudo-inverse of the measurement matrix $A \in \mathbb{R}^{m \times n}$ is given and is the minimizer of

$$P = \arg\min_P \|I - PA\|_F \,. \tag{B.1}$$

Multiplying A with a pseudo-inverse can severely magnify the smallest singular values and the noise. Regularization can prevent that, which means that the pseudo-inverse is the minimizer of

$$P_\lambda = \arg\min_P \|I - PA\|_F + \lambda \|P\|_F \,. \tag{B.2}$$

Below the closed form solution to the minimization problem (B.2) is given, where the solution is dependent on the rank of the measurement matrix A. For the derivation of the regularized rank dependent pseudo-inverse we use the same algebraic rules as in Appendix A, including (A.2) and (A.3). By applying the definition of the Frobenius norm, we know that the minimizer of (B.1) is the solution to

$$\begin{aligned} &\frac{\partial}{\partial P}\left[\operatorname{tr}\left\{ (I-PA)^T (I-PA) \right\} + \lambda \operatorname{tr}\left\{ \left(P^T P\right) \right\}\right] = 0 \\ \text{and therefore } &\frac{\partial}{\partial P}\left[\operatorname{tr}\left\{ \left(I - A^T P^T - PA + A^T P^T PA\right) \right\}\right] + \lambda \frac{\partial}{\partial P}\left[\operatorname{tr}\left\{ \left(P^T P\right) \right\}\right] = 0. \end{aligned} \tag{B.3}$$

Following the same procedure as in Appendix A, we first define

$$
\begin{aligned}
d\,\mathrm{tr}\Big\{\left(A^TP^TPA\right)\Big\} &= \mathrm{tr}\Big\{d\left(A^TP^TPA\right)\Big\} && \text{due to (A.3)}\\
&= \mathrm{tr}\Big\{d\left(AA^TP^TP\right)\Big\}\\
&= \mathrm{tr}\Big\{d\left(AA^TP^T\right)P\Big\} + \mathrm{tr}\Big\{AA^TP^Td\left(P\right)\Big\} && \text{due to (A.2)}\\
&= \mathrm{tr}\Big\{AA^Td\left(P\right)^TP\Big\} + \mathrm{tr}\Big\{AA^TP^Td\left(P\right)\Big\} && \text{due to (A.3)}\\
&= \mathrm{tr}\Big\{PAA^Td\left(P\right)^T\Big\} + \mathrm{tr}\Big\{AA^TP^Td\left(P\right)\Big\} && \text{due to (A.2) and (A.3)}\\
&= \mathrm{tr}\Big\{AA^TP^Td\left(P\right)\Big\} + \mathrm{tr}\Big\{AA^TP^Td\left(P\right)\Big\} && \text{(B.4)}\\
&= \mathrm{tr}\Big\{\left(2AA^TP^T\right)d\left(P\right)\Big\}, && \text{(B.5)}
\end{aligned}
$$

thus

$$\frac{\partial\,\mathrm{tr}\Big\{A^TP^TPA\Big\}}{\partial P} = 2AA^TP^T.$$

By performing a similar calculation it can be concluded that

$$\frac{\partial\,\mathrm{tr}\Big\{PA\Big\}}{\partial P} = A,\quad \frac{\partial\,\mathrm{tr}\Big\{A^TP^T\Big\}}{\partial P} = A \text{ and } \frac{\partial\,\mathrm{tr}\Big\{P^TP\Big\}}{\partial P} = 2P^T.$$

This means that equation (B.3) is equal to

$$
\begin{aligned}
-2A + 2AA^TP^T + 2\lambda P^T &= 0\\
\text{which gives } P_\lambda &= A^T\left(AA^T + \lambda I\right)^{-1}.
\end{aligned}
$$

By changing minimization problem (B.2) to

$$P_\lambda = \arg\min_P \|I - AP\|_F + \lambda\|P\|_F \tag{B.6}$$

one obtains the right pseudo-inverse and is denoted by

$$P_\lambda = \left(AA^T + \lambda I\right)^{-1}A^T.$$

When the measurement matrix does not have a full row or column rank, then the closed-from solution to minimization problem (B.2) is [24]:

$$P_\lambda = V_{\hat{r}} \Psi_{\hat{r}} U_{\hat{r}}^T \tag{B.7}$$

where $\hat{r} \leq r = \text{rank}(A)$, $A = U\Sigma V^T$, $V_{\hat{r}}$ consists of the first $\hat{r}$ columns of V, $U_{\hat{r}}$ consists of the first $\hat{r}$ columns of U, and $\Psi_{\hat{r}} = \text{diag}\left(\frac{\sigma_1}{\sigma_1^2+\lambda}, \ldots, \frac{\sigma_{\hat{r}}}{\sigma_{\hat{r}}^2+\lambda}\right)$. For more background on the derivation of (B.7), we refer to [24].

Appendix C

Additional results MNIST

C.1 Reconstruction method 1: residual network

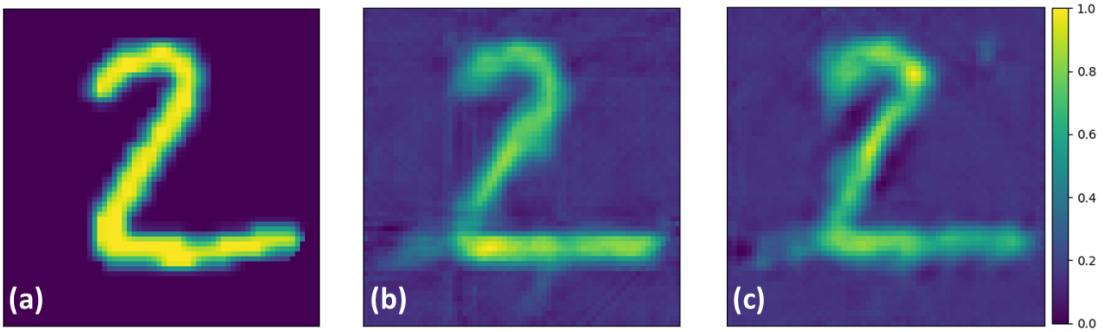


Figure C.1: *Reconstruction by the residual network of digit 2, with in* ***(a)*** *the ground truth,* ***(b)*** *FBP and* ***(c)*** *proposed method.*

