Università degli Studi dell'Insubria
Dipartimento di Scienza e Alta Tecnologia (DiSAT)
Ph.D. program in Computer Science and the Mathematics of Computation
XXXVII Cycle

Graph Laplacian–Based Strategies and Convex Optimization via Primal–Dual Methods

Doctoral dissertation of:
Stefano Aleotti

Supervisor:
Prof. Marco Donatelli
Coordinator:
Prof.ssa Barbara Carminati

Thesis submitted in October, 2024

To my two beloved grandfathers, Luciano: one who raised me but left before seeing me begin my studies, and the other who supported me but could not witness its completion.

Abstract

This thesis focuses on the analysis of different variational approaches for solving inverse problems. In the first part, we examine the graph Laplacian operator within an $\ell^2 - \ell^q$ framework, where $q \leq 1$. A key challenge in using this linear operator is its dependence on an initial reconstruction, which can be obtained through a general reconstruction method. However, we demonstrate that, under very weak assumptions on the chosen reconstruction method, the resulting strategy is both convergent and stable, achieving high–quality final reconstructions. Additionally, we analyze the fractional graph Laplacian operator, showing that the use of fractional powers can surpass the standard approach by providing more detailed final images.

The second part of this thesis considers a more general framework, where the optimization problem consists of the sum of a differentiable term and a non–smooth but convex term. The variable metric approach we propose results in a convergent method that fixes a priori the number of nested iterations required to compute inexact approximations of the proximal gradient step. We also introduce an iterated Tikhonov–based strategy, which accelerates convergence while maintaining high-quality reconstructions. In the context of image deblurring, the variable metric approach can be reinterpreted as a right preconditioning strategy. Therefore, the final section is devoted to the analysis of a left preconditioning approach.

Contents

Introduction

Inverse problems typically arise when we seek to retrive information about internal or otherwise hidden data from external measurements. Such situations occur in various fields such as medicine, biology, engineering, and astronomy [16, 85]. A classic example is X-ray tomography [88, 68, 112, 113], where we aim to infer information about the internal state of tissues using data collected by measuring the intensity variations of X-ray beams as they pass through the patient. In general, given a system and its input, the *forward problem* involves computing the output. In the case of tomography, the forward problem consists of calculating the intensity variations of X-ray beams, which can be measured directly by a detector. The true challenge, however, lies in determining the absorption coefficient of the object from the measured data. This is the *inverse problem*, where the goal is to compute either the input or the system itself, given the other two quantities.

Another common issue in the class of inverse problems is the restoration of blurred and noisy images [82]. The operator associated with the forward problem, known as the blur operator, introduces blur by convolving the "true" object with a specific kernel called the Point Spread Function (PSF). The PSF essentially describes how the pixel intensities of an image are mixed to produce the blur. In real-world applications, the PSF may be measured or sometimes be unknown, leading to what is referred to as *blind deconvolution*. However, for the purposes of this thesis, we assume the PSF is given. Image deblurring involves several crucial aspects that must be carefully considered. For instance, when capturing a picture of an object, we represent only a finite region of the scene, excluding elements outside the field of view (FOV). If the image is blurred, pixels near the boundary may be significantly influenced by elements outside the FOV, which need to be inferred to achieve sharp reconstruction. Imposing Boundary Conditions (BCs) helps approximate what is outside the FOV. In this thesis, we assume periodic BCs, meaning the image repeats itself outside the FOV. This choice is primarily motivated by the fact that BCs affect the structure of the blur operator A. In the case of periodic BCs, we can exploit the Fast Fourier Transform (FFT) to reduce the computational cost of recovering the true object. Nonetheless, the quality of the final reconstruction can be further improved by adopting more sophisticated and realistic BCs, such as reflective [105] and anti-reflective boundary conditions [131].

Another crucial aspect of inverse problems is that, in almost every situation, we do not have access to the true measurement of the output of the forward model but rather some

imprecise and noisy observations. In X-ray tomography, for instance, the intensity variation is affected by the accuracy of the detector. In image deblurring, noise can come from various sources, such as fluctuations during the recording process or background photons.

To better define this class of problems, in the early 20th century, Hadamard introduced the concept of *ill-posed problems*. A linear problem is considered *well-posed* if it satisfies the following

Definition 0.0.1 (Hadamard)**.** *A problem is well-posed if the solution:*

(i) exists;

(ii) is unique;

(iii) depends continuously on the data.

If at least one of Hadamard's conditions is not satisfied, the problem is said to be ill-posed. Violations of conditions (i) and (ii) are typically less problematic, as in real-world applications we are often working with data corrupted by noise. In such cases, the notion of a solution can be relaxed, and if the solution is not unique, we can impose additional constraints (e.g., minimal norm) to recover uniqueness. In contrast, violations of condition (iii) pose significant numerical challenges, as the discretization of continuous ill-posed problems give rise to severely ill-conditioned systems. To address this, several regularization techniques have been proposed, such as Tikhonov's method [135] and its generalizations, Truncated Singular Value Decomposition (TSVD) [37], and iterative regularization methods (e.g., Landweber, conjugate gradient, etc.) [16]. These topics, along with a brief introduction to the image deblurring problem and Computed Tomography (CT), will be partially covered in Chapter 1.

In the last decade, there has been growing interest in nonlocal models and techniques from graph theory [72, 73, 119, 7]. Typically, in the classical general Tikhonov framework, the regularization operator is chosen as a discretization of first- or second-order Euclidean differential operators [129, 61]. However, recent works have further investigated graph-based approaches in the context of image deblurring and computerized tomography, exploiting the graph Laplacian as a regularization operator [20, 29, 19, 103]. The primary drawback of the graph-based approach is that the graph Laplacian operator depends on an initial approximation of the true solution. A key challenge lies in constructing the graph from a signal that accurately approximates the main features of the true solution. However, the available data are often heavily corrupted by noise. Consequently, the graph constructed from such noisy data may lead to poor outcomes in imaging tasks like deblurring or tomographic reconstruction.

In Chapter 2, we address this issue by considering an $\ell^2 - \ell^1$ variational model that employs the graph Laplacian as the regularization operator. Notably, we maintain the dependence of the graph Laplacian on a preliminary approximation of the solution, which can be obtained using any reconstruction method from the literature. This allows the regularization term to be both data-dependent and adaptive to the observed noise. This introduces an additional

layer of complexity to the theoretical analysis of the method. Nevertheless, we have demonstrated that the proposed strategy constitutes a valid regularization method, and we have rigorously established both its convergence and stability properties.

In Chapter 3, we extended the analysis of the graph Laplacian operator by considering its fractional power. Recently, fractional differential operators have been investigated for enhancing diffusion, particularly in denoising problems [5, 141]. Moreover, the fractional graph Laplacian has recently attracted the attention within the community working on complex networks [15, 21], as it allows for the exploration of non-local dynamics that can spread information across the graph. The main drawback of this strategy is that the fractional graph Laplacian is a full matrix even if the graph Laplacian is sparse. Therefore, approximation tools need to be explored to perform computations with the fractional graph Laplacian. To reduce the computational cost associated with the graph-based operator, we employed a spectral approximation strategy proposed in [134]. This approach involves using a few iterations of the Lanczos algorithm to obtain a good approximation of a filtering function of the graph Laplacian, thus significantly improving computational efficiency.
To test its performance, we considered an $\ell^2 - \ell^q$ model [46, 63, 95, 83] with $q \leq 1$, incorporating the fractional graph Laplacian as the regularization operator.

The second part of this thesis focuses on convex optimization for regularized ill-posed problems. We consider more general models consisting of the sum of two terms: one is typically differentiable, while the other is convex but non-smooth. This type of model is central to several inverse problems in imaging, such as deblurring, denoising, super-resolution, and others [8, 16, 44, 58]. Among the various first-order methods used to solve such problems, proximal-gradient methods [12, 54, 56] are particularly advantageous. They offer mild-to-moderate accuracy while keeping the computational cost per iteration low. However, these methods come with two practical limitations. First, when the chosen step length is too small, convergence to the desired solution can be slow. One potential solution is to accelerate the scheme either by computing the proximal-gradient step using a variable metric that incorporates some second-order information of the differentiable part [24, 49, 71, 98], or by introducing an extrapolation step that leverages information from previous iterations [11, 117]. Second, proximal-gradient methods assume that the proximity operator of the non-smooth term can be computed in closed form. This assumption is not valid for several important regularization terms, such as Total Variation and overlapping group Lasso [8]. In Chapter 4, we review key results from convex analysis and smooth and non-smooth optimization, along with a brief introduction to proximal-gradient methods and their inexact variants.

In Chapter 5, we focus on proximal–gradient methods that combine acceleration techniques based on variable metrics and extrapolation, while allowing for inexact proximal evaluations. We address the issue of potentially underestimating the step length parameter by employing a backtracking procedure [12] that dynamically computes it. Furthermore, the use of a variable metric in the computations aims to capture some second-order information of the smooth part of the objective function. Practical choices include the Hessian matrix or

its regularized versions [99, 142], as well as Hessian approximations based on Quasi-Newton strategies [71, 86, 89, 98]. The extrapolation parameter is computed according to a predetermined sequence, initially proposed for smooth problems by Nesterov [114] and later successfully adapted to non-smooth problems by Beck and Teboulle [12]. This approach guarantees an optimal $\mathcal{O}(1/n^2)$ convergence rate for the function values. An additional advantage of the proposed method is that it computes a fixed number of primal-dual iterates to approximate a variable metric proximal-gradient step, taken from the extrapolated iterate. From a theoretical standpoint, we prove the convergence of the sequence of iterates towards a minimum point of the problem, under a relaxed monotonicity assumption on the scaling matrices and a shrinking condition on the extrapolation parameters.

Lastly, in Chapter 6, we demonstrate that, in the context of the image deblurring problem, the variable metric approach can be interpreted as a right preconditioning strategy applied to the linear system of equations associated with the blurring problem. In this light, we analyze the left preconditioning approach, which results in a faster and more computationally efficient method compared to the standard method. However, a significant drawback of adopting this strategy is the alteration of the data fidelity norm in the model, which is induced by the chosen preconditioner. As a result, the solution obtained using the left preconditioning approach may differ from that achieved with the standard variable metric strategy. To address this issue, we propose introducing non-stationarity in the preconditioner to recover the original norm in the data fidelity as the iterations progress.

Part I

The Graph Laplacian operator

Inverse Problems in imaging and Regularization

The first chapter of this work is primarily intended to provide preliminary knowledge about inverse problems and regularization. To this end, the first section reviews some standard concepts from numerical linear algebra, such as the Singular Value Decomposition (SVD) theorem and related results. This will help in understanding why inverse problems are challenging to solve and why naïve solutions are inadequate, necessitating the use of regularization techniques like variational models.

In this thesis, we will consider two types of inverse problems arising in imaging applications: image deblurring and X-ray computed tomography. Summarizing all the theory related to these two cases in just a few pages would be impossible. However, in Sections §1.2 and §1.3, we will highlight all the significant details that will be used in later chapters. In particular, we will describe how the forward problem is modeled and how to handle the inverse, ill-posed system. We will utilize the previously introduced SVD to derive useful information about the behavior of certain quantities, such as the noise that corrupts all available data. By the end of Section §1.3, the necessity for strategies to solve ill-posed problems will become evident. Consequently, Section §1.4 will introduce standard regularization techniques, transforming the previously ill-posed problems into well-posed ones (according to Hadamard's definition (0.0.1)).

The last section will be devoted to the central concept of this initial part: the graph Laplacian operator. We will describe in detail how to construct this operator by associating a graph with an image in the most natural way. Additionally, we will discuss the various definitions of the graph Laplacian and its different variants. By the end of Section §1.5, it will be clear how valuable this operator can be when incorporated into a variational problem designed to solve an ill-posed problem.

1.1 Preliminaries

Before devoting ourselves to the description of the image deblurring problem, we summarize some useful theoretical results that will be employed throughout this manuscript. For further

details and a more comprehensive analysis, the reader is referred to [74] and [133].

Definition 1.1.1. *Let $A \in \mathbb{C}^{m\times n}$ with $m \geq n$. We define the singular values of A as*

$$\sigma_i = \sqrt{\lambda_i}, \qquad i = 1, \dots, n, \tag{1.1}$$

where λ_i are the eigenvalues of $A^H A$.

Remark 1.1.2. *Because $A^H A$ is positive semidefinite, we can rearrange the singular values in nonincreasing order that is*

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n \geq 0.$$

In particular, we notice that

$$\|A\|_2 = \sqrt{\rho(A^H A)} = \sqrt{\lambda_1} = \sigma_1.$$

Theorem 1.1.3. *Let $A \in \mathbb{C}^{m\times n}$, then there exist the decomposition*

$$A = U\Sigma V^H, \tag{1.2}$$

where $U \in \mathbb{C}^{m\times m}$ and $V \in \mathbb{C}^{n\times n}$ are unitary matrices, while $\Sigma \in \mathbb{R}^{m\times n}$ is a rectangular diagonal matrix with diagonal elements the singular values of A arranged in nonascending order.

Remark 1.1.4. *If $A \in \mathbb{C}^{m\times n}$ has $\operatorname{rank}(A) = r$ with $r < \min(m, n)$, then the singular values of A are such that*

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > \sigma_{r+1} = \ldots = \sigma_n = 0.$$

Definition 1.1.5. *Let $A \in \mathbb{C}^{m\times n}$ with rank$(A) = r$ with $r < \min(m, n)$. We define the* compact SVD *of A as*

$$A = U_r \Sigma_r V_r^H = \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \mathbf{v}_i^H, \tag{1.3}$$

where $\mathbf{u}_i \in \mathbb{C}^m$, $i = 1, \dots, m$, and $\mathbf{v}_i \in \mathbb{C}^n$, $i = 1, \dots, n$, are the columns of U and V, respectively.

Remark 1.1.6. *With the same assumptions in the Definition 1.1.5 we have that*

- $\operatorname{Im}(A) = \operatorname{span}\{\mathbf{u}_1, \dots \mathbf{u}_r\}$,
- $\operatorname{Ker}(A) = \operatorname{span}\{\mathbf{v}_{r+1}, \dots, \mathbf{v}_n\}$.

In a similar way we have that

Definition 1.1.7. *Let $A \in \mathbb{C}^{m\times n}$ with rank$(A) = r \leq \min(m, n)$ and let $A = U\Sigma V^H$ its*

SVD. We define the truncated SVD (TSVD) *of A of order $s \leq r$ the matrix*

$$A_s = \sum_{i=1}^{s} \sigma_i \mathbf{u}_i \mathbf{v}_i^H.$$

Using the SVD, we can define the pseudo inverse $A^\dagger$ of $A \in \mathbb{C}^{m\times n}$. In particular, if $m = n$ and A has full rank, then $A^\dagger$ coincide with A^{-1}.

Definition 1.1.8. *Let $A = U\Sigma V^H \in \mathbb{C}^{m\times n}$, with rank$(A) = r \leq \min(n, m)$. Then we define the pseudo inverse of Moore-Penrose as*

$$A^\dagger = V\Sigma^\dagger U^H, \tag{1.4}$$

where

$$\Sigma^\dagger = \begin{bmatrix} \frac{1}{\sigma_1} & & & & \\ & \ddots & & & \\ & & \frac{1}{\sigma_r} & & \\ & & & 0 & \\ & & & & \ddots \end{bmatrix}_{n\times m}. \tag{1.4 bis}$$

Definition 1.1.9. *Let $A \in \mathbb{C}^{m\times n}$. The* condition number *of A is defined as*

$$\mu_2(A) = \|A\|_2 \|A^\dagger\|_2. \tag{1.5}$$

Remark 1.1.10. *If $A \in \mathbb{C}^{m\times n}$ has* rank$(A) = r \leq n$, *then*

$$\mu_2(A) = \frac{\sigma_1}{\sigma_r}. \tag{1.6}$$

1.2 Image deblurring

Blurred images are a common issue that can arise for a variety of reasons, the most common being when we take a picture with a camera and the lenses are out of focus. Besides the obvious applications in personal photography, this can be a significant problem in astronomical imaging and the medical field.

In the following pages, we will describe how images can be modeled using mathematical tools and how a blurred image relates to a sharp one. For the sake of completeness and clearness, the model problem will be derived in the 1D case (the case of signals). However, since we are mainly interested to image restoration problems (2D case), we will highlight the crucial differences between the two cases when necessary.

1.2.1 The model problem

A digital image is composed of pixels, each assigned an intensity value that characterizes the color of a small area of the scene. The more pixels we have, the better will be the resolution of the image. If we imagine to associate each pixel's intensity value with an element of a large matrix, it makes sense to represent a gray scale image with a matrix $X \in \mathbb{R}^{m\times n}$, where $N = mn$ is the total number of pixels. The values of X lie within the range $[0, 255]$, where zero corresponds to a black pixel and 255 to a white one. For notation, $X \in \mathbb{R}^{m\times n}$ represents the true, sharp image, while $B \in \mathbb{R}^{m\times n}$ denotes the recorded blurred image. Furthermore, we will use $\mathbf{x} = \text{vec}(X)$ to indicate the vector representation of the matrix X, that is,

$$X = \begin{bmatrix} \mathbf{x}_1 & | & \dots & | & \mathbf{x}_n \end{bmatrix} \iff \mathbf{x} = \text{vec}(X) = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_n \end{bmatrix} \in \mathbb{R}^N.$$

A first crucial assumption is that the blurring process is linear and can be formalized as the convolution between a particular kernel, known as the *Point Spread Function (PSF)*, and the true image. This assumption is justified as, in many situations, blur is indeed linear or can be well-approximated by a linear model. From a theoretical perspective, once we discretize the convolutional operator, the image deblurring problem involves solving the linear system

$$A\mathbf{x} = \mathbf{b},$$

where $A \in \mathbb{R}^{N\times N}$ is the so-called *blurring operator*, and $\mathbf{b}$ (resp. $\mathbf{x}$) is the vector representation of B (resp. X). However, in real-world applications, the blurred image B, and consequently its vector representation $\mathbf{b}$, is affected by some noise E. This may come from various sources, such as fluctuations during the recording process or approximation errors when representing the image with a limited number of digits. Therefore, the observed image can be expressed as

$$B = B^{\text{exact}} + E,$$

which implies

$$\mathbf{b} = \text{vec}(B) = \text{vec}(B^{\text{exact}}) + \text{vec}(E) = \mathbf{b}^{\text{exact}} + \mathbf{e},$$

where $\mathbf{b}^{\text{exact}} = A\mathbf{x}_{\text{gt}}$ for the ground truth image $\mathbf{x}_{\text{gt}} = \text{vec}(X_{\text{exact}})$. The presence of noise, combined with the ill-posed nature of the problem, makes the image reconstruction task difficult to solve. One might think that the naïve solution

$$\mathbf{x}_{\text{naïve}} = A^\dagger \mathbf{b},$$

where $A^\dagger$ is the pseudo-inverse of A, will yield the desired reconstruction. However, this ap-

proach fails because the naïve solution can be split into the two following components

$$\mathbf{x}_{\text{naïve}} = A^{\dagger}\mathbf{b} = A^{\dagger}\mathbf{b}^{\text{exact}} + A^{\dagger}\mathbf{e}, \tag{1.7}$$

and the final reconstruction will be dominated by the influence of the inverted noise $A^{\dagger}\mathbf{e}$. Indeed, from the definition (1.1.5) of the compact SVD, we have that

$$A^{\dagger} = \sum_{i=1}^{r} \frac{1}{\sigma_i} \mathbf{v}_i \mathbf{u}_i^H,$$

where $r = \text{rank}(A)$. Using this expression to compute the inverse of the noise, we obtain that

$$A^{\dagger}\mathbf{e} = V\Sigma^{\dagger}U^H\mathbf{e} = \sum_{i=1}^{r} \frac{\mathbf{u}_i^H\mathbf{e}}{\sigma_i}\mathbf{v}_i. \tag{1.8}$$

To understand why the inverted noise (1.8) dominates the naïve reconstruction (1.7), it is important to note that the following properties generally hold for image deblurring problems:

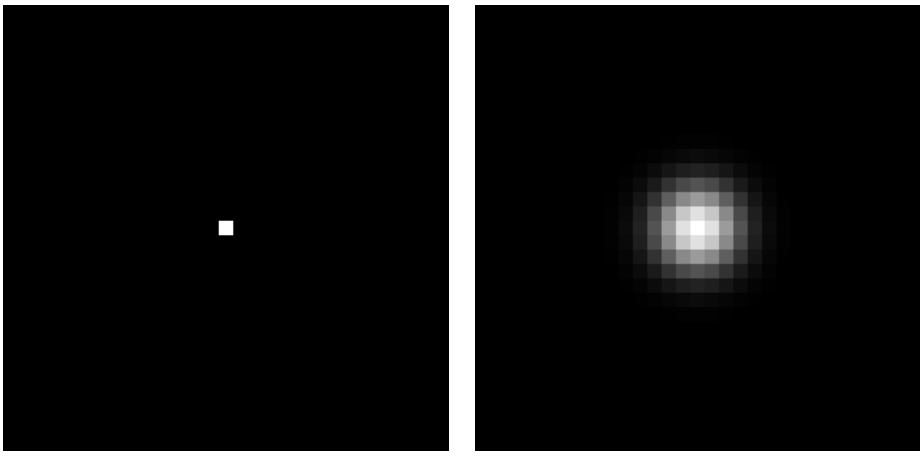
- The values $|\mathbf{u}_i^H\mathbf{e}|$ are small and nearly uniform across all i;
- A is severely ill-conditioned, meaning that the singular values σ_i decay to zero without any significant gap. Consequently, the condition number of A

  $$\mu_2(A) = \frac{\sigma_1}{\sigma_r}$$

  is very large and σ_i approaches the machine precision when i approaches r.
- Singular vectors corresponding to the smallest singular values represent highest frequency information.

This implies that the coefficients $\frac{\mathbf{u}_i^H\mathbf{e}}{\sigma_i}$ increase as i increases, which in turn severely degrades the quality of the final reconstruction. This analysis suggests one potential solution: the use of the TSVD to eliminate the high-frequency components of the inverted noise. However, given that the singular values of the blurring operator A decay to zero without any significant gap, it is crucial to determine the optimal number of singular values to retain. A more widely used approach involves introducing regularization techniques that aim to reduce the problem's sensitivity to perturbations in the collected data. The main idea, as will be described in Section §1.4, is to replace the original ill-posed problem with a nearby well-posed one.

This introductive part shows that the image deblurring problem is not as straightforward as one might expect. Various factors can influence the difficulty of recovering a sharp image from a blurred and noisy observation. In later chapters, we will present some variational approaches that provide good approximations by exploiting different strategies in the regularization technique. However, to better understand what comes next, it is essential to have a clear idea of all the relevant elements that play a crucial role in the image deblurring



(a) Point source (b) PSF

Figure 1.1: *An example of point source and point spread function.*

process.

1.2.2 The blurring operator A

This section is devoted to this purpose. We will define different types of Point Spread Functions (PSFs) and demonstrate how the blur operator is derived from the discretization of the convolutional process. We will also discuss the structure that the blur operator might inherit from the boundary conditions imposed on our problem and how it is related to the PSF. A more complete and comprehensive analysis about image deblurring problem can be found in [82]. In the previous section, we introduced the idea that the blurring process is linear, essentially consisting of a convolution between a specific kernel and the true image. In the discrete case, this can be formalized as a linear system of equations, where the blurring process is captured by the blurring operator A. This establishes a direct relationship between A and the PSF used in the process.

Point Spread Function (PSF)

The blurring process essentially involves spreading the brightness of each pixel across its neighboring ones and the PSF describes how this occurs. Suppose a blurring phenomenon takes place, and we can replicate the exact process on random images. Now, consider a completely black image with just one bright pixel. The resulting blurred image will show the brightness of that pixel spread over its neighboring pixels, as illustrated in Figure 1.1. The image with just one non-black pixel is referred to as the *point source*, while the resulting blurred image is the PSF.

Mathematically, the point source corresponds to the canonical basis vector $\mathbf{e}_i$, which is an array of zeros except for the ith component, which has a value of 1. This also implies that

the blurring process

$$A\mathbf{e}_i = \left[\begin{array}{c|c|c|c} \mathbf{a}_1 & \mathbf{a}_2 & \dots & \mathbf{a}_n \end{array}\right] \mathbf{e}_i = \mathbf{a}_i$$

returns the ith column of the blurring operator. If we repeat this for all i, we obtain the entire blurring matrix A.

In what follows, we assume that the PSF is *spatially invariant*, meaning that all pixels are blurred in the same way. Moreover, we observe that the PSF is confined to a small area around its *center*, which is the pixel location of the point source. This implies that blurring is typically a local phenomenon, allowing us to conserve storage by representing the PSF with a matrix K of smaller dimensions than the original image. In some cases, the PSF can be described analytically and constructed from a function rather than through experimentation. For instance, the elements $K_{i,j}$ of the PSF array for out-of-focus blur are given by

$$K_{i,j} = \begin{cases} \frac{1}{\pi r^2} & \text{if } (i-k)^2 + (j-l)^2 \leq r^2, \\ 0 & \text{otherwise,} \end{cases}$$

where (k, l) is the center of K, and r is the radius of the blur. In the case of atmospheric turbulence, the PSF can be described as a two-dimensional Gaussian function whose elements are given by

$$K_{i,j} = \exp\left(-\frac{1}{2}\begin{bmatrix} i-k \\ j-l \end{bmatrix}^T \begin{bmatrix} s_1^2 & \rho^2 \\ \rho^2 & s_2^2 \end{bmatrix}^{-1} \begin{bmatrix} i-k \\ j-l \end{bmatrix}\right),$$

where s_1, s_2, and ρ determine the width and orientation of the PSF, which is again centered at the element (k, l). Further details and examples aobut PSF can be found in [82].

The blurring process

For the sake of notational simplicity, we just consider the $1-$dimensional problem. The assumptions of locality and spatial invariance of the PSF, impose a particular structure on the blurring operator A. Indeed, these imply that the convolutional kernel $K(x, s)$ is invariant by translation and, by recalling that a blurred image can be obtained by convolving a PSF with a true image, we can formalize the blurring process as

$$b(s) = \int_a^b K(s-t)x(t)\mathrm{d}t, \tag{1.9}$$

where $s \in [a, b]$. Discretizing (1.9) on the N grid points $t_i = a + ih$ with $h = \frac{b-a}{N}$, we obtain

$$b(t_i) = \int_a^b K(t_i - t)x(t)\mathrm{d}t. \tag{1.10}$$

Using the rectangle rule on the same grid points, we have

$$b(t_i) \approx h \sum_{j=0}^{N-1} K(t_i - t_j)x(t_j). \tag{1.11}$$

Because $t_i - t_j = (i-j)h$, let

$$K_{i-j} = hK((i-j)h)$$

and substituting in (1.11), we obtain

$$b(t_i) \approx \sum_{j=0}^{N-1} K_{i-j} x(t_j), \quad i = 0, \dots N-1.$$

This can be rewritten as the linear system

$$\underbrace{\begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_{N-1} \end{bmatrix}}_{\mathbf{b}} = \underbrace{\begin{bmatrix} K_0 & K_{-1} & \dots & K_{-N+1} \\ K_1 & K_0 & \dots & K_{-N+2} \\ \vdots & & \ddots & \vdots \\ K_{N-1} & K_{N-2} & \dots & K_0 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{N-1} \end{bmatrix}}_{\mathbf{x}}, \tag{1.12}$$

where $x_j = x(t_j)$ and $b_j = b(t_j)$ for $j = 0, \dots, N-1$. The coefficients $\{K_j\}_{j=-N+1,\dots,N-1}$ are the elements of the PSF. As we have seen before, in Figure 1.1, the PSF is confined in a small area and so has a much smaller dimension than the real image. This means that we can assume that only the coefficients K_j for $j = -l, \dots, l$ are non zero for some $0 < l < \frac{N}{2}$. Moreover, we already noted that when we consider a source point near the edge then some pixels of the PSF may be outside the boundary. For this two reasons, defining as w_j and y_j the pixels in the original scene that are actually outside the field of view, the complete linear system is

$$\begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_{N-1} \end{bmatrix} = \begin{bmatrix} K_l & \dots & K_0 & \dots & K_{-l} & 0 & \dots & 0 \\ 0 & K_l & \dots & K_0 & \dots & K_{-l} & \ddots & \vdots \\ \vdots & \ddots & \ddots & & \ddots & & \ddots & 0 \\ 0 & \dots & 0 & K_l & \dots & K_0 & \dots & K_{-l} \end{bmatrix} \begin{bmatrix} y_0 \\ \vdots \\ y_{l-1} \\ x_0 \\ \vdots \\ x_{N-1} \\ w_0 \\ \dots \\ w_{l-1} \end{bmatrix}. \tag{1.13}$$

Boundary conditions

With straightforward computations we have obtained an explicit representation of the blurring problem. Unfortunately, the linear system (1.13) is underdetermined. Moreover, since images represent just a limited portion of a scene that extends forever in all directions, we could have problems to recover some details of the real image near the boundaries. Consider for instance a point source that has the bright pixel close to the boundary. Then, in the corresponding PSF some pixels may be outside the field of view that is we are loosing some

informations.

To overcome this problem, we have to assume certain conditions about the behavior of the sharp image outside the boundary. There are many different types of boundary conditions that can be assumed. In the following discussion, we will focus primarily on zero Dirichlet and periodic conditions. However, it is worth mentioning the existence of reflective and anti-reflective boundary conditions [60, 115, 131], or other strategies for dealing with the boundary effects [67, 17], which often achieve better reconstructions near the boundary . With the *zero boundary conditions*, we assume that the sharp image is black outside the boundary. This can be interpreted as embedding the real image X in a larger one that is

$$\tilde{X} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & X & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

Zero boundary conditions are particularly effective when considering astronomical images, where there are only a few bright pixels representing stars or planets while the rest of the image is black. Under this assumption, we can rewrite the blurring operator A in (1.13) as

$$A = \begin{bmatrix} K_0 & K_{-1} & \dots & K_{-l} & 0 & \dots & 0 \\ K_1 & K_0 & K_{-1} & \dots & K_{-l} & \ddots & \vdots \\ \vdots & & \ddots & & & \ddots & 0 \\ K_l & & & \ddots & & & K_{-l} \\ 0 & \ddots & & & \ddots & & \vdots \\ \vdots & \ddots & \ddots & & & \ddots & K_{-1} \\ 0 & \dots & 0 & K_l & \dots & K_1 & K_0 \end{bmatrix}. \tag{1.14}$$

Note that, with zero Dirichlet boundary conditions, the matrix A assumes a Toeplitz structure, where the coefficients are derived from the elements of the PSF. Thus, boundary conditions impose structures on the blurring operator.

Another possible choice is to assume that the image X repeats itself in all directions. These are called *periodic boundary conditions*, which can be interpreted as the sharp image X being repeated outside the field of view, that is

$$\tilde{X} = \begin{bmatrix} X & X & X \\ X & X & X \\ X & X & X \end{bmatrix}.$$

This assumption implies a specific relationship between the pixels inside and outside the field of view, namely

$$y_j = x_{N+j-l} \quad \text{and} \quad w_j = x_j, \quad \forall j \in \{0, \dots, l-1\}.$$

By incorporating this relation into the underdetermined linear system (1.13), we can rewrite the linear operator A as

$$A = \begin{bmatrix} K_0 & K_{-1} & \dots & K_{-l} & 0 & \dots & 0 & K_l & \dots & K_1 \\ K_1 & K_0 & K_{-1} & \dots & K_{-l} & \ddots & & \ddots & \ddots & \vdots \\ \vdots & & \ddots & & & \ddots & \ddots & & \ddots & K_l \\ K_l & & & \ddots & & & \ddots & \ddots & & 0 \\ 0 & \ddots & & & \ddots & & & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & & & \ddots & & & \ddots & 0 \\ 0 & & \ddots & \ddots & & & \ddots & & & K_{-l} \\ K_{-l} & \ddots & & \ddots & \ddots & & & \ddots & & \vdots \\ \vdots & \ddots & \ddots & & \ddots & \ddots & & & \ddots & K_{-1} \\ K_{-1} & \dots & K_{-l} & 0 & \dots & 0 & K_l & \dots & K_1 & K_0 \end{bmatrix}. \tag{1.15}$$

Under periodic boundary conditions, the blurring operator A acquires a circulant structure. This is particularly advantageous from a computational perspective, as circulant matrices can be diagonalized using the Fourier transform, allowing matrix–vector products involving A to be computed efficiently.

So far, all computations and analyses have been restricted to the 1D case. In two dimensions these ideas can be extended similarly, with a slight modification in the structure of A. Specifically, with zero Dirichlet boundary conditions, the blur operator A becomes a block Toeplitz with Toeplitz blocks (BTTB), while with periodic boundary conditions, it becomes a block circulant with circulant blocks (BCCB).

Remark 1.2.1. *BCCB matrices can be diagonalized using the 2-dimensional Fourier transform. This also allows us to define the 2-dimensional Fast Fourier Transform (FFT2).*

To conclude this brief introduction to the image deblurring problem, we show that, given a PSF and assuming periodic boundary conditions, we can compute the spectral decomposition of the blur operator A straightforwardly. For notational simplicity, let $\mathbb{F}_2$ denote the 2-dimensional Fourier matrix. Since A is BCCB, we can write

$$A = \mathbb{F}_2^H \Lambda \mathbb{F}_2, \tag{1.16}$$

where Λ is a diagonal matrix containing the eigenvalues of A. To compute all the diagonal elements of Λ, we note that

$$A = \mathbb{F}_2^H \Lambda \mathbb{F}_2 \implies \mathbb{F}_2 A = \Lambda \mathbb{F}_2 \implies \mathbb{F}_2 \mathbf{a}_1 = \Lambda \mathbf{f}_1 = \frac{1}{\sqrt{N}} \boldsymbol{\lambda},$$

where $\boldsymbol{\lambda} = (\lambda_1, \dots, \lambda_N)$ is the vector of eigenvalues and $\mathbf{a}_1$ and $\mathbf{f}_1$ are the first columns of A and $\mathbb{F}_2$, respectively. Thereby, to obtain the spectral decomposition, we only need to

compute $\sqrt{N}\mathbb{F}_2\mathbf{a}_1$. From (1.15), we observe that the first column of A is simply a circular shift of the coefficient array of the PSF. In the 1D case, given the PSF as

$$\text{PSF} = [0, \ldots, 0, K_{-l}, \ldots, K_0, \ldots, K_l, 0, \ldots, 0]\,,$$

the circular shift of the components with respect to the center of the PSF K_0, is given by

$$\text{Circshift}(\text{PSF}, K_0) = [K_0, \ldots, K_l, 0, \ldots, 0, K_{-l}, \ldots, K_{-1}]\,.$$

In the 2D case, we shift the components in both directions. If the PSF with center $[2, 2]$ is defined as

$$\text{PSF} = \begin{bmatrix} K_{11} & K_{12} & K_{13} \\ K_{21} & K_{00} & K_{23} \\ K_{31} & K_{32} & K_{33} \end{bmatrix},$$

then its circular shift becomes

$$\text{Circshift}(\text{PSF}, K_{00}) = \begin{bmatrix} K_{00} & K_{23} & K_{21} \\ K_{32} & K_{33} & K_{31} \\ K_{12} & K_{13} & K_{11} \end{bmatrix}.$$

1.3 Computed Tomography (CT)

In medical imaging, Computed Tomography (CT) is a diagnostic procedure that combines X-ray and computer technology to analyze the inside of the body. CT imaging provides detailed informations of various body parts, including bones, muscles, and organs. The fundamental concept involves reconstructing the interior of the patient using multiple one-dimensional slices of the object of interest. These slices are obtained by passing several parallel X-ray beams through the object. The acquired data are based on the variations in X-ray intensity. Specifically, let I_0 be the initial intensity of each X-ray beam. As the beam passes through the object, its intensity may be reduced due to the presence of different tissues. The intensity that we measure at the end, I_1, represents the attenuation of the beam. To gain a comprehensive understanding of the object's geometry, we can repeat these measurements from different angles. Collecting all the intensity variations for each X–ray beam at each angle yields what is known as the *sinogram* of the object. In Figure 1.2 we depicted this process. The dark gray rectangle represent the radiation source that emits a beam of X–ray (dashed lines) that go throught the object placed in the center of the circumference. On the opposite part of the radiation source we have the radiation detector. It basically measures the difference of intensity for each X—ray beam, caused by X–ray attenuation through tissues, yielding to the blue curve. This will correspond to one column of the right hand side image. In order to obtain the full sinogram we need to rotate the system source–detector around the central object. Before describing the mathematical model for X–ray tomography, we need some simplifying assumptions. The real task of CT problem is to reconstruct an image that represents what is called the *attenuation coefficients*

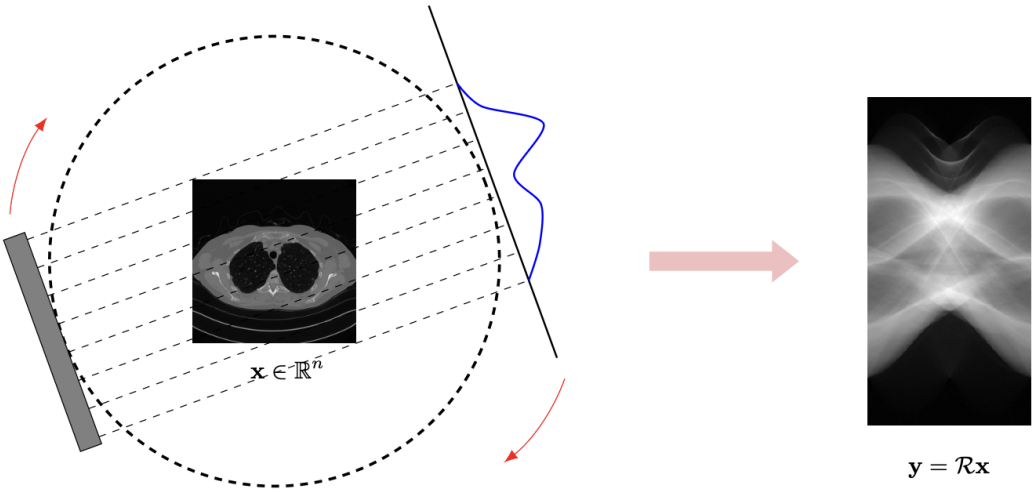


Figure 1.2: A schematic rapresentation of the CT acquisition process. The operator $\mathcal{R}$ represents the Radon transform, which will be mathematically defined in Section 1.3.2.

of the interior of a patient. Since this is a quantity that is strictly related to the density of the item we are considering, detailed information about the internal density of the object are needed. However, these are nearly impossible to recover by merely inspecting the sinogram. For this reason, we will assume that

- All X–ray beams are *monochromatic*. That is photons propagates with the same energy and at the same frequency;
- X–ray beams have zero width;
- X–ray beams are not subject of *refraction* and *diffraction*. That is they do not scatter or bend when in contact with surfaces.

Even if these assumptions are not entirely accurate, for our purpose they are sufficiently close to reality.

1.3.1 Beer's law and X-ray tomography

Now that the idea of the physical model for CT problem is clear, we are going to take a closer look to the mathematical aspect. Let $\mathbf{x}_{\text{gt}} \in \mathbb{R}^n$ be the attenuation coefficient, we indicate the proportion of photons absorbed at a distance $\mathbf{s} \in \mathbb{R}^2$ from the origin as $\mathbf{x}_{\text{gt}}(\mathbf{s})$.

In CT problem, we know the initial and final intensities, I_0 and I_1, of a single beam. What we want is to being able to use these information to determine the attenuation coefficient over the path of the beam. The physics governing the absportion of X–rays by tissues is described by the Beer–Lambert law

$$\frac{\mathrm{d}I(\mathbf{s})}{\mathrm{d}\mathbf{s}} = -\mathbf{x}_{\text{gt}}(\mathbf{s})I(\mathbf{s})\mathrm{d}\mathbf{s}. \tag{1.17}$$

Let $I(\mathbf{s}_0) = I_0$ be the initial intensity at $\mathbf{s}_0$, we can solve equation (1.17) for a generic point $\mathbf{s} \in \mathbb{R}^2$ to obtain

$$I(\mathbf{s}) = I_0 \exp\left(- \int_{L[\mathbf{s}_0, \mathbf{s}]} \mathbf{x}_{\text{gt}}(\mathbf{z}) \mathrm{d}\mathbf{z} \right),$$

where $L[\mathbf{s}_0, \mathbf{s}]$ represent the trajectory of the X–ray from $\mathbf{s}_0$ to $\mathbf{s}$. Moreover, let $I(\mathbf{s}_1) = I_1$ be the final intensity at $\mathbf{s}_1$ we obtain

$$\ln\left(\frac{I_0}{I_1}\right) = \exp\left(- \int_{L[\mathbf{s}_0, \mathbf{s}_1]} \mathbf{x}_{\text{gt}}(\mathbf{z}) \mathrm{d}\mathbf{z} \right). \tag{1.18}$$

Since I_1 is measured in the detector and I_0 is known by design, we aim at reconstructing the attenuation coefficient $\mathbf{x}_{\text{gt}}$ from the intensity measurements $g(t) = \ln\left(\frac{I_1}{I_0}\right)$, where $t \in \mathbb{R}$ represents the measured ray's position. As anticipated, to tackle this problem efficiently, different data are collected from various angles by rotating the source around the object.

Before we dive deeper into solving equation (1.17), let us first introduce the coordinate system we will use. The Cartesian coordinate system struggles with handling vertical lines that have an infinite slope, while the polar coordinate system is not well-suited for systems that rely on parallel lines. In the following, we introduce what is called a *point normal* parametrization for a line. Let $a, b, c \in \mathbb{R}$ with $a, b \neq 0$, we represent a line l in $\mathbb{R}^2$ with the standard equation

$$\frac{a}{\sqrt{a^2+b^2}} + \frac{b}{\sqrt{a^2+b^2}} = \frac{c}{\sqrt{a^2+b^2}}.$$

Since $\omega = \left(\frac{a}{\sqrt{a^2+b^2}}, \frac{b}{\sqrt{a^2+b^2}}\right)$ lay on the unit circle S^1, there exist $\theta \in [0, 2\pi]$ such that $\omega = (\cos\theta, \sin\theta)$. Let $t = \frac{c}{\sqrt{a^2+b^2}}$ we can parametrize a line l in $\mathbb{R}^2$ by means of t and θ namely

$$l_{t,\theta} = \{\mathbf{z} \in \mathbb{R}^2 : \langle \mathbf{z}, (\cos\theta, \sin\theta) \rangle = t\}. \tag{1.19}$$

Figure 1.3 (a) offers a visual illustration of how we parametrized the set of all lines. The idea is the following: given the distance t of the line l from the origin and the angle θ that t have to form with the horizontal axis, the points z on the line are all and only those whose projections along the direction of $\omega = (\cos\theta, \sin\theta)$ have magnitude t. Figure 1.3 (b) illustrates an alternative way to define a parametrization for a line l in $\mathbb{R}^2$. Let $\theta \in [0, 2\pi]$ and, as before, consider a point $\omega = (\cos\theta, \sin\theta)$ on the unit circle S^1. The idea behind this parametrization is based on the observation that the vector $\omega^\perp = (-\sin\theta, \cos\theta)$ is orthogonal to ω. Therefore, given a distance $t \in \mathbb{R}$, we can describe all the points of the line $l_{t,\theta}$ as a linear combination of the vector $t\omega$ and $\omega^\perp$, formally

$$l_{t,\theta} = \{(t\cos\theta - c\sin\theta, t\sin\theta + c\cos\theta) : c \in \mathbb{R}\}. \tag{1.20}$$

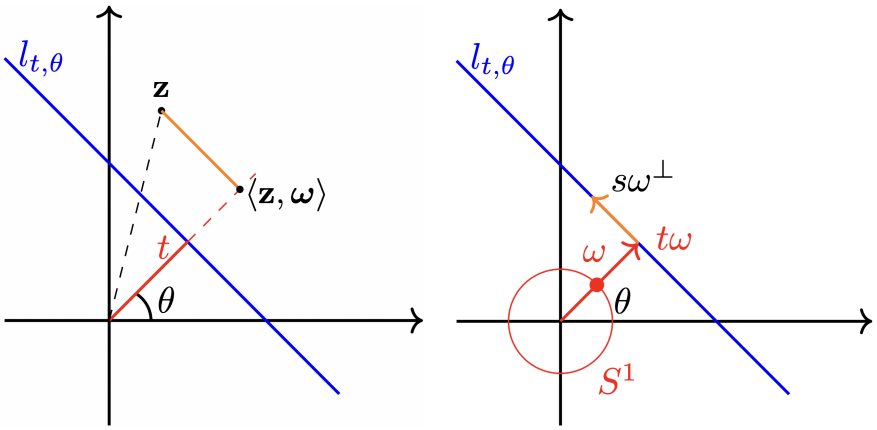


Figure 1.3: Illustration of two different parametrizations for a line $l \in \mathbb{R}^2$.

1.3.2 The Radon transform and Central slice theorem

We are now ready to introduce the core results in X–ray tomography that will be used to recover the attenuation coefficient $\mathbf{x}_{\text{gt}}$ in equation (1.18).

Definition 1.3.1 (Radon Transform)**.** *Let $t \in \mathbb{R}$, $\theta \in [0, 2\pi]$, and f be a continuos function defined on $\mathbb{R}^2$. Then, the Radon transform $(\mathcal{R}f)(t, \theta)$ of f is defined as*

$$(\mathcal{R}f)(t,\theta) = \int_{-\infty}^{\infty} f(t\cos\theta - c\sin\theta, t\sin\theta + c\cos\theta)\mathrm{d}c. \tag{1.21}$$

For a medical imaging problem, it is reasonable to further assume that the function f has compact support as we are only concerned with finite areas (or slices) of an object. For example, in X-ray tomography, we deal with finite slices of an object, which implies that the attenuation coefficient must be equal zero outside some finite region. The Radon transform allow us to determine the total density of a certain function f along a given parametrized line $l_{t,\theta}$. The integral $\mathcal{R}f(t, \theta)$ represent the right hand side of the Beer–Lambert equation (1.18). In other words, the measured data $\ln\left(\frac{I_0}{I_1}\right)$ corresponds to the Radon transform of the attenuation coefficient $\mathbf{x}_{\text{gt}}$ along the line $L[\mathbf{s}_0, \mathbf{s}_1]$. Therefore, for our purposes, it is essential to find an inversion formula.

To this end, we need two additional key results. Before presenting them, we briefly recall the definitions of the Fourier transform and the Fourier inversion theorem.

Definition 1.3.2 (1–D Fourier transform)**.** *Let $f \in L^1(\mathbb{R})$ be an absolutely integrable function on $\mathbb{R}$. For all $\xi \in \mathbb{R}$, we define the 1–D Fourier transform $\mathcal{F}_1 f$ of f as*

$$(\mathcal{F}_1 f)(\xi) = \int_{-\infty}^{\infty} f(x)\mathrm{e}^{-2\pi i\xi x}\mathrm{d}x.$$

Definition 1.3.3 (Schwartz space)**.** *Let $f\colon \mathbb{R}^n \to \mathbb{R}$ and let α, β be two multiindexes. Let*

$$\|f\|_{\alpha,\beta} = \sup_{\mathbf{x}\in\mathbb{R}^n} |\mathbf{x}^\alpha D^\beta f(\mathbf{x})|, \qquad D^\beta = \frac{\partial^{|\beta|}}{\partial x_1^{\beta_1}\dots\partial x_n^{\beta_n}}.$$

The Schwartz space $\mathcal{S}$ onto $\mathbb{R}^n$ is defined as the functional space

$$\mathcal{S}(\mathbb{R}^n) \coloneqq \{f \in C^\infty(\mathbb{R}^n) : \|f\|_{\alpha,\beta} < \infty,\ \forall\alpha,\beta\}.$$

Definition 1.3.4. *Let $f \in L^1(\mathbb{R})$, we define the inverse Fourier transform $\mathcal{F}_1^{-1}$ of f as*

$$(\mathcal{F}_1^{-1}f)(x) = \frac{1}{2\pi}\int_{-\infty}^{\infty} f(\xi)\mathrm{e}^{2\pi i\xi x}\mathrm{d}\xi.$$

Theorem 1.3.5 (Fourier inversion theorem)**.** *Let $f \in \mathcal{S}(\mathbb{R})$, then*

$$\mathcal{F}_1^{-1}(\mathcal{F}_1 f)(x) = f(x). \tag{1.22}$$

All this results and definition can be extended to a general dimension n although for our purposes we will only utilize the two–dimensional analogs.

Definition 1.3.6 (2–D Fourier transform)**.** *Let $f \in L^1(\mathbb{R}^2)$. For all $\boldsymbol{\xi} = (\xi_1, \xi_2) \in \mathbb{R}^2$, we define the 2–D Fourier transform $\mathcal{F}_2 f$ of f as*

$$(\mathcal{F}_2 f)(\boldsymbol{\xi}) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\mathrm{e}^{-2\pi i(x\xi_1+y\xi_2)}\mathrm{d}x\mathrm{d}y. \tag{1.23}$$

Definition 1.3.7. *Let $f \in L^1(\mathbb{R}^2)$, we define the 2–D inverse Fourier transform $\mathcal{F}_2^{-1}$ of f as*

$$(\mathcal{F}_2^{-1}f)(\mathbf{x}) = \frac{1}{4\pi^2}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\xi_1,\xi_2)\mathrm{e}^{2\pi i(x\xi_1+y\xi_2)}\mathrm{d}\xi_1\mathrm{d}\xi_2.$$

We can now state and prove the central slice theorem that will give us a remarkable relationship between the two dimensional Fourier transform and the one dimensional Fourier transform of the Radon transform.

Theorem 1.3.8. *Let $f \in L^1(\mathbb{R}^2)$. Then, for all $\rho \in \mathbb{R}$ and $\theta \in [0, 2\pi]$ we have that*

$$(\mathcal{F}_2 f)(\rho\cos\theta, \rho\sin\theta) = (\mathcal{F}_1\mathcal{R}f)(\rho). \tag{1.24}$$

Proof. From the definition of the 2–D Fourier transform, we have

$$(\mathcal{F}_2 f)(\rho\cos\theta, \rho\sin\theta) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\mathrm{e}^{-2\pi i\rho(x\cos\theta+y\sin\theta)}\mathrm{d}x\mathrm{d}y. \tag{1.25}$$

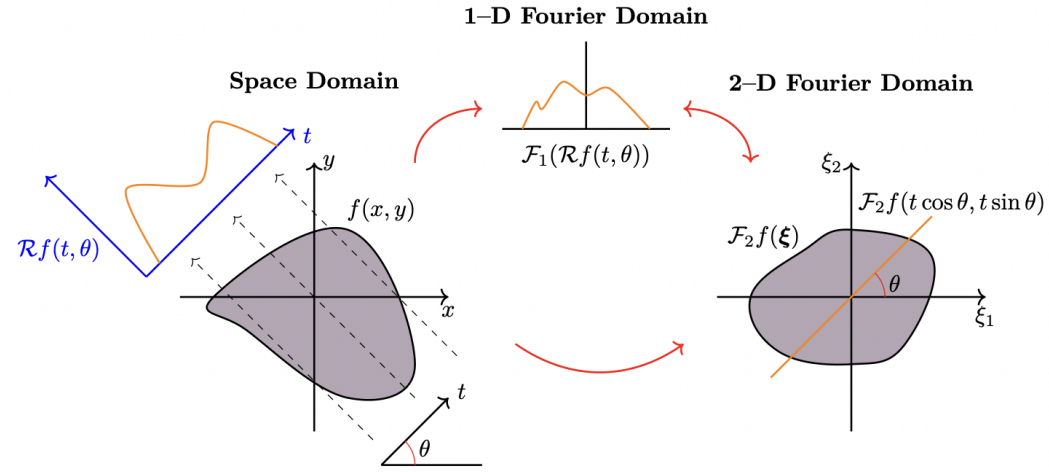


Figure 1.4: A schematic rapresentation of the Fourier central slice theorem.

Recalling the line parametrization (1.20), we implement the change of variables

$$x(t,c) = t\cos\theta - c\sin\theta \qquad y(t,c) = t\sin\theta + c\cos\theta,$$

obtaining that the determinant of the Jacobian of the transformation is given by

$$\det \begin{bmatrix} \frac{\partial x}{\partial t} & \frac{\partial x}{\partial c} \\ \frac{\partial y}{\partial t} & \frac{\partial y}{\partial c} \end{bmatrix} = 1.$$

Moreover, from the first interpretration (1.19), we know that $t = x\cos\theta + y\sin\theta$. Combining all together, we can rewrite the right hand side of Equation (1.25) as

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(t\cos\theta - c\sin\theta, t\sin\theta + c\cos\theta)\mathrm{e}^{-2\pi i t\rho}\mathrm{d}t\mathrm{d}c.$$

Since $\mathrm{e}^{2\pi i t\rho}$ has no dependence on c we can rearrange the integral as

$$\int_{-\infty}^{\infty}\left(\int_{-\infty}^{\infty} f(t\cos\theta - c\sin\theta, t\sin\theta + c\cos\theta)\mathrm{d}c\right)\mathrm{e}^{-2\pi i t\rho}\mathrm{d}t,$$

and the thesis follow by noting that the inner integral is the Radon transform $\mathcal{R}f(t,\theta)$ of f, while the outer integral coincide with the 1–D Fourier transform. □

The previous theorem states that the one-dimensional Fourier transform of a projected function (the Radon transform) is equal to the two-dimensional Fourier transform of the original function, evaluated along the slice through the origin that is parallel to the line on which the function was projected. This concept is depicted in Figure 1.4. On the left side, we report the space domain where the original function f is defined, along with its Radon transform. In the central part, we show the 1-D Fourier transform of $\mathcal{R}f(t,\theta)$, and finally, on the right side, the 2-D Fourier domain completes the picture.

