A Fast Algorithm for Ill-posed Linear Inverse Problems

F.F. Redel

Master of Science Thesis

Delft University of Technology

Delft Center for Systems and Control

A Fast Algorithm for Ill-posed Linear Inverse Problems

Master of Science Thesis

For the degree of Master of Science in Systems and Control at Delft University of Technology

F.F. Redel

November 2, 2022

Faculty of Mechanical, Maritime and Materials Engineering (3mE) · Delft University of Technology

Abstract

Ill-posed Linear Inverse Problems arise in various research domains, such as control engineering and image processing. Having a fast algorithm is a great benefit when working with high-dimensional signals, such as images. However, fast convergence and iterations with low computational complexity are challenging.

In this master thesis report, we propose an exact smooth reformulation of an ill-posed Linear Inverse Problem. Subsequently, we present a novel algorithm, the Fast Linear Inverse Problem Solver (FLIPS), associated with the new problem formulation. We show that in most metrics, the algorithm outperforms state-of-the-art methods like Chambolle-Pock (CP) and the Constrained Split Augmented Lagrangian Shrinkage Algorithm (C-SALSA) in terms of speed. Finally, associated with this algorithm, we present an open-source MATLAB package that includes the proposed algorithm and state-of-the-art methods.

Table of Contents

Acknowledgements **vii**

1 Introduction **1**

1-1 Ill-posed Linear Inverse Problems 1
1-2 Sparse Representation 1
1-2-1 The Sparse Coding Problem 2
1-3 Dictionary Learning 3
1-3-1 Dictionary Learning Algorithms 4
1-4 Applications 5
1-4-1 Sparse Coding Problem Applications 5
1-4-2 Dictionary Learning Applications 6
1-5 Preliminaries 6
1-6 Notations 7

2 Problem Description **9**

2-1 Problem Definition 9
2-1-1 Objective Master Thesis 9
2-1-2 Problem Formulation 9
2-2 State-of-the-art 10
2-2-1 C-SALSA 10
2-2-2 Chambolle-Pock 11
2-2-3 Fast Iterative Shrinkage-Thresholding Algorithm 12
2-3 Contribution 12

3 Problem Reformulation **15**

3-1 Smooth Convex Optimization Problem 15
3-1-1 Assumptions and Definitions 15
3-1-2 Exact Non-smooth Reformulation 16
3-1-3 Exact Smooth Reformulation 17

4 Fast Linear Inverse Problem Solver **19**
4-1 General Form 19
4-1-1 Descent Directions 20
4-1-2 Step Size 21
4-1-3 Stopping Condition 22
4-1-4 Remarks Fast Linear Inverse Problem Solver Algorithm 22

5 Numerical Results **23**
5-1 Experiment Details 24
5-1-1 Open-source Package 24
5-1-2 Initial and Optimal Solution 24
5-1-3 Image Preprocessing 24
5-2 Tuning Experiments 25
5-2-1 Tuning CP 25
5-2-2 Tuning C-SALSA 26
5-2-3 Tuning FLIPS 28
5-2-4 Validation Reconstruction Error 30
5-3 Experiments 31
5-3-1 Full Image Experiment 31
5-3-2 Comparison with State-of-the-art 32
5-3-3 Dictionary Learning 37

6 Conclusion **41**

7 Discussion and Recommendations **43**
7-1 Discussion 43
7-1-1 DCT-Dictionary 43
7-1-2 Linear Oracle 44
7-1-3 Projected Gradient Descent 44
7-1-4 Blurry Image Recovery 128x128 Experiment 45
7-1-5 Limiting Storage Experiments 45
7-1-6 Limiting Signal Size 46
7-1-7 Visual Explanation Denoising Problem 46
7-1-8 Visual Explanation Inpainting Problem 47
7-2 Recommendations 48
7-2-1 Large Inpainting Experiment 48
7-2-2 Application To Other Problems 48
7-2-3 Convergence Rate 49
7-2-4 Parfor Loop 49

A Proofs 51
A-1 Proofs for Non-smooth Reformulation 51
A-2 Proofs for Smoothness 54
A-3 Proofs for Stepsize Selection 57

Bibliography 59

Glossary 63
List of Acronyms 63

Acknowledgements

First and foremost, I thank my daily supervisor dr. M.R. Sheriff for his guidance and help during the entire thesis process. He pushed me to my limits and ensured I understood every piece of the work. Furthermore, I would like to thank my supervisor dr. P. Mohajerin Esfahani for his assistance with difficult problems and his enthusiasm during all our meetings. He inspired me to aim for the best and challenged me every time. Finally, I would like to thank all my family members, friends, roommates, and boyfriend for being patient with me during frustrating Matlab errors and days of problem solving.

Delft, University of Technology
November 2, 2022

F.F. Redel

Chapter 1

Introduction

In this introduction, the background information of this report is presented. First, the ill-posed Linear Inverse Problem is introduced, followed by the importance of sparse representations and the associating Sparse Coding problem. Secondly, Dictionary Learning (DL) is presented, in which the Sparse Coding problem is incorporated. Subsequently, the applications of the Sparse Coding problem and Dictionary Learning are described. The final sections elaborate on the preliminaries and notations of this report.

1-1 Ill-posed Linear Inverse Problems

A Linear Inverse Problem (LIP) can arise in many applications, such as machine learning or signal processing [21]. When solving an LIP, one tries to recover the original signal given limited or noisy measurements. In common practice, this results in ill-posed LIPs [12]. The problem is ill-posed since it is highly sensitive to changes in the data, which directly highlights the difficulty of these types of problems. To solve ill-posed LIPs, some additional information or regularization is required. For example, one characteristic that suffices is that natural signals are known to admit a sparse representation under an appropriate basis, such as the Fourier Transform.

1-2 Sparse Representation

Natural signals, like audio signals or images, tend to have low-dimensional characteristics even though the signals themselves could be of very high dimension [47]. *Sparsity* is a beneficial low-dimensional feature that has shown tremendous utility lately. It has been observed that natural signals if represented using an appropriate basis, have non-zero coefficients corresponding to very few basis elements, i.e., they admit a *sparse representation*. For example, it is observed that only 5-10 percent of the so-called Discrete Cosine Transform (DCT) coefficients of natural images are non-zero [31]. The presence of sparsity has lately resulted in many novel signal processing methods with a wide range of applications like denoising, in which noise is removed from signals, and Compressed Sensing (CS), in which high-dimensional signals are compressed to low-dimensional representations [22], [17].

A sparse representation is a representation in which the signal x admits a sparse linear correlation of the dictionary D and the sparse representation vector f. The linear correlation can be described as

$$x = Df. \tag{1-1}$$

In Wright et al. [47], the base elements are the training samples of the data set stored as columns in 'D', but other elements can be chosen or learned as well.

CS deserves special mentioning since it pushed sparse representations to the forefront. Sparsity plays an important role in reducing the number of measurements needed to recover a high-dimensional signal, which defines CS [25]. The relation between the high-dimensional signal $x \in \mathbb{R}^{n_x}$, the measurement matrix $C \in \mathbb{R}^{n_y \times n_x}$, and the measurements $y \in \mathbb{R}^{n_y}$ with $n_y \ll n_x$ can be denoted as

$$y = Cx. \tag{1-2}$$

The recovery of the signal x is an undetermined problem since C has fewer rows than columns and is therefore non-invertible. However, since the signal x is sparse under a given basis D, the measurements can uniquely determine x under mild conditions. This relation is described as

$$y = CDf. \tag{1-3}$$

Solving the optimization problem for f is called the Sparse Coding Problem.

1-2-1 The Sparse Coding Problem

The Sparse Coding problem refers to finding a sparse representation f that satisfies $x = Df$, for a given dictionary D. This is typically done by solving an optimization problem whose objective function is known to prioritise sparse solutions. One of the most common approaches is to solve the regularized formulation [32]

$$\min_{f \in \mathbb{R}^{n_f}} \frac{1}{2} \|x - Df\|^2 + \lambda \|f\|_1 . \tag{1-4}$$

The term $\|x - Df\|^2$ denotes the minimization of the reconstruction error using the ℓ_2-norm, which enforces the representation to fit the original signal. The second term $\|f\|_1$ enforces sparsity using the ℓ_1-norm. The regularization parameter λ weighs the trade-off between sparsity and the reconstruction error. Instead of the ℓ_1-norm, the ℓ_0-norm could be applied to enforce sparsity since it counts the number of non-zero elements. However, using the ℓ_0-norm results in a non-convex optimization problem, whereas the ℓ_1-norm gives rise to a convex problem. Under mild conditions, the ℓ_1-norm also enforces sparsity and recovers the correct solution [25, 18]. Thus, the ℓ_1-norm is a preferred choice in practice.

In many image processing problems, like image denoising and CS, it is useful to solve

$$\min_{f} \|f\|_1 \quad \text{s.t.} \quad x = CDf, \tag{1-5}$$

which is commonly referred to as *Basis Pursuit* [2]. If the measurements are noisy, then (1-5) is modified as

$$\min_{f} \|f\|_1 \quad \text{s.t.} \quad \|x - Df\|^2 \leqslant \epsilon, \tag{1-6}$$

where $\epsilon > 0$ is a parameter to be selected depending on the measurement noise [47, 2]. This problem is referred to as *Basis Pursuit Denoising (BPDN)*.

One can apply suitable techniques to solve the basis pursuit problem (1-5) such as the Orthogonal Matching Pursuit (OMP). The greedy fashion of this algorithm is appropriate for high-dimensional signals [46]. 'Greedy' refers to the fact that the algorithm makes local optimal decisions but often does not converge to the optimal solution. For each iteration in the algorithm, the dictionary column correlated the strongest to the remaining measurement vector is chosen. Subsequently, the contribution to the measurement vector is subtracted and the next iteration will start. After m iterations, there will be a small or no residual $\|x - Df\|$ left. In Chapter 2, more examples of Sparse Coding algorithms will be described, i.e., Chambolle-Pock and C-SALSA.

1-3 Dictionary Learning

In CS, it is assumed that the signal x is sparse in some known (and also fixed) dictionary D as a basis for the sparse representation. In that case, D is referred to as an analytical dictionary. Some common examples of analytical dictionaries include classical dictionaries such as Fourier, wavelets, countourlets, and curvelets [40]. The use of an analytical dictionary has the advantage of a fast implementation [35]. However, the dictionary to be used as a parameter in (1-4) can also be learned so that it is data-dependent. A data-dependent dictionary is more likely to produce a sparser representation compared to analytical dictionaries. The task of learning a dictionary from the data is known as *Dictionary Learning (DL)* [38]. However, the downside of learned dictionaries is that they must be learned from the data, thus adding additional computation cost. Combinations of analytical and learned dictionaries have been tried, for example, by Ophir et al. [35], and Rubinstein et al. [40].

In DL, $x_i \in \mathbb{R}^{n_x}$ for $i \in \{1, 2, \ldots, N\}$ are the signals and $D = [d_1 d_2 \ldots d_{n_f}] \in \mathbb{R}^{n_x \times n_f}$ is the dictionary to be learned from x_i so that $f_i \in \mathbb{R}^{n_f}$ is sparse for every signal

$$x_i = Df_i. \tag{1-7}$$

The base elements are usually referred to as 'atoms' and are the columns of the dictionary D, denoted as d_i [47]. Informally speaking, the dictionary can be seen as a book with words (atoms), with the sparse representation of a signal being a selection of a few words. These dictionaries D are typically 'overcomplete' ($n_x < n_f$), meaning that there are more atoms than the dimension of the signals, which covers a broader range of signal phenomena compared to undercomplete dictionaries [38]. Also, images represented with overcomplete dictionaries have higher a compression, and higher accuracy [29]. The problem of representing a signal x_i with an overcomplete dictionary D is equivalent to solving an underdetermined inverse problem $Df_i = x_i$ which does not have a unique solution for any x_i.

The goal in DL is to learn a dictionary D together with the sparse representation $(f_i)_i$ such that the reconstruction error $\|x_i - Df_i\|$ of the signal x_i is minimal [29]. Note that this is not the same as CS, where the goal is to find a sparse representation f_i given a fixed dictionary D. DL is typically performed by solving the optimization problem with the regularized formulation [32]

$$\min_{D\in\mathcal{D},(f_i)_i} \frac{1}{N}\sum_{i=1}^{N}(\frac{1}{2}\|x_i - Df_i\|^2 + \lambda\|f_i\|_1). \tag{1-8}$$

To avoid the entries of D resulting in arbitrarily large coefficients and, as a consequence, Problem (1-8) resulting in an ill-posed problem, the dictionary columns are usually constrained to have an ℓ_2-norm less than one. In other words, the set $\mathcal{D}$ of feasible dictionaries is

$$\mathcal{D} \triangleq \left\{D \in \mathbb{R}^{n_x\times n_f} \text{ s.t. } \forall j = 1,\dots,n_f, \|d_j\| \leqslant 1\right\}. \tag{1-9}$$

The DL problem can be formulated in multiple ways besides the problem presented in (1-8). The set $X = [x_1 x_2 \dots x_N] \in \mathbb{R}^{n_x\times N}$ is a collection of signals $x_i \in \mathbb{R}^{n_x}$ which results in another commonly used description when the sparsity degree p is known [44]:

$$\min_{D\in\mathcal{D},f\in\mathbb{R}^{n_f\times n_x}} \frac{1}{2}\|X - Df\|^2 \quad \text{s.t.} \quad \|f\|_1 \leqslant p. \tag{1-10}$$

In case the maximum reconstruction error ϵ is the known limiting factor, the following optimization problem could be used

$$\min_{D\in\mathcal{D},f\in\mathbb{R}^{n_f\times n_x}} \|f\|_1 \quad \text{s.t.} \quad \|X - Df\|^2 \leqslant \epsilon. \tag{1-11}$$

The DL problems described in Equation (1-8), Equation (1-10) and Equation (1-11) are non-convex optimization problems due to the bi-linear product Df_i for each i. However, since the optimization problems in $(f_i)_i$ keeping D fixed is convex (and vice versa), we can iteratively solve the problems by optimizing over one variable while keeping the other one fixed. The optimization over the sparse representation vectors $(f_i)_i$ results in solving the Sparse Coding problem for each i. Whereas, the optimization over the dictionary D is a quadratic program and is usually referred to as the dictionary update step.

1-3-1 Dictionary Learning Algorithms

The DL algorithm aims to find an optimal learned dictionary D and a sparse representation $(f_i)_i$ as described in Equation (1-8). The iterative approach that is commonly used in DL algorithms alternates between updating the dictionary D and the sparse representation $(f_i)_i$ while keeping the other one fixed, denoted as [42]

$$\begin{cases} \min\limits_{f_i} & \frac{1}{N}\|f_i\|_1 + \lambda\|x_i - Df_i\|^2, \quad \forall i \in 1,2,\dots N \text{ and} \\ \min\limits_{D\in\mathcal{D}} & \frac{1}{N}\sum_{i=1}^{N}\|x_i - Df_i\|^2. \end{cases} \tag{1-12}$$

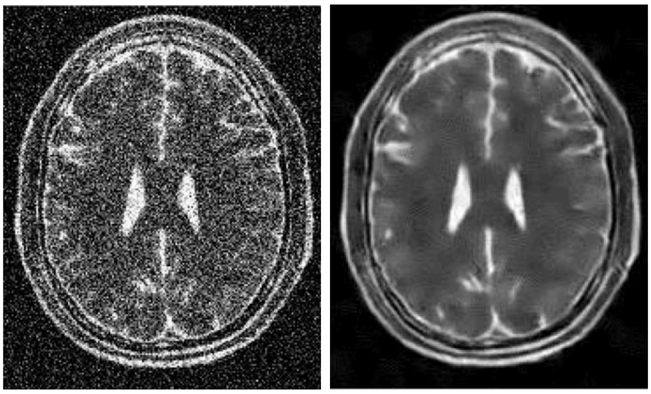
The Sparse Coding problem, described in the first line in (1-12), is the most time-consuming part of this iterative approach [33, 38]. Therefore, if a fast DL algorithm is desired, one should aim to minimize the computational complexity of the Sparse Coding stage. One algorithm that has been commonly used is the K-SVD algorithm, a method based on generalized K-means [2].

K-SVD is a dictionary learning algorithm that iterates between sparse coding the training samples on the current dictionary and updating the dictionary atoms to get the best fit [2]. The update of the atoms is combined with the update of the sparse representation, which speeds up the process. The algorithm applies the Singular-Value-Decomposition (SVD) process to compute the atom update step. The update step is done atom-by-atom rather than using an inefficient matrix inverse which is the case in several other algorithms. A downside of the K-SVD algorithm is that it can get caught in local minima due to the non-convexity. However, there are tricks to decrease the chance, for example, removing the least used atom every ten iterations. Many extensions to this algorithm have been proposed in the literature. For example, Zhang et al. propose D-KSVD in which the algorithm has an extra classification error in the objective function resulting in an algorithm applicable to face recognition [51]. An overview of more DL methods can be found in [38].

1-4 Applications

1-4-1 Sparse Coding Problem Applications

Applying the Sparse Coding problem to medical image processing techniques highlights the importance of Sparse Coding algorithms. Figure 1-1 shows an example of denoising medical MRI images by Sparse Coding [30]. In the recovered images, it is easier to distinguish brain structures than in noisy images. Note that before applying the Sparse Coding problem to medical images, further research needs to be done.



(a) Noisy MRI-image as input for the denoising problem [30].

(b) Recovered MRI-image as output of the denoising problem [30].

Figure 1-1: Result for the Sparse Coding problem applied to a noisy MRI-image.

Furthermore, in many other domains, Compressed Sensing can be applied to recover high-dimensional data from low-dimensional measurements, for example in speech and audio signals [14], rapid MR imaging [31], ECG signals [37] and extracting impulse components in engineering applications [13].

1-4-2 Dictionary Learning Applications

DL can be applied to many signal reconstruction problems and is typically used in computational image processing. One of the most important fields is the field of medical imaging. For medical purposes, it is a commonly encountered difficulty to extract noise-free high-dimensional data. Therefore, many problems arise when trying to improve these types of images. Note that these problems are also Sparse Coding problems when an analytical dictionary is chosen.

One application is the deblurring problem, where the task is to reconstruct the original image given its blurry version [15]. An example of a blurry image is a picture that is moved, i.e., the shutter speed was too long when the picture was shot, resulting in a blurry image. In that case, the non-blurry image can be retrieved by applying a deblurring algorithm. Note that in image processing, it is common to work with images divided into smaller image patches instead of the entire image at once to avoid high computational complexity. For local blurring kernels, small (and thus local) image patches suffice for reconstructing the original image. However, if a non-localized blurring kernel is considered, high-level features across the entire image should be captured as well. Therefore, large image patches are required for reconstructing images with non-local blurring kernels compared to local blurring kernels.

The advantage of working with large image patches also appears for the inpainting problem in which a missing part of an audio signal or image is reconstructed [45, 48]. Take for example a low-resolution image of a license plate shot by the police that is unreadable. The license plate code can be reconstructed by filling in the missing pixels. Small missing regions of the image can be reconstructed for small image patches. However, if large areas of the image are missing, the surrounding small image patches do not store the high-level features needed for reconstruction [44]. Therefore, large image patches have an advantage. One paper achieving good results for this problem is the paper of Sulam et al., where large areas of face images are reconstructed by using small images (100×100) as the input [43]. A visual explanation of the idea behind the inpainting problem is discussed in Section 7-1-8.

1-5 Preliminaries

Continuity and Smoothness

A *continuous function* is a function for which a continuous variation of the argument results in a continuous variation of the function value. For every value y and x in its domain, $f(y)$ and $f(x)$ are defined, and [24]

$$\lim_{x \to y} f(x) = f(y). \tag{1-13}$$

A continuously differentiable function f is *β-smooth* if the gradient ∇f is β-Lipschitz, that is if for all $x, y \in \mathcal{X}$,

$$\|\nabla f(y) - \nabla f(x)\| \leqslant \beta \|y - x\|. \tag{1-14}$$

If f is β-smooth, then for any $x, y \in \mathcal{X}$

$$|f(y) - f(x) - \langle \nabla f(x), y - x \rangle| \leqslant \frac{\beta}{2} \|y - x\|^2. \tag{1-15}$$

Smoothness is a stronger case than continuity. An example of a continuous but non-smooth function is $f(x) = |x|$. This function is non-smooth since it is non-differentiable at $x = 0$.

Convex Function

Consider convex set $\mathcal{X} \subset \mathbb{R}^d$. The function $f : \mathcal{X} \to \mathbb{R}$ is *convex* if [24]

$$f((1-\theta)x + \theta y) \leqslant (1-\theta)f(x) + \theta f(y), \ \forall x, y \in X, \forall \theta \in [0,1]. \tag{1-16}$$

Convex Conjugate

Consider function $f : \mathcal{X} \to \mathbb{R} \cup \{-\infty, +\infty\}$, then the *convex conjugate* function is defined by $f^* : \mathcal{X}^* \to \mathbb{R} \cup \{-\infty, +\infty\}$ and $f^*(x^*)$ is defined by [5]

$$f^*(x^*) := \sup \ \{\langle x^*, x\rangle - f(x) : x \in \mathcal{X}\}. \tag{1-17}$$

Proximal Operator

Consider a closed proper convex function $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$. The *proximal operator* $\mathrm{prox}_f : \mathbb{R}^n \to \mathbb{R}^n$ of f is defined by [36]

$$\mathrm{prox}_f(v) = \underset{x}{\mathrm{argmin}} \ \left(f(x) + \frac{1}{2}\|x - v\|_2^2\right). \tag{1-18}$$

PSNR

Peak-Signal-to-Noise Ratio (PSNR) is a well-known image quality metric. The PSNR between image f and g, both of size M×N, is given by [26]

$$PSNR(f,g) = 10\log_{10}\left(\frac{255^2}{\frac{1}{MN}\sum\limits_{i=1}^{M}\sum\limits_{j=1}^{N}(f_{i,j} - g_{i,j})^2}\right). \tag{1-19}$$

RMSE

The *Root Mean Square Error (RMSE)* is a commonly used metric to measure model performance and is defined by [9]

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x - x^*)^2}. \tag{1-20}$$

1-6 Notations

Standard notations are used throughout the report. Let n be a positive integer, $\mathbb{H}_n$ be an n-dimensional Hilbert space and $\|\cdot\|$ be the corresponding norm. The adjoint of a matrix M is denoted by M^a. The gradient of a continuously differentiable function $f(a)$ is denoted by $\nabla(f(a))$ and the interior of a set $\mathcal{K}$ as $\mathrm{int}(\mathcal{K})$.