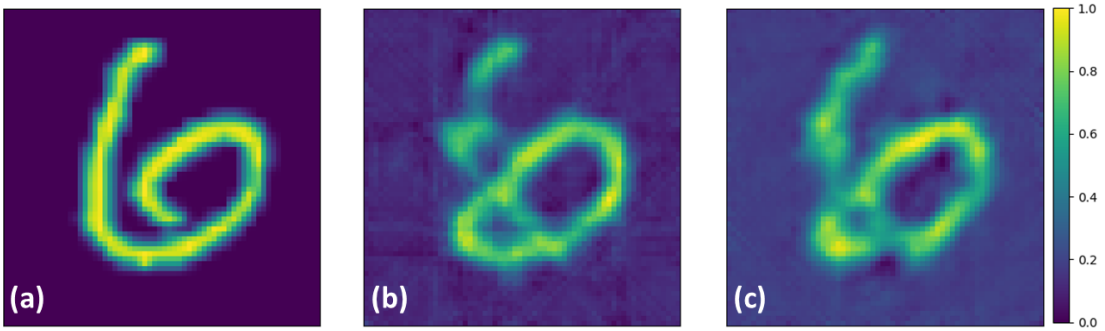


Figure C.2: *Reconstruction by the residual network of digit 6, with in* ***(a)*** *the ground truth,* ***(b)*** *FBP and* ***(c)*** *proposed method.*

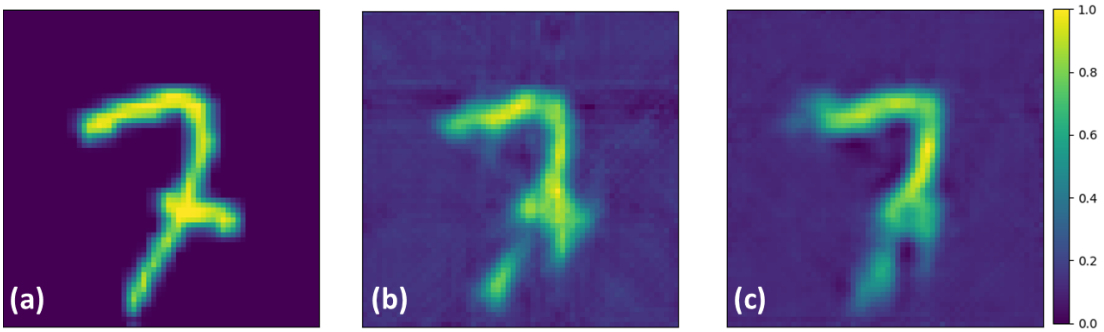


Figure C.3: *Reconstruction by the residual network of digit 7, with in* ***(a)*** *the ground truth,* ***(b)*** *FBP and* ***(c)*** *proposed method.*

C.2 Reconstruction method 2: U-Net

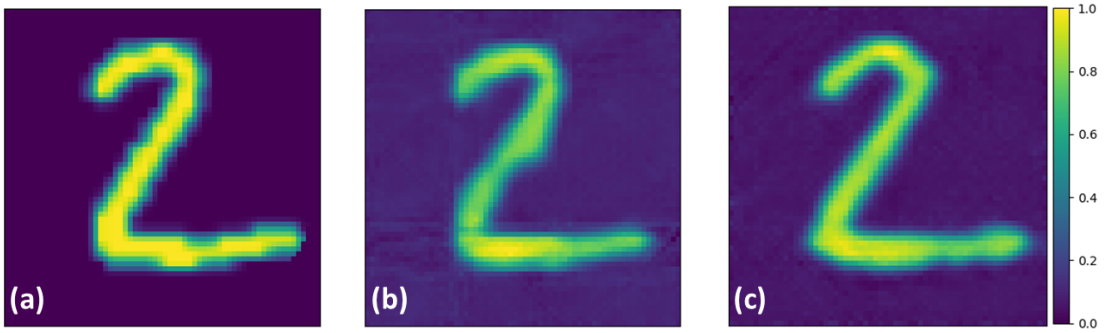


Figure C.4: *Reconstruction by U-Net of digit 2, with in* ***(a)*** *the ground truth,* ***(b)*** *FBP and* ***(c)*** *proposed method.*

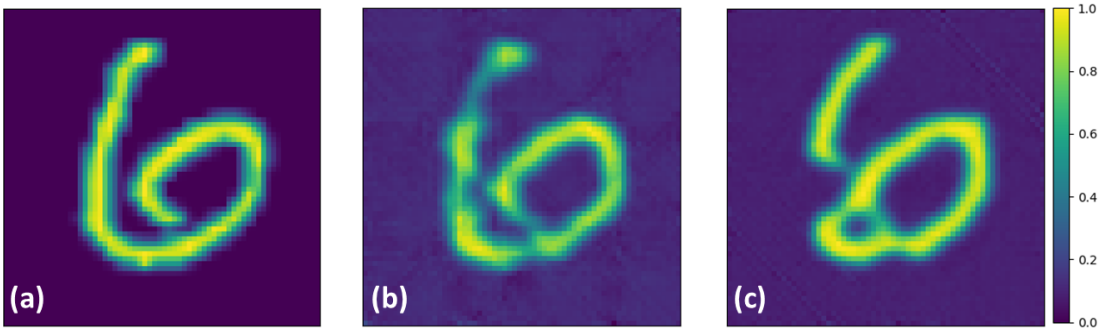


Figure C.5: *Reconstruction by U-Net of digit 6, with in* ***(a)*** *the ground truth,* ***(b)*** *FBP and* ***(c)*** *proposed method.*

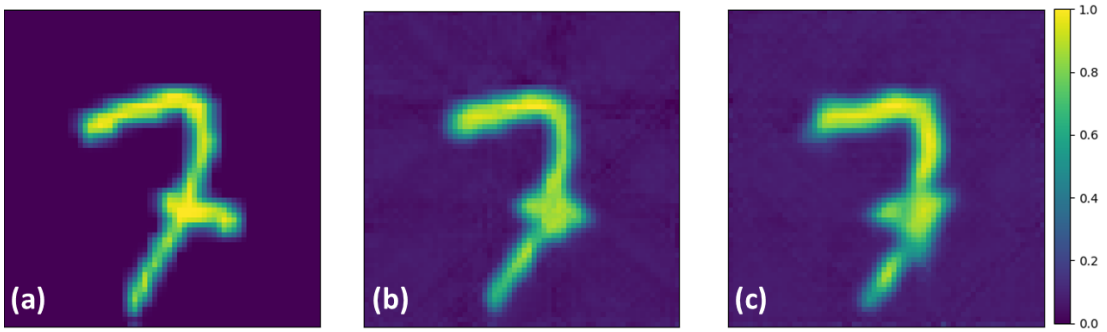


Figure C.6: *Reconstruction by U-Net of digit 7, with in* ***(a)*** *the ground truth,* ***(b)*** *FBP and* ***(c)*** *proposed method.*