1.3.3 Filtered Back projection

In this last part, we finally provide a strategy to reconstruct the attenuation coefficient of an object. Recall that, from a physical perspective, the Radon transform $\mathcal{R}f(t,\theta)$ gives us the total density of the object f along a line $l(t,\theta)$. This is determined by measuring the initial and final intensities of an X-ray beam as it passes through the object along that line (Beer-Lambert law). By repeating this process for multiple different lines, we can create a single slice of the object. Moreover, by varying the angle θ of the X-rays, we can generate the complete sinogram of the object. If we were then able to somehow backprojecting these densities onto the plane, we can recreate our original object.

The most straightforward approach is to average the values of $\mathcal{R}f(t,\theta)$ over all the lines passing through a specific point. For a given angle $\theta \in [0, 2\pi]$, and defining the direction $\omega = (\cos\theta, \sin\theta)$, the line in the family $\{l_{t,\theta} : t \in \mathbb{R}\}$ that passes through a point (x, y) is given by $t = \langle (x, y), \omega \rangle$. Thus, the *back-projection formula* is defined as

$$\mathcal{B}\mathcal{R}(x,y) = \frac{1}{\pi}\int_0^{\pi} \mathcal{R}f(x\cos\theta + y\sin\theta, \theta)\mathrm{d}\theta. \tag{1.26}$$

Backprojecting in only a few directions θ is an extremely inaccurate way of reconstructing even a simple object. However, even if we significantly increase the number of backprojections, the recreated image still suffers from a considerable amount of noise. In fact, regardless of how many directions we use for backprojection, we will not be able to perfectly reconstruct our image using the backprojection formula stated in equation (1.26). To make this process more effective, we need to find a way to filter out some of the noise that the backprojection formula introduces, thereby achieving a clearer and more accurate representation of our object. To accomplish this, we define the *filtered backprojection*.

Theorem 1.3.9. *Let $f \in L^2(\mathbb{R}^2)$ be an absolute integrable function defined on $\mathbb{R}^2$. Then,*

$$f(x,y) = \frac{1}{2}\mathcal{B}\left(\mathcal{F}_1^{-1}[|S|\mathcal{F}_1(\mathcal{R}f(S,\theta)]\right)(x,y). \tag{1.27}$$

The important factor in equation (1.29) is given by the $|S|$ multiplier that occurs between the Fourier transform and its inverse. We call this additional term a filter to the Radon transform, giving us the name for the filtered backprojection formula. However, is not easy to deal with equation (1.27). Fully expanded it would include several infinite integrals and it would be nice if we could somehow further simplify this to a more useful form. To this aim, suppose that there exist some function $\varphi(t)$ whose Fourier transform is equal to $|S|$, i.e. $\mathcal{F}_1\varphi(S) = |S|$. Then, we can rewrite the filtered backprojection formula (1.27) as

$$f(x,y) = \frac{1}{2}\mathcal{B}\left(\mathcal{F}_1^{-1}[\mathcal{F}_1\varphi\mathcal{F}_1(\mathcal{R}f(S,\theta))]\right)(x,y).$$

Let $\star$ indicates the 2D convolution operator and let $f, g \in L^1(\mathbb{R})$ then, from the convolution

theorem, we know that

$$(\mathcal{F}_1 f)(\xi) \star (\mathcal{F}_1 g)(\xi) = \mathcal{F}_1(f \star g)(\xi). \tag{1.28}$$

Applying equation (1.28) to our filtered back projection formula lead us to obtain the final formulation

$$f(x, y) = \frac{1}{2}\mathcal{B}\left(\varphi \star \mathcal{R}\right)(x, y). \tag{1.29}$$

Many others aspects could be considered and analyzed. For instance, we can only compute an approximation of the filtered backprojection formula (1.29) because there is no function φ whose Fourier transform is exactly equal to the absolute–value function. To partially overcome this problem one can introduce the concept of *band limited function*. This would allows us to explore some classical filters used in X–ray tomography like the *Ram–Lak* and the *Hann* filters. More detailed information on these topics can be found in [68]. For our purposes, we conclude this introductory section on X–ray tomography with a final observation.

In the previous section, we discussed how the image deblurring problem was modeled by the convolution between a kernel (PSF) and the image $\mathbf{x}_{\text{gt}}$ that we aim to recover. Similarly, we noted that the CT problem is defined by the Radon transform of the attenuation coefficient that we want to reconstruct. In both cases, the convolution operator and the Radon transform are integral operators and since we cannot work directly in a continuous framework, we must discretize these operators. This clearly leads to ill-posed problems, which can be formalized in both cases by the linear system of equations

$$A\mathbf{x} = \mathbf{b},$$

where A could represent the blur operator, in the image deblurring problem, or the discretization of the Radon transform, in the CT problem. Similarly, the observed data $\mathbf{b}$ could represent the blurred image corrupted by noise or the intensity measurement of the X-ray beam affected by measurement errors. Lastly, the solution $\mathbf{x}$ that we seek to recover could be the true image or the attenuation coefficient of an object. However, even though both problems can be described by a linear system, the dimensions of the involved operator depend on different factors. For image deblurring, the dimensions are influenced by the number of pixels in the ground truth image. In the CT problem, the dimensions depend on the number of rays n_d considered for each angular scan and the number n_θ of sampled angles. Modern medical protocols tend to minimize the number of projection angles due to their relationship with patient radiation exposure, which greatly affects A by making it non-injective in practice. This brings us back to the problem already observed in the case of image deblurring, which is the need for specific techniques to solve these types of inverse problems.

1.4 Regularization

From previous sections we understand that ill–posed problems may arise in many different circumstances and naïve strategies can not be applied to recover the original image $\mathbf{x}_{\text{gt}} \in \mathbb{R}^n$ due to the presence of some perturbation in the collected data. Image deblurring, CT, and many other inverse problems we could aim at solving can be formalized with the linear system of equations

$$A\mathbf{x} = \mathbf{b}^\delta, \tag{1.30}$$

where $A \in \mathbb{R}^{m\times n}$ is typically a severely ill-conditioned operator, $\mathbf{b}^\delta \in \mathbb{R}^m$ represents the measured data corrupted by some noise $\boldsymbol{\eta}_\delta \in \mathbb{R}^m$, and $\mathbf{x} \in \mathbb{R}^n$ denotes the unknown image we would like to recover. We further assume that the measured data $\mathbf{b}^\delta$ satisfies

$$\mathbf{b}^\delta = \mathbf{b} + \boldsymbol{\eta}_\delta, \qquad \|\boldsymbol{\eta}_\delta\| \leq \delta, \tag{1.31}$$

where $\mathbf{b} \in \mathbb{R}^m$ represents the unobserved, noise–free data, and $\delta > 0$ serves as an upper bound of the noise level. The first challenge in solving the linear system (1.30) is that a solution may not exist, or there could be infinitely many possible one. To guarantee at least the existence of a solution, we replace the linear system in (1.30) with the least-squares problem

$$\arg\min_{\mathbf{x}\in\mathbb{R}^n} \|A\mathbf{x} - \mathbf{b}^\delta\|_2^2. \tag{1.32}$$

Problem (1.32) can also be derived from (1.30) by following a maximum likelihood statistical approach [132], assuming that the noise in the data is Gaussian.

Although a solution to (1.32) exists, it may not be unique. To achieve uniqueness, one possible strategy is to select the solution with the minimum norm among all potential solutions. In the noise–free case, this would be given by $\mathbf{x}^\dagger = A^\dagger\mathbf{b}$. However, this approach is impractical because, in reality, we do not have access to the noise-free data $\mathbf{b}$, and in the presence of noise, the minimum norm solution $\mathbf{x}^\dagger$ coincides with the naive one. Moreover, since our problem does not depend continuosly on the data, it is clear that we need a proper strategy to compute approximate solutions that are less sensitive to fluctuation in the observation. Let $\mathbf{x}_{\text{gt}} \in \mathbb{R}^n$ be the real image such that $A\mathbf{x}_{\text{gt}} = \mathbf{b}$, from classical perturbation theory we know that

$$\frac{\|\mathbf{x}_{\text{gt}} - \mathbf{x}\|_2}{\|\mathbf{x}_{\text{gt}}\|_2} \leq \mu_2(A)\frac{\|\boldsymbol{\eta}_\delta\|_2}{\|\mathbf{b}\|_2}.$$

Since the condition number of the operator A is typically large, this implies that the computed solution $\mathbf{x}$ may be far from the real one $\mathbf{x}_{\text{gt}}$. For image deblurring, an intuitive strategy, suggested by the analysis of the inverse of the noise described in the first section, is to use the TSVD to discard all singular values of the blur operator A below a certain tolerance. Although it is easy to compute the TSVD solution $\mathbf{x}_{TSVD}$ for different truncation parameters k, the need for the SVD of A, or at least the computation of the leading k singular values and vectors, makes this method computationally overwhelming for large-

scale problems. Therefore, there is a need for other regularization strategies that are better suited for large computational problems.

1.4.1 Tikhonov regularization

The most successful and widely used regularization method is the *Tikhonov regularization* [135]. It explicitly incorporates the regularity requirement in the formulation of the problem. In its simplest form, the regularized solution $\mathbf{x}_\lambda$ is defined as the solution of the penalized minimization problem

$$\min_{\mathbf{x}\in\mathbb{R}^n} \|A\mathbf{x} - \mathbf{b}^\delta\|_2^2 + \lambda\|\mathbf{x}\|_2^2. \tag{1.33}$$

The functional we aim at minimizing in (1.33) is defined by the summation of two terms. The first one, $\|A\mathbf{x} - \mathbf{b}^\delta\|_2^2$, is called the *data fidelity term* and measures the goodness–of–fit, that is how well the solution $\mathbf{x}$ predict the collected data $\mathbf{b}^\delta$. Clearly, if the data fidelity term is too large then the solution $\mathbf{x}$ cannot be considered as a good approximation of the real one. However, we should not make the residual too small, as this may result in overfitting the noise in the data. The second term in (1.33) is called the *regularization term* and it measures the regularity of the solution [80]. In general, it can be chosen in many different ways since it encodes prior information about the true solution [47]. The balance between the two terms is controlled by the *regularization parameter* $\lambda > 0$. The larger we choose it the more weight is given to the regularization. On the other hand, the smaller the λ, the more importance is given to the data fidelity term resulting in solution that are less regular. For instance, if $\lambda = 0$ we obtain the naïve solution that will be completely corrupted by the presence of the noise. The minimization of a functional, as in (1.33), which can be expressed as the sum of a data fidelity term and a regularization term, can be approached using a maximum a posteriori framework [70], always starting from the problem (1.30).

The regularized solution of problem (1.33) can be explicitely computed. Indeed, by differentiating both terms, we obtain that

$$\mathbf{x}_\lambda = (A^TA + \lambda I)^{-1}A^T\mathbf{b}^\delta. \tag{1.34}$$

Further insights about the Tikhonov solution $\mathbf{x}_\lambda$ can be obtained using the SVD of the blur operator A. Indeed, let $A = U\Sigma V^H$ be the SVD of A, we have

$$\begin{aligned}\mathbf{x}_\lambda &= (V\Sigma^T\Sigma V^H + \lambda VV^H)^{-1}V\Sigma^TU^H\mathbf{b}^\delta \\ &= V(\Sigma^T\Sigma + \lambda I)^{-1}\Sigma^TU^H\mathbf{b}^\delta \\ &= V\Phi_\lambda\Sigma^\dagger U^H\mathbf{b}^\delta,\end{aligned}$$

where $\Phi_\lambda \in \mathbb{R}^{n\times n}$ is the diagonal matrix with

$$(\Phi_\lambda)_{i,i} = \begin{cases} \frac{\sigma_i^2}{\sigma_i^2+\lambda}, & i = 1,\dots,r, \\ 0 & i = r+1,\dots,n, \end{cases} \tag{1.35}$$

where $r = \text{rank}(A)$. If we insert the singular values and vectors, we get

$$\mathbf{x}_\lambda = \sum_{i=1}^{r} \left(\frac{\sigma_i^2}{\sigma_i^2 + \lambda^2} \right) \frac{\mathbf{u}_i^T \mathbf{b}^\delta}{\sigma_i} \mathbf{v}_i.$$

Tikhonov regularization can be categorized as a *filtering method*. In general, this class of regularization methods produces solutions $\mathbf{x}_{\text{reg}}$ that can be represented as a filtered SVD expansion of the form

$$\mathbf{x}_{\text{reg}} = \sum_{i=1}^{r} \varphi_i \frac{\mathbf{u}_i^T \mathbf{b}^\delta}{\sigma_i} \mathbf{v}_i,$$

where φ_i are the *filter factors* associated with the method. In the case of Tikhonov regularization, this corresponds to a *low-pass filter*. Specifically, for a fixed λ, the filter factors φ_i are close to 1 when σ_i is large, preserving the low-frequency components, while $\varphi_i \approx 0$ when σ_i is small, thereby eliminating the high–frequency components.

Remark 1.4.1. *The TSVD can be seen as a filtering method with filter factors defined as*

$$\varphi_i = \begin{cases} 1, & \text{if } i \leq s \\ 0, & \text{if } i > s \end{cases},$$

where the parameter s indicates the number of singular values we want to keep.

It is well known that it is often possible to improve the quality of the regularized solution determined by Tikhonov regularization by replacing problem (1.33) with

$$\min_{\mathbf{x} \in \mathbb{R}^n} \|A\mathbf{x} - \mathbf{b}^\delta\|_2^2 + \lambda \|L\mathbf{x}\|_2^2, \tag{1.36}$$

where $L \in \mathbb{R}^{s \times n}$ is called *regularization matrix*. The minimization problem (1.33) is commonly referred to as Tikhonov regularization in *standard form*, while (1.36) is refferd to as Tikhonov regularization in *general form*. Let $\mathcal{N}(L)$ and $\mathcal{R}(L)$ denote the null space and the range of the linear operator L, we assume that L is chosen so that

$$\mathcal{N}(L) \cap \mathcal{N}(A) = \{\mathbf{0}\}.$$

The assumption that the intersection between the null spaces of the blur operator A and the regularization matrix L is trivial is necessary to ensure that problem (1.36) admits a unique solution for any $\lambda > 0$. Popular choices for the regularization matrix L include framelet operators and differential operators [129, 61, 45, 38]. Recently, fractional differential operators have also been explored to enhance diffusion, particularly in the context of denoising problems [5, 141]. In the following chapters, we will select L to be the graph Laplacian of properly constructed graph obtained from a given approximation of the true image. We will demonstrate how accurate the reconstruction can be when the regularization term encodes the correct information. Additionally, we will explore a fractional extension of the graph Laplacian operator that can achieve even more accurate reconstructions.

1.4.2 Iterative regularization methods

The regularization methods discussed so far are designed for problems where it is feasible to compute the SVD or compute the Tikhonov solution via the least squares formulation (1.36) solving the associated linear system. However, many real world applications lead to large matrices where these factorization are too computational and time demanding. The essence of an effective regularization method lies in performing matrix–vector multiplications avoiding any factorization of the operators involved to minimize the overall computational costs. Moreover, one should be able to select the regularization parameter λ without solving the problem from scratch for each new parameter. This two requirements lead to the introduction of iterative regularization methods where the number of iterations plays the role of regularization parameter. This implies that in the early stage the filtered solution $\mathbf{x}_k$ tends to become increasingly accurate approximation of the exact solution. However, at later stages, the iterates start to diverge from the sharp image and instead converge to the naïve solution. This behavior is called *semiconvergence* and if we are able to interrupt the iteration at the right moment, then we obtain a large–scale regularization method.

Landweber method

The prototypical iterative regularization algorithm for least–squares problem is the Landweber method [93, 62]. This is the simplest gradient descent algorithm for solving problem (1.33). It can also be interpeted as the Richardson method applied to the normal equations

$$A^H A\mathbf{x} = A^H \mathbf{b}^\delta,$$

and its iterations read as

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha A^H(\mathbf{b}^\delta - A\mathbf{x}_k). \tag{1.37}$$

If $\alpha \in (0, \frac{2}{\sigma_1^2})$, then the Landweber method is convergent since the spectral radius of $I - \alpha A^H A$ is smaller than one. As done for the Tikhonov regularization, we can get further insights about the behavior of the Landweber algorithm by interpreting it as a filtering method. Taken

$$\mathbf{x}_0 = \mathbf{0}$$

as initial value and exploting the iterations (1.37), we have

$$\begin{aligned}
\mathbf{x}_{k+1} &= (I - \alpha A^H A)\mathbf{x}_k + \alpha A^H \mathbf{b}^\delta \\
&= (I - \alpha A^H A)[(I - \alpha A^H A)\mathbf{x}_{k-1} + \alpha A^H \mathbf{b}^\delta] + \alpha A^H \mathbf{b}^\delta \\
&= (I - \alpha A^H A)^2 \mathbf{x}_{k-1} + (I - \alpha A^H A)\alpha A^H \mathbf{b}^\delta + \alpha A^H \mathbf{b}^\delta \\
&= \dots \\
&= (I - \alpha A^H A)^{k+1} \underbrace{\mathbf{x}_0}_{=0} + \alpha \sum_{i=0}^{k} (I - \alpha A^H A)^i A^H \mathbf{b}^\delta \\
&= \alpha \sum_{i=0}^{k} (I - \alpha A^H A)^i A^H \mathbf{b}^\delta.
\end{aligned}$$

Substituting the SVD form $A = U\Sigma V^H$ in the above computations, we obtain

$$\begin{aligned}
\mathbf{x}_{k+1} &= \alpha \sum_{i=0}^{k} (I - \alpha V \Sigma^T U^H U \Sigma V^H)^i V \Sigma^T U^H \mathbf{b}^\delta \\
&= \alpha \sum_{i=0}^{k} V (I - \alpha \Sigma^T \Sigma)^i V^H V \Sigma^T U^H \mathbf{b}^\delta \\
&= V M_k U^H \mathbf{b}^\delta, \qquad \text{(i)}
\end{aligned}$$

where $M_k = \alpha \sum_{i=0}^{k} (I - \alpha \Sigma^T \Sigma)^i \Sigma^T \in \mathbb{R}^{n \times m}$. Since $\operatorname{rank}(A) = r$, we can use the compact SVD to write

$$M_k = \begin{bmatrix} N_k & 0 \\ 0 & 0 \end{bmatrix}, \qquad N_k = \alpha \sum_{i=0}^{k} (I_r - \alpha \Sigma_r^T \Sigma_r)^i \Sigma_r^T \in \mathbb{R}^{r \times r}. \qquad \text{(ii)}$$

To conclude, set

$$\begin{aligned}
\tilde{\Phi}_k &= \alpha \sum_{i=0}^{k} (I_r - \alpha \Sigma_r^T \Sigma_r)^i \Sigma_r^T \Sigma_r \\
&= \operatorname{diag}_{j=1,\dots,r}(\theta_j^{(k)}), \qquad \text{(iii)}
\end{aligned}$$

where

$$\theta_j^{(k)} = \alpha \sum_{i=0}^{k} (1 - \alpha \sigma_j^2)^i \sigma_j^2 = 1 - (1 - \alpha \sigma_j^2)^{k+1}. \qquad \text{(iv)}$$

Thus, replacing (ii) and (iii) in equation (i), we can rewrite the Landweber iterative scheme as

$$\mathbf{x}_{k+1} = V M_k U^H \mathbf{b}^\delta = V \underbrace{\begin{bmatrix} \tilde{\Phi}_k & 0 \\ 0 & 0 \end{bmatrix}}_{\Phi_k} \Sigma^\dagger U^H \mathbf{b}^\delta, \qquad (1.38)$$

Lemma 1.4.2. *Let $A \in \mathbb{R}^{m\times n}$ be a matrix such that* $\operatorname{rank}(A) = r < \min(n, m)$ *and* $\alpha \in \left(0, \frac{2}{\sigma_1^2}\right)$. *Set* $\mathbf{x}_0 = \mathbf{0}$ *and let* $\{\mathbf{x}_k\}_{k\in\mathbb{N}} \subseteq \mathbb{R}^n$ *be the sequence of points by the Landweber iterations. Then,*

$$\lim_{k\to+\infty} \mathbf{x}_k = A^\dagger \mathbf{b}^\delta.$$

Proof. Since $\mathbf{x}_0 = \mathbf{0}$ by assumption, the sequence of points $\mathbf{x}_k$ can be expressed as in equation (1.38). To conclude, observe that, since $\alpha \in \left(0, \frac{2}{\sigma_1^2}\right)$, it follows

$$|1 - \alpha\sigma_j^2| < 1, \quad j = 1, \dots, r.$$

Thus, from equation (iv), we have

$$\lim_{k\to\infty} \theta_j^{(k)} = 1, \quad j = 1, \dots, r,$$

which concludes the proof.

□

Iterated Tikhonov

A slightly more sophisticated iterative regularization method is the Iterated Tikhonov algorithm. .

Let $\mathbf{x}_0$ be an avaiable approximation of the minimal norm solution $\mathbf{x}^\dagger$ and consider the minimization problem

$$\min_{\mathbf{x}\in\mathbb{R}^n} \|A\mathbf{x} - \mathbf{b}^\delta\|_2^2 + \lambda\|\mathbf{x} - \mathbf{x}_0\|_2^2. \tag{1.39}$$

If no approximation of $\mathbf{x}^\dagger$ is known, we can set $\mathbf{x}_0 = 0$, reducing problem (1.39) to the standard Tikhonov formulation (1.33). By defining the initial residual vector $\mathbf{r}_0$ and the error approximation $\mathbf{h}$ as

$$\mathbf{r}_0 = \mathbf{b}^\delta - A\mathbf{x}_0, \qquad \mathbf{h} = \mathbf{x}^\dagger - \mathbf{x}_0,$$

we can rewrite the initial problem (1.39) as the variational problem

$$\min_{\mathbf{h}\in\mathbb{R}^n} \|A\mathbf{h} - \mathbf{r}_0\|_2^2 + \lambda\|\mathbf{h}\|_2^2. \tag{1.40}$$

Therefore, with a suitable choice of the regularization parameter $\lambda > 0$, if we are able to compute an approximate solution $\mathbf{h}_1$ of (1.40), we can obtain an improved approximation $\mathbf{x}_1$ of $\mathbf{x}^\dagger$ by simply defining

$$\mathbf{x}_1 = \mathbf{x}_0 + \mathbf{h}_1. \tag{1.41}$$

Once the first refined approximation $\mathbf{x}_1$ is computed, we can iterate this procedure by computing the new residual vector $\mathbf{r}_1 = \mathbf{b}^\delta - A\mathbf{x}_1$ and solving problem (1.40) again with $\mathbf{r}_1$ replacing $\mathbf{r}_0$, to obtain a new approximation of the error $\mathbf{h}_2$. Repeated application of this refinement strategy defines what is called the Iterated Tikhonov method. The iterations can

be compactly expressed as

$$\mathbf{x}_{k+1} = \mathbf{x}_k + (A^H A + \lambda I)^{-1} A^H \mathbf{r}_k, \quad k = 0, 1, \ldots, \tag{1.42}$$

where $I \in \mathbb{R}^{n \times n}$ is the identity matrix, and they can be terminated with the aid of some stopping criteria.

Remark 1.4.3. *By making explicit the residual vector in (1.42), we have*

$$\mathbf{x}_{k+1} = \mathbf{x}_k + (A^H A + \lambda I)^{-1} A^H (\mathbf{b}^\delta - A\mathbf{x}_k), \quad k = 0, 1, \ldots.$$

Thus, the iterated Tikhonov method can be interpreted as a preconditioned version of the Landweber method in which the damping parameter α in (1.37) has been replaced by the preconditioner $P = (A^H A + \lambda I)^{-1}$ [120].

Similar preconditioning strategies can be explored also for other Krylov methods or using other filtering techniques [79, 14].

Remark 1.4.4. *The iteration matrix T associated with the scheme (1.42) is given by*

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

Therefore, since the spectral radius satisfies $\rho(T) < 1$ for all $\lambda > 0$, we can conclude that the iterated Tikhonov method is convergent.

All the previous analysis was conducted considering the standard form of the Tikhonov model (1.39). However, in [31], the author extends the formulation of the iterated Tikhonov regularization to include a general penalty term. From this perspective, the new iteration scheme is written as

$$\mathbf{x}_{k+1} = \mathbf{x}_k + (A^H A + \lambda L^H L)^{-1} A^H \mathbf{r}_k, \quad k = 0, 1, \ldots, \tag{1.43}$$

where L is the regularization operator. Unfortunately, the convergence of this iterative method is not as straightforward as in the standard case, and additional assumptions on A and L are required.

Remark 1.4.5. *Note that in both the standard and generalized cases, the matrix $P^{-1} = \left(A^H A + \lambda L^H L\right)$ (or $P^{-1} = \left(A^H A + \lambda I\right)$ in the standard case) represents the Hessian matrix of the regularized least squares problem. Thus, the damping parameter α is replaced with a second–order method.*

The choice of the regularization parameter λ in the iterated Tikhonov method is important to accelerate the convergence without spoiling the quality of the computed solution, and many strategies have been proposed in the literature [77, 57]. In general, there are two main strategies for selecting the regularization parameter. The simpler one, consider λ to be independent from the number of iterations k and the iterative method is said to be *stationary*. On the other hand, in the *non–stationary* case, the value of λ changes at each iteration [78].

In many applications the non–stationary version of the iterated Tikhonov regularization has been found to give more accurate approximations of $\mathbf{x}^\dagger$ and often even faster convergence than its stationary version [77]. A commonly used choice for the regularization parameter in the non–stationary case is to consider a geometric sequence, namely,

$$\lambda_k = \lambda_0 q^k, \qquad \lambda_0 > 0, \quad 0 < q < 1, \quad k = 0, 1, \ldots. \tag{1.44}$$

Although we will not address them in this thesis, there exist sophisticated Krylov methods, such as Conjugate Gradient and others, that possess regularization properties similar to those described here[76, 41].

1.4.3 Choosing the regularization parameter

We have at our disposal several different regularization methods. However, the main thing we are still missing is a reliable and automated technique for choosing the regularization parameter, i.e. s, for TSVD, or λ in the case of Tikhonov regularization [78, 81, 124]. What we would like is an efficient, and reliable method for computing the regularization parameter which does not require the computation of the SVD, which is infeasible for large problems. Unfortunately, such a method has yet to be found. What we have at our disposal is a collection of strategies which, under certain conditions, will provide a good estimate of the regularization parameter. However, all of them can and possibly will occasionally fail to produce good results. Any method developed for choosing the regularization parameter shoud seek to minimize the error in the regularized solution. For a better understanding on how the error is influenced by this choice, we will take a closer look at the approximation error. For simplicity, the outset for our discussion is the Tikhonov regularization.

Recall from the previous section that the filtered solution of the Tikhonov regularization is given by

$$\mathbf{x}_\lambda = V \Phi_\lambda \Sigma^\dagger U^H \mathbf{b}^\delta, \tag{1.45}$$

where the diagonal matrix Φ_λ is the filtering matrix defined in equation (1.35). We recall that the observed data consists of an exact signal plus some additive noise, i.e.,

$$\mathbf{b}^\delta = A\mathbf{x}_{\text{gt}} + \boldsymbol{\eta}_\delta.$$

It follows that the error in the Tikhonov regularized solution (1.45) is then given by

$$\begin{aligned}
\mathbf{x}_{\text{gt}} - \mathbf{x}_\lambda &= \mathbf{x}_{\text{gt}} - V \Phi_\lambda \Sigma^\dagger U^H \mathbf{b}^\delta \\
&= \mathbf{x}_{\text{gt}} - V \Phi_\lambda \Sigma^\dagger U^H A \mathbf{x}_{\text{gt}} - V \Phi_\lambda \Sigma^\dagger U^H \boldsymbol{\eta}_\delta \\
&= \underbrace{V(I - \Phi_\lambda) V^H \mathbf{x}_{\text{gt}}}_{\Delta \mathbf{x}_{\text{bias}}} - \underbrace{V \Phi_\lambda \Sigma^\dagger U^H \boldsymbol{\eta}_\delta}_{\Delta \mathbf{x}_{\text{pert}}}.
\end{aligned}$$

The first term $\Delta \mathbf{x}_{\text{bias}}$ in the above expression is called the *regularization error* and it comes from the introduction of the filtering in the reconstruction. The second error term $\Delta \mathbf{x}_{\text{pert}}$,

is the *perturbation error* and it is due to the inversion and filtering of the noise component in the data. The main purpose of regularization techniques is to prevent this perturbation error from blowing up in magnitude and corrupting the solution.

The goal of the regularization parameter λ is to balance the size of the two errors term $\Delta\mathbf{x}_{\text{bias}}$ and $\Delta\mathbf{x}_{\text{pert}}$. Precisely, if λ is chosen very small, then all the filter factors φ_i are close to 1. Thus, the filtering matrix Φ_λ is close to the identity matrix and $\Delta\mathbf{x}_{\text{bias}}$ is small while $\Delta\mathbf{x}_{\text{pert}}$ is large. On the opposite, if λ is chosen to be large, then many filter factors φ_i are small and hence $\Delta\mathbf{x}_{\text{pert}}$ will be also small but since Φ_λ will not be close to the identity anymore, then $\Delta\mathbf{x}_{\text{bias}}$ will be large. Thus, we would like to compute a proper value for the regularization parameter λ that balances the action of the two error terms.

To this aim, the simplest method is the *discrepancy principle* and it basically requires to choose λ such that

$$\|A\mathbf{x}_\lambda - \mathbf{b}^\delta\|_2 < \tau\|\delta\|_2, \tag{1.46}$$

where δ is the noise level defined in (1.31) and τ is a user–defined parameter and is usually chosen as $\tau = 1.01$. Due to its simplicity, the discrepancy principle is often the favored parameter–choice method in theoretical analysis of regularization methods, that is when proving that a regularized solution converges to the exact solution as the noise level $\delta \to 0$. Moreover, it requires only an efficient root finder for the problem in equation (1.46). The main drawback of the discrepancy principle is that we often do not know $\|\delta\|_2$ exactly but maybe we just know a rough estimate. Unfortunately, the quality of the computed regularization parameter λ is very sensitive to the accuracy of the estimate of the noise. In particular, a too small estimate can lead to dramatic undersmoothing while too large values can cause oversmoothing in the solution.

A different strategy for computing the regularization parameter looks for the value of λ such that $A\mathbf{x}_\lambda$ predicts the noise–free data $\mathbf{b}$ as well as possible. It is possible to carry out the derivation of this method and the easiest way is by considering the TSVD filtering method. However, we do not report all the analysis here which can be found with further details in [81]. The main question is: how can we estimate the regularization parameter for a given problem in which the noise–free observation $\mathbf{b}$ is not available? In cross validation, one separates the given data into two sets ad uses one of the sets to compute a solution which is then used to predict the elements in the other set. For example, we could leave out b_i^δ, the ith element of $\mathbf{b}^\delta$, and then compute the Tikhonov solution based on the reduced problem

$$\mathbf{x}_\lambda^{(i)} = \left((A^{(i)})^T A^{(i)} + \lambda I_{n-1}\right)^{-1} (A^{(i)})^T \mathbf{b}^{\delta,(i)},$$

where $A^{(i)}$ and $\mathbf{b}^{\delta,(i)}$ are the shortened version of A and $\mathbf{b}^\delta$ with the ith row and element, respectively, left out. Then, we can use $\mathbf{x}_\lambda^{(i)}$ to predict the element b_i^δ that was left out via the "missing" row of A, through the expression $A(i,:)\mathbf{x}_\lambda^{(i)}$. Leaving out some technical arguments, the goal is then to choose the regularization parameter λ such that it solves the

minimization problem

$$\min_{\lambda} \frac{1}{m} \sum_{i=1}^{m} \left(\frac{A(i,:)\mathbf{x}_\lambda - b_i^\delta}{1 - h_{i,i}} \right),$$

where $\mathbf{x}_\lambda$ is the Tikhonov solution, and $h_{i,i}$ are the diagonal elements of the matrix $A(A^TA + \lambda I)^{-1}A^T$.

Unfortunately, the diagonal elements $h_{i,i}$ will change if we permute the rows of A, and thus the solution depends on the particular ordering of the data. The method of *Generalized Cross Validation (GCV)* was introduced to remedy this inconvenience, by replacing each diagonal element $h_{i,i}$ with the average of the diagonal elements. Thus, the simplified minimization problem takes the form

$$\min_{\lambda} \frac{1}{m} \sum_{i=1}^{m} \left(\frac{A(i,:)\mathbf{x}_\lambda - b_i^\delta}{1 - \text{trace}(A(A^TA + \lambda I)^{-1}A^T)/m} \right)^2. \tag{1.47}$$

Negletcting a factor m, the GCV parameter–choice method for Tikhonov regularization thus takes the form

$$\lambda_{GCV} = \min_{\lambda} \frac{\|A\mathbf{x}_\lambda - \mathbf{b}^\delta\|_2^2}{\left(m - \sum_{i=1}^{n} \varphi_i\right)^2}.$$

1.5 The Graph Laplacian

In the previous section, we introduced and analyzed different regularization strategies to reduce the sensitivity of reconstructions from the presence of noise. In the next two chapters, we will explore different variational models designed to achieve better results compared to the standard methods discussed so far. However, the core structure of all these variational models will be based on the general Tikhonov regularization framework, that is

$$\min_{\mathbf{x} \in \mathbb{R}^n} \|A\mathbf{x} - \mathbf{b}^\delta\|_2^2 + \lambda \|L\mathbf{x}\|_2^2.$$

In this final section, we will focus on a specific choice for the regularization matrix L, which will play a crucial role in the subsequent chapters within a broader framework. The effectiveness of this linear operator is rooted in the natural connection between images and graphs. Furthermore, we will demonstrate how to intuitively encode valuable image information within the graph structure, resulting in an efficient regularization operator.

1.5.1 Graph Theory

An *unweighted graph* is a pair $G = (P, E)$, where P is the vertex set and $E \subseteq P \times P$ is the set containing all the edges. A graph G is said to be undirected if $(i, j) \in E$ implies that $(j, i) \in E$, otherwise we say that the graph is directed. Sometimes, a graph G is associated with a measure $\omega : E \to \mathbb{R}^+$ which associates each edge of the graph with a unique positive value that is referred to as the weight of the edge. This leads us to the following

Definition 1.5.1. *A weighted graph over a vertex set P is a quadruple $G = (P, E, w, \mu)$*

given by:

- *A nonnegative* edge-weight *function* $w\colon E \to [0,\infty)$ *that satisfies*

  i) Symmetry: $w(p,q) = w(q,p)$ *for every* $p,q \in P$*;*

  ii) No self-loops: $w(p,p) = 0$ *for every* $p \in P$*.*

- *A positive* node measure $\mu\colon P \to (0,\infty)$.

The definition of weighted graph can be relaxed, for instance, by allowing non-symmetric edge-weight functions and self-loops. We will say that two nodes are connected if $w(p,q) > 0$, and in that case we write $p \sim q$. A finite *walk* is a finite sequence of nodes $\{p_i\}_{i=0}^{k}$ such that $w(p_i, p_{i+1}) > 0$ for $i = 0,\dots,k-1$. A subset $Q \subseteq P$ is *connected* if for every pair of nodes $p,q \in Q$ there is a finite walk such that $p_0 = p$, $p_k = q$, and each p_i belongs to Q. A connected subset $Q \subseteq P$ is a *connected component* of P if it is maximal with respect to the ordering of inclusion.

Let $G = (P, E, w, \mu)$ be an undirected, weighted and connected graph with neither multi–edges nor self–loops. Assuming that $|P| = n$, where $|\mathcal{A}|$ is the cardinality of the set $\mathcal{A}$, i.e., n is the number of nodes in G, we define the function spaces

$$\mathcal{V}_P \coloneqq \{\mathbf{x} \mid \mathbf{x} : P \to \mathbb{R}\}, \qquad \mathcal{E} \coloneqq \{\varphi : E \to \mathbb{R}\}.$$

Assuming an implicit ordering of the elements in P, it is clear that $\mathcal{V}_P \simeq \mathbb{R}^n$. Hence, we use the same notation for finite-dimensional vectors, even when referring to elements $\mathbf{x} \in \mathcal{V}_P$. For a parameter $r \in [0,1]$, we defined the inner products on $\mathcal{V}_P$ and $\mathcal{E}$ (and hence inner product norms $\|\cdot\|_{\mathcal{V}_P}$ and $\|\cdot\|_{\mathcal{E}}$) as

$$\langle \mathbf{x}, \mathbf{y} \rangle_{\mathcal{V}_P} \coloneqq \sum_{p \in P} \mathbf{x}(p)\mathbf{y}(p)\mu(p)^r, \qquad \langle \varphi, \Phi \rangle_{\mathcal{E}} \coloneqq \frac{1}{2} \sum_{p,q \in P} \varphi(p,q)\Phi(p,q)w(p,q).$$

Then, the graph variants of the gradient and Laplacian operators are defined as

Definition 1.5.2. *The* graph gradient $\nabla\colon \mathcal{V}_P \to \mathcal{E}$ *and the* graph Laplacian $\Delta\colon \mathcal{V}_P \to \mathcal{V}_P$ *associated to the weighted graph* $G = (P, E, w, \mu)$ *are defined by the actions*

$$(\nabla \mathbf{x})_{p,q} \coloneqq \begin{cases} \mathbf{x}(q) - \mathbf{x}(p), & \text{if } (p,q) \in E, \\ 0, & \text{otherwise,} \end{cases}$$

and

$$\Delta \mathbf{x}(p) \coloneqq \frac{1}{\mu(p)^r} \sum_{q \in P} w(p,q)\left(\mathbf{x}(p) - \mathbf{x}(q)\right). \tag{1.48}$$

Remark 1.5.3. *The graph Laplacian operator* (1.48) *is positive, semi–definite, and self–adjoint with respect to* $\mathcal{V}_P$*, and it does not depend on the ordering (or labeling) of the elements in vertex set* P*. Moreover, the graph variants of the gradient and Laplacian operators are*

related via

$$\langle \mathbf{x}, \Delta \mathbf{y} \rangle_{\mathcal{V}_P} = \langle \nabla \mathbf{x}, \nabla \mathbf{y} \rangle_{\mathcal{E}}.$$

We can rewrite the graph Laplacian operator (1.48) in a more compact way. Indeed, given a weighted graph $G = (P, E, w, \mu)$, it can be represented it by means of the so called *adjacency matrix*. Let again be $|P| = n$, then we denote the adjacency matrix $\Omega \in \mathbb{R}^{n \times n}$ of the graph G as

$$\Omega_{p,q} = \begin{cases} w(p,q) & \text{if } (p,q) \in E, \\ 0 & \text{otherwise.} \end{cases}$$

Let D denote the *degree matrix*, a diagonal matrix with elements given by

$$D_{p,p} = \deg(p) = \sum_{q \in P} \Omega_{p,q}, \qquad p \in P,$$

and let $\mathcal{D}$ denote the diagonal matrix whose elements are defined as $\mathcal{D}_{p,p} = \mu(p)$. Then, the graph Laplacian can be defined as the linear operator

$$\Omega = \mathcal{D}^{-r}(D - \Omega). \tag{1.49}$$

The choice of r is important. Suppose that $\mu(p) = \deg(p)$, then, for $r = 0$, $\Delta = D - \Omega$ is the standard *unnormalised (or combinatorical) Laplacian*. For $r = 1$, we obtain the so called *random walk Laplacian*. There is also an important Laplacian not covered by this definition: the *symmetric normalised Laplacian* $\Delta \coloneqq I - D^{-\frac{1}{2}} \Omega D^{-\frac{1}{2}}$.

1.5.2 Graph associated to an image

In this section, we will explore the relationship between images and graphs, and how the information contained within an image can be encoded into the graph Laplacian operator.

Given an image $X \in \mathbb{R}^{n \times m}$, we aim to construct a weighted graph $G = (P, E, w, \mu)$ associated with it. Since an image consists of a grid of pixels, it is natural to identify each pixel as a node in the graph. Specifically, we can represent each pixel as an ordered pair $p = (i, j) \in \mathbb{Z}^2$, where $i = 1, \ldots, n$ and $j = 1, \ldots, m$. In this way, the set of nodes P corresponds directly to the set of pixels in the image. To fully define the weighted graph, we need to specify two additional components: the weight function $w \colon E \to \mathbb{R}$, which assigns a weight to each edge $e \in E$ connecting two nodes, and the node measure $\mu \colon P \to (0, \infty)$, which assigns a positive measure to each node in the graph. We would like to design those functions in order to capture the relationships bewtween neighboring pixels. Indeed, they can be designed to reflect various image features, such as intensity differences, spatial proximity, or other characteristics, making the graph representation a powerful tool for image analysis.

Distance-based graph creation

Given a finite set of nodes $P = \{p \mid p = 1, \dots, n\}$, fix a distance $\mathrm{dist}(\cdot,\cdot)$ on the set P and a nonnegative function $h_{\mathrm{d}} \colon \mathbb{R} \to [0, +\infty)$ such that $h_{\mathrm{d}}(0) = 0$. Then it follows that

$$w_{\mathrm{d}}(p, q) \coloneqq h_{\mathrm{d}}(\mathrm{dist}(p, q))$$

is an edge-weight function on P since w_{d} satisfies Definition 1.5.1. It is based on the geometric properties of P induced by the distance $\mathrm{dist}(\cdot,\cdot)$. The magnitude of the connections between two nodes p and q is then regulated by h_{d}. Fix now an element $\mathbf{x} \in \mathcal{V}_P$, another nonnegative function $h_{\mathrm{i}} \colon \mathbb{R} \to [0, +\infty)$ and finally define

$$w_{\mathbf{x}}(p, q) \coloneqq \underbrace{w_{\mathrm{d}}(p, q)}_{\text{geometry}} \cdot \underbrace{h_{\mathrm{i}}(|\mathbf{x}(p) - \mathbf{x}(q)|)}_{\mathbf{x}\text{ intensity}}. \tag{1.50}$$

The edge-weight function (1.50) depends on both the "physical" distance between two nodes, thanks to w_{d}, and the "variation of intensity" of $\mathbf{x}$, thanks to $h_{\mathrm{i}}(|\mathbf{x}(p) - \mathbf{x}(q)|)$. This dual dependence has a twofold effect: it can separate nodes that reside in different and unrelated regions of the space, and it weights the magnitude of the connections depending on the difference of intensities of $\mathbf{x}$.

Now choose any strictly positive function, which may depend on $\mathbf{x}$,

$$\mu_{\mathbf{x}}(p) > 0 \quad \forall\, p \in P. \tag{1.51}$$

Therefore, following Definition 1.5.1, for any fixed $\mathbf{x} \in \mathcal{V}_P$, we have defined a graph G, on the set P, induced by $\mathbf{x}$. We will write then $\Delta_{\mathbf{x}}$ to indicate the associated graph Laplacian, as in equation (1.48).

Our aim is to construct a graph Laplacian operator associated to an image that will work as a regularization operator. Because of what we said before, an intuitive choice for the geometry part of the weight function (1.50) is to take $w_{\mathrm{d}}(p, q)$ dependent on the "physical" distance between pixels, that is

$$\mathrm{dist}(p, q) \coloneqq \|p - q\|_1 \quad \text{and} \quad h_{\mathrm{d}}(t) \coloneqq \mathbb{1}_{(0,R]}(t),$$

where $\mathbb{1}_{(0,R]}$ is the indicator function of the set $(0, R]$ and R is a control parameter that tells the maximum distance allowed between two pixels to be neighbors. If $0 < \|p-q\|_1 \leq R$, then p and q are connected with an edge of magnitude 1, that is $w_{\mathrm{d}}(p, q) = 1$. To enhance more connections between neighbouring pixels, another possible choice is to replace the ℓ^1–norm with the ℓ^∞–norm. For better understanding of how to construct a graph Laplacian on an image, in the following we will just consider the ℓ^1–norm case.

To understand how to define the intensity term in the definition (1.50) of the weight function, recall that a gray-scale image is given by the light intensities of its pixels. That is, a gray-

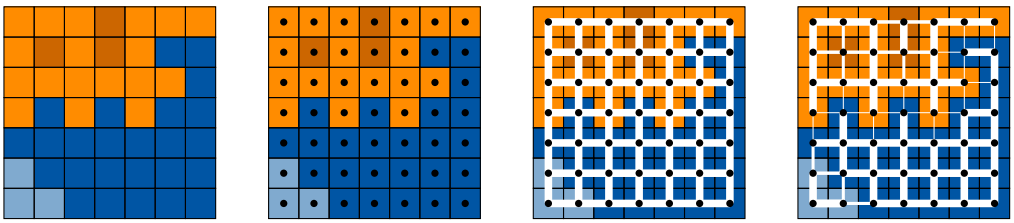


Figure 1.5: Simple outline of how to build a graph from an image $\mathbf{x}$. To be read from left to right. Left: a 7×7 pixels image made by orange-like and blue-like square pixels. The color intensity of each pixel is given by the pixel-wise evaluation of a function $\mathbf{x}$. Center-left: each pixel corresponds to one node, represented by a black circle. Since the pixels are disposed on a grid, each node can be associated to an ordered pair in $\mathbb{Z}^2$. Center-right: the geometric edge-weight function w_{d} in eq. (1.50) is given by $\mathbb{1}_{(0,1]}(\|p-q\|_1)$, that is, two nodes p, q are connected if and only if $\|p-q\|_1 = 1$, and in that case the magnitude of the connection is one. Right: the magnitude of an edge between two nodes is then weighted by $h_{\mathrm{i}}(\|\mathbf{x}(p)-\mathbf{x}(q)\|) \in (0,1]$, where $h_{\mathrm{i}}(t) = \exp\{-t^2/\sigma^2\}$ is the Gaussian function. The role of h_{i} is to measure the difference of intensity between two adjacent pixels, and it is close to zero when two pixels have very different color intensities. This is represented by the different thicknesses of the edges connecting two adjacent pixels, where a thick edge means a very similar color intensity and a thin edge means a very different color intensity.

scale image can be represented by a function $\mathbf{x} \in \mathcal{V}_P$ such that $\mathbf{x}(p) \in [0,1]$, where 0 means black and 1 means white. A common choice to weight the connection of two different pixels by their light intensities is to use the Gaussian function, that is

$$h_{\mathrm{i}}(t) \coloneqq \mathrm{e}^{-\frac{t^2}{\sigma^2}}, \qquad \sigma > 0.$$

The reason lies in the relationship between the heat kernel and the discretization of the Laplacian on a manifold. See [40, 73, 103] for applications of the Gaussian function to define the weights of the graph Laplacian. We arrive then at the definition of the edge-weight function in (1.50), applied to our case, that is

$$w_{\mathbf{x}}(p,q) = \mathbb{1}_{(0,R]}(\|p-q\|_1)\mathrm{e}^{-\frac{|\mathbf{x}(\mathrm{p})-\mathbf{x}(\mathrm{q})|^2}{\sigma^2}}. \tag{1.52}$$

The values of σ only modify the shape of the Gaussian weight function: small values of σ correspond to a tight distribution, while larger values result in wider curves. This means that we can control how close the intensity of two pixels should be in order to obtain strong connections. Conversely, R influences the sparsity of the graph Laplacian operator controlling the number of connections between neighbouring pixels. Specifically, a larger R leads to a denser matrix, which implies that each matrix-vector product requires more computations. However, the useful values of R are usually small. Indeed, it is unlikely that pixels that are on opposite side of an image are correlated in some way. It is more common that the stronger correlation are in a limited neighbour of each pixel. For these reasons, standard choices for these parameters are $R \leq 5$, to keep the computational cost low at each iteration, and $\sigma \leq 10^{-3}$ to avoid strong connections between uncorrelated pixels whose intensities are sufficiently close to each other. These choices are used, for instance, in [3]. For an even better understanding on how this strategy works, in Figure 1.5, we outlined how

to build a graph from an image $\mathbf{x}$. To make it as simple as possible we just choose $R = 1$ so each pixels have connections with just its first neighbours. Looking at Figure 1.5, note that if one replace the ℓ^1–norm with the ℓ^∞ one in the definition of the edge weight function (1.52), then we would also have diagonal connections between pixels. This is clearly useful because pixels on the diagonal could be still related to the pixel considered and consequently we can encode more information of the image in the graph.

Briefly, to conclude, consider the case of colored images, that is a little bit different. For example, in the RGB representation, the color of a pixel is given by the combination of the light intensities of three channels R(ed), G(green), and B(lue). Therefore, in principle a colored image should be regarded as a function $\mathbf{x} : P \to \mathbb{R}^3$, where $\mathbf{x}(p) = (\mathbf{x}_R(p), \mathbf{x}_G(p), \mathbf{x}_B(p))$ is a vector-valued function whose elements represent the light intensities for each channel. However, it is common to assume that the ill-posed operator A acts independently on each channel, and therefore the regularization is made on each channel separately. If for some reason this assumption can not be made, then we can simply modify the definition (1.50) in the following way:

$$w_{\mathbf{x}}(p, q) \coloneqq w_{\mathrm{d}}(p, q) \cdot h_{\mathrm{i}}(\|\mathbf{x}(p) - \mathbf{x}(q)\|),$$

where $\|\cdot\|$ is any appropriate norm in $\mathbb{R}^3$.

The graphLa+Ψ method

In this chapter we investigate a variational method for ill-posed problems, named graphLa+Ψ, which embeds a graph Laplacian operator in the regularization term. In Section §1.5, we highlighted the natural connection between graphs and images. However, one of the main drawbacks of using the graph Laplacian operator in imaging problems is that it requires the computation of an initial reconstruction. Ideally, we would like to incorporate information about the true solution directly into the regularization term. Unfortunately, the observed image $\mathbf{b}^\delta$ is completely corrupted by noise, and in the case of the image deblurring problem, we also have to deal with blur. Even worse, in the CT problem, the pixels of the sinogram provide no direct information about the attenuation coefficient of the object we aim to reconstruct. The novelty of this method lies in constructing the graph Laplacian based on a preliminary approximation of the solution, which is obtained using any existing reconstruction method Ψ_Θ from the literature. In this way, the regularization term is both dependent on and adaptive to the observed data and noise. This introduces an additional layer of complexity in the theoretical analysis of the method. Nevertheless, we have been able to demonstrate that graphLa+Ψ is a valid regularization method, and we have rigorously established both its convergence and stability properties.

While the first part of this chapter will be devoted to the theoretical analysis of the method, in the latest part we present selected numerical experiments in 2D computerized tomography, wherein we integrate the graphLa+Ψ method with various reconstruction techniques Ψ, including Filter Back Projection (graphLa+FBP), general Tikhonov (graphLa+Tik), Total Variation (graphLa+TV), and a trained deep neural network (graphLa+Net). The graphLa+Ψ approach significantly enhances the quality of the approximated solutions for each method Ψ. Notably, graphLa+Net is outperforming, offering a robust and stable application of deep neural networks in solving inverse problems.

2.1 The model setting

Since we are dealing with inverse problems, the model problem we considered is the one already introduced in (1.30) that is the linear system of equation

$$A\mathbf{x} = \mathbf{b}^\delta, \tag{2.1}$$

where $A\colon X \simeq \mathbb{R}^n \to Y \simeq \mathbb{R}^m$ reprensents a discretized version of a linear operator that is inherently ill–posed. Given a fixed $\mathbf{x}_{\text{gt}} \in \mathbb{R}^n$ and let $\mathbf{b} \coloneqq A\mathbf{x}_{\text{gt}}$, we want to recover a good approximation of the ground-truth $\mathbf{x}_{\text{gt}}$ from a noisy observation $\mathbf{b}^\delta$ of $\mathbf{b}$ such that

$$\mathbf{b}^\delta \coloneqq \mathbf{b} + \boldsymbol{\eta}_\delta, \qquad \|\boldsymbol{\eta}_\delta\| \leq \delta,$$

where $\boldsymbol{\eta}_\delta$ is a random perturbation and $\delta > 0$ is the noise intensity. To solve our model equation some regularization strategy is needed and therefore we consider the standard variational problem

$$\mathbf{x}_\lambda^\delta \in \underset{\mathbf{x}\in X}{\arg\min} \left\{ \frac{1}{2}\|A\mathbf{x} - \mathbf{b}^\delta\|_2^2 + \lambda\|L\mathbf{x}\|_1 \right\}, \tag{2.2}$$

where L is a linear mapping, characterized by the property that

$$\ker(L) \cap \ker(A) = \{\mathbf{0}\}, \tag{2.3}$$

as described in Section §1.4.1. Differently from the previous chapter, in the following, we will consider the ℓ^1–norm in the regularization term instead of the standard ℓ^2–norm. As a regularization operator we would like to consider the distance–based graph Laplacian operator introduced in Section §1.5 as done in [20, 29, 19]. Generally speaking, embedding a graph-based operator in the regularization term acts as a guiding mechanism for the overall regularization process. This operator helps in identifying the "correct" neighborhood to concentrate the reconstruction efforts on, by capturing specific features that can be inferred from the observed signal $\mathbf{b}^\delta$. Initially, a graph structure G is constructed from a discretized signal to incorporate features such as interfaces and discontinuities. This discrete space, which heavily depends on the signal itself, can provide more insights into the neighborhood where $\mathbf{x}_{\text{gt}}$ resides than a flat manifold like the Euclidean space. Then, by choosing a suitable graph operator L, the optimization process in equation (2.2) is oriented towards the (supposed) neighborhood of $\mathbf{x}_{\text{gt}}$.