Chapter 2

Problem Description

In this chapter, the research problem of this master thesis is presented. First, the problem is defined, and the mathematical problem formulation is given. Subsequently, the state-of-the-art is described presenting the current approaches for solving this problem. Finally, the contribution of this master thesis report is presented.

2-1 Problem Definition

2-1-1 Objective Master Thesis

It has many advantages to have a Sparse Coding algorithm that can handle large image patches, as described in Chapter 1 for non-local deblurring and inpainting of large missing pieces of images. Furthermore, it is desired to have a fast algorithm for problems such as Dictionary Learning (DL) where the Sparse Coding problem must be solved once every iteration for every patch independently. By fast algorithms, we try to minimize the total computation time. A product of two terms defines the total computation time; the number of iterations and the time each iteration takes. The number of iterations needed is related to the convergence rate; a higher convergence rate results in fewer iterations. The time each iteration takes is related to the image patch size. Bigger image patches result in a higher computational complexity and therefore take more time. Hence, if the algorithm's speed increases, it can handle bigger image patches within the same amount of time. *Therefore, the objective of this master thesis report is to find an algorithm that can solve an ill-posed Linear Inverse Problem (LIP) of the form* (2-1) *with less computation time than state-of-the-art methods.*

2-1-2 Problem Formulation

In the next chapters, we consider the optimization problem over atomic norm $c(f)$ (like the ℓ_1-norm and the nuclear norm) in which we try to reconstruct variable f given noisy linear measurements x, while allowing a reconstruction error ϵ

$$\begin{cases} \min\limits_{f \in \mathbb{R}^{n_f}} & c(f) \\ s.t. & \|x - \phi(f)\| \leqslant \epsilon. \end{cases} \tag{2-1}$$

Note that Problem (2-1) is not a Sparse Coding problem by definition, as presented in Chapter 1. It is a general formulation of ill-posed Linear Inverse Problems as constrained convex optimization. It is however the Sparse Coding problem denoted in Equation (1-4) for $c(f) = \|f\|_1$ and $\phi(f) = Df$. The problem is known as the Matrix Completion problem if $c(f) = \|f\|_*$ with $f \in \mathbb{R}^{n \times m}$, $\|.\|_*$ the nuclear norm and the constraint $X_{i,j} = f_{i,j}\ \forall i, j \in \mathcal{F}$ with $\mathcal{F}$ the set of observations [8]. Besides these two examples, this problem formulation has many more applications.

2-2 State-of-the-art

Observe that the ℓ_1-norm is not differentiable at every point, particularly around the sparse solutions of Problem (2-1). Therefore, the vanilla gradient descent method is not permitted. A common remedy to solve the issue of non-differentiability is to work with sub-gradients instead. However, the Sub-Gradient Descent (SGD) method for generic convex problems converges only with a rate of $\mathcal{O}(1/\sqrt{k})$ [7].

A few methods that represent the state-of-the-art for solving Problem (2-1) are highlighted. First, two algorithms are presented: Constrained Split Augmented Lagrangian Shrinkage Algorithm (C-SALSA) and the Chambolle-Pock (CP) algorithm. Subsequently, the Fast Iterative Shrinkage-Thresholding Algorithm (FISTA) is presented. FISTA is highlighted since it is one of the most commonly used algorithms in practise, and it is well-known for its fast convergence. FISTA solves the generalized Sparse Coding formulation described in Equation (1-4), instead of Equation (2-1). Therefore, FISTA can not be compared directly to C-SALSA and CP.

2-2-1 C-SALSA

Problem (2-1) can be rewritten into the following formulation by incorporating the constraint in the objective using an indicator function that penalizes a violation of the constraint

$$\min_{f \in \mathbb{H}} c(f) + \mathbb{1}_{B[x,\epsilon]}(\phi(f)), \tag{2-2}$$

where $\mathbb{1}_B(\phi(f))$ is the indicator function given by

$$\mathbb{1}_{B[x,\epsilon]}(\phi(f)) = \begin{cases} 0, & \text{if} \quad \phi(f) \in B[x, \epsilon] \\ +\infty, & \text{if} \quad \phi(f) \notin B[x, \epsilon]. \end{cases} \tag{2-3}$$

Here, B is the feasible closed set of the constraint of Problem (2-1).

Formulating Problem (2-1) as Problem (2-2) opens up the possibility to use C-SALSA [1]. C-SALSA is an algorithm in which the Alternating Direction Method of Multipliers (ADMM) is applied to Problem (2-2). For this problem, ADMM solves the LIP based on variable splitting using an Augmented Lagrangian Method (ALM). A new variable is created in variable splitting to get a separable objective while adding a new linear constraint. The new variable $u = \phi(f)$ reformulates the problem as

$$\begin{cases} \min\limits_{f\in\mathbb{H}, u\in\mathbb{H}} & c(f) + \mathbb{1}_{B[x,\epsilon]}(u) \\ \text{subject to} & u = \phi(f). \end{cases} \tag{2-4}$$

For easy proximal operators, Problem (2-4) may be easier to solve than the unconstrained formulation (2-2). The ALM iterates between optimizing the optimization variable and the Lagrange multipliers until it converges. The speed of C-SALSA relies heavily on ϕ. If ϕ satisfies $\phi^\top\phi = I$, further simplifications in the algorithm can be done, which improves its speed. The general algorithm with Lagrangian variable d is of the form

$$\begin{cases} (f_{k+1}) \in \arg\min\limits_{f,u} c(f) + \phi(u) + \frac{\mu}{2}\left\|\phi(f) - u - d_k\right\|_2^2 \\ (u_{k+1}) \in \arg\min\limits_{f,u} c(f) + \phi(u) + \frac{\mu}{2}\left\|\phi(f) - u - d_k\right\|_2^2 \\ d_{k+1} = d_k - \left(\phi(f)_{k+1} - u_{k+1}\right). \end{cases} \tag{2-5}$$

2-2-2 Chambolle-Pock

Consider the convex conjugate formulation described in Section 1-5 and observe that $(\mathbb{1}^*_{B[x,\epsilon]})^* = \mathbb{1}_{B[x,\epsilon]}$. Applied to Problem (2-2) and introducing dual variable u, we have

$$(\mathbb{1}_{B[x,\epsilon]}(u))^* = \langle x, u\rangle + \epsilon\|u\|_2, \tag{2-6}$$

which results in

$$\mathbb{1}_{B[x,\epsilon]}(\phi(f)) = \max_{u\in\mathbb{H}} \langle\phi(f), u\rangle - (\langle x, u\rangle + \epsilon\|u\|_2). \tag{2-7}$$

Incorporating Problem (2-7) in Problem (2-2) results in the following equivalent min-max reformulation

$$\min_{f\in\mathbb{H}} \max_{u\in\mathbb{H}} \; c(f) + \langle\phi(f), u\rangle - (\langle x, u\rangle + \epsilon\|u\|_2). \tag{2-8}$$

To apply CP, the mappings $f \to c(f)$ and $u \to \langle x\ ,\ u\rangle + \epsilon\|u\|_2$ must be proximal friendly, which is the case in many relevant problems like BPDN where $c(f) = \|f\|_1$ [20]. CP is a primal-dual algorithm where the primal variable f and the dual variable u are simultaneously updated at each iteration, as in

$$\begin{cases} f^+ = \arg\min\limits_{f} \; c(f) + \frac{1}{2\tau}\left\|f' - f\right\|^2 \\ \tilde{f} = f^+ + \theta(f^+ - f), \; \theta \in (0,1] \\ u^+ = \arg\min\limits_{u} \; \|u\| + \frac{1}{2\sigma}\left\|u' - u\right\|^2, \end{cases} \tag{2-9}$$

where the variables $u'(u,\phi)$ and $f'(f,\phi)$ are computed each iteration. The algorithm has an ergodic convergence rate of $\mathcal{O}(1/k)$, which improves upon the $\mathcal{O}(1/\sqrt{k})$ rate in Sub-gradient Descent algorithms, and is currently the best convergence guarantee that exists for Problem (2-1) [11].

2-2-3 Fast Iterative Shrinkage-Thresholding Algorithm

FISTA is a faster version of the earlier developed Iterative Shrinkage-Thresholding Algorithm (ISTA) [4]. This algorithm is widely used in the domain of image reconstruction. In ISTA and FISTA, problems of the following form are considered

$$\min_f F(f) + G(f), \tag{2-10}$$

in which $F(f)$ is convex and has an easy proximal operator, and $G(f)$ is convex, smooth, and continuously differentiable. For the Sparse Coding problem, this algorithm considers the regularized formulation (1-4) as written in Problem (2-10) with

$$F(f) = \|f\|_1 \text{ and } G(f) = \lambda\|x - Df\|_2^2. \tag{2-11}$$

In ISTA, the update of the optimization variable f is denoted as

$$f_{t+1} = \text{prox}_{\frac{\lambda}{\beta}\|\cdot\|_1}(f_t - \eta\nabla G(f_t)), \tag{2-12}$$

where $\nabla G(f_t)$ is the gradient of $G(f)$ at point f_t, η is the step size and $\text{prox}_{\frac{\lambda}{\beta}\|\cdot\|_1}(.)$ is the proximal operator. The proximal operator is, in this case, the shrinkage operator, after which the algorithms are named. In FISTA, a momentum step is added, which is of the form

$$\begin{cases} z_{t+1} &= \text{prox}_{\frac{\lambda}{\beta}\|\cdot\|_1}(f_t - \eta\nabla G(f_t)) \\ f_{t+1} &= (1-\gamma)z_{t+1} + \gamma z_t, \end{cases} \tag{2-13}$$

with γ as the momentum parameter. The algorithm is an extension of the classical gradient algorithm. ISTA is known for a slow convergence of the order $\mathcal{O}(1/k)$ while FISTA, using a momentum term, is achieving a convergence rate of $\mathcal{O}(1/k^2)$.

2-3 Contribution

The contribution of this master thesis report can be described in three main aspects.

1. We reformulate Problem (2-1) into a smooth convex min-max problem of which the maximization problem is solved algebraically. Therefore, this reformulation tackles the non-differentiability of Problem (2-1). Furthermore, we provide the associated proofs for smoothness and convexity. With this new formulation, new possibilities for solving Problem (2-1) arise.

2. We present a novel algorithm, the Fast Linear Inverse Problem Solver (FLIPS) for solving the newly formulated problem. A linear and a quadratic oracle are derived, presenting the descent direction, and a method to solve the optimal step size by applying exact line search. We show that in most metrics the algorithm outperforms the state-of-the-art methods like CP [10] and C-SALSA [1] in terms of convergence.

3. Finally, associated with this algorithm, we present an open-source MATLAB package that includes the proposed algorithm, CP, and C-SALSA[1]. Furthermore, this package presents the application to the image denoising problem, which serves as an example. Note that the package is placed on an open-source website after the associated paper is submitted.

[1] https://filesender.surf.nl/?s=download&token=dbd4c38c-06f0-4a0d-a1fb-ff53b92aacdf

Chapter 3

Problem Reformulation

This chapter elaborates on the exact reformulation of Problem (2-1) as a smooth convex optimization problem. First, we present the reformulation of Problem (2-1) into a non-smooth problem, after which the challenges and solutions regarding smoothness are presented. The proofs are directly retrieved from our paper that is soon to be published and therefore has yet to have a reference. The proofs will be discussed in Appendix A of this report.

3-1 Smooth Convex Optimization Problem

Problem (2-1), which is not differentiable at every point, will be reformulated into a smooth convex optimization problem. The reformulation will be elaborated on in three steps. Firstly, in Section 3-1-1, we introduce the assumptions and definitions of the problem reformulation. Secondly, we present the non-smooth problem reformulation, in Section 3-1-2. Thirdly, in Section 3-1-3, we present (in Theorem 3-1.1) the exact smooth reformulation of the original LIP (2-1) as a smooth convex problem.

3-1-1 Assumptions and Definitions

Let us assume that the original Problem (2-1) is feasible.

Assumption 3-1.1. *We shall assume throughout this report that* $\|x\| > \epsilon > 0$ *and that the corresponding LIP* (2-1) *is strictly feasible, i.e., there exists* $f \in \mathbb{H}$ *such that* $\|x - \phi(f)\| < \epsilon$.

To simplify the notation of the proofs, we introduce a new mapping $e : \mathbb{R}^{n_x} \longrightarrow [0, +\infty)$ defined as

$$e\,(h) := \min_{\theta \in \mathbb{R}} \|x - \theta\phi(h)\|^2 = \begin{cases} \|x\|^2 & \text{if } \phi(h) = 0, \\ \|x\|^2 - \dfrac{|\langle x\ ,\ \phi(h)\rangle|^2}{\|\phi(h)\|^2} & \text{if } \phi(h) \neq 0. \end{cases} \tag{3-1}$$

Furthermore, we introduce the mapping $\mathcal{K}(\bar{\epsilon})$. Consider the family of convex cones $\{\mathcal{K}(\bar{\epsilon}) : \bar{\epsilon} \in (0, \epsilon]\}$ defined by

$$\mathcal{K}(\bar{\epsilon}) := \left\{ h \in \mathbb{H} : \langle x \, , \, \phi(h) \rangle \ > \ 0, \ \text{and } e\,(h) \leqslant \bar{\epsilon}^2 \right\}, \ \text{for every } \bar{\epsilon} \in (0, \epsilon]. \tag{3-2}$$

Equivalently, observe that $h \in \mathcal{K}(\bar{\epsilon})$ if and only if $\langle x \, , \, \phi\,(h) \rangle \geqslant \|\phi\,(h)\| \sqrt{(\|x\|^2 - \bar{\epsilon}^2)}$. It follows immediately that $\mathcal{K}(\bar{\epsilon})$ is convex for every $\bar{\epsilon} \in (0, \epsilon]$.

3-1-2 Exact Non-smooth Reformulation

First, we reformulate Problem (2-1) as a non-smooth optimization problem (3-4) for which the definition of $\eta\,(h)$ is given in Equation (3-3), and equivalently in Equation (3-6). Secondly, we will prove that this formulation is a smooth problem formulation under some conditions in Section 3-1-3.

Proposition 3-1.2 (Non-smooth reformulation)**.** *Consider the parameters of LIP* (2-1) *under the setting of Assumption 3-1.1. Let the map* $\eta : \mathcal{K}(\epsilon) \longrightarrow [0, +\infty)$ *be defined by*

$$\eta\,(h) := \frac{\|x\|^2 - \epsilon^2}{\langle x \, , \, \phi\,(h) \rangle \ + \ \|\phi\,(h)\| \sqrt{\epsilon^2 - e\,(h)}}, \tag{3-3}$$

and let $B_c := \{h \in \mathbb{H} : c\,(h) \leqslant 1\}$*. Then the LIP* (2-1) *is equivalent to the minimization problem*

$$\left\{ \min_{h \in B_c \cap \mathcal{K}(\epsilon)} \ \eta\,(h) \right. \quad . \tag{3-4}$$

In other words, h^* *is an optimal solution to* (3-4) *with an optimal value* c^* *if and only if* $c^* h^*$ *is an optimal solution to the LIP* (2-1)*.*

Remark 3-1.3. *Looking ahead, it follows (from the proof of Claim A-1.3) that for any* $h \in \mathcal{K}(\epsilon)$*,* $\eta\,(h)$ *is the smallest positive root of the quadratic equation*

$$\|x - \eta\phi(h)\|^2 = \epsilon^2. \tag{3-5}$$

Equivalently, we also have

$$\eta\,(h) = \frac{\langle x \, , \, \phi\,(h) \rangle \ - \ \|\phi\,(h)\| \sqrt{\epsilon^2 - e\,(h)}}{\|\phi\,(h)\|^2} \ \textit{for every } h \in \mathcal{K}(\epsilon). \tag{3-6}$$

An expression is derived for the gradient of $\eta\,(h)$ by applying Danskin's theorem [16]. We refer the interested reader to Chapter A for the proof of Lemma 3-1.4.

Lemma 3-1.4 (Gradients of η)**.** *The function* $\eta : \mathcal{K}(\epsilon) \longrightarrow [0, +\infty)$ *is convex, differentiable at every* $h \in \operatorname{int}(\mathcal{K}(\epsilon)) = \{h \in \mathbb{H} : \langle x \, , \, \phi(h) \rangle > 0, \ \textit{and } e\,(h) < \epsilon^2\}$*, and the derivative is given by*

$$\nabla\eta(h) = \frac{-\eta\,(h)}{\|\phi\,(h)\| \sqrt{\epsilon^2 - e\,(h)}} \, \phi^a\Big(x - \eta\,(h)\,\phi(h)\Big) \quad \textit{for all } h \in \operatorname{int}(\mathcal{K}(\epsilon)), \tag{3-7}$$

where ϕ^a *is the adjoint operator of* ϕ*.*

3-1-3 Exact Smooth Reformulation

The mapping $\eta : B_c \cap \mathcal{K}(\epsilon) \longrightarrow$ is not smooth for two reasons. (i) Consider any $h \in B_c \cap \mathcal{K}(\epsilon)$, then it is immediate from Equation (3-3) that $\eta(\theta h) = \frac{C}{\theta}$ for some $C > 0$. Thus, η achieves arbitrarily large values (and arbitrarily high curvature) as $\|h\| \longrightarrow 0$. Thereby, the mapping $(0, 1] \ni \theta \longmapsto \eta(\theta h)$ is not smooth, and consequently, the mapping $\eta : B_c \cap \mathcal{K}(\epsilon) \longrightarrow \mathbb{R}$ can not be smooth. (ii) It must be observed that η is not differentiable on the boundary of the cone $\mathcal{K}(\epsilon)$. Moreover, as $e\,(h) \uparrow \epsilon^2$, i.e., h approaches the boundary of the cone $\mathcal{K}(\epsilon)$ from its interior, it is apparent from Equation (3-7) that the gradients of η are unbounded.

By avoiding these two scenarios (which will be made more formal shortly), η is smooth.

Proposition 3-1.5 (Smoothness of η). *Consider the LIP* (2-1) *under the setting of Assumption 3-1.1 and let c^* be its optimal value. For every $\bar{\eta} > c^*$ and $\bar{\epsilon} \in (0, \epsilon)$, let*

$$\mathcal{H}\,(\bar{\epsilon}, \widehat{\eta}) := \{h \in B_c \cap \mathcal{K}(\bar{\epsilon}) : \eta\,(h) \leqslant \widehat{\eta}\}, \tag{3-8}$$

and let $\beta\,(\bar{\epsilon}, \widehat{\eta})$ be a constant given by

$$\beta\,(\bar{\epsilon}, \widehat{\eta}) := \frac{\widehat{\eta}^3 \epsilon^2 \,\|\phi\|_o^2 \,(\,\|x\| + \epsilon)}{(\,\|x\| - \epsilon)^2 \,(\epsilon^2 - \bar{\epsilon}^2)^{3/2}}. \tag{3-9}$$

Then the mapping $\eta : \mathcal{H}\,(\bar{\epsilon}, \widehat{\eta}) \longrightarrow [0, +\infty)$ is $\beta\,(\bar{\epsilon}, \widehat{\eta})$-smooth. In other words, the inequality

$$\|\nabla\eta(h) - \nabla\eta(h')\| \;\leqslant\; \beta\,(\bar{\epsilon}, \widehat{\eta})\; \|h - h'\| \;\; \textit{holds for all } h, h' \in \mathcal{H}\,(\bar{\epsilon}, \widehat{\eta})\,. \tag{3-10}$$

A sufficient $\bar{\epsilon}$ is needed to avoid the non-differentiability on the boundary of the cone $\mathcal{K}(\epsilon)$. The choice of $\bar{\epsilon}$ will be described in Lemma (3-1.6) and Remark (3-1.7). The Lemma is empirically validated by the $\bar{\epsilon}$-validation experiment in Section 5-2-4.