Appendix D

Additional results LoDoPaB

D.1 Reconstruction method 1: residual network

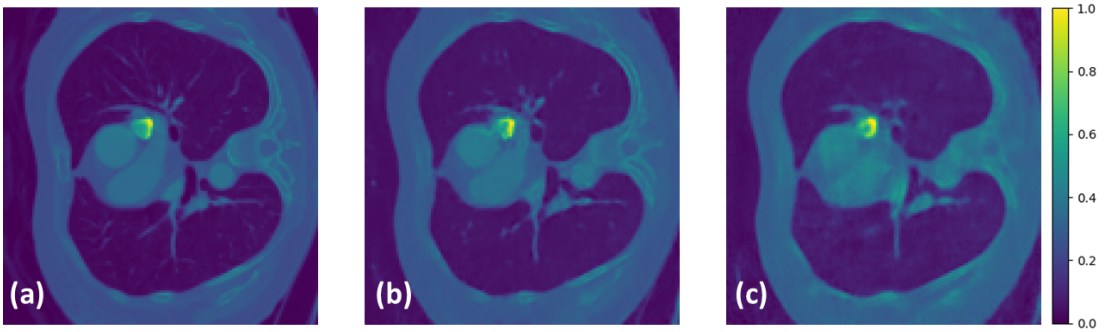


Figure D.1: *Reconstruction by the residual network, with in* ***(a)*** *the ground truth,* ***(b)*** *FBP and* ***(c)*** *proposed method.*

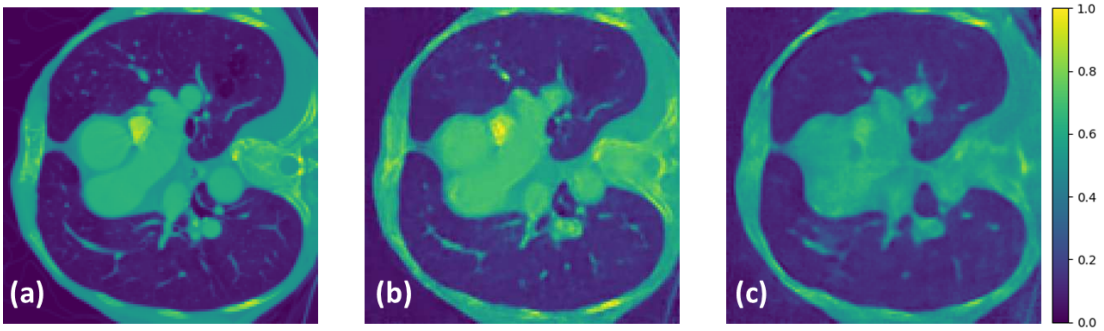


Figure D.2: *Reconstruction by the residual network, with in* ***(a)*** *the ground truth,* ***(b)*** *FBP and* ***(c)*** *proposed method.*

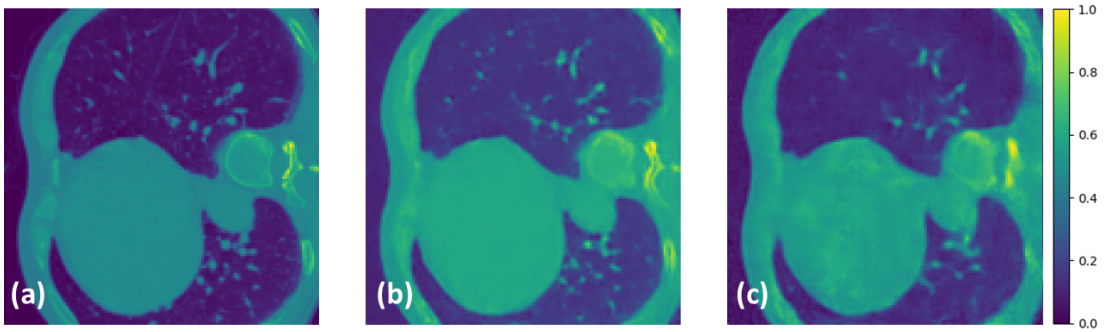


Figure D.3: *Reconstruction by the residual network, with in* ***(a)*** *the ground truth,* ***(b)*** *FBP and* ***(c)*** *proposed method.*

D.2 Reconstruction method 2: U-Net

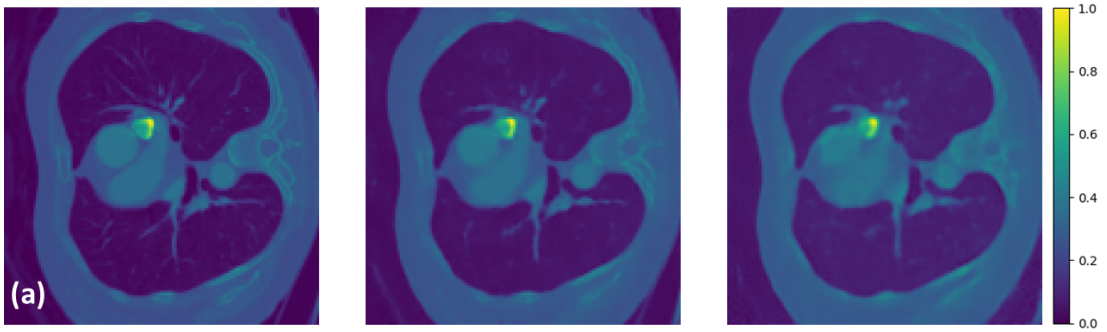


Figure D.4: *Reconstruction by U-Net, with in* ***(a)*** *the ground truth,* ***(b)*** *FBP and* ***(c)*** *proposed method.*

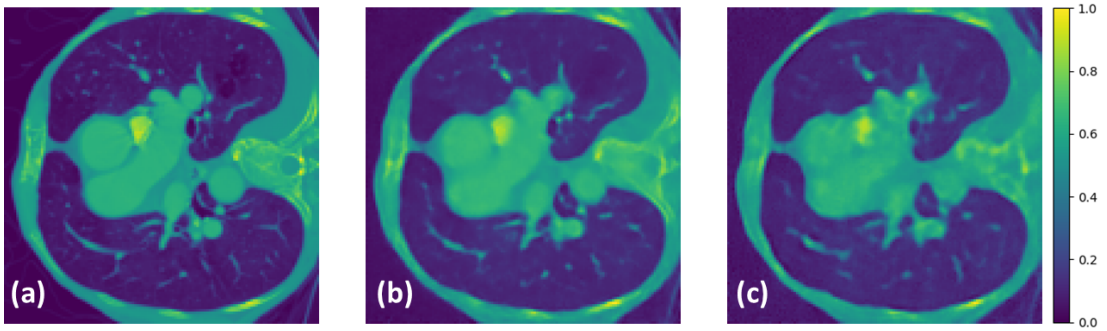


Figure D.5: *Reconstruction by U-Net, with in* ***(a)*** *the ground truth,* ***(b)*** *FBP and* ***(c)*** *proposed method.*