As anticipated, the key point is to construct the graph from a signal that closely approximates the primary features of $\mathbf{x}_{\text{gt}}$. In [103], it was observed that generating the graph G directly from the observed and noisy data $\mathbf{b}^\delta$ results in poor outcomes for imaging tasks such as deblurring or tomographic reconstruction. This is because $\mathbf{b}^\delta$ exists in a different domain compared to $\mathbf{x}_{\text{gt}}$. To address this, we apply an initial preprocessing step, transforming $\mathbf{b}^\delta$ to $\Psi(\mathbf{b}^\delta)$, and subsequently constructing a graph from $\Psi(\mathbf{b}^\delta)$. This preprocessing step involves a reconstruction map $\Psi\colon Y \to X$, from the space of observations Y to the domain X where $\mathbf{x}_{\text{gt}}$ lives. This can be achieved, for example, by employing a standard Tikhonov filter (1.34) or the Filter Back Projection (FBP) method (1.29), depending on the inverse problem to handle. In this context, a very interesting choice for Ψ_Θ is represented by the class of Deep Neural Networks (DNNs). They are characterized by a vast number of trainable parameters that are optimized by minimizing a loss function over a large dataset. With the exponential growth in dedicated computational power, DNNs have achieved state-of-the-art performance across a range of applications. However, the use of DNNs in solving ill-posed

inverse problems comes with significant drawbacks, particularly concerning stability and their "black-box" nature. Firstly, DNNs are often sensitive to data perturbations and have a tendency to produce hallucinations, i.e. false yet realistic-looking artifacts, see [6, 51]. Secondly, the complexity of their internal mechanisms, which involve millions of parameters and nonlinear mappings, makes them challenging to understand or explain [136]. These issues contribute to a general skepticism about the reliability of DNNs, especially in real-world scenarios like medical imaging, where accuracy, stability, and reliability are critical.

To define the novel graphLa+Ψ method, the initial preprocessing step involves selecting a family of reconstructor maps

$$\Psi_\Theta \colon Y \to X,$$

where $\Theta = \Theta(\delta, \mathbf{b}^\delta)$ is a family of parameters that can depend on δ and $\mathbf{b}^\delta$. It is important to note that the pair (Ψ_Θ, Θ) is very general and may not be a convergent regularizing method. As detailed in the last part of Section §1.5, a grayscale image can be regarded as a function $\mathbf{x} \colon P \to \mathbb{R}^n$, where P is the set of nodes of the graph associated to $\mathbf{x} \in \mathbb{R}^n$. Therefore, let

$$\Psi_\Theta^\delta \coloneqq \Psi_\Theta(\mathbf{b}^\delta) \in X \simeq \mathcal{V}_P,$$

we can define the graph Laplacian induced by Ψ_Θ^δ and we will denote it by $\Delta_{\Psi_\Theta^\delta}$. The graphLa+Ψ method consists of computing a minimizer of (2.2) with $L = \Delta_{\Psi_\Theta^\delta}$, that is,

$$\mathbf{x}^\delta_{\Psi_\Theta^\delta,\lambda} \in \operatorname*{arg\,min}_{\mathbf{x}\in X} \left\{ \frac{1}{2}\|A\mathbf{x} - \mathbf{b}^\delta\|_2^2 + \lambda \|\Delta_{\Psi_\Theta^\delta}\mathbf{x}\|_1 \right\}. \tag{2.4}$$

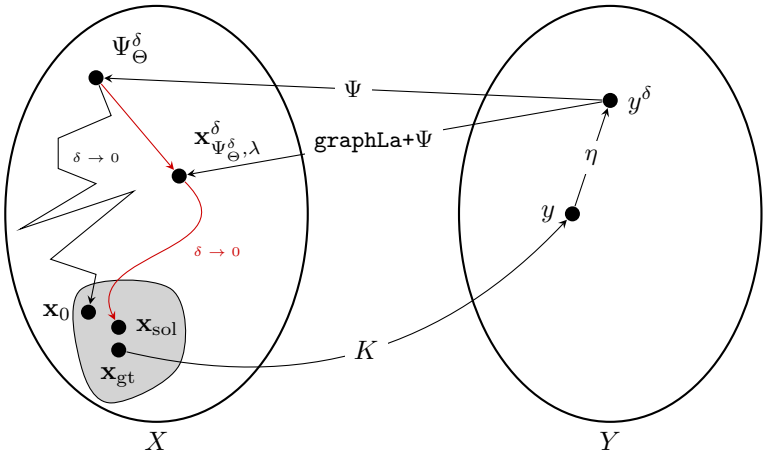


Figure 2.1: A schematic representation of the graphLa+Ψ method. The reconstructors Ψ_Θ do not necessarily need to be a regularization method, and this is represented by the piecewise linear path of Ψ_Θ^δ as δ goes to 0. However, when combined with the graph Laplacian in the Tikhonov method (2.2), it generates a convergent and stable regularization operator, that is, graphLa+Ψ, which is represented by the smooth red path.

In the next section we will demonstrate that under certain, albeit very weak, hypotheses, graphLa+Ψ is a convergent and stable regularization method in the limit as $\delta \to 0$. The key

ingredients of this analysis will be the properties of the graph Laplacian operator. Informally speaking, it helps to "chain" the reconstructors Ψ_Θ in the original and well-established regularization method (2.2). For these reasons, and due to the minimal assumptions about Ψ_Θ, it becomes feasible to select reconstructors that may not be convergent regularizing methods or those whose regularization properties lack rigorous proof yet show empirical effectiveness in certain applications, like for instance DNNs. Owing to the influence of the graph Laplacian, the overall graphLa+Ψ method maintains regularization and stability regardless. In Figure (2.1), we depict a visual representation of the method.

All the methodology and theory we develop herein apply broadly and are not limited solely to ill-posed inverse problems in imaging. However, to provide a complete analysis of the method and its performances, we will focus on 2D computerized tomography applications.

2.2 Theoretical Analysis

This part will be devoted to the theoretical analysis of the graphLa+Ψ method. As a baseline assumption, we say that the unperturbed observation $\mathbf{b} \in Y \simeq \mathbb{R}^m$ is the realization of the action of A on an element $\mathbf{x}_{\text{gt}} \in X$. That is,

Hypothesis 2.2.1. *There exists* $\mathbf{x}_{\text{gt}}$ *such that* $A\mathbf{x}_{\text{gt}} = \mathbf{b}$.

Consider now a family of operators $\{\Psi_\Theta : Y \to X\}$ which we generally refer to as *reconstructors*. We do not need to assume that Ψ_Θ is necessarily linear, and with $\Theta \in \mathbb{R}^k$ we denote all the parameters on which it depends. This family of reconstructors is at the core of our graphLa+Ψ method. It has the very important role of giving a first approximation of $\mathbf{x}_{\text{gt}}$, since upon this approximation we will construct the graph Laplacian operator. Although we would like to keep the reconstructor Ψ_Θ as general as possible, we need to enforce some (weak) regularity.

Hypothesis 2.2.2. *There exists an element* $\mathbf{x}_0 \in X$ *and a parameter choice rule* $\Theta = \Theta(\delta, \mathbf{b}^\delta)$ *such that*

$$\|\Psi_{\Theta(\delta,\mathbf{b}^\delta)}(\mathbf{b}^\delta) - \mathbf{x}_0\|_2 \to 0 \qquad \text{as } \delta \to 0.$$

The above Hypothesis 2.2.2 will guarantee that the graphLa+Ψ method is a convergent regularization method. Additionally, stability will be proved by further assuming that

Hypothesis 2.2.3. *Let* $\mathbf{b}^\delta$ *and* $\Theta := \Theta(\delta, \mathbf{b}^\delta)$ *be fixed, and* $\{\delta_k\}$ *and* $\{\mathbf{b}^{\delta_k}\}$ *be sequences such that* $\delta_k \to \delta$ *and* $\mathbf{b}^{\delta_k} \to \mathbf{b}^\delta$ *for* $k \to \infty$. *Writing* $\Theta_k := \Theta(\delta_k, \mathbf{b}^{\delta_k})$, *then*

$$\Theta_k \to \Theta \quad \text{and} \quad \|\Psi_{\Theta_k}(\mathbf{b}^{\delta_k}) - \Psi_\Theta(\mathbf{b}^\delta)\|_2 \to 0 \text{ for } k \to \infty.$$

In the next two examples, we make clear that the above assumptions are pretty weak and they can be satisfied by several large classes of reconstructors. In particular, the pair (Ψ_Θ, Θ) does not need to be a convergent regularization method.

Example 2.2.4. *A simple example of a family of reconstructors that satisfies Hypothesis 2.2.2 is the case when we identify them with a single, (locally) Lipschitz continuous operator. That is, fix* $\Theta \equiv \hat{\Theta}$ *for every* δ *and* $\mathbf{b}^\delta$*, and choose an operator* $\Psi_{\hat{\Theta}}$ *such that* $\|\Psi_{\hat{\Theta}}(\mathbf{b}_1) - \Psi_{\hat{\Theta}}(\mathbf{b}_2)\|_2 \leq L\|\mathbf{b}_1 - \mathbf{b}_2\|_2$*. Thanks to equation (2.1) and the Lipschitz condition, Hypothesis 2.2.2 is then verified with* $\mathbf{x}_0 := \Psi_{\hat{\Theta}}(\mathbf{b})$*. In the same way, Hypothesis 2.2.3 is verified again by the Lipschitz property of* $\Psi_{\hat{\Theta}}$ *and because* $\hat{\Theta}$ *is fixed and independent of* k*.*

The situation of the preceding example will occur later, when $\Psi_{\hat{\Theta}}$ is implemented as a trained DNN.

Example 2.2.5. *A less trivial family of reconstructors that satisfy Hypothesis 2.2.2 is the one given by any typical regularization operators. Let* $\Theta \in (0, \infty)$ *and* Ψ_Θ *be continuous, and* $A^\dagger$ *be the usual Moore-Penrose pseudo-inverse of* A*. Fix* $\mathbf{x}_0 := A^\dagger \mathbf{b}$ *and observe that* $A\mathbf{x}_0 = \mathbf{b}$*, since* $\mathbf{b} \in \mathrm{range}(A)$ *by Hypothesis 2.2.1. By definition of regularization operator, there exists a parameter choice rule* $\Theta = \Theta(\delta, \mathbf{b}^\delta)$ *such that*

$$\sup\{\|\Psi_{\Theta(\delta,\mathbf{b}^\delta)}(\mathbf{b}^\delta) - \mathbf{x}_0\|_2 \mid \mathbf{b}^\delta \in Y,\ \|\mathbf{b}^\delta - A\mathbf{x}_0\|_2 \leq \delta\} \to 0 \quad \text{as } \delta \to 0,$$

and Hypothesis 2.2.2 is then verified. About Hypothesis 2.2.3, this is a bit more involved and depends on the regularization method itself and the parameter choice rule. For instance, consider a standard Tikhonov reconstruction method, that is

$$\Psi_\Theta(\mathbf{b}) := \left(A^T A + \Theta I\right)^{-1} A^T \mathbf{b},$$

where $\Theta \in (0, \infty)$*. Let* $\Theta = \Theta(\delta, \mathbf{b}^\delta)$ *defined by the discrepancy principle (1.46). Then,* $\Theta_k \to \Theta = \Theta(\delta, \mathbf{b}^\delta)$ for $k \to \infty$*. Setting*

$$\mathcal{T}^k = (A^T A + \Theta_k I)^{-1} \quad \text{and} \quad \mathcal{T} = (A^T A + \Theta I)^{-1},$$

it holds that

$$\|\Psi_{\Theta_k}(\mathbf{y}^{\delta_k}) - \Psi_\Theta(\mathbf{b}^\delta)\|_2 \leq \underbrace{\|(\mathcal{T}^k - \mathcal{T})A^T \mathbf{b}^{\delta_k}\|_2}_{\mathbf{I}} + \underbrace{\|\mathcal{T}A^T(\mathbf{b}^\delta - \mathbf{b}^{\delta_k})\|_2}_{\mathbf{II}}.$$

Now,

$$\mathbf{I} \leq |\Theta - \Theta_k| \|\mathcal{T}^k\| \|\mathcal{T}A^T\| \|\mathbf{b}^{\delta_k}\|_2 \leq \frac{|\Theta - \Theta_k|}{2\Theta_k\sqrt{\Theta}} \|\mathbf{b}^{\delta_k}\|_2 \to 0,$$

and

$$\mathbf{II} \leq \|(A^T A + \Theta I)^{-1} A^T\| \|\mathbf{b}^\delta - \mathbf{b}^{\delta_k}\|_2 \leq \frac{1}{2\sqrt{\Theta}} \|\mathbf{b}^\delta - \mathbf{b}^{\delta_k}\| \to 0.$$

Before turning to the theoretical analysis of the graphLa+Ψ method, consider the regularization term $\mathcal{R}(\mathbf{x}, \mathbf{b}^\delta) := \|\Delta_{\Psi^\delta_\Theta} \mathbf{x}\|_1$ of equation (2.4). Firstly, as anticipated, unlike typical regularization methods, $\mathcal{R}$ depends not only on $\mathbf{x}$ but also on the data $\mathbf{b}^\delta$. This complicates

any attempt at studying the convergence of equation (2.4) for $\delta \to 0$.

Indicating with $w_{\Psi^\delta_\Theta}$ the edge-weight function defined in (1.50) with $\mathbf{x}$ replaced by Ψ^δ_Θ, we have that minimizing $\mathcal{R}$ means to force $\mathbf{x}^\delta_{\Psi^\delta_\Theta,\lambda}$ to be constant on the regions where $w_{\Psi^\delta_\Theta}(p,q)$ is "large". Indeed, this is a direct consequence of the graph Laplacian definition (1.48) implying

$$|\Delta_{\Psi^\delta_\Theta}\mathbf{x}(p)| = \frac{1}{\mu_{\Psi^\delta_\Theta}(p)}\left|\sum_{q\in P} w_{\Psi^\delta_\Theta}(p,q)(\mathbf{x}(p)-\mathbf{x}(q))\right|.$$

More simply, we can say that $\mathcal{R}$ helps to distinguish the regions of uniformity from the regions of interfaces. In intuition, once Ψ^δ_Θ reconstructs the region of interfaces in $\mathbf{x}_{\text{gt}}$, the final solution $\mathbf{x}^\delta_{\Psi^\delta_\Theta,\lambda}$ will be a good approximation of the ground truth $\mathbf{x}_{\text{gt}}$, even though Ψ^δ_Θ deviates from $\mathbf{x}_{\text{gt}}$ in other aspects.

Lastly, the ℓ^1-norm in $\mathcal{R}$ is introduced to enforce sparsity and preserve discontinuities on the approximated solution $\mathbf{x}^\delta_{\Psi^\delta_\Theta,\lambda}$. This is mainly in view of the imaging applications we will consider. However, all the theory developed here works with the ℓ^1-norm replaced by any ℓ^r-norm for $r > 1$.

2.2.1 Existence of solution and well–posedeness of graphLa+Ψ

In the previous section we focused on the choice of the family of operators Ψ_Θ and we introduced all the necessary assumptions that will be used to prove convergence and stability of the graphLa+Ψ method. Moreover, we also commented on the choice of the regularization term in the variational problem (2.4). Despite everything, we need a definition of solution for equation (2.1). Under Hypothesis 2.2.2, let $\mathbf{x}_0$ and $\Theta = \Theta(\delta, \mathbf{b}^\delta)$ be such that

$$\mathbf{x}_0 \coloneqq \lim_{\delta\to 0} \Psi_{\Theta(\delta,\mathbf{b}^\delta)}(\mathbf{b}^\delta). \tag{2.5}$$

Definition 2.2.6. *We call* $\mathbf{x}_{\text{sol}}$ *a* graph-minimizing solution *with respect to* $\mathbf{x}_0$*, defined in* (2.5)*, if* $A\mathbf{x}_{\text{sol}} = \mathbf{b}$ *and*

$$\|\Delta_{\mathbf{x}_0}\mathbf{x}_{\text{sol}}\|_1 = \min\{\|\Delta_{\mathbf{x}_0}\mathbf{x}\|_1 \mid \mathbf{x}\in X,\ A\mathbf{x}=\mathbf{b}\}. \tag{2.6}$$

Remark 2.2.7. *A graph-minimizing solution* $\mathbf{x}_{\text{sol}}$ *is a pre-image of* $\mathbf{b}$ *which minimizes the functional* $\mathcal{R}(\mathbf{x}) = \|\Delta_{\mathbf{x}_0}\mathbf{x}\|_1$*. If the operator* K *is injective, then there exists one and only one graph-minimizing solution and* $\mathbf{x}_{\text{sol}} = \mathbf{x}_{\text{gt}}$*, thanks to Hypothesis 2.2.1. However, in general,* $\mathbf{x}_{\text{gt}}$ *is not necessarily a graph-minimizing solution when* K *is not injective. Loosely speaking, a graph-minimizing solution is an approximation of the (inaccessible) ground-truth with respect to some a-posteriori information encoded in* $\mathbf{x}_0$*, as per equation* (2.5)*.*

Let us indicate with $w_{\Psi^\delta_\Theta}$ and w_0 the edge-weight functions in equation (1.50) induced by Ψ^δ_Θ and $\mathbf{x}_0$, respectively. To make our analysis work, we need three last hypotheses which are related to properties of $w_{\Psi^\delta_\Theta}$ and $\Delta_{\Psi^\delta_\Theta}$.

Hypothesis 2.2.8. *For every $p, q \in P$, $w_{\Psi^\delta_\Theta}(p,q) > 0$ if and only if $w_0(p,q) > 0$.*

The following lemma is an immediate consequence of the above hypothesis.

Lemma 2.2.9. *Under Hypothesis 2.2.8, there is an invariant subspace $V \subseteq \mathcal{V}_P$ such that $\ker(\Delta_{\Psi^\delta_\Theta}) = \ker(\Delta_{\mathbf{x}_0}) = V$ for every Ψ^δ_Θ.*

Proof. The null space of a generic graph Laplacian, as per equation (1.48), is given by the subspace of functions which are constant on the connected components of the node set P, see for example [90, Lemmas 0.29 and 0.31]. By Hypothesis 2.2.8, it is easy to check that a sequence $\{p_i\}_{i=0}^k$ is a walk with respect to $w_{\Psi^\delta_\Theta}$ if and only if it is a walk with respect to w_0. Therefore, all the connected components of P, identified by $w_{\Psi^\delta_\Theta}$ and w_0, are invariant. This concludes the proof. □

The invariant subspace V replaces $\ker(L)$ in the typical null-space condition (2.3), invoked for functionals of the form (2.2).

Hypothesis 2.2.10. $\ker(A) \cap V = \{\mathbf{0}\}$.

The last assumption we need is on the intensity function h_i in equation (1.50) and the node measure $\mu_\mathbf{x}$ in equation (1.51).

Hypothesis 2.2.11. *For every fixed $p \in P$, the function h_i and the map $\mathbf{x} \mapsto \mu_\mathbf{x}(p)$ are Lipschitz continuous.*

The above hypotheses are not difficult to check in practice. In the numerical experiments we will provide specific choices of equations (1.50) and (1.51), and we will show that all the previous assumptions are satisfied.

In the following we will always assume that all the previous assumption hold true. In order to prove the existence and uniqueness of the graph-minimizing solution and the well-posedness of the variational problem (2.4), we define the functional

$$\Gamma(\mathbf{x}) \coloneqq \frac{1}{2}\|A\mathbf{x} - \mathbf{b}^\delta\|_2^2 + \lambda\|\Delta_{\Psi^\delta_\Theta}\mathbf{x}\|_1. \tag{2.7}$$

Recall that a (nonnegative) functional $\Gamma\colon X \to [0,\infty)$ is *coercive* if $\Gamma(\mathbf{x}) \to \infty$ for $\|\mathbf{x}\| \to \infty$, where $\|\cdot\|$ can be any norm on $X \simeq \mathbb{R}^n$. Thanks to Hypothesis 2.2.10, we can prove the following

Lemma 2.2.12. *Γ is coercive for every fixed $\lambda > 0$ and $\delta \geq 0$.*

Proof. Let V be the invariant $\ker(\Delta_{\Psi^\delta_\Theta})$ from Lemma 2.2.9, and let us indicate with π and $\pi_\perp$ the projection into V and $V^\perp$, respectively. In general, it holds that

$$\inf_{\substack{\mathbf{u}\in V^\perp \\ \mathbf{u}\neq\mathbf{0}}} \frac{\|\Delta_{\Psi^\delta_\Theta}\mathbf{u}\|_1}{\|\mathbf{u}\|_1} \geq \gamma_1 > 0. \tag{2.8}$$

Since $\ker(A) \cap V = \{\mathbf{0}\}$ by Hypothesis 2.2.10, then it also holds that

$$\inf_{\substack{\mathbf{v}\in V\\ \mathbf{v}\neq\mathbf{0}}} \frac{\|A\mathbf{v}\|_2}{\|\mathbf{v}\|_2} \geq \gamma_2 > 0. \tag{2.9}$$

Fix a sequence $\{\mathbf{x}_j\}$ such that $\|\mathbf{x}_j\|_2 \to \infty$. We want to show that $\Gamma(\mathbf{x}_j) \to \infty$. Clearly, for $j \to \infty$

$$\|\mathbf{x}_j\|_2 \to \infty \quad \text{if and only if} \quad \|\pi\mathbf{x}_j\|_2^2 + \|\pi_\perp \mathbf{x}_j\|_1 \to \infty.$$

There are two cases:

(i) $\lim_j \|\pi_\perp \mathbf{x}_j\|_1 = \infty$;

(ii) $\liminf_j \|\pi_\perp \mathbf{x}_j\|_1 \leq c < \infty$ and $\lim_j \|\pi\mathbf{x}_j\|_2^2 = \infty$.

If we are in (i), then by equation (2.8) we have that

$$\begin{aligned}\lambda\gamma_1\|\pi_\perp\mathbf{x}_j\|_1 &\leq \lambda\|\Delta_{\Psi_\Theta^\delta}\pi_\perp\mathbf{x}_j\|_1 \\ &\leq \frac{1}{2}\|A\mathbf{x}_j - \mathbf{b}^\delta\|_2^2 + \lambda\|\Delta_{\Psi_\Theta^\delta}\mathbf{x}_j\|_1 = \Gamma(\mathbf{x}_j).\end{aligned} \tag{2.10}$$

On the other hand, if we are in the (ii) case, then by equation (2.9) we obtain

$$\begin{aligned}\frac{\gamma_2}{4}\|\pi\mathbf{x}_j\|_2^2 \leq \frac{1}{4}\|A\pi\mathbf{x}_j\|_2^2 &\leq \frac{1}{2}\|A\mathbf{x}_j - \mathbf{b}^\delta\|_2^2 + \lambda\|\Delta_{\Psi_\Theta^\delta}\mathbf{x}_j\|_1 + \frac{1}{2}\|A\pi_\perp\mathbf{x}_j - \mathbf{b}^\delta\|_2^2 \\ &= \Gamma(\mathbf{x}_j) + \frac{1}{2}\|A\pi_\perp\mathbf{x}_j - \mathbf{b}^\delta\|_2^2.\end{aligned} \tag{2.11}$$

Moreover, from (ii) it also follows that $\liminf_j \|A\pi_\perp\mathbf{x}_j - \mathbf{b}^\delta\|_2^2$ is bounded. Passing to the $\liminf$ in both equations (2.10) and (2.11) conclude the proof. □

Remark 2.2.13. *The result in Lemma 2.2.12 still holds if we replace Ψ_Θ^δ with $\mathbf{x}_0$ in Γ. The proof remains unchanged in this case.*

Proposition 2.2.14. *There exists a graph-minimizing solution $\mathbf{x}_{\text{sol}}$.*

Proof. Let

$$c \coloneqq \inf\{\|\Delta_{\mathbf{x}_0}\mathbf{x}\|_1 \mid \mathbf{x} \in X,\ A\mathbf{x} = \mathbf{b}\},$$

which is well-defined thanks to Hypothesis 2.2.1. Therefore there exists a sequence $\{\mathbf{x}_j\}$ such that $A\mathbf{x}_j = \mathbf{b}$ for every j and $\lim_j \|\Delta_{\mathbf{x}_0}\mathbf{x}_j\|_1 = c$. In particular, there exists $c_1 > 0$ such that

$$\|\Delta_{\mathbf{x}_0}\mathbf{x}_j\|_1 \leq c_1 \quad \text{for every } j. \tag{2.12}$$

There are two possible cases:

(i) $\|\mathbf{x}_j\|_2 \leq c_2$ for some $c_2 > 0$, for every j,

(ii) there exists a subsequence $\{\mathbf{x}_{j'}\}$, such that $\lim_{j'} \|\mathbf{x}_{j'}\|_2 = \infty$.

If we are in case (i), then by compactness and continuity arguments we can conclude that there exists $\mathbf{x}_{\text{sol}}$ such that

$$\lim_{j'} \mathbf{x}_{j'} = \mathbf{x}_{\text{sol}} \quad \text{and} \quad \begin{cases} A\mathbf{x}_{\text{sol}} = \mathbf{b}, \\ \|\Delta_{\mathbf{x}_0}\mathbf{x}_{\text{sol}}\|_1 = c, \end{cases}$$

that is, $\mathbf{x}_{\text{sol}}$ is a graph-minimizing solution with respect to $\mathbf{x}_0$.

Suppose now we are in case (ii). By Lemma 2.2.12 and Remark 2.2.13, $\Gamma(\mathbf{x}_{j'}) \to \infty$ for any fixed λ, δ. Since $\|A\mathbf{x}_{j'} - \mathbf{b}^\delta\|_2^2 = \|\mathbf{b} - \mathbf{b}^\delta\|_2^2 \leq \delta^2$ for every j', it follows necessarily that $\lim_{j'} \|\Delta_{\mathbf{x}_0}\mathbf{x}_{j'}\|_1 = \infty$. This leads to an absurdity in light of (2.12). □

Proposition 2.2.15. *For every fixed $\delta, \lambda > 0$ and $\mathbf{b}^\delta \in Y$, there exists a solution $\mathbf{x}^\delta_{\Psi^\delta_\Theta,\lambda}$ for the variational problem (2.4).*

Proof. From Lemma 2.2.12, the nonnegative functional Γ is coercive on a finite dimensional vector space. By standard theory, there exists a minimizer, see for example [10, Proposition 11.15]. □

Corollary 2.2.16. *If A is injective, then $\mathbf{x}_{\text{sol}}$ and $\mathbf{x}^\delta_{\Psi^\delta_\Theta,\lambda}$ are unique.*

Proof. The uniqueness of $\mathbf{x}_{\text{sol}}$ is straightforward. On the other hand, the uniqueness of $\mathbf{x}^\delta_{\Psi^\delta_\Theta,\lambda}$ is derived from the fact that if A is injective then the functional $\mathbf{x} \mapsto \|A\mathbf{x} - \mathbf{b}^\delta\|_2^2$ is strongly convex. This property leads to the strong (and therefore strict) convexity of Γ. According to [10, Corollary 11.9], the desired result follows. □

Remark 2.2.17. *Without the injectivity property, uniqueness can fail. The main culprit is the ℓ^1-norm in the regularization term. However, it is possible to achieve uniqueness in a less stringent manner, namely, for every $\mathbf{b}^\delta$ outside a set of negligible measures. The approach should be in line with [42, 4], but adapted to this specific context. That being said, relaxing the assumptions to regain the uniqueness of the solution falls beyond the scope of the current work.*

2.2.2 Convergence and Stability analysis

We are almost ready to prove convergence and stability of the graphLa+Ψ method. As we will see, the main difficulty is given by the regularization term $\mathcal{R}(\mathbf{x}, \mathbf{b}^\delta)$, which depends on the observed data $\mathbf{b}^\delta$. Therefore, all standard techniques can not be applied straightforwardly. A crucial role will be played by the following two lemmas, which guarantee uniform convergence of $\Delta_{\Psi^\delta_\Theta}$, and a special uniform coercivity property for the Γ functional of equation (2.7).

Lemma 2.2.18. *Let $\Theta = \Theta(\delta, \mathbf{b}^\delta)$ and $\mathbf{x}_0$ be defined as in Hypothesis 2.2.2. For every $\mathbf{x} \in X$ it holds that*

$$\|\Delta_{\Psi^\delta_\Theta}\mathbf{x} - \Delta_{\mathbf{x}_0}\mathbf{x}\|_1 \leq c\|\mathbf{x}\|_1\|\Psi^\delta_\Theta - \mathbf{x}_0\|_2 \to 0 \quad \text{as } \delta \to 0,$$

where c is a positive constant independent of $\mathbf{x}$.

Proof. Indicating with k^δ_{pq} the elements of the matrix $K^\delta \coloneqq \Delta_{\Psi^\delta_\Theta} - \Delta_{\mathbf{x}_0}$, then

$$\|\Delta_{\Psi^\delta_\Theta}\mathbf{x} - \Delta_{\mathbf{x}_0}\mathbf{x}\|_1 \leq \|K^\delta\|\|\mathbf{x}\|_1 = \left(\max_{q\in P}\sum_{p\in P}|k^\delta_{pq}|\right)\|\mathbf{x}\|_1, \tag{2.13}$$

where $\|K^\delta\|$ is the induced matrix 1-norm. Making explicit now the values of k^δ_{pq}, we have

$$\begin{aligned}
\sum_{p\in P}|k^\delta_{pq}| &= |k^\delta_{qq}| + \sum_{\substack{p\in P\\ p\neq q}}|k^\delta_{pq}| \\
&= \left|\sum_{\substack{\ell\in P\\ \ell\neq q}}\frac{w_{\Psi^\delta_\Theta}(q,\ell)}{\mu_{\Psi^\delta_\Theta}(q)} - \frac{w_0(q,\ell)}{\mu_0(q)}\right| + \sum_{\substack{p\in P\\ p\neq q}}\left|\frac{w_{\Psi^\delta_\Theta}(p,q)}{\mu_{\Psi^\delta_\Theta}(p)} - \frac{w_0(p,q)}{\mu_0(p)}\right| \\
&\leq \sum_{\substack{p\in P\\ p\neq q}}\left|\frac{w_{\Psi^\delta_\Theta}(p,q)}{\mu_{\Psi^\delta_\Theta}(q)} - \frac{w_0(p,q)}{\mu_0(q)}\right| + \sum_{\substack{p\in P\\ p\neq q}}\left|\frac{w_{\Psi^\delta_\Theta}(p,q)}{\mu_{\Psi^\delta_\Theta}(p)} - \frac{w_0(p,q)}{\mu_0(p)}\right|,
\end{aligned} \tag{2.14}$$

where in the last inequality we used the symmetry of the edge-weight functions $w_{\Psi^\delta_\Theta}$ and w_0. To simplify the notation, define

$$t_{\delta,p,q} \coloneqq |\Psi^\delta_\Theta(p) - \Psi^\delta_\Theta(q)|, \quad t_{0,p,q} \coloneqq |\mathbf{x}_0(p) - \mathbf{x}_0(q)|.$$

Let us observe that, for every fixed triple $p, q, l \in P$,

$$\begin{aligned}
\left|\frac{w_{\Psi^\delta_\Theta}(p,q)}{\mu_{\Psi^\delta_\Theta}(l)} - \frac{w_0(p,q)}{\mu_0(l)}\right| &= w_{\mathrm{d}}(p,q)\left|\frac{h_{\mathrm{i}}(t_{\delta,p,q})}{\mu_{\Psi^\delta_\Theta}(l)} - \frac{h_{\mathrm{i}}(t_{0,p,q})}{\mu_0(l)}\right| \\
&= \frac{w_{\mathrm{d}}(p,q)}{\mu_{\Psi^\delta_\Theta}(l)\mu_0(l)}\left|\mu_0(l)h_{\mathrm{i}}(t_{\delta,p,q}) - \mu_{\Psi^\delta_\Theta}(l)h_{\mathrm{i}}(t_{0,p,q})\right|.
\end{aligned} \tag{2.15}$$

Let us recall now that, by Hypothesis 2.2.11, we have

$$|h_{\mathrm{i}}(t_{\delta,p,q}) - h_{\mathrm{i}}(t_{0,p,q})| \leq L'|t_{\delta,p,q} - t_{0,p,q}|, \quad |\mu_{\Psi^\delta_\Theta}(l) - \mu_0(l)| \leq L''_l\|\Psi^\delta_\Theta - \mathbf{x}_0\|_2. \tag{2.16}$$

By adding and subtracting the auxiliary term $\mu_0(l)h_{\mathrm{i}}(t_{0,p,q})$, it holds that

$$\begin{aligned}\left|\mu_0(l)h_{\mathrm{i}}(t_{\delta,p,q}) - \mu_{\Psi^\delta_\Theta}(l)h_{\mathrm{i}}(t_{0,p,q})\right| &\leq \mu_0(l)|h_{\mathrm{i}}(t_{\delta,p,q}) - h_{\mathrm{i}}(t_{0,p,q})| + |\mu_{\Psi^\delta_\Theta}(l) - \mu_0(l)|h_{\mathrm{i}}(t_{0,p,q}) \\ &\leq L' \max_l\{\mu_0(l)\}|t_{\delta,p,q} - t_{0,p,q}| + \max_l\{L''_l\}\max_{p,q}\{h_{\mathrm{i}}(t_{0,p,q})\}\|\Psi^\delta_\Theta - \mathbf{x}_0\|_2.\end{aligned} \tag{2.17}$$

Let us bound now $|t_{\delta,p,q} - t_{0,p,q}|$:

$$\begin{aligned}|t_{\delta,p,q} - t_{0,p,q}| &= \big||\Psi^\delta_\Theta(p) - \Psi^\delta_\Theta(q)| - |\mathbf{x}_0(p) - \mathbf{x}_0(q)|\big| \\ &\leq \big|\big(\Psi^\delta_\Theta(p) - \mathbf{x}_0(p)\big) + \big(\mathbf{x}_0(q) - \Psi^\delta_\Theta(q)\big)\big| \\ &\leq \big|\Psi^\delta_\Theta(p) - \mathbf{x}_0(p)\big| + \big|\Psi^\delta_\Theta(q) - \mathbf{x}_0(q)\big| \\ &\leq 2\|\Psi^\delta_\Theta - \mathbf{x}_0\|_2.\end{aligned} \tag{2.18}$$

Therefore, denoting with

$$\begin{aligned}\underline{\mu}_\delta &\coloneqq \min_l\{\mu_{\Psi^\delta_\Theta}(l)\}, \quad \underline{\mu}_0 \coloneqq \min_l\{\mu_0(l)\}, \\ \overline{\mu}_0 \coloneqq \max_l\{\mu_0(l)\}, \quad \overline{L''} &\coloneqq \max_k\{L''_l\}, \quad \overline{h}_{\mathrm{i}} \coloneqq \max_{p,q}\{h_{\mathrm{i}}(t_{0,p,q})\},\end{aligned}$$

and using equations (2.17) and (2.18) into equation (2.15), we have

$$\left|\frac{w_{\Psi^\delta_\Theta}(p,q)}{\mu_{\Psi^\delta_\Theta}(l)} - \frac{w_0(p,q)g(\delta,\mathbf{x}_0)}{\mu_0(l)}\right| \leq w_{\mathrm{d}}(p,q)\frac{2L'\overline{\mu}_0 + \overline{L''}\,\overline{h}_{\mathrm{i}}}{\underline{\mu}_\delta\,\underline{\mu}_0}\|\Psi^\delta_\Theta - \mathbf{x}_0\|_2. \tag{2.19}$$

Now, define

$$\overline{d} \coloneqq \max_{p,q}\{w_{\mathrm{d}}(p,q)\}, \qquad \overline{\kappa} \coloneqq \max_q\left\{\mathrm{card}\{p \in P \mid w_{\mathrm{d}}(p,q) \neq 0\}\right\}.$$

Then, combining equations (2.14) and (2.19), for every fixed q it holds that

$$\begin{aligned}\sum_{p\in P}|k^\delta_{pq}| &\leq \frac{2\left(2L'\overline{\mu}_0 + \overline{L''}\,\overline{h}_{\mathrm{i}}\right)}{\underline{\mu}_\delta\,\underline{\mu}_0}\|\Psi^\delta_\Theta - \mathbf{x}_0\|_2\sum_{p\in P}w_{\mathrm{d}}(p,q) \\ &\leq \frac{2\overline{d}\overline{\kappa}\left(2L'\overline{\mu}_0 + \overline{L''}\,\overline{h}_{\mathrm{i}}\right)}{\underline{\mu}_\delta\,\underline{\mu}_0}\|\Psi^\delta_\Theta - \mathbf{x}_0\|_2.\end{aligned} \tag{2.20}$$

Finally, from equation (2.16) and Hypothesis 2.2.2, there exists δ_0 such that

$$\underline{\mu}_\delta \geq \frac{1}{2}\underline{\mu}_0 \quad \text{for every} \quad 0 \leq \delta \leq \delta_0,$$

and by defining

$$c \coloneqq \frac{4\overline{d}\overline{\kappa}\left(2L'\overline{\mu}_0 + \overline{L''}\,\overline{h}_{\mathrm{i}}\right)}{\underline{\mu}_0^2},$$

the thesis follows from (2.13) and (2.20). □

Remark 2.2.19. *The constant c may depend on the dimension n. However, by selecting appropriate values for w_{d}, h_{i} and $\mu_{\mathbf{x}}$, which vary based on specific applications, it is possible to make c independent of the dimension n of the vector space $\mathcal{V}_P \simeq X$. For example, if we fix the edge-weight function w_{d} independently of n and set $\mu_{\mathbf{x}} \equiv \mu$ with $\mu > 0$ as a positive constant for every $\mathbf{x}$, then $\overline{d\kappa}$ becomes independent of n and $\overline{L''} = 0$, thereby making c independent of n as well. In the numerical experiments the chosen edge–weight function and node measure ensure that c is independent of n.*

Lemma 2.2.20. *Let $\Theta = \Theta(\delta, \mathbf{b}^\delta)$ be the parameter choice rule as in Hypothesis 2.2.2 and let $\delta_k \to 0$. Write $\Theta_k := \Theta(\delta_k, \mathbf{b}^{\delta_k})$ and fix $\lambda > 0$. For any sequence $\{\mathbf{x}_k\}$ such that $\limsup_k \|\mathbf{x}_k\|_2 = \infty$, then*

$$\limsup_k \frac{1}{2}\|A\mathbf{x}_k - \mathbf{b}\|_2^2 + \lambda \|\Delta_{\Psi^{\delta_k}_{\Theta_k}} \mathbf{x}_k\|_1 = \infty.$$

Proof. Let V be the invariant null space from Lemma 2.2.9. Define

$$\gamma_k := \inf_{\substack{\mathbf{u}\in V^\perp \\ \|\mathbf{u}\|_1=1}} \|\Delta_{\Psi^{\delta_k}_{\Theta_k}} \mathbf{u}\|_1 > 0 \quad \text{and} \quad \gamma_0 := \inf_{\substack{\mathbf{u}\in V^\perp \\ \|\mathbf{u}\|_1=1}} \|\Delta_{\mathbf{x}_0}\mathbf{u}\|_1 > 0.$$

By Lemma 2.2.18, it holds that

$$\forall\, \mathbf{u} \in V^\perp \text{ s.t. } \|\mathbf{u}\|_1 = 1, \quad \|\Delta_{\mathbf{x}_0}\mathbf{u}\|_1 - \frac{\gamma_0}{2} \le \|\Delta_{\Psi^{\delta_k}_{\Theta_k}} \mathbf{u}\|_1 \qquad \text{for } k \ge N = N(\gamma_0).$$

Therefore, it follows that

$$\gamma_k \ge \frac{\gamma_0}{2} \qquad \forall k \ge N(\gamma_0).$$

In particular, there exists $\hat{\gamma} > 0$ such that

$$\|\Delta_{\Psi^{\delta_k}_{\Theta_k}} \mathbf{x}\|_1 \ge \hat{\gamma}\|\pi_\perp \mathbf{x}\|_1 \qquad \forall\, \mathbf{x},\ \forall\, k.$$

The rest of the proof follows like in Lemma 2.2.12. □

The next theorem presents a convergence result for the graphLa+Ψ method (2.4). The overall proof follows a fairly standard approach even though, since the regularizing term in equation (2.4) depends on the data $\mathbf{b}^\delta$ as well, it involves a few nontrivial technical aspect. We will use a slight modification of the notation introduced in equation (2.7). Specifically, we define

$$\Gamma_k(\mathbf{x}) := \frac{1}{2}\|A\mathbf{x} - \mathbf{b}^{\delta_k}\|_2^2 + \lambda_k \|\Delta_{\Psi^{\delta_k}_{\Theta_k}} \mathbf{x}\|_1.$$

Theorem 2.2.21 (Convergence)**.** *Assume that $\lambda\colon (0, +\infty) \to (0, +\infty)$ satisfies*

$$\lim_{\delta\to 0} \lambda(\delta) = 0, \tag{2.21a}$$

$$\lim_{\delta\to 0} \frac{\delta^2}{\lambda(\delta)} = 0. \tag{2.21b}$$

Fix a sequence $\{\delta_k\}$ *such that*

$$\lim_{k\to\infty} \delta_k = 0, \qquad \|\mathbf{b}^{\delta_k} - \mathbf{b}\|_2 \leq \delta_k,$$

and set $\lambda_k \coloneqq \lambda(\delta_k)$. *Let* $\Theta = \Theta(\delta, \mathbf{b}^\delta)$ *be the parameter choice rule as in Hypothesis 2.2.2, and set* $\Theta_k \coloneqq \Theta(\delta_k, \mathbf{b}^{\delta_k})$. *Then every sequence* $\{\mathbf{x}_k\}$ *of elements that minimize the functional (2.4), with* δ_k *and* Θ_k, *has a convergent subsequence. The limit* $\mathbf{x}_{\text{sol}}$ *of the convergent subsequence* $\{\mathbf{x}_{k'}\}$*is a graph-minimizing solution with respect to* $\mathbf{x}_0$, *and*

$$\lim_{k'} \|\Delta_{\Psi_{\Theta_{k'}}^{\delta_{k'}}} \mathbf{x}_{k'}\|_1 = \|\Delta_{\mathbf{x}_0}\mathbf{x}_{\text{sol}}\|_1.$$

If $\mathbf{x}_{\text{sol}}$ *is unique, then* $\mathbf{x}_k \to \mathbf{x}_{\text{sol}}$.

Proof. The sequence $\{\mathbf{x}_k\}$ is well-posed thanks to Proposition 2.2.15. Fix a graph-minimizing solution $\mathbf{x}_{\text{sol}}$ as in Definition 2.2.6, which exists because of Proposition 2.2.14. Then, by definition, it holds that

$$\Gamma_k(\mathbf{x}_k) \leq \Gamma_k(\mathbf{x}_{\text{sol}}) \leq \frac{\delta_k^2}{2} + \lambda_k\|\Delta_{\mathbf{x}_0}\mathbf{x}_{\text{sol}}\|_1 + \lambda_k c\|\mathbf{x}_{\text{sol}}\|_1\|\Psi_{\Theta_k}^{\delta_k} - \mathbf{x}_0\|_2 \to 0 \tag{2.22}$$

as $k \to \infty$, where the last inequality comes from Lemma 2.2.18, and the convergence to zero is granted by equation (2.21a). Therefore, $\|A\mathbf{x}_k - \mathbf{b}^{\delta_k}\|_2^2 \to 0$, and in particular

$$\|A\mathbf{x}_k - \mathbf{b}\|_2 \leq \|A\mathbf{x}_k - \mathbf{b}^{\delta_k}\|_2 + \delta_k \to 0 \quad \text{as } k \to \infty. \tag{2.23}$$

Since

$$\lambda_k\|\Delta_{\Psi_{\Theta_k}^{\delta_k}} \mathbf{x}_k\|_1 \leq \Gamma_k(\mathbf{x}_k),$$

then by equations (2.21b) and (2.22), we get

$$\limsup_k \|\Delta_{\Psi_{\Theta_k}^{\delta_k}} \mathbf{x}_k\|_1 \leq \|\Delta_{\mathbf{x}_0}\mathbf{x}_{\text{sol}}\|_1. \tag{2.24}$$

Let $\lambda^+ \coloneqq \max\{\lambda_k \mid k \in \mathbb{N}\}$. Then, combining equations (2.23) and (2.24), it holds

$$\frac{1}{2}\|A\mathbf{x}_k - \mathbf{b}\|_2^2 + \lambda^+\|\Delta_{\Psi_{\Theta_k}^{\delta_k}} \mathbf{x}_k\|_1 \leq c < \infty,$$

and from Lemma 2.2.20 we deduce that $\{\mathbf{x}_k\}$ is bounded. Therefore, there exists a convergent subsequence $\{\mathbf{x}_{k'}\}$ which converges to a point $\mathbf{x}^*$. The limit point $\mathbf{x}^*$ satisfies $A\mathbf{x}^* = \mathbf{b}$ thanks to equation (2.23). Moreover, by the boundedness of $\{\mathbf{x}_{k'}\}$ and the uniform convergence granted by Lemma 2.2.18, we infer that

$$\|\Delta_{\mathbf{x}_0}\mathbf{x}^*\|_1 = \lim_{k'} \|\Delta_{\Psi_{\Theta_{k'}}^{\delta_{k'}}} \mathbf{x}_{k'}\|_1.$$

Applying equation (2.24), we finally get

$$\|\Delta_{\mathbf{x}_0}\mathbf{x}^*\|_1 = \lim_{k'} \|\Delta_{\Psi^{\delta_{k'}}_{\Theta_{k'}}}\mathbf{x}_{k'}\|_1 \leq \|\Delta_{\mathbf{x}_0}\mathbf{x}_{\text{sol}}\|_1 \leq \|\Delta_{\mathbf{x}_0}\mathbf{x}^*\|_1.$$

That is, $\mathbf{x}^*$ is a graph-minimizing solution. If the graph-minimization solution is unique, then we have just proven that every subsequence of $\{\mathbf{x}_k\}$ has a subsequence converging to $\mathbf{x}^*$, and therefore $\mathbf{x}_k \to \mathbf{x}_{\text{sol}}$ by a standard topological argument. □

For the proof of the stability result we need a couple of preliminary lemmas.

Lemma 2.2.22. *For all $\mathbf{x} \in X$ and $\mathbf{b}^{\delta_{k_1}}, \mathbf{b}^{\delta_{k_2}} \in Y$, we have*

$$\Gamma_{k_1}(\mathbf{x}) \leq 2\Gamma_{k_2}(\mathbf{x}) + \left\|\mathbf{b}^{\delta_{k_1}} - \mathbf{b}^{\delta_{k_2}}\right\|_2^2 + \left\|(\Delta_{\Psi^{\delta_{k_1}}_{\Theta_{k_1}}} - \Delta_{\Psi^{\delta_{k_2}}_{\Theta_{k_2}}})\mathbf{x}\right\|_1$$

Proof. By standard p-norm inequalities, it holds

$$\begin{aligned}
\Gamma_{k_1}(\mathbf{x}) &= \frac{1}{2}\|A\mathbf{x} - \mathbf{b}^{\delta_{k_1}}\|_2^2 + \lambda\|\Delta_{\Psi^{\delta_{k_1}}_{\Theta_{k_1}}}\mathbf{x}\|_1 \\
&\leq \|A\mathbf{x} - \mathbf{b}^{\delta_{k_2}}\|_2^2 + \|\mathbf{b}^{\delta_{k_1}} - \mathbf{b}^{\delta_{k_2}}\|_2^2 + \lambda\|\Delta_{\Psi^{\delta_{k_1}}_{\Theta_{k_1}}}\mathbf{x}\|_1 \\
&\leq 2\Gamma_{k_2}(\mathbf{x}) + \|\mathbf{b}^{\delta_{k_1}} - \mathbf{b}^{\delta_{k_2}}\|_2^2 + \|(\Delta_{\Psi^{\delta_{k_1}}_{\Theta_{k_1}}} - \Delta_{\Psi^{\delta_{k_2}}_{\Theta_{k_2}}})\mathbf{x}\|_1.
\end{aligned}$$

□

Lemma 2.2.23. *Let δ_k, Θ_k, and Θ defined as in Hypothesis 2.2.3. It holds that*

$$\|\Delta_{\Psi^{\delta_k}_{\Theta_k}}\mathbf{x} - \Delta_{\Psi^{\delta}_{\Theta}}\mathbf{x}\|_1 \leq c\|\mathbf{x}\|_1\|\Psi^{\delta_k}_{\Theta_k} - \Psi^{\delta}_{\Theta}\|_2 \to 0 \quad \text{as } k \to \infty,$$

where c is a positive constant independent of $\mathbf{x}$.

Proof. The proof is similar to Lemma 2.2.18 using Hypothesis 2.2.3. □

Theorem 2.2.24 (Stability)**.** *Let now $\mathbf{b}^{\delta}$ be fixed and $\{\delta_k\}$ and $\{\mathbf{b}^{\delta_k}\}$ be sequences such that $\delta_k \to \delta$ and $\mathbf{b}^{\delta_k} \to \mathbf{b}^{\delta}$ for $k \to \infty$. Then every sequence $\{\mathbf{x}_k\}$ with*

$$\mathbf{x}_k \in \arg\min_{\mathbf{x}\in X}\left\{\frac{1}{2}\|A\mathbf{x} - \mathbf{b}^{\delta_k}\|_2^2 + \lambda\|\Delta_{\Psi^{\delta_k}_{\Theta_k}}\mathbf{x}\|_1\right\},$$

has a converging subsequence $\{\mathbf{x}_{k'}\}$ such that

$$\lim_{k'}\mathbf{x}_{k'} \in \arg\min_{\mathbf{x}\in X}\left\{\frac{1}{2}\|A\mathbf{x} - \mathbf{b}^{\delta}\|_2^2 + \lambda\|\Delta_{\Psi^{\delta}_{\Theta}}\mathbf{x}\|_1\right\}.$$

Proof. Because $\mathbf{x}_k$ is a minimizer of Γ_k, we have

$$\Gamma_k(\mathbf{x}_k) \leq \Gamma_k(\mathbf{x}), \quad \forall\, \mathbf{x} \in X. \tag{2.25}$$

Choose now a vector $\bar{\mathbf{x}} \in X$. By applying the previous equation to $\mathbf{x} = \bar{\mathbf{x}}$ and using twice Lemma 2.2.22, it follows that

$$\begin{aligned}
\Gamma(\mathbf{x}_k) &\leq 2\Gamma_k(\mathbf{x}_k) + \|\mathbf{b}^\delta - \mathbf{b}^{\delta_k}\|_2^2 + \|(\Delta_{\Psi^\delta_\Theta} - \Delta_{\Psi^{\delta_k}_{\Theta_k}})\mathbf{x}_k\|_1 \\
&\leq 2\Gamma_k(\bar{\mathbf{x}}) + \|\mathbf{b}^\delta - \mathbf{b}^{\delta_k}\|_2^2 + \|(\Delta_{\Psi^\delta_\Theta} - \Delta_{\Psi^{\delta_k}_{\Theta_k}})\mathbf{x}_k\|_1 \\
&\leq 4\Gamma(\bar{\mathbf{x}}) + 3\|\mathbf{b}^\delta - \mathbf{b}^{\delta_k}\|_2^2 + 2\|(\Delta_{\Psi^\delta_\Theta} - \Delta_{\Psi^{\delta_k}_{\Theta_k}})\bar{\mathbf{x}}\|_1 + \|(\Delta_{\Psi^\delta_\Theta} - \Delta_{\Psi^{\delta_k}_{\Theta_k}})\mathbf{x}_k\|_1 \\
&\leq 4\Gamma(\bar{\mathbf{x}}) + 3\|\mathbf{b}^\delta - \mathbf{b}^{\delta_k}\|_2^2 + (2\|\bar{\mathbf{x}}\|_1 + \|\mathbf{x}_k\|_1)\|\Psi^\delta_\Theta - \Psi^{\delta_k}_{\Theta_k}\|_2,
\end{aligned}$$

where we used Lemma 2.2.23 in the last inequality. Arguing as in the proof of Theorem 2.2.21 and adapting Lemma 2.2.20, it is possible to show that $\{\mathbf{x}_k\}$ is bounded. From Hypothesis 2.2.3, and since $\mathbf{b}^{\delta_k}$ converges to $\mathbf{b}^\delta$, then there exist $k_0 \in \mathbb{N}$ such that

$$M := 4\Gamma(\bar{\mathbf{x}}) + 1 \geq \Gamma(\mathbf{x}_k), \quad \forall\, k \geq k_0.$$

Thus, for all $\lambda > 0$, the set

$$\mathcal{M}_\lambda(M) := \{\mathbf{x} \in \mathbb{R}^n \mid \frac{1}{2}\|A\mathbf{x} - \mathbf{b}^\delta\|_2^2 + \lambda\|\Delta_{\Psi^\delta_\Theta}\mathbf{x}\|_1 \leq M\}$$

is sequentially pre-compact with respect to the norm topology. Additionally, since the sequence $(\mathbf{x}_k)$ is contained in $\mathcal{M}_\lambda(M)$ for $k \geq k_0$, it possesses a converging subsequence.

Now let $(\mathbf{x}_{k'})$ denote an arbitraty subsequence of $(\mathbf{x}_k)$ that converges to $\tilde{\mathbf{x}} \in \mathcal{D}$ with respect to the norm topology. By continuity, it holds

$$\frac{1}{2}\|A\mathbf{x} - \mathbf{b}^\delta\|_2^2 = \lim_{k'} \frac{1}{2}\|A\mathbf{x}_{k'} - \mathbf{b}_{k'}\|_2^2.$$

Moreover, since $\|\Delta_{\Psi^{\delta_k}_{\Theta_k}} \cdot \|_1$ is at least lower semicontinous, we have

$$\begin{aligned}
\frac{1}{2}\|A\tilde{\mathbf{x}} - \mathbf{b}^\delta\|_2^2 + \lambda\|\Delta_{\Psi^\delta_\Theta}\tilde{\mathbf{x}}\|_1 &\leq \liminf_{k'} \frac{1}{2}\|A\mathbf{x}_{k'} - \mathbf{b}_{k'}\|_2^2 + \lambda \liminf_{k'} \|\Delta_{\Psi^{\delta_{k'}}_{\Theta_{k'}}}\mathbf{x}_{k'}\|_1 \\
&\leq \limsup_{k'} \|A\mathbf{x}_{k'} - \mathbf{b}_{k'}\|_2^2 + \lambda\|\Delta_{\Psi^{\delta_{k'}}_{\Theta_{k'}}}\mathbf{x}_{k'}\|_1 \\
&\leq \lim_{k'} \frac{1}{2}\|A\mathbf{x} - \mathbf{b}_{k'}\|_2^2 + \lambda\|\Delta_{\Psi^{\delta_{k'}}_{\Theta_{k'}}}\mathbf{x}\|_1 \\
&= \frac{1}{2}\|A\mathbf{x} - \mathbf{b}^\delta\|_2^2 + \lambda\|\Delta_{\Psi^\delta_\Theta}\mathbf{x}\|_1, \quad \mathbf{x} \in \mathcal{D}.
\end{aligned}$$

This implies that $\tilde{\mathbf{x}}$ is a minimizer of Γ. □

2.3 Experimental setup

In the previous section, we conducted a theoretical analysis of the graphLa+Ψ method, where, after some effort, we established the convergence and stability of the method under

certain assumptions. However, before we demonstrate the performance of this strategy in 2D Computerized Tomography applications, several key details must be clarified.