Lemma 3-1.6. *Consider the LIP* (2-1)*, and let f^* be the optimal solution. Then we have $e(f^*) < \epsilon^2$. Therefore, there exists $\bar{\epsilon} \in (0, \epsilon)$ such that $e(f^*) < \bar{\epsilon}^2 < \epsilon^2$.*

Remark 3-1.7 (Finding $\bar{\epsilon}$). *Consider the problem of recovering some true signal f^* from its noisy linear measurements $x = \phi(f^*) + w$, where w is some additive measurement noise. Then, ϵ is chosen in* (2-1) *such that the probability $\mathbb{P}\,(\|w\| \leqslant \epsilon)$ is very high. Then for* any $\bar{\epsilon} \in (0, \epsilon)$ *we have $e(f^*) < \bar{\epsilon}^2$ with probability at least $\mathbb{P}\,(\|w\| \leqslant \epsilon) \cdot \mathbb{P}\left(\left|\left\langle w\;,\; \frac{\phi(f^*)}{\|\phi(f^*)\|}\right\rangle\right|^2 > \epsilon^2 - \bar{\epsilon}^2\right)$. Thus, in practise, based on available noise statistics, one could select $\bar{\epsilon}$ to be just smaller than ϵ.*

Finally, it results in the following theorem for the smooth reformulation of Problem (2-1) into Problem (3-11).

Theorem 3-1.1: Smooth Reformulation

Consider any $(\bar{\epsilon}, \widehat{\eta})$ such that $e(f^*) \leqslant \bar{\epsilon}^2 < \epsilon^2$ and $c^* < \widehat{\eta}$. Then the optimization problem

$$\min_{h \,\in\, \mathcal{H}(\bar{\epsilon}, \widehat{\eta})} \quad \eta\,(h) \tag{3-11}$$

is a smooth convex optimization problem. Moreover, h^* is a solution to (3-11) if and only if $c^* h^*$ is an optimal solution to (2-1).

Chapter 4

Fast Linear Inverse Problem Solver

This chapter proposes a novel algorithm to solve the smooth optimization problem as described in Chapter 3. First, the general form of the algorithm is presented, followed by a more detailed explanation of the three main steps, the descent direction, the step size, and the stopping condition. Subsequently, the algorithm is presented in detail at the end of this chapter, associated with a few remarks regarding the computational aspects.

4-1 General Form

The Fast Linear Inverse Problem Solver (FLIPS) we propose to solve Problem (2-1) is presented. The algorithm iteratively updates the solution until a stopping criterion is met. Each iteration, a descent direction $d(h)$ is computed and a step γ is taken towards this direction to decrease the objective cost of Problem (3-11). The first part of the algorithm presents the descent direction $d(h)$, which is computed as $d(h) = g(h) - h$. As $g(h)$ is unknown, we present two oracles to compute $g(h)$. The second part presents the exact step size selection. With both the descent direction $d(h)$ and the step size γ, we can perform the following update on h

$$h^+ = h + \gamma(h)d(h). \tag{4-1}$$

This update will be repeated until a convergence criterion is met, as denoted in the third part of this section. The FLIPS-algorithm to solve Problem (2-1) is presented in Algorithm 1.

Algorithm 1 Algorithm for an ill-posed LIP of the form (2-1)

Input: Measurement x, linear mapping ϕ, reconstruction error ϵ, smoothness parameter β, acceleration parameter ρ, $d(h_0) = 0$
Output: Sparse representation h
Initialise: $h_0 = \phi \backslash x$
while stopping criterion is not met **do**
  Finding the descent direction d **through oracle** (Section 4-1-1)
  $d(h) = g(h) - h$
  Performing step with optimal step size γ (Section 4-1-2)
  $h^+ = h + \gamma d(h)$
  Checking the stopping criterion (Section 4-1-3)
end while

4-1-1 Descent Directions

For both descent direction oracles, the oracle's general optimization problem is described. Subsequently, the application to Problem (2-1) with $c(f) = \|f\|_1$ is presented as an example. The Linear and Quadratic Oracle refer to the linear and quadratic descent direction optimization problem, respectively.

Linear Oracle: Finding the descent direction $d(h)$ by applying Frank-Wolfe results in solving the following linear optimization problem [23, 28]

$$g(h) \in \left\{ \underset{g \in B_c}{\arg\min} \quad \langle \nabla \eta(h) \ , \ g \rangle \right. . \tag{4-2}$$

Example 4-1.1. *For problem* (2-1) *with* $c(f) = \|f\|_1$ *and Linear oracle* (4-2)*, the Hölder inequality can be applied, which results in the following equations [49]*

$$g(h)_i = \begin{cases} -\operatorname{sgn}(\nabla \eta(h)_i), & \text{if } \nabla \eta(h_i) = \|\nabla \eta(h)\|_\infty, \ \forall i \in \mathbb{R}^{n_f} \\ 0, & \text{if } \nabla \eta(h_i) \neq \|\nabla \eta(h)\|_\infty, \ \forall i \in \mathbb{R}^{n_f} \end{cases} \tag{4-3}$$

Note that in this case, due to the infinity norm $\|.\|_\infty$*, it is likely that* $g(h)$ *will contain just one or a few non-zero indices. When big steps are taken along this direction, many indices will move to zero quickly, resulting in sparse solutions.*

Quadratic Oracle: The Quadratic Oracle is a variation of the problem formulation of Frank-Wolfe. In addition to Problem (4-2), a quadratic term $\frac{\beta}{2}\|g - h\|_2^2$ is added. Applied to Problem (2-1), if set B_c is projection friendly, we can easily apply Projected Gradient Descent (PGD) or Accelerated Projected Gradient Descent (APGD) to the following quadratic problem [50]

$$g(h) \in \left\{ \underset{g \in B_c}{\arg\min} \quad \langle \nabla \eta(h) \ , \ g \rangle \ + \ \frac{\beta}{2} \|g - h\|_2^2 \right. . \tag{4-4}$$

Example 4-1.2. *Let* Π_{B_c} *denote the projection onto set* B_c *and* β *the smoothness parameter. Then the following formulation presents the PGD*

$$g(h) = \Pi_{B_c}(h - \frac{1}{\beta}\nabla \eta(h)), \tag{4-5}$$

For $c(f) = \|f\|_1$, Π_{B_c} is the projection onto the ℓ_1-ball [20]. If one considers a form that is not projection friendly, one should apply the Linear oracle instead.

Example 4-1.3. *For APGD we apply the Nesterov gradient acceleration in which the gradient step is taken along a linear combination of the current gradient $\nabla\eta(h)$ and the direction of the previous iteration $d(h^-)$ [50]*

$$g(h) = \Pi_{B_c}(h - \frac{1}{\beta}(\nabla\eta(h) + \rho d(h^-)), \tag{4-6}$$

where ρ weights the momentum of the acceleration. If $\rho = 0$, this formulation is equal to the PGD.

4-1-2 Step Size

The reformulation of Problem (2-1) allows us to compute the explicit solution of the optimal step size without significantly increasing the computational cost. We take some $\bar{\epsilon} \in (\bar{\epsilon}(f^*), \epsilon)$. Then the solution is of the form

$$\gamma(h) = \begin{cases} \underset{\gamma\in[0,1]}{\arg\min} & \eta(h + \gamma d(h)) \\ \text{subject to} & h_\gamma \in \mathcal{K}(\bar{\epsilon}). \end{cases} \tag{4-7}$$

The solution to this problem is described below in Lemma (4-1.4) and Proposition (4-1.5).

Lemma 4-1.4. *Consider problem* (A-19).

Let $\widehat{\gamma}(h) \coloneqq \max\{\gamma \in [0,1] : h_\gamma \in \mathcal{K}(\bar{\epsilon})$ be the maximum step size, then

$$\widehat{\gamma}(h) = \begin{cases} +\infty & a \geqslant 0, \\ \dfrac{-b - \sqrt{b^2 - 4ac}}{2a}] & a < 0 \end{cases} \tag{4-8}$$

where a, b and c are defined as

$$\begin{aligned} a &= \|\phi(d)\|^2 (\bar{\epsilon}^2 - e(d)) \\ b &= 2\langle x\ ,\ \phi(h)\rangle \langle x\ ,\ \phi(d)\rangle - (\|x\|^2 - \bar{\epsilon}^2)\langle\phi(h\ ),\ \phi(d)\rangle \\ c &= \|\phi(h)\|^2 (\bar{\epsilon}^2 - e(h)). \end{aligned} \tag{4-9}$$

Proposition 4-1.5. *The solution of problem* (4-7) *gives rise to the following optimal step size γ*

$$\gamma = \begin{cases} 0 & \textit{if } \left.\dfrac{d\eta(h_\gamma)}{d\gamma}\right|_{\gamma=0} > 0 \\ \gamma(h) & \textit{if } \left.\dfrac{d\eta(h_\gamma)}{d\gamma}\right|_{\gamma=0} \leqslant 0, \left.\dfrac{d\eta(h_\gamma)}{d\gamma}\right|_{\gamma=\widehat{\gamma}(h)} \leqslant 0 \\ \dfrac{-s \pm \sqrt{s^2 - 4ru}}{2r} & \textit{if } \left.\dfrac{d\eta(h_\gamma)}{d\gamma}\right|_{\gamma=0} \leqslant 0, \left.\dfrac{d\eta(h_\gamma)}{d\gamma}\right|_{\gamma=\widehat{\gamma}(h)} > 0, \end{cases} \tag{4-10}$$

where $\widehat{\gamma}(h)$ is defined by Lemma 4-1.4 and

$$\begin{aligned}
r &= \|\phi(d)\|^2 \left(\bar{\epsilon}^2 - e\left(d\right)\right) \\
s &= 2\left\langle x\ ,\ \phi(h)\right\rangle \left\langle x\ ,\ \phi(d)\right\rangle - \left(\|x\|^2 - \bar{\epsilon}^2\right)\left\langle \phi(h\ ),\ \phi(d)\right\rangle \\
u &= \frac{2\left\langle x\ ,\ \phi(d)\right\rangle \left\langle x\ ,\ \phi(h)\right\rangle \left\langle \phi(h)\ ,\ \phi(d)\right\rangle}{\|\phi(d)\|^2} - \frac{\|\phi(h)\|^2 \left\langle x\ ,\ \phi(d)\right\rangle^2}{\|\phi(d)\|^2} \dots \\
&\quad - \frac{\left(\|x\|^2 - \bar{\epsilon}^2\right)\left\langle \phi(h)\ ,\ \phi(d)\right\rangle^2}{\|\phi(d)\|^2}.
\end{aligned} \tag{4-11}$$

The proofs of these two propositions can be found in Appendix A.

4-1-3 Stopping Condition

We propose two stopping criteria. Both criteria occur when there is no direction to decrease the objective.

1. Stop if $g(h) = h$ since $d(h) = g(h) - h = 0$ and $h^+ = h + \gamma \cdot 0 = h$.
2. Stop if $\gamma(h) = 0$ since $h^+ = h + 0 \cdot d(h) = h$.

4-1-4 Remarks Fast Linear Inverse Problem Solver Algorithm

Two main computations are reused throughout the algorithm. First, within the descent direction computations, the matrix multiplication $\phi^a\phi$ needs to be computed once and can be reused throughout each iteration. Secondly, $\phi(h)$ is only computed in the first iteration, and can be updated as $\phi(h^+) = (1-\gamma)\phi(h) + \gamma\phi(g)$ which has lower computational cost compared to the multiplication $\phi(h^+)$.

Two more aspects are highlighted regarding the computational cost of the algorithm. Since $\phi(h)$ and $\phi(d)$ are already computed in the first step of the algorithm, the computation of $\gamma(h)$ is not adding significant computational cost. Subsequently, observe that the computation $\phi(g)$ is easy since g is sparse.

Chapter 5

Numerical Results

In this chapter, the numerical results are presented. All experiments are applied to the image denoising problem except for the final experiment. First of all, the experiment details will be given regarding the open-source package; the initial solution, and the image preprocessing. Secondly, the following experiments are described to get an overview of FLIPS' performance.

- *Section 5-2-1, 5-2-2, 5-2-3*: Parameter tuning experiments for the Fast Linear Inverse Problem Solver (FLIPS), Chambolle-Pock (CP) and the Constrained Split Augmented Lagrangian Shrinkage Algorithm (C-SALSA).
- *Section 5-2-4*: An $\bar{\epsilon}$-validation experiment to empirically validate Lemma 3-1.6.
- *Section 5-3-1*: A full image size experiment using FLIPS for different inputs.
- *Section 5-3-2*: Comparing state-of-the-art methods experiments:
  - Experiment in which a large (128×128) image patch is recovered, showing the applicability to high-dimensional data of all three methods.
  - Full image experiment to highlight the difference in convergence rates of all three methods.
  - An experiment showing the differences regarding the CPU-times for all three methods for different patch sizes.
  - An experiment regarding the sparsity over the iterations of all three methods.
- *Section 5-3-3*: A final experiment in which the algorithm is implemented within a Dictionary Learning algorithm applied to the denoising and inpainting problem to prove its wide applicability.

5-1 Experiment Details

5-1-1 Open-source Package

The MATLAB code associated with this algorithm can be found in the footnote[1]. Note that the package is placed on an open-source website as soon as our associated paper is submitted. This package contains the FLIPS as well as the C-SALSA and CP algorithm. An application to the image denoising problem can be found within the package. For FLIPS, the Quadratic Oracle of Algorithm 1 was implemented, but the Linear Oracle can be used by activating the appropriate section. Consult the 'Read.me' file for further instructions.

C-SALSA was implemented according to Algorithm C-SALSA-1 in [1]. One simplification within the algorithm was applied considering the computational complexity. The multiplication $(\phi^\top \phi)^{-1}$ was computed just once at the start to avoid performing this computation once every iteration, which reduces a significant amount of the computation time. This simplification creates a fair comparison since the precomputation $\phi^a \phi$ was also applied in FLIPS as described in Chapter 4.

CP was implemented according to [10, 11]. No precomputation was applied in this case since the algorithm does not contain the computation of the form $(\phi^\top \phi)$.

5-1-2 Initial and Optimal Solution

The initial solution h_0 used throughout all experiments is the least squares solution. This solution is of the form $h_0 = \phi \backslash x$. The least squares solution minimizes the sum of squares of the residuals and is of the form

$$h_0 = \arg\min_h \|\phi(h) - x\|^2 . \tag{5-1}$$

In FLIPS, the initial solution is normalized to have a unit ℓ_1-norm to fit the constraints. The optimal value f^* of Problem (2-1) is determined by FLIPS after 1000 iterations for all experiments.

5-1-3 Image Preprocessing

The pixels of the input image were first scaled into the range between 0 and 1, after which the image was resized to the chosen image dimensions. The noise of these images was added by applying the MATLAB function `imnoise` with Gaussian noise. For all experiments, the noise mean is set to zero, and the variance σ is chosen appropriate to the experiment at hand. Since image signals are usually of high dimension, a sliding patch approach is implemented to retrieve small patches of the input image, which speeds up the algorithm.

In the following experiments, note that a 'full image' experiment is referred to as an experiment in which an entire image is used as input using a sliding patch approach. For example, if we have an image of dimension 4×4, this would mean that the first patch of size 2×2 would consist of the pixels $\{1, 5, 2, 6\}$, see Figure 5-1. The second patch would start one position

[1] https://filesender.surf.nl/?s=download&token=dbd4c38c-06f0-4a0d-a1fb-ff53b92aacdf

down and consists of $\{5, 9, 6, 10\}$ and so on. If the first column is covered, which means, in this case, that three patches have been selected, then the sliding approach moves to the second column, selecting the fourth patch with the pixels $\{2, 6, 3, 7\}$ until the entire image is covered. A single patch is used in experiments with big image patches (128×128) since an entire image with the sliding patch approach would take many hours to compute. For smaller patches ($\leqslant$32×32), the sliding patch approach is applied.

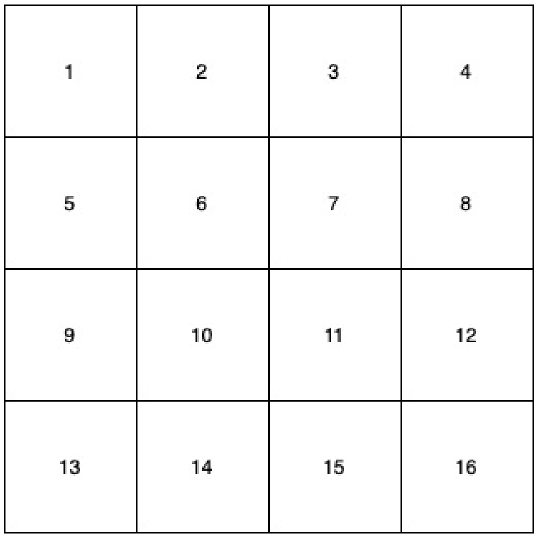


Figure 5-1: Graphical representation of the sliding patch approach.

Problem (2-1) will be solved independently for every patch, and the patches will be reconstructed back to one image at the end. In all Sparse Coding experiments, the DCT-dictionary is chosen as the linear mapping ϕ. In all cases, the value for ϵ is chosen as $\epsilon = \sqrt{\sigma n_x}$. Note that the noise variance σ is assumed to be known beforehand and is therefore not tuned.

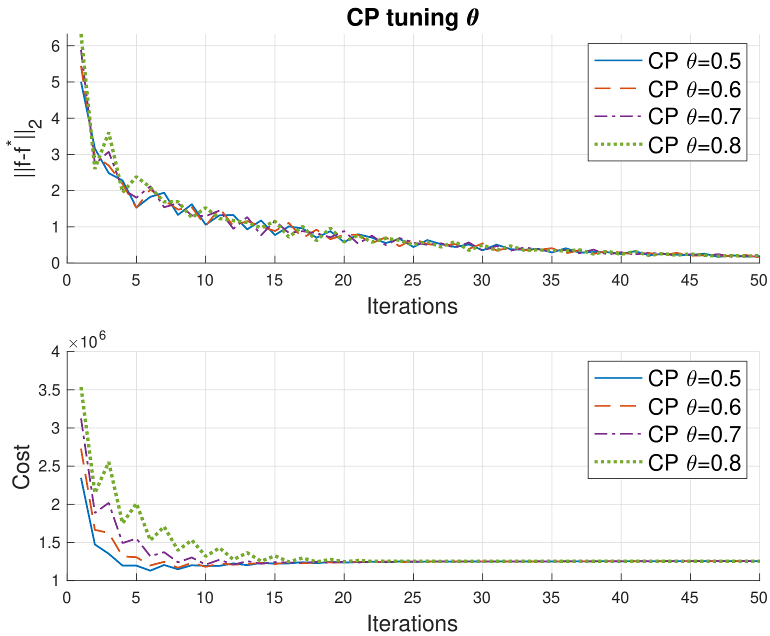
5-2 Tuning Experiments

This section shows the conducted tuning experiments to find the best hand-tuned parameters for each method. All experiments in this chapter are run on a laptop with Apple M1 Pro and 16GB RAM using MATLAB 2022a.

5-2-1 Tuning CP

The CP algorithm in [11], contains three tuning parameters; the primal step size σ, the dual step size τ, and the momentum parameter θ. Both σ and τ are set to $\sigma = \frac{1}{\sigma_m}$ and $\tau = \frac{1}{\sigma_m}$ with σ_m representing the maximum singular value of the linear mapping ϕ [11]. The value of the momentum step parameter $\theta \in [0, 1]$ is hand-tuned.

Figure 5-2 shows the results of the tuning experiment for θ with a full image input for the patch size 32×32 with noise variance $\sigma = 0.0055$ with sliding patch approach. The size 32×32 was chosen by making a trade-off between low computation time and the desire for an application to high-dimensional signals. The noise level $\sigma = 0.0055$ was chosen since this amount of noise is clearly visible in the input images and can almost be completely removed in the output images. The first metric used in this graph is the average sub-optimality $\|f - f^*\|$, where f^* is the optimal solution. The second metric is the total cost of the objective function of Problem (2-8) of all patches at each iteration. In the top graph of Figure 5-2, it can be seen that the differences between the different values of θ are negligible. In the bottom graph, observe that $\theta = 0.5$ results in a small undershoot and $\theta = 0.7$ and $\theta = 0.8$ show slower convergence compared to $\theta = 0.6$. Therefore, $\theta = 0.6$ was chosen. This parameter is kept constant throughout all experiments since other patch size experiments gave similar results.



7

Figure 5-2: Image denoising results for different values of θ for the 200×200 'cameraman' image with sliding patch approach for the patch size 32×32 and $\sigma = 0.0055$.

5-2-2 Tuning C-SALSA

C-SALSA contains one tuning parameter; the augmented Lagrangian penalty parameter μ [1]. For patch sizes smaller than 64×64, we take $\mu = 2.5$, and for patch sizes bigger than or equal to 64×64, we take $\mu = 3$. In Figure 5-3 and the zoomed Figure 5-4, the different graphs show the different results for each value of μ for an experiment with 32×32 patches. It can be seen that the higher μ is chosen, the faster convergence in the sub-optimality graph. However, it also results in higher cost for the initial iterations, shown in the second graph. Therefore, a trade-off was made, and the value $\mu = 2.5$ was chosen. The same experiment was done for the 64×64 patches which resulted in $\mu = 3$.

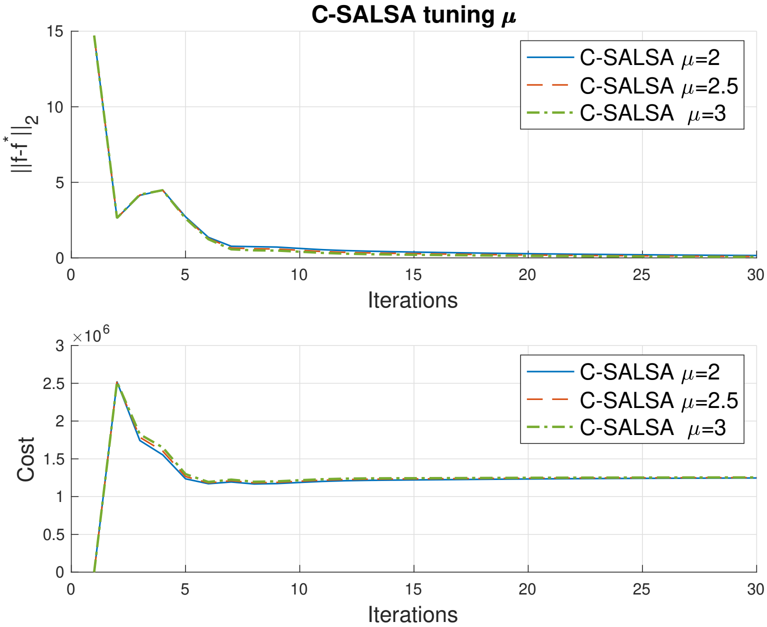


Figure 5-3: Image denoising results for different values of μ for the 200×200 'cameraman' image with sliding patch approach for the patch size 32×32 and $\sigma = 0.0055$.