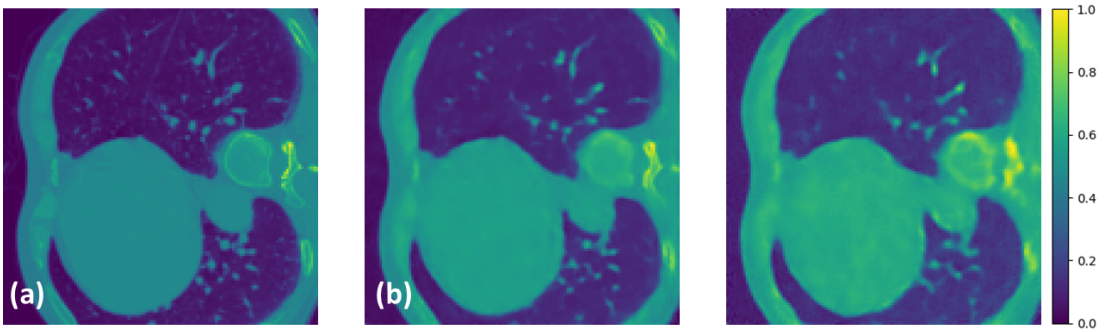


Figure D.6: *Reconstruction by U-Net, with in* ***(a)*** *the ground truth,* ***(b)*** *FBP and* ***(c)*** *proposed method.*

Appendix E

Connection mutual coherence and regularization

This chapter contains the details on the relationship between regularization and the mutual coherence of the preconditioned forward mapping. The first section proofs for a full rank 2×2 matrix that an increasing regularization parameter λ brings about an increasing mutual coherence of the preconditioned forward mapping. In the second section some aspects of the first section are discussed in details. In the final section a geometrical perspective on regularization and mutual coherence is given. This gives an intuitive explanation on why the observed effect in section E.1 holds for bigger matrices that are either full rank or rank deficient.

E.1 Effect regularization on mutual coherence

Proposition 2. *Given a 2×2 full rank matrix $A = U\Sigma V^T$, $\sigma_1 > \sigma_2$ and the regularized pseudo-inverse P_λ defined in (3.7), then the mutual coherence of the preconditioned forward mapping $P_\lambda A$ increases when λ goes up.*

Proof. The pseudo-inverse is defined as

$$P_\lambda = VSU^T$$

where $S = \text{diag}(\frac{\sigma_1}{\sigma_1^2+\lambda}, \frac{\sigma_2}{\sigma_2^2+\lambda})$. This means that

$$P_\lambda A = \begin{bmatrix} | & | \\ v_1 & v_2 \\ | & | \end{bmatrix} \begin{bmatrix} — & \frac{\sigma_1^2}{\sigma_1^2+\lambda} v_1 & — \\ — & \frac{\sigma_2^2}{\sigma_2^2+\lambda} v_2 & — \end{bmatrix} \tag{E.1}$$

where v_i is the i'th column of V. Therefore,

$$\begin{aligned} P_\lambda A &= \begin{bmatrix} \frac{\sigma_1^2}{\sigma_1^2+\lambda}v_{11}^2 + \frac{\sigma_2^2}{\sigma_2^2+\lambda}v_{12}^2 & \frac{\sigma_1^2}{\sigma_1^2+\lambda}v_{11}v_{21} + \frac{\sigma_2^2}{\sigma_2^2+\lambda}v_{12}v_{22} \\ \frac{\sigma_1^2}{\sigma_1^2+\lambda}v_{11}v_{21} + \frac{\sigma_2^2}{\sigma_2^2+\lambda}v_{12}v_{22} & \frac{\sigma_1^2}{\sigma_1^2+\lambda}v_{21}^2 + \frac{\sigma_2^2}{\sigma_2^2+\lambda}v_{22}^2 \end{bmatrix} \\ &= \begin{bmatrix} \frac{\sigma_1^2(\sigma_2^2+\lambda)v_{11}^2+\sigma_2^2(\sigma_1^2+\lambda)v_{12}^2}{(\sigma_1^2+\lambda)(\sigma_2^2+\lambda)} & \frac{\sigma_1^2(\sigma_2^2+\lambda)v_{11}v_{21}+\sigma_2^2(\sigma_1^2+\lambda)v_{12}v_{22}}{(\sigma_1^2+\lambda)(\sigma_2^2+\lambda)} \\ \frac{\sigma_1^2(\sigma_2^2+\lambda)v_{11}v_{21}+\sigma_2^2(\sigma_1^2+\lambda)v_{12}v_{22}}{(\sigma_1^2+\lambda)(\sigma_2^2+\lambda)} & \frac{\sigma_1^2(\sigma_2^2+\lambda)v_{21}^2+\sigma_2^2(\sigma_1^2+\lambda)v_{22}^2}{(\sigma_1^2+\lambda)(\sigma_2^2+\lambda)} \end{bmatrix} \\ &=: B. \end{aligned}$$

The mutual coherence of a matrix $P_\lambda A$ is the biggest off-diagonal element in absolute sense of the Gram matrix of the normalized matrix $\tilde{B}$, where the normalized matrix $\tilde{B}$ is defined as

$$\tilde{B} = \begin{bmatrix} \frac{\sigma_1^2(\sigma_2^2+\lambda)v_{11}^2+\sigma_2^2(\sigma_1^2+\lambda)v_{12}^2}{\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{11}^2+\sigma_2^4(\sigma_1^2+\lambda)^2v_{12}^2}} & \frac{\sigma_1^2(\sigma_2^2+\lambda)v_{11}v_{21}+\sigma_2^2(\sigma_1^2+\lambda)v_{12}v_{22}}{\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{21}^2+\sigma_2^4(\sigma_1^2+\lambda)^2v_{22}^2}} \\ \frac{\sigma_1^2(\sigma_2^2+\lambda)v_{11}v_{21}+\sigma_2^2(\sigma_1^2+\lambda)v_{12}v_{22}}{\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{11}^2+\sigma_2^4(\sigma_1^2+\lambda)^2v_{12}^2}} & \frac{\sigma_1^2(\sigma_2^2+\lambda)v_{21}^2+\sigma_2^2(\sigma_1^2+\lambda)v_{22}^2}{\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{21}^2+\sigma_2^4(\sigma_1^2+\lambda)^2v_{22}^2}} \end{bmatrix}.$$