First, while we have discussed the behavior of the solutions of the variational model (2.4), we have not yet explained how to compute these solutions in practice. Indeed, when dealing with variational problems of the form (2.4), a variety of optimization techniques can be employed depending on the structure and properties of the problem. Common methods may include gradient-based algorithms or proximal gradient methods, among others. Each technique has its advantages, such as handling non-differentiable terms, exploiting convexity, or ensuring convergence properties. The choice of the optimization method often depends on the specific features of the variational model considered, including the nature of the regularization and fidelity term. To compute approximate solutions for our specific problem (2.4), we used the *Majorization-Minimization (MM) algorithm* with a Generalized Krylov Subspace (GKS) strategy to reduce the computational cost. This algorithm, that will be described in the next chapter, is particularly effective in handling problems where the objective function can be decomposed into simpler surrogate functions that majorize the original function, allowing for more manageable iterative minimization steps.

The formulation of the variational model relies also on the construction of the graph Laplacian operator, which is determined by the choice of the edge-weight function and the node measure. These elements must be carefully selected to satisfy the conditions outlined in Hypothesis 2.2.8, 2.2.10, and 2.2.11. Furthermore, in our initial discussion about the family of reconstructors $\Psi_\Theta \colon Y \to X$, we suggested that deep neural networks (DNNs) could be a promising choice. Since we employ a DNN in our numerical examples, it is necessary to describe its architecture and how it is utilized within this framework. This section will address all these remaining details to provide a comprehensive understanding before moving on to the experimental results.

2.3.1 Graph Laplacian construction

In this work, we consider a distance-based graph Laplacian operator, as detailed in Section §1.5. The proper functioning of this operator critically depends on the choice of the edge-weight function, $w_{\mathbf{x}}$, and the node measure, $\mu_{\mathbf{x}}$. Since the convergence and stability of the graphLa+Ψ method, as established in Section 2.2, rely on several hypotheses, it is crucial to define $w_{\mathbf{x}}$ and $\mu_{\mathbf{x}}$ in such a way that all these assumptions are met.

For the numerical experiments, to compute the graph Laplacian operator we use the following

$$w_{\mathbf{x}}(p,q) = \mathbb{1}_{(0,R]}(\|p-q\|_\infty)\mathrm{e}^{-\frac{|\mathbf{x}(p)-\mathbf{x}(q)|^2}{\sigma^2}}, \quad \mu_{\mathbf{x}}(p) = \mu_{\mathbf{x}} \coloneqq \sqrt{\sum_{p,q\in P} w_{\mathbf{x}}^2(p,q)}. \tag{2.26}$$

The two following propositions show that, with those choices, both $\omega_{\mathbf{x}}$ and $\mu_{\mathbf{x}}$ grant the validity of Hypothesis 2.2.8, 2.2.10 and 2.2.11. Moreover, to conclude this part, the final corollary will show that the constant c that appears in Lemma 2.2.18 is independent from

the dimension n of the vector space $\mathcal{V}_P \simeq X = \mathbb{R}^n$.

Proposition 2.3.1. *The edge-weight function $\omega_{\mathbf{x}}$ ensures the validity of Hypothesis 2.2.8, and of Hypothesis 2.2.10 when A represents the discrete Radon operators introduced in Section §1.3.*

Proof. Using the same notation as in equation (1.50) and (2.26), since $h_{\mathrm{i}}(t) = \mathrm{e}^{-\frac{t^2}{\sigma^2}} > 0$ for every t, then $\omega_{\mathbf{x}}(p,q) > 0$ if and only if $w_{\mathrm{d}}(p,q) > 0$. From the choices in equation (2.26), $w_{\mathrm{d}}(p,q) = \mathbb{1}_{(0,R]}(\|p-q\|_\infty)$ and is independent of $\mathbf{x}$. This proves Hypothesis 2.2.8.

It is immediate to check that the whole set of pixels P is connected with respect to w_{d}, and therefore is connected with respect to $w_{\mathbf{x}}$, for any $\mathbf{x}$. As a consequence, indicating with V the invariant subspace in Lemma 2.2.9, it holds that $V = \ker(\Delta_{\mathbf{x}}) = \{t\mathbf{1} \mid t \in \mathbb{R}\}$ for any $\mathbf{x}$, where $\mathbf{1} \in \mathcal{V}_P$ is the constant function $\mathbf{1}(p) = 1$ for every $p \in P$. For a proof, see [90, Lemmas 0.29 and 0.31].

Since A is a discrete Radon operator, then $A\mathbf{1} > \mathbf{0}$ for any possible configuration of A, such as the number of angles n_a or the number of pixels n_d of the detector. This is due to the fact that each row of A represents line integrals. In particular, this means that $\ker(A) \cap V = \{\mathbf{0}\}$, which is Hypothesis 2.2.10. □

Proposition 2.3.2. *$\omega_{\mathbf{x}}$ and $\mu_{\mathbf{x}}$ satisfy Hypothesis 2.2.11.*

Proof. We need to show that h_{i} and $\mathbf{x}(p) \mapsto \mu_{\mathbf{x}}(p)$ in equation (2.26) are Lipschitz. The first part is trivial, since $h_{\mathrm{i}}(t) = \mathrm{e}^{-\frac{t^2}{\sigma^2}}$ is a smooth function with bounded derivative for every $\sigma^2 > 0$.

Let us observe now that $\mu_{\mathbf{x}}(p) = \|W_{\mathbf{x}}\|_F$ for every $p \in P$, where $W_{\mathbf{x}}$ is the adjacency matrix associated to $w_{\mathbf{x}}$ and $\|\cdot\|_F$ is the Frobenius norm. Therefore, for any $\mathbf{x}, \mathbf{y} \in \mathcal{V}_P$ and any $p \in P$, it holds

$$|\mu_{\mathbf{x}}(p) - \mu_{\mathbf{y}}(p)| = |\|W_{\mathbf{x}}\|_F - \|W_{\mathbf{y}}\|_F| \leq \|W_{\mathbf{x}} - W_{\mathbf{y}}\|_F. \tag{2.27}$$

A generic element of $W_{\mathbf{x}} - W_{\mathbf{y}}$ in position (p,q) is given by

$$w_{\mathbf{x}}(p,q) - w_{\mathbf{y}}(p,q) = w_{\mathrm{d}}(p,q)\left(h_{\mathrm{i}}(|\mathbf{x}(p) - \mathbf{x}(q)|) - h_{\mathrm{i}}(|\mathbf{y}(p) - \mathbf{y}(q)|)\right).$$

Using the same arguments as in (2.16) and (2.18) of Lemma 2.2.18, it is possible to show that

$$|w_{\mathbf{x}}(p,q) - w_{\mathbf{y}}(p,q)| \leq 2w_{\mathrm{d}}(p,q)L'\|\mathbf{x} - \mathbf{y}\|_2,$$

where L' is the Lipschitz constant of h_{i}. From equation (2.26), it holds

$$\begin{aligned} \overline{d} = \max_{p,q}\{w_{\mathrm{d}}(p,q)\} &= 1, \\ \max_{q\in P}\{\mathrm{card}\{p \in P \mid w_{\mathrm{d}}(p,q) \neq 0\}\} &= \max_{q\in P}\{\mathrm{card}\{p \in P \mid \|p-q\|_\infty \leq R\} - 1\} = \overline{\kappa}, \end{aligned}$$

where $\overline{\kappa} = (2R+1)^2 - 1$. Therefore, we have

$$\|W_{\mathbf{x}} - W_{\mathbf{y}}\|_F = \sqrt{\sum_{q\in P}\sum_{p\in P} |w_{\mathbf{x}}(p,q) - w_{\mathbf{y}}(p,q)|^2} \;\leq\; 2L'\overline{d}\overline{\kappa}\sqrt{n}\|\mathbf{x}-\mathbf{y}\|_2,$$

and from equation (2.27) we conclude that $\mathbf{x} \mapsto \mu_{\mathbf{x}}(p)$ is Lipschitz with constant $L'' = 2L'\overline{d}\overline{\kappa}\sqrt{n}$, for every $p \in P$. □

Corollary 2.3.3. *With the choices in equation (2.26), the constant c in Lemma 2.2.18 is independent of n.*

Proof. Using the same notation as in the proof of Lemma 2.2.18, it is an almost straightforward application of Proposition 2.3.2. Indeed, with the choices in equation (2.26) we have

$$\overline{d} = 1; \quad \overline{\kappa} = (2R+1)^2 - 1 \quad \overline{h}_{\mathrm{i}} = 1, \quad \overline{L}'' = 2L'\overline{\kappa}\sqrt{n},$$

where L' is the Lipschitz constant of h_{i}. Recalling that $\mathbf{x}(p) \in [0,1]$, then $\inf_p\{\mu_{\mathbf{x}}(p)\} \geq ne^{-\sigma^{-2}}$ and $\sup_p\{\mu_{\mathbf{x}}(p)\} \leq n$ for every $\mathbf{x}$, and from equation (2.20) we can fix

$$c = \frac{2\overline{\kappa}\,(2nL' + 2L'\overline{\kappa}\sqrt{n})}{n^2 e^{-2\sigma^{-2}}},$$

which is uniformly bounded with respect to n. □

Let us remark that with the choice of $\mu_{\mathbf{x}}$ in equation (2.26), even if the Lipschitz constant of $\mathbf{x} \mapsto \mu_{\mathbf{x}}(p)$ increases with n thanks to Corollary (2.3.3) and Lemma 2.2.18, the convergence of $\Delta_{\Psi_\Theta^\delta}$ for $\delta \to 0$ is uniform with respect to n. This reflects the observations made in [19], where it was introduced the node measure in equation (2.26) to uniformly bound the spectrum of $I + \Delta_{\mathbf{x}}^T\Delta_{\mathbf{x}}$ with respect to the dimension n, and guarantee then a fast convergence of the `lsqr` algorithm.

2.3.2 DNN and graphLa+Net

Among all the possible choices for the family of reconstructors Ψ_Θ, we propose to use a DNN. In this case, the set of parameters Θ contains matrices and vectors, which are the building blocks of DNNs. Informally speaking, a DNN is a long chain of compositions of affine operators and nonlinear activation functions. The set of parameters Θ is then trained by minimizing a loss function over a large number of data as it will be detailed in the next section. When considering Ψ_Θ as a DNN, we specify it by calling the method `graphLa+Net`. A first overview of `graphLa+Net` was proposed in [22].

Note that, in principle, it is possible to make the network parameters Θ independent on the noise level δ by training it multiple times for different values of δ. However, this is rarely done in practice due to the significant amount of time and energy consumption it would require. This limitation has always been a crucial challenge in employing DNNs for regularizing ill-posed problems, as it necessitates the estimate of an optimal noise level δ which is suitable

for different applications. Additionally, opting for $\delta = 0$ is generally not a good choice due to the typical high sensitivity of DNNs to the noise, as observed in [110, 6, 66].

Nonetheless, the regularizing property of graphLa+Ψ effectively addresses this issue. In the subsequent discussion, we will consider DNNs as reconstructors with a fixed $\hat{\Theta}$, where $\hat{\Theta}$ has been trained over a *noiseless* dataset. As will be shown in the numerical examples, the resulting graphLa+Net is not only regularizing and stable but also significantly superior in performance, despite the inherent instability of the original DNN.

The architecture

The considered DNN is a modified version of the U-net, detailed in [128], called Residual U-net (ResU-net) and it was proposed in [65]. The structure of this modified U-net is illustrated in Figure 2.2. U-net is a widely recognized multi-scale Convolutional Neural Network architecture, known for its effectiveness in processing images with global artifacts. This fully convolutional network features a symmetrical encoder-decoder structure, employing strided convolutions to expand its receptive field. The encoder layers' strides create distinct levels of resolution within the network. Each level comprises a fixed number of blocks, where a block consists of a convolutional layer with a fixed number of channels, followed by batch normalization and a ReLU activation function. The number of convolutional channels is doubled at each successive level, starting from a baseline number in the first layer. In the ResU-net case, the network is designed with four levels and a baseline of 64 convolutional channels. As mentioned, in the U-net neural network, the decoder mirrors the encoder but uses upsampling convolutional layers in place of strided convolutions. Furthermore, to preserve high-frequency details, skip connections link the final layer of each encoder level to the corresponding first layer of the decoder. Instead, for the ResU-net have reconfigured the skip connections to work as additions rather than concatenations, a strategy aimed at reducing the total number of parameters. Moreover, we introduce a residual connection that links the input and output layers directly, which implies that the network learns the residual mapping between the input and the expected output. The importance of the residual connection has been observed in [75], where the authors proved that the residual manifold containing the artifacts is easier to learn than the true image manifold.

The considered DNN is Lipschitz continuous by construction. Once the optimal parameters $\hat{\Theta}$ are estimated during the learning phase, the resulting DNN can be employed as a reconstructor Ψ, which remains Lipschitz continuous. Consequently, Hypothesis 2.2.2 is satisfied, as highlighted in Example 2.2.4. In general, the Lipschitz constant L depends on several factors and can be large, which may impact the uniform convergence of $\Delta_{\Psi^\delta_{\hat{\Theta}}}$ in Lemma 2.2.18. However, in the numerical experiments, no such issues were observed, and the overall stability of the graphLa+Net method was excellent.

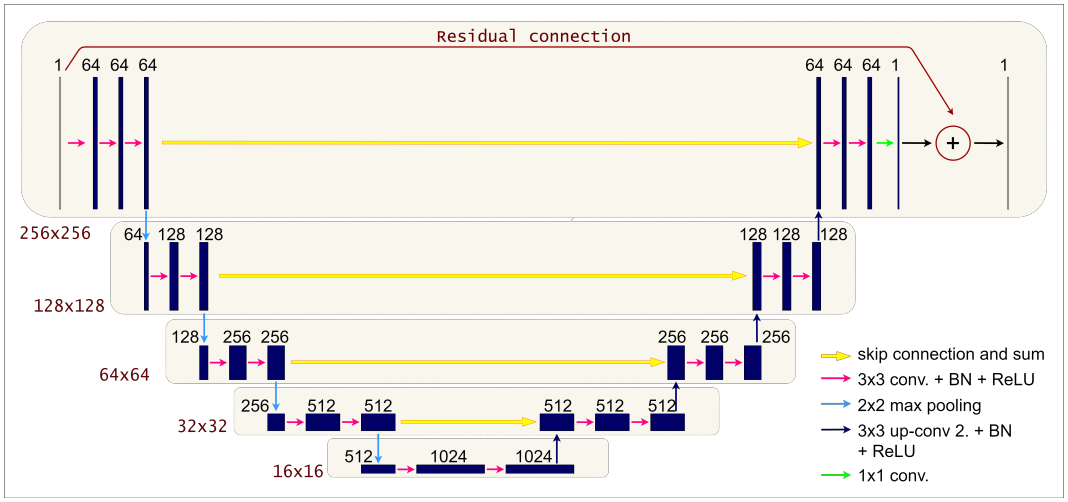
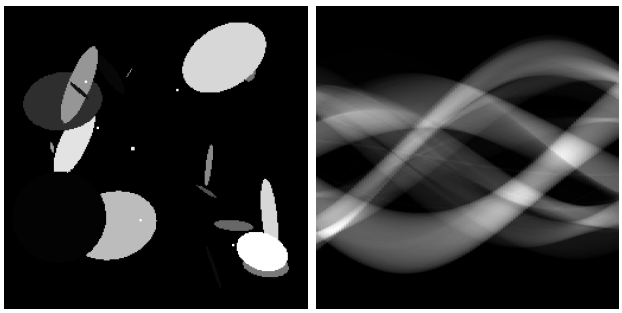


Figure 2.2: A diagram of the ResU-net architecture.

2.4 Numerical experiments

We evaluate the graphLa+Ψ algorithms using two distinct image datasets of X–rays CT. The first is the COULE dataset, which comprises synthetic images with a resolution of 256×256 pixels. These images feature ellipses and lines of varying gray intensities against a dark background. This dataset is publicly available on Kaggle [64]. The second dataset is a subsampled version of the AAPM Low Dose CT Grand Challenge dataset, provided by the Mayo Clinic [108]. It contains real chest CT image acquisitions, each also at a resolution of 256×256 pixels.

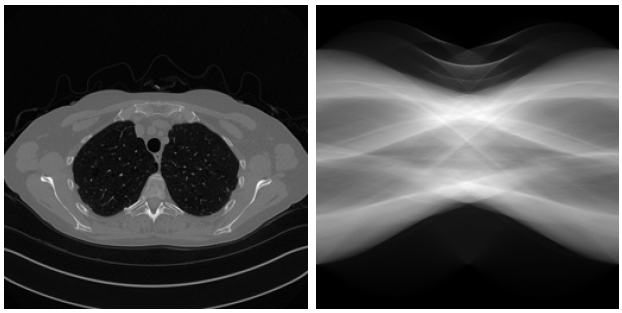
To simulate the sinogram $\mathbf{b}^{\delta}$, we consider n_a different angles evenly distributed within the closed interval $[0, 179]$, where $n_a = 60$ in the experiments with COULE dataset and $n_a = 180$ in the experiments with Mayo dataset. The sinograms are generated using the IRtools toolbox [69].



(a) True image $\mathbf{x}_{\text{gt}}$ (b) Sinogram

Figure 2.3: (a): Example of a $\mathbf{x}_{\text{gt}}$ image from COULE dataset. (b): The resulting sinogram

In Figures 2.3 and 2.4, we present an example of the true image and its resulting sinogram



(a) True image $\mathbf{x}_{\text{gt}}$ (b) Sinogram

Figure 2.4: (a): Example of a $\mathbf{x}_{\text{gt}}$ image from Mayo dataset. (b): The resulting sinogram

of size $n_d \times n_a$, where $n_d = \lfloor \sqrt{2n} \rfloor$ is the number of pixels of the detector. To simulate real-world conditions, we add white Gaussian noise $\boldsymbol{\eta}_\delta$ to the sinogram at an intensity level of ε, indicating that the norm of the noise is ε times the norm of the sinogram. In particular, we compute $\mathbf{b}^\delta$ as:

$$\mathbf{b}^\delta = \mathbf{b} + \varepsilon \|\mathbf{b}\| \frac{\boldsymbol{\eta}_\delta}{\|\boldsymbol{\eta}_\delta\|}.$$

For the DNN described in the previous section, we randomly selected 400 pairs of images from COULE and 3,305 pairs of images from Mayo as training sets, all of the form $(\mathbf{x}_{\text{gt}}, \mathbf{b})$. Indeed, following the discussion outlined in Section §2.3.2, we trained the DNN over the training sets in a supervised manner without extra noise. The process involves finding $\hat{\Theta}$ that minimizes the Mean Squared Error (MSE) between the predicted reconstruction $\Psi_\Theta(\mathbf{b})$ and the ground-truth solution $\mathbf{x}_{\text{gt}}$ that is

$$\hat{\Theta} = \arg\min_{\Theta} \frac{1}{n} \|\Psi_\Theta(\mathbf{b}) - \mathbf{x}_{\text{gt}}\|_2^2.$$

Once the training process is completed, we set $\Theta \equiv \hat{\Theta}$. We did not apply any regularization technique in the training phase. Since the ResU-net architecture is fully convolutional, the input required by the network is an image. Hence, the input sinogram $\mathbf{b}$ has to be pre-processed through a fast algorithm mapping the sinogram to a coarse reconstructed image, such as FBP [110, 87] or a few iterations of a regularizing algorithm [111, 65]. For those experiments, we choose the FBP. The networks have been trained on an NVIDIA RTX A4000 GPU card with 16Gb of VRAM, for a total of 50 epochs and a batch size of 10, arresting it after the loss function stopped decreasing. We used Adam optimizer with a learning rate of 0.001, $\beta_1 = 0.9$, and $\beta_2 = 0.999$, in all the experiments.

This section is divided into two parts, each of which concentrate on different tests for a specific dataset. In both scenarios, we rigorously examined the performance of the `graphLa+Ψ` method to provide a comprehensive analysis of the method's robustness and adaptability. More in detail, we consider a wide range of reconstructors Ψ, including FBP (`graphLa+FBP`), general Tikhonov (`graphLa+Tik`), Total Variation (`graphLa+TV`), and the trained DNN

(graphLa+Net) described in the previous Section §2.3.2. In all cases, the ground truth images $\mathbf{x}_{\text{gt}}$ are drawn outside of the training sets defined for the DNN. In particular, for the general Tikhonov method, we solved the variational problem

$$\arg\min_{\mathbf{x}\in\mathbb{R}^n} \frac{1}{2}\|A\mathbf{x} - \mathbf{b}^\delta\|_2^2 + \lambda\|L\mathbf{x}\|_2^2, \tag{2.28}$$

where L is defined as the finite difference approximation of the 2D gradient. To compute approximate solutions of (2.28), we used a Krylov subspace strategy, searching for a solution within a lower–dimensional subspace ($n = 50$). We estimated an appropriate value for the regularization parameter λ using the Generalized Cross Validation (GCV) criterion (1.47).

On the other hand, for the Total Variation method, we replace the ℓ^2–norm in (2.28) with the ℓ^1–norm. Although this differs from the standard definition of the TV operator introduced in [129], it still enforces sparsity in the coefficients of the gradient of the solution. For this and all the other cases, other than general Tikhonov, we used a Majorization–Minimization strategy combined with a Generalized Krylov Subspace approach, as proposed in [95]. This algorithm will be described in detail in the next chapter. Our implementation incorporates a restarting strategy for the Krylov subspace [36], along with an automatic estimation of the regularization parameter λ using the discrepancy principle (1.46). This value of λ may differ significantly from the one used to compute the initial reconstruction $\Psi_\Theta(\mathbf{b})$ for some reconstructor Ψ_Θ.

For comparison, we will include the reconstruction achieved by our method using the ground truth image $\mathbf{x}_{\text{gt}}$ as a first approximation, labeled as graphLa+$\mathbf{x}_{\text{gt}}$. This serves as an upper bound reference for the effectiveness of all the graphLa+Ψ methods.

The quantitative results of our experiments will be measured by the Relative Reconstruction error (RRE) and the Peak Signal-to-Noise Ratio (PSNR), where

$$\text{RRE}(\mathbf{x}) \coloneqq \frac{\|\mathbf{x}_{\text{gt}} - \mathbf{x}\|^2}{\|\mathbf{x}_{\text{gt}}\|^2}, \qquad \text{PSNR}(\mathbf{x}) \coloneqq 20\log_{10}\left(\frac{255}{\|\mathbf{x}_{\text{gt}} - \mathbf{x}\|}\right),$$

and by the Structural Similarity Index (SSIM) [139]. Finally, all the numerical tests are replicable and the codes can be downloaded from [1].

2.4.1 Example 1: COULE

In this first example, we tested our proposal on an image of the COULE test set acquired by $n_a = 60$ projections, a detector shape of $n_d = \lfloor 256\sqrt{2} \rfloor$, and corrupted with white Gaussian noise with level intensity of 2%. Note that, since $n = 256^2 = 65536$ and $m = n_d \cdot n_a = 21720$, then $m \ll n$, meaning that the problem is highly sparse. Regarding the parameter selection for the edge-weight function in equation (2.26), we chose $R = 5$ and $\sigma = 10^{-3}$.

The quality of the reconstructions achieved with different operators Ψ is presented in Table 2.1. The upper part display the values of the initial reconstructors Ψ, while the middle

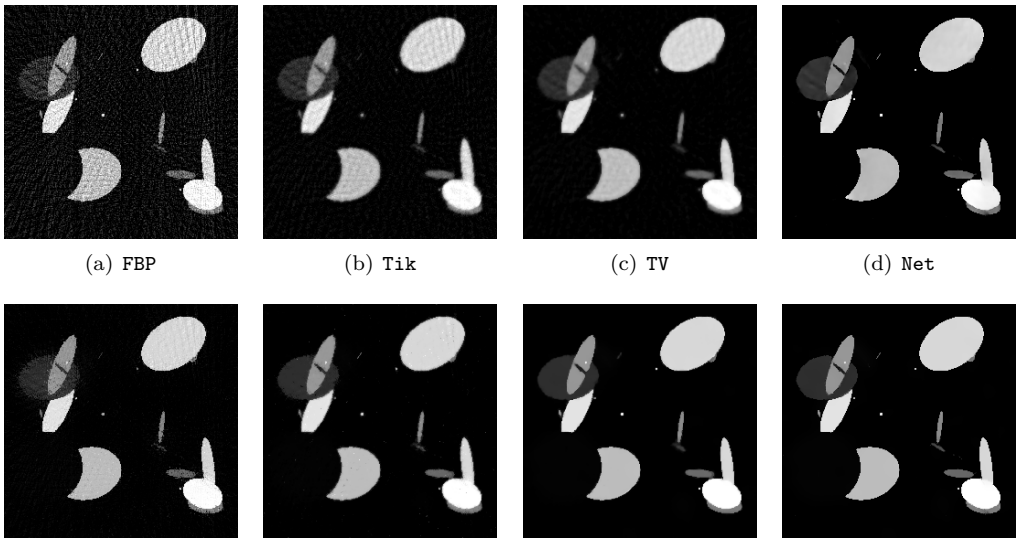
Initial reconstructors Ψ	**RRE**	**SSIM**	**PSNR**
FBP	0.1215	0.1220	18.3101
Tik	0.0622	0.3280	24.1306
TV	0.0450	0.6793	26.9320
Net	0.0205	0.9396	33.7714
graphLa+Ψ			
graphLa+FBP	0.0364	0.6419	28.7701
graphLa+Tik	0.0352	0.8874	29.0812
graphLa+TV	0.0228	0.9697	32.8313
graphLa+Net	**0.0156**	**0.9724**	**36.1128**
graphLa+$\mathbf{x}_{\text{gt}}$	0.0063	0.9820	43.9905
Other comparison methods			
ISTA	0.1769	0.8682	25.3177
FISTA	0.1776	0.8883	25.2839
NETT	0.0302	0.7531	30.4040

Table 2.1: Quality of initial and final reconstruction for different Ψ for the COULE dataset.

part shows the values obtained by combining graphLa+Ψ with the corresponding initial reconstructor Ψ. To provide a more comprehensive and complete analysis of the performance of our proposal, we also tested three other solvers. Given that the ground truth is sparse, we considered an $\ell^2 - \ell^1$ problem with a Haar wavelet regularization operator and applied both FISTA and ISTA. Since both algorithms heavily depend on the choice of a proper steplength, we computed it by estimating the Lipschitz constant of the operator $A^T A$ using ten iterations of the power method. Lastly, we considered a regularization DNN-based method called NETT [101], where the convolutional DNN is trained to reduce the artifacts. The main regularization parameter is tuned by hand in order to get the possible best outcome. The NETT method achieved better results than the proximal methods in terms of both RRE and PSNR. Moreover, the middle part of Table 2.1 shows that graphLa+TV and graphLa+Net are the optimal choices.

Notably, the graphLa+Ψ method results in a substantially greater improvement across all metrics for all initial reconstructors Ψ, with the highest performance attained by graphLa+Net. As further confirmation, Figure 2.5 displays the reconstructions obtained for different Ψ. The level of details and sharpness in the graphLa+Net image is incomparable with all the other methods, besides graphLa+TV, that achieve similar performance.

As a final investigation into the capabilities of our proposal, we tested the stability of the method against varying noise intensity. In Figure 2.6, we present the PSNR and SSIM values for various levels of noise. Similar to the previous analysis, the purple line represents the reconstruction obtained by utilizing the true image $\mathbf{x}_{\text{gt}}$ to compute the graph Laplacian. Notably, even though our neural network was trained with a 0% noise level, the graphLa+Net method consistently outperforms all other cases across all noise levels. Additionally, Figure 2.6 shows that the integration of the graph Laplacian with the DNN serves as an effective regularization method, in contrast to the standalone application of the DNN. Indeed, while the accuracy of the DNN does not improve as the noise intensity approaches zero, pairing

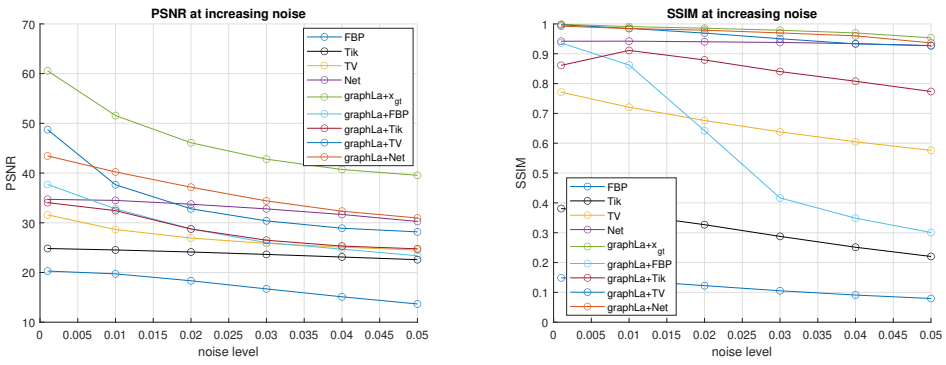


(a) FBP (b) Tik (c) TV (d) Net

(e) graphLa+FBP (f) graphLa+Tik (g) graphLa+TV (h) graphLa+Net

Figure 2.5: Initial and final reconstructions using graphLa+Ψ method for different Ψ_Θ.

it with the graph Laplacian results in a significant accuracy improvement, consistent with the effects of a regularization method.



(a) PSNR (b) SSIM

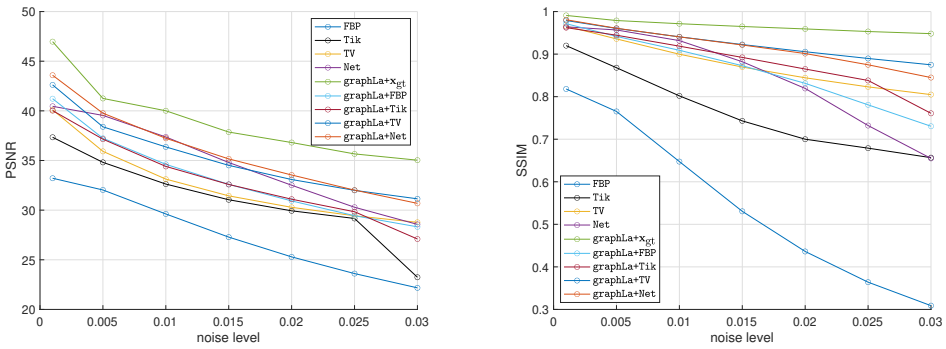
Figure 2.6: PSNR and SSIM for different levels of noise and different reconstructors Ψ_Θ for the COULE dataset.

2.4.2 Example 2: Mayo

In this second example, we test our proposal on an image of the Mayo test set acquired by $n_a = 180$ projections, a detector shape of $n_d = \lfloor 256\sqrt{2} \rfloor$, and corrupted with white Gaussian noise with level intensity of 1%. Regarding the parameter selection for the edge-weight function in equation (2.26), we chose $R = 5$ and $\sigma = 2 \times 10^{-4}$, except for the graphLa+Net method for which we used $R = 3$ and $\sigma = 10^{-3}$.

The Mayo dataset reflects a real-world scenario, indeed the ground truth images $\mathbf{x}_{\text{gt}}$, used

for generating sinograms and for comparison, are not the actual true images. Instead, they are reconstructions obtained from varying numbers of projections and multiple iterations of an appropriate reconstruction algorithm, inherently containing some level of noise. Consequently, comparing metrics for a fixed level of additional noise, as done in Table 2.1, is a bit less informative. Instead, it is still interesting to evaluate the graphLa+Ψ method across different reconstructors Ψ_Θ and various levels of noise intensity using both PSNR and SSIM metrics. As previously noted in the COULE example, the graphLa+Net method consistently outperforms all other cases, even if the neural network was trained with a 0% noise level. In Figure 2.8, we present a visual inspection of some of the reconstructions.



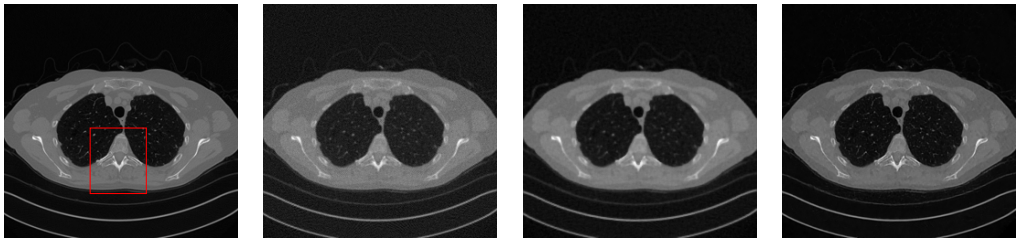
(a) PSNR (b) SSIM

Figure 2.7: PSNR and SSIM for different levels of noise and different reconstructors Ψ_Θ for the Mayo dataset.

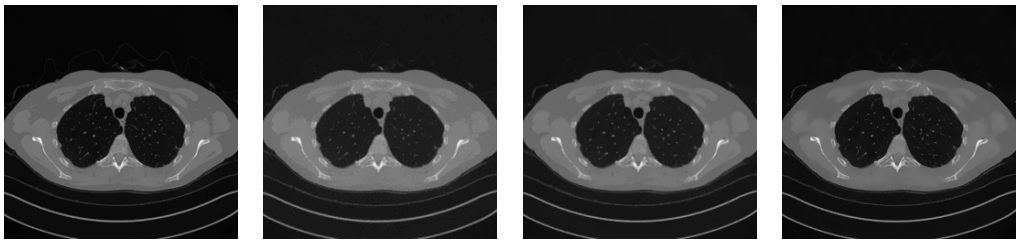
Notably, the graphLa+Net image exhibits the sharpest quality compared to all other cases, and being very close to the upper limit given by graphLa+$\mathbf{x}_{\text{gt}}$. As additional confirmation, in Figure 2.9 we zoom on the central part of the considered image. In this way, we can clearly note that the graphLa+Net approach achieves also an extraordinary quality of detail in the reconstruction.

2.5 Conclusions

In this chapter, we introduced and analyzed a novel regularization method that utilizes the distance–based graph Laplacian operator constructed from an initial approximation of the solution provided by a reconstructor Ψ_Θ. We demonstrated that, under certain, albeit very weak, assumptions on the reconstructor Ψ_Θ, the graphLa+Ψ method is both convergent and stable. The proposed numerical examples showed that graphLa+Ψ significantly improves the quality of reconstructions for any initial reconstructor Ψ. Furthermore, by taking advantage of the regularization properties of graphLa+Ψ, we proposed using a DNN as the initial reconstructor Ψ_Θ. This new hybrid method, called graphLa+Net, combines the regularization benefits of a standard variational approach with the high accuracy of a DNN. The result is a stable regularization method that achieves superior accuracy.

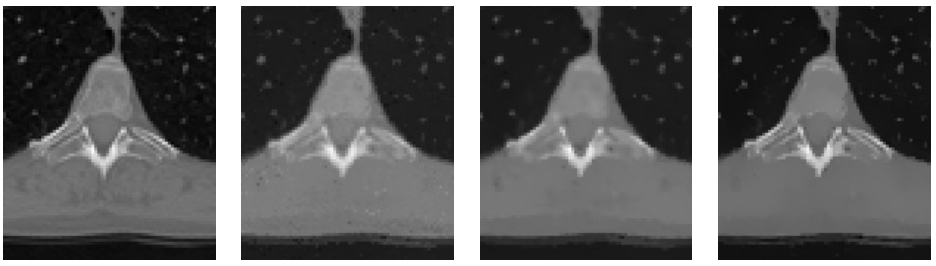


(a) True image $\mathbf{x}_{\text{gt}}$ (b) `Tik` (c) `TV` (d) `Net`



(e) `graphLa+`$\mathbf{x}_{\text{gt}}$ (f) `graphLa+Tik` (g) `graphLa+TV` (h) `graphLa+Net`

Figure 2.8: Initial and final reconstructions using `graphLa+`Ψ method for different Ψ_Θ.



(a) True image $\mathbf{x}_{\text{gt}}$ (b) `graphLa+Tik` (c) `graphLa+TV` (d) `graphLa+Net`

Figure 2.9: Zoom in of the central lower part for different methods.

Future work will focus on refining the choice of the edge-weight function $w_{\mathbf{x}}$, including developing an automatic rule for estimating its parameters. Anticipating the topic of the next chapter, a natural extension of this work, both in theory and in practice, would be to consider the fractional graph Laplacian operator in place of the standard one. As we will see, the use of a fractional exponent enhances the diffusion of information throughout the graph, leading to improved reconstructions.

The fractional graph Laplacian

This is the final chapter of the first part of this work, which focuses on the graph Laplacian operator. Here, we consider a more general regularization approach by substituting the original problem with the $\ell^2 - \ell^q$ functional, where $q \leq 1$. This type of model penalizes the distance between the measured data and the reconstructed data while promoting sparsity in some features of the computed solution. We further propose to use the fractional Laplacian of a properly constructed graph in the ℓ^q term to compute highly accurate reconstructions of the desired images.

A simple model is employed with a fully automatic method that does not require the tuning of any parameters. This method is used to construct the graph, and enhanced diffusion on the graph is achieved by using a fractional exponent in the Laplacian operator. Since the fractional Laplacian is a global operator, i.e. its matrix representation is completely dense, it cannot be explicitly formed and stored. To overcome this limitation, we propose replacing it with an approximation in an appropriate Krylov subspace. This approach can be viewed as an extension of the `graphLa+`Ψ method. Indeed, if one sets $q = 1$ for the regularization term and eliminates the fractional exponent by setting it to one, the method reduces to the `graphLa+`Ψ method with a fixed family of reconstructions Ψ_Θ.

In the final part of the chapter, we demonstrate, from a theoretical perspective and under reasonable assumptions, that the algorithm serves as a regularization method. To evaluate the performance of the fractional version of the graph Laplacian, selected numerical examples in image deblurring and computer tomography are presented.

3.1 The model problem

Using the same notation as before, we consider here the $\ell^2 - \ell^q$ regularization

$$\arg\min_{\mathbf{x}\in\mathbb{R}^n} \frac{1}{2}\|A\mathbf{x} - \mathbf{b}^\delta\|_2^2 + \frac{\lambda}{q}\|L\mathbf{x}\|_q^q, \tag{3.1}$$

where $A \in \mathbb{R}^{m\times n}$ is the discretization of an integral operator, e.g., a blurring matrix, $\mathbf{b}^\delta \in \mathbb{R}^m$ collects some measurements that we assume are corrupted by errors, and $\mathbf{x} \in \mathbb{R}^n$ is an unknown two-dimensional image with n pixels. $L \in \mathbb{R}^{s\times n}$ represent a general regularization operator, $\lambda > 0$ is the regularization parameter and $0 < q \leq 2$. We define $\|\mathbf{x}\|_q^q = \sum_{i=1}^n |x_i|^q$ and we refer to this quantity as ℓ^q-norm, even though, if $q < 1$, this is not a norm since it does not satisfy the triangular inequality. A Bayesian justification of (3.1) was given in [32], while its application to statistics was explored in [33].

When $q \leq 1$, the ℓ^q-norm approximates the so-called ℓ^0-norm that counts the non-vanishing entries of a vector. Therefore, in this case, it is beneficial to select the regularization operator L such that $L\mathbf{x}^\dagger$ is as sparse as possible. It was shown in [35] that, if $L\mathbf{x}^\dagger$ is sparse, the quality of the computed solutions increases as q approaches 0. Popular choices are framelet operators and differential operators. Fractional differential operators have also been investigated to enhance diffusion, in particular with denoising problems [5, 141].

Clearly, a widely used choice for selecting L is to consider the graph Laplacian of a properly constructed graph obtained from a given approximation of $\mathbf{x}^\dagger$; see, e.g., [19, 30, 91, 100, 107, 118, 134, 140]. In our case, we replace the standard definition of the graph Laplacian with its fractional extension. The fractional graph Laplacian has recently attracted the attention of the community working on complex networks [15, 21]. It allows to explore non-local dynamics that can spread the information in the graph. The drawback of this strategy is that the fractional graph Laplacian is a full matrix even if the graph Laplacian is sparse. Therefore, approximation tools need to be explored to perform computations with the fractional graph Laplacian operator. In this direction, the spectral approximation of the graph Laplacian proposed in [134] is very useful and will be employed in our method. In detail, the authors explore the use of the Lanczos method for filtering signals on graphs and observe that only few Lanczos iterations are sufficient to obtain a good approximation of a filtering function of the graph Laplacian.

In this chapter, we expand the algorithmic proposal in [30]. In the latter, the authors first constructed an approximation of the solution of (1.32) with Tikhonov regularization, i.e., by setting $q = 2$ in (3.1). Starting from this approximation they constructed a graph Laplacian to use as a regularization operator in (3.1) with $q < 1$. Here we improve this method as follows. We employ an improved algorithm for the minimization of (3.1) recently proposed in [36]. Moreover, instead of considering the standard graph Laplacian Δ, we consider the fractional extension Δ^α with $\alpha > 0$, where the graph is computed by the approximation of $\mathbf{x}^\dagger$ obtained by Δ. In practice, we add a further step to the algorithm proposed in [30] updating the graph and forcing enhanced diffusion by a fractional exponent. Since in our case the reconstructor Ψ_Θ is fixed, we can relax the notation from that used in the previous chapter by simply indicating with Δ the graph Laplacian operator.

Finally, we prove that the proposed method is a regularization method, i.e., that the computed solutions converge to the exact one as $\delta \to 0$ under some suitable assumptions. Though we have two parameters λ and α to estimate, the proposed approach is completely

automatic. This is achieved by combining the Discrepancy Principle (DP) (1.46) and the whiteness residual principle (see [96]) which requires that the residual $A\mathbf{x} - \mathbf{b}^\delta$ is as white as possible.

3.1.1 The MM–GKS strategy

The Majorization Minimization (MM) algoritm has been developed for the general class of $\ell^p - \ell^q$ regularizations. However, in (3.1), we set $p = 2$ even though the algorithms proposed in [83] allow for a general $0 < p \leq 2$. Note that, the `graphLa+Ψ` method is nothing more than the MM–GKS strategy applied to problem (3.1) with $L = \Delta_{\Psi^\delta_\Theta}$ and $q = 1$.

Since for $q \leq 1$, the minimized functional in (3.1) is non-smooth, as a first step we substitute it with a smooth approximation. Let $\varepsilon > 0$ be a fixed parameter and denote by

$$\Phi_{q,\varepsilon}(t) = \left(\sqrt{t^2 + \varepsilon^2}\right)^q.$$

Assuming that ε is small enough, we can approximate $\|\mathbf{x}\|_q^q$ by

$$\|\mathbf{x}\|_q^q \approx \sum_{i=1}^{n} \Phi_{q,\varepsilon}(x_i), \qquad \mathbf{x} \in \mathbb{R}^n.$$

Note that the function on the right-hand side is everywhere differentiable, while the one on the left is not differentiable if at least one of the components of $\mathbf{x}$ vanishes. Therefore, we substitute problem (3.1) by

$$\min_{\mathbf{x}\in\mathbb{R}^n} \mathcal{J}_\varepsilon(\mathbf{x}), \tag{3.2}$$

where

$$\mathcal{J}_\varepsilon(\mathbf{x}) = \frac{1}{2}\|A\mathbf{x} - \mathbf{b}^\delta\|_2^2 + \frac{\mu}{q}\sum_{i=1}^{s} \Phi_{q,\varepsilon}((L\mathbf{x})_i).$$

The MM algorithm constructs a sequence $\{\mathbf{x}_k\}_{k\in\mathbb{N}}$ that converges to a stationary point of $\mathcal{J}_\varepsilon$. Let $\mathbf{x}_k$ be the current approximation of the solution of (3.2), the MM method first determines a quadratic functional $\mathcal{Q}(\mathbf{x}, \mathbf{x}_k)$ that majorizes $\mathcal{J}_\varepsilon$ everywhere and that is tangent to it in $\mathbf{x}_k$. Then, the new iterate $\mathbf{x}_{k+1}$ is the minimizer of $\mathcal{Q}(\mathbf{x}, \mathbf{x}_k)$.

Given $\mathcal{J}_\varepsilon$ and $\mathbf{x}_k$ one can construct infinitely many quadratic tangent majorants $\mathcal{Q}(\mathbf{x}, \mathbf{x}_k)$. In [83] the authors proposed two choices. We describe here the so-called fixed majorant. The name derives from the fact that, in the one-dimensional case, it coincides with a parabola whose leading coefficient does not depend on $\mathbf{x}_k$. Fix $\varepsilon > 0$, let $\mathbf{u}_k = L\mathbf{x}_k$ and

$$\boldsymbol{\omega}_k = \mathbf{u}_k \left(1 - \left(\frac{(\mathbf{u}^{(k)})^2 + \varepsilon^2}{\varepsilon^2}\right)^{q/2-1}\right),$$

where all operations are meant element-wise, then,

$$\mathcal{Q}(\mathbf{x}, \mathbf{x}_k) = \frac{1}{2}\|A\mathbf{x} - \mathbf{b}^\delta\|_2^2 + \frac{\mu\varepsilon^{q-2}}{2}\left(\|L\mathbf{x}\|_2^2 - 2\langle \boldsymbol{\omega}_k, L\mathbf{x}\rangle\right) + c,$$

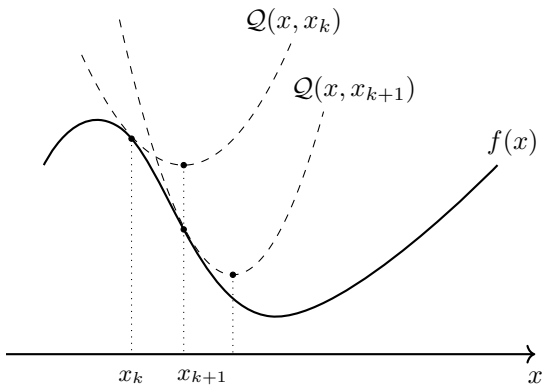


Figure 3.1: Schematic rapresentation of the MM algorithm.

where c is a constant that does not depend on $\mathbf{x}$. Note that $\mathcal{Q}(\mathbf{x}, \mathbf{x}_k)$ is a quadratic tangent majorant of $\mathcal{J}_\varepsilon$ in $\mathbf{x}_k$; see [83]. The approximation $\mathbf{x}_{k+1}$ is obtained by minimizing $\mathcal{Q}$ with respect to $\mathbf{x}$, i.e.,

$$\mathbf{x}_{k+1} = \arg\min_{\mathbf{x}\in\mathbb{R}^n} \left\| \begin{bmatrix} A \\ \nu^{1/2} L \end{bmatrix} \mathbf{x} - \begin{bmatrix} \mathbf{b}^\delta \\ \nu^{1/2} \boldsymbol{\omega}_k \end{bmatrix} \right\|_2^2, \tag{3.3}$$

where $\nu = \mu\varepsilon^{q-2}$. Therefore, we solve the least squares problem (3.3) at each iteration. A schematic rapresentation for a 1D model problem of the MM strategy is depicted in Figure 3.1.

An approximate solution of (3.3) can be computed in a subspace of $\mathbb{R}^n$ of fairly small dimension. Let $V_k \in \mathbb{R}^{n\times\hat{k}}$ be a matrix with orthonormal columns. Assuming that the columns of V_k span the search subspace, we look for a solution of the form

$$\mathbf{x}_{k+1} = V_k \mathbf{y}_{k+1}, \tag{3.4}$$

where $\mathbf{y}_{k+1}$ is obtained solving

$$\mathbf{y}_{k+1} = \arg\min_{\mathbf{y}\in\mathbb{R}^{\hat{k}}} \left\| \begin{bmatrix} AV_k \\ \nu^{1/2} L V_k \end{bmatrix} \mathbf{y} - \begin{bmatrix} \mathbf{b}^\delta \\ \nu^{1/2} \boldsymbol{\omega}_k \end{bmatrix} \right\|_2^2. \tag{3.5}$$

Note that (3.5) is obtained by plugging $\mathbf{x} = V_k\mathbf{y}$ in (3.3). Since the matrices AV_k and LV_k have more rows than columns, we can compute just the first $\hat{k}$ rows of R and the first $\hat{k}$ columns of Q in their QR factorizations. This is called *economic* (or economy-size) QR factorization. These factorizations read

$$\begin{aligned} AV_k &= Q_A R_A \quad \text{with} \quad Q_A \in \mathbb{R}^{m\times\hat{k}}, \quad R_A \in \mathbb{R}^{\hat{k}\times\hat{k}}, \\ LV_k &= Q_L R_L \quad \text{with} \quad Q_L \in \mathbb{R}^{s\times\hat{k}}, \quad R_L \in \mathbb{R}^{\hat{k}\times\hat{k}}, \end{aligned} \tag{3.6}$$

where Q_A and Q_L have orthonormal columns and R_A and R_L are upper triangular. Plugging

the decompositions (3.6) in (3.5), we obtain

$$\mathbf{y}_{k+1} = \arg\min_{\mathbf{y}\in\mathbb{R}^{\hat{k}}} \left\| \begin{bmatrix} R_A \\ \nu^{1/2} R_L \end{bmatrix} \mathbf{y} - \begin{bmatrix} Q_A^T \mathbf{b}^\delta \\ \nu^{1/2} Q_L^T \boldsymbol{\omega}_k \end{bmatrix} \right\|_2^2,$$

which can be solved with direct methods since $\hat{k} \ll n$;

Computing the residual of the normal equation associated with (3.3) and recalling that $\mathbf{x}_{k+1} = V_k \mathbf{y}_{k+1}$, we obtain

$$\mathbf{r}_{k+1} = A^T(AV_k\mathbf{y}_{k+1} - \mathbf{b}^\delta) + \nu L^T(LV_k\mathbf{y}_{k+1} - \boldsymbol{\omega}_k).$$

Following [138], at each iteration, we expand the search subspace by adding to the basis V_k the normalized residual, i.e.,

$$V_{k+1} = [V_k, \mathbf{v}_{k+1}], \qquad \mathbf{v}_{k+1} = \mathbf{r}_{k+1} / \left\|\mathbf{r}_{k+1}\right\|_2 .$$

Note that, in exact arithmetic, $\mathbf{v}_{k+1}$ is orthogonal to the space spanned by the columns of V_k.

Since the following computations are exactly the same for both matrices A and L, we can describe them using a generic matrix $C \in \{A, L\}$.

Following [55], the QR factorizations of CV_{k+1} is computed by updating the QR factorization $CV_k = Q_C R_C$, cf. (3.6), according to

$$CV_{k+1} = [CV_k, C\mathbf{v}_{k+1}] = [Q_C, \mathbf{q}_C] \begin{bmatrix} R_C & \mathbf{r}_C \\ \mathbf{0}^T & \tau_C \end{bmatrix},$$

where

$$\begin{array}{rclcrclcrcl} \tilde{\mathbf{v}}_{k+1} & = & C\mathbf{v}_{k+1}, & & \mathbf{r}_C & = & Q_C^T \tilde{\mathbf{v}}_{k+1}, & & & & \\ \tilde{\mathbf{q}}_C & = & \tilde{\mathbf{v}}_{k+1} - Q_C \mathbf{r}_C, & & \tau_C & = & \|\tilde{\mathbf{q}}_C\|_2, & & \mathbf{q}_C & = & \tilde{\mathbf{q}}_C / \tau_C, \end{array}$$

We now briefly discuss the strategy proposed in [36] to reduce the computational cost of the MM algorithm. The authors observed that in real applications only a few vectors of the Krylov subspace are actually used and that most of the coefficients of $\mathbf{y}_k$ almost vanish. Therefore, they propose to restart the space every r iterations. More in details, if $k \equiv 0 \bmod r$ we set

$$V_k = \mathbf{x}_k / \|\mathbf{x}_k\|_2 \in \mathbb{R}^n.$$

We compute CV_k and its economic QR factorization is easily obtained as

$$CV_k = Q_C R_C, \quad \text{with} \quad Q_C = CV_k / \|CV_k\|_2 \quad \text{and} \quad R_C = \|CV_k\|_2,$$

We then proceed with the iterations as in the MM method. In [36] the authors proved that

the obtained algorithm is a descent method, i.e., it holds

$$\mathcal{J}_\varepsilon\left(\mathbf{x}_{k+1}\right) \leq \mathcal{J}_\varepsilon\left(\mathbf{x}_k\right).$$

Moreover, there exists a converging subsequence $\mathbf{x}^{(k_j)}$. Extensive numerical experience, however, suggests that the whole sequence converges and there is no need to consider subsequences.

Remark 3.1.1. *A commonly used stopping criterion halts the iterations of the method when the difference between two consecutive iterates is smaller than a prescribed tolerance γ, that is*

$$\frac{\|\mathbf{x}_{k+1} - \mathbf{x}_k\|_2}{\|\mathbf{x}_k\|_2} < \gamma.$$

However, thanks to equation (3.4), it is not necessary to compute both $\mathbf{x}_{k+1}$ and $\mathbf{x}_k$ directly. Instead, we can consider the difference between their projections onto the Krylov subspace, namely

$$\frac{\left\|\mathbf{y}_{k+1} - \begin{bmatrix}\mathbf{y}_k \\ 0\end{bmatrix}\right\|_2}{\|\mathbf{y}_k\|_2} < \gamma.$$

This allows us to perform all computations within the lower-dimensional subspace, applying equation (3.4) only to compute the final solution.