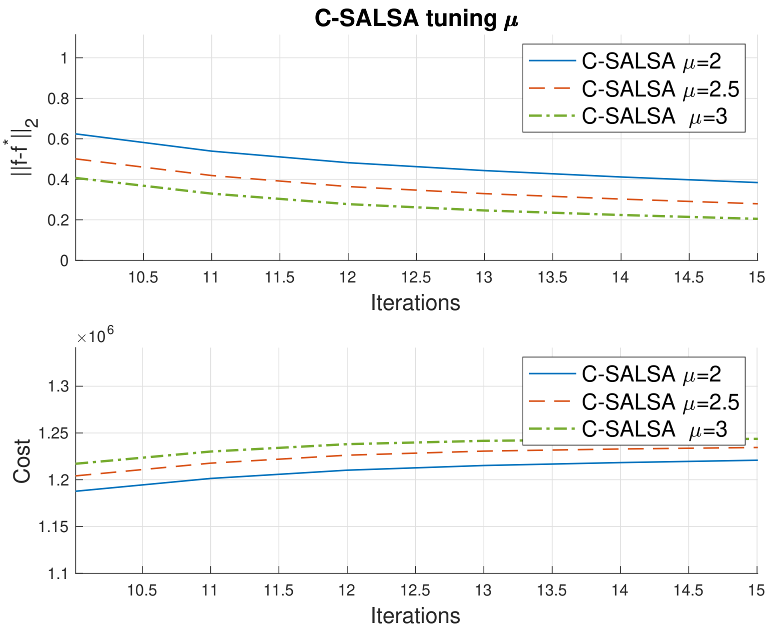


Figure 5-4: Zoomed image denoising results for different values of μ for the 200×200 'cameraman' image with sliding patch approach for the patch size 32×32 and $\sigma = 0.0055$.

5-2-3 Tuning FLIPS

Applying FLIPS with the Quadratic Oracle of Algorithm 1 contains two tuning parameters; the acceleration parameter ρ, and the smoothness parameter β. The smoothness parameter is hand-tuned for every patch size independently. However, the acceleration parameter is kept constant for different patch sizes since the experiments for different patch sizes resulted in the same optimal parameter. FLIPS in combination with the Linear Oracle, does not require any tuning parameter. The Quadratic Oracle was applied throughout the experiments since it outperforms the Linear Oracle in terms of speed for the denoising problem.

Since many vectors and matrices in this algorithm contain just a few non-zeros, the MATLAB function `sparse` is used throughout the algorithm to reduce storage. This function only stores the non-zero values and their positions and assumes all others to be zero. Applying this function did not significantly improve the speed for the other methods. Furthermore, as mentioned in Chapter 4, the matrix multiplication $\phi^\top \phi$ in the descent direction step of the Algorithm 1 is used once every iteration. However, since this multiplication remains constant, it can be precomputed once at the start of the algorithm and used throughout. Since $\phi(h)$ is the largest matrix of the entire algorithm, this reduces the computational complexity remarkably. Subsequently, $\phi(h)$ only needs to be computed once and can be updated every iteration according to Algorithm 1.

Three different values for ρ are shown in Figure 5-5. Note that the difference between the graphs is minimal. Therefore, a zoomed frame is shown in Figure 5-6. Both graphs highlight that $\rho = 0.7$ is the optimal value considering the sub-optimality and the cost of all three options. This value appears optimal throughout experiments for different patch sizes and is therefore kept constant throughout the experiments.

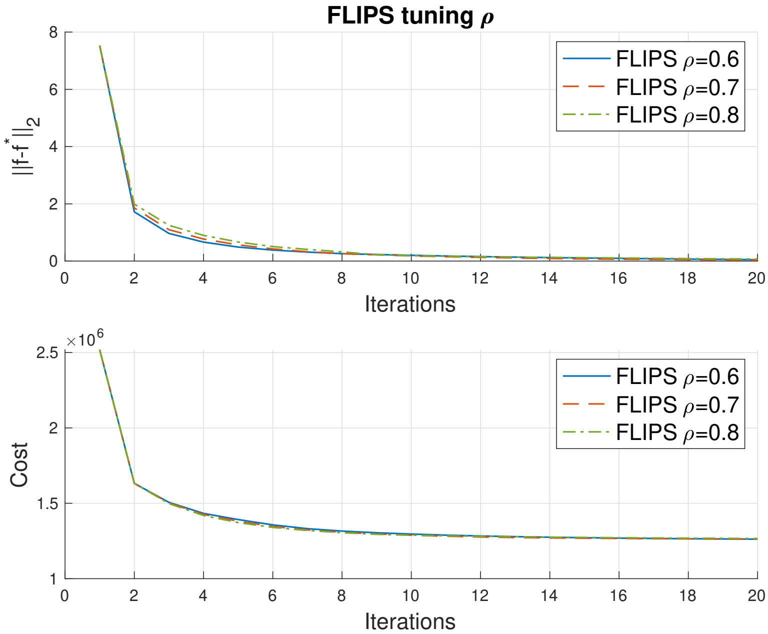


Figure 5-5: Image denoising results for different values of ρ for the 200×200 'cameraman' image with sliding patch approach for the patch size 32×32 and $\sigma = 0.0055$.

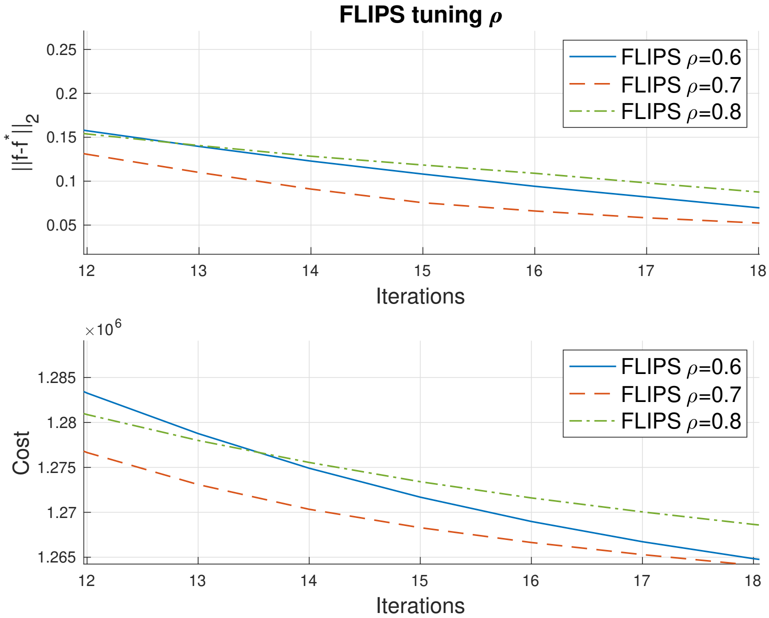


Figure 5-6: Zoomed image denoising results for different values of ρ for the 200×200 'cameraman' image with sliding patch approach for the patch size 32×32 and $\sigma = 0.0055$.

The smoothness parameters are tuned for every patch size independently. The experiment for patch size 32×32 is highlighted. Figure 5-7 shows the graphs of three different smoothness parameters. Again, the differences are minimal; thus, a zoomed plot is presented in Figure 5-8. For this experiment, it must be seen that the optimal acceleration parameter is $\beta = \frac{1}{2.4 \cdot 10^{-4}}$ considering the three options. These tuning experiments were done for every patch size.

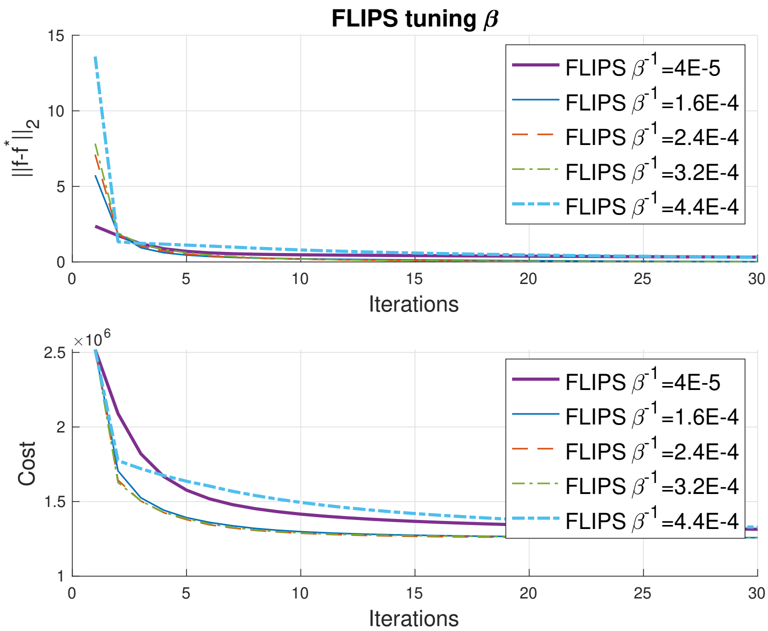


Figure 5-7: Image denoising results for different values of β for the 200×200 'cameraman' image with sliding patch approach for the patch size 32×32 and $\sigma = 0.0055$.

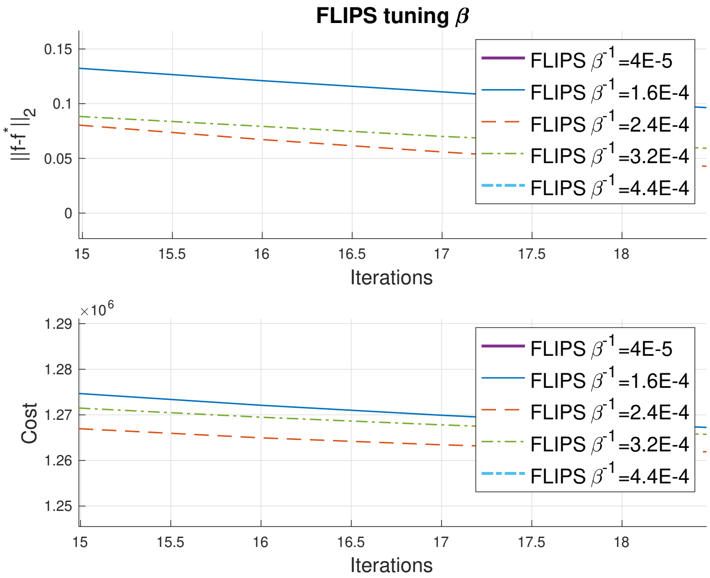


Figure 5-8: Zoomed image denoising results for different values of β for the 200×200 'cameraman' image with sliding patch approach for the patch size 32×32 and $\sigma = 0.0055$.

5-2-4 Validation Reconstruction Error

To empirically validate Lemma 3-1.6, an experiment was conducted to check $e(f^*) < \bar{\epsilon}^2$. In the histogram in Figure 5-9, $e(f^*)$ is computed for every patch and placed within the appropriate bandwidth of 0.01. Note that a full 200×200 image with sliding patch approach results in 28561 image patches. For example the section $5.60 < e(f^*) < 5.61$ counts 3043 patches. Most importantly, it can be seen that there is a clear gap between the maximum $e(f^*)$ and $\bar{\epsilon}^2$. Therefore, we can empirically verify Lemma 3-1.6 for this specific experiment.

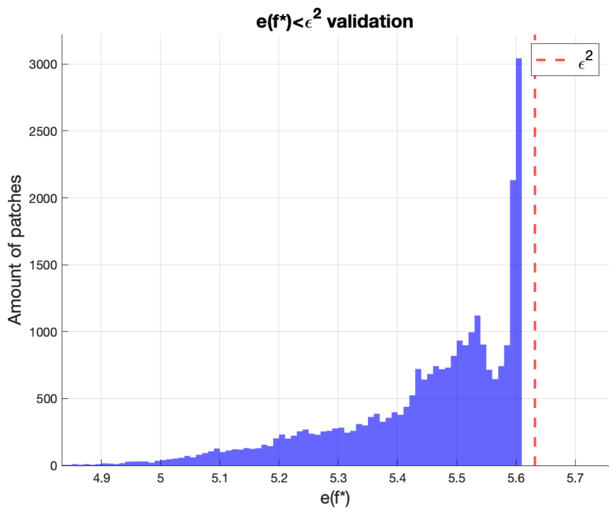


Figure 5-9: Image denoising results of $e(f^*)$ for all patches for the denoising problem with 200×200 'cameraman image' with 32×32 sliding patch approach for 50 iterations with noise variance $\sigma = 0.0055$.

5-3 Experiments

The experiments can be divided into three categories. First, an application with an entire image is presented to show the overall performance of FLIPS for different inputs. Secondly, FLIPS is compared to state-of-the-art methods. Finally, the Dictionary Learning experiment is presented to show the applicability of FLIPS to this method.

5-3-1 Full Image Experiment

This experiment shows that the image denoising problem can be solved by applying FLIPS. Consider the following input images; the ‘cameraman’, ‘Barbara’ [41] and the ‘boat’ image [19], which are standard images in the field of image processing. The ‘cameraman’ image is of unknown origin but is owned by MIT. All three images are of size 200×200 and are applied to the image denoising problem. The results are shown in Figure 5-10.

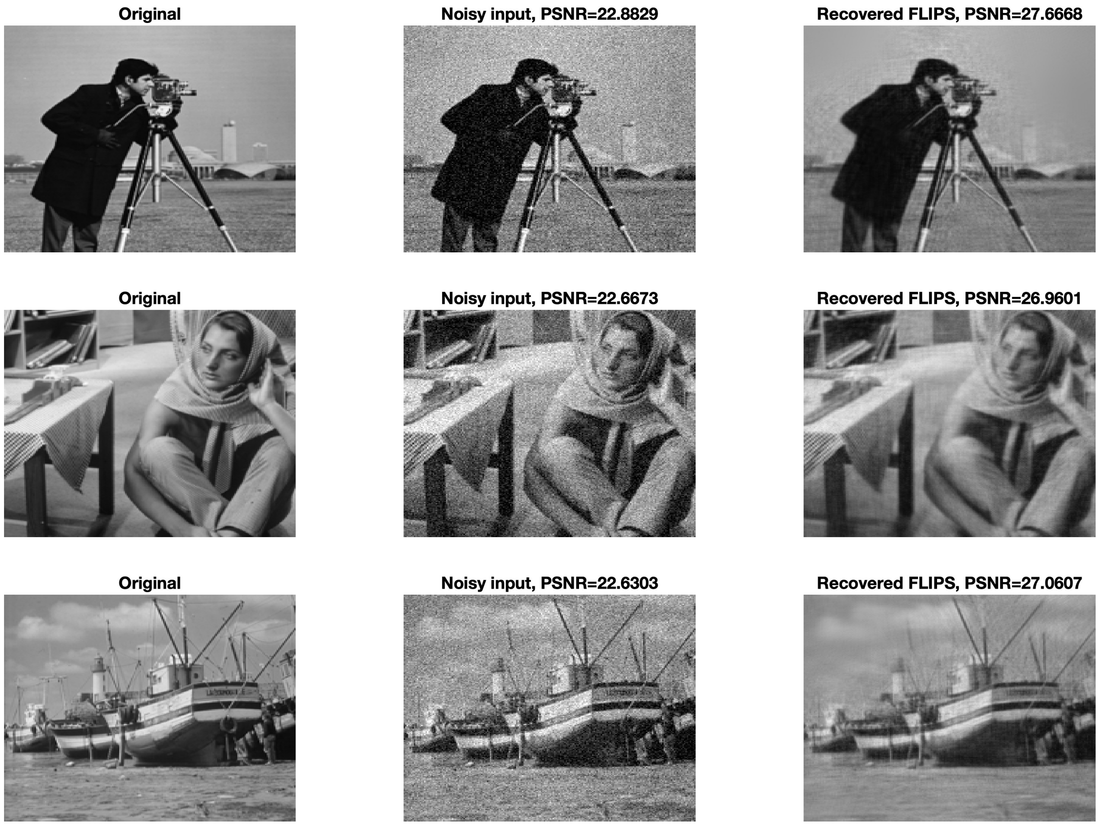


Figure 5-10: Image denoising results for FLIPS for the ‘cameraman’, ‘Barbara’, and ‘boat’ images with the sliding patch approach with patch size 32×32. From left to right, the original image, the noisy input, and the recovered image.

5-3-2 Comparison with State-of-the-art

The performance of FLIPS compared to CP and C-SALSA is described in this section. First of all, a large image patch experiment is shown. Secondly, a full image experiment is presented in which the recovered images, distance to the optimality graphs, and CPU-times until convergence are presented. Finally, we present a sparsity robustness experiment.

Large Image Patch Experiment

The large image patch experiment shows that the image denoising problem can be solved by applying FLIPS for big image patches. One 128×128 patch is taken from the 400×400 'cameraman' image. It can be seen that after two iterations, most of the noise has been removed by FLIPS but not by C-SALSA and CP. C-SALSA removed all the noise after around three iterations and CP after around 50. The associated PSNR values above the images also highlight that FLIPS converges faster than the other methods in this experiment.

Full Image Experiment

The full image patch experiment shows the application of the three methods to images of the size 200×200. We show the recovered images, the distance to optimality graph, and the associated CPU-times until convergence. The experiment was run for 80 iterations for all CP, C-SALSA and FLIPS. The recovered images can be seen in Figure 5-12. It must be seen that all three methods converge to the same recovered image. Furthermore, note that the PSNR-value of FLIPS is slightly higher than the other two methods, which indicates that applying FLIPS results in better recoveries compared to CP and C-SALSA. However, an improvement this small should be neglected.

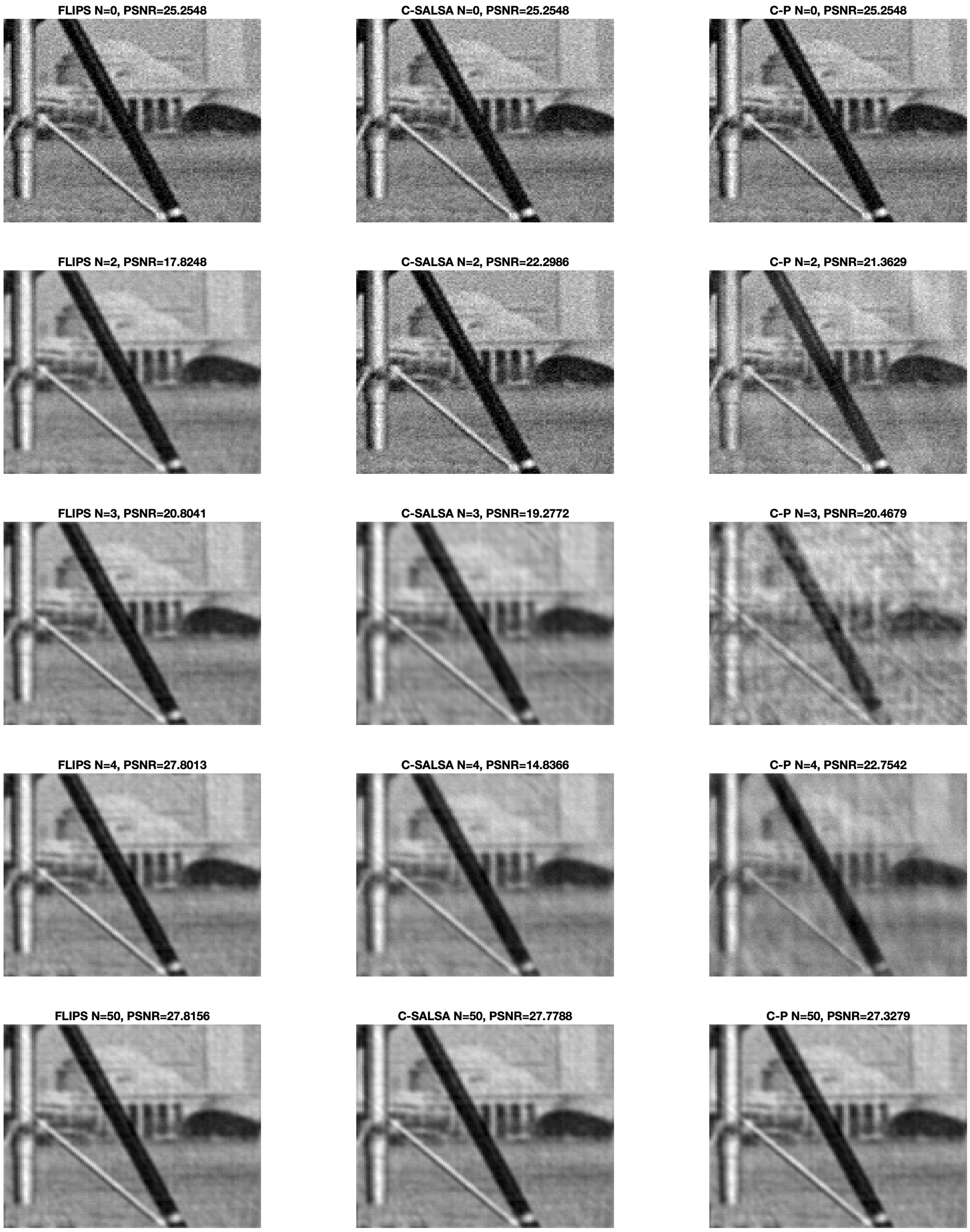


Figure 5-11: Image denoising results for different iterations for the methods FLIPS, C-SALSA, and CP with corresponding PSNR values. The image is one patch 128×128 from the 400×400 'cameraman' image. The noise variance $\sigma = 0.003$ and $\epsilon = \sqrt{\sigma n_x}$. N denotes the number of iterations.