This means that the Gram matrix $G = \tilde{B}^T\tilde{B}$ is symmetric and of the form

$$\begin{aligned} G &= \begin{bmatrix} 1 & x(\lambda) \\ x(\lambda) & 1 \end{bmatrix}, \\ \text{where } x(\lambda) &= \frac{\sigma_1^2(\sigma_2^2+\lambda)v_{11}^2+\sigma_2^2(\sigma_1^2+\lambda)v_{12}^2}{\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{11}^2+\sigma_2^4(\sigma_1^2+\lambda)^2v_{12}^2}} \cdot \frac{\sigma_1^2(\sigma_2^2+\lambda)v_{11}v_{21}+\sigma_2^2(\sigma_1^2+\lambda)v_{12}v_{22}}{\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{21}^2+\sigma_2^4(\sigma_1^2+\lambda)^2v_{22}^2}} \\ &\quad + \frac{\sigma_1^2(\sigma_2^2+\lambda)v_{11}v_{21}+\sigma_2^2(\sigma_1^2+\lambda)v_{12}v_{22}}{\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{11}^2+\sigma_2^4(\sigma_1^2+\lambda)^2v_{12}^2}} \cdot \frac{\sigma_1^2(\sigma_2^2+\lambda)v_{21}^2+\sigma_2^2(\sigma_1^2+\lambda)v_{22}^2}{\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{21}^2+\sigma_2^4(\sigma_1^2+\lambda)^2v_{22}^2}}. \end{aligned} \tag{E.2}$$

This equation can be simplified by using the fact that V is unitary, i.e.

$$V^TV = VV^T = I. \tag{E.3}$$

Since $V = \begin{bmatrix} v_{11} & v_{12} \\ v_{21} & v_{22} \end{bmatrix}$ and (E.3) holds, it is clear that

$$\begin{aligned} v_{11}v_{11} + v_{21}v_{21} = 1, \quad v_{11}v_{11} + v_{12}v_{12} = 1 \\ v_{22}v_{22} + v_{21}v_{21} = 1, \quad v_{22}v_{22} + v_{12}v_{12} = 1 \\ v_{11}v_{12} + v_{21}v_{22} = 0, \quad v_{11}v_{21} + v_{12}v_{22} = 0. \end{aligned} \tag{E.4}$$

The use of the equations in (E.4) simplifies (E.2) to

$$x(\lambda) = \frac{\sigma_1^4(\sigma_2^2+\lambda)^2v_{11}v_{21} + \sigma_2^4(\sigma_1^2+\lambda)^2v_{12}v_{22}}{\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{11}^2+\sigma_2^4(\sigma_1^2+\lambda)^2v_{12}^2} \cdot \sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{21}^2+\sigma_2^4(\sigma_1^2+\lambda)^2v_{22}^2}} \tag{E.5}$$

$$x(\lambda) \coloneqq \frac{g(\lambda)}{h(\lambda)}, \tag{E.6}$$

$$\text{where } g(\lambda) = \sigma_1^4(\sigma_2^2+\lambda)^2v_{11}v_{21} + \sigma_2^4(\sigma_1^2+\lambda)^2v_{12}v_{22} \tag{E.7}$$

$$\text{and } h(\lambda) = \sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{11}^2+\sigma_2^4(\sigma_1^2+\lambda)^2v_{12}^2} \cdot \sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{21}^2+\sigma_2^4(\sigma_1^2+\lambda)^2v_{22}^2}. \tag{E.8}$$

We show that the absolute value of this off-diagonal element, i.e. the mutual coherence, increases when λ increases. The derivative with respect to λ is

$$x'(\lambda) = \frac{h(\lambda)g'(\lambda) - h'(\lambda)g(\lambda)}{h^2(\lambda)}$$

$$\begin{aligned}
\text{where } h(\lambda)g'(\lambda) = &\frac{2\sigma_1^{12}(\sigma_2^2+\lambda)^5 v_{11}v_{21} + 2\sigma_1^4\sigma_2^8(\sigma_1^2+\lambda)^4(\sigma_2^2+\lambda)v_{11}v_{12}^2v_{21}v_{22}^2}{\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{11}^2 + \sigma_2^4(\sigma_1^2+\lambda)^2v_{12}^2}\cdot\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{21}^2 + \sigma_2^4(\sigma_1^2+\lambda)^2v_{22}^2}} \\
&+ \frac{2\sigma_1^8\sigma_2^4(\sigma_1^2+\lambda)^2(\sigma_2^2+\lambda)^3 v_{11}v_{21}(v_{12}^2v_{21}^2 + v_{11}^2v_{22}^2) + 2\sigma_2^{12}(\sigma_1^2+\lambda)^5 v_{12}^3v_{22}^3}{\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{11}^2 + \sigma_2^4(\sigma_1^2+\lambda)^2v_{12}^2}\cdot\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{21}^2 + \sigma_2^4(\sigma_1^2+\lambda)^2v_{22}^2}} \\
&+ \frac{2\sigma_1^8\sigma_2^4(\sigma_1^2+\lambda)(\sigma_2^2+\lambda)^4 v_{11}^2v_{12}v_{21}^2v_{22}}{\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{11}^2 + \sigma_2^4(\sigma_1^2+\lambda)^2v_{12}^2}\cdot\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{21}^2 + \sigma_2^4(\sigma_1^2+\lambda)^2v_{22}^2}} \\
&+ \frac{2\sigma_1^4\sigma_2^8(\sigma_1^2+\lambda)^3(\sigma_2^2+\lambda)^2 v_{12}v_{22}(v_{12}^2v_{21}^2 + v_{11}^2v_{22}^2)}{\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{11}^2 + \sigma_2^4(\sigma_1^2+\lambda)^2v_{12}^2}\cdot\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{21}^2 + \sigma_2^4(\sigma_1^2+\lambda)^2v_{22}^2}}
\end{aligned}$$