3.2 Fractional graph Laplacian

This part is devoted to the analysis of the fractional graph Laplacian operator and, in the last part, we will demostrate that putting it all together, the resulting algorithm is a regularization algorithm. Before all of this, recall that the definition of graph that we are using is based on the physical distance between pixels and on the intensity of each of them. In the same way as in the `graphLa+Ψ` method, from now on the edge–weight function $w_{\mathbf{x}}$ and the node measure $\mu_{\mathbf{x}}$ will be defined as in equation (2.26), that is

$$w_{\mathbf{x}}(p,q) = \mathbb{1}_{(0,R]}(\|p-q\|_\infty)\mathrm{e}^{-\frac{|\mathbf{x}(p)-\mathbf{x}(q)|^2}{\sigma^2}}, \quad \mu_{\mathbf{x}}(p) = \mu_{\mathbf{x}} \coloneqq \sqrt{\sum_{p,q\in P} w_{\mathbf{x}}^2(p,q)}.$$

In this way we can rewrite the graph Laplacian operator Δ as

$$\Delta = \frac{D-\Omega}{\|\Omega\|_F},$$

where Ω is the adjacency matrix associated to the considered graph and D is the degree matrix (see Section §1.5). Since $w_{\mathbf{x}}$ is symmetric so are the adjacency matrix Ω and the graph Laplacian Δ. Moreover, the graph Laplacian is a positive semi-definite operator, i.e. $\forall \mathbf{x} \in \mathbb{R}^n$, $\mathbf{x}^T\Delta\mathbf{x} \geq 0$, and it is also stochastic by rows. Thus, we have that

$$\ker(\Delta) \supseteq \operatorname{span}\{\mathbf{1}\},$$

where $\mathbf{1}$ is the constant vector with all components equal to one. This property will be exploited later when we will describe how to approximate the fractional power of Δ.

3.2.1 Initial reconstruction

As already discussed, the choice of the initial approximation of $\mathbf{x}^\dagger$ that we use to construct the graph Laplacian Δ plays a crucial role. In this case, we compute an initial reconstruction by solving (3.1) with L defined as the finite difference approximation of the 2D gradient, which is

$$L = \begin{bmatrix} L_1 \otimes I \\ I \otimes L_1 \end{bmatrix} \qquad \text{with} \qquad L_1 = \begin{bmatrix} -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \\ 1 & & & -1 \end{bmatrix}, \tag{3.7}$$

where $\otimes$ is the Kronecker product and I is the identity matrix. The size of the two square matrices L_1 and I depends on the size of the image $\mathbf{x}^\dagger$ to restore. For simplicity, if we assume that $\mathbf{x}^\dagger \in \mathbb{R}^{n\times n}$, then the matrices L_1 and I have the same size of $\mathbf{x}^\dagger$. Therefore, the matrix $L \in \mathbb{R}^{2n^2\times n^2}$ is extremely sparse. To compute an approximate solution to problem (3.1) with L defined as in (3.7), we employ the GKS strategy, similar to what was done for problem (3.3), exploiting equation (3.4). To compute a proper value of the regularization parameter λ, we use the GCV strategy (1.47).

On the other hand, in the MM iterations (3.3) we consider a non-stationary λ_k (and in turn ν_k) and we use the discrepancy principle (1.46) to determine the regularization parameter. Specifically, at each iteration k, we select λ_k such that

$$\|A\mathbf{x}_{k+1} - \mathbf{b}^\delta\|_2 = \tau\delta,$$

with $\tau > 1$. In [35] the authors proved that such λ_k exists and the iterates converge (up to subsequences) to a certain $\hat{\mathbf{x}}$ that satisfies the DP as well. In particular, exploiting again equation (3.4), we apply the discrepancy principle to determine the regularization parameter λ_k in the Krylov subspace of dimension $\hat{k}$ [92]. For completeness, we report all the computations in Algorithm 1.

After computing an initial approximation $\widehat{\mathbf{x}}$ of $\mathbf{x}^\dagger$, we use $\widehat{\mathbf{x}}$ to construct the graph Laplacian Δ. Note that by construction Δ is symmetric and hence in Algorithm 1 the only operation involving Δ is the matrix-vector product. Moreover, since Δ is a sparse matrix, the matrix-vector product can be computed efficiently with a linear cost in n. Therefore, the same Algorithm 1 can be used to solve the $\ell^2 - \ell^q$ problem in (3.1) with Δ as regularization operator. We report the computations in Algorithm 2.

3.2.2 Krylov approximation of Δ^α

To further improve the quality of the reconstruction $\mathbf{x}^*$ obtained by Algorithm 2, we construct a new graph based on $\mathbf{x}^*$ and we take the α-th power of the new graph Laplacian

Algorithm 1: Nonstationary $\ell^2 - \ell^q$

Input : A, $\mathbf{b}^\delta$, δ, q, L, $\mathbf{x}^0$, ε, τ, K, r, γ

Construct $V_0 \in \mathbb{R}^{n \times \hat{k}}$ such that $V_0^T V_0 = I$;

Compute and store AV_0 and LV_0, and their economic QR factorizations $AV_0 = Q_A R_A$; $LV_0 = Q_L R_L$;

for $k = 0, 1, \ldots, K$ **do**

  if *($k \equiv 0 \bmod r$) and ($k \neq 0$)* **then**

    $V_k = \mathbf{x}_k / \|\mathbf{x}_k\|_2$;

    Compute and store AV_k;

    $R_A = \|AV_k\|_2$;

    $Q_A = AV_k / R_A$;

    Compute and store LV_k;

    $R_L = \|LV_k\|_2$;

    $Q_L = LV_k / R_L$;

  $\mathbf{u}^{(k)} = L\mathbf{x}_k$;

  $\boldsymbol{\omega}_k = \mathbf{u}^{(k)} \left(1 - \left(\frac{(\mathbf{u}^{(k)})^2 + \varepsilon^2}{\varepsilon^2}\right)^{q/2-1}\right)$;

  $\mathbf{y}_{k+1} = \arg\min_{\mathbf{y}} \|R_A \mathbf{y} - Q_A^T \mathbf{b}^\delta\|_2^2 + \nu^{(k)} \|R_L \mathbf{y} - Q_L^T \boldsymbol{\omega}_k\|_2^2$, where ν^k is such that $\|R_A \mathbf{y}_{k+1} - Q_A^T \mathbf{b}^\delta\|_2 = \tau\delta$;

  if $\left\| \mathbf{y}_{k+1} - \begin{bmatrix} \mathbf{y}_k \\ 0 \end{bmatrix} \right\|_2 \leq \gamma \|\mathbf{y}_k\|_2$ **then**

    exit;

  $\mathbf{r}_{k+1} = A^T(AV_k \mathbf{y}_{k+1} - \mathbf{b}^\delta) + \nu L^T(LV_k \mathbf{y}_{k+1} - \boldsymbol{\omega}_k)$;

  $\mathbf{v}_{k+1} = \mathbf{r}_{k+1} / \|\mathbf{r}_{k+1}\|_2$;

  $V_{k+1} = [V_k,\ \mathbf{v}_{k+1}]$;

  $AV_{k+1} = [AV_k,\ A\mathbf{v}_{k+1}]$;

  $LV_{k+1} = [LV_k,\ L\mathbf{v}_{k+1}]$;

  Update the QR factorizations of AV_{k+1} and LV_{k+1};

$\mathbf{x}^* = V_k \mathbf{y}_{k+1}$;

Output: $\mathbf{x}^*$

Δ in order to better diffuse the information along the graph. Finally, we solve the $\ell^2 - \ell^q$ problem in (3.1), for fixed $\alpha > 0$, with Δ^α as regularization operator.

However, we are now faced with the following issues, Δ^α is a full matrix and hence it cannot be explicitly formed, therefore, it has to be approximated. Moreover, we have to provide an automatic rule for the computation of a suitable value for α. In what follows, we firstly describe how to perform matrix-vector products with Δ^α. A possible strategy to automatically estimate proper values of the fractional exponent α will be described in the second to last part of this section.

Let $\alpha > 0$, we wish to solve

$$\arg\min_{\mathbf{x}} \frac{1}{2}\|A\mathbf{x} - \mathbf{b}^\delta\|_2^2 + \frac{\lambda}{q}\|\Delta^\alpha \mathbf{x}\|_q^q. \tag{3.8}$$

Recall that, by construction, Δ is symmetric and positive semidefinite, hence it is possible to determine an orthonormal basis of eigenvectors of Δ. Let $\mu_j \geq 0$ be the eigenvalues of Δ for $j = 1, \ldots, n$, and let Q be the matrix formed by the eigenvectors of Δ, then

$$\Delta^\alpha = Q\Lambda^\alpha Q^T, \tag{3.9}$$

where $\Lambda^\alpha = \operatorname{diag}(\mu_1^\alpha, \ldots, \mu_n^\alpha)$, is symmetric and positive semidefinite. Unfortunately, it is computationally too expensive to compute the spectral decomposition of Δ.

We now discuss how we implement the MM algorithm when applied to the minimization (3.8). The only difference with Algorithm 1 is in the computation of the matrix-vector products with Δ^α and $(\Delta^\alpha)^T$. Since Δ^α is symmetric and positive semidefinite as Δ, it is sufficient to discuss how to implement the matrix-vector products with Δ^α without explicitly constructing the matrix Δ^α.

We distinguish the case $\alpha \in \mathbb{N}$ and $\alpha \in \mathbb{R}^+ \setminus \mathbb{N}$. In the first case, if $\alpha = 1$, there is nothing to discuss, therefore, we assume $\alpha > 1$. Since Δ is sparse, it is computationally attractive to perform the matrix-vector product as

$$\Delta^\alpha \mathbf{x} = \underbrace{\Delta(\Delta(\ldots(\Delta\mathbf{x})))}_{\alpha \text{ times}},$$

instead of to explicitly form the matrix Δ^α.

If $\alpha \in \mathbb{R}^+ \setminus \mathbb{N}$, since $n \gg 1$, we cannot either explicitly form the matrix Δ^α or compute its spectral decomposition (3.9). Therefore, following the proposal in [134], to compute $\Delta^\alpha \mathbf{x}$ for a given $\mathbf{x}$, we project the problem in the Krylov subspace

$$\mathcal{K}_d(\Delta, \mathbf{x}) = \operatorname{span}\left\{\mathbf{x}, \Delta\mathbf{x}, \ldots, \Delta^{d-1}\mathbf{x}\right\},$$

where we assume $d \ll n$ small enough so that the dimension of $\mathcal{K}_d(\Delta, \mathbf{x})$ is d. Using d steps of the Lanczos algorithm (see, e.g., [74]), with starting vector $\mathbf{x}$, we obtain the following

relation

$$\Delta V_d = V_{d+1} T_{d+1,d},$$

where $V_{d+1} \in \mathbb{R}^{n\times(d+1)}$ has orthonormal columns that spans the space $\mathcal{K}_{d+1}(\Delta, \mathbf{x})$ and $T_{d+1,d} \in \mathbb{R}^{(d+1)\times d}$ is of the form

$$T_{d+1,d} = \begin{bmatrix} \alpha_1 & \beta_2 & & & & \\ \beta_2 & \alpha_2 & \beta_3 & & & \\ & \beta_3 & \ddots & \ddots & & \\ & & \ddots & \ddots & \beta_{d-1} & \\ & & & \beta_{d-1} & \alpha_{d-1} & \beta_d \\ & & & & \beta_d & \alpha_d \\ & & & & & \beta_{d+1} \end{bmatrix}.$$

We use the matrix $T_{d,d}$, i.e., the leading $d \times d$ block of $T_{d+1,d}$, to approximate the matrix-vector product $\Delta^\alpha \mathbf{x}$ as

$$\begin{aligned} \Delta^\alpha \mathbf{x} &\approx \left(V_d T_{d,d} V_d^T\right)^\alpha \mathbf{x} \\ &= \left(V_d U \Sigma U^T V_d^T\right)^\alpha \mathbf{x} \\ &= V_d U \Sigma^\alpha U^T V_d^T \mathbf{x}, \end{aligned}$$

where $T_{d,d} = U\Sigma U^T$ is the eigenvalue decomposition of $T_{d,d}$, that exists since $T_{d,d}$ is symmetric. If $d \ll n$ the computation of U and Σ can be performed with small computational burden.

Algorithm 2: Graph Laplacian $\ell^2 - \ell^q$

Input : A, $\mathbf{b}^\delta$, δ, q, L, $\mathbf{x}^0$, ε, τ, K, r, γ, σ, R, d

Compute, using Algorithm 1 with inputs A, $\mathbf{b}^\delta$, δ, q, L, $\mathbf{x}^0$, ε, τ, K, r, the approximation $\hat{\mathbf{x}}$;

Construct the adjacency matrix

$$\Omega_{p,q} = \begin{cases} \mathrm{e}^{-(\widehat{X}_{p_1,p_2} - \widehat{X}_{q_1,q_2})^2/\sigma} & \text{if } 0 < \|\mathbf{p}-\mathbf{q}\|_\infty \le R, \\ 0 & \text{otherwise,} \end{cases}$$

where $\hat{\mathbf{x}} = \mathrm{vec}\left(\hat{X}\right)$, p and q are the lexicographic indexes of $\mathbf{p} = \begin{bmatrix} p_1, & p_2 \end{bmatrix}^T$ and $\mathbf{q} = \begin{bmatrix} q_1, & q_2 \end{bmatrix}^T$, respectively;

Construct the diagonal matrix $D_{p,p} = \sum_{q=1}^n \Omega_{p,q}$;

$\Delta = \frac{D-\Omega}{\|\Omega\|_F}$;

Compute, using Algorithm 1 with inputs A, $\mathbf{b}^\delta$, δ, q, Δ, $\mathbf{x}^0$, ε, τ, K, r, the approximation $\mathbf{x}^*$, where every product with Δ is performed using sparse matrices;

Output: $\mathbf{x}^*$

3.2.3 The fractional exponent α

According to the analysis and the numerical results in [134], only a few iterations of the Lanczos method are sufficient to obtain a good approximation of the graph Laplacian, in particular, when a filtering function is applied to it. Therefore, the dimension d of the Krylov subspace introduced in the previous subsection is not crucial and even a small d is enough to obtain a good approximation. For instance, we fix $d = 10$ in the numerical results for different applications (deblurring and computer tomography) and images of different sizes.

In (3.8), we need to determine two parameters, the regularization parameter λ and the fractional parameter α. Moreover, we would like to ensure that our choices guarantee that the obtained algorithm is a regularization method. Due to the Bakushinskii veto [9], in order to construct a regularization method, we need to assume that an accurate estimate of the norm of the noise δ is available.

We proceed as follows. Let $0 < \alpha_{\min} < \alpha_{\max}$ be two fixed values and let $J \in \mathbb{N}$ be given. We define

$$\alpha_j = \alpha_{\min} + j\frac{\alpha_{\max} - \alpha_{\min}}{J}, \quad j = 0, 1, \ldots, J.$$

For each j we consider the minimization problem

$$\mathbf{x}_\lambda = \arg\min_{\mathbf{x}} \frac{1}{2}\|A\mathbf{x} - \mathbf{b}^\delta\|_2^2 + \frac{\lambda}{q}\|\Delta^{\alpha_j}\mathbf{x}\|_q^q.$$

We would like to select λ so that the discrepancy principle is satisfied. This can be done a posteriori, as described in [34], by trying several values of λ and selecting the largest one such that $\|A\mathbf{x}_\lambda - \mathbf{b}^\delta\|_2 \leq \tau\delta$. However, this may become computationally expensive if J is large or if many values of λ are considered. Therefore, we follow the strategy proposed in [35] and described in Algorithm 1, where the only difference is that every multiplication with Δ^{α_j} is performed using the Lanczos algorithm as described above. Therefore, for each α_j, we compute a $\mathbf{x}_j$ such that

$$\|A\mathbf{x}_j - \mathbf{b}^\delta\|_2 = \tau\delta, \quad j = 0, \ldots, J.$$

We now discuss how we select the solution $\mathbf{x}_j$. Following the idea in [34, 96], we wish to select $\mathbf{x}_j$ using the residual whiteness principle. Ideally, if $\mathbf{x}_j = \mathbf{x}^\dagger$ for a certain j, then $\mathbf{r}_j = \mathbf{b}^\delta - A\mathbf{x}_j = \boldsymbol{\eta}_\delta$ and, therefore, $\mathbf{r}_j$ would be white since $\boldsymbol{\eta}_\delta$ defined in (1.31) has this property. We propose to select j such that $\mathbf{r}_j$ is as white as possible. We consider the measure of whiteness, introduced by Lanza et al. [96], defined by

$$\mathcal{W}(\mathbf{r}) = \frac{\|\mathbf{r} \star \mathbf{r}\|_2^2}{\|\mathbf{r}\|_2^4}, \tag{3.10}$$

where $\star$ denotes the two-dimensional convolution. The computation of $\mathcal{W}(\mathbf{r})$ can be performed cheaply thanks to the convolution theorem. Let $\mathbb{F}$ be the discrete Fourier matrix,

then

$$\mathcal{W}(\mathbf{r}) = \frac{\| \, |\mathbb{F}\mathbf{r}|^2 \, \|_2^2}{\|\mathbb{F}\mathbf{r}^4\|_2},$$

where $|\cdot|$ denotes the modulus of a complex number and the operations are meant element-wise. Using the function $\mathcal{W}$ we compute

$$\widehat{j} = \arg\min_j \mathcal{W}(\mathbf{r}_j)$$

and select our approximate solution as $\mathbf{x}^* = \mathbf{x}_{\widehat{j}}$. All the computations are summarized in Algorithm 3.

Algorithm 3: Fractional Graph Laplacian $\ell^2 - \ell^q$

Input : A, $\mathbf{b}^\delta$, δ, q, Δ, $\mathbf{x}^0$, ε, τ, K, r, γ, σ, R, $\alpha_{\min}$, $\alpha_{\max}$, J, d

Compute the approximation $\hat{\mathbf{x}}$, using Algorithm 2 with inputs A, $\mathbf{b}^\delta$, δ, q, Δ, $\mathbf{x}^0$, ε, τ, K, r, ;

Construct the adjacency matrix

$$\Omega_{p,q} = \begin{cases} \mathrm{e}^{-(\widehat{X}_{p_1,p_2} - \widehat{X}_{q_1,q_2})^2/\sigma} & \text{if } 0 < \|\mathbf{p} - \mathbf{q}\|_\infty \leq R, \\ 0 & \text{otherwise,} \end{cases}$$

where $\hat{\mathbf{x}} = \mathrm{vec}\left(\hat{X}\right)$, p and q are the lexicographic indexes of $\mathbf{p} = \begin{bmatrix} p_1, & p_2 \end{bmatrix}^T$ and $\mathbf{q} = \begin{bmatrix} q_1, & q_2 \end{bmatrix}^T$, respectively;

Construct the diagonal matrix $D_{p,p} = \sum_{q=1}^n \Omega_{p,q}$;

$\Delta = \frac{D-\Omega}{\|\Omega\|_F}$;

for *j=1,...,J* **do**

  Compute the approximation $\mathbf{x}_j$, using Algorithm 1 with inputs A, $\mathbf{b}^\delta$, δ, q, Δ^{α_j}, $\mathbf{x}^0$, ε, τ, K, r, where every product with Δ^{α_j} is performed using d steps of Lanczos, as discussed above;

  $\boldsymbol{\theta}_j = \mathbb{F}\left(\mathbf{b}^\delta - A\mathbf{x}_j\right)$;

  $w_j = \frac{\||\boldsymbol{\theta}_j|^2\|^2}{\|\boldsymbol{\theta}_j\|^4}$;

$\widehat{j} = \arg\min_j\{w_j\}$;

$\mathbf{x}^* = \mathbf{x}_{\widehat{j}}$;

Output: $\mathbf{x}^*$

3.2.4 Theoretical results

Before we turn to the last part on numerical tests, we now discuss some theoretical properties of our method. In particular, we wish to show that Algorithm 3 is a regularization method, i.e., that, if $\delta_j \searrow 0$ as $j \to \infty$, denoting by $\mathbf{x}_j^*$ the solution obtained with data $\mathbf{b}^{\delta_j}$, where $\|\mathbf{b} - \mathbf{b}^{\delta_j}\|_2 \leq \delta_j$, then

$$\limsup_{j\to\infty} \|\mathbf{x}_j^* - \mathbf{x}^\dagger\|_2 = 0.$$

In order to prove this, as it was done in [35], we need to assume that $A \in \mathbb{R}^{m\times n}$ is of full rank, that $m \geq n$, and that $\mathbf{b} \in \mathcal{R}(A)$. This ensures that the least squares solution of

$$\min_{\mathbf{x}} \|A\mathbf{x} - \mathbf{b}\|_2$$

is unique and coincides with $\mathbf{x}^\dagger$. If A is not of full rank, one may consider the slightly modified problem

$$\min_{\mathbf{x}} \|\widetilde{A}\mathbf{x} - \widetilde{\mathbf{b}}\|_2,$$

with

$$\widetilde{A} = \begin{bmatrix} A \\ \theta I \end{bmatrix} \quad \text{and} \quad \widetilde{A} = \begin{bmatrix} \mathbf{b} \\ \mathbf{0} \end{bmatrix},$$

where $I \in \mathbb{R}^{n\times n}$ denotes the identity matrix and $\theta \in \mathbb{R}^+$ is a small number. This usually does not change the numerical results, especially if θ is smaller than the machine precision, and we do not consider this modification in our computations.

We are now in a position to show our main result.

Theorem 3.2.1. *Let $A \in \mathbb{R}^{m\times n}$ be of full column rank with $m \geq n$ and let $\mathbf{b} \in \mathcal{R}(A)$. Let $\left\{\mathbf{b}^{\delta_j}\right\}_{j\in\mathbb{N}} \subset \mathbb{R}^m$ be a sequence of vectors such that*

$$\|\mathbf{b}^{\delta_j} - \mathbf{b}\|_2 \leq \delta_j,$$

with $\delta_j \searrow 0$ as $j \to \infty$. Denote by $\mathbf{x}_j^*$ the output of Algorithm 3 with input data $\mathbf{b}^{\delta_j}$, then there exists a converging subsequence $\left\{\mathbf{x}_{j_k}^*\right\}_{j_k\in\mathbb{N}}$ such that

$$\limsup_{j_k\to\infty} \|\mathbf{x}_{j_k}^* - \mathbf{x}^\dagger\|_2 = 0,$$

where $\mathbf{x}^\dagger = A^\dagger\mathbf{b}$.

Proof. The proof is similar to the one in [35]. Let us first observe that, by construction, for all $j \in \mathbb{N}$, we have

$$\|A\mathbf{x}_j^* - \mathbf{b}^{\delta_j}\|_2 = \tau\delta_j.$$

Using the fact that

$$\|A\mathbf{x}_j^* - \mathbf{b}^{\delta_j}\|_2 \geq \left|\|A\mathbf{x}_j^*\|_2 - \|\mathbf{b}^{\delta_j}\|_2\right| \geq \|A\mathbf{x}_j^*\|_2 - \|\mathbf{b}^{\delta_j}\|_2,$$

we have

$$\begin{aligned} \|A\mathbf{x}_j^*\|_2 &\leq \|A\mathbf{x}_j^* - \mathbf{b}^{\delta_j}\|_2 + \|\mathbf{b}^{\delta_j}\|_2 \\ &= \tau\delta_j + \|\mathbf{b}^{\delta_j} - \mathbf{b} + \mathbf{b}\|_2 \\ &\leq \tau\delta_j + \|\mathbf{b}^{\delta_j} - \mathbf{b}\|_2 + \|\mathbf{b}\|_2 \\ &\leq (1+\tau)\delta_j + \|\mathbf{b}\|_2. \end{aligned} \tag{i}$$

Let σ_n denote the smallest singular value of A. Since we assumed that A is of full column rank and $m \geq n$ we have that $\sigma_n > 0$, therefore

$$\|A\mathbf{x}\|_2 \geq \sigma_n \|\mathbf{x}\|_2 \qquad \forall \mathbf{x} \in \mathbb{R}^n. \tag{ii}$$

Combining these two inequalities (i) and (ii), and the fact that δ_j is monotonically decreasing, we obtain

$$\|\mathbf{x}_j^*\|_2 \leq \frac{(1+\tau)\delta_1 + \|\mathbf{b}\|_2}{\sigma_n}, \quad \forall j \in \mathbb{N},$$

i.e., that the sequence $\left\{\mathbf{x}_j^*\right\}_{j\in\mathbb{N}}$ is uniformly bounded. Since $\left\{\mathbf{x}_j^*\right\}_{j\in\mathbb{N}}$ is uniformly bounded, it admits a converging subsequence $\left\{\mathbf{x}_{j_k}^*\right\}_{j_k\in\mathbb{N}}$. In particular, we obtain

$$\begin{aligned}
0 &\leq \limsup_{j_k\to\infty} \|\mathbf{x}_{j_k}^* - \mathbf{x}^\dagger\|_2 \leq \limsup_{j_k\to\infty} \frac{1}{\sigma_n}\|A\mathbf{x}_{j_k}^* - A\mathbf{x}^\dagger\|_2 \\
&= \limsup_{j_k\to\infty} \frac{1}{\sigma_n}\|A\mathbf{x}_{j_k}^* - \mathbf{b}\|_2 = \limsup_{j_k\to\infty} \frac{1}{\sigma_n}\|A\mathbf{x}_{j_k}^* - \mathbf{b}^{\delta_{j_k}} + \mathbf{b}^{\delta_{j_k}} - \mathbf{b}\|_2 \\
&\leq \limsup_{j_k\to\infty} \frac{1}{\sigma_n}\left\{\|A\mathbf{x}_{j_k}^* - \mathbf{b}^{\delta_{j_k}}\|_2 + \|\mathbf{b}^{\delta_{j_k}} - \mathbf{b}\|_2\right\} \\
&\leq \limsup_{j_k\to\infty} \frac{1}{\sigma_n}(1+\tau)\delta_{j_k} = 0
\end{aligned}$$

which concludes the proof. □

3.3 Numerical experiments

In this final section we show some numerical examples obtained using the fractional graph Laplacian. We compare our results with the methods proposed in [35] and [30]. The $\ell^2 - \ell^q$ TV algorithm proposed in [35] is Algorithm 1 where the operator L is the gradient defined in (3.7). The $\ell^2 - \ell^q$ algorithm with graph Laplacian proposed in [30] is Algorithm 2 up to the modification of the restarting strategy described in Section 3.1.1, which reduces the computational time without deteriorating the quality of the restored image. The initial approximation $\hat{\mathbf{x}}$ used to construct the graph in Algorithm 2 is computed solving the $\ell^2 - \ell^2$ TV method, i.e., the minimization problem (3.1) with $q = 2$ and L being the gradient defined in (3.7). This solution $\hat{\mathbf{x}}$ can be computed by the fast Fourier transform for image deblurring or by a generalized Krylov subspace method [125], where the regularization parameter λ is estimated by the generalized cross validation (1.47).

As it was shown in [35], the quality of the reconstructions increases as q approaches 0. However, a too small value of q may lead to numerical instability. Hence, we set $q = 0.1$.

Lastly, for our fractional graph Laplacian $\ell^2 - \ell^q$ method we use Algorithm 3 according to the following strategy:

1. compute an initial reconstruction $\hat{\mathbf{x}}$ by solving the $\ell^2 - \ell^2$ TV problem;

2. construct the graph associated to $\hat{\mathbf{x}}$ and compute a better reconstruction $\mathbf{x}^*$ by Algorithm 2 ($\ell^2 - \ell^q$ graph Laplacian);
3. construct the graph associated to $\mathbf{x}^*$ and compute a new reconstruction by Algorithm 3 ($\ell^2 - \ell^q$ fractional graph Laplacian).

Vast numerical experience suggests that this combination of the three algorithms reliably products extremely accurate approximate solutions. Moreover, thanks to the projection in the GKS and Krylov subspace as well as the restart technique employed, the computational cost of the procedure is reasonable and the computations can be easily performed on any machine.

As already done for the `graphLa+Ψ` method, we compare the strategies above in terms of accuracy using the Relative Restoration Error (RRE) and the Peak Signal to Noise Ration (PSNR) that we recall are defined as

$$\mathrm{RRE}(\mathbf{x}) = \frac{\|\mathbf{x} - \mathbf{x}_{\mathrm{gt}}\|_2}{\|\mathbf{x}_{\mathrm{gt}}\|_2}, \qquad \mathrm{PSNR}(\mathbf{x}) = 20 \log_{10} \left(\frac{255}{\|\mathbf{x} - \mathbf{x}_{\mathrm{gt}}\|_2} \right).$$

Moreover, we consider also the Structure SIMilarity index (SSIM), introduced in [139]. The definition of the SSIM is extremely involved, here we simply recall that this statistical index measures how structurally similar two images are, in particular, the higher the SSIM the more similar the images are, and its highest achievable value is 1.

We will consider two different applications: a deblurring problem and a X–rays Computer Tomography (CT) reconstruction.

We set the restarting parameter $r = 30$ and the smoothing parameter $\varepsilon = 10^{-1}$. We would like to stress that the results obtained by the algorithm is not very sensitive to the choice of this parameters. Regarding the edge—weight function $w_{\mathbf{x}}$, the coefficient of sparsity was chosen as $R = 5$ while we set $\sigma = 10^{-3}$ for the variance. Lastly, we stop the iterations of all considered algorithms as soon as either

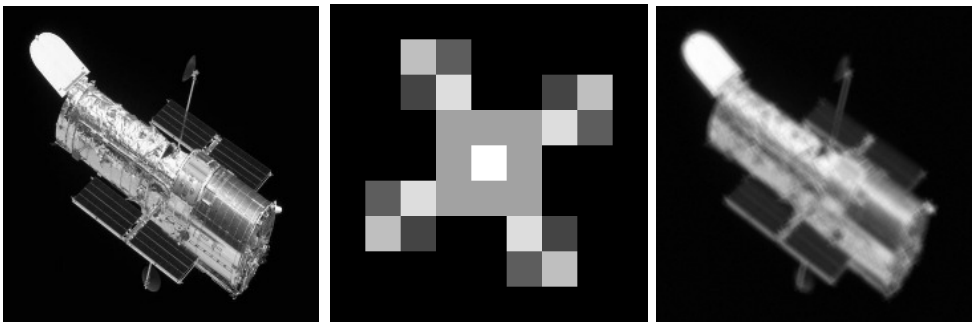
$$\frac{\|\mathbf{x}_{k+1} - \mathbf{x}_k\|_2}{\|\mathbf{x}_k\|_2} \leq 10^{-4}$$

or the maximum number of iterations, i.e., $K = 500$, is reached.

3.3.1 Example 1

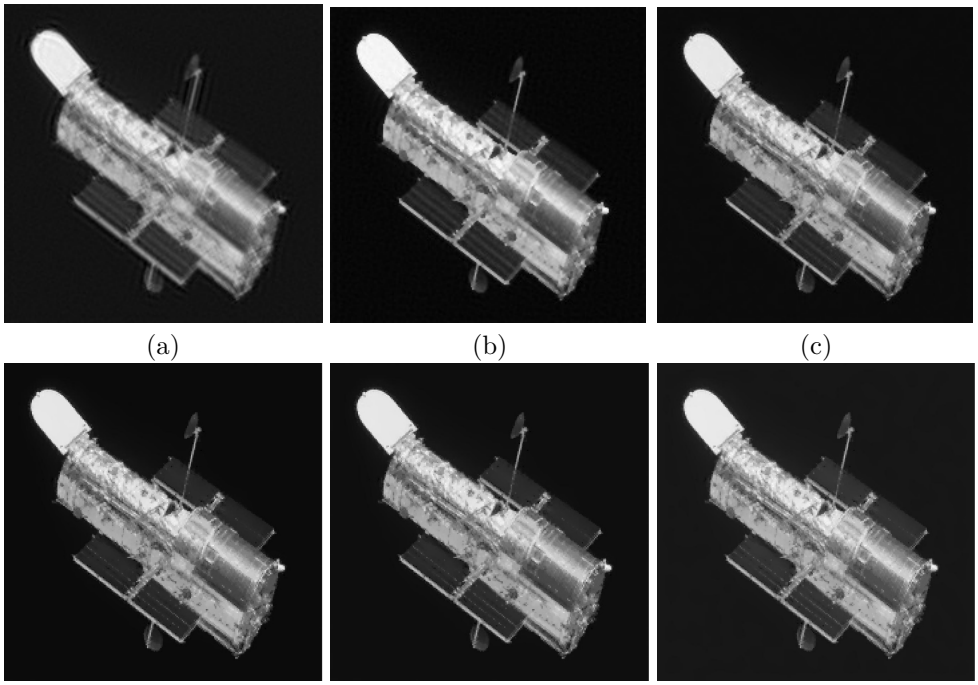
In our first example we consider a 256×256 pixels image of the Hubble space telescope. We blur it with a PSF of dimension 9×9 pixels and we add white Gaussian noise such that $\|\boldsymbol{\eta}_\delta\|_2 = 0.01\|\mathbf{b}\|_2$. We say that, in the case, the noise level is 1%. We crop the image to simulate realistic data and boundary effects; see, e.g., [82]. Since the image has a black background we impose zero boundary conditions. Figure 3.2 shows the true image, the PSF, and the observed picture.

In Figure 3.3 we report the reconstructions obtained using the considered methods. In



(a) (b) (c)

Figure 3.2: Example 1. (a) true image (238 × 238 pixels), (b) PSF (9 × 9 pixels), (c) blurred image corrupted by 1% of white Gaussian Noise (238 × 238 pixels).



(a) (b) (c)

(d) (e) (f)

Figure 3.3: Example 1. Reconstructions obtained with four different methods. (a) $\ell^2 - \ell^2$ with TV, (b) $\ell^2 - \ell^q$ with TV, (c) $\ell^2 - \ell^q$ with the graph Laplacian by Algorithm 1. (d)-(e)-(f) $\ell^2 - \ell^q$ with the fractional graph Laplacian by Algorithm 3 with fractional exponent $\alpha = 1.5,\ 1.6,\ 2$ respectively.

Figure 3.3(a) we show the approximate solution obtained with the $\ell^2 - \ell^2$ model with TV regularization, while, in Figure 3.3(b), we report the reconstruction obtained with the $\ell^2 - \ell^q$ model, like before, with L being the TV operator (Algorithm 1). In Figures 3.3(c) and 3.3(d) we consider the standard graph Laplacian (Algorithm 2), that is $\alpha = 1$, and the fractional graph Laplacian with $\alpha = 1.5$ in the $\ell^2 - \ell^q$ setting (Algorithm 3), respectively. We also report the results obtained with $\alpha = 1.6$ and $\alpha = 2$ in Figures 3.3(e) and 3.3(f). Although visual inspection seems to suggest that there are no differences with the case $\alpha = 1.5$, the computed reconstructions achieve a higher value of the PSNR and the SSIM respectively. The considered statistics for the numerical results obtained with the four different methods are reported in Table 3.1.

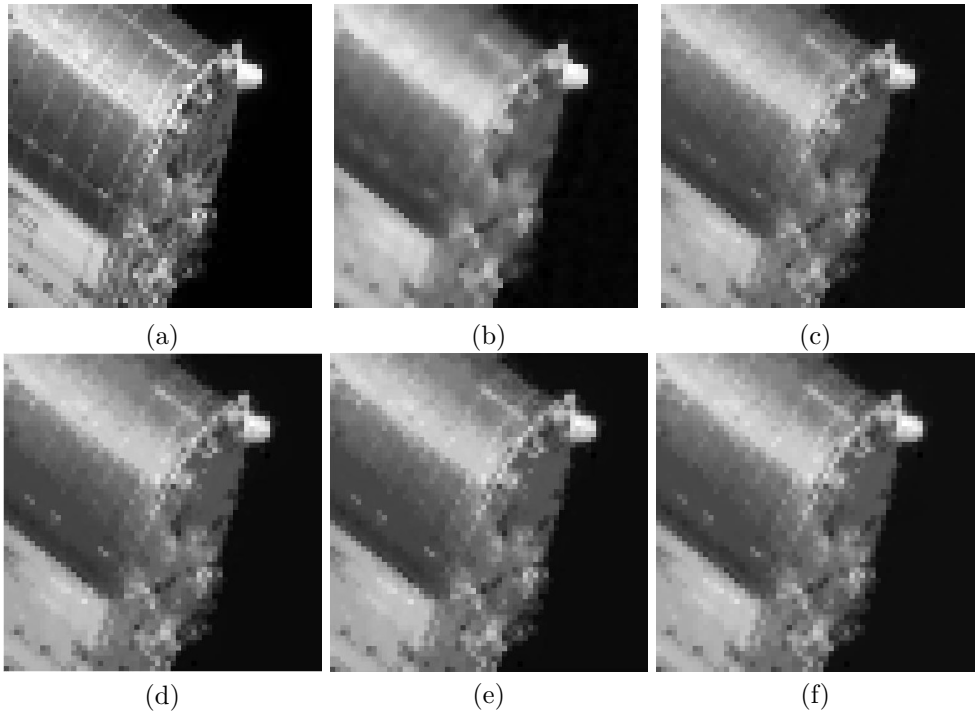


Figure 3.4: Example 1. Blow-up on the frontal part of the Hubble: (a) original image, (b) $\ell^2 - \ell^q$ TV, (c) $\ell^2 - \ell^q$ graph Laplacian, (d)-(e)-(f) $\ell^2 - \ell^q$ fractional graph Laplacian with $\alpha = 1.5, 1.6, 2$, respectively.

In Figure 3.4 we show a blow-up of the lower-right part of the image. We observe that, if one can properly choose the fractional exponent α, then it is possible to accurately reconstruct the details of the image. To this aim, we compute the value of the fractional exponent α using the residual whiteness principle (3.10).

Figures 3.5(a) and 3.5(b) depict the behavior of PSNR and SSIM, respectively, for different values of the fractional parameter α. These two measures are used to evaluate the accuracy of the computed solutions, we recall that high values of these quantities correspond to more accurate reconstructions. We highlight with a red asterisk the value of the fractional exponent computed using the residual whiteness principle. We observe that this criterion

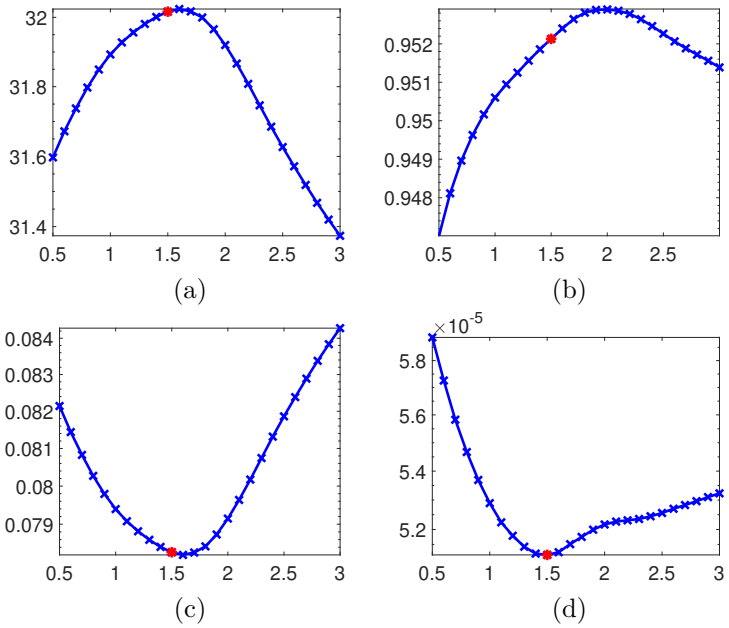


Figure 3.5: Example 1. Behavior of (a) PSNR, (b) SSIM, (c) RRE, and (d) residual whiteness $\mathcal{W}$, for different values of the fractional exponent α.

provides a fairly accurate estimate of the optimal value, i.e., the one that maximizes the two functionals. Moreover, we can observe that, in this case, the choice of a fractional exponent different from 1 improves the quality of the results with respect to $\alpha = 1$. This means that the fractional graph Laplacian can be a better regularizer than the standard graph Laplacian, provided that one can estimate the fractional exponent properly. Finally, Figures 3.5(c) and 3.5(d) report the RRE and the residual whiteness function for different values of α. As before, the red asterisk denotes the minimizer of the functional (3.10), which is also close to the value that minimizes the RRE.

3.3.2 Example 2

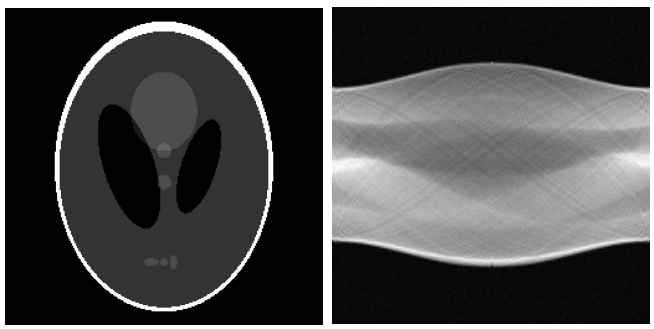


Figure 3.6: Example 2. (a) true image (128 × 128 pixels, (b) observed sinogram corrupted with 2% of white Gaussian noise (181 × 180 pixels).

The second example is a CT reconstruction problem. We construct this example using the IRtools toolbox [69]. The original image is the Shepp-Logan Phantom of dimension 128×128 pixels, it is shined with 181 parallel beams at 180 equispaced angles between 0 and π. Moreover, we perturb the sinogram $\mathbf{b} \in \mathbb{R}^{181\times180}$ with white Gaussian noise $\boldsymbol{\eta}_\delta$ with noise level 2%, i.e., $\|\boldsymbol{\eta}_\delta\|_2 = 0.02\|\mathbf{b}\|_2$. The real image and the observed sinogram are shown in Figure 3.6.

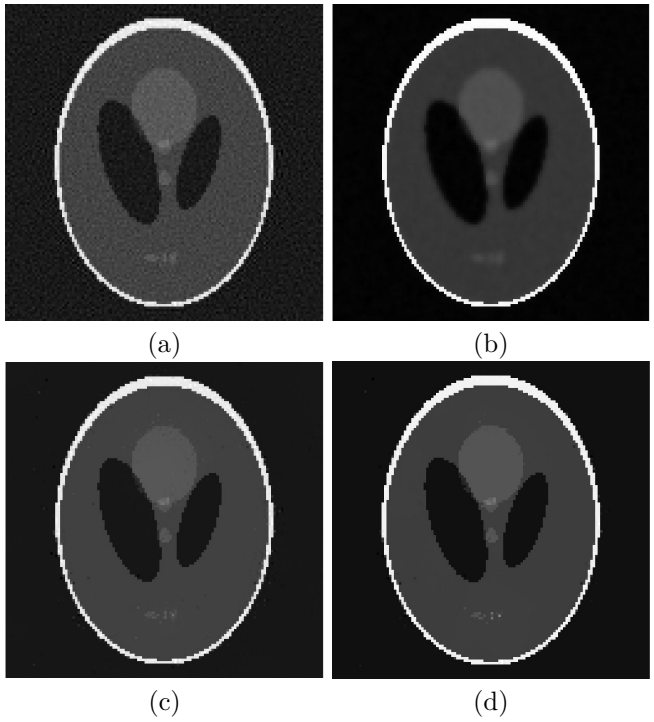


Figure 3.7: Example 2. Reconstructions obtained with four considered methods. (a) $\ell^2 - \ell^2$ with TV, (b) $\ell^2 - \ell^q$ with TV, (c) $\ell^2 - \ell^q$ with graph Laplacian, and (d) $\ell^2 - \ell^q$ with fractional graph Laplacian with $\alpha = 0.5$.

In Figure 3.7 we compare the different reconstructions obtained with the same methods we used for the deblurring example. We follow the same strategy described before for the deblurring problem and we set the parameters of Algorithm 3 as in the previous example. Despite the fact that the initial approximation provided by the $\ell^2 - \ell^2$ method, reported in Figure 3.7(a), is not very accurate, since its SSIM is only 0.6378, our proposal was able to provide an almost optimal reconstruction, reported in Figure 3.7(d), that achieves a SSIM value of 0.9929. All the artifacts present in the first approximation have been completely removed in the final reconstruction. The numerical results obtained for the four different cases can be found in Table 3.1.

In Figure 3.8 we reported the PSNR, SSIM, RRE, and values of the whiteness residual $\mathcal{W}$ for different values of α. We observe that the quality of the reconstruction strongly depends on the choice of the fractional exponent. A red asterisk highlights the value of α that minimizes $\mathcal{W}$ in (3.10). Once again, we note that this strategy provides extremely accurate

Table 3.1: Quality of the computed reconstructions for the considered methods.

Example	Method	RRE	SSIM	PSNR
Example 1	$\ell^2 - \ell^2$ TV	0.1318	0.8695	27.49
	$\ell^2 - \ell^q$ TV (Alg. 1)	0.0933	0.9114	30.49
	$\ell^2 - \ell^q$ $\alpha = 1$ (Alg. 2)	0.0857	0.9445	31.23
	$\ell^2 - \ell^q$ $\alpha = 1.5$ (Alg. 3)	0.0783	0.9521	32.02
	$\ell^2 - \ell^q$ $\alpha = 1.6$	0.0782	0.9524	32.03
	$\ell^2 - \ell^q$ $\alpha = 2$	0.0791	0.9529	31.92
Example 2	$\ell^2 - \ell^2$ TV	0.1468	0.6378	28.88
	$\ell^2 - \ell^q$ TV (Alg. 1)	0.0539	0.9593	37.58
	$\ell^2 - \ell^q$ $\alpha = 1$ (Alg. 2)	0.0560	0.9878	37.24
	$\ell^2 - \ell^q$ $\alpha = 0.5$ (Alg. 3)	0.0396	0.9926	40.27

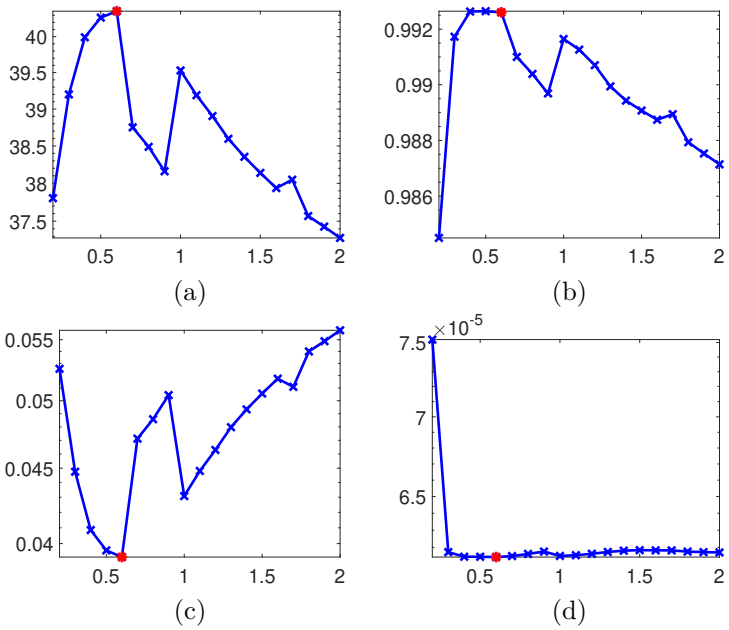


Figure 3.8: Example 2. Behavior of (a) PSNR, (b) SSIM, (c) RRE, and (d) residual whiteness $\mathcal{W}$, for different values of the fractional exponent α.

values for α.

3.4 Conclusions

In this chapter, we developed an algorithm for solving some ill-posed image reconstruction problems. The main innovation of this strategy is the use of a fractional exponent in the graph Laplacian within the regularization term to diffuse information across the graph. This significantly improves the quality of the computed reconstructions, as demonstrated in the last section, where various selected numerical examples were analyzed. Moreover, the algorithm is entirely automatic and, given a reasonably accurate estimate of the noise level corrupting the data, does not require any parameter tuning. From a theoretical standpoint, we were also able to demonstrate that the proposed method is a regularization method, and we have also analyzed its theoretical properties in detail.

Potential extensions of this method could involve its application to non-linear problems or scenarios with different types of noise, such as impulse noise or Cauchy noise. Furthermore, the fractional graph Laplacian could be integrated into other optimization schemes since the Krylov approximation described in Subsection 3.2.2 ensures that the computational cost of evaluating the fractional power is not excessively high.

Part II

Convex Optimization

Principles of convex optimization

When dealing with inverse problems, a crucial aspect is identifying the most appropriate variational model for the specific problem at hand. In the previous section, we discussed general Tikhonov-based models using the ℓ^q "norm" for regularization, where $0 < q \leq 1$. To compute approximate solutions, we employed the MM-GKS method, although, in principle, any other suitable solver could have been applied.

In what follows, we will consider a more general framework. Specifically, we aim to solve problems of the form

$$\arg\min_{\mathbf{x}\in\mathbb{R}^n} F(\mathbf{x}) = \arg\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x}) + g(\mathbf{x}), \tag{4.1}$$

where f is assumed to be differentiable, while g is convex but potentially non-smooth. Such problems frequently arise in numerous applications, including image deblurring, computed tomography, and others. For example, the `graphLa+`Ψ method can be traced back to (4.1) by simply setting

$$f(\mathbf{x}) = \frac{1}{2}\|A\mathbf{x} - \mathbf{b}^\delta\|^2 \qquad \text{and} \qquad g(\mathbf{x}) = \|\Delta_{\Psi^\delta_\Theta}\mathbf{x}\|_1.$$

In this second part we will indicate with only $\|\cdot\|$ the ℓ^2- norm without specifying the subscript index any more. The combination of a smooth term f and a non-smooth convex term g allows for a wide range of regularization strategies, making this formulation highly versatile in practical applications.

In the first part of this section, we will revisit some important definitions and key results from convex analysis that will be used throughout the remaining chapters. Understanding these fundamental concepts is essential, as they provide the theoretical foundation for optimization techniques in convex settings. Next, we will focus on the simpler scenario where F is supposed to be differentiable. In this case, we introduce and analyze the gradient descent method and further extend this by discussing its preconditioned version, which can offer improved convergence rates under certain conditions.

Finally, we will examine standard approaches for solving (4.1) in the non-smooth case, such as the well-known *Fast Iterative Soft-Thresholding Algorithm (FISTA)*. While FISTA and

similar proximal gradient methods are widely used for problems with non-smooth terms, they present notable drawbacks that limit the use of certain regularization terms, such as the TV. To address this limitation, we conclude by introducing inexact primal-dual methods, which offer a more flexible and robust framework for solving convex optimization problems.

4.1 Convex Analysis

As anticipated, in this initial section, we will briefly introduce the concept of convexity and some related results. Convexity is a fundamental concept in optimization, especially when dealing with non-smooth optimization problems.

From a notational perspective, we define:

$$\overline{\mathbb{R}} = \mathbb{R} \cup \{+\infty\}, \quad \mathbb{R}_- = \mathbb{R} \cap (-\infty, 0], \quad \mathbb{R}_+ = \mathbb{R} \cap [0, +\infty), \quad \mathbb{R}_{++} = \mathbb{R} \cap (0, +\infty).$$

Definition 4.1.1 (Relative interior)**.** *Let $C \subset \mathbb{R}^n$ be a set. We define the relative interior of C as*

$$\text{relint}(C) := \{\mathbf{x} \in C \mid \exists\ \epsilon > 0 \quad s.t. \quad B(\mathbf{x}, \epsilon) \cap \text{aff(C)} \subset C\},$$

where $B(\mathbf{x}, \epsilon)$ is the ball centered in $\mathbf{x}$ with radius ϵ while $\text{aff}(C)$ is the affine hull of C.

Definition 4.1.2 (Proper function)**.** *A function $f \colon \mathbb{R}^n \to \overline{\mathbb{R}}$ is said to be proper if and only if*

$$\exists\ \mathbf{x} \in \mathbb{R}^n \text{ such that } f(\mathbf{x}) \neq +\infty.$$

We define $\mathcal{P} := \big\{f : \mathbb{R}^n \to \overline{\mathbb{R}} : f \text{ is proper}\big\}$.

Definition 4.1.3 (Effective domain)**.** *The effective domain of a function $f \in \mathcal{P}$ is the set*

$$\text{dom}(f) := \{\mathbf{x} \in \mathbb{R}^n : f(\mathbf{x}) < \infty\}$$

Definition 4.1.4 (Coercive function)**.** *A function $f \in \mathcal{P}$ is said to be coercive if and only if*

$$\lim_{\|\mathbf{x}\| \to +\infty} f(\mathbf{x}) = +\infty.$$

Definition 4.1.5 (Lower semi-continuous function)**.** *A function $f : \mathbb{R}^n \to \overline{\mathbb{R}}$ is said to be lower semi-continuous (lsc) at the point $\mathbf{x} \in \mathbb{R}^n$ if and only if*

$$f(\mathbf{x}) \leq \liminf_{\mathbf{y} \to \mathbf{x}} f(\mathbf{y}),$$

or, equivalently, if and only if $\forall \{\mathbf{x}_k\}_{k \in \mathbb{N}} \subset \mathbb{R}^n$ that converges to $\mathbf{x}$,

$$f(\mathbf{x}) \leq \liminf_{k \to \infty} f(\mathbf{x}_k).$$

A function $f \colon \mathbb{R}^n \to \overline{\mathbb{R}}$ is said lsc if and only if it is lsc at every point $\mathbf{x} \in \mathbb{R}^n$.

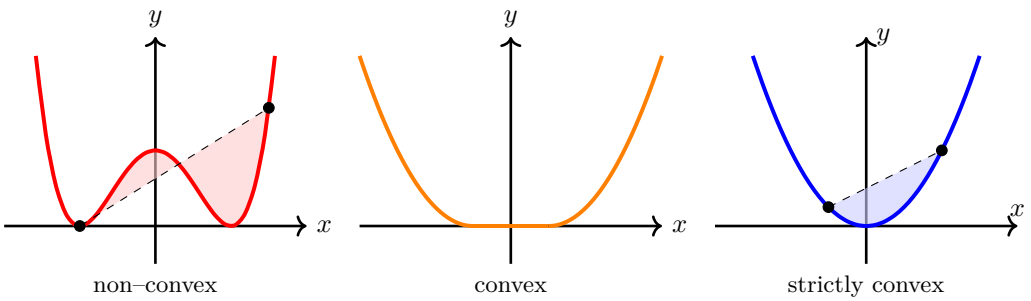


Figure 4.1: Example of non–convex, convex and strictly convex functions.