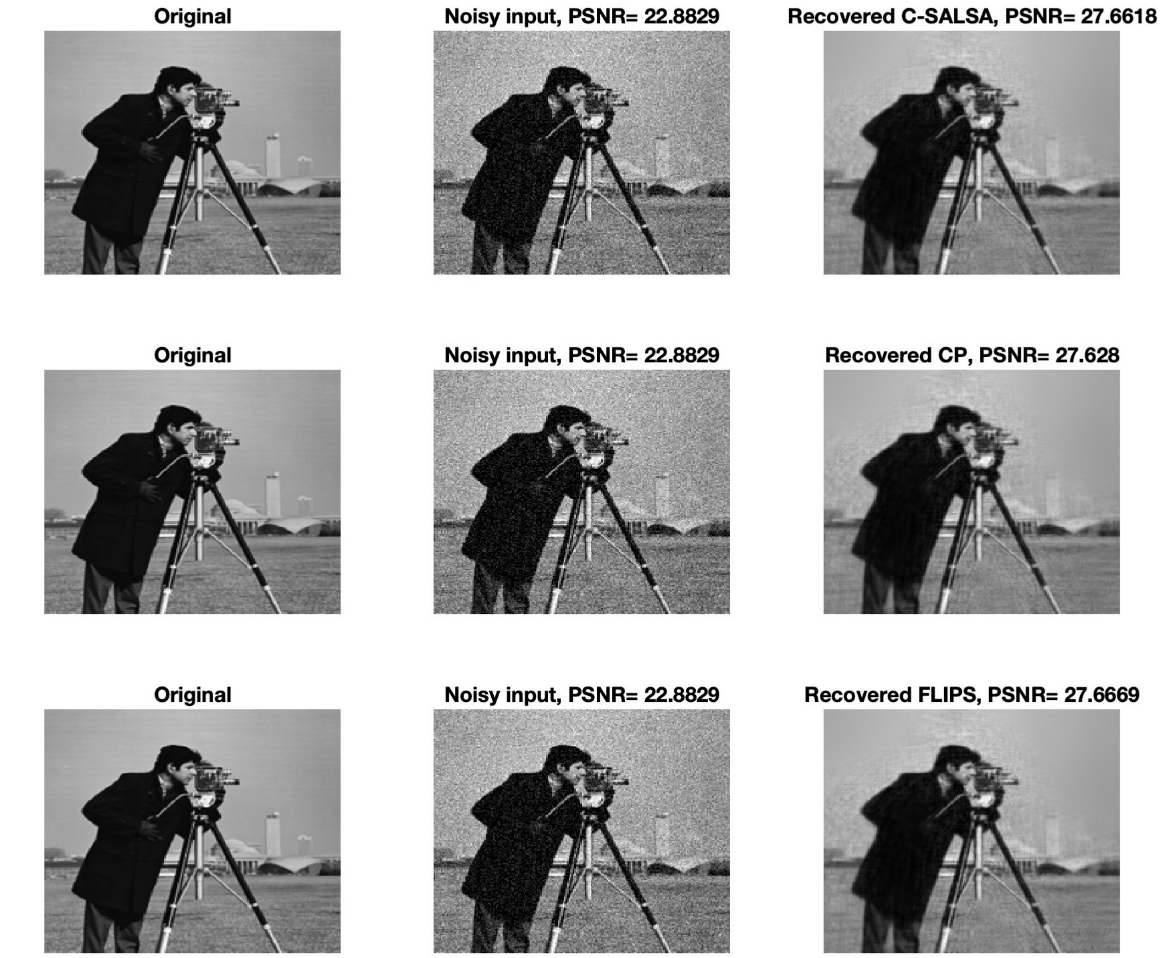


Figure 5-12: Image denoising results for C-SALSA, CP and FLIPS for the 200×200 'cameraman' image with sliding patch approach for the patch size 32×32 and $\sigma = 0.0055$. From left to right: the original image, the noisy input, and the recovered image.

The performance of all three methods can be compared by plotting the Euclidean norm as $\|f - f^*\|$ over the iterations. Observe that the graph presents the mean of all patches. It can be seen in Figure 5-13 and Figure 5-14 that with a threshold of $\|f - f^*\| < 0.1$, FLIPS converges after 16 iterations, C-SALSA after 30 iterations and CP after 70 iterations (which is outside the graph's domain). To highlight the initial convergence, only 50 of the 80 iterations are plotted in Figure 5-13.

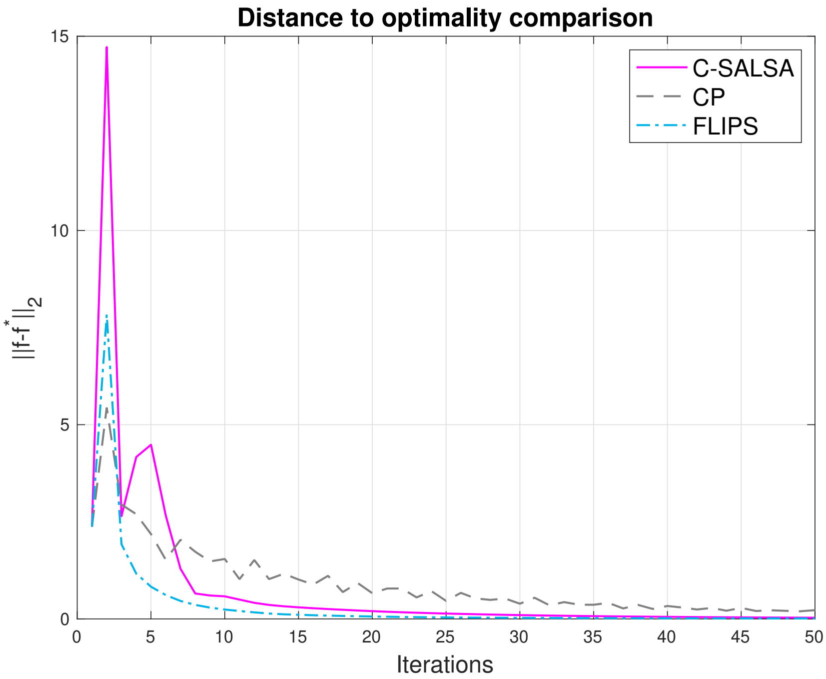


Figure 5-13: Average image denoising results of all patches for C-SALSA, CP and FLIPS for the 200×200 'cameraman' image with sliding patch approach for the patch size 32×32 and $\sigma = 0.0055$.

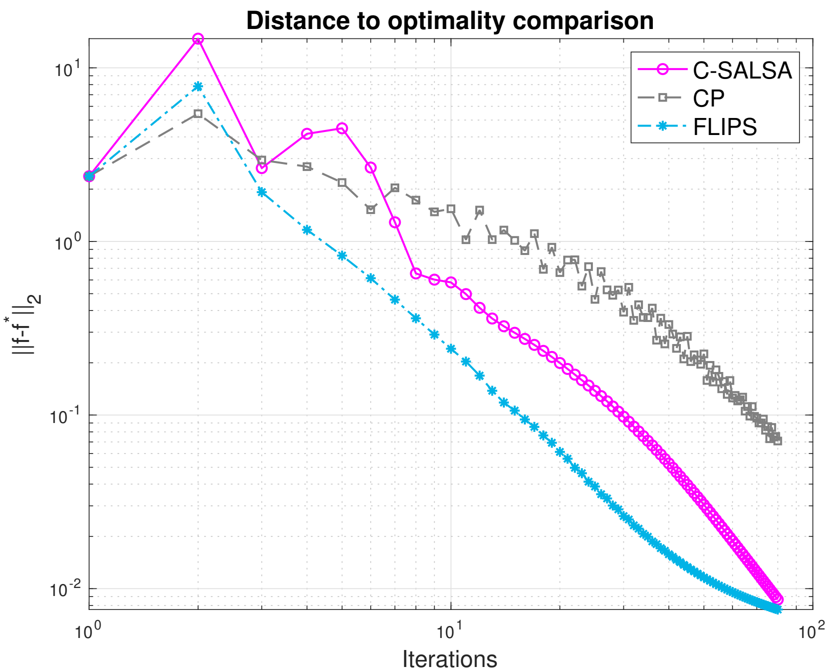


Figure 5-14: Average image denoising results of all patches for C-SALSA, CP and FLIPS for the 200×200 'cameraman' image with sliding patch approach for the patch size 32×32 and $\sigma = 0.0055$.

Since our algorithm aims to reduce computation time, the CPU-times until convergence are compared between the methods. We use the Root Mean Square Error (RMSE) as a tool to measure the convergence. The RMSE, as explained in the preliminaries in Chapter 1, measures the error between the current f and the optimal f^*. First, we evaluated the RMSE over the iterations in a denoising experiment for every patch size independently. The largest value after which the RMSE stopped decreasing significantly was chosen. It was found that the threshold RMSE=$5 \cdot 10^{-3}$ was sufficient for all patch sizes. The results of the CPU-times are shown in Figure 5-15. For example for the 64×64 patch size experiment, it can be seen that FLIPS needs $2.58 \cdot 10^3$s CPU-time, C-SALSA needs $1.44 \cdot 10^4$s CPU-time and CP needs $2.87 \cdot 10^5$s CPU-time. Associated with this graph, the number of iterations needed until convergence (CA) is denoted in Table 5-1. Observe that FLIPS needs fewer iterations until convergence for all experiments with varying patch sizes and has a lower CPU-time for patches >8×8 compared to CP and C-SALSA.

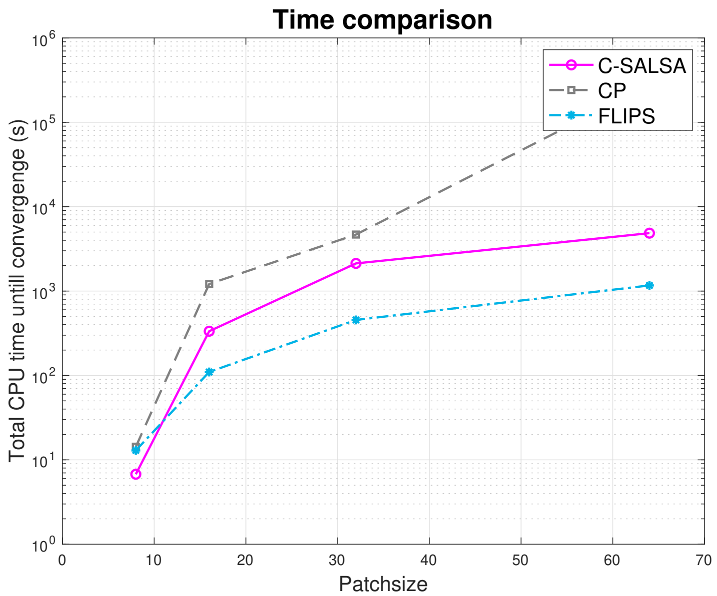


Figure 5-15: Image denoising results for C-SALSA, CP and FLIPS for the 200×200 'cameraman' image with sliding patch approach for the patch size 32×32 and $\sigma = 0.0055$.

Table 5-1: Image denoising results associated with Figure 5-15. For all patch sizes and methods, the average number of iterations after which the method converged (CA) is presented.

	4×4 CA (iterations)	**8×8 CA (iterations)**	**16×16 CA (iterations)**	**32×32 CA (iterations)**	**64×64 CA (iterations)**
CP	58.1	44.9	46.1	51.3	47.3
C-SALSA	21.1	22.3	22.8	22.1	18.2
FLIPS	6.0	8.7	9.8	13.9	11.3

Sparsity Robustness Experiment

In addition to the algorithm's speed, the desire to have sparse solutions should be noted. Figure 5-16 shows the sparsity levels of all three methods, measured by the ℓ_0-norm. FLIPS is likely to converge fast to many zero indices. However, C-SALSA and CP converge to small indices, but not exactly zero. Therefore, the levels are denoted for different thresholds s_t. A threshold $s_t = 10^{-3}$ represents that all absolute values smaller than 10^{-3} are considered zero. From Figure 5-16, it can be concluded that FLIPS is more robust to this threshold than the other methods. However, note that after many iterations, all three methods converge to the same solution.

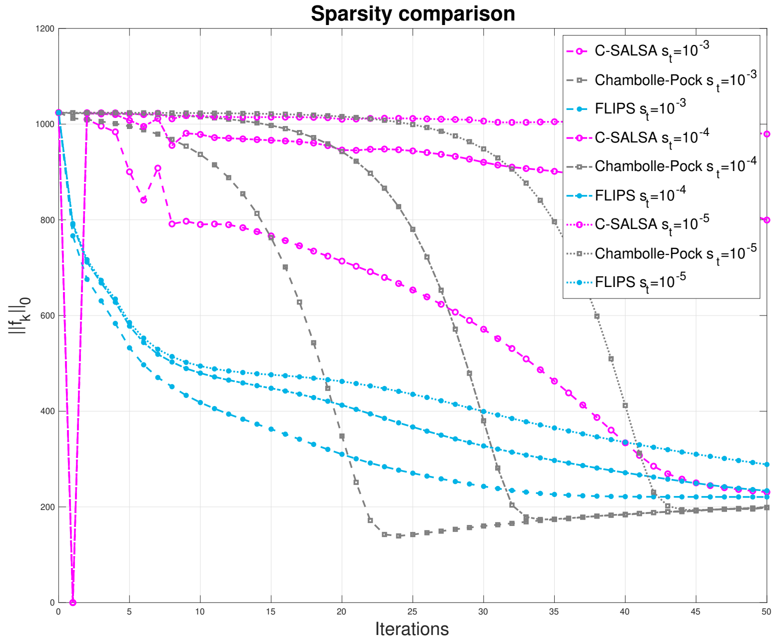


Figure 5-16: Sparsity image denoising results for C-SALSA, CP and FLIPS for the 200×200 'cameraman' image with sliding patch approach for the patch size 32×32 and $\sigma = 0.0055$. $\|f_k\|_0$ denotes the ℓ_0-norm which counts the number of nonzero indices of f_k.

A visual explanation of the idea behind the Sparse Coding problem applied to the denoising problem is discussed in Chapter 7.

5-3-3 Dictionary Learning

As introduced in Chapter 1, Dictionary Learning is an algorithm in which the dictionary is not predefined but learned throughout the algorithm. Since the Sparse Coding problem is incorporated once for every iteration of the algorithm, it is a suitable experiment to test FLIPS in this setting. For this experiment, the well-known K-SVD algorithm introduced by Aharon et al. [2] is considered, using the efficient implementation of [39]. In these papers, Orthogonal Matching Pursuit (OMP) is applied as a Sparse Coding algorithm which will be replaced by FLIPS. Furthermore, two commonly used tricks are implemented. First of all, once every five iterations, the least used atom is replaced by the normalized training signal that is least well-represented. Secondly, once every five iterations, one of the two most similar atoms, measured by the absolute inner product, is replaced by the least well-represented training signal.

The initial input dictionary is half the Discrete Cosine Transform (DCT)-dictionary, and half random numbers, with all columns normalized to unit length. FLIPS is set to a maximum of 20 iterations, and the K-SVD iterations have a maximum of 50. In the Dictionary Learning part, the training data is the 'cameraman' image without any additional noise with sliding patch approach and patch size 8×8. In FLIPS, the smoothness parameter is set to $\beta = 100$ and the momentum parameter $\rho = 0.7$ and epsilon $\epsilon = 0.89$. After the Dictionary Learning part, the learned dictionary is tested. Therefore, the Sparse Coding problem is applied to the test data, which is the 200×200 'Lena' image with added Gaussian noise with noise variance $\sigma = 0.0025$ by the function `imnoise`. Epsilon is tuned as $\epsilon = 1.2\sqrt{\sigma n_x}$ and the smoothness parameter as $\beta = 20000$. The tuning criterion was to have non-blurry but also noise-free reconstruction measured with the author's best effort by vision. Figure 5-17 shows the learned dictionary. The DCT-dictionary is shown on the right. Figure 5-18 shows the original image, the noisy input image, and the recovered image by FLIPS, respectively. It can be seen that FLIPS is suitable to be included in the Dictionary Learning algorithm and that the learned dictionary can be applied in the image denoising problem using FLIPS.

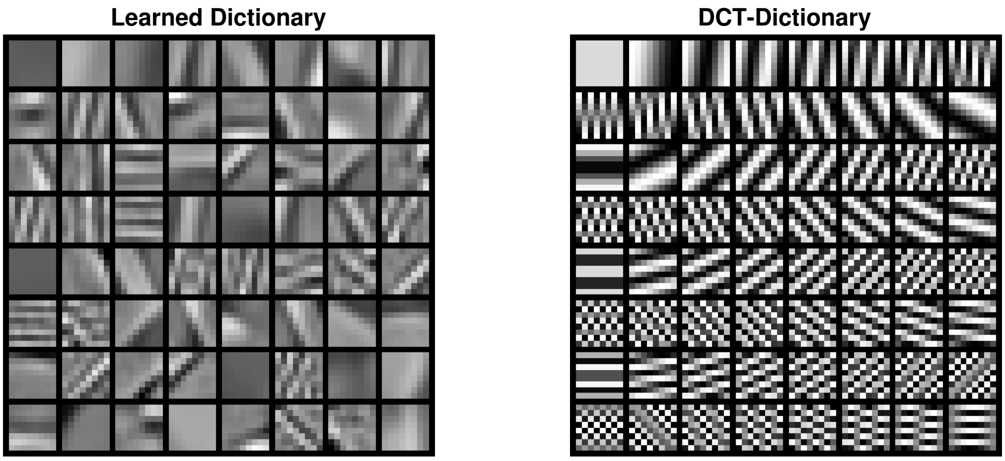


Figure 5-17: Left: Learned dictionary by the K-SVD Dictionary Learning experiment with FLIPS for the 200×200 noise-free 'cameraman' image with sliding patch approach for patch size 8×8. Right: DCT-dictionary.

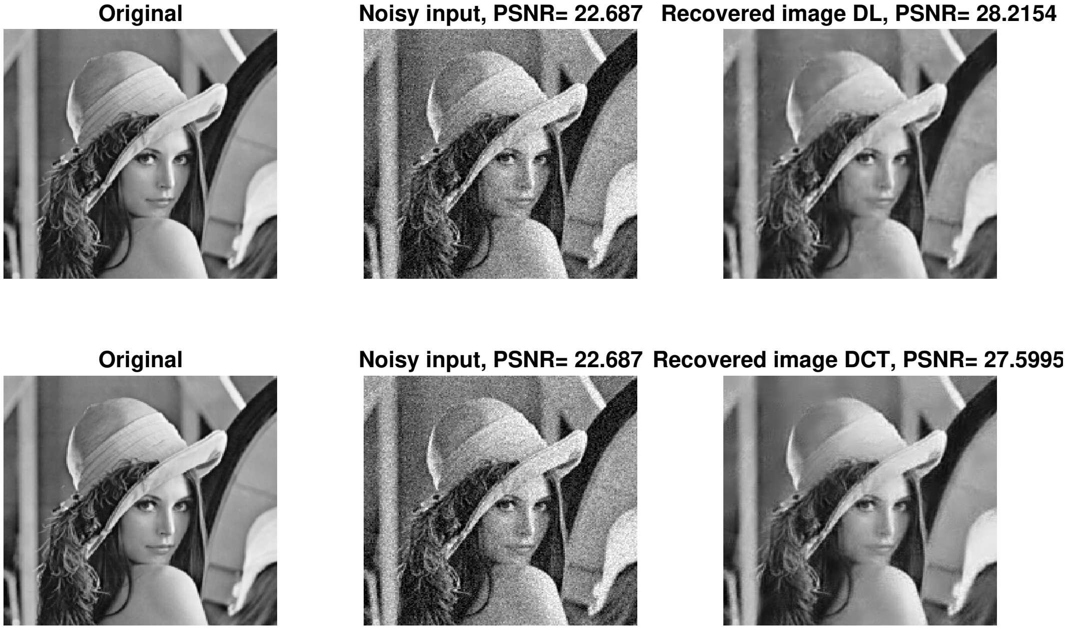


Figure 5-18: Dictionary Learning experiment results for K-SVD with FLIPS for the 100×100 noise-free 'cameraman' image with sliding patch approach for patch size 8×8 followed by the Sparse Coding application with FLIPS for the 200×200 'Lena' image with patch size 8×8 and additional Gaussian noise with noise variance $\sigma = 0.0055$. From left to right: the original image, the noisy input, and the recovered image for the application with the learned dictionary and the DCT-dictionary.