$$\begin{aligned}
\text{and } h'(\lambda)g(\lambda) = &\frac{2\sigma_1^{12}(\sigma_2^2+\lambda)^5 v_{11}v_{21} + 2\sigma_1^4\sigma_2^8(\sigma_1^2+\lambda)^2(\sigma_2^2+\lambda)^3 v_{11}v_{12}^2v_{21}v_{22}^2}{\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{11}^2 + \sigma_2^4(\sigma_1^2+\lambda)^2v_{12}^2}\cdot\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{21}^2 + \sigma_2^4(\sigma_1^2+\lambda)^2v_{22}^2}} \\
&+ \frac{\sigma_1^8\sigma_2^4(\sigma_1^2+\lambda)(\sigma_2^2+\lambda)^4 v_{11}v_{21}(v_{12}^2v_{21}^2 + v_{11}^2v_{22}^2) + 2\sigma_2^{12}(\sigma_1^2+\lambda)^5 v_{12}^3v_{22}^3}{\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{11}^2 + \sigma_2^4(\sigma_1^2+\lambda)^2v_{12}^2}\cdot\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{21}^2 + \sigma_2^4(\sigma_1^2+\lambda)^2v_{22}^2}} \\
&+ \frac{\sigma_1^8\sigma_2^4(\sigma_1^2+\lambda)^2(\sigma_2^2+\lambda)^3 v_{11}^2v_{21}(v_{12}^2v_{21}^2 + v_{11}^2v_{22}^2)}{\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{11}^2 + \sigma_2^4(\sigma_1^2+\lambda)^2v_{12}^2}\cdot\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{21}^2 + \sigma_2^4(\sigma_1^2+\lambda)^2v_{22}^2}} \\
&+ \frac{\sigma_1^4\sigma_2^8(\sigma_1^2+\lambda)^3(\sigma_2^2+\lambda)^2 v_{12}v_{22}(v_{12}^2v_{21}^2 + v_{11}^2v_{22}^2)}{\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{11}^2 + \sigma_2^4(\sigma_1^2+\lambda)^2v_{12}^2}\cdot\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{21}^2 + \sigma_2^4(\sigma_1^2+\lambda)^2v_{22}^2}} \\
&+ \frac{\sigma_1^4\sigma_2^8(\sigma_1^2+\lambda)^4(\sigma_2^2+\lambda) v_{12}v_{22}(v_{12}^2v_{21}^2 + v_{11}^2v_{22}^2)}{\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{11}^2 + \sigma_2^4(\sigma_1^2+\lambda)^2v_{12}^2}\cdot\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{21}^2 + \sigma_2^4(\sigma_1^2+\lambda)^2v_{22}^2}} \\
&+ \frac{2\sigma_1^8\sigma_2^4(\sigma_1^2+\lambda)^2(\sigma_2^2+\lambda)^3 v_{11}^2v_{12}v_{21}^2v_{22}}{\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{11}^2 + \sigma_2^4(\sigma_1^2+\lambda)^2v_{12}^2}\cdot\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{21}^2 + \sigma_2^4(\sigma_1^2+\lambda)^2v_{22}^2}}.
\end{aligned}$$

This means that $h(\lambda)g'(\lambda) - h'(\lambda)g(\lambda)$ can be written as

$$\begin{aligned}
&\frac{\sigma_1^4\sigma_2^8((\sigma_1^2+\lambda)^3(\sigma_2^2+\lambda)^2 - (\sigma_1^2+\lambda)^4(\sigma_2^2+\lambda))v_{12}v_{22}(v_{12}^2v_{21}^2 + v_{11}^2v_{22}^2)}{\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{11}^2 + \sigma_2^4(\sigma_1^2+\lambda)^2v_{12}^2}\cdot\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{21}^2 + \sigma_2^4(\sigma_1^2+\lambda)^2v_{22}^2}} \\
&+ \frac{\sigma_1^8\sigma_2^4((\sigma_1^2+\lambda)^2(\sigma_2^2+\lambda)^3 - (\sigma_1^2+\lambda)(\sigma_2^2+\lambda)^4)v_{11}v_{21}(v_{12}^2v_{21}^2 + v_{11}^2v_{22}^2)}{\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{11}^2 + \sigma_2^4(\sigma_1^2+\lambda)^2v_{12}^2}\cdot\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{21}^2 + \sigma_2^4(\sigma_1^2+\lambda)^2v_{22}^2}} \\
&+ \frac{2\sigma_1^4\sigma_2^8((\sigma_1^2+\lambda)^4(\sigma_2^2+\lambda) - (\sigma_1^2+\lambda)^3(\sigma_2^2+\lambda)^2)v_{11}v_{12}^2v_{21}v_{22}^2}{\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{11}^2 + \sigma_2^4(\sigma_1^2+\lambda)^2v_{12}^2}\cdot\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{21}^2 + \sigma_2^4(\sigma_1^2+\lambda)^2v_{22}^2}} \\
&- \frac{2\sigma_1^8\sigma_2^4((\sigma_1^2+\lambda)(\sigma_2^2+\lambda)^4 - (\sigma_1^2+\lambda)^2(\sigma_2^2+\lambda)^3)v_{11}^2v_{12}v_{21}v_{22}}{\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{11}^2 + \sigma_2^4(\sigma_1^2+\lambda)^2v_{12}^2}\cdot\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2v_{21}^2 + \sigma_2^4(\sigma_1^2+\lambda)^2v_{22}^2}},
\end{aligned} \tag{E.9}$$

where it is clear that all denumerators of (E.9) are equal and strictly positive for $\lambda > 0$. Since the columns of V have unit length and the determinant of an unitairy matrix is ± 1, it is clear that $v_{11} = \pm v_{22}$, $v_{21} = \pm v_{12}$ and $v_{12}v_{22}$ should have an opposite sign compared to that of $v_{11}v_{21}$. First it is assumed that v_{11} and v_{21} have the same sign, which means that $v_{12}v_{22} < 0$.

Since $\sigma_1 > \sigma_2$ it is clear $(\sigma_1^2+\lambda)^3(\sigma_2^2+\lambda)^2 - (\sigma_1^2+\lambda)^4(\sigma_2^2+\lambda) < 0$ and therefore the first line of (E.9) is strictly positive for $\lambda > 0$. By counting the powers of $(\sigma_1^2+\lambda)$ and $(\sigma_2^2+\lambda)$, one can conclude that all other lines of (E.9) are strictly positive for $\lambda > 0$. This means that the derivative of the mutual coherence with respect to λ is strictly positive for $\lambda > 0$ when v_{11} and v_{21} have the same sign. Since v_{11} and v_{21} have the same sign and $v_{11}v_{21} = -v_{12}v_{22}$, it is clear that $x(\lambda)$ in (E.5) is strictly positive for $\lambda > 0$. Therefore, the mutual coherence always increases when λ expands.