Definition 4.1.6 (Lipschitz continuous function)**.** *A function $f\colon \mathbb{R}^n \to \mathbb{R}^m$ is said to be L-Lipschitz continuous with constant $L \in \mathbb{R}_{++}$ if and only if*

$$\|f(\mathbf{x}) - f(\mathbf{y})\| \leq L\|\mathbf{x} - \mathbf{y}\| \quad \forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^n.$$

Definition 4.1.7 (Convex function)**.** *A function $f \in \mathcal{P}$ is said to be convex if and only if*

$$f(\lambda\mathbf{x} + (1-\lambda)\mathbf{y}) \leq \lambda f(\mathbf{x}) + (1-\lambda)f(\mathbf{y}) \quad \forall \mathbf{x}, \mathbf{y} \in \operatorname{dom}(f),\ \forall \lambda \in [0,1]. \tag{4.2}$$

Moreover, f is said strictly convex if and only if

$$f(\lambda\mathbf{x} + (1-\lambda)\mathbf{y}) < \lambda f(\mathbf{x}) + (1-\lambda)f(\mathbf{y}) \quad \forall \mathbf{x}, \mathbf{y} \in \operatorname{dom}(f),\ \mathbf{x} \neq \mathbf{y},\ \forall \lambda \in (0,1). \tag{4.3}$$

In Figure 4.1 we reported an example of a non–convex function, a convex function and a strictly convex function.

Definition 4.1.8 (Convex set)**.** *A set $S \subseteq \mathbb{R}^n$ is convex if for any two points $\mathbf{x}, \mathbf{y} \in S$ and any $\lambda \in [0,1]$, the point $\lambda\mathbf{x} + (1-\lambda)\mathbf{y}$ is also in S.*

Definition 4.1.9 (Epigraph)**.** *The epigraph of a function $f\colon \mathbb{R} \to \overline{\mathbb{R}}$ is the set*

$$\operatorname{epi}(f) = \{(\mathbf{x}, t) \in \mathbb{R}^n \times \mathbb{R} : f(\mathbf{x}) \leq t\}$$

Proposition 4.1.10 (Convexity of a function and convexity of its epigraph)**.** *A function $f \in \mathcal{P}$ is convex if and only if its epigraph is a convex set.*

Lemma 4.1.11 (Operations that preserve convexity)**.** *Let $f, g \in \mathcal{P}$ be convex functions, then*

1. *$f + g$ is convex;*
2. *αf is convex for $\alpha \in \mathbb{R}_{++}$;*
3. *$f(A\mathbf{x} + \mathbf{b})$ is convex for $A \in \mathbb{R}^{m \times n}$, $\mathbf{b} \in \mathbb{R}^m$;*

Corollary 4.1.12. *$f(\mathbf{x}) = \|A\mathbf{x} + \mathbf{b}\|^2$ is convex for $A \in \mathbb{R}^{m \times n}$, $\mathbf{b} \in \mathbb{R}^m$.*

Lemma 4.1.13 (Operations that preserve strict convexity)**.** *Let $f \in \mathcal{P}$ be a strictly convex function and let $g \in \mathcal{P}$ be a convex function, then*

1. *$f + g$ is strictly convex;*
2. *αf is strictly convex for $\alpha \in \mathbb{R}_{++}$;*
3. *if $A \in \mathbb{R}^{m \times n}$ is injective, $f(A\mathbf{x} + \mathbf{b})$ is strictly convex for $\mathbf{b} \in \mathbb{R}^m$;*

Definition 4.1.14 (Strongly convex function)**.** *A function $f \in \mathcal{P}$ is said strongly convex of parameter $\mu > 0$ (or μ-strongly convex) if and only if $\forall \mathbf{x}, \mathbf{y} \in \mathrm{dom}(f)$ and $\forall \lambda \in [0, 1]$ it holds*

$$f(\lambda \mathbf{x} + (1 - \lambda)\mathbf{y}) \leq \lambda f(\mathbf{x}) + (1 - \lambda) f(\mathbf{y}) - \frac{\lambda(1 - \lambda)}{2} \mu \|\mathbf{x} - \mathbf{y}\|^2.$$

We can observe that a strongly convex function is also strictly convex and that a strictly convex function is also convex.

Lemma 4.1.15 (Operations that preserve strong convexity)**.** *Let $f \in \mathcal{P}$ be a μ-strongly convex function and let $g \in \mathcal{P}$ be a convex function, then*

(i) $f + g$ is μ-strongly convex;

(ii) αf is $\alpha\mu$-strongly convex for $\alpha \in \mathbb{R}_{++}$;

(iii) Given $\mathbf{b} \in \mathbb{R}^n$, $h(\mathbf{x}) = f(\mathbf{x} + \mathbf{b})$ is μ-strongly convex.

Proposition 4.1.16 (Characterization of strongly convex functions)**.** *A function $f \in \mathcal{P}$ is μ-strongly convex if and only if $g(\mathbf{x}) = f(\mathbf{x}) - \frac{\mu}{2}\|\mathbf{x}\|^2$ is convex.*

Proposition 4.1.17 (Growth of strongly convex functions)**.** *Let $f \in \mathcal{P}$ be a μ-strongly convex function and $\bar{\mathbf{x}} \in \arg\min_{\mathbf{x} \in \mathbb{R}^n} f(\mathbf{x})$, then*

$$f(\mathbf{x}) \geq f(\bar{\mathbf{x}}) + \frac{\mu}{2}\|\mathbf{x} - \bar{\mathbf{x}}\|^2 \quad \forall \mathbf{x} \in \mathbb{R}^n.$$

In the upcoming part, we will briefly focus on smooth optimization and line search methods. For a more comprehensive and detailed analysis, we refer the reader to [116, 104, 18].

4.1.1 Smooth Optimization

As anticipated, as a preliminary step, we focus on the simplest minimization problem

$$\arg\min_{\mathbf{x} \in \mathbb{R}^n} f(\mathbf{x}), \tag{4.4}$$

where f is a differentiable function. This will allow us to introduce one of the most standard strategies for solving such problems: the *gradient descent method*. Before proceeding, it is crucial to establish the conditions under which a solution exists and how it can be characterized. To this end, we will first review some fundamental definitions and key results.

Definition 4.1.18. *Let $f : \Lambda \subseteq \mathbb{R}^n \to \mathbb{R}$. A point $\mathbf{x}^* \in \Lambda$ is said to be a* global minimum *if*

$$f(\mathbf{x}) \geq f(\mathbf{x}^*), \quad \forall \mathbf{x} \in \Lambda.$$

Definition 4.1.19. *Let $f : \Lambda \subseteq \mathbb{R}^n \to \mathbb{R}$. A point $\mathbf{x}^* \in \Lambda$ is said to be a* local minimum *if $\exists \epsilon > 0$ such as*

$$f(\mathbf{x}) \geq f(\mathbf{x}^*), \quad \forall \mathbf{x} \in B(\mathbf{x}^*, \epsilon) \cap \Lambda.$$

Moreover, if

$$f(\mathbf{x}) > f(\mathbf{x}^*) \quad \forall \mathbf{x} \in B(\mathbf{x}^*, \epsilon) \cap \Lambda,$$

then $\mathbf{x}^*$ is said to be a strictly local minimum.

Definition 4.1.20. *Given $\mathbf{x} \in \Lambda \subseteq \mathbb{R}^n$, a vector $\mathbf{d} \in \mathbb{R}^n$ is said to be an admissible direction at $\mathbf{x}$ if $\exists \bar{\alpha}$ such that*

$$\mathbf{x} + \alpha \mathbf{d} \in \Lambda \quad \forall \alpha \in (0, \bar{\alpha}).$$

The following results will give necessary conditions for a point to be a minimum.

Lemma 4.1.21. *Let $f : \Lambda \subseteq \mathbb{R}^n \to \mathbb{R}$, $f \in C^1_\Lambda$ and let $\mathbf{x}^* \in \Lambda$ a local minimum of f. Then, for all admissible direction $\mathbf{d} \in \mathbb{R}^n$ at $\mathbf{x}^*$ we have*

$$\nabla f(\mathbf{x}^*)\mathbf{d} \geq 0.$$

If $\mathbf{x}^*$ is an interior point of Λ, then

$$\nabla f(\mathbf{x}^*) = 0.$$

Lemma 4.1.22. *Let $f : \Lambda \subseteq \mathbb{R}^n \to \mathbb{R}$, $f \in C^2_\Lambda$ and let $\mathbf{x}^* \in \Lambda$ a local minimum of f. Then, for all $\mathbf{d} \in \mathbb{R}^n$ admissible direction at $\mathbf{x}^*$ such that $\nabla f(\mathbf{x}^*) = 0$, we have*

$$\mathbf{d}^T H_f(\mathbf{x}^*)\mathbf{d} \geq 0,$$

where $H_f(\mathbf{x})$ is the Hessian matrix of f.

Remark 4.1.23. *If $\mathbf{x}^* \in \Lambda$ is an interior local minimum, then $H_f(\mathbf{x}^*)$ is positive semidefinite.*

Lemma 4.1.24. *Let $f : \Lambda \subseteq \mathbb{R}^n \to \mathbb{R}$, $f \in C^2_\Lambda$ and let $\mathbf{x}^* \in \Lambda$ be an interior point. If*

(i) $\nabla f(\mathbf{x}^*) = 0$,

(ii) $H_f(\mathbf{x}^*)$ *is positive definite,*

then $\mathbf{x}^*$ is a strictly local minimum.

So far, we have not assumed the function to be convex, but only required to be sufficiently smooth. However, when convexity is introduced, we can derive additional insights about

the behavior of the function and its minimizers. For instance, numerical methods typically converge to local minima, and in the presence of convexity, we can determine when a local minimum is guaranteed to be a global one.

Lemma 4.1.25. *Let Λ be a convex set and $f : \Lambda \subseteq \mathbb{R}^n \to \mathbb{R}$, $f \in C^1_\Lambda$. Then, f is convex if and only if*

$$f(\mathbf{y}) \geq f(\mathbf{x}) + \nabla f(\mathbf{x})(\mathbf{y} - \mathbf{x}), \quad \forall \mathbf{x}, \mathbf{y} \in \Lambda. \tag{4.5}$$

Lemma 4.1.26. *Let $\Lambda \subseteq \mathbb{R}^n$ be a convex set and $f : \Lambda \subseteq \mathbb{R}^n \to \mathbb{R}$, $f \in C^2_\Lambda$. Then f is convex if and only if $H_f(\mathbf{x})$ is positive definite for all $\mathbf{x} \in \Lambda$.*

Remark 4.1.27. *If $H_f(\mathbf{x}^*)$ is positive definite then f is locally convex in $\mathbf{x}^*$. Indeed, because of continuity, there exist $\epsilon > 0$ such that for all $\mathbf{x} \in \Lambda = B_(\mathbf{x}^*, \epsilon)$ we have that $H_f(\mathbf{x})$ is positive semidefinite.*

Theorem 4.1.28. *Let $\Lambda \subseteq \mathbb{R}^n$ be a convex set with at least an interior point and let $f : \Lambda \to \mathbb{R}$, $f \in C^2_\Lambda$ and convex. Then*

1. *Local and global minimum coincide,*
2. *The set of all minimum points is convex.*

4.1.2 Line search method

One crucial aspect we did not mention yet, is how to actually compute solution of our initial problem (4.4). To this aim, two main strategies can be considered:

- Let $\mathbf{p}^{(0)} \in \mathbb{R}^n$ be what is called a descent direction for the function f. Then, it is possible to search along the direction $\mathbf{p}^{(0)}$ for a new point that achieves a lower function value. Once the minimum of f along $\mathbf{p}^{(0)}$ is found, a new descent direction $\mathbf{p}^{(1)}$ can be selected, and the same strategy can be repeated. The optimal step size α_0 along $\mathbf{p}^{(0)}$ can be determined by solving the one-dimensional minimization problem

  $$\min_\alpha f(\mathbf{x} + \alpha \mathbf{p}^{(0)}).$$

  This leads to a sequence of points given by

  $$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k \mathbf{p}^{(k)},$$

  which, under suitable assumptions, will converge to a stationary point of f.

- Consider a specific model that closely approximates the function f, for which the minimum can be computed. After determining this minimum, a new approximation can be defined in the neighborhood of the resulting point, and the same strategy is repeated. The MM strategy introduced in Section 3.1.1 follows this approach by approximating the objective function at each step with a tangent quadratic majorant.

The first class of methods are known as *line-search* methods, which we will describe in

the following pages. The second class are referred to as *trust-region* methods. To provide a comprehensive analysis of line-search methods, we begin by properly defining a descent direction for a function f. This will naturally leads to the introduction of the gradient descent methods.

Definition 4.1.29. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a continuous function and $\mathbf{x} \in \mathbb{R}^n$. A vector $\mathbf{p} \in \mathbb{R}^n$ is said to be a* descent direction *for f at $\mathbf{x}$ if $\exists\ \bar{\alpha} > 0$ such that $\forall\ 0 < \alpha < \bar{\alpha}$ we have that*

$$f(\mathbf{x} + \alpha\mathbf{p}) < f(\mathbf{x}).$$

Lemma 4.1.30. *Let $f : \mathbb{R}^n \to \mathbb{R}$, $f \in C^1$ and $\mathbf{p} \in \mathbb{R}^n$. If*

$$\nabla f(\mathbf{x})^T \mathbf{p} < 0,$$

then $\mathbf{p}$ is a descent direction for f at $\mathbf{x}$.

Remark 4.1.31. *If $\nabla f(\mathbf{x}^k)^T \mathbf{p}^{(k)} < 0$, for all $k > 0$, then at each step we can move along the direction $\mathbf{p}^{(k)}$ achieving lower function values, provided that α_k is small enough.*

There are many different possible choices for the descent direction $\mathbf{p}^{(k)}$. The simplest one is to choose

$$\mathbf{p}^{(k)} = -\nabla f(\mathbf{x}^{(k)}). \tag{4.6}$$

Indeed, thanks to Lemma (4.1.30), we know that $\mathbf{p}^{(k)}$ in (4.6) is a descent direction for f at $\mathbf{x}^{(k)}$ since

$$\nabla f(\mathbf{x}^{(k)})^T \mathbf{p}^{(k)} = -\nabla f(\mathbf{x}^{(k)})^T \nabla f(\mathbf{x}^{(k)}) = -\|\nabla f(\mathbf{x}^{(k)})\|^2 < 0.$$

More in general, given a positive definite matrix P_k, we can choose

$$\mathbf{p}^{(k)} = -P_k^{-1} \nabla f(\mathbf{x}^{(k)}). \tag{4.7}$$

This is again a descent direction for f at $\mathbf{x}^{(k)}$ since we have that

$$\nabla f(\mathbf{x}^{(k)})^T \mathbf{p}^{(k)} = -\nabla f(\mathbf{x}^{(k)})^T P_k^{-1} \nabla f(\mathbf{x}^{(k)}) = -\|\nabla f(\mathbf{x}^{(k)})\|^2_{P_k^{-1}} < 0.$$

Different choices for the matrix P_k lead to different methods:

- If $P_k = I$, then $\mathbf{p}^{(k)} = -\nabla f(\mathbf{x}^{(k)})$ and these are called *Gradient methods* and they differ in the choice of α_k. These are first order methods and they converge linearly provided some conditions on the step–length α are satisfied.
- If $P_k = H_f(\mathbf{x}^{(k)})$, then we obtain the so called *Newton methods* that are second order methods and have a quadratic rate of convergence although is just local.
- If $P_k \approx H_f(\mathbf{x}^{(k)})$, then we have the *Quasi-Newton methods*.

More generally, when the gradient of f is premultiplied by a positive definite matrix P, the resulting method is referred to as a *preconditioned gradient descent method*.

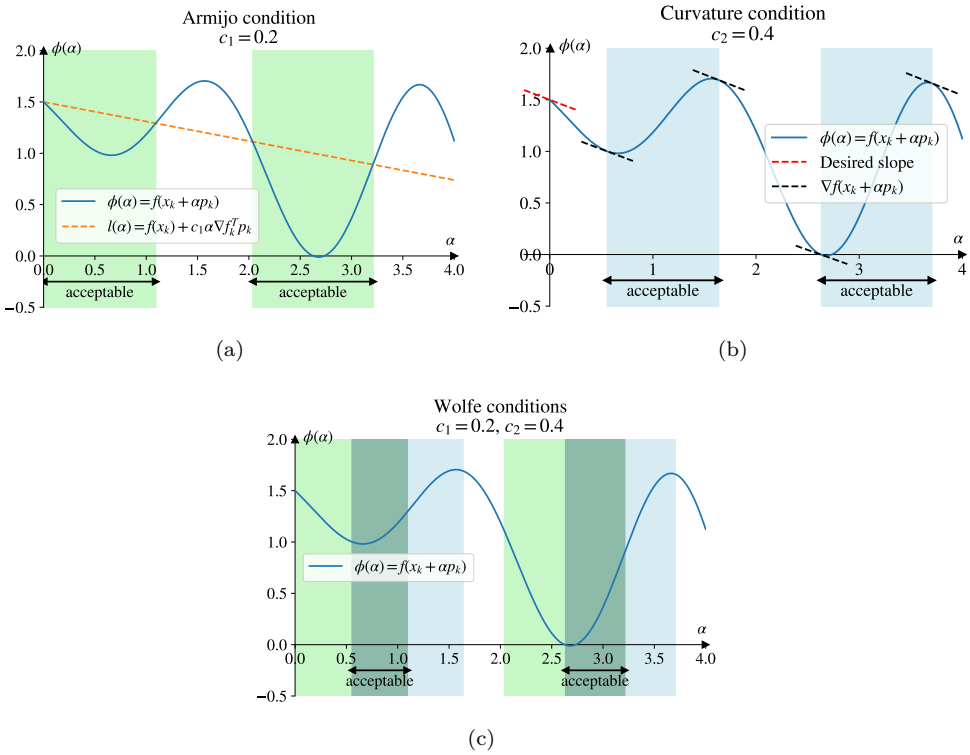


Figure 4.2: An illustration of the Armijo and curvature conditions applied to a one–dimensional problem.

The last aspect that remains to be determined is how to properly select the steplength α_k, which is a delicate issue in line search methods. Indeed, underestimating the steplength can result in slow convergence toward the minimum, while overestimating α_k may prevent the method from converging. To address these issues, stronger assumptions on the steplength are often required, such as the so-called *Armijo conditions* and *Wolfe conditions*.

Definition 4.1.32 (Armijo condition)**.** *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a differentiable function at $\mathbf{x} \in \mathbb{R}^n$. Let $\mathbf{p} \in \mathbb{R}^n$ be a descent direction at $\mathbf{x}$ for f. Fix $c_1 \in (0, 1)$. The Armijo condition is satisfied for $\alpha > 0$ if*

$$f(\mathbf{x} + \alpha \mathbf{p}) \leq f(\mathbf{x}) + \alpha c_1 \nabla f(\mathbf{x})^T \mathbf{p}. \tag{4.8}$$

Definition 4.1.33 (Wolfe conditions)**.** *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a differentiable function at $\mathbf{x} \in \mathbb{R}^n$. Let $\mathbf{p} \in \mathbb{R}^n$ be a descent direction at $\mathbf{x}$ for f and fix $c_1 \in (0, 1)$ and $c_2 \in (c_1, 1)$. The Wolfe conditions are satisfied for $\alpha > 0$ if*

$$i)\ f(\mathbf{x} + \alpha \mathbf{p}) \leq f(\mathbf{x}) + \alpha c_1 \nabla f(\mathbf{x})^T \mathbf{p}$$
$$ii)\ \nabla f(\mathbf{x} + \alpha \mathbf{p})^T \mathbf{p} \geq c_2 \nabla f(\mathbf{x})^T \mathbf{p}.$$

Note that the first condition is the Armijo condition while the second condition is called curvature condition.

Indicate with $\phi(\alpha)$ and $l(\alpha)$ the right and left hand side respectively of condition (4.8), it is possible to rewrite condition (ii) in Definition 4.1.33 as

$$\frac{\mathrm{d}}{\mathrm{d}\alpha}\phi(\alpha) \geq \frac{c_2}{c_1}\frac{\mathrm{d}}{\mathrm{d}\alpha}l(\alpha).$$

In Figure 4.2 we report the Armijo and Wolfe conditions for the estimate of a proper value of the steplength α. However, in practice, it is usually used just the Armijo condition (4.8) combined with a *backtracking* approach. The idea is to fix initial values of the steplength $\alpha_k = \bar{\alpha}$ and $\rho \in (0,1)$. Then, while $\phi(\alpha) < l(\alpha)$, we multiply α_k by ρ until condition (4.8) is not satisfied.

To conclude this brief introduction about smooth optimization and line search methods, we recall two results on the Wolfe conditions.

Theorem 4.1.34. *Let $f : \mathbb{R}^n \to \mathbb{R}$, $f \in C^1$. Let $\mathbf{p}^{(k)}$ be a descent direction at $\mathbf{x}^{(k)}$, and assume that f is bounded below along the ray $\{\mathbf{x}^{(k)} + \alpha\mathbf{p}^{(k)}, \alpha > 0\}$. Then if $0 < c_1 < c_2 < 1$, there exist intervals of steplengths satisfying the Wolfe conditions in Definition 4.1.33.*

Theorem 4.1.35. *Let $f : \mathbb{R}^n \to \mathbb{R}$ and $\mathbf{x}^{(0)} \in \mathbb{R}^n$ such that $f \in C^1_\Lambda$ where*

$$\Lambda = \{\mathbf{x} \in \mathbb{R}^n : f(\mathbf{x}) \leq f(\mathbf{x}^{(0)})\}.$$

Consider $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k\mathbf{p}^{(k)}$, where $\mathbf{p}^{(k)}$ is a descent direction for f at $\mathbf{x}^{(k)}$ and α_k satisfy the Wolfe conditions (i) and (ii) in Definition (4.1.33). Furthermore, suppose that $\nabla f(\mathbf{x})$ is L–Lipschitz continuous. Then,

$$\sum_{k=0}^{+\infty} \cos^2(\theta_k)\|\nabla f(\mathbf{x}^{(k)})\|^2 < +\infty,$$

where θ_k is the angle between the gradient of f and $\mathbf{p}^{(k)}$, i.e. it is such that

$$\cos(\theta_k) = \frac{-\nabla f(\mathbf{x}^{(k)})^T\mathbf{p}^{(k)}}{\|\mathbf{p}^{(k)}\|\|\nabla f(\mathbf{x}^{(k)})\|}.$$

Remark 4.1.36. *From Theorem 4.1.35 follows that*

$$\lim_{k\to+\infty} \cos^2(\theta_k)\|\nabla f(\mathbf{x}^{(k)})\|^2 = 0.$$

If $\cos^2(\theta_k) \geq \delta > 0$ for all k, then we have that

$$\lim_{k\to+\infty} \|\nabla f(\mathbf{x}^{(k)})\|^2 = 0,$$

that is the sequence $\{\mathbf{x}^{(k)}\}$ converges to a stationary point. To guarantee that the stationary point is a minimum we need further assumptions on the Hessian matrix.

4.2 Non–smooth Optimization

When the function g in (4.1) is nondifferentiable, the optimization techniques analysed in the previous section become inadequate. Among the several numerical strategies designed to address (4.1) in the non–smooth case, *proximal–gradient methods* [52, 54] have earned a great popularity in the last years for their simplicity and low computational cost per iteration, which make them particularly suited for large-scale optimization problems. Such algorithms deal with the functions f and g separately, by alternating a forward gradient step on the differentiable (possibly nonconvex) term f with a backward proximal step onto the convex non–differentiable term g. In particular, the backward step requires the evaluation of what is called the *proximal operator*, which is nothing else than the generalization of the notion of projection onto a convex set to a general convex function.

In what follows we will firstly recall the main notion concerning subdifferential calculus and the proximity operator followed by an overview of proximal–gradient methods and related convergence results for the convex case.

4.2.1 Subdifferential calculus

Definition 4.2.1. *The* conjugate function $f^* : \mathbb{R}^n \to \overline{\mathbb{R}}$ *of a convex function* $f : \mathbb{R}^n \to \overline{\mathbb{R}}$ *is defined as*

$$f^*(\mathbf{y}) = \sup_{\mathbf{x}\in\mathbb{R}^n} \mathbf{y}^T\mathbf{x} - f(\mathbf{x}).$$

The biconjugate function $f^{**} : \mathbb{R}^n \to \overline{\mathbb{R}}$ *of* f *is defined as* $f^{**} := (f^*)^*$*, i.e.*

$$f^{**}(y) = \sup_{\mathbf{x}\in\mathbb{R}^n} \mathbf{y}^T\mathbf{x} - f^*(\mathbf{x}).$$

Example 4.2.2. *The conjugate of the indicator function* ι_Λ *of a non empty set* $\Lambda \subseteq \mathbb{R}^n$ *is*

$$\iota^*_\Lambda(\mathbf{y}) = \sup_{\mathbf{x}\in\Lambda} \mathbf{y}^T\mathbf{x}, \quad \forall\ \mathbf{y} \in \mathbb{R}^n,$$

namely the support function of Λ*. In particular:*

- *if* Λ *is the nonnegative orthant, then* $\iota^*_{\mathbb{R}^n_+} = \iota_{\mathbb{R}^n_-}$*;*
- *if* Λ *is a linear subspace, then* $\iota^*_\Lambda = \iota_{\Lambda^\perp}$*.*

Example 4.2.3. *Consider* $f(\mathbf{x}) = \lambda\|\mathbf{x}\|$ *where* $\lambda \in \mathbb{R}_+$*. Then*

$$\begin{aligned} f^*(\mathbf{y}) &= \sup_{\mathbf{x}\in\mathbb{R}^n} \mathbf{y}^T\mathbf{x} - \lambda\|\mathbf{x}\| \\ &= \sup_{t\in\mathbb{R}_+} \left(\sup_{\|\mathbf{x}\|=1} \mathbf{y}^T(t\mathbf{x}) - t\lambda\|\mathbf{x}\| \right) \\ &= \sup_{t\in\mathbb{R}_+} t(\|\mathbf{y}\| - \lambda), \end{aligned}$$

where the last equality is obtained by recalling that $\|\mathbf{y}\| = \sup_{\|\mathbf{x}\|=1} \mathbf{y}^T\mathbf{x}$. *Therefore*

$$f^*(\mathbf{y}) = \begin{cases} 0, & \text{if } \|\mathbf{y}\| \leq \lambda \\ \infty, & \text{otherwise} \end{cases} = \iota_{B(0,\lambda)}(\mathbf{y}).$$

Example 4.2.4. *Let* $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A\mathbf{x} + \mathbf{b}^T\mathbf{x}$, *where* $A \in \mathbb{R}^{n\times n}$ *is a symmetric positive definite matrix and* $\mathbf{b} \in \mathbb{R}^n$. *Then the conjugate function of* f *is*

$$f^*(\mathbf{y}) = \sup_{\mathbf{x}\in\mathbb{R}^n} \mathbf{y}^T\mathbf{x} - f(\mathbf{x}) = \sup_{\mathbf{x}\in\mathbb{R}^n} \left[-\frac{1}{2}\mathbf{x}^T A\mathbf{x} + (\mathbf{y}-\mathbf{b})^T\mathbf{x}\right] \equiv \varphi(\mathbf{x}).$$

Since φ *is concave and differentiable, its maximum is attained in the unique point* $\mathbf{x}^* \in \mathbb{R}^n$ *such that* $\nabla\varphi(\mathbf{x}^*) = 0$, *that is* $\mathbf{x}^* = A^{-1}(\mathbf{y}-\mathbf{b})$. *Then*

$$f^*(\mathbf{y}) = \varphi(\mathbf{x}^*) = \frac{1}{2}(\mathbf{y}-\mathbf{b})^T A^{-1}(\mathbf{y}-\mathbf{b}). \tag{4.9}$$

Proposition 4.2.5. *Suppose that* $f : \mathbb{R}^n \to \overline{\mathbb{R}}$ *is given by a separable sum of convex functions, i.e.*

$$f(\mathbf{x}) = \sum_{i=1}^r f_i(\mathbf{x}^{(i)}),$$

where $f_i : \mathbb{R}^{n_i} \to \overline{\mathbb{R}}$ *is convex for* $i = 1, \ldots, r$, $\mathbf{x}^{(i)} \in \mathbb{R}^{n_i}$, *and* $\sum_{i=1}^r n_i = n$. *Then*

$$f^*(\mathbf{y}) = \sum_{i=1}^r f_i^*(\mathbf{y}^{(i)}), \quad \forall\ \mathbf{y} \in \mathbb{R}^n,\ \mathbf{y}^{(i)} \in \mathbb{R}^{n_i}, \quad i = 1, \ldots, r.$$

Proof. From the definition of conjugate function we have

$$\begin{aligned} f^*(\mathbf{y}) &= \sup_{\mathbf{x}\in\mathbb{R}^n} \left(\mathbf{y}^T\mathbf{x} - \sum_{i=1}^r f_i(\mathbf{x}^{(i)})\right) = \sup_{\mathbf{x}\in\mathbb{R}^n} \left(\sum_{i=1}^r \mathbf{y}^{(i)^T}\mathbf{x}^{(i)} - f_i(\mathbf{x}^{(i)})\right) \\ &= \sum_{i=1}^r \left(\sup_{\mathbf{x}^{(i)}\in\mathbb{R}^n} \mathbf{y}^{(i)^T}\mathbf{x}^{(i)} - f_i(\mathbf{x}^{(i)})\right) = \sum_{i=1}^r f_i^*(\mathbf{y}^{(i)}). \end{aligned}$$

□

Lemma 4.2.6. *Let* $f : \mathbb{R}^n \to \overline{\mathbb{R}}$ *be a convex function and* $f^* : \mathbb{R}^n \to \overline{\mathbb{R}}$ *its conjugate function. Then the following inequalities hold true:*

(i) (Fenchel's inequality) $f^*(\mathbf{y}) + f(\mathbf{x}) \geq \mathbf{y}^T\mathbf{x}, \quad \forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^n$.

(ii) $f(\mathbf{x}) \geq f^{**}(\mathbf{x}), \quad \forall\ \mathbf{x} \in \mathbb{R}^n$.

Proof. (i) It is an immediate consequence of the definition of conjugate function.
(ii) The Fenchel's inequality and the definition of biconjugate function lead to the following

relations:

$$f(\mathbf{x}) \geq \mathbf{y}^T\mathbf{x} - f^*(\mathbf{y})\ \forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^n \quad \iff \quad f(\mathbf{x}) \geq \sup_{\mathbf{y}\in\mathbb{R}^n} \mathbf{y}^T\mathbf{x} - f^*(\mathbf{y})\ \forall \mathbf{x} \in \mathbb{R}^n$$
$$\iff \quad f(\mathbf{x}) \geq f^{**}(\mathbf{x})\ \forall \mathbf{x} \in \mathbb{R}^n.$$

□

Theorem 4.2.7 (Biconjugate theorem)**.** *If $f\colon \mathbb{R}^n \to \overline{\mathbb{R}}$ is a lower semicontinuous and convex function then $f^{**} = f$.*

Definition 4.2.8. *[127, Definition 8.3] Let $f\colon \mathbb{R}^n \to \overline{\mathbb{R}}$ and $\mathbf{x} \in \operatorname{dom}(f)$. The* Fréchet subdifferential *of f at $\mathbf{x}$ is the set*

$$\hat{\partial} f(\mathbf{x}) = \left\{ \mathbf{v} \in \mathbb{R}^n : \liminf_{\mathbf{y}\to\mathbf{x}, \mathbf{y}\neq\mathbf{x}} \frac{1}{\|\mathbf{x}-\mathbf{y}\|}(f(\mathbf{y}) - f(\mathbf{x}) - (\mathbf{y}-\mathbf{x})^T\mathbf{v}) \geq 0 \right\}.$$

*The limiting-*subdifferential *(or simply subdifferential) of f at $\mathbf{x}$ is defined as*

$$\partial f(\mathbf{x}) = \{\mathbf{v} \in \mathbb{R}^n : \exists\ \{\mathbf{y}_k\}_{k\in\mathbb{N}} \subseteq \mathbb{R}^n,\ \mathbf{v}_k \in \hat{\partial} f(\mathbf{y}_k)\ \forall k \in \mathbb{N} \text{ such that } \mathbf{y}_k \to \mathbf{x},\ f(\mathbf{y}_k) \to f(\mathbf{x}) \text{ and } \mathbf{v}_k \to \mathbf{v}\}.$$

Finally, we define $\operatorname{dom}(\partial f) = \{\mathbf{x} \in \operatorname{dom}(f) : \partial f(\mathbf{x}) \neq \emptyset\}$.

Remark 4.2.9. *The above definition implies that $\hat{\partial} f(\mathbf{x}) \subseteq \partial f(\mathbf{x})$ for all $\mathbf{x} \in \mathbb{R}^n$, where the first set is convex and closed while the second one is closed [127, Theorem 8.6].*

Lemma 4.2.10. *Let $f : \mathbb{R}^n \to \overline{\mathbb{R}}$ be a proper, convex function. Then for any $\mathbf{x} \in \operatorname{dom}(f)$*

$$\hat{\partial} f(\mathbf{x}) = \partial f(\mathbf{x}) = \{\mathbf{v} \in \mathbb{R}^n :\ f(\mathbf{y}) \geq f(\mathbf{x}) + (\mathbf{y}-\mathbf{x})^T\mathbf{v},\quad \forall \mathbf{y} \in \mathbb{R}^n\}. \tag{4.10}$$

Proof. See [127, Proposition 8.12]. □

Remark 4.2.11. *Lemma 4.2.10 asserts that, when the function is convex, both the Fréchet and limiting-subdifferential in Definition 4.2.8 coincides with the usual subdifferential of convex analysis [126, p. 214] also known as Fenchel subdifferential.*

Example 4.2.12. *Let $f(x) = |x|$. By using Lemma 4.2.10, it is easy to see that*

$$\partial f(x) = \begin{cases} \{1\}, & \text{if } x > 0 \\ [-1, 1], & \text{if } x = 0 \\ \{-1\}, & \text{if } x < 0. \end{cases}$$

Note that the subgradient of f is an interval at the origin (see Figure 4.3).

Example 4.2.13. *Consider the indicator function ι_Λ of a non empty, convex set $\Lambda \subseteq \mathbb{R}^n$.*

By directly using equation (4.10), we have

$$\partial \iota_\Omega(\mathbf{x}) = \{\mathbf{v} \in \mathbb{R}^n : \ \mathbf{v}^T(\mathbf{y} - \mathbf{x}) \leq 0\} = N_\Lambda(\mathbf{x}),$$

where $N_\Lambda(\mathbf{x})$ denotes the normal cone to the convex set Λ at the point $\mathbf{x} \in \Lambda$ [126, p. 15].

Proposition 4.2.14 (Subgradient of a differentiable function)**.** *Let $f : \mathbb{R}^n \to \overline{\mathbb{R}}$ be a proper convex function. If f is differentiable at $\mathbf{x} \in \mathbb{R}^n$, then*

$$\partial f(\mathbf{x}) = \{\nabla f(\mathbf{x})\}. \tag{4.11}$$

Proposition 4.2.15 (Subgradient of a sum)**.** *Let $f_1, f_2 : \mathbb{R}^n \to \overline{\mathbb{R}}$ be proper convex functions. Then*

$$\partial f_1(\mathbf{x}) + \partial f_2(\mathbf{x}) \subseteq \partial(f_1 + f_2)(\mathbf{x}). \tag{4.12}$$

Moreover, if $\operatorname{relint}(\operatorname{dom}(f_1)) \cap \operatorname{relint}(\operatorname{dom}(f_2)) \neq \emptyset$, then

$$\partial f_1(\mathbf{x}) + \partial f_2(\mathbf{x}) = \partial(f_1 + f_2)(\mathbf{x}). \tag{4.13}$$

Proposition 4.2.16 (Subgradient of a scalar multiple)**.** *Let f be a proper convex function and $\lambda \in \mathbb{R}_{++}$. Then*

$$\partial(\lambda f)(\mathbf{x}) = \lambda \partial f(\mathbf{x}). \tag{4.14}$$

Proposition 4.2.17 (Subgradient of composition with linear mapping)**.** *Let $f : \mathbb{R}^n \to \overline{\mathbb{R}}$ be a proper convex function and $A \in \mathbb{R}^{m \times n}$. Then,*

$$\partial(f \circ A)(\mathbf{x}) \supset A^T \partial f(A\mathbf{x}), \quad \forall \mathbf{x} \in \operatorname{dom}(f). \tag{4.15}$$

If exist $\mathbf{x}_0$ such that $A\mathbf{x}_0 \in \operatorname{relint}(\operatorname{dom}(f))$, we have

$$\partial(f \circ A)(\mathbf{x}) = A^T \partial f(A\mathbf{x}), \quad \forall \mathbf{x} \in \operatorname{dom}(f). \tag{4.16}$$

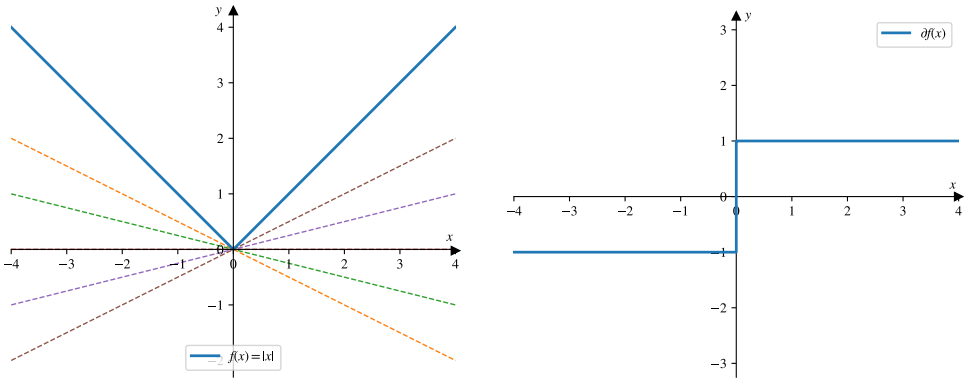
In the nondifferentiable case, it is possible to formulate the necessary optimality condition for a point to be a minimum of a function f in terms of its subdifferential. The following result is the analogous of the differentiable case.

Proposition 4.2.18. *Let $f : \mathbb{R}^n \to \overline{\mathbb{R}}$ be a proper function.*

(i). If $\mathbf{x} \in \mathbb{R}^n$ is a local minimizer of f, then $\mathbf{0} \in \partial f(\mathbf{x})$.

(ii). If f is also convex, $\mathbf{x} \in \mathbb{R}^n$ is a global minimizer if and only if $\mathbf{0} \in \partial f(\mathbf{x})$.

Proof. (i) If $\mathbf{x}$ is a local minimizer, then there exists $\rho > 0$ such that $f(\mathbf{y}) \geq f(\mathbf{x})$ for all $\mathbf{y} \in B(\mathbf{x}, \rho)$, which implies that $\mathbf{0} \in \hat{\partial} f(\mathbf{x})$. Remark 4.2.9 allows to conclude the proof.
(ii) The implication from left to right follows from item (i), while the converse is obtained by substituting $\mathbf{v} = \mathbf{0}$ in Lemma 4.2.10. □



(a) (b)

Figure 4.3: Visual representation of the subgradient of $f(x) = |x|$.

Definition 4.2.19. *A point* $\mathbf{x} \in \mathbb{R}^n$ *is* stationary *for a function* $f : \mathbb{R}^n \to \overline{\mathbb{R}}$ *if* $\mathbf{x} \in \mathrm{dom}(f)$ *and* $\mathbf{0} \in \partial f(\mathbf{x})$.

4.2.2 The proximal operator

The notion of proximal (or proximity) operator was first introduced by Moreau in [109]. Here we give its most general definition with respect to a symmetric positive definite matrix.

Definition 4.2.20. *The proximity operator associated to a function* $f \colon \mathbb{R}^n \to \overline{\mathbb{R}}$ *in the metric induced by a symmetric positive definite matrix* $P \in \mathbb{R}^{n \times n}$ *is defined as*

$$\mathrm{prox}_f^P(\mathbf{x}) = \arg\min_{\mathbf{z} \in \mathbb{R}^n} f(\mathbf{z}) + \frac{1}{2}\|\mathbf{z} - \mathbf{x}\|_P^2, \quad \forall \mathbf{x} \in \mathbb{R}^n. \tag{4.17}$$

Remark 4.2.21. *When* $P = I_n$, *we write* $\mathrm{prox}_f^{I_n} = \mathrm{prox}_f$.

Note that, in general, $\mathrm{prox}_f^P : \mathbb{R}^n \to \mathbb{R}^n$ is a multi-valued map, and it might also happen that $\mathrm{prox}_f^P(\mathbf{x}) = \emptyset$ at some point $\mathbf{x} \in \mathbb{R}^n$. However, existence and uniqueness of the proximal point may be guaranteed under convexity and lower semicontinuity assumptions.

Proposition 4.2.22. *If* $f \colon \mathbb{R}^n \to \overline{\mathbb{R}}$ *is proper, convex and lsc, then* $\mathrm{prox}_f^P(\mathbf{x})$ *exists and is unique for all* $\mathbf{x} \in \mathbb{R}^n$ *and*

$$\mathbf{y} = \mathrm{prox}_f^P(\mathbf{x}) \iff P(\mathbf{x} - \mathbf{y}) \in \partial f(\mathbf{y}). \tag{4.18}$$

Proof. The function $\varphi(\mathbf{z}) = f(\mathbf{z}) + \frac{1}{2}\|\mathbf{z} - \mathbf{x}\|_P^2$ is strictly convex and, thus, it admits at most one minimum point. Furthermore, since φ is also strongly convex, it is coercive and therefore the minimum point exists and is unique. By applying the first order optimality

condition to the convex function φ, we have

$$\begin{aligned} \mathbf{y} = \operatorname{prox}_f^P(\mathbf{x}) &\iff \mathbf{0} \in \partial\varphi(\mathbf{y}) && \text{(item } (ii) \text{ of Proposition 4.2.18)} \\ &\iff \mathbf{0} \in \partial f(\mathbf{y}) + P(\mathbf{y}-\mathbf{x}) && \text{(Proposition 4.2.15)} \\ &\iff P(\mathbf{x}-\mathbf{y}) \in \partial f(\mathbf{y}). \end{aligned}$$

□

Remark 4.2.23. *By setting* $\mathbf{w} = P(\mathbf{x}-\mathbf{y})$ *in equation (4.18), it follows that* $\mathbf{w} \in \partial f(\mathbf{y})$ *if and only if* $\mathbf{y} = \operatorname{prox}_f^P(\mathbf{y} + P^{-1}\mathbf{w})$.

Example 4.2.24. *The proximal operator of the indicator function* ι_Λ *with* $\Lambda \subseteq \mathbb{R}^n$ *non empty, closed and convex set, coincides with the projection operator onto* Λ*, indeed*

$$\operatorname{prox}_{\iota_\Lambda}^P(\mathbf{x}) = \mathcal{P}_{\Lambda,P}(\mathbf{x}) = \arg\min_{\mathbf{z}\in\Lambda} \|\mathbf{z}-\mathbf{x}\|_P^2.$$

Proximity operators are therefore a generalization of projection operators.

The proximal operator allows to give a further equivalent definition of stationary point for problem (4.1), in analogy with what already known for the differentiable case.

Proposition 4.2.25. *Let* $F : \mathbb{R}^n \to \overline{\mathbb{R}}$ *be as in problem (4.1), where* f *is a continuously differentiable function on an open set* $\Lambda_0 \supseteq \operatorname{dom}(g)$ *and* g *is proper, convex and lsc. Fix* $\alpha \in \mathbb{R}_{++}$ *and let* $P \in \mathbb{R}^{n\times n}$ *be a symmetric positive definite matrix. Then*

$$\mathbf{x}^* \text{ is stationary for } f \iff \mathbf{x}^* = \operatorname{prox}_{\alpha g}^P(\mathbf{x}^* - \alpha P^{-1}\nabla f(\mathbf{x}^*)).$$

Proof. By item (ii) of Lemma 4.2.15, we have $\partial F(\mathbf{x}^*) = \{\nabla f(\mathbf{x}^*)\} + \partial(\mathbf{x}^*)$. Therefore, the following equivalences hold:

$$\begin{aligned} \mathbf{0} \in \partial F(\mathbf{x}^*) &\iff \mathbf{0} \in \alpha\left(\{\nabla f(\mathbf{x}^*)\} + \partial g(\mathbf{x}^*)\right) \\ &\iff -\alpha\nabla f(\mathbf{x}^*) \in \partial(\alpha g)(\mathbf{x}^*). \end{aligned}$$

The thesis now follows by recalling Remark 4.2.23. □

Definition 4.2.26. *Let* $f : \mathbb{R}^n \longrightarrow \overline{\mathbb{R}}$ *be a proper, convex function. The* resolvent *of the subdifferential* ∂f *with respect to the symmetric positive definite matrix* P *is the mapping* $(I_n + P^{-1}\partial f)^{-1} : \mathbb{R}^n \to \mathbb{R}^n$ *defined as*

$$(I_n + P^{-1}\partial f)^{-1}(\mathbf{x}) = \left\{\mathbf{y} \in \mathbb{R}^n : \ \mathbf{x} \in (I_n + P^{-1}\partial f)(\mathbf{y})\right\}, \quad \forall\, \mathbf{x} \in \mathbb{R}^n.$$

Proposition 4.2.27. *Let* $f\colon \mathbb{R}^n \to \overline{\mathbb{R}}$ *be a proper, convex and lsc function and* P *a symmetric positive definite matrix. Then*

$$(I_n + P^{-1}\partial f)^{-1}(\mathbf{x}) = \operatorname{prox}_f^P(\mathbf{x}), \quad \forall\, \mathbf{x} \in \mathbb{R}^n$$

and thus $(I_n + P^{-1}\partial f)^{-1}$ *is single-valued.*

Proof. By Definition 4.2.26 of resolvent, we have

$$\begin{aligned}\mathbf{y} \in (I_n + P^{-1}\partial f)^{-1}(\mathbf{x}) &\iff \mathbf{x} \in (I_n + P^{-1}\partial f)(\mathbf{y}) = \mathbf{y} + P^{-1}\partial f(\mathbf{y}) \\ &\iff (\mathbf{x} - \mathbf{y}) \in P^{-1}\partial f(\mathbf{y}) \\ &\iff P(\mathbf{x} - \mathbf{y}) \in \partial f(\mathbf{y}) \\ &\iff \mathbf{y} = \operatorname{prox}_f^P(\mathbf{x}),\end{aligned}$$

where the last equivalence follows from (4.18). □

From now on, we denote by $\mathcal{S}_+(\mathbb{R}^n)$ the set of all $n \times n$ symmetric positive definite matrices. We use $\mathcal{D}_\varsigma \subseteq \mathcal{S}_+(\mathbb{R}^n)$ to indicate the subset of matrices whose eigenvalues belong to the interval $[\varsigma, \infty)$, and we denote by $\mathcal{D}_\varsigma^\mu \subseteq \mathcal{S}_+(\mathbb{R}^n)$ the subset of matrices whose eigenvalues belong to the interval $[\varsigma, \mu]$.

Remark 4.2.28. *If* $P \in \mathcal{D}_\varsigma^\mu$*, then we have*

$$\varsigma\|\mathbf{x}\|^2 \leq \|\mathbf{x}\|_P^2 \leq \mu\|\mathbf{x}\|^2, \quad \forall\, \mathbf{x} \in \mathbb{R}^n, \tag{4.19}$$

and likewise

$$\mu^{-1}\|\mathbf{x}\|^2 \leq \|\mathbf{x}\|_{P^{-1}}^2 \leq \varsigma^{-1}\|\mathbf{x}\|^2, \quad \forall\, \mathbf{x} \in \mathbb{R}^n. \tag{4.20}$$

Lemma 4.2.29. *Let* $f\colon \mathbb{R}^n \to \overline{\mathbb{R}}$ *be a proper, convex and lsc function and* $P \in \mathcal{D}_{\frac{1}{\mu}}^\mu$*. Then the proximal operator* prox_f^P *is Lipschitz continuous with constant* μ^2*, i.e.*

$$\|\operatorname{prox}_f^P(\mathbf{x}) - \operatorname{prox}_f^P(\tilde{\mathbf{x}})\| \leq \mu^2\|\mathbf{x} - \tilde{\mathbf{x}}\|, \qquad \forall \mathbf{x}, \tilde{\mathbf{x}} \in \mathbb{R}^n. \tag{4.21}$$

Proof. Setting $\mathbf{y} = \operatorname{prox}_f^P(\mathbf{x})$ and $\tilde{\mathbf{y}} = \operatorname{prox}_f^P(\tilde{\mathbf{x}})$, the following relations are obtained by applying (4.18) to $\mathbf{y}$ and $\tilde{\mathbf{y}}$, respectively:

$$\begin{aligned}f(\mathbf{z}) &\geq f(\mathbf{y}) + (\mathbf{z} - \mathbf{y})^T P(\mathbf{x} - \mathbf{y}) \quad \forall\, \mathbf{z} \in \mathbb{R}^n \\ f(\tilde{\mathbf{z}}) &\geq f(\tilde{\mathbf{y}}) + (\tilde{\mathbf{z}} - \tilde{\mathbf{y}})^T P(\tilde{\mathbf{z}} - \tilde{\mathbf{y}}) \quad \forall\, \tilde{\mathbf{z}} \in \mathbb{R}^n.\end{aligned}$$

Choosing $\mathbf{z} = \tilde{\mathbf{y}}$, $\tilde{\mathbf{z}} = \mathbf{y}$ and combining the two inequalities yields

$$\left(\mathbf{x} - \operatorname{prox}_f^P(\mathbf{x}) - \tilde{\mathbf{x}} + \operatorname{prox}_f^P(\tilde{\mathbf{x}})\right)^T P\left(\operatorname{prox}_f^P(\mathbf{x}) - \operatorname{prox}_f^P(\tilde{\mathbf{x}})\right) \geq 0,$$

or equivalently

$$\left\|\operatorname{prox}_f^P(\mathbf{x}) - \operatorname{prox}_f^P(\tilde{\mathbf{x}})\right\|_P^2 \leq (\mathbf{x} - \tilde{\mathbf{x}})^T P\left(\operatorname{prox}_f^P(\mathbf{x}) - \operatorname{prox}_f^P(\tilde{\mathbf{x}})\right).$$

Since $P \in \mathcal{D}^{\mu}_{\frac{1}{\mu}}$ and by using the Cauchy-Schwarz inequality, we obtain

$$\left\|\text{prox}_f^P(\mathbf{x}) - \text{prox}_f^P(\tilde{\mathbf{x}})\right\|^2 \leq \mu^2 \left\|\text{prox}_f^P(\mathbf{x}) - \text{prox}_f^P(\tilde{\mathbf{x}})\right\| \|\mathbf{x} - \tilde{\mathbf{x}}\|$$

and thus the thesis holds. □

Proposition 4.2.30. *Suppose that $f\colon \mathbb{R}^n \to \overline{\mathbb{R}}$ is given by a separable sum of convex functions, i.e.*

$$f(\mathbf{x}) = \sum_{i=1}^{r} f_i(\mathbf{x}^{(i)}),$$

where $f_i : \mathbb{R}^{n_i} \to \overline{\mathbb{R}}$ is proper, convex and lsc for $i = 1, \ldots, r$ and $\sum_{i=1}^{r} n_i = n$. Then

$$\text{prox}_f(\mathbf{x}) = \prod_{i=1}^{r} \text{prox}_{f_i}(\mathbf{x}^{(i)}) = \left(\text{prox}_{f_1}(\mathbf{x}^{(1)}), \ldots, \text{prox}_{f_r}(\mathbf{x}^{(r)})\right), \quad \forall\ \mathbf{x} \in \mathbb{R}^n.$$

Example 4.2.31 (ℓ_1-norm)**.** *Consider $f(\mathbf{x}) = \lambda\|\mathbf{x}\|_1$ with $\lambda \in \mathbb{R}_{++}$, where $\|\mathbf{x}\|_1 = \sum_{i=1}^{n} |x_i|$ is the ℓ_1-norm. Since f is a separable function in $\mathbf{x} = (x_1, \ldots, x_n)$, the proximal operator of f can be computed component-wise as*

$$(\text{prox}_f(\mathbf{x}))_i = \text{prox}_{\lambda|\cdot|}(x_i), \quad i = 1, \ldots, n.$$

From the equivalence (4.18) we have

$$\begin{aligned} y_i = \text{prox}_{\lambda|\cdot|}(x_i) &\iff x_i - y_i \in \partial(\lambda|\cdot|)(y_i) \\ &\iff y_i = x_i - w_i, \quad w_i \in \partial(\lambda|\cdot|)(y_i) \end{aligned}$$

and by computing the subdifferential $\partial(\lambda|\cdot|)$, we obtain

$$\begin{aligned} (\text{prox}_f(\mathbf{x}))_i &= \begin{cases} x_i - \lambda, & \text{if } x_i > \lambda \\ 0, & \text{if } x_i \in [-\lambda, \lambda] \\ x_i + \lambda, & \text{if } x_i < -\lambda \end{cases} \\ &= \text{sign}(x_i)\max\{|x_i| - \lambda, 0\}, \quad i = 1, \ldots, n, \\ &= (\mathcal{T}(\lambda, \mathbf{x}))_i. \end{aligned}$$

This is the so-called soft-thresholding (or shrinkage) operator.