Furthermore, FLIPS was applied to the inpainting problem using the learned dictionary. In the original 100×100 input image, random groups of 2×2 pixels were set to zero. These pixels need to be recovered to pixels that 'fit' the image. The best 'fit' is found by trying to find a sparse representation of the pixels that have not been removed. As a result, the pixels that were removed are replaced by the selection of the dictionary atoms. This way, the entire image can be reconstructed, as can be seen in Figure 5-19. The reconstructed image is close to the original image, even though not all information was available due to the missing pixels. A visual explanation of the idea behind the Sparse Coding problem applied to the inpainting problem is discussed in Chapter 7.

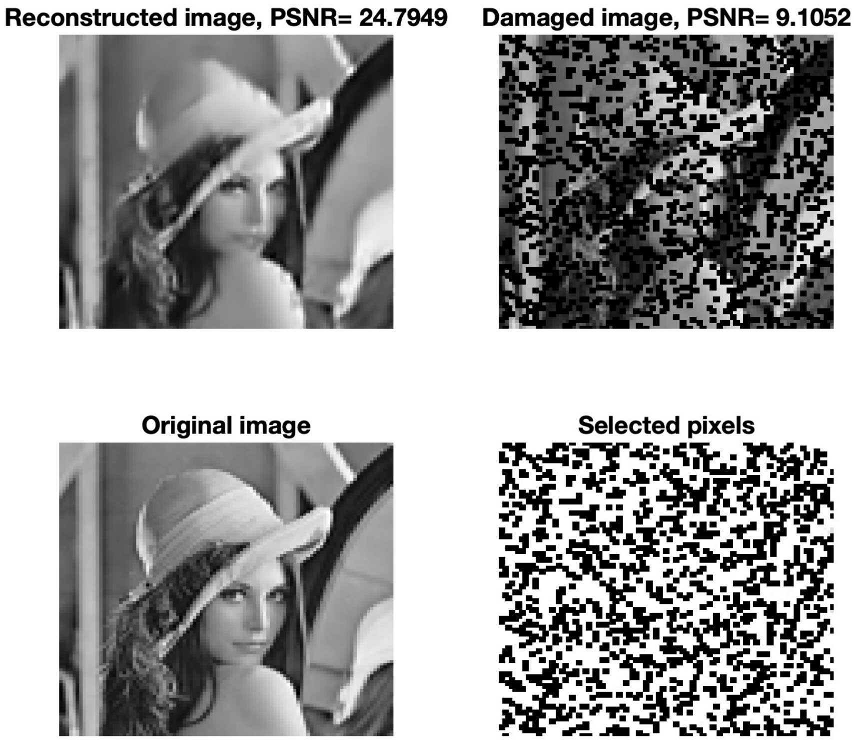


Figure 5-19: Results for the inpainting problem for the 100×100 'Lena' image with sliding patch approach for 100 FLIPS iterations and epsilon = 0.24.

Chapter 6

Conclusion

A concise conclusion is drawn to finalise the results of this master thesis report. First and foremost, let us repeat the objective of this report. The objective is to find an algorithm that can solve the ill-posed Linear Inverse Problem (2-1) with less computation time than state-of-the-art methods. We want the algorithm to converge fast, therefore decreasing the computation time. As a result, the algorithm would be able to work with higher dimensional signals compared to state-of-the-art methods.

As a result of the problem reformulation, we present the Fast Linear Inverse Problem Solver (FLIPS). The method can be applied to the denoising problem for various images, shown in Section 5-3-1. More precisely, the method can be applied to any natural signal known to admit a sparse representation under an appropriate basis. Furthermore, the method converges faster than Constrained Split Augmented Lagrangian Shrinkage Algorithm (C-SALSA) and Chambolle-Pock (CP) and converges to the same optimal solution, as can be seen in Section 5-3-2. Mainly due to the fast convergence, FLIPS outperforms the other methods in terms of CPU-times in the time comparison experiment in Section 5-3-2. For example, in the 16×16 experiment in Section 5-3-2, C-SALSA needs twice as many iterations, and CP even needs five times more iterations compared to FLIPS. This reduction in the number of iterations results in a tremendous decrease in computation time. Moreover, the algorithm results in sparse solutions and is robust to different thresholds in the ℓ_0-norm experiment, which is not the case for CP and C-SALSA. Finally, it is proven that the algorithm can be applied in Dictionary Learning for denoising and inpainting problems. Therefore, the overall conclusion based on these experiments is that FLIPS outperforms state-of-the-art methods in terms of speed and is widely applicable.

Chapter 7

Discussion and Recommendations

This chapter presents the discussion and recommendations for future research. First, the fact that only the DCT-dictionary and a learned dictionary are considered in this report will be discussed. Secondly, the Linear Oracle and Projected Gradient Descent will be discussed, which are not considered in the numerical results. Thirdly, the large area inpainting problem will be discussed. Fourthly, the blurry recovery of the 128×128 experiment in Section 5-3-2 will be highlighted, followed by the discussion on the limited storage of the experiments. Fifthly, a visual explanation of the denoising and inpainting problem is given. Besides the discussion, a few recommendations are presented that could be investigated further as an addition to this master thesis report.

7-1 Discussion

7-1-1 DCT-Dictionary

For all experiments, the commonly chosen DCT-dictionary was chosen. This dictionary was chosen since it is commonly used, and thus creates a fair comparison [38, 34, 3, 27, 6]. Furthermore, a learned dictionary was applied in the Dictionary Learning experiment. However, other dictionaries, such as wavelets, countourlets could also be applied and could result in different recoveries of images. A dictionary that 'fits' the data better will be likely to result in a better recovery of the noisy image. Observe that the DCT-dictionary only contains straight-lined patterns in the atoms, see Figure 5-17. Also, observe that the learned dictionary in Figure 5-17 contains curved structures in the atoms. This indicates that the straight lines of the DCT-dictionary might not be optimal for sparse representations of images. Furthermore, the DCT-dictionary is a square dictionary. As mentioned in Section 1-3, an overcomplete dictionary covers a broader range of signal phenomena compared to undercomplete dictionaries, which is another reason why other types of dictionaries should be tried as well.

7-1-2 Linear Oracle

In all numerical results of the Fast Linear Inverse Problem Solver (FLIPS) in Section 5-2 and Section 5-3, only the Quadratic Oracle of FLIPS is highlighted since the Quadratic Oracle outperforms the Linear Oracle tremendously. However, an experiment is presented to prove that the Linear Oracle can be applied as well.

One advantage is that there are no tuning parameters for the Linear Oracle, presented in Section 4-1-1. However, a disadvantage is that this method results in very small step sizes to avoid violating the cone constraint, see Figure 7-1. Therefore, this method needs many iterations to converge to the optimal solution. Note that in Figure 7-1, the objective cost of Problem (4-2) is still decreasing in the last iterations of the graph and is therefore not converged yet. Another difficulty is that the method needs a large value for the reconstruction error ϵ to avoid violation of the cone constraint (3-2) in Problem (3-10). Because of these issues, applying the Linear Oracle to the denoising problem is not recommended. However, it is not tested for other types of signals or problems that could fit the Linear Oracle better.

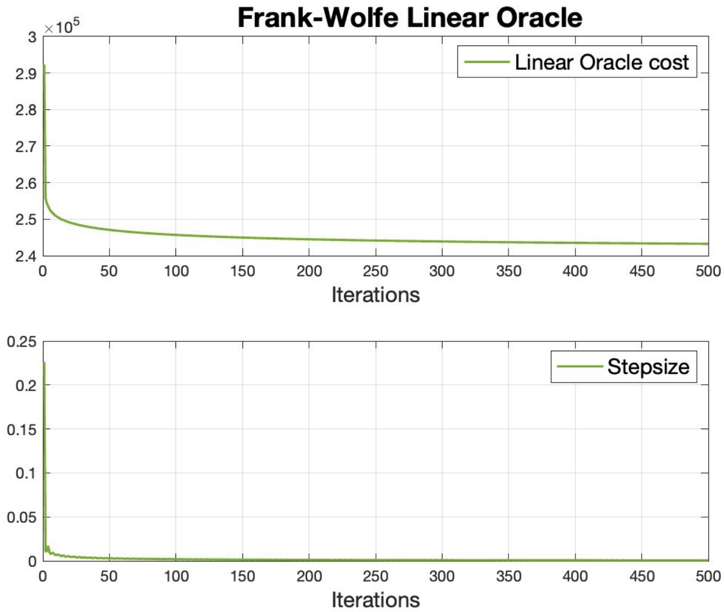


Figure 7-1: Results of the Linear Oracle experiment FLIPS for the 200×200 'cameraman' image with a sliding patch approach for the patch size 32×32 and $\sigma = 0.0055$. From top to bottom, the cost over the iterations and the step size over the iterations.

7-1-3 Projected Gradient Descent

Next to the state-of-the-art methods, Chambolle-Pock (CP) and Constrained Split Augmented Lagrangian Shrinkage Algorithm (C-SALSA), Projected Gradient Descent was implemented, directly applied to Problem (3-11). In this method, the step size γ needs to be tuned. Note that Projected Gradient Descent is not the same as the Quadratic Oracle. The Projected Gradient Descent is of the form $h^+ = \Pi_{B_c}(h - \delta\nabla\eta(h))$, whereas the Quadratic Oracle computes Equation (4-5): $g(h) = \Pi_{B_c}(h - \frac{1}{\beta}\nabla\eta(h))$ and applies exact line search.

Just like the Linear Oracle, this method also required small step sizes δ to remain within the constraints. Therefore, it also converged very slowly and was not considered in this thesis report. Figure 7-2 shows the objective cost over the iterations for a simple experiment in which a random 8×8 patch of the 200×200 cameraman image is applied to the denoising problem with Gaussian noise with variance $\sigma = 0.0055$ for 200 iterations. It can be seen that the Projected Gradient Descent method converges very slowly compared to FLIPS.

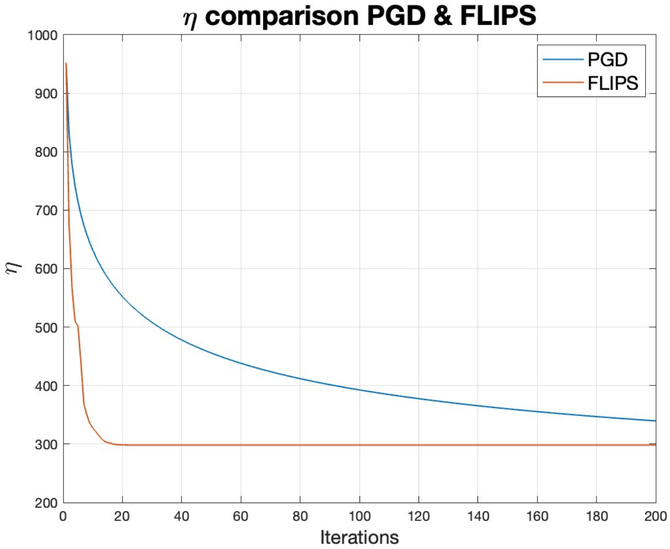


Figure 7-2: Image denoising results of the PGD and FLIPS comparison for the denoising problem with 8×8 patch size for the 200×200 cameraman image with Gaussian noise with noise variance $\sigma = 0.0055$ for 200 iterations with PGD stepsize $\gamma = 0.000031$.

7-1-4 Blurry Image Recovery 128x128 Experiment

In Figure 5-11, the recovered image still contains some blurry and noisy aspects. However, all three algorithms converge to the same solution. Therefore, it must be highlighted that this recovery is the optimal solution for this signal, even though not all of the noise has been removed. This recovery shows a limit of denoising for large image patches. Note that for the denoising problem, it is common to select small image patches since they result in better denoised recovered signals.

7-1-5 Limiting Storage Experiments

The aim of the presented algorithm FLIPS was to improve the speed and increase the maximum patch size compared to state-of-the-art methods. Therefore, applying the algorithm to the biggest patch sizes presented in state-of-the-art literature would be a good experiment. Take for example the FISTA paper, in which an experiment is conducted for patch size 256×256 [4]. However, the bottleneck was that storing a 256^2x256^2 dictionary exceeded the storage limit of the laptop. One way to avoid this limit is to work with the MATLAB functions `idct2` and `dct2`, instead of matrices, for ϕ. For example, the multiplication $\phi \cdot h$

could be replaced by `idct2`(h). These functions directly apply the DCT- and inverse DCT-transform to each index instead of a big matrix multiplication with ϕ. Note that for a single 128×128 patch experiment, this method is 150 times faster. However, this was not applied in the experiments, due to late discovery and limited time.

7-1-6 Limiting Signal Size

A remark should be made regarding the size of the input signals. Images of size 200×200 have been used as input for the experiments. However, note that for a 64×64 experiment, FLIPS took 15 minutes to converge all patches (or compute the maximum number of iterations), while CP needed around 14 hours, see Section 5-3-2. Even though FLIPS is tremendously faster, it would still take hours to work with images of size 2000×2000. Furthermore, for the sliding patch approach, we would have 3751969 patches, needing 115 GB to store the signals, which exceeds the maximum storage of the average computer.

7-1-7 Visual Explanation Denoising Problem

One could try to imagine the idea behind the Sparse Coding problem applied to the denoising problem. We will explain this by using visuals. Take for example the original patch, the noisy patch, the recovered patch as displayed in Figure 7-3, and a part of the DCT dictionary in Figure 7-4. The Sparse Coding problem denotes the task of representing the noisy input with a linear combination of just a few of the dictionary atoms. When selecting the combination of atoms, one tries to get the 'best' reconstruction for most of the pixels. The 'best' reconstruction is measured with the ℓ_2-norm between the noisy input and the reconstruction. Since we are limited to using a few atoms, the selection of atoms can only recover the general structure of the noisy patch. A part of the linear combination of patches for this particular input is shown in Figure 7-5. The weight of each patch is shown above the atoms. As one tries to imagine the combination of: $10.4796 \cdot$ atom #1 + $5.5142 \cdot$ atom #2 + $1.0295 \cdot$ atom #3 − $1.0015 \cdot$ atom #4, it would result in a recovered image shown on the right of Figure 7-3. Note that this combination of dictionary atoms is not the exact combination selected for this recovery but a small selection with the largest weights.

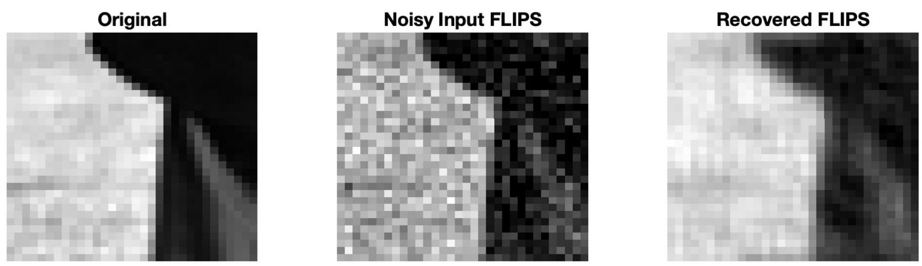


Figure 7-3: Original, noisy, and recovered patch of the 200×200 'cameraman' image for the patch size 32×32 and $\sigma = 0.0055$.

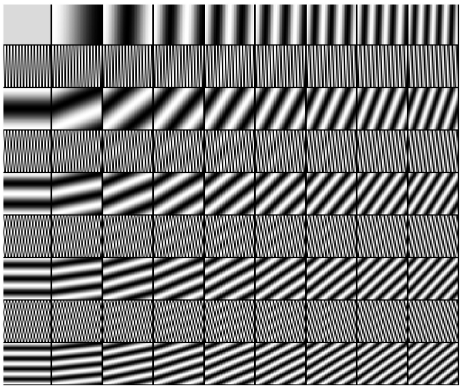


Figure 7-4: Part of the DCT-dictionary for 32×32 patches.

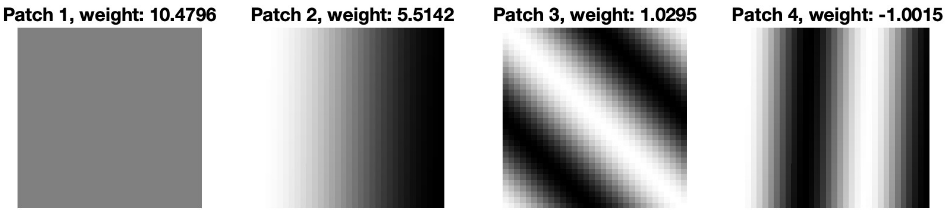


Figure 7-5: Part of the combination of dictionary atoms to recover the noisy input of Figure 7-3. The weight associated with each dictionary atom is denoted above the images.

7-1-8 Visual Explanation Inpainting Problem

In the inpainting experiment in Section 5-3-3, a random selection of 2×2 pixels have been removed, shown as black pixels in Figure 7-6 on the right. The process of finding a 'good' reconstruction for the removed pixels is explained. A 'good' reconstruction is a reconstruction in which the recovered pixels fit the surrounding area.

Let $x_n \in \mathbb{R}^{n_n}$ be the part of $x \in \mathbb{R}^{n_x}$ containing only the original pixels, i.e., the pixels that have not been removed. Observe that $n_n \leqslant n_x$. Furthermore, consider a local dictionary D_l, in which the rows of the original dictionary D, associated with the missing pixels, have been removed. Find the sparse representation vector f by solving Problem (2-1) using local dictionary D_l and the partial measurement x_n. To recover the entire image patch, use the original dictionary D to compute the reconstruction Df. As a result, the removed pixels have been reconstructed to 'fit' the surrounding pixels.

Take for example the patch in the upper right corner of the 'Lena' image, shown on the left in Figure 7-6. A sparse representation of the non-removed pixels was found by solving Problem (2-1). The four atoms with the largest absolute values of the sparse representation vector f, are shown in Figure 7-7. One could try to image that the weighted combination of these four atoms results in the reconstructed patch on the left in Figure 7-6.

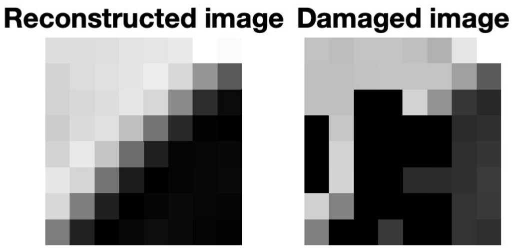


Figure 7-6: Inpainting problem one image patch in the upper right corner of the 100×100 'Lena' image with sliding patch approach for 100 FLIPS iterations and epsilon = 0.24.

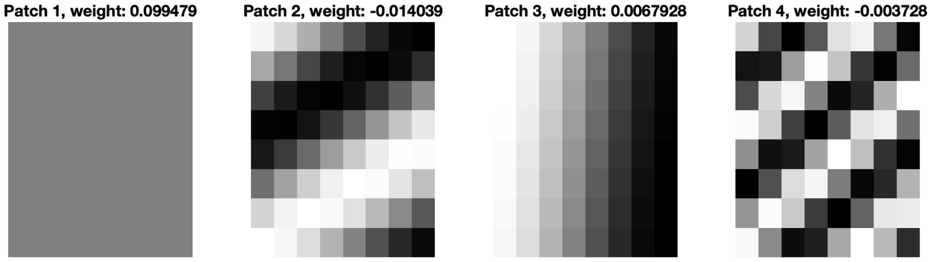


Figure 7-7: Part of the combination of dictionary atoms to recover the inpainting input of Figure 7-6. The weight associated with each dictionary atom is denoted above the images.

7-2 Recommendations

7-2-1 Large Inpainting Experiment

In Chapter 1, it was mentioned that for large image inpainting, the need to work with large image patches arises. However, in the inpainting experiment in Chapter 5, the areas to be inpainted are of the size 2×2 pixels. To prove that the algorithm would be applicable to large area inpainting, an experiment could be conducted in which large image patches are applied to images with large areas of missing pixels. However, due to time limitations, this experiment was not conducted.

7-2-2 Application To Other Problems

In this report, FLIPS is tested with images as input data. However, the algorithm could also be tested on other signals and problems. As mentioned in Section 2-1-2, the Matrix Completion problem is a good candidate due to its high-dimensional data and the problem formulation. Moreover, signals with known sparsity, such as audio signals, could be tested.

Besides the application to the denoising problem and Dictionary Learning, the application to Compressed Sensing and the deblurring problem could be explored as well. Both problems consider high-dimensional data in which FLIPS could be much faster than state-of-the-art methods.

7-2-3 Convergence Rate

Up to this stage, FLIPS' convergence rate has not been formally proven. However, if provided together with the algorithm, it would make a stronger case to compare it with state-of-the-art methods.

7-2-4 Parfor Loop

One final recommendation is presented on the application side of the algorithm. MATLAB versions from R2008a contain the function `parfor`. This function is recommended to speed up the process. This function creates the possibility of running multiple loops at the same time. However, in that case, storing variables in the same places is not possible. In other words, the loops should be separable. With the current implementation, the sparse representation vector h is stored in a matrix, making the for loop non-separable. One way to solve this issue is to store the variables in different places and merge them at the end. However, since this would create thousands of variables, it was decided not to implement this function.

Appendix A

Proofs

A-1 Proofs for Non-smooth Reformulation

For any $h \in B_c$, let us consider the maximization problem

$$\begin{cases} \sup\limits_{\lambda} & L(\lambda, h) \coloneqq 2\sqrt{\langle \lambda \ , \ x \rangle - \epsilon \left\| \lambda \right\|} \ - \ \langle \lambda \ , \ \phi(h) \rangle \\ \text{subject to} & \langle \lambda \ , \ x \rangle - \epsilon \left\| \lambda \right\| > 0. \end{cases} \tag{A-1}$$

We provide an explicit characterization of the solution to the maximization problem (A-1). Interestingly, whenever the solution exists, it can be computed without much additional computation from its explicit characterization.