In the opposite case it is assumed that v_{11} and v_{21} have an opposite sign. Following the same reasoning as above it is clear that v_{12} and v_{22} have the same sign. Since $\sigma_1 > \sigma_2$ it can be concluded that the sum of the first two lines of (E.9) is strictly negative for $\lambda > 0$. The same holds when the third and fourth line are summed and obviously the last two of (E.9). This means that the derivative of the mutual coherence with respect to λ is strictly negative for $\lambda > 0$. Furthermore, $x(\lambda)$ in (E.5) is also strictly negative for $\lambda > 0$. Since the mutual coherence is equal to the absolute value of the off-diagonal entry $x(\lambda)$, we can conclude that the mutual coherence is also increasing when v_{11} and v_{21} have an opposite sign for an increasing λ. □

E.2 v_{11} and v_{21} having the same sign

The values of A determine whether v_{11} and v_{21} have the same sign. For a 2×2 matrix $A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$ the matrix V is defined as [1]:

$$V = \begin{bmatrix} v_{11} & v_{12} \\ v_{21} & v_{22} \end{bmatrix} = \begin{bmatrix} \frac{s_{11}}{|s_{11}|}\cos(\Phi) & -\frac{s_{22}}{|s_{22}|}\sin(\Phi) \\ \frac{s_{11}}{|s_{11}|}\sin(\Phi) & \frac{s_{22}}{|s_{22}|}\cos(\Phi) \end{bmatrix}, \tag{E.10}$$

$$\text{where } \Phi = \frac{1}{2}\arctan 2(2ab+2cd, a^2-b^2+c^2-d^2), \tag{E.11}$$

$$\theta = \frac{1}{2}\arctan 2(2ac+2bd, a^2+b^2-c^2-d^2), \tag{E.12}$$

$$\begin{aligned} s_{11} &= (ac_\theta + cs_\theta)c_\Phi + (bc_\theta + ds_\theta)s_\Phi, \\ c_\Phi &= \cos(\Phi), \\ s_\Phi &= \sin(\Phi), \\ c_\theta &= \cos(\theta), \\ s_\theta &= \sin(\theta), \end{aligned}$$

where

$$\arctan 2(y,x) = \begin{cases} \operatorname{sgn}(y)\cdot\arctan\left(\left|\frac{y}{x}\right|\right) & x > 0 \\ \operatorname{sgn}(y)\cdot\frac{\pi}{2} & x = 0,\ y \neq 0 \\ \text{undefined} & x = 0,\ y = 0 \\ \operatorname{sgn}(y)\cdot\left(\pi - \arctan\left(\left|\frac{y}{x}\right|\right)\right) & x < 0. \end{cases} \tag{E.13}$$

Due to equation (E.10), it can be concluded that v_{11} and v_{21} have a different sign when $\Phi < 0$. This can only happen when $y < 0$ in (E.13), which therefore implies that $2ab + 2cd < 0$ in (E.11). If $a_1 = \begin{bmatrix} a \\ c \end{bmatrix}$ is the first column vector of matrix A, then $2ab + 2cd < 0$ when the second

column vector of $a_2 = \begin{bmatrix} b \\ d \end{bmatrix}$ lies in the red half-plane illustrated in Figure E.2. It does not matter in which quadrant a_1 lies, the red region is always the opposite half-plane with a_1 as the normal vector.

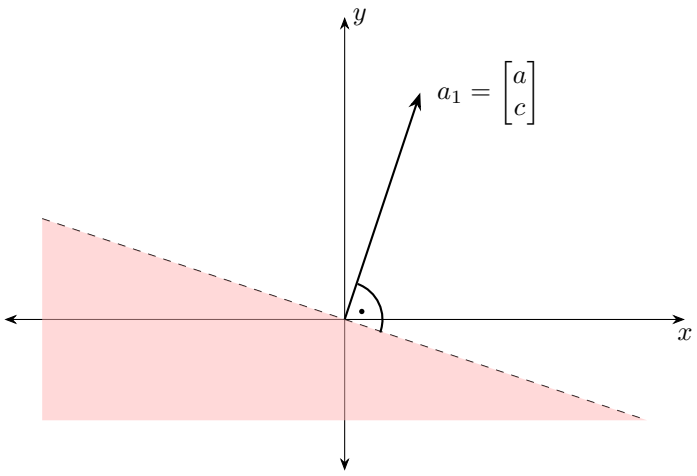


Figure E.1: *Illustration of the region (in red) for a_2 to lie in such that v_{11} and v_{21} have a different sign.*

E.3 Regularization and mutual coherence: a geometrical perspective

In this section the influence of regularization is given from a geometrical perspective. This explanation will give an intuition on why the effect explained in section E.1 holds for bigger (close to) ill-posed matrices, which are of particular interest in this thesis. It is clear that

$$P_\lambda A = \begin{bmatrix} \frac{\sigma_1^2(\sigma_2^2+\lambda)v_{11}^2+\sigma_2^2(\sigma_1^2+\lambda)v_{12}^2}{(\sigma_1^2+\lambda)(\sigma_2^2+\lambda)} & \frac{\sigma_1^2(\sigma_2^2+\lambda)v_{11}v_{21}+\sigma_2^2(\sigma_1^2+\lambda)v_{12}v_{22}}{(\sigma_1^2+\lambda)(\sigma_2^2+\lambda)} \\ \frac{\sigma_1^2(\sigma_2^2+\lambda)v_{11}v_{21}+\sigma_2^2(\sigma_1^2+\lambda)v_{12}v_{22}}{(\sigma_1^2+\lambda)(\sigma_2^2+\lambda)} & \frac{\sigma_1^2(\sigma_2^2+\lambda)v_{21}^2+\sigma_2^2(\sigma_1^2+\lambda)v_{22}^2}{(\sigma_1^2+\lambda)(\sigma_2^2+\lambda)} \end{bmatrix}. \tag{E.14}$$

Since the definition of the mutual coherence is

$$\mu(A) = \max_{i \neq j} \frac{\left|\langle a_i, a_j \rangle\right|}{\|a_i\|_2 \|a_j\|_2} \tag{E.15}$$

and in case of a 2×2 matrix equation (E.15) is equal to

$$\left|\cos(\theta)\right| = \max_{i \neq j} \frac{\left|a_i \cdot a_j\right|}{\|a_i\|_2 \|a_j\|_2} \tag{E.16}$$

where θ is the angle between the vectors a_i and a_j. Now we show that $\left|\cos(\theta)\right|$ goes to 1 when λ increases in $P_\lambda A$ in case when A is (close to) an ill-posed forward mapping. From equation