Proposition 4.2.32 (Moreau decomposition)**.** *Given a proper, convex, lsc function $f\colon \mathbb{R}^n \to \overline{\mathbb{R}}$, its conjugate $f^*\colon \mathbb{R}^n \to \overline{\mathbb{R}}$, $\alpha \in \mathbb{R}_{++}$, and P a symmetric positive definite matrix, the following identity holds*

$$\text{prox}_{\alpha f}^P(\mathbf{x}) + \alpha P^{-1}\,\text{prox}_{\alpha^{-1}f^*}^{P^{-1}}(\alpha^{-1}P\mathbf{x}) = \mathbf{x},\ \forall \mathbf{x} \in \mathbb{R}^n.$$

Proof. The Moreau decomposition follows from the properties characterizing the subdiffer-

ential and the conjugate of a function. Indeed, given $\mathbf{x} \in \mathbb{R}^n$, let $\mathbf{y} = \operatorname{prox}_{\alpha f}^P(\mathbf{x})$. In the light of equation (4.18) and Lemma 4.2.35, we obtain

$$\begin{aligned} \mathbf{y} = \operatorname{prox}_{\alpha f}^P(\mathbf{x}) &\iff \alpha^{-1} P(\mathbf{x} - \mathbf{y}) \in \partial f(\mathbf{y}) \\ &\iff \mathbf{y} \in \partial f^*(\alpha^{-1} P(\mathbf{x} - \mathbf{y})). \end{aligned}$$

By setting $\mathbf{w} = \alpha^{-1} P(\mathbf{x} - \mathbf{y})$, the last differential inclusion becomes

$$\mathbf{x} - \alpha P^{-1} \mathbf{w} \in \partial f^*(\mathbf{w})$$

or, equivalently

$$P^{-1} \left(\alpha^{-1} P \mathbf{x} - \mathbf{w} \right) \in \partial(\alpha^{-1} f^*)(\mathbf{w}).$$

Applying again equation (4.18) yields

$$\mathbf{x} - \mathbf{y} = \alpha P^{-1} \operatorname{prox}_{\alpha^{-1} f^*}^{P^{-1}}(\alpha^{-1} P \mathbf{x}),$$

which concludes the proof. □

Example 4.2.33 (ℓ_2-norm)**.** *Let $f(\mathbf{x}) = \lambda \|x\|$ with $\lambda \in \mathbb{R}_{++}$. By the Moreau decomposition we have*

$$\operatorname{prox}_f(\mathbf{x}) = \mathbf{x} - \operatorname{prox}_{f^*}(\mathbf{x}), \quad \forall\ \mathbf{x} \in \mathbb{R}^n.$$

From Example 4.2.3, it is known that $f^* = \iota_{B(0,\lambda)}$. Thus $\operatorname{prox}_{f^*}(\mathbf{x}) = \mathcal{P}_{B(0,\lambda)}(\mathbf{x}) = \lambda \mathbf{x}/\|\mathbf{x}\|$ and in conclusion

$$\operatorname{prox}_f(\mathbf{x}) = \begin{cases} \left(1 - \frac{\lambda}{\|\mathbf{x}\|}\right) \mathbf{x}, & \text{if } \|\mathbf{x}\| > \lambda \\ \mathbf{0}, & \text{if } \|\mathbf{x}\| \leq \lambda. \end{cases}$$

Note that, when $n = 1$, the above formula reduces to the scalar soft-thresholding operation seen in Example 4.2.31.

Example 4.2.34 (Composite functions)**.** *Let $f(\mathbf{x}) = h(W\mathbf{x})$, where $W \in \mathbb{R}^{m \times n}$, $m \geq n$, is a semi-orthogonal matrix, i.e.,*

$$W^T W = \nu I_n, \quad \nu > 0,$$

and $g \colon \mathbb{R}^m \to \overline{\mathbb{R}}$ is a proper, convex, lsc function. A simple application of equation (4.18) shows that

$$\operatorname{prox}_f^P(\mathbf{x}) = \nu^{-1} W^T \operatorname{prox}_{\nu h}^P(W\mathbf{x}).$$

Hence, in this special case, when prox_h^P has a simple closed-form expression, so does prox_f^P. As an example, any function $f(\mathbf{x}) = \|W\mathbf{x}\|$ with W semi-orthogonal has an explicit formula for its proximal operator. However, it is important to note that, for a general matrix W, there is no explicit expression of prox_f^P in terms of prox_h^P and W.

We conclude this part on the proximal operator by stating a result that connects the subd-

ifferential of a function with its proximal operator.

Lemma 4.2.35. *Let $f\colon \mathbb{R}^n \to \overline{\mathbb{R}}$ be a proper convex lsc function. For all $\alpha, \beta \in \mathbb{R}{+}{+}$ the following statements are equivalent:*

(i). $\mathbf{x} = \operatorname{prox}_{\alpha f}(\mathbf{x} + \alpha\mathbf{y})$,

(ii). $\mathbf{y} \in \partial f(\mathbf{x})$,

(iii). $f(\mathbf{x}) + f^*(\mathbf{y}) = \mathbf{y}^T\mathbf{x}$,

(iv). $\mathbf{x} \in \partial f^*(\mathbf{y})$,

(v). $\mathbf{y} = \operatorname{prox}_{\beta f^*}(\mathbf{y} + \beta\mathbf{x})$.

4.2.3 Proximal gradient methods

The structure of the objective function F in (4.1) can be effectively exploited by the class of *proximal-gradient* or *forward-backward* (FB) algorithms [13, 12, 54]. These are first-order iterative methods that alternate, at each iteration, a *forward* gradient step on the differentiable part f, followed by a *backward* proximal step on the convex, non-smooth term g. The general iterative scheme of these methods is given by

$$\mathbf{x}_{k+1} = \operatorname{prox}_{\alpha_k g}(\mathbf{x}_k - \alpha_k \nabla f(\mathbf{x}_k)), \quad k = 0, 1, 2, \ldots, \tag{4.22}$$

where $\alpha_k \in \mathbb{R}_{++}$ is a scalar steplength. In the subsequent discussion and related convergence results, we will always assume that the proximal operator of g is known in its exact form. This poses a crucial drawback for proximal gradient methods, as, in real-world applications, the proximity operator of the non-smooth term is often not computable in closed form. For instance, when considering image deblurring or X-ray tomography, the widely used TV regularizer does not have a closed-form expression for its proximity operator. In the last part of this chapter, we will explore how we can eventually overcome this issue by introducing inexactness in the computation of the proximal operator.

Before delving into our overview of proximal-gradient methods, we will provide two different interpretations of the iterative method in (4.22).

- **Fixed point algorithm:** from Proposition 4.2.25, we know that a necessary condition for $\mathbf{x}^* \in \mathbb{R}^n$ to be a solution of (4.1) is

  $$\begin{aligned} \mathbf{x}^* &= \operatorname{prox}_{\alpha g}(\mathbf{x}^* - \alpha\nabla f(\mathbf{x}^*)) \\ &= (I_n + \alpha\partial g)^{-1}(I_n - \alpha\nabla f)(\mathbf{x}^*) \quad \text{(Proposition 4.2.27)}. \end{aligned}$$

  Hence $\mathbf{x}^*$ is a stationary point of (4.1) if and only $\mathbf{x}^*$ is a fixed point for the forward–backward operator $(I_n + \alpha\partial g)^{-1}(I_n - \alpha\nabla f)$. Then (4.22) can be seen as the sequence generated by the fixed point algorithm applied to $(I_n + \alpha\partial g)^{-1}(I_n - \alpha\nabla f)$.

- **Quadratic approximation:** the FB iteration (4.22) can also be interpreted as the

minimization of a reasonable local approximation of the objective function. Indeed, some algebra shows that

$$\begin{aligned}
\mathbf{x}^{(k+1)} &= \operatorname{prox}_{\alpha_k g}\Big(\mathbf{x}^{(k)} - \alpha_k \nabla f(\mathbf{x}^{(k)})\Big) \\
&= \arg\min_{\mathbf{x}\in\mathbb{R}^n} \frac{1}{2\alpha_k}\|\mathbf{x} - (\mathbf{x}^{(k)} - \alpha_k \nabla f(\mathbf{x}^{(k)}))\|^2 + g(\mathbf{x}) \\
&= \arg\min_{\mathbf{x}\in\mathbb{R}^n} \underbrace{f(\mathbf{x}^{(k)}) + \nabla f(\mathbf{x}^{(k)})^T(\mathbf{x} - \mathbf{x}^{(k)}) + \frac{1}{2\alpha_k}\|\mathbf{x} - \mathbf{x}^{(k)}\|^2}_{:=q_{\alpha_k}(\mathbf{x})} + g(\mathbf{x}) \qquad (4.23) \\
&= \arg\min_{\mathbf{x}\in\mathbb{R}^n} h_{\alpha_k}(\mathbf{x}). \qquad (4.24)
\end{aligned}$$

Thus, at each iteration, we see that the function f is being replaced by the local quadratic approximation q_{α_k}, i.e., the linearized part of f regularized by a quadratic proximal term, which measures the local error in the approximation.

When the objective function F is convex, the convergence analysis of the scheme (4.22) is strictly related to the fundamental key property

$$F(\mathbf{x}_{k+1}) \leq h_{\alpha_k}(\mathbf{x}_{k+1}), \quad \forall\, k \in \mathbb{N}. \qquad (4.25)$$

In other words, the steplength α_k must be chosen in such a way that the local approximation h_{α_k} majorizes the approximated function F at the proximal point $\mathbf{x}_{k+1}$. To this aim, we will assume that ∇f is $L-$Lipschitz continuous with $L \in \mathbb{R}_{++}$, i.e.,

$$\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\| \leq L\|\mathbf{x} - \mathbf{y}\|, \quad \forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^n. \qquad (4.26)$$

In this way, we just need to relate the steplength α_k to the Lipschitz constant L of ∇f, as suggested by the following

Lemma 4.2.36 (Descent lemma)**.** *Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be a continuously differentiable function satisfying equation (4.26). Then*

$$f(\mathbf{y}) \leq f(\mathbf{x}) + \nabla f(\mathbf{x})^T(\mathbf{y} - \mathbf{x}) + \frac{L}{2}\|\mathbf{x} - \mathbf{y}\|^2, \quad \forall\, \mathbf{x}, \mathbf{y} \in \mathbb{R}^n.$$

Proof. Let $h\colon \mathbb{R} \to \mathbb{R}$ be such that $h(t) = f\big(\mathbf{x} + t(\mathbf{y} - \mathbf{x})\big)$, for all $t \in \mathbb{R}$. The chain rule

yields $\dfrac{\mathrm{d}h(t)}{\mathrm{d}t} = \nabla f\big(\mathbf{x}+t(\mathbf{y}-\mathbf{x})\big)^T(\mathbf{y}-\mathbf{x})$. Moreover, we have

$$\begin{aligned}
f(\mathbf{y}) - f(\mathbf{x}) &= h(1) - h(0) = \int_0^1 \frac{\mathrm{d}h(t)}{\mathrm{d}t}\,\mathrm{d}t = \int_0^1 (\mathbf{y}-\mathbf{x})^T \nabla f\big(\mathbf{x}+t(\mathbf{y}-\mathbf{x})\big)\,\mathrm{d}t \\
&\leq \int_0^1 (\mathbf{y}-\mathbf{x})^T \nabla f(\mathbf{x})\,\mathrm{d}t + \left| \int_0^1 (\mathbf{y}-\mathbf{x})^T \big(\nabla f\big(\mathbf{x}+t(\mathbf{y}-\mathbf{x})\big) - \nabla f(\mathbf{x})\big)\,\mathrm{d}t \right| \\
&\leq \int_0^1 (\mathbf{y}-\mathbf{x})^T \nabla f\big(\mathbf{x}\big)\,\mathrm{d}t + \int_0^1 \|\mathbf{x}-\mathbf{y}\| \cdot \|\nabla f\big(\mathbf{x}+t(\mathbf{y}-\mathbf{x})\big) - \nabla f(\mathbf{x})\|\,\mathrm{d}t \\
&\leq (\mathbf{y}-\mathbf{x})^T \nabla f\big(\mathbf{x}\big) + \|\mathbf{x}-\mathbf{y}\| \int_0^1 Lt\|\mathbf{x}-\mathbf{y}\|\,\mathrm{d}t \\
&= (\mathbf{y}-\mathbf{x})^T \nabla f\big(\mathbf{x}\big) + \frac{L}{2}\|\mathbf{x}-\mathbf{y}\|^2.
\end{aligned}$$

□

A direct consequence of Lemma 4.2.36 is that condition (4.25) is automatically guaranteed whenever $\alpha_k \in (0, 1/L]$. When the Lipschitz constant of ∇f is known, one could simply select

$$\alpha_k = \frac{1}{L} \quad \forall k \in \mathbb{N}.$$

However, when the steplength is underestimated the speed of convergence of proximal gradient method is drastically reduced. On the other hand, if the Lipschitz constant L is not known or cannot be easily computed, such a difficulty may be overcome by employing a backtracking condition based on the descent Lemma 4.2.36. More in details, once fixed the values $L_0 \in \mathbb{R}_{++}$, $\rho > 1$, the parameter α_k is selected as:

$$\alpha_k = \frac{1}{L_k}, \tag{4.27}$$

where $L_k = \rho^{i_k} L_{k-1}$ and i_k is the smallest nonnegative integer such that

$$f(\mathbf{x}^{(k+1)}) \leq f(\mathbf{x}^{(k)}) + (\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)})^T \nabla f(\mathbf{x}^{(k)}) + \frac{L_k}{2}\|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\|^2, \tag{4.28}$$

where $\mathbf{x}^{(k+1)}$ is computed by means of (4.22) combined with (4.27). It should be noted that the above backtracking strategy is well-defined since, thanks to Lemma 4.2.36, condition (4.28) is always satisfied for $L_k \geq L$.

We conclude this part on proximal gradient methods with two particular instances of the iterative scheme (4.22) applied to the optimization problem

$$\operatorname*{arg\,min}_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x}) + \lambda\|\mathbf{x}\|_1, \tag{4.29}$$

where f represents a general differentiable data fidelity term whose gradient is $L-$Lipschitz continuous and λ is the regularization term.

Iterative Soft Thresholding Algorithm (ISTA) Firstly, recall from Example 4.2.31 that the proximity operator of the ℓ^1 norm is the soft thresholding operator $\mathcal{T}$. Therefore, when we apply the iterative forward–backward scheme (4.22) to problem (4.29), we obtain the following iterative scheme

$$\mathbf{x}_{k+1} = \mathcal{T}(\alpha_k \lambda, \mathbf{x}_k - \alpha_k \nabla f(\mathbf{x}_k)). \tag{4.30}$$

This method is known as the *Iterative Soft Thresholding Algorithm (ISTA)*. In Algorithm 4, we summarize ISTA along with the backtracking strategy described earlier.

Algorithm 4: ISTA with backtracking

Choose the starting point $\mathbf{x}_0 \in \mathrm{dom}(g)$ and let $L_{-1} \in \mathbb{R}_{++}$, $\rho > 1$.
FOR $k = 0, 1, 2, \ldots$

STEP 1. Compute the smallest nonnegative integer i_k such that $L_k = \rho^{i_k} L_{k-1}$ satisfies

$$f(\mathbf{x}_{k+1}) \leq f(\mathbf{x}_k) + (\mathbf{x}_{k+1} - \mathbf{x}_k)^T \nabla f(\mathbf{x}_k) + \frac{L_k}{2} \|\mathbf{x}_{k+1} - \mathbf{x}_k\|^2.$$

STEP 2. Compute

$$\mathbf{x}_{k+1} = \mathcal{T}\left(\frac{\lambda}{L_k}, \mathbf{x}_k - \frac{1}{L_k} \nabla f(\mathbf{x}_k)\right).$$

END

To provide a full account about the ISTA iterative scheme, we summarize some convergence results in the following

Theorem 4.2.37. *Let $F\colon \mathbb{R}^n \to \overline{\mathbb{R}}$ be as in problem (4.1), where f is convex, continuously differentiable and satisfies Assumption 4.26, and g is proper, convex and lsc. Suppose that (4.1) admits at least one solution. Let $\{\mathbf{x}_k\}_{k\in\mathbb{N}}$ be the sequence generated by Algorithm 4. Then, let L be the Lipschitz constant of ∇f, if $\alpha_k = \alpha \in (0, \frac{1}{L}]$, there holds:*

(i) the sequence $\{\mathbf{x}_k\}_{k\in\mathbb{N}}$ converges to a solution of problem (4.1).

(ii) $F(\mathbf{x}_k) - F(\mathbf{x}^*) \leq \frac{\|x_0 - \mathbf{x}^*\|}{2\alpha k} = \mathcal{O}(\frac{1}{k})$ for any solution $\mathbf{x}^*$.

Morevoer, if f or g are strongly convex with parameters μ_f, $\mu_g > 0$, and let $\mu \coloneqq \mu_f + \mu_g$, then

$$F(\mathbf{x}_k) - F(\mathbf{x}^*) + \frac{1 + \alpha\mu_g}{2\alpha} \|\mathbf{x}_k - \mathbf{x}^*\|^2 \leq \omega^k \frac{(1 + \alpha\mu_g)\|\mathbf{x}_0 - \mathbf{x}^*\|}{2\alpha},$$

where $\omega = \frac{1-\alpha\mu_f}{1+\alpha\mu_g} < 1$.

Fast ISTA (FISTA)

Though appealing for their simplicity, proximal-gradient methods often exhibit a slow speed of convergence. This is a common issue shared by all first order methods, both in differentiable and non-differentiable settings. In the literature, two significant strategies have been devised to accelerate forward-backward schemes: adding an extrapolation step and adopting a variable metric in the computation of the proximal operator. In what follows,

we will consider the first strategy applied to the ISTA algorithm, while the variable metric approach will be analyzed in the next chapter.

An acceleration strategy was first introduced by Nesterov in [114], initially suited for gradient methods and subsequently extended to proximal-gradient methods. The idea consists of adding a preliminary step, called the *extrapolation step*, to the iterative scheme under consideration. Namely, the accelerated version of a forward-backward scheme is given by

$$\begin{cases} \bar{\mathbf{x}}_k = \mathbf{x}_k + \gamma_k \left(\mathbf{x}_k - \mathbf{x}_{k-1}\right), \\ \mathbf{x}_{k+1} = \operatorname{prox}_{\alpha_k g}\left(\bar{\mathbf{x}}_k - \alpha_k \nabla f(\bar{\mathbf{x}}_k)\right), \end{cases} \tag{4.31}$$

where γ_k is the *extrapolation parameter*. If we define

$$\gamma_k^{FISTA} = \frac{t_k - 1}{t_{k+1}}, \qquad \begin{cases} t_0 = 0 \\ t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2} \end{cases}, \tag{4.32}$$

and we apply the iterative scheme (4.31) to problem (4.29) with $\gamma_k = \gamma_k^{FISTA}$, we obtain the so called *Fast Iterative Soft Thresholding Algorithm (FISTA)*

$$\begin{cases} \bar{\mathbf{x}}_k = \mathbf{x}_k + \gamma_k^{FISTA} \left(\mathbf{x}_k - \mathbf{x}_{k-1}\right), \\ \mathbf{x}_{k+1} = \mathcal{T}\left(\alpha_k \lambda, \bar{\mathbf{x}}_k - \alpha_k \nabla f(\bar{\mathbf{x}}_k)\right). \end{cases} \tag{4.33}$$

In Algorithm 5 we resumed the FISTA method implemented with the backtracking strategy proposed before.

Algorithm 5: FISTA with backtracking

Choose $\mathbf{x}_0 \in \operatorname{dom}(g)$, $L_{-1} \in \mathbb{R}_{++}$, $\rho > 1$. Set $\mathbf{x}_{-1} = \mathbf{x}_0$.
FOR $k = 0, 1, 2, \ldots$

STEP 1. Compute the extrapolated point

$$\bar{\mathbf{x}}_k = \mathbf{x}_k + \gamma_k^{FISTA} \left(\mathbf{x}_k - \mathbf{x}_{k-1}\right)$$

STEP 2. Compute the smallest nonnegative integer i_k such that $L_k = \rho^{i_k} L_{k-1}$ satisfies

$$f(\mathbf{x}_{k+1}) \leq f(\bar{\mathbf{x}}_k) + (\mathbf{x}_{k+1} - \bar{\mathbf{x}}_k)^T \nabla f(\bar{\mathbf{x}}_k) + \frac{L_k}{2} \|\mathbf{x}_{k+1} - \bar{\mathbf{x}}_k\|^2.$$

STEP 3. Compute

$$\mathbf{x}_{k+1} = \mathcal{T}\left(\frac{\lambda}{L_k}, \bar{\mathbf{x}}_k - \frac{1}{L_k} \nabla f(\bar{\mathbf{x}}_k)\right).$$

END

As done for the ISTA iterative scheme, we conclude this part by resuming a convergence result for its accelerated version.

Theorem 4.2.38. *Let $F\colon \mathbb{R}^n \to \overline{\mathbb{R}}$ be as in problem (4.1), where f is convex, continuously differentiable and satisfies Assumption 4.26, and g is proper, convex and lsc. Suppose that*

(4.1) *admits at least one solution. Let* $\{\mathbf{x}_k\}_{k\in\mathbb{N}}$ *be the sequence generated by Algorithm 5. Then, let* L *be the Lipschitz constant of* ∇f*, if* $\alpha_k = \alpha \in (0, \frac{1}{L}]$*, there holds:*

(i) the sequence $\{\mathbf{x}_k\}_{k\in\mathbb{N}}$ *converges to an optimal solution of problem* (4.1)*.*

(ii) For every $k \geq 1$*:*

$$f(\mathbf{x}_k) - f(\mathbf{x}^*) \leq \frac{2\|\mathbf{x}_0 - \mathbf{x}^*\|^2}{\alpha(k+1)^2}$$

for any optimal solution $\mathbf{x}^*$*.*

4.2.4 A Nested Primal–Dual method (NPD)

From now on and throughout the rest of the thesis, we will slightly change the notation. Specifically, we will use $\mathbb{R}^d$ and $\mathbb{R}^{d'}$ in place of $\mathbb{R}^n$ and $\mathbb{R}^m$, respectively. This change is primarily due to our intention to use a double index notation, where n will serve as one of the two indices, thereby enhancing clarity and understanding.

In the last part of the previous section, we addressed the problem of slow convergence of proximal gradient methods towards the minimum point of our initial problem (4.1) by introducing acceleration techniques such as extrapolation. However, a crucial issue with these first-order iterative methods is that the proximity operator of the non-smooth term g in (4.1) is always assumed to be known in its exact form. As noted earlier, this is a strong assumption since it cannot be satisfied in many real-world problems. In this part we will consider the accelered version of [48] proposed in [26].

In the following, we will describe a possible remedy to this issue. Since the analysis can be carried out more easily in a more general framework, we will consider the model

$$\arg\min_{\mathbf{x}\in\mathbb{R}^d} f(\mathbf{x}) + h(W\mathbf{x}), \tag{4.34}$$

where f, h, and W are defined such that:

Hypothesis 4.2.39.

(i) $f : \mathbb{R}^d \to \mathbb{R}$ *is convex and differentiable with an* L*-Lipschitz continuous gradient;*

(ii) $h : \mathbb{R}^{d'} \to \overline{\mathbb{R}}$ *is a proper convex lsc function;*

(iii) $W \in \mathbb{R}^{d'\times d}$ *and there exists* $\mathbf{x}_0$ *such that* $W\mathbf{x}_0 \in \operatorname{relint}(\operatorname{dom}(h))$*;*

(iv) Problem (4.34) *has at least one solution.*

We remark that the assumption on W is needed to guarantee that the subdifferential rule $\partial(h \circ W)(\mathbf{x}) = W^T \partial h(W\mathbf{x})$ holds, so that we can interpret the minimum points of (4.34) as solutions of appropriate variational equations, as stated below.

Lemma 4.2.40. *[48, Lemma 3.1] Under Hypothesis 4.2.39, a point* $\hat{\mathbf{x}} \in \mathbb{R}^d$ *is a solution of*

problem (4.34) *if and only if the following conditions hold*

$$\begin{cases} \nabla f(\hat{\mathbf{x}}) + W^T \hat{\mathbf{y}} = 0, \\ \hat{\mathbf{y}} = \mathrm{prox}_{\beta\alpha^{-1}h^*}(\hat{\mathbf{y}} + \beta\alpha^{-1} W\hat{\mathbf{x}}), \end{cases} \qquad \forall\ \alpha, \beta > 0. \tag{4.35}$$

From now on, the optimization problem (4.34) will serve as our reference model, and we will always assume that Hypothesis 4.2.39 is satisfied. Using the biconjugate Theorem 4.2.7, our model problem (4.34) can be equivalently reformulated as the convex–concave saddle point problem

$$\min_{\mathbf{x}\in\mathbb{R}^d} \max_{\mathbf{y}\in\mathbb{R}^{d'}} \mathcal{L}(\mathbf{x}, \mathbf{y}) \equiv f(\mathbf{x}) + \mathbf{y}^T W\mathbf{x} - h^*(\mathbf{y}), \tag{4.36}$$

where $\mathcal{L}(\mathbf{x}, \mathbf{y})$ denotes the primal-dual function. A solution of (4.36) is any point $(\hat{\mathbf{x}}, \hat{\mathbf{y}}) \in \mathbb{R}^d \times \mathbb{R}^{d'}$ such that

$$\mathcal{L}(\hat{\mathbf{x}}, \mathbf{y}) \le \mathcal{L}(\hat{\mathbf{x}}, \hat{\mathbf{y}}) \le \mathcal{L}(\mathbf{x}, \hat{\mathbf{y}}), \quad \forall\ \mathbf{x} \in \mathbb{R}^d,\ \forall\ \mathbf{y} \in \mathbb{R}^{d'}. \tag{4.37}$$

Formulation (4.36) will be particularly useful when we will state the convergence results of the so-called *Nested Primal-Dual methods (NPD)*. This class of algorithms is especially beneficial when the proximity operator of the non–smooth term $h \circ W$ cannot be computed in closed form. Indeed, if we can compute the proximal operator of the Fenchel convex conjugate h^*, then we can approximate $\mathrm{prox}_{h\circ W}(\mathbf{a})$ for a generic point $\mathbf{a} \in \mathbb{R}^d$ as stated in the following

Theorem 4.2.41. *Let f, g and W be defined as in problem* (4.34) *under Hypothesis 4.2.39. Given $\mathbf{a} \in \mathbb{R}^d$, $\alpha > 0, 0 < \beta < \frac{2}{\|W\|^2}$, and $\mathbf{y}^0 \in \mathbb{R}^{d'}$, consider the dual sequence $\{\mathbf{y}^k\}_{k\in\mathbb{N}} \subset \mathbb{R}^{d'}$ defined as follows*

$$\mathbf{y}^{k+1} = \mathrm{prox}_{\beta\alpha^{-1}h^*}\left(\mathbf{y}^k + \beta\alpha^{-1} W(\mathbf{a} - \alpha W^T \mathbf{y}^k)\right), \quad \forall k \in \mathbb{N}. \tag{4.38}$$

Then there exist $\hat{\mathbf{y}} \in \mathbb{R}^{d'}$ such that

(i) $\lim_{k\to\infty} \mathbf{y}^k = \hat{\mathbf{y}}$;

(ii) $\mathrm{prox}_{\alpha h\circ W}(\mathbf{a}) = \mathbf{a} - \alpha W^T \hat{\mathbf{y}}$.

Proof. Set $\hat{\mathbf{a}} = \mathrm{prox}_{\alpha h\circ W}(\mathbf{a})$. From the definition of proximity operator the differential inclusion characterizing the proximal point $\hat{\mathbf{a}}$ is

$$\hat{\mathbf{a}} - \mathbf{a} + \alpha W^T \hat{\mathbf{y}} = \mathbf{0}, \qquad \text{with } \hat{\mathbf{y}} \in \partial h(W\hat{\mathbf{a}}). \tag{4.39}$$

Using Lemma 4.2.35, the above variational equation can be equivalently written as

$$\hat{\mathbf{a}} = \mathbf{a} - \alpha W^T \hat{\mathbf{y}}, \quad \text{with } \hat{\mathbf{y}} = \mathrm{prox}_{\beta\alpha^{-1}h^*}\left(\hat{\mathbf{y}} + \beta\alpha^{-1} W\hat{\mathbf{a}}\right), \tag{4.40}$$

where $\alpha, \beta > 0$ and h^* is the Fenchel convex conjugate of h. By replacing the first equation

of (4.40) into the second one, we obtain

$$\hat{\mathbf{y}} = \text{prox}_{\beta\alpha^{-1}h^*}\left(\hat{\mathbf{y}} + \beta\alpha^{-1}W(\mathbf{a} - \alpha W^T\hat{\mathbf{y}})\right), \tag{4.41}$$

that is, $\hat{\mathbf{y}}$ is a fixed point of the operator $T(\mathbf{y}) = \text{prox}_{\beta\alpha^{-1}h^*}\left(\mathbf{y} + \beta\alpha^{-1}W(\mathbf{a} - \alpha W^T\mathbf{y})\right)$. Consequently, the dual sequence $\{\mathbf{y}^k\}_{k\in\mathbb{N}}$ defined in (4.38) can be interpreted as the fixed–point iteration applied to the operator T. In turn, this implies that if $\{\mathbf{y}^k\}_{k\in\mathbb{N}}$ converges, then it must converge to a fixed point $\hat{\mathbf{y}}$ of T. Finally, let us consider the minimum dual problem

$$\arg\min_{\mathbf{y}\in\mathbb{R}^m} \underbrace{\alpha^{-1}\left(\frac{1}{2\alpha}\|\mathbf{a} - \alpha W^T\mathbf{y}\|^2\right)}_{:=\Psi(\mathbf{y})} + \alpha^{-1}h^*(\mathbf{y}). \tag{4.42}$$

We see that the objective function in (4.42) is given by the sum of a convex and differentiable part $\Psi(\mathbf{y})$ plus a convex term h^* and, in addition, the gradient of Ψ can be written as

$$\nabla\Psi(\mathbf{y}) = -\alpha^{-1}W(\mathbf{a} - \alpha W^T\mathbf{y}).$$

Therefore, we have $\mathbf{y}^{k+1} = \text{prox}_{\beta\alpha^{-1}h^*}(\mathbf{y}^k - \beta\nabla\Psi(\mathbf{y}^k))$, namely the sequence $\{\mathbf{y}^k\}_{k\in\mathbb{N}}$ can be seen as the forward–backward method applied to (4.42). Since $0 < \beta < \frac{2}{\|W\|^2}$ and $\|W\|^2$ is the Lipschitz constant of $\nabla\Psi$, it follows that $\{\mathbf{y}^k\}_{k\in\mathbb{N}}$ converges to a vector $\hat{\mathbf{y}}$ satisfying (4.41), which conluced the proof. □

Remark 4.2.42. *Note that by combining items (i) and (ii) of Theorem 4.2.41 it holds that*

$$\text{prox}_{\alpha h\circ W}(\mathbf{a}) = \lim_{k\to\infty} \mathbf{a} - \alpha W^T \text{prox}_{\beta\alpha^{-1}h^*}(\mathbf{y}^k + \beta\alpha^{-1}W(\mathbf{a} - \alpha W^T\mathbf{y}^k)).$$

Hence, we can compute an approximation of the proximal operator of $h \circ W$ by a finite number of steps of a primal-dual procedure, provided that the operator $\text{prox}_{\beta\alpha^{-1}h^*}$ *and the matrix-vector product with the linear operators W and W^T are easily computable.*

Since we no longer require the exact evaluation of the proximity operator, these methods are also referred to as *inexact proximal-gradient methods*, and their iterative scheme can be defined as

$$\begin{cases} \bar{\mathbf{x}}_n = \mathbf{x}_n + \gamma_n\left(\mathbf{x}_n - \mathbf{x}_{n-1}\right), \\ \mathbf{x}_{n+1} \approx \text{prox}_{\alpha_n g}(\bar{\mathbf{x}}_n - \alpha_n\nabla f(\bar{\mathbf{x}}_n)), \end{cases} \tag{4.43}$$

where we replace the equality sign in (4.22) with the approximation symbol $\approx$.

Let $k_{\max}$ be the maximum number of inner iterations used to compute the dual sequence (4.38). Recalling that L is the Lipschitz constant of the gradient of f, the resulting NPD algorithm is summarized in Algorithm 6. In the implementation, a "warm-up strategy" is also considered, meaning that each inner primal-dual loop is "warm started" with the outcome of the previous one. As shown in [26], this strategy is sufficient to guarantee the convergence of the iterates to a solution of (4.34), provided the accuracy in the proximal evaluation is preset.

Algorithm 6: Nested Primal-Dual (NPD) method

Choose $\mathbf{x}_{-1} \in \mathbb{R}^d$, $\mathbf{x}_0 = \mathbf{x}_{-1}$, $\mathbf{y}_{-1}^{k_{\max}} \in \mathbb{R}^{d'}$, $0 < \alpha < \frac{1}{L}$, $0 < \beta < \frac{1}{\|W\|^2}$, $k_{\max} \in \mathbb{N}$.

For $n = 0, 1, \ldots$

1. Choose $\gamma_n \geq 0$ and compute the extrapolated point
$$\bar{\mathbf{x}}_n = \mathbf{x}_n + \gamma_n(\mathbf{x}_n - \mathbf{x}_{n-1}).$$
2. Set $\mathbf{y}_n^0 = \mathbf{y}_{n-1}^{k_{\max}}$.
3. Compute $k_{\max}$ primal-dual iterates:
For $k = 0, 1, \ldots, k_{\max} - 1$
$$\begin{aligned} \mathbf{x}_n^k &= \bar{\mathbf{x}}_n - \alpha \nabla f(\bar{\mathbf{x}}_n) - \alpha W^T \mathbf{y}_n^k \\ \mathbf{y}_n^{k+1} &= \operatorname{prox}_{\beta\alpha^{-1}h^*}(\mathbf{y}_n^k + \beta\alpha^{-1} W \mathbf{x}_n^k). \end{aligned}$$
4. Compute $\mathbf{x}_n^{k_{\max}} = \bar{\mathbf{x}}_n - \alpha \nabla f(\bar{\mathbf{x}}_n) - \alpha W^T \mathbf{y}_n^{k_{\max}}$.
5. Compute the next iterate as
$$\tilde{\mathbf{x}}_n = \frac{1}{k_{\max}} \sum_{k=1}^{k_{\max}} \mathbf{x}_n^k.$$

To conclude this brief overview of the NPD method, we state, without proof, a theoretical result concerning the convergence of the iterates generated by Algorithm 6. This is *Theorem 2* in [26], and its proof can be found therein.

Theorem 4.2.43 (Convergence of NPD)**.** *Suppose that f, h, and W satisfy Assumption 4.2.39. Let $\{(\mathbf{x}_n, \mathbf{y}_n^0)\}_{n\in\mathbb{N}}$ be the primal-dual sequence generated by the NPD Algorithm 6 with $\alpha_n = \alpha \in (0, \frac{1}{L}]$ and $\beta_n = \beta \in (0, \|W\|^{-2})$ for all $n \in \mathbb{N}$. Suppose also that the inertial parameters $\{\gamma_n\}_{n\in\mathbb{N}}$ satisfy*

$$\sum_{n=0}^{\infty} \gamma_n \|\mathbf{x}_n - \mathbf{x}_{n-1}\| < \infty. \tag{4.44}$$

Then, the following statements hold:

(i) the sequence $\{(\mathbf{x}_n, \mathbf{y}_n^0)\}_{n\in\mathbb{N}}$ is bounded;

(ii) the sequence $(\{\mathbf{x}_n, \mathbf{y}_n^0)\}_{n\in\mathbb{N}}$ converges to a solution of (4.36) and therefore the primal sequence $\{\mathbf{x}_n\}_{n\in\mathbb{N}}$ converges to a solution of the initial problem (4.34).

A NPD Iterated Tikhonov Method

In the previous chapter, we described the proximal gradient approach used to solve a regularized convex optimization problem of the form

$$\arg\min_{\mathbf{x}\in\mathbb{R}^d} \; f(\mathbf{x}) + h(W\mathbf{x}), \tag{5.1}$$

with f, h, and W satisfying Hypothesis 4.2.39. These methods are widely employed in imaging applications and can be accelerated by adopting variable metrics and/or extrapolation steps. However, when the proximity operator of h cannot be computed in closed form, an approximation is needed. To this aim, a nested primal–dual solver is often implemented. One crucial issue in the inexact computation of the proximal operator is the computational bottleneck that arises when increasing accuracy in the computation is required.

In what follows, we propose a nested primal–dual method for the efficient solution of regularized convex optimization problems. Our proposed method approximates a variable metric proximal–gradient step with extrapolation by performing a prefixed number of primal–dual iterates, while adjusting the steplength parameter through an appropriate backtracking procedure. Choosing a prefixed number of inner iterations allows the algorithm to maintain a low computational cost per iteration.

The first part of this chapter is devoted to the description of the proposed method and its theoretical analysis. We prove the convergence of the sequence of iterates to a solution of the considered problem under a relaxed monotonicity assumption on the scaling matrices and a shrinking condition on the extrapolation parameters. In the last part, we investigate the numerical performance of our proposed method by equipping it with a scaling matrix inspired by the Iterated Tikhonov method. The numerical results demonstrate that the combination of such scaling matrices and Nesterov-like extrapolation parameters yields effective acceleration towards the solution of the problem.

5.1 The variable metric strategy

As anticipated, in our proposal we consider proximal–gradient methods where acceleration techniques based on both variable metrics and extrapolation are combined and inexact proximal evaluations are allowed, namely

$$\begin{cases} \bar{\mathbf{x}}_n = \mathbf{x}_n + \gamma_n(\mathbf{x}_n - \mathbf{x}_{n-1}), \\ \mathbf{x}_{n+1} \approx \operatorname{prox}^{P_n}_{\alpha_n h \circ W}(\bar{\mathbf{x}}_n - \alpha_n P_n^{-1} \nabla f(\bar{\mathbf{x}}_n)), \end{cases} \qquad \forall\ n \geq 0, \tag{5.2}$$

where $\alpha_n > 0$ and $P_n \in \mathbb{R}^{d \times d}$ are, respectively, the steplength parameter and the symmetric positive definite scaling matrix, $\operatorname{prox}^{P_n}_{\alpha_n h \circ W}$ is the proximal operator of $\alpha_n h \circ W$ with respect to the norm induced by P_n, and $\gamma_n \geq 0$ is the extrapolation parameter.

The parameters α_n and P_n define the *variable metric* with respect to which the proximal–gradient point $\mathbf{x}_{n+1}$ in (5.2) is computed. While α_n represents the inverse of a local Lipschitz constant of the gradient that is dynamically computed through a backtracking procedure [12, 23], the matrix P_n aims at identifying some second order information of the smooth part of the objective function. Practical choices for P_n include the Hessian matrix or its regularized versions [99, 142], Hessian approximations based on Quasi-Newton strategies [71, 86, 89, 98], or diagonal matrices obtained by the split gradient strategy for nonnegatively constrained problems [25, 27, 94]. The extrapolation parameter γ_n, as for the case of NPD, is computed according to the prefixed sequence formerly proposed for smooth problems by Nesterov [114] and then successfully adapted to nonsmooth problems by Beck and Teboulle [12], which guarantees an optimal $\mathcal{O}(1/n^2)$ convergence rate for the function values.

Similarly to what was done for the NPD method in Section §4.2.4, the proximal operator with a metric defined by a symmetric positive definite matrix P_n can be approximated by an appropriate sequence of primal–dual iterates. This is typically achieved by means of a nested iterative solver, which is applied, at each iteration, to the minimization problem associated to the computation of the proximal–gradient point. Provided that the Fenchel convex conjugate h^* has an easy-to-compute proximal operator, then the primal–dual routine involves only the computation of $\nabla f(\mathbf{x}_n)$, $\operatorname{prox}_{h^*}$ and the matrix-vector products with W, W^T, P_n^{-1} [27, 48, 137]. The following result generalizes the one derived in Section §4.2.4 for the NPD method by taking into account the presence of the scaling matrix P_n.

Lemma 5.1.1. *Suppose that $h\colon \mathbb{R}^{d'} \to \mathbb{R} \cup \{\infty\}$ and $W \in \mathbb{R}^{d' \times d}$ satisfy Assumption 4.2.39(ii)-(iii). Let $P \in \mathbb{R}^{d \times d}$ be a symmetric positive definite matrix and $\mathbf{a} \in \mathbb{R}^d$. Choose $\alpha > 0$, $0 < \beta < 2/\|WP^{-1}W^T\|$, $\mathbf{y}^0 \in \mathbb{R}^{d'}$, define the sequence*

$$\mathbf{y}^{k+1} = \operatorname{prox}_{\beta\alpha^{-1}h^*}(\mathbf{y}^k + \beta\alpha^{-1}W(\mathbf{a} - \alpha P^{-1}W^T\mathbf{y}^k)), \quad \forall\ k \geq 0, \tag{5.3}$$

and its limit $\hat{\mathbf{y}} = \lim_{k\to\infty} \mathbf{y}^k$. Then we have

$$\operatorname{prox}^P_{\alpha h \circ W}(\mathbf{a}) = \mathbf{a} - \alpha P^{-1}W^T\hat{\mathbf{y}}. \tag{5.4}$$

Proof. Let $\hat{\mathbf{a}} = \text{prox}^P_{\alpha h \circ W}(\mathbf{a})$, which is equivalent to writing

$$\hat{\mathbf{a}} = \arg\min_{\mathbf{x}\in\mathbb{R}^d} \frac{1}{2\alpha}\|\mathbf{x}-\mathbf{a}\|_P^2 + h(W\mathbf{x}).$$

By applying Lemma 4.2.40 to the above minimization problem, we obtain the following system of variational equations

$$\begin{aligned} \hat{\mathbf{a}} = \text{prox}^P_{\alpha h\circ W}(\mathbf{a}) &\iff \begin{cases} P(\hat{\mathbf{a}}-\mathbf{a}) + \alpha W^T\hat{\mathbf{y}} = 0 \\ \hat{\mathbf{y}} = \text{prox}_{\beta\alpha^{-1}h^*}\left(\hat{\mathbf{y}} + \beta\alpha^{-1}W\hat{\mathbf{a}}\right) \end{cases} \\ &\iff \begin{cases} \hat{\mathbf{a}} = \mathbf{a} - \alpha P^{-1}W^T\hat{\mathbf{y}} \\ \hat{\mathbf{y}} = \text{prox}_{\beta\alpha^{-1}h^*}\left(\hat{\mathbf{y}} + \beta\alpha^{-1}W\hat{\mathbf{a}}\right). \end{cases} \end{aligned} \tag{5.5}$$

Consider the sequences $\{\mathbf{a}^k\}_{k\in\mathbb{N}}$ and $\{\mathbf{y}^k\}_{k\in\mathbb{N}}$ (with arbitrary $\mathbf{y}^0$) given by

$$\begin{cases} \mathbf{a}^k = \mathbf{a} - \alpha P^{-1}W^T\mathbf{y}^k \\ \mathbf{y}^{k+1} = \text{prox}_{\beta\alpha^{-1}h^*}\left(\mathbf{y}^k + \beta\alpha^{-1}W\mathbf{a}^k\right) \end{cases} \qquad k = 0, 1, \dots, \tag{5.6}$$

or equivalently

$$\mathbf{y}^{k+1} = \text{prox}_{\beta\alpha^{-1}h^*}\left(\mathbf{y}^k + \beta\alpha^{-1}W(\mathbf{a} - \alpha P^{-1}W^T\mathbf{y}^k)\right), \quad k = 0, 1, \dots.$$

In virtue of (5.5), the sequence $\{\mathbf{y}^k\}_{k\in\mathbb{N}}$ can be interpreted as a fixed-point iteration applied to the operator

$$T(\mathbf{y}) = \text{prox}_{\beta\alpha^{-1}h^*}(\mathbf{y} + \beta\alpha^{-1}W(\mathbf{a} - \alpha P^{-1}W^T\mathbf{y})). \tag{5.7}$$

By Banach fixed-point theorem, the sequence $\{\mathbf{y}^k\}_{k\in\mathbb{N}}$ converges to a fixed-point of T provided that T is a contraction. Since the operator $\text{prox}_{\beta\alpha^{-1}h^*}$ is non-expansive, we have

$$\begin{aligned} \|T(\mathbf{y}) - T(\mathbf{x})\| &\le \|\mathbf{y} - \mathbf{x} + \beta\alpha^{-1}(\alpha WP^{-1}W^T\mathbf{x} - \alpha WP^{-1}W^T\mathbf{y})\| \\ &= \|(I - \beta WP^{-1}W^T)(\mathbf{y}-\mathbf{x})\| \\ &\le \|I - \beta WP^{-1}W^T\|\|\mathbf{y}-\mathbf{x}\|, \end{aligned}$$

so T is a contraction as long as $\|I - \beta WP^{-1}W^T\| < 1$. Since P is symmetric, it holds $\|I - \beta WP^{-1}W^T\| = \rho(I - \beta WP^{-1}W^T)$, where $\rho(\cdot)$ denotes the spectral radius of a matrix, and hence T is a contraction if and only if $0 < \beta < 2/\|WP^{-1}W^T\|$. Therefore, $\{\mathbf{y}^k\}_{k\in\mathbb{N}}$ converges to a fixed-point $\hat{\mathbf{y}}$ of T, and by continuity, the sequence $\mathbf{a}^k$ also converges to the point $\hat{\mathbf{a}} = \mathbf{a} - \alpha P^{-1}W^T\hat{\mathbf{y}}$. Thus (5.5) yields the thesis. □

We can distinguish between two different approaches in the literature for computing an approximation of the proximal gradient point. On the one hand, as anticipated, one could require increasing accuracy, which means that the number of inner iterations of the nested solver grows unbounded as the outer iterations proceed [27, 71, 86, 123, 137]. The main

drawback of this approach is that the computational cost per iteration may increase in an unsustainable manner, leading to a computational bottleneck in a few iterations. On the other hand, one could compute the approximated proximal–gradient point by means of a prefixed number of inner iterations, while employing an appropriate starting condition for the inner solver; this is advantageous in order to keep the computational cost per iteration low and fixed. Some methods exploiting this latter approach are available in the literature; see e.g. [11], which considers an inexact version of the popular FISTA that corresponds to (5.2) with P_n equal to the identity matrix, γ_n selected according to Nesterov's sequence, and $\mathbf{x}_{n+1}$ computed through a prefixed number of primal–dual inner iterates with null initialization for the inner solver; [48], where the scheme (5.2) is presented without extrapolation nor variable metrics and is equipped with a prefixed number of primal–dual inner iterates, which are warm-started with the outcome of the previous outer iteration; [26], where the authors generalize the algorithm in [48] by adding an extrapolation step.

Nonetheless, to the best of our knowledge, a method of the form (5.2) equipped with a prefixed number of inner iterations that include simultaneously variable steplengths, metrics, and extrapolation, has yet to be proposed in the literature.

5.2 A nested primal–dual variable metric method

In this section, we propose and analyse our nested primal–dual variable metric method for solving problem (4.34). The resulting scheme can be considered as a variable metric proximal–gradient algorithm with extrapolation, where the proximal operator is approximated by means of a prefixed number of primal–dual steps. Note that our proposed method generalizes the one in [26], by including a backtracking procedure for the steplength and the selection of a variable scaling matrix.

5.2.1 The proposed method

We report our proposed method in Algorithm 7. It requires the choice of the prefixed number of primal–dual iterates $k_{\max} \in \mathbb{N}$, the initial guesses $\mathbf{x}_0 = \mathbf{x}_{-1} \in \mathbb{R}^d$, $\mathbf{y}_{-1}^{k_{\max}} \in \mathbb{R}^{d'}$, the parameters $\epsilon, \delta \in (0,1)$, an approximation of the Lipschitz constant $L_{-1} > 0$ and the metric parameters $\varsigma > 0$, $\alpha_{-1} = \epsilon/L_{-1}$, $P_{-1} \in \mathcal{D}_\varsigma$.

In Step 1, we choose the extrapolation parameter $\gamma_n \geq 0$ and compute the extrapolated iterate $\bar{\mathbf{x}}_n$ according to (5.8), whereas Step 2 is devoted to the choice of the scaling matrix $P_n \in \mathcal{D}_\varsigma$.

At Step 3, we initialize the approximation of the Lipschitz constant L_n as the value computed at the previous iteration, the primal steplength as $\alpha_n = \epsilon/L_n$, the dual steplength as $\beta_n = \epsilon/\|P_n^{-1}\|\|W\|^2$, and the initial dual iterate as $\mathbf{y}_n^0 = \mathbf{y}_{n-1}^{k_{\max}}$, i.e., the inner primal–dual loop is *warm-started* with the outcome of the loop at the previous iteration. Such a warm-start strategy is borrowed from the works [26, 48] and is crucial to guarantee the convergence of the algorithm.

Algorithm 7: Nested Primal-Dual Variable Metric method

Choose $k_{\max} \in \mathbb{N}$, $\mathbf{x}_{-1} \in \mathbb{R}^d$, $\mathbf{x}_0 = \mathbf{x}_{-1}$, $\mathbf{y}_{-1}^{k_{\max}} \in \mathbb{R}^{d'}$, $\epsilon \in (0,1)$, $\delta \in (0,1)$, $L_{-1} > 0$, $\varsigma > 0$, $\alpha_{-1} = \epsilon/L_{-1}$, $P_{-1} \in \mathcal{D}_\varsigma$, $\beta_{-1} = \epsilon/(\|P_{-1}^{-1}\|\|W\|^2)$.

FOR $n = 0, 1, \ldots$

1. Choose $\gamma_n \geq 0$ and compute the extrapolated point
$$\bar{\mathbf{x}}_n = \mathbf{x}_n + \gamma_n(\mathbf{x}_n - \mathbf{x}_{n-1}). \tag{5.8}$$
2. Choose $P_n \in \mathcal{D}_\varsigma$.
3. Set $i_n = 0$, $L_n = L_{n-1}$, $\alpha_n = \epsilon/L_n$, $\beta_n = \epsilon/(\|P_n^{-1}\|\|W\|^2)$, $\mathbf{y}_n^0 = \mathbf{y}_{n-1}^{k_{\max}}$.
4. Compute $k_{\max}$ primal-dual iterates:
   FOR $k = 0, 1, \ldots, k_{\max} - 1$
$$\mathbf{x}_n^k = \bar{\mathbf{x}}_n - \alpha_n P_n^{-1}\nabla f(\bar{\mathbf{x}}_n) - \alpha_n P_n^{-1} W^T \mathbf{y}_n^k \tag{5.9}$$
$$\mathbf{y}_n^{k+1} = \mathrm{prox}_{\beta_n \alpha_n^{-1} h^*}(\mathbf{y}_n^k + \beta_n \alpha_n^{-1} W \mathbf{x}_n^k). \tag{5.10}$$
5. Compute $\mathbf{x}_n^{k_{\max}} = \bar{\mathbf{x}}_n - \alpha_n P_n^{-1}\nabla f(\bar{\mathbf{x}}_n) - \alpha_n P_n^{-1} W^T \mathbf{y}_n^{k_{\max}}$.
6. Compute
$$\tilde{\mathbf{x}}_n = \frac{1}{k_{\max}} \sum_{k=1}^{k_{\max}} \mathbf{x}_n^k. \tag{5.11}$$
7. IF $f(\tilde{\mathbf{x}}_n) \leq f(\bar{\mathbf{x}}_n) + \nabla f(\bar{\mathbf{x}}_n)^T(\tilde{\mathbf{x}}_n - \bar{\mathbf{x}}_n) + \frac{L_n}{2}\|\tilde{\mathbf{x}}_n - \bar{\mathbf{x}}_n\|_{P_n}^2$
   Go to Step 8.
   ELSE
   Set $i_n = i_n + 1$, $L_n = L_{n-1}/\delta^{i_n}$, $\alpha_n = \epsilon/L_n$ and go to Step 4.
8. Set $\mathbf{x}_{n+1} = \tilde{\mathbf{x}}_n$.

Steps 4-7 define the backtracking procedure that is needed for adaptively computing the approximation of the Lipschitz constant L_n, the corresponding steplength α_n and the next iterate $\mathbf{x}_{n+1}$. First, we compute $k_{\max}$ primal–dual iterates according to Lemma 5.1.1 (Step 4). More precisely, equations (5.9)-(5.10) represent $k_{\max}$ iterations of the primal–dual method (5.3) with $\mathbf{a} = \bar{\mathbf{x}}_n - \alpha_n P_n^{-1}\nabla f(\bar{\mathbf{x}}_n)$, $\alpha = \alpha_n$, $P = P_n$ and $\beta = \beta_n$; then, in virtue of Lemma 5.1.1, the sequence $\{\mathbf{x}_n^k\}_{k=0}^{k_{\max}-1}$ can be considered as approximating the proximal–gradient point $\mathrm{prox}_{\alpha_n h \circ W}^{P_n}(\bar{\mathbf{x}}_n - \alpha_n P_n^{-1}\nabla f(\bar{\mathbf{x}}_n))$. Next, we compute an additional primal iterate $\mathbf{x}_n^{k_{\max}}$ (Step 5) and average the primal iterates $\{\mathbf{x}_n^k\}_{k=1}^{k_{\max}}$ over the number of inner iterations (Step 6). Finally, we check a backtracking condition based on Lemma 4.2.36 (Step 7); if the condition is satisfied, then we accept the average of the primal iterates as the next iterate (Step 8), otherwise we increase L_n by a factor $1/\delta$ and repeat the backtracking procedure.