Proposition A-1.1. *Consider the maximization problem* (A-1) *in the setting of Assumption 3-1.1. Then, the following assertions hold.*

1. *The maximization problem* (A-1) *is bounded if and only if* $h \in \mathcal{K}(\epsilon)$*, and the maximal value is* $\eta(h)$*. In other words,*

$$\eta(h) \ = \ \sup_{\lambda \in \Lambda} \ L(\lambda, h) \ . \tag{A-2}$$

2. *The maximization problem* (A-1) *admits a maximizer* $\lambda(h)$ *if and only if* $h \in \operatorname{int}(\mathcal{K}(\epsilon)) = \{h \in \mathbb{H} : \langle x \ , \ \phi(h) \rangle > 0, \ \textit{and} \ e(h) < \epsilon^2\}$*, which is unique, and given by*

$$\lambda(h) = \frac{\eta(h)}{\left\| \phi(h) \right\| \sqrt{\epsilon^2 - e(h)}} (x - \eta(h) \, \phi(h)). \tag{A-3}$$

Proof of Proposition A-1.1. The proof relies heavily on [42, Lemma 35, 36] under the setting $r = 2, q = 0.5$, and $\delta = 0$. We shall divide the proof of the proposition into three claims.

Claim A-1.2 (unboundedness)**.** *If* $h \notin \mathcal{K}(\epsilon)$*, then the maximization problem is unbounded.*

Proof. We first recall from [42, Lemma 36 and assertion (iii) of Lemma 35] that the maximal value of (A-1) is unbounded if and only if there exists a λ' such that the two following inequalities are satisfied simultaneously

$$\langle \lambda' \, , \, \phi(h) \rangle \; \leqslant \; 0 \; < \; \langle \lambda' \, , \, x \rangle - \epsilon \, \| \lambda' \| \, . \tag{A-4}$$

Since $h \notin \mathcal{K}(\epsilon)$, either $\langle x \, , \, \phi\,(h) \rangle < 0$ or $e\,(h) > \epsilon^2$. On the one hand, if $\langle x \, , \, \phi\,(h) \rangle < 0$, then we observe that $\lambda' = x$ satisfies the two inequalities of (A-4) since $\|x\| > \epsilon$. On the other hand, if $e\,(h) > \epsilon^2$, then by considering $\lambda' = x - \frac{\langle x \, , \, \phi(h) \rangle}{\|\phi(h)\|^2}\phi(h)$, we first observe that $\langle \lambda' \, , \, \phi(h) \rangle = 0$, and by Pythagoras theorem, we have $\|\lambda'\|^2 = e\,(h)$. It is now easily verified that λ' satisfies the two inequalities (A-4) simultaneously since

$$\begin{cases} \langle \lambda' \, , \, \phi(h) \rangle \; = \; \langle x \, , \, \phi\,(h) \rangle - \dfrac{\langle x \, , \, \phi\,(h) \rangle}{\|\phi\,(h)\|^2} \langle \phi(h) \, , \, \phi(h) \rangle \; = \; 0 \\ \langle \lambda' \, , \, x \rangle - \epsilon \, \|\lambda'\| \; = \; \|x\|^2 - \dfrac{|\langle x \, , \, \phi\,(h) \rangle|^2}{\|\phi\,(h)\|^2} - \epsilon\sqrt{e\,(h)} \; = \; \sqrt{e\,(h)}\big(\sqrt{e\,(h)} - \epsilon\big) \; > \; 0. \end{cases}$$

Thus the claim holds. □

Claim A-1.3 (optimal value)**.** *If $h \in \mathcal{K}(\epsilon)$, then the maximal value of* (A-1) *is finite and equal to $\eta\,(h)$ as given in* (3-3)*.*

Proof. We now recall from [42, Lemma 36, and (51)-Lemma 35], that the maximal value of (A-1) is bounded if and only if the following minimum exists

$$\min \; \{\theta \geqslant 0 : \|x - \theta\phi(h)\| \leqslant \epsilon\}. \tag{A-5}$$

Clearly, the minimum in (A-5) exists whenever the minimization problem is feasible. Suppose there exists some $\theta' \geqslant 0$ such that $\|x - \theta'\phi(h)\| \leqslant \epsilon$, it is immediately seen that

$$\begin{cases} \langle x \, , \, \phi\,(h) \rangle \; \geqslant \; \dfrac{1}{2\theta'}\Big((\,\|x\|^2 - \epsilon^2) + \theta'^2 \, \|\phi\,(h)\|^2 \Big) \; > \; 0, \text{ and} \\ e\,(h) \; = \; \min\limits_{\theta \in \mathbb{R}} \; \|x - \theta\phi(h)\|^2 \; \leqslant \; \|x - \theta'\phi(h)\|^2 \leqslant \epsilon^2. \end{cases}$$

Thus, $h \in \mathcal{K}(\epsilon)$. On the contrary, if $h \in \mathcal{K}(\epsilon)$, then it also seen similarly that $\theta' = \frac{\langle x \, , \, \phi(h) \rangle}{\|\phi(h)\|^2}$ is feasible for (A-5). Thus, the maximal value of (A-1), and the minimum in (A-5) is finite if and only if $h \in \mathcal{K}(\epsilon)$.

It is immediately realised that the value of the minimum in (A-5) corresponds to the smaller root of the quadratic equation $\|x - \theta\phi(h)\|^2 = \epsilon^2$. Dividing throughout therein by θ^2, we obtain a different quadratic equation

$$\frac{1}{\theta^2}(\,\|x\|^2 - \epsilon^2) \; - \; \frac{2}{\theta}\,\langle x \, , \, \phi\,(h) \rangle \; + \; \|\phi\,(h)\|^2 = 0.$$

Selecting the larger root (and hence smaller θ) gives us that for every $h \in \mathcal{K}(\epsilon)$, the optimal value of (A-1) (and (A-5)) is

$$\frac{\|x\|^2 - \epsilon^2}{\langle x \, , \, \phi\,(h) \rangle \; + \; \|\phi\,(h)\| \, \sqrt{\epsilon^2 - e\,(h)}} = \eta\,(h)\,.$$

Alternatively, if one selects the smaller root of the quadratic equation $\|x - \theta\phi(h)\|^2 = \epsilon^2$, one gets the expression of eta provided in 3-6. □

Claim A-1.4 (Optimality condition)**.** *The first order necessary and sufficient optimality condition* $0 = \frac{\partial}{\partial\lambda} L(\lambda, h)$ *for the maximization problem* (A-1) *has a unique solution (if it admits)* $\lambda(h)$ *as given in* (A-3)*, and it satisfies*

$$\eta(h) \;=\; L(\lambda(h), h) \;=\; \sqrt{\langle \lambda(h)\,,\, x\rangle - \epsilon\,\|\lambda(h)\|}. \tag{A-6}$$

Proof. Rearranging terms in the optimality condition $0 = \frac{\partial}{\partial\lambda} L(\lambda, h)$, we obtain

$$\frac{\epsilon}{\|\lambda(h)\|}\lambda(h) \;=\; x \;-\; \sqrt{\langle \lambda(h)\,,\, x\rangle - \epsilon\,\|\lambda(h)\|}\;\phi(h). \tag{A-7}$$

By taking inner product with $\lambda(h)$ throughout in (A-7), it is readily seen that $\sqrt{\langle \lambda(h)\,,\, x\rangle - \epsilon\,\|\lambda(h)\|} = \langle \lambda(h)\,,\, \phi(h)\rangle$. Using this equality to evaluate $L(\lambda(h), h)$ and observing that the optimal value of (A-1) is equal to $\eta(h)$ gives us (A-6).[1]

Since $\eta(h)$ is known, using (A-6) together with (A-7) allows us to solve for $\lambda(h)$ in (A-7). We first see that $\lambda(h)$ satisfies the optimality condition (A-7) if and only if there exists some $r > 0$ such that $\lambda(h) = r(x - \eta(h)\,\phi(h))$. Observe that

$$\begin{aligned}
\eta(h) &= \sqrt{\langle \lambda(h)\,,\, x\rangle - \epsilon\,\|\lambda(h)\|} \\
&= \sqrt{r}\sqrt{\langle x - \eta(h)\,\phi(h)\,,\, x\rangle - \epsilon\,\|x - \eta(h)\,\phi(h)\|} \\
&= \sqrt{r}\sqrt{\|x - \eta(h)\,\phi(h)\|^2 + \langle x - \eta(h)\,\phi(h)\,,\, \eta(h)\,\phi(h)\rangle - \epsilon^2} \\
&= \sqrt{r\eta(h)}\sqrt{\langle x\,,\, \phi(h)\rangle - \eta(h)\,\|\phi(h)\|^2}.
\end{aligned}$$

Observing that $\langle x\,,\, \phi(h)\rangle - \eta(h)\,\|\phi(h)\|^2 = \|\phi(h)\|\,\sqrt{\epsilon^2 - e(h)}$, we finally conclude that the optimality condition (A-7) has a unique solution $\lambda(h)$ given by

$$\lambda(h) = \frac{\eta(h)}{\|\phi(h)\|\,\sqrt{\epsilon^2 - e(h)}}(x - \eta(h)\,\phi(h)).$$

□

To complete the proof of the proposition, it now only remains to be shown that the maximization problem (A-1) admits a unique optimal solution $\lambda(h)$ if and only if $h \in \mathrm{int}(\mathcal{K}(\epsilon))$. Firstly, if indeed $h \in \mathrm{int}(\mathcal{K}(\epsilon))$, then it is clear that $\lambda(h)$ satisfies both the feasibility condition $0 < \sqrt{\langle \lambda(h)\,,\, x\rangle - \epsilon\,\|\lambda(h)\|}$ (it follows from (A-6)) and the fist order optimality conditions (it follows from Claim A-1.4). Secondly, if $h \in \mathcal{K}(\epsilon) \setminus \mathrm{int}(\mathcal{K}(\epsilon))$, we have $e(h) = \epsilon^2$. Now, if a solution $\lambda(h)$ exists, it *must* be of the form in (A-3) and should satisfy (A-6). We see that both of these conditions fail and hence a solution to the maximization problem (A-1) cannot exist. the proof of the proposition is now complete. □

Proof of Lemma 3-1.4. From assertion (i) of Proposition A-1.1, it is inferred that $\eta : \mathcal{K}(\epsilon) \longrightarrow [0, +\infty)$ is a pointwise maximum of the linear function $L(\lambda, h)$, hence convex. We also observe

[1] Observe that by evaluating squared norm on both sides of (A-7) and using (A-6) also gives rise to the quadratic equation $\epsilon^2 = \|x - \eta(h)\,\phi(h)\|^2$ for $\eta(h)$.

from assertion (ii) of Proposition A-1.1 that the maximization problem (A-1) admits a solution $\lambda(h)$ if and only if $h \in \mathcal{K}(\bar{\epsilon})$. From Danskin's theorem [16], we conclude that the function $\eta : \mathcal{K}(\epsilon) \longrightarrow [0, +\infty)$ is differentiable if and only if the maximizer $\lambda(h)$ in (A-1) exists. Thus, $\eta : \mathcal{K}(\epsilon) \longrightarrow [0, +\infty)$ is differentiable at every $h \in \mathcal{K}(\bar{\epsilon})$, and the derivative is given by $\nabla\eta(h) = -\phi^a(\lambda(h))$. Substituting for $\lambda(h)$ from (A-3), we immediately get (3-7). $\square$

Proof of proposition 3-1.2. The original problem (2-1) was reformulated as a min-max problem in [42, Theorem 10, p. 10]. Denoting $\Lambda := \{\lambda \in \mathbb{R}^{n_x} : \langle \lambda\,,\, x\rangle - \epsilon \|\lambda\| > 0\}$, and by considering $r = 2, q = 0.5, \delta = 0$ in [42, Theorem 10] we see that the LIP (2-1) is equivalent to the min-max problem

$$\begin{cases} \min\limits_{h \,\in\, B_c} \; \sup\limits_{\lambda \,\in\, \Lambda} \; L(\lambda, h) \,. \end{cases} \tag{A-8}$$

Moreover, from [42, Theorem 10, assertion (ii)-a], it also follows that $h^* \in \arg\min\limits_{h \,\in\, B_c} \left\{ \sup\limits_{\lambda \,\in\, \Lambda} \; L(\lambda, h) \right\}$ if and only if c^*h^* is an optimal solution to the LIP (2-1). In view of Proposition A-1.1, solving the maximization problem over λ immediately implies that

$$h^* \in \arg\min_{h \in B_c \cap \mathcal{K}(\epsilon)} \; \eta(h) \,,$$

if and only if c^*h^* is an optimal solution to the LIP (2-1). The proof is now complete. $\square$

A-2 Proofs for Smoothness

Lemma A-2.1. *Let us consider the mapping $\Lambda \ni \lambda \longmapsto l(\lambda) := \sqrt{\langle \lambda\,,\, x\rangle - \epsilon \|\lambda\|}$, and let $H(\lambda)$ be its* hessian *evaluated at $\lambda \in \Lambda$. Then the smallest and largest absolute values, $\bar{\sigma}$ and $\widehat{\sigma}$ respectively, of the eigenvalues of $H(\lambda)$ are given by*

$$\begin{cases} \bar{\sigma} \;=\; \dfrac{\left(\|x\|^2 - \epsilon^2\right)}{8(l(\lambda))^3} \left(1 \;-\; \sqrt{1 \;-\; \dfrac{8\epsilon(l(\lambda))^6}{\left(\|x\|^2 - \epsilon^2\right)^2 \|\lambda\|^3}}\right) \\ \widehat{\sigma} \;=\; \dfrac{\left(\|x\|^2 - \epsilon^2\right)}{8(l(\lambda))^3} \left(1 \;+\; \sqrt{1 \;-\; \dfrac{8\epsilon(l(\lambda))^6}{\left(\|x\|^2 - \epsilon^2\right)^2 \|\lambda\|^3}}\right). \end{cases} \tag{A-9}$$

Proof. First of all, we observe that since $\lambda \longmapsto l(\lambda)$ is differentiable everywhere on Λ, and the gradients are given by $\nabla l(\lambda) = \frac{1}{2l(\lambda)}\left(x - \frac{\epsilon}{\|\lambda\|}\lambda\right)$. Differentiating again w.r.t. λ, we easily verify that the hessian is given by

$$H(\lambda) = \frac{-\epsilon}{2l(\lambda)\|\lambda\|}\left(\mathbb{I} \;-\; \frac{1}{\|\lambda\|^2}\lambda\lambda^\top\right) \;-\; \frac{1}{4(l(\lambda))^3}\left(x - \frac{\epsilon}{\|\lambda\|}\lambda\right)\left(x - \frac{\epsilon}{\|\lambda\|}\lambda\right)^\top . \tag{A-10}$$

First, suppose that λ and x are linearly independent, observe that the subspace $S := \operatorname{span}\{\lambda, x - \frac{\epsilon}{\|\lambda\|}\lambda\}$ is invariant under the linear transformation given by the hessian matrix $H(\lambda)$, and it is identity on the orthogonal complement of S. Then it is evident that the hessian has $n_x - 2$ eigenvalues equal to $-\epsilon/l(\lambda)\|\lambda\|$ and the two other distinct eigenvalues corresponding to the restriction of $H(\lambda)$ onto S. Selecting $\{\lambda,\ x - \frac{\epsilon}{\|\lambda\|}\lambda\}$ as a basis for S, the linear mapping of the hessian is given by the matrix

$$T = \begin{pmatrix} 0 & \frac{\epsilon l(\lambda)}{2\|\lambda\|^3} \\ \frac{-1}{4l(\lambda)} & \frac{-\left(\|x\|^2-\epsilon^2\right)}{4(l(\lambda))^3} \end{pmatrix}. \tag{A-11}$$

It is a straightforward exercise to verify that $-\bar{\sigma}$ and $-\widehat{\sigma}$ are indeed the two distinct eigenvalues of T and consequently, the remaining two eigenvalues of the hessian $H(\lambda)$. Since the rest of the eigenvalues are $-\epsilon/l(\lambda)\|\lambda\|$, it remains to be shown that $\bar{\sigma} \leqslant \epsilon/2l(\lambda)\|\lambda\| \leqslant \widehat{\sigma}$. We establish it by producing $u_1, u_2 \in S$ such that

$$\bar{\sigma} \ \leqslant \ \frac{|\langle u_1 \ , \ H(\lambda)u_1\rangle|}{\|u_1\|^2} \ \leqslant \ \frac{\epsilon}{2l(\lambda)\,\|\lambda\|} \ \leqslant \ \frac{|\langle u_2 \ , \ H(\lambda)u_2\rangle|}{\|u_2\|^2} \ \leqslant \ \widehat{\sigma}. \tag{A-12}$$

Observe that the inequalities $\bar{\sigma} \ \leqslant \ \frac{|\langle u_1 \ , \ H(\lambda)u_1\rangle|}{\|u_1\|^2}$, and $\frac{|\langle u_2 \ , \ H(\lambda)u_2\rangle|}{\|u_2\|^2} \ \leqslant \ \widehat{\sigma}$ readily hold for any $u_1, u_2 \in S$ since $-\bar{\sigma}, -\widehat{\sigma}$ are the two eigenvalues of $H(\lambda)$ when restricted to the subspace S. Considering $u_1 = (l(\lambda))^2 x + \left(\|x\|^2 - \frac{\epsilon\langle\lambda \ , \ x\rangle}{\|\lambda\|}\right)\lambda$ and $u_2 = \lambda - \frac{\|\lambda\|^2}{\langle\lambda \ , \ x\rangle}$, it is easily verified that $\left\langle x - \frac{\epsilon}{\|\lambda\|}\lambda \ , \ u_1\right\rangle = 0$, and $\langle\lambda \ , \ u_2\rangle = 0$. Moreover, we also get the inequalities

$$\begin{cases} \langle u_1 \ , \ H(\lambda)u_1\rangle \ = \ \dfrac{-\epsilon}{2l(\lambda)\,\|\lambda\|}\,\|u_1\|^2 + \dfrac{\epsilon}{2l(\lambda)\,\|\lambda\|}\dfrac{|\langle\lambda \ , \ u_1\rangle|^2}{\|\lambda\|^2} & \geqslant \dfrac{-\epsilon}{2l(\lambda)\,\|\lambda\|}\,\|u_1\|^2, \\ \langle u_2 \ , \ H(\lambda)u_2\rangle \ = \ \dfrac{-\epsilon}{2l(\lambda)\,\|\lambda\|}\,\|u_2\|^2 - \dfrac{1}{4(l(\lambda))^3}\left|\left\langle x - \dfrac{\epsilon}{\|\lambda\|}\lambda \ , \ u_2\right\rangle\right|^2 & \leqslant \dfrac{-\epsilon}{2l(\lambda)\,\|\lambda\|}\,\|u_2\|^2. \end{cases}$$

Since the hessian $H(\lambda)$ is negative semidefinite, the inequalities (A-12) are obtained at once.

To complete the proof for the case when λ and x are linearly dependent, we first see that the expressions in (A-9) are continuous w.r.t. λ. Also, it is evident that the mapping $\Lambda \ni \lambda \longmapsto H(\lambda)$ is continuous. Since the eigenvalues of a matrix vary continuously, these two limits must be the same. The proof is now complete. □

Lemma A-2.2. *For every $\bar{\eta} > c^*$ and $\bar{\epsilon} \in (0, \epsilon)$, let $C(\bar{\epsilon}, \widehat{\eta}) := \frac{\|x\|-\epsilon}{\epsilon\widehat{\eta}}\sqrt{\epsilon^2 - \bar{\epsilon}^2}$. Consider the convex set*

$$\Lambda(\bar{\epsilon}, \widehat{\eta}) := \{\lambda \in \Lambda : \ l(\lambda) \geqslant C(\bar{\epsilon}, \widehat{\eta})\,\|\lambda\|\}, \tag{A-13}$$

then the mapping $l(\lambda) : \Lambda(\bar{\epsilon}, \widehat{\eta}) \longrightarrow (0, +\infty)$ is $\frac{\epsilon C(\bar{\epsilon},\widehat{\eta})^3}{2\left(\|x\|^2-\epsilon^2\right)}$ strongly concave.