(E.14) and (E.16), it can be concluded that

$$\begin{aligned}\cos(\theta) &= \max_{i\neq j} \frac{a_i \cdot a_j}{\|a_i\|_2 \|a_j\|_2} \\ &= \frac{\sigma_1^4(\sigma_2^2+\lambda)^2 v_{11}v_{21} + \sigma_2^4(\sigma_1^2+\lambda)^2 v_{12}v_{22}}{\sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2 v_{11}^2 + \sigma_2^4(\sigma_1^2+\lambda)^2 v_{12}^2} \cdot \sqrt{\sigma_1^4(\sigma_2^2+\lambda)^2 v_{21}^2 + \sigma_2^4(\sigma_1^2+\lambda)^2 v_{22}^2}},\end{aligned} \tag{E.17}$$

which is the same expression as (E.5). Then it holds that

$$\begin{aligned}\lim_{\lambda\to\infty} \cos(\theta) &= \frac{\lambda^2\sigma_1^4 v_{11}v_{21} + \lambda^2\sigma_2^4 v_{12}v_{22}}{\sqrt{\lambda^2\sigma_1^4 v_{11}^2 + \lambda^2\sigma_2^4 v_{12}^2} \cdot \sqrt{\lambda^2\sigma_1^4 v_{21}^2 + \lambda^2\sigma_2^4 v_{22}^2}}, \\ &= \frac{\sigma_1^4 v_{11}v_{21} + \sigma_2^4 v_{12}v_{22}}{\sqrt{\sigma_1^4 v_{11}^2 + \sigma_2^4 v_{12}^2} \cdot \sqrt{\sigma_1^4 v_{21}^2 + \sigma_2^4 v_{22}^2}}.\end{aligned} \tag{E.18}$$

Since the forward mapping A is considered to be (close to) ill-posed, it is clear that certain columns of that forward mapping are almost aligned. When columns are close to being aligned the condition number of the matrix is big, which means that there is a large gap between the biggest and smallest singular value. This means that

$$\lim_{\lambda\to\infty} \left|\cos(\theta)\right| = \left| \frac{\sigma_1^4 v_{11}v_{21}}{\sqrt{\sigma_1^4 v_{11}^2} \cdot \sqrt{\sigma_1^4 v_{21}^2}} \right| = 1 \tag{E.19}$$

and therefore $\theta \to 0$ or $\theta \to \pi$ when $\lambda \to \infty$. This means that columns that are close to being aligned will get more aligned when λ increases. Therefore, the mutual coherence increases when λ goes up.

When larger $n \times n$ (close to) ill-posed forward mappings are considered, it holds that

$$P_\lambda A = \begin{bmatrix} | & & | \\ v_1 & \dots & v_n \\ | & & | \end{bmatrix} \begin{bmatrix} — & \frac{\sigma_1^2}{\sigma_1^2+\lambda} v_1 & — \\ & \vdots & \\ — & \frac{\sigma_n^2}{\sigma_n^2+\lambda} v_n & — \end{bmatrix}. \tag{E.20}$$

Due to linearity it is expected that regularization has the same effect on $P_\lambda A$ as in the case of a 2×2 matrix. For rectangular $m \times n$ or $n \times m$ forward mappings, the $P_\lambda A$ has the same form as presented in (E.20), but then with dimensions $n \times n$ and $m \times m$, respectively. Therefore, due to linearity, the same growth of the mutual coherence is also expected here. For square rank deficient forward mappings that are close to being ill-posed, where the rank $= r < n$, it holds that

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

$$P_\lambda = V_r S U_r^T,$$

where $\Sigma = \text{diag}(\sigma_1, \dots, \sigma_r, \underbrace{0, \dots, 0}_{n-r})$, $S = \text{diag}(\frac{\sigma_1}{\sigma_1^2+\lambda}, \dots, \frac{\sigma_r}{\sigma_r^2+\lambda})$, V_r consists of the first r columns

of V and U_r consists of the first r columns of U. Then

$$\begin{aligned} P_\lambda A &= V_r S U_r^T U \Sigma V^T \\ &= V_r \left[\begin{array}{ccc|c} \frac{\sigma_1^2}{\sigma_1^2+\lambda} & & & \\ & \ddots & & \mathbf{0} \\ & & \frac{\sigma_r^2}{\sigma_r^2+\lambda} & \underbrace{\qquad}_{n-r} \end{array}\right] V^T \\ &= \begin{bmatrix} | & & | \\ v_1 & \dots & v_r \\ | & & | \end{bmatrix} \begin{bmatrix} — & \frac{\sigma_1^2}{\sigma_1^2+\lambda} v_1 & — \\ & \vdots & \\ — & \frac{\sigma_r^2}{\sigma_r^2+\lambda} v_r & — \end{bmatrix}, \end{aligned} \tag{E.21}$$

which is $r-$dimensional subspace of (E.20). Due to linearity it is expected that regularization has the same effect on the columns of A, leading to an increasing mutual coherence of $P_\lambda A$ for an expanding λ. For rank deficient rectangular matrices the same reasoning can be used. Also in this case it is expected that the mutual coherence rises when λ increases.

Appendix F

Including right preconditioners

A right preconditioner can add important prior information about x to the inverse problem. The effect of this addition can be shown via the Bayesian view on inverse problem. Then, according to section 2.4.2, the most likely solution is

$$\begin{aligned} x_{\text{map}} &= \arg\min_x \left\|P_1(y - Ax - e_0)\right\|^2 + \left\|P_2(x - x_0)\right\|^2, \\ &:= \arg\min_x V_2(x), \end{aligned}$$

which means that the optimal solution satisfies

$$\begin{aligned} &\nabla V_2(x) = -A^T P_1^T P_1(y - Ax - e_0) + P_2^T P_2(x - x_0) = 0, \\ &\text{thus } x_2^* = \left(A^T P_1^T P_1 A + P_2^T P_2\right)^{-1} \left(A^T P_1^T P_1(y - e_0) + P_2^T P_2 x_0\right). \end{aligned}$$

Therefore, the Hessian is equal to

$$\nabla^2 V_2(x) = A^T P_1^T P_1 A + P_2^T P_2, \tag{F.1}$$

which leads to

$$\pi_{\text{posterior}}(x|y) \sim \mathcal{N}\left(x_2^*, \left(A^T P_1^T P_1 A + P_2^T P_2\right)^{-1}\right). \tag{F.2}$$

Due to the positive semi-definiteness of $P_2^T P_2$ it is clear that the variance of the posterior is lower when a right preconditioner is included. Thus, the number of possible solutions decreases when also a right preconditioner is considered. When $A^T P_1^T P_1 A + P_2^T P_2$ is large, the mean of $\pi_{\text{posterior}}(x|y)$ possibly converges to its correct value. Furthermore, the posterior also suggests that when $A^T P_1^T P_1 A$ is close to zero, the distribution of the posterior is roughly equal to the prior distribution, which means that the measurements are ambiguous.