Proof of Lemma A-2.2. Firstly, since $\lambda \longmapsto l(\lambda)$ is concave and $C(\bar{\epsilon}, \widehat{\eta}) > 0$, it is obvious that $\Lambda(\bar{\epsilon}, \widehat{\eta})$ is a convex set. Secondly, we know that $\sqrt{1-\theta^2} < 1 - \frac{\theta^2}{2}$ for every $\theta \in [0, 1]$. Using this inequality, we have

$$1 - \sqrt{1 \ - \ \frac{8\epsilon(l(\lambda))^6}{\left(\|x\|^2 - \epsilon^2\right)^2\|\lambda\|^3}} \ > \ \frac{4\epsilon(l(\lambda))^6}{\left(\|x\|^2 - \epsilon^2\right)^2\|\lambda\|^3}.$$

Thus, we see that $\bar{\sigma} > \frac{\epsilon l(\lambda)^3}{2\left(\|x\|^2-\epsilon^2\right)\|\lambda\|^3} \geqslant \frac{\epsilon C(\bar{\epsilon},\widehat{\eta})^3}{2\left(\|x\|^2-\epsilon^2\right)}$. Since $\bar{\sigma}$ was the smallest absolute value among the eigenvalues of the hessian $H(\lambda)$, the claim holds. □

Lemma A-2.3. *For every $\bar{\eta} > c^*$ and $\bar{\epsilon} \in (0, \epsilon)$, if $\mathcal{H}(\bar{\epsilon}, \widehat{\eta}) := \{h \in B_c \cap \mathcal{K}(\bar{\epsilon}) : \eta(h) \leqslant \widehat{\eta}\}$, then for every $h \in \mathcal{H}(\bar{\epsilon}, \widehat{\eta})$, we have $\lambda(h) \in \Lambda(\bar{\epsilon}, \widehat{\eta})$, and thus,*

$$\eta(h) \;=\; \max_{\lambda \in \Lambda(\bar{\epsilon}, \widehat{\eta})} \; L(\lambda, h) \;\; \textit{for every } h \in \mathcal{H}(\bar{\epsilon}, \widehat{\eta}). \tag{A-14}$$

Proof. For every $h \in \mathcal{H}(\bar{\epsilon}, \widehat{\eta}) \subset \mathcal{H}$, first of all, we know from Proposition A-1.1 that $\lambda(h) = \arg\max_{\lambda \in \Lambda} \; L(\lambda, h)$ for $\lambda(h)$ as in (A-3). On the one hand, for every $h \in \mathcal{H}(\bar{\epsilon}, \widehat{\eta})$, we have the inequality

$$\begin{aligned}
\frac{l(\lambda(h))}{\|\lambda(h)\|} \;&=\; \frac{\eta(h)}{\|\lambda(h)\|} \text{ since } l(\lambda(h)) = \eta(h) \text{ from (A-6)} \\
&=\; \frac{\|\phi(h)\| \sqrt{\epsilon^2 - e(h)}}{\|x - \eta(h)\, \phi(h)\|} \;=\; \frac{\|\phi(h)\|}{\epsilon} \sqrt{\epsilon^2 - e(h)}, \text{ from (3-5)}, \\
&>\; \|\phi(h)\| \, \frac{\sqrt{\epsilon^2 - \bar{\epsilon}^2}}{\epsilon}, \text{ since } e(h) < \bar{\epsilon}^2 \text{ for } h \in \mathcal{H}(\bar{\epsilon}, \widehat{\eta}).
\end{aligned} \tag{A-15}$$

On the other hand, for every $h \in \mathcal{H}(\bar{\epsilon}, \widehat{\eta})$, we also see from (3-3) that

$$\begin{aligned}
\frac{1}{\eta(h)} &= \frac{\|\phi(h)\|}{(\|x\|^2 - \epsilon^2)} \Big(\frac{\langle x \,,\, \phi(h) \rangle}{\|\phi(h)\|} + \sqrt{\epsilon^2 - e(h)} \Big) \\
&< \frac{\|\phi(h)\|}{(\|x\|^2 - \epsilon^2)} (\|x\| + \epsilon) \text{ from the C-S inequality, and } e(h) < \epsilon^2, \\
&= \frac{\|\phi(h)\|}{\|x\| - \epsilon},
\end{aligned} \tag{A-16}$$

which immediately gives the lower bound $\|\phi(h)\| > \frac{\|x\| - \epsilon}{\eta(h)}$. Employing this lower bound in (A-15), we finally have

$$\frac{l(\lambda(h))}{\|\lambda(h)\|} \;>\; \frac{\|x\| - \epsilon}{\eta(h)} \, \frac{\sqrt{\epsilon^2 - \bar{\epsilon}^2}}{\epsilon} \;>\; \frac{\|x\| - \epsilon}{\epsilon \widehat{\eta}} \sqrt{\epsilon^2 - \bar{\epsilon}^2} \;=\; C(\bar{\epsilon}, \widehat{\eta}).$$

Thus, it follows that for every $h \in \mathcal{H}(\bar{\epsilon}, \widehat{\eta})$, the corresponding maximizer $\lambda(h)$ of (A-1) satisfies the inclusion $\lambda(h) \in \Lambda(\bar{\epsilon}, \widehat{\eta})$. Moreover, since $\lambda(h) = \arg\max_{\lambda \in \Lambda} \; L(\lambda, h)$, and $\Lambda(\bar{\epsilon}, \widehat{\eta}) \subset \Lambda$, the inclusion $\lambda(h) \in \Lambda(\bar{\epsilon}, \widehat{\eta})$ also immediately implies that

$$\eta(h) \;=\; \max_{\lambda \in \Lambda} \; L(\lambda, h) \;=\; \max_{\lambda \in \Lambda(\bar{\epsilon}, \widehat{\eta})} \; L(\lambda, h).$$

The proof is now complete. □

Proof of Proposition 3-1.5. Let us define

$$(-l)^*(y) := \sup_{\lambda \in \Lambda(\bar{\epsilon}, \widehat{\eta})} \; l(\lambda) + \langle \lambda \,,\, y \rangle, \tag{A-17}$$

and let $\mathrm{dom}((-l)^*) = \{y : (-l)^*(y) < +\infty\}$. From Lemma A-2.2, we know that the mapping $\Lambda(\bar{\epsilon}, \widehat{\eta}) \ni \lambda \longmapsto -l(\lambda)$ is $\frac{\epsilon C(\bar{\epsilon},\widehat{\eta})^3}{2(\|x\|^2-\epsilon^2)}$-strongly convex, and therefore, the mapping $\mathrm{dom}((-l)^*) \ni y \longmapsto (-l)^*(y)$ is $\frac{2(\|x\|^2-\epsilon^2)}{\epsilon C(\bar{\epsilon},\widehat{\eta})^3}$-smooth. In other words, we have the inequality

$$\|\nabla(-l)^*(y) - \nabla(-l)^*(y')\| \leqslant \frac{2(\|x\|^2-\epsilon^2)}{\epsilon C(\bar{\epsilon},\widehat{\eta})^3}\|y-y'\| \text{ for all } y, y' \in \mathrm{dom}((-l)^*). \tag{A-18}$$

For every $h \in \mathcal{H}(\bar{\epsilon}, \widehat{\eta})$, we see from (A-14), that

$$\begin{aligned}
\eta(h) &= \max_{\lambda\in\Lambda(\bar{\epsilon},\widehat{\eta})} L(\lambda, h) \\
&= \max_{\lambda\in\Lambda(\bar{\epsilon},\widehat{\eta})} 2l(\lambda) - \langle \lambda \,,\, \phi(h)\rangle \\
&= 2(-l)^*(-0.5\phi(h)).
\end{aligned}$$

Clearly, $h \in \mathcal{H}(\bar{\epsilon}, \widehat{\eta})$ if and only if $-0.5\phi(h) \in \mathrm{dom}((-l)^*)$ and for every such h, we also have $\nabla\eta(h) = -\phi^a\Big(\nabla(-l)^*(-0.5\phi(h))\Big)$. Moreover, due to smoothness of $(-l)^*$, it immediately follows from (A-18) that

$$\begin{aligned}
\|\nabla\eta(h) - \nabla\eta(h')\| &\leqslant \|\phi^a\| \, \|\nabla(-l)^*(-0.5\phi(h)) - \nabla(-l)^*(-0.5\phi h')\| \\
&\leqslant \|\phi^a\| \frac{2(\|x\|^2-\epsilon^2)}{\epsilon C(\bar{\epsilon},\widehat{\eta})^3}\|0.5\phi(h-h')\| \\
&\leqslant \frac{\|\phi\|_o^2(\|x\|^2-\epsilon^2)}{\epsilon C(\bar{\epsilon},\widehat{\eta})^3}\|h-h'\| \ .
\end{aligned}$$

Finally, substituting for $C(\bar{\epsilon}, \widehat{\eta})$ from Lemma A-2.2, we get the inequality (3-10). □

A-3 Proofs for Stepsize Selection

Let us consider the following optimization problem in which we find the optimal step size given the explicit formulation of $\eta(h)$

$$\gamma(h) = \begin{cases} \arg\min_{\gamma\in[0,\widehat{\gamma}(h)]} \eta(h + \gamma d(h)). \end{cases} \tag{A-19}$$

The explicit solution of this optimization problem can be found without much additional computations.

Proof of proposition 4-1.4. The maximum stepsize $\widehat{\gamma}(h)$ must satisfy $h_\gamma \in \mathcal{K}(\bar{\epsilon})$. Recall that that $h \in \mathcal{K}(\bar{\epsilon})$ if and only if $\langle x \,,\, \phi(h)\rangle \geqslant \|\phi(h)\| \sqrt{(\|x\|^2 - \bar{\epsilon}^2)}$. This gives rise to the following inequality $\langle x \,,\, \phi(h_\gamma)\rangle \geqslant \|\phi(h_\gamma)\| \sqrt{(\|x\|^2 - \bar{\epsilon}^2)}$ which is of the form $q(\gamma) \leqslant 0$ with

$$q(\gamma) = a\gamma^2 + b\gamma + c, \tag{A-20}$$

with the definitions for a, b and c as described in proposition 4-1.4. Three different domains for a are considered separately

1. $a > 0$. Since $c \geqslant 0$, we have that $\frac{a}{c} > 0$. Note that the domain must be continuous since $\eta\,(h)$ is a convex function and therefore $q(\gamma) \nless 0$. This results in the domain $\gamma \in [0, +\infty]$.

2. $a = 0$. Since $c \geqslant 0$ we have $|\,\langle x\ ,\ \phi(h)\rangle\,| \geqslant \sqrt{\|x\|^2 - \epsilon^2}\,\|\phi(h)\|$. While $a = 0$, we have $|\,\langle x\ ,\ \phi(d)\rangle\,| = \sqrt{\|x\|^2 - \epsilon^2}\,\|\phi(d)\|$. Combining these two equations and applying Cauchy-Schwartz we get
$|\,\langle x\ ,\ \phi(h)\rangle\,|\,|\,\langle x\ ,\ \phi(d)\rangle\,| \geqslant (\|x\|^2 - \epsilon^2)\,\langle\phi(d)\ ,\ \phi(h)\rangle$. From this equation it is obvious that we have $b \geqslant 0$ and therefore $\gamma = \frac{-c}{2b} < 0$. This gives the following domain $\gamma \in [0, +\infty]$.

3. $a < 0$. Since $c \geqslant 0$, the product of the roots of quadratic equation $q(\gamma) = 0$ are real and of opposite signs. Therefore, $\widehat{\gamma}(h)$ can be picked as the positive root of the quadratic equation $q(\gamma) = 0$, given by $\gamma \in [0, \frac{-b-\sqrt{b^2-4ac}}{2a}]$.

$\square$

Proof. Setting $\frac{\partial\eta(h_\gamma)}{\partial\gamma}$ equal to zero will give a quadratic function of the form $r\gamma^2 + s\gamma + u = 0$. If the solution to this quadratic function lies between the bounds of γ, which is the case if $\frac{\partial\eta(h_\gamma)}{\partial\gamma}|_{\gamma=0} \leqslant 0$ and $\frac{\partial\eta(h_\gamma)}{\partial\gamma}|_{\gamma=\widehat{\gamma}(h)} \geqslant 0$ hold, we define $\gamma = \frac{-s\pm\sqrt{s^2-4ru}}{2r}$. If the solution is negative, we define $\gamma = 0$ and if the $\gamma \geqslant \widehat{\gamma}(h)$ we define $\gamma = \widehat{\gamma}(h)$.

$\square$

Bibliography

[1] Manya V. Afonso, José M. Bioucas-Dias, and Mário A. T. Figueiredo. An Augmented Lagrangian Approach to the Constrained Optimization Formulation of Imaging Inverse Problems. *IEEE Transactions On Image Processing*, 2009.

[2] Michal Aharon, Michael Elad, and Alfred Bruckstein. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54(11):4311–4322, 2006.

[3] Chenglong Bao, Jian Feng Cai, and Hui Ji. Fast sparsity-based orthogonal dictionary learning for image restoration. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3384–3391. Institute of Electrical and Electronics Engineers Inc., 2013.

[4] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

[5] Jean-Michel Bismut and Ing Zieur. Conjugate Convex Functions in Optimal Stochastic Control. Technical report, Journal of Mathematical Analysis and Applications, 1973.

[6] Ashish Bora, Ajil Jalal, Eric Price, and Alexandros G. Dimakis. Compressed Sensing using Generative Models. In *Proceedings of the 34th International Conference on Machine Learning*, pages 537–546, 3 2017.

[7] Stephen Boyd, Lin Xiao, and Almir Mutapcic. Subgradient Methods. Technical report, Notes for EE392o, StanfordUniversity, Autumn,2003, 2003.

[8] Emmanuel J. Candès and Benjamin Recht. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9(6):717–772, 12 2009.

[9] T. Chai and R. R. Draxler. Root mean square error (RMSE) or mean absolute error (MAE)? -Arguments against avoiding RMSE in the literature. *Geoscientific Model Development*, 7(3):1247–1250, 6 2014.

[10] Antonin Chambolle and Thomas Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. Technical report, HAL open science, 2010.

[11] Antonin Chambolle and Thomas Pock. On the ergodic convergence rates of a first-order primaldual algorithm. *Mathematical Programming*, 159(1-2):253–287, 9 2016.

[12] Venkat Chandrasekaran, Benjamin Recht, Pablo A. Parrilo, and Alan S. Willsky. The Convex Geometry of Linear Inverse Problems. *Found Comput Math*, 2010.

[13] Xuefeng Chen, Zhaohui Du, Jimeng Li, Xiang Li, and Han Zhang. Compressed sensing based on dictionary learning for extracting impulse components. *Signal Processing*, 96(PART A):94–109, 2014.

[14] Mads Græsbøll Christensen, Jan Østergaard, and Søren Holdt Jensen. On compressed sensing and its application to speech and audio signals. In *Conference Record - Asilomar Conference on Signals, Systems and Computers*, pages 356–360, 2009.

[15] Florent Couzinie-Devy, Julien Mairal, Francis Bach, and Jean Ponce. Dictionary Learning for Deblurring and Digital Zoom. *CoRR*, abs/1110.0957, 2011.

[16] Dimitri Panteli Bertsekas. Control of uncertain systems with a set-membership description of the uncertainty. Technical report, Massachusetts Institute of Technology, 1971.

[17] David L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 4 2006.

[18] David L Donoho. For Most Large Underdetermined Systems of Linear Equations the Minimal 1-norm Solution Is Also the Sparsest Solution. *Communications on Pure and Applied Mathematics*, 59:797–829, 2006.

[19] Dr. -Ing. Hongwei Zheng. Boat. Technical report, https://cms.uni-konstanz.de/fileadmin/archive/informatik-saupe/fileadmin/informatik/ag-saupe/Webpages/lehre/dip_w0910/demos.html, 2009.

[20] John Duchi, Shaitti-C Org, and Tushar Chandra. Efficient Projections onto the l1-Ball for Learning in High Dimensions Google, Mountain View, CA 94043 Shai Shalev-Shwartz. Technical report, Proceedings of the 25th International Conference on Machine Learning, 2008.

[21] Armin Eftekhari and Peyman Mohajerin Esfahani. The Nonconvex Geometry of Linear Inverse Problems. *arXiv*, 1 2021.

[22] Michael Elad and Michal Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image Processing*, 15(12):3736–3745, 2006.

[23] Marguerite Frank and Wolfel Pr. An algorithm for quadratic programming. *Naval Research Logistics*, 3 1956.

[24] Aurelien Garivier. Machine Learning 8: Convex Optimization for Machine Learning. Technical report, Université de Lyon, 2018.

[25] Sivan Gleichman and Yonina C. Eldar. Blind Compressed Sensing. *IEEE Transactions on Information Theory*, 57(10):6958–6975, 10 2011.

[26] Alain Horé and Djemel Ziou. Image quality metrics: PSNR vs. SSIM. In *Proceedings - International Conference on Pattern Recognition*, pages 2366–2369, 2010.

[27] Ori Bryt Inbal Horev and Ron Rubinseitn. *Adaptive Image Compression using Sparse Dictionaries*. IEEE, 2012.

[28] Martin Jaggi. Revisiting Frank-Wolfe: Projection-Free Sparse Convex Optimization. Technical report, Ecole Polytechnique, 2013.

[29] Kenneth Kreutz-Delgado, Joseph F Murray, Bhaskar D Rao, Kjersti Engan, Te-Won Lee, and Terrence J Sejnowski. Dictionary Learning Algorithms for Sparse Representation. *Neural Computation*, 15(2):349–396, 2003.

[30] Shutao Li, Haitao Yin, and Leyuan Fang. Group-sparse representation with dictionary learning for medical image denoising and fusion. *IEEE Transactions on Biomedical Engineering*, 59(12):3450–3459, 2012.

[31] Michael Lustig, David Donoho, and John M. Pauly. Sparse MRI: The application of compressed sensing for rapid MR imaging. *Magnetic Resonance in Medicine*, 58(6):1182–1195, 12 2007.

[32] Julien Mairal, Francis Bach FRANCISBACH, Jean Ponce, Ecole Normale Supérieure, and Guillermo Sapiro. Online Dictionary Learning for Sparse Coding. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 689–696, 2009.

[33] Julien Mairal, Guillermo Sapiro, and Michael Elad. Learning Multiscale Sparse Representations for Image and Video Restoration. *Multiscale Modeling & Simulation*, 7(1):214–241, 2007.

[34] Bruno A Olshausen and David J Fieldt. Sparse Coding with an Overcomplete Basis Set: A Strategy Employed by V1 ? Technical Report 23, Vision Res., 1997.

[35] Boaz Ophir, Michael Lustig, and Michael Elad. Multi-scale dictionary learning using wavelets. *IEEE Journal on Selected Topics in Signal Processing*, 5(5):1014–1024, 2011.

[36] N Parikh, S Boyd, Neal Parikh, and Stephen Boyd. Proximal Algorithms. Technical Report 3, Foundations and Trends in Optimization, 2013.

[37] Luisa F. Polania and Kenneth E. Barner. Multi-scale dictionary learning for compressive sensing ECG. In *2013 IEEE Digital Signal Processing and Signal Processing Education Meeting, DSP/SPE 2013 - Proceedings*, pages 36–41. IEEE Computer Society, 2013.

[38] Ron Rubinstein, Alfred M. Bruckstein, and Michael Elad. Dictionaries for sparse representation modeling. *Proceedings of the IEEE*, 98(6):1045–1057, 2010.

[39] Ron Rubinstein, Michael Zibulevsky, and Michael Elad. Efficient Implementation of the K-SVD Algorithm Using Batch Orthogonal Matching Pursuit. Technical report, Technion - Computer Science Department, 2008.

[40] Ron Rubinstein, Michael Zibulevsky, and Michael Elad. Double sparsity: Learning sparse dictionaries for sparse signal approximation. *IEEE Transactions on Signal Processing*, 58(3 PART 2):1553–1564, 2010.

[41] Jerome M. Shapiro. Embedded Image Coding Using Zerotrees of Wavelet Coefficients. *IEEE Transactions on Signal Processing*, 41(12):3445–3462, 1993.

[42] Mohammed Rayyan Sheriff and Debasish Chatterjee. Novel min-max reformulations of Linear Inverse Problems. *arXiv preprint arXiv:2007.02448.*, 7 2020.

[43] Jeremias Sulam and Michael Elad. Large inpainting of face images with Trainlets. *IEEE Signal Processing Letters*, 23(12):1839–1843, 2016.

[44] Jeremias Sulam, Boaz Ophir, Michael Zibulevsky, and Michael Elad. Trainlets: Dictionary Learning in High Dimensions. *CoRR*, 2016.

[45] Georg Taubock, Shristi Rajbamshi, and Peter Balazs. Dictionary Learning for Sparse Audio Inpainting. *IEEE Journal on Selected Topics in Signal Processing*, 15(1):104–119, 1 2021.

[46] Joel A. Tropp and Anna C. Gilbert. Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Transactions on Information Theory*, 53(12):4655–4666, 2007.

[47] John Wright, Allen Y. Yang, Arvind Ganesh, S. Shankar Sastry, and Yi Ma. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(2):210–227, 2009.

[48] Zongben Xu and Jian Sun. Image inpainting by patch propagation using patch sparsity. *IEEE Transactions on Image Processing*, 19(5):1153–1165, 2010.

[49] Wei H Yang. On generalized holder inequality. Technical Report 5, Nonlinear Analysis Theory, Methods & Applications, Great Britain, 1991.

[50] Ye. E. Nesterov. A method of solving a convex programming problem with convergence rate O(1/kˆ2). *Soviet Math dokl.*, 27(2), 1983.

[51] Qiang Zhang and Baoxin Li. Discriminative K-SVD for dictionary learning in face recognition. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2691–2698, 2010.

Glossary

List of Acronyms

ADMM	Alternating Direction Method of Multipliers
ALM	Augmented Lagrangian Method
APGD	Accelerated Projected Gradient Descent
BPDN	Basis Pursuit Denoising
CP	Chambolle-Pock
CS	Compressed Sensing
C-SALSA	Constrained Split Augmented Lagrangian Shrinkage Algorithm
DCT	Discrete Cosine Transform
DL	Dictionary Learning
FISTA	Fast Iterative Shrinkage-Thresholding Algorithm
FLIPS	Fast Linear Inverse Problem Solver
LIP	Linear Inverse Problem
ISTA	Iterative Shrinkage-Thresholding Algorithm
OMP	Orthogonal Matching Pursuit
PGD	Projected Gradient Descent
PSNR	Peak-Signal-to-Noise Ratio
RMSE	Root Mean Square Error
SGD	Sub-Gradient Descent
SVD	Singular-Value-Decomposition