Remark 5.2.1. *Note that the backtracking procedure at Steps 4-7 terminates in a finite number of steps, as the backtracking condition at Step 7 is accepted whenever $L_n \geq L/\varsigma$ (see Lemma 4.2.36). Furthermore, the sequence $\{\alpha_n\}_{n\in\mathbb{N}}$ is nonincreasing thanks to Step 3 and the backtracking procedure at Steps 4-7. Note that $\{\alpha_n\}_{n\in\mathbb{N}}$ is also bounded away from zero: indeed, either $L_n < L/\varsigma$ and hence $\alpha_n > \epsilon\varsigma/L$ for all $n \geq 0$, or otherwise let $n^* \geq 0$ be the first iteration such that $L_{n^*} \geq L/\varsigma$; in the latter case, if $n^* = 0$, then Lemma 4.2.36 and Steps 7-8 of Algorithm 7 imply $\alpha_n = \alpha_0$ for all $n \geq 0$, otherwise it must be $\delta L_{n^*} < L/\varsigma$ and*

thus $\alpha_n \geq \alpha_{n*} > (\epsilon\delta\varsigma)/L$ *for all* $n \geq 0$. *In conclusion, the following inequalities hold:*

$$0 < \min\left\{\alpha_0, \frac{\epsilon\delta\varsigma}{L}\right\} \leq \alpha_n \leq \alpha_{n-1} \leq \alpha_{-1}, \quad \forall\ n \geq 0. \tag{5.12}$$

5.2.2 Convergence analysis

In this section, we perform the convergence analysis of Algorithm 7. Under suitable conditions on the extrapolation parameter and the scaling matrix, we show that the sequence of iterates $\{\mathbf{x}_n\}_{n\in\mathbb{N}}$ converges to a solution of problem (4.34). To proceed with the convergence proof, we first present some technical lemmas that will be essential for the analysis of the proposed algorithm.

Lemma 5.2.2. *[48, Lemma 3.3] Let* $\varphi\colon \mathbb{R}^d \to \mathbb{R}\cup\{\infty\}$ *be proper, convex, and lsc, and* $\mathbf{x}, \mathbf{e} \in \mathbb{R}^d$. *Then the equality* $\mathbf{y} = \operatorname{prox}_{\varphi}(\mathbf{x}+\mathbf{e})$ *is equivalent to the following inequality:*

$$\|\mathbf{y}-\mathbf{z}\|^2 \leq \|\mathbf{x}-\mathbf{z}\|^2 - \|\mathbf{x}-\mathbf{y}\|^2 + 2\langle \mathbf{y}-\mathbf{z}, \mathbf{e}\rangle + 2\varphi(\mathbf{z}) - 2\varphi(\mathbf{y}), \quad \forall\ \mathbf{z}\in\mathbb{R}^d.$$

Lemma 5.2.3. *[130, Lemma 1] Let* $\{a_n\}_{n\in\mathbb{N}}$, $\{b_n\}_{n\in\mathbb{N}}$, $\{c_n\}_{n\in\mathbb{N}}$ *be sequences of real nonnegative numbers, with* $\{b_n\}_{n\in\mathbb{N}}$ *being a monotone nondecreasing sequence, satisfying the following recursive property*

$$a_n^2 \leq b_n + \sum_{k=1}^{n} c_k a_k, \quad \forall\ n \geq 1. \tag{5.13}$$

Then the following inequality holds:

$$a_n \leq \frac{1}{2}\sum_{k=1}^{n} c_k + \left(b_n + \left(\frac{1}{2}\sum_{k=1}^{n} c_k\right)^2\right)^{\frac{1}{2}}, \quad \forall\ n \geq 1. \tag{5.14}$$

Lemma 5.2.4. *[122] Let* $\{a_n\}_{n\in\mathbb{N}}$, $\{b_n\}_{n\in\mathbb{N}}$ *and* $\{c_n\}_{n\in\mathbb{N}}$ *be sequences of real nonnegative numbers such that* $a_{n+1} \leq (1+b_n)a_n + c_n$ *and* $\sum_{n=0}^{\infty} b_n < \infty$, $\sum_{n=0}^{\infty} c_n < \infty$. *Then, the sequence* $\{a_n\}_{n\in\mathbb{N}}$ *converges.*

For our purposes, we define the sequence of matrices $\{D_n\}_{n\in\mathbb{N}} \subseteq \mathbb{R}^{d'\times d'}$ as

$$D_n = I_{d'} - \beta_n W P_n^{-1} W^T, \quad \forall\ n \geq 0. \tag{5.15}$$

Note that, by construction, the matrix D_n are real and symmetric since $P_n \in \mathcal{D}_\varsigma$. Moreover, we can prove the following.

Lemma 5.2.5. *Let* $\{D_n\}_{n\in\mathbb{N}}$ *be the sequence of matrices defined as in (5.15) with* $\beta_n = \frac{\epsilon}{\|P_n^{-1}\|\|W\|^2}$, $\epsilon \in (0,1)$. *Then, for all* $n \geq 0$, D_n *is a real symmetric positive definite matrix and its eigenvalues are bounded away from zero by a constant independent of* n.

Proof. We just need to prove that D_n is positive definite and, since it is symmetric, it is sufficient to show that $\lambda_{\min}(D_n) > 0$. In this respect, we have that

$$\begin{aligned}\lambda_{\min}(D_n) &= 1 - \beta_n \lambda_{\max}(W P_n^{-1} W^T) \\ &= 1 - \beta_n \|W P_n^{-1} W^T\| \\ &\geq 1 - \beta_n \|P_n^{-1}\| \|W\|^2 \\ &= 1 - \epsilon > 0, \quad \forall\, n \geq 0, \end{aligned} \tag{5.16}$$

where the first inequality follows from the submultiplicative property of the spectral norm and (4.20), and the last equality is due to the choice of β_n in Algorithm 7. The final result holds because $\epsilon \in (0, 1)$. □

Based on the previous result, we can consider the norm induced by D_n, i.e.,

$$\begin{aligned}\|\mathbf{y}\|_{D_n}^2 &= \|(I - \beta_n W P_n^{-1} W^T)^{\frac{1}{2}} \mathbf{y}\|^2 \\ &= \mathbf{y}^T (I - \beta_n W P_n^{-1} W^T) \mathbf{y} \\ &= \|\mathbf{y}\|^2 - \beta_n \|P_n^{-\frac{1}{2}} W^T \mathbf{y}\|^2, \quad \forall\, \mathbf{y} \in \mathbb{R}^{d'}. \end{aligned} \tag{5.17}$$

We now discuss the assumptions needed on the inertial parameters $\{\gamma_n\}_{n\in\mathbb{N}}$ and the scaling matrices $\{P_n\}_{n\in\mathbb{N}}$ in order to ensure the convergence of Algorithm 7.

Hypothesis 5.2.6. *(i) The sequence of inertial parameters $\{\gamma_n\}_{n\in\mathbb{N}}$ of Algorithm 7 complies with the following condition*

$$\sum_{n=0}^{\infty} \gamma_n \|\mathbf{x}_n - \mathbf{x}_{n-1}\| < \infty. \tag{5.18}$$

(ii) The sequence of scaling matrices $\{P_n\}_{n\in\mathbb{N}}$ of Algorithm 7 is chosen so that

$$P_n \preceq (1 + \zeta_{n-1}) P_{n-1}, \ \forall\, n \geq 0, \quad \text{where } \zeta_{n-1} \geq 0, \ \sum_{n=0}^{\infty} \zeta_{n-1} < \infty. \tag{5.19}$$

Remark 5.2.7. *Condition (5.18) is the same assumption required for the convergence of the NPD method in Theorem 4.2.43. It can be easily implemented in practice, as it depends only on the past iterates $\mathbf{x}_n, \mathbf{x}_{n-1}$. As observed in [26], (5.18) has the practical effect of "shrinking" the inertial parameter when the iteration number n is large.*

In a similar fashion, condition (5.19) is forcing the matrices $\{P_n\}_{n\in\mathbb{N}}$ to converge to a symmetric positive definite matrix at a sufficiently fast rate controlled by the parameters sequence $\{\zeta_n\}_{n\in\mathbb{N}}$. Such a condition can be easily enforced if one selects the scaling matrix as a diagonal matrix, computed according to a specific update rule such as the Majorization–Minimization strategy or the Split-Gradient decomposition [24, 25, 49], and then constrains its elements to an interval of diminishing size.

Remark 5.2.8. *If condition (5.19) is assumed to be satisfied, then [53, Lemma 2.1] and (4.20) yield the following matrix relations*

$$((1+\zeta_{n-1})P_{n-1})^{-1} \preceq P_n^{-1} \preceq \frac{1}{\varsigma} I_d. \tag{5.20}$$

Since the spectral norm is monotone [50, Ex. 2.2-10], the above relation implies

$$\varsigma \leq \frac{1}{\|P_n^{-1}\|} \leq \frac{1+\zeta_{n-1}}{\|P_{n-1}^{-1}\|}.$$

Consequently, the parameters $\{\beta_n\}_{n\in\mathbb{N}}$ satisfy the following inequalities

$$0 < \frac{\epsilon\varsigma}{\|W\|^2} \leq \beta_n \leq (1+\zeta_{n-1})\beta_{n-1}, \quad \forall\ n \geq 0. \tag{5.21}$$

Remark 5.2.9. *Regarding the sequence of matrices $\{D_n\}_{n\in\mathbb{N}}$, we derive the following inequalities*

$$\begin{aligned}
\|\mathbf{y}\|_{D_n}^2 &= \|\mathbf{y}\|^2 - \beta_n \|W^T\mathbf{y}\|_{P_n^{-1}}^2 \\
&\leq \|\mathbf{y}\|^2 - \frac{\beta_n}{1+\zeta_{n-1}} \|W^T\mathbf{y}\|_{P_{n-1}^{-1}}^2 \\
&= \frac{(1+\zeta_{n-1})\|\mathbf{y}\|^2 - \beta_n \|W^T\mathbf{y}\|_{P_{n-1}^{-1}}^2}{1+\zeta_{n-1}} \\
&= \frac{\|\mathbf{y}\|_{D_{n-1}}^2 + \zeta_{n-1}\|\mathbf{y}\|^2 + (\beta_{n-1}-\beta_n)\|W^T\mathbf{y}\|_{P_{n-1}^{-1}}^2}{1+\zeta_{n-1}} \\
&\leq \|\mathbf{y}\|_{D_{n-1}}^2 + \zeta_{n-1}\|\mathbf{y}\|^2 + \frac{(\beta_{n-1}-\beta_n)\|W\|^2}{\varsigma}\|\mathbf{y}\|^2 \\
&\leq \left(1 + \frac{\zeta_{n-1}}{1-\epsilon} + \frac{(\beta_{n-1}-\beta_n)\|W\|^2}{\varsigma(1-\epsilon)}\right) \|\mathbf{y}\|_{D_{n-1}}^2,
\end{aligned}$$

where the first inequality follows from (5.20) and the last one is due to the combination of (4.19) and (5.16). In conclusion, there exists a sequence $\{\tilde{\zeta}_{n-1}\}_{n\in\mathbb{N}}$ such that

$$D_n \preceq \left(1+\tilde{\zeta}_{n-1}\right) D_{n-1},\ \forall\ n \geq 0, \quad \text{where } \tilde{\zeta}_{n-1} \geq \zeta_{n-1} \geq 0,\ \sum_{n=0}^{\infty} \tilde{\zeta}_{n-1} < \infty. \tag{5.22}$$

Lemma 5.2.10. *[53, Lemma 2.3] Let $\{P_n\}_{n\in\mathbb{N}} \subseteq \mathcal{D}_\varsigma^\mu$ be a sequence of scaling matrices satisfying Assumption 5.2.6(ii). Then, there exists $P \in \mathcal{D}_\varsigma$ such that $\lim_{n\to\infty} P_n = P$ pointwise.*

The following result contains some crucial descent inequalities involving the primal–dual function $\mathcal{L}$ and the iterates generated by Algorithm 7. The proof of this result is similar to the one of [26, Lemma 7], although some modifications are needed in order to address the presence of the variable metric and the backtracking procedure on the steplength α_n, which were absent in [26]. Unlike in [26], the inequalities are not given for a generic primal–dual

sequence, rather they specifically hold at the iterates obtained by averaging the primal and dual inner iterates of Algorithm 7.

Lemma 5.2.11. *Suppose that Assumption 4.2.39 holds. Let $\{(\mathbf{x}_n, \mathbf{y}_n^0)\}_{n\in\mathbb{N}}$ be the primal-dual sequence generated by Algorithm 7, and let $(\hat{\mathbf{x}}, \hat{\mathbf{y}}) \in \mathbb{R}^d \times \mathbb{R}^{d'}$ be a solution of the primal-dual problem (4.36).*

(i) Define the sequence of dual iterates $\{\tilde{\mathbf{y}}_n\}_{n\in\mathbb{N}}$ as

$$\tilde{\mathbf{y}}_n = \frac{1}{k_{\max}} \sum_{k=0}^{k_{\max}-1} \mathbf{y}_n^{k+1}, \quad \forall\ n \geq 0. \tag{5.23}$$

Then, for all $n \geq 0$ and for all $\mathbf{x}' \in \mathbb{R}^d$, we have

$$\begin{aligned} &\mathcal{L}(\mathbf{x}_{n+1}, \tilde{\mathbf{y}}_n) + \frac{1}{2\alpha_n}\|\mathbf{x}_{n+1} - \mathbf{x}'\|_{P_n}^2 \\ &\leq \mathcal{L}(\mathbf{x}', \tilde{\mathbf{y}}_n) + \frac{1}{2\alpha_n}\|\bar{\mathbf{x}}_n - \mathbf{x}'\|_{P_n}^2 - \frac{1}{2}\left(\frac{1}{\alpha_n} - L_n\right)\|\mathbf{x}_{n+1} - \bar{\mathbf{x}}_n\|_{P_n}^2. \end{aligned} \tag{5.24}$$

(ii) For all $n \geq 0$ and for all $\mathbf{y}' \in \mathbb{R}^{d'}$, we have

$$\begin{aligned} &\mathcal{L}(\mathbf{x}_{n+1}, \mathbf{y}') + \frac{\alpha_n}{2\beta_n k_{\max}}\|\mathbf{y}_{n+1}^0 - \mathbf{y}'\|_{D_n}^2 \\ &\leq \mathcal{L}(\mathbf{x}_{n+1}, \tilde{\mathbf{y}}_n) + \frac{\alpha_n}{2\beta_n k_{\max}}\|\mathbf{y}_n^0 - \mathbf{y}'\|_{D_n}^2 - \frac{1}{k_{\max}}\sum_{k=0}^{k_{\max}-1}\frac{\alpha_n}{2\beta_n}\|\mathbf{y}_n^k - \mathbf{y}_n^{k+1}\|_{D_n}^2. \end{aligned} \tag{5.25}$$

(iii) For all $n \geq 0$, for all $\mathbf{x}' \in \mathbb{R}^d$ and $\mathbf{y}' \in \mathbb{R}^{d'}$, we have

$$\begin{aligned} &\mathcal{L}(\mathbf{x}_{n+1}, \mathbf{y}') + \frac{1}{2\alpha_n}\|\mathbf{x}_{n+1} - \mathbf{x}'\|_{P_n}^2 + \frac{\alpha_n}{2\beta_n k_{\max}}\|\mathbf{y}_{n+1}^0 - \mathbf{y}'\|_{D_n}^2 \\ &\leq \mathcal{L}(\mathbf{x}', \tilde{\mathbf{y}}_n) + \frac{1}{2\alpha_n}\|\bar{\mathbf{x}}_n - \mathbf{x}'\|_{P_n}^2 + \frac{\alpha_n}{2\beta_n k_{\max}}\|\mathbf{y}_n^0 - \mathbf{y}'\|_{D_n}^2 \\ &- \frac{1}{2}\left(\frac{1}{\alpha_n} - L_n\right)\|\mathbf{x}_{n+1} - \bar{\mathbf{x}}_n\|_{P_n}^2 - \frac{1}{k_{\max}}\sum_{k=0}^{k_{\max}-1}\frac{\alpha_n}{2\beta_n}\|\mathbf{y}_n^k - \mathbf{y}_n^{k+1}\|_{D_n}^2. \end{aligned} \tag{5.26}$$

Proof. (i) We start by observing that

$$\begin{aligned} \mathbf{x}_{n+1} = \tilde{\mathbf{x}}_n &= \frac{1}{k_{\max}}\sum_{k=0}^{k_{\max}-1}\mathbf{x}_n^{k+1} \\ &= \frac{1}{k_{\max}}\sum_{k=0}^{k_{\max}-1}(\bar{\mathbf{x}}_n - \alpha_n P_n^{-1}\nabla f(\bar{\mathbf{x}}_n) - \alpha_n P_n^{-1} W^T \mathbf{y}_n^{k+1}) \\ &= \bar{\mathbf{x}}_n - \alpha_n P_n^{-1}\nabla f(\bar{\mathbf{x}}_n) - \alpha_n P_n^{-1} W^T \tilde{\mathbf{y}}_n. \end{aligned} \tag{5.27}$$

Then, we can proceed from the value $\mathcal{L}(\mathbf{x}', \tilde{\mathbf{y}}_n)$ as follows:

$$\begin{aligned}\mathcal{L}(\mathbf{x}', \tilde{\mathbf{y}}_n) &= f(\mathbf{x}') + \langle W\mathbf{x}', \tilde{\mathbf{y}}_n\rangle - h^*(\tilde{\mathbf{y}}_n)\\ &\geq f(\bar{\mathbf{x}}_n) + \langle \mathbf{x}' - \bar{\mathbf{x}}_n, \nabla f(\bar{\mathbf{x}}_n)\rangle + \langle W\mathbf{x}', \tilde{\mathbf{y}}_n\rangle - h^*(\tilde{\mathbf{y}}_n)\\ &= f(\bar{\mathbf{x}}_n) + \langle \mathbf{x}_{n+1} - \bar{\mathbf{x}}_n, \nabla f(\bar{\mathbf{x}}_n)\rangle - \langle \mathbf{x}_{n+1} - \mathbf{x}', \nabla f(\bar{\mathbf{x}}_n)\rangle\\ &+ \langle W\mathbf{x}', \tilde{\mathbf{y}}_n\rangle - h^*(\tilde{\mathbf{y}}_n)\\ &\geq f(\mathbf{x}_{n+1}) - \frac{L_n}{2}\|\mathbf{x}_{n+1} - \bar{\mathbf{x}}_n\|^2_{P_n} - \langle \mathbf{x}_{n+1} - \mathbf{x}', \nabla f(\bar{\mathbf{x}}_n)\rangle\\ &+ \langle W\mathbf{x}', \tilde{\mathbf{y}}_n\rangle - h^*(\tilde{\mathbf{y}}_n),\end{aligned}$$

where the first inequality follows from the convexity of f, and the second one is based on the backtracking condition at Step 7 of Algorithm 7. By summing and subtracting the term $W^T\tilde{\mathbf{y}}_n$ in the second argument of the scalar product $\langle \mathbf{x}_{n+1} - \mathbf{x}', \nabla f(\bar{\mathbf{x}}_n)\rangle$, we can extend the chain of inequalities as below:

$$\begin{aligned}\mathcal{L}(\mathbf{x}', \tilde{\mathbf{y}}_n) &\geq f(\mathbf{x}_{n+1}) - \frac{L_n}{2}\|\mathbf{x}_{n+1} - \bar{\mathbf{x}}_n\|^2_{P_n} - \langle \mathbf{x}_{n+1} - \mathbf{x}', \nabla f(\bar{\mathbf{x}}_n) + W^T\tilde{\mathbf{y}}_n\rangle\\ &+ \langle W\mathbf{x}', \tilde{\mathbf{y}}_n\rangle + \langle \mathbf{x}_{n+1} - \mathbf{x}', W^T\tilde{\mathbf{y}}_n\rangle - h^*(\tilde{\mathbf{y}}_n)\\ &= f(\mathbf{x}_{n+1}) + \langle \mathbf{x}_{n+1}, W^T\tilde{\mathbf{y}}_n\rangle - h^*(\tilde{\mathbf{y}}_n)\\ &- \frac{1}{\alpha_n}\langle \mathbf{x}_{n+1} - \mathbf{x}', P_n(\alpha_n P_n^{-1}\nabla f(\bar{\mathbf{x}}_n) + \alpha_n P_n^{-1}W^T\tilde{\mathbf{y}}_n)\rangle\\ &- \frac{L_n}{2}\|\mathbf{x}_{n+1} - \bar{\mathbf{x}}_n\|^2_{P_n}\\ &= \mathcal{L}(\mathbf{x}_{n+1}, \tilde{\mathbf{y}}_n) + \frac{1}{\alpha_n}\langle \mathbf{x}_{n+1} - \mathbf{x}', P_n(\mathbf{x}_{n+1} - \bar{\mathbf{x}}_n)\rangle - \frac{L_n}{2}\|\mathbf{x}_{n+1} - \bar{\mathbf{x}}_n\|^2_{P_n}\\ &= \mathcal{L}(\mathbf{x}_{n+1}, \tilde{\mathbf{y}}_n) + \frac{1}{2\alpha_n}\|\mathbf{x}_{n+1} - \mathbf{x}'\|^2_{P_n} + \frac{1}{2\alpha_n}\|\mathbf{x}_{n+1} - \bar{\mathbf{x}}_n\|^2_{P_n}\\ &- \frac{1}{2\alpha_n}\|\bar{\mathbf{x}}_n - \mathbf{x}'\|^2_{P_n} - \frac{L_n}{2}\|\mathbf{x}_{n+1} - \bar{\mathbf{x}}_n\|^2_{P_n},\end{aligned}$$

where the second equality follows from (5.27) together with the definition of $\mathcal{L}$ in (4.36), and the third one is due to the application of the three-point equality

$$\|\mathbf{a} - \mathbf{c}\|^2_{P_n} = \|\mathbf{a} - \mathbf{b}\|^2_{P_n} + \|\mathbf{b} - \mathbf{c}\|^2_{P_n} + 2\langle \mathbf{a} - \mathbf{b}, P_n(\mathbf{b} - \mathbf{c})\rangle,$$

with $\mathbf{a} = \mathbf{x}'$, $\mathbf{b} = \mathbf{x}_{n+1}$, $\mathbf{c} = \bar{\mathbf{x}}_n$. By rearranging the various terms, we get (5.24).
(ii) Starting from the value $\mathcal{L}(\mathbf{x}_{n+1}, \mathbf{y}_n^{k+1})$ and applying Lemma 5.2.2 with $\varphi = \beta_n\alpha_n^{-1}h^*$,

$\mathbf{v} = \mathbf{y}_n^{k+1}$, $\mathbf{u} = \mathbf{y}_n^k$, $e = \beta\alpha_n^{-1} W\mathbf{x}_n^k$ and $z = \mathbf{y}'$, we get

$$\begin{aligned}
\mathcal{L}(\mathbf{x}_{n+1}, \mathbf{y}_n^{k+1}) &= f(\mathbf{x}_{n+1}) + \langle W\mathbf{x}_{n+1}, \mathbf{y}_n^{k+1}\rangle - h^*(\mathbf{y}_n^{k+1}) \\
&\geq f(\mathbf{x}_{n+1}) + \langle W\mathbf{x}_{n+1}, \mathbf{y}_n^{k+1}\rangle - h^*(\mathbf{y}') - \langle \mathbf{y}_n^{k+1} - \mathbf{y}', W\mathbf{x}_n^k\rangle \\
&+ \frac{\alpha_n}{2\beta_n}\|\mathbf{y}_n^{k+1} - \mathbf{y}'\|^2 - \frac{\alpha_n}{2\beta_n}\|\mathbf{y}_n^k - \mathbf{y}'\|^2 + \frac{\alpha_n}{2\beta_n}\|\mathbf{y}_n^k - \mathbf{y}_n^{k+1}\|^2 \\
&= \mathcal{L}(\mathbf{x}_{n+1}, \mathbf{y}') + \langle \mathbf{y}_n^{k+1} - \mathbf{y}', W(\mathbf{x}_{n+1} - \mathbf{x}_n^{k+1})\rangle \\
&- \langle \mathbf{y}_n^{k+1} - \mathbf{y}', W(\mathbf{x}_n^k - \mathbf{x}_n^{k+1})\rangle \\
&+ \frac{\alpha_n}{2\beta_n}\|\mathbf{y}_n^{k+1} - \mathbf{y}'\|^2 - \frac{\alpha_n}{2\beta_n}\|\mathbf{y}_n^k - \mathbf{y}'\|^2 + \frac{\alpha_n}{2\beta_n}\|\mathbf{y}_n^k - \mathbf{y}_n^{k+1}\|^2. \qquad (5.28)
\end{aligned}$$

From Step 4 of Algorithm 7 and (5.27), we also have the following relations

$$\begin{aligned}
\mathbf{x}_{n+1} - \mathbf{x}_n^{k+1} &= \alpha_n P_n^{-1} W^T(\mathbf{y}_n^{k+1} - \tilde{\mathbf{y}}_n) \qquad (5.29) \\
\mathbf{x}_n^k - \mathbf{x}_n^{k+1} &= \alpha_n P_n^{-1} W^T(\mathbf{y}_n^{k+1} - \mathbf{y}_n^k).
\end{aligned}$$

Plugging the above relations inside (5.28) yields

$$\begin{aligned}
\mathcal{L}(\mathbf{x}_{n+1}, \mathbf{y}_n^{k+1}) &\geq \mathcal{L}(\mathbf{x}_{n+1}, \mathbf{y}') + \alpha_n\langle W^T(\mathbf{y}_n^{k+1} - \mathbf{y}'), P_n^{-1}W^T(\mathbf{y}_n^{k+1} - \tilde{\mathbf{y}}_n)\rangle \\
&- \alpha_n\langle W^T(\mathbf{y}_n^{k+1} - \mathbf{y}'), P_n^{-1}W^T(\mathbf{y}_n^{k+1} - \mathbf{y}_n^k)\rangle \\
&+ \frac{\alpha_n}{2\beta_n}\|\mathbf{y}_n^{k+1} - \mathbf{y}'\|^2 - \frac{\alpha_n}{2\beta_n}\|\mathbf{y}_n^k - \mathbf{y}'\|^2 + \frac{\alpha_n}{2\beta_n}\|\mathbf{y}_n^k - \mathbf{y}_n^{k+1}\|^2. \qquad (5.30)
\end{aligned}$$

Note that the first scalar product in the previous inequality can be lower bounded as follows

$$\begin{aligned}
\langle W^T(\mathbf{y}_n^{k+1} - \mathbf{y}'), P_n^{-1}W^T(\mathbf{y}_n^{k+1} - \tilde{\mathbf{y}}_n)\rangle &= \frac{1}{2}\|P_n^{-\frac{1}{2}}W^T(\mathbf{y}_n^{k+1} - \mathbf{y}')\|^2 \\
&+ \frac{1}{2}\|P_n^{-\frac{1}{2}}W^T(\mathbf{y}_n^{k+1} - \tilde{\mathbf{y}}_n)\|^2 \\
&- \frac{1}{2}\|P_n^{-\frac{1}{2}}W^T(\tilde{\mathbf{y}}_n - \mathbf{y}')\|^2 \\
&\geq \frac{1}{2}\|P_n^{-\frac{1}{2}}W^T(\mathbf{y}_n^{k+1} - \mathbf{y}')\|^2 \\
&- \frac{1}{2}\|P_n^{-\frac{1}{2}}W^T(\tilde{\mathbf{y}}_n - \mathbf{y}')\|^2,
\end{aligned}$$

whereas the second one can be rewritten as

$$\begin{aligned}
-\langle W^T(\mathbf{y}_n^{k+1} - \mathbf{y}'), P_n^{-1}W^T(\mathbf{y}_n^{k+1} - \mathbf{y}_n^k)\rangle &= -\frac{1}{2}\|P_n^{-\frac{1}{2}}W^T(\mathbf{y}_n^{k+1} - \mathbf{y}')\|^2 \\
&- \frac{1}{2}\|P_n^{-\frac{1}{2}}W^T(\mathbf{y}_n^{k+1} - \mathbf{y}_n^k)\|^2 \\
&+ \frac{1}{2}\|P_n^{-\frac{1}{2}}W^T(\mathbf{y}_n^k - \mathbf{y}')\|^2.
\end{aligned}$$

By inserting the previous relations inside (5.30) and recalling the definition of D_n and $\|\cdot\|_{D_n}$

in (5.15)-(5.17), we come to

$$\mathcal{L}(\mathbf{x}_{n+1}, \mathbf{y}_n^{k+1}) \geq \mathcal{L}(\mathbf{x}_{n+1}, \mathbf{y}') - \frac{\alpha_n}{2}\|P_n^{-\frac{1}{2}} W^T(\tilde{\mathbf{y}}_n - \mathbf{y}')\|^2 + \frac{\alpha_n}{2\beta_n}\|\mathbf{y}_n^{k+1} - \mathbf{y}'\|^2 - \frac{\alpha_n}{2\beta_n}\|\mathbf{y}_n^k - \mathbf{y}'\|_{D_n}^2 + \frac{\alpha_n}{2\beta_n}\|\mathbf{y}_n^k - \mathbf{y}_n^{k+1}\|_{D_n}^2. \tag{5.31}$$

We proceed as follows

$$\begin{aligned}
\mathcal{L}(\mathbf{x}_{n+1}, \tilde{\mathbf{y}}_n) &= \mathcal{L}\left(\mathbf{x}_{n+1}, \frac{1}{k_{\max}} \sum_{k=0}^{k_{\max}-1} \mathbf{y}_n^{k+1}\right) \\
&\geq \frac{1}{k_{\max}} \sum_{k=0}^{k_{\max}-1} \mathcal{L}(\mathbf{x}_{n+1}, \mathbf{y}_n^{k+1}) \\
&\geq \mathcal{L}(\mathbf{x}_{n+1}, \mathbf{y}') - \frac{\alpha_n}{2}\|P_n^{-\frac{1}{2}} W^T(\tilde{\mathbf{y}}_n - \mathbf{y}')\|^2 \\
&+ \frac{1}{k_{\max}} \sum_{k=0}^{k_{\max}-1} \frac{\alpha_n}{2\beta_n}\|\mathbf{y}_n^{k+1} - \mathbf{y}'\|^2 - \frac{\alpha_n}{2\beta_n}\|\mathbf{y}_n^k - \mathbf{y}'\|_{D_n}^2 + \frac{\alpha_n}{2\beta_n}\|\mathbf{y}_n^k - \mathbf{y}_n^{k+1}\|_{D_n}^2 \\
&\geq \mathcal{L}(\mathbf{x}_{n+1}, \mathbf{y}') - \frac{1}{k_{\max}} \sum_{k=0}^{k_{\max}-1} \frac{\alpha_n}{2}\|P_n^{-\frac{1}{2}} W^T(\mathbf{y}_n^{k+1} - \mathbf{y}')\|^2 \\
&+ \frac{1}{k_{\max}} \sum_{k=0}^{k_{\max}-1} \frac{\alpha_n}{2\beta_n}\|\mathbf{y}_n^{k+1} - \mathbf{y}'\|^2 - \frac{\alpha_n}{2\beta_n}\|\mathbf{y}_n^k - \mathbf{y}'\|_{D_n}^2 + \frac{\alpha_n}{2\beta_n}\|\mathbf{y}_n^k - \mathbf{y}_n^{k+1}\|_{D_n}^2,
\end{aligned}$$

where the first inequality is obtained by concavity of $\mathcal{L}(\mathbf{x}_{n+1}, \cdot)$, the second one from applying (5.31), and the third one follows from the definition of $\tilde{\mathbf{y}}_n$ and the concavity of $-\|P_n^{-\frac{1}{2}} W^T(\cdot - \mathbf{y}')\|^2$. We further extend the above chain of inequalities as done below

$$\begin{aligned}
\mathcal{L}(\mathbf{x}_{n+1}, \tilde{\mathbf{y}}_n) &\geq \mathcal{L}(\mathbf{x}_{n+1}, \mathbf{y}') + \frac{1}{k_{\max}} \sum_{k=0}^{k_{\max}-1} \frac{\alpha_n}{2\beta_n}\|\mathbf{y}_n^{k+1} - \mathbf{y}'\|_{D_n}^2 - \frac{\alpha_n}{2\beta_n}\|\mathbf{y}_n^k - \mathbf{y}'\|_{D_n}^2 \\
&+ \frac{\alpha_n}{2\beta_n}\|\mathbf{y}_n^k - \mathbf{y}_n^{k+1}\|_{D_n}^2 \\
&= \mathcal{L}(\mathbf{x}_{n+1}, \mathbf{y}') + \frac{\alpha_n}{2\beta_n k_{\max}}\|\mathbf{y}_n^{k_{\max}} - \mathbf{y}'\|_{D_n}^2 - \frac{\alpha_n}{2\beta_n k_{\max}}\|\mathbf{y}_n^0 - \mathbf{y}'\|_{D_n}^2 \\
&+ \frac{1}{k_{\max}} \sum_{k=0}^{k_{\max}-1} \frac{\alpha_n}{2\beta_n}\|\mathbf{y}_n^k - \mathbf{y}_n^{k+1}\|_{D_n}^2 \\
&= \mathcal{L}(\mathbf{x}_{n+1}, \mathbf{y}') + \frac{\alpha_n}{2\beta_n k_{\max}}\|\mathbf{y}_{n+1}^0 - \mathbf{y}'\|_{D_n}^2 - \frac{\alpha_n}{2\beta_n k_{\max}}\|\mathbf{y}_n^0 - \mathbf{y}'\|_{D_n}^2 \\
&+ \frac{1}{k_{\max}} \sum_{k=0}^{k_{\max}-1} \frac{\alpha_n}{2\beta_n}\|\mathbf{y}_n^k - \mathbf{y}_n^{k+1}\|_{D_n}^2,
\end{aligned}$$

where the first inequality has been obtained by recalling once again the definition of D_n in (5.15), and the last equality is a consequence of the warm-start strategy at Step 3 of Algorithm 7. By rearranging terms, we finally get inequality (5.25).
(iii) Inequality (5.26) follows by summing (5.24) with (5.25). □

The main convergence result for Algorithm 7 is stated below. The line of proof employed is analogous to the one in [26, Theorem 2], even though it must be adapted to the presence of variable metrics and variable steplengths.

Theorem 5.2.12. *Suppose that Assumption 4.2.39 holds and let* $(\hat{\mathbf{x}}, \hat{\mathbf{y}}) \in \mathbb{R}^d \times \mathbb{R}^{d'}$ *be a solution of the primal-dual problem (4.36). Let* $\{(\mathbf{x}_n, \mathbf{y}_n^0)\}_{n\in\mathbb{N}}$ *be the primal-dual sequence generated by Algorithm 7. Suppose that the inertial parameters* $\{\gamma_n\}_{n\in\mathbb{N}}$ *and scaling matrices* $\{P_n\}_{n\in\mathbb{N}}$ *of Algorithm 7 comply with Assumption 5.2.6. Then the following statements hold true.*

(i) The sequence $\{(\mathbf{x}_n, \mathbf{y}_n^0)\}_{n\in\mathbb{N}}$ *is bounded.*

(ii) Given a saddle point $(\hat{\mathbf{x}}, \hat{\mathbf{y}})$ *solution of (4.36), the sequence* $\{\beta_{n-1}k_{\max}\|\hat{\mathbf{x}} - \mathbf{x}_n\|^2_{P_{n-1}} + \alpha^2_{n-1}\|\hat{\mathbf{y}} - \mathbf{y}_n^0\|^2_{D_{n-1}}\}_{n\in\mathbb{N}}$ *converges.*

(iii) The sequence $\{(\mathbf{x}_n, \mathbf{y}_n^0)\}_{n\in\mathbb{N}}$ *converges to a solution of (4.36).*

Before going into the technical details of the proof, let us briefly summarize how we intend to prove each item of Theorem 5.2.12.

(i) The proof of item (i) relies on an upper bound on the quantity $\|\mathbf{x}_{n+1} - \hat{\mathbf{x}}\|_{P_n}$, which is obtained by employing Lemma 5.2.3 in combination with Assumption 5.2.6.

(ii) Item (ii) follows by applying Lemma 5.2.4 to the sequence $\{\beta_{n-1}k_{\max}\|\hat{\mathbf{x}} - \mathbf{x}_n\|^2_{P_{n-1}} + \alpha^2_{n-1}\|\hat{\mathbf{y}} - \mathbf{y}_n^0\|^2_{D_{n-1}}\}_{n\in\mathbb{N}}$, which is possible thanks to item (i).

(iii) For item (iii), we first show that any limit point $(\mathbf{x}^\dagger, \mathbf{y}^\dagger) \in \mathbb{R}^d \times \mathbb{R}^{d'}$ of the sequence $\{(u^n, \mathbf{y}_n^0)\}_{n\in\mathbb{N}}$ is a solution to problem (4.36); then, the thesis follows by applying item (ii) with the choice $(\hat{\mathbf{x}}, \hat{\mathbf{y}}) = (\mathbf{x}^\dagger, \mathbf{y}^\dagger)$.

Proof. (i) Consider inequality (5.26) with $\mathbf{x}' = \hat{\mathbf{x}}$, $\mathbf{y}' = \hat{\mathbf{y}}$ and the term $-\frac{1}{2}\left(\frac{1}{\alpha_n} - L_n\right)\|\mathbf{x}_{n+1} - \bar{\mathbf{x}}_n\|^2_{P_n} - \frac{1}{k_{\max}}\sum_{k=0}^{k_{\max}-1}\frac{\alpha_n}{2\beta_n}\|\mathbf{y}_n^k - \mathbf{y}_n^{k+1}\|^2_{D_n}$ discarded, which amounts to writing

$$\begin{aligned}\mathcal{L}(\mathbf{x}_{n+1}, \hat{\mathbf{y}}) + \frac{1}{2\alpha_n}\|\mathbf{x}_{n+1} - \hat{\mathbf{x}}\|^2_{P_n} + \frac{\alpha_n}{2\beta_n k_{\max}}\|\mathbf{y}_{n+1}^0 - \hat{\mathbf{y}}\|^2_{D_n} \\ \leq \mathcal{L}(\hat{\mathbf{x}}, \tilde{\mathbf{y}}_n) + \frac{1}{2\alpha_n}\|\bar{\mathbf{x}}_n - \hat{\mathbf{x}}\|^2_{P_n} + \frac{\alpha_n}{2\beta_n k_{\max}}\|\mathbf{y}_n^0 - \hat{\mathbf{y}}\|^2_{D_n}.\end{aligned}$$

By observing that $\mathcal{L}(\mathbf{x}_{n+1}, \hat{\mathbf{y}}) \geq \mathcal{L}(\hat{\mathbf{x}}, \tilde{\mathbf{y}}_n)$, due to the fact that $(\hat{\mathbf{x}}, \hat{\mathbf{y}})$ is a solution of the primal-dual problem (4.36), and multiplying the above inequality by $\alpha_n\beta_n$, we deduce the inequality

$$\frac{\beta_n}{2}\|\mathbf{x}_{n+1} - \hat{\mathbf{x}}\|^2_{P_n} + \frac{\alpha_n^2}{2k_{\max}}\|\mathbf{y}_{n+1}^0 - \hat{\mathbf{y}}\|^2_{D_n} \leq \frac{\beta_n}{2}\|\bar{\mathbf{x}}_n - \hat{\mathbf{x}}\|^2_{P_n} + \frac{\alpha_n^2}{2k_{\max}}\|\mathbf{y}_n^0 - \hat{\mathbf{y}}\|^2_{D_n}. \quad (5.32)$$

By recalling that the sequences $\{\alpha_n\}_{n\in\mathbb{N}}$ and $\{\beta_n\}_{n\in\mathbb{N}}$ satisfy (5.12) and (5.21), respectively,

and applying conditions (5.19) and (5.22) to inequality (5.32), we get

$$\begin{aligned}
&\frac{\beta_n}{2}\|\mathbf{x}_{n+1}-\hat{\mathbf{x}}\|^2_{P_n}+\frac{\alpha_n^2}{2k_{\max}}\|\mathbf{y}^0_{n+1}-\hat{\mathbf{y}}\|^2_{D_n}\\
&\leq (1+\tilde{\zeta}_{n-1})(1+\zeta_{n-1})\left(\frac{\beta_{n-1}}{2}\|\bar{\mathbf{x}}_n-\hat{\mathbf{x}}\|^2_{P_{n-1}}+\frac{\alpha_{n-1}^2}{2k_{\max}}\|\mathbf{y}^0_n-\hat{\mathbf{y}}\|^2_{D_{n-1}}\right)\\
&= (1+\chi_{n-1})\left(\frac{\beta_{n-1}}{2}\|\bar{\mathbf{x}}_n-\hat{\mathbf{x}}\|^2_{P_{n-1}}+\frac{\alpha_{n-1}^2}{2k_{\max}}\|\mathbf{y}^0_n-\hat{\mathbf{y}}\|^2_{D_{n-1}}\right),
\end{aligned}\tag{5.33}$$

where $\chi_{n-1}=\zeta_{n-1}+\tilde{\zeta}_{n-1}+\zeta_{n-1}\tilde{\zeta}_{n-1}$ for all $n\geq 0$ and $\{\chi_n\}_{n\in\mathbb{N}}$ is still a nonnegative summable sequence. Since $\bar{\mathbf{x}}_n=\mathbf{x}_n+\gamma_n(\mathbf{x}_n-\mathbf{x}_{n-1})$, an application of the Cauchy-Schwarz yields

$$\begin{aligned}
&\frac{\beta_n}{2}\|\mathbf{x}_{n+1}-\hat{\mathbf{x}}\|^2_{P_n}+\frac{\alpha_n^2}{2k_{\max}}\|\mathbf{y}^0_{n+1}-\hat{\mathbf{y}}\|^2_{D_n}\\
&\leq (1+\chi_{n-1})\left(\frac{\beta_{n-1}}{2}\|\mathbf{x}_n-\hat{\mathbf{x}}\|^2_{P_{n-1}}+\frac{\alpha_{n-1}^2}{2k_{\max}}\|\mathbf{y}^0_n-\hat{\mathbf{y}}\|^2_{D_{n-1}}\right)\\
&+\beta_{n-1}(1+\chi_{n-1})\left(\frac{\gamma_n^2}{2}\|\mathbf{x}_n-\mathbf{x}_{n-1}\|^2_{P_{n-1}}+\gamma_n\langle\mathbf{x}_n-\hat{\mathbf{x}},P_{n-1}(\mathbf{x}_n-\mathbf{x}_{n-1})\rangle\right)\\
&\leq (1+\chi_{n-1})\left(\frac{\beta_{n-1}}{2}\|\mathbf{x}_n-\hat{\mathbf{x}}\|^2_{P_{n-1}}+\frac{\alpha_{n-1}^2}{2k_{\max}}\|\mathbf{y}^0_n-\hat{\mathbf{y}}\|^2_{D_{n-1}}\right)\\
&+\beta_{n-1}(1+\chi_{n-1})\left(\frac{\gamma_n^2}{2}\|\mathbf{x}_n-\mathbf{x}_{n-1}\|^2_{P_{n-1}}+\gamma_n\|\mathbf{x}_n-\hat{\mathbf{x}}\|_{P_{n-1}}\|\mathbf{x}_n-\mathbf{x}_{n-1}\|_{P_{n-1}}\right).
\end{aligned}\tag{5.34}$$

By applying recursively inequality (5.34), we get

$$\begin{aligned}
&\frac{\beta_n}{2}\|\mathbf{x}_{n+1}-\hat{\mathbf{x}}\|^2_{P_n}+\frac{\alpha_n^2}{2k_{\max}}\|\mathbf{y}^0_{n+1}-\hat{\mathbf{y}}\|^2_{D_n}\\
&\leq\left(\prod_{k=0}^{n}(1+\chi_{k-1})\right)\left(\frac{\beta_{-1}}{2}\|\mathbf{x}_0-\hat{\mathbf{x}}\|^2_{P_{-1}}+\frac{\alpha_{-1}^2}{2k_{\max}}\|\mathbf{y}^0_0-\hat{\mathbf{y}}\|^2_{D_{-1}}\right)\\
&+\sum_{k=0}^{n}\beta_{k-1}\left(\prod_{i=k}^{n}(1+\chi_{i-1})\right)\left(\frac{\gamma_k^2}{2}\|\mathbf{x}_k-\mathbf{x}_{k-1}\|^2_{P_{k-1}}+\gamma_k\|\mathbf{x}_k-\hat{\mathbf{x}}\|_{P_{k-1}}\|\mathbf{x}_k-\mathbf{x}_{k-1}\|_{P_{k-1}}\right).
\end{aligned}$$

Now, if we consider the sequence

$$\Lambda_n=\prod_{k=0}^{n}(1+\chi_{k-1}),\quad n\geq 0,$$

we note that $\Lambda_n=(1+\chi_{n-1})\Lambda_{n-1}$, which implies that $\{\Lambda_n\}_{n\in\mathbb{N}}$ is an increasing and convergent sequence, see Lemma 5.2.4. By setting $\Lambda=\lim_{n\to\infty}\Lambda_n$, it follows that $\Lambda_n\leq\Lambda$

for all $n \geq 0$, $\prod_{i=k}^{n}(1+\chi_{i-1}) \leq \Lambda$ for $0 \leq k \leq n$, and the previous inequality yields

$$\frac{\beta_n}{2}\|\mathbf{x}_{n+1}-\hat{\mathbf{x}}\|_{P_n}^2+\frac{\alpha_n^2}{2k_{\max}}\|\mathbf{y}_{n+1}^0-\hat{\mathbf{y}}\|_{D_n}^2 \leq \Lambda\left(\frac{\beta_{-1}}{2}\|\mathbf{x}_0-\hat{\mathbf{x}}\|_{P_{-1}}^2+\frac{\alpha_{-1}^2}{2k_{\max}}\|\mathbf{y}_0^0-\hat{\mathbf{y}}\|_{D_{-1}}^2\right)$$
$$+\sum_{k=0}^{n}\beta_{k-1}\Lambda\left(\frac{\gamma_k^2}{2}\|\mathbf{x}_k-\mathbf{x}_{k-1}\|_{P_{k-1}}^2+\gamma_k\|\mathbf{x}_k-\hat{\mathbf{x}}\|_{P_{k-1}}\|\mathbf{x}_k-\mathbf{x}_{k-1}\|_{P_{k-1}}\right). \tag{5.35}$$

By discarding the term proportional to $\|\mathbf{y}_{n+1}^0-\hat{\mathbf{y}}\|_{D_n}^2$ on the left-hand side of the previous inequality, multiplying both sides by a factor $2/\beta_n$, using the inequalities $(\epsilon\varsigma)/\|W\|^2 \leq \beta_k \leq (1+\zeta_{k-1})\beta_{k-1} \leq \Lambda_k\beta_{-1} \leq \Lambda\beta_{-1}$ for $k=0,\dots,n$, and adding the term $\frac{2\Lambda^2\|W\|^2}{\epsilon\varsigma}\gamma_{n+1}\|\mathbf{x}_{n+1}-\hat{\mathbf{x}}\|_{P_n}\|\mathbf{x}_{n+1}-\mathbf{x}_n\|_{P_n}$ to the right-hand side, one gets

$$\|\mathbf{x}_{n+1}-\hat{\mathbf{x}}\|_{P_n}^2 \leq \frac{\Lambda\|W\|^2}{\epsilon\varsigma}\left(\beta_{-1}\|\mathbf{x}_0-\hat{\mathbf{x}}\|_{P_{-1}}^2+\frac{\alpha_{-1}^2}{k_{\max}}\|\mathbf{y}_0^0-\hat{\mathbf{y}}\|_{D_{-1}}^2\right.$$
$$\left.+\sum_{k=0}^{n}\Lambda\beta_{-1}\gamma_k^2\|\mathbf{x}_k-\mathbf{x}_{k-1}\|_{P_{k-1}}^2\right)+\sum_{k=0}^{n+1}\frac{2\Lambda^2\|W\|^2\beta_{-1}}{\epsilon\varsigma}\gamma_k\|\mathbf{x}_k-\mathbf{x}_{k-1}\|_{P_{k-1}}\|\mathbf{x}_k-\hat{\mathbf{x}}\|_{P_{k-1}}.$$

At this point, we can apply Lemma 5.2.3 with $\mathbf{a}_n = \|\mathbf{x}_n-\hat{\mathbf{x}}\|_{P_{n-1}}$, $\mathbf{b}_n = \frac{\Lambda\|W\|^2}{\epsilon\varsigma}\beta_{-1}\|\mathbf{x}_0-\hat{\mathbf{x}}\|_{P_{-1}}^2+\frac{\Lambda\|W\|^2}{\epsilon\varsigma}\frac{\alpha_{-1}^2}{k_{\max}}\|\mathbf{y}_0^0-\hat{\mathbf{y}}\|_{D_{-1}}^2+\frac{\Lambda\|W\|^2}{\epsilon\varsigma}\sum_{k=0}^{n}\Lambda\beta_{-1}\gamma_k^2\|\mathbf{x}_k-\mathbf{x}_{k-1}\|_{P_{k-1}}^2$ and $c_n = \frac{2\Lambda^2\|W\|^2\beta_{-1}}{\epsilon\varsigma}\gamma_n\|\mathbf{x}_n-\mathbf{x}_{n-1}\|_{P_{n-1}}$, thus obtaining

$$\begin{aligned}\|\mathbf{x}_{n+1}-\hat{\mathbf{x}}\|_{P_n} \leq{}& \frac{1}{2}\sum_{k=0}^{n+1}\frac{2\Lambda^2\|W\|^2\beta_{-1}}{\epsilon\varsigma}\gamma_k\|\mathbf{x}_k-\mathbf{x}_{k-1}\|_{P_{k-1}}\\ &+\left(\frac{\Lambda\|W\|^2}{\epsilon\varsigma}\left(\beta_{-1}\|\mathbf{x}_0-\hat{\mathbf{x}}\|_{P_{-1}}^2+\frac{\alpha_{-1}^2}{k_{\max}}\|\mathbf{y}_0^0-\hat{\mathbf{y}}\|_{D_{-1}}^2\right.\right.\\ &\left.+\sum_{k=0}^{n}\Lambda\beta_{-1}\gamma_k^2\|\mathbf{x}_k-\mathbf{x}_{k-1}\|_{P_{k-1}}^2\right)\\ &\left.+\left(\frac{1}{2}\sum_{k=0}^{n+1}\frac{2\Lambda^2\|W\|^2\beta_{-1}}{\epsilon\varsigma}\gamma_k\|\mathbf{x}_k-\mathbf{x}_{k-1}\|_{P_{k-1}}\right)^2\right)^{\frac{1}{2}}.\end{aligned} \tag{5.36}$$

By recursively applying condition (5.19) to (5.18), we get

$$\begin{aligned}\sum_{k=0}^{\infty}\gamma_k\|\mathbf{x}_k-\mathbf{x}_{k-1}\|_{P_{k-1}} &\leq \sum_{k=0}^{\infty}\gamma_k\Lambda_k\|\mathbf{x}_k-\mathbf{x}_{k-1}\|_{P_{-1}}\\ &\leq \Lambda\|P_{-1}\|^{\frac{1}{2}}\sum_{k=0}^{\infty}\gamma_k\|\mathbf{x}_k-\mathbf{x}_{k-1}\| < \infty.\end{aligned} \tag{5.37}$$

By applying the previous inequality to (5.36) and recalling that $P_n \in \mathcal{D}_\varsigma$, it follows that the sequence $\{\mathbf{x}_n\}_{n\in\mathbb{N}}$ is bounded. Finally, by discarding the term proportional to $\|\mathbf{x}_{n+1}-\hat{\mathbf{x}}\|_{P_n}^2$ in (5.35), employing again (5.37), the boundedness of $\{\mathbf{x}_n\}_{n\in\mathbb{N}}$, the lower bound (5.16) on the eigenvalues of D_n and $\beta_{k-1} \leq \Lambda\beta_{-1}$, we conclude that also $\{\mathbf{y}_n^0\}_{n\in\mathbb{N}}$ is bounded.

(ii) Item (i) guarantees the existence of a constant $M > 0$ such that $\|\mathbf{x}_n-\hat{\mathbf{x}}\|_{P_n} \leq M$ for

all $n \geq 0$. Furthermore, we have $\beta_{n-1} \leq \Lambda\beta_{-1}$ for all $n \geq 0$. Then, from (5.34), we deduce the following inequality

$$\begin{aligned}
&\beta_n k_{\max}\|\mathbf{x}_{n+1}-\hat{\mathbf{x}}\|^2_{P_n} + \alpha_n^2\|\mathbf{y}^0_{n+1}-\hat{\mathbf{y}}\|^2_{D_n} \\
&\quad\leq (1+\chi_{n-1})(\beta_{n-1}k_{\max}\|\mathbf{x}_n-\hat{\mathbf{x}}\|^2_{P_{n-1}} + \alpha_{n-1}^2\|\mathbf{y}^0_n-\hat{\mathbf{y}}\|^2_{D_{n-1}}) \\
&\quad+ \Lambda\beta_{-1}k_{\max}(1+\chi_{n-1})(\gamma_n^2\|\mathbf{x}_n-\mathbf{x}_{n-1}\|^2_{P_{n-1}} + 2M\gamma_n\|\mathbf{x}_n-\mathbf{x}_{n-1}\|_{P_{n-1}}).
\end{aligned}$$

The previous inequality and (5.37) allow us to apply Lemma 5.2.4 with $a_n = \beta_{n-1}k_{\max}\|\mathbf{x}_n - \hat{\mathbf{x}}\|^2_{P_{n-1}} + \alpha_{n-1}^2\|\mathbf{y}^0_n - \hat{\mathbf{y}}\|^2_{D_{n-1}}$, $b_n = \chi_{n-1}$ and $c_n = \Lambda\beta_{-1}k_{\max}(1+\chi_{n-1})(\gamma_n^2\|\mathbf{x}_n-\mathbf{x}_{n-1}\|^2_{P_{n-1}} + 2M\gamma_n\|\mathbf{x}_n - \mathbf{x}_{n-1}\|_{P_{n-1}})$, thus we get the thesis.
(iii) In virtue of (5.12)-(5.21), the sequences $\{\alpha_n\}_{n\in\mathbb{N}}$ and $\{\beta_n\}_{n\in\mathbb{N}}$ both satisfy the hypotheses of Lemma 5.2.4 and are bounded away from zero; hence, there exist $\alpha > 0$ and $\beta > 0$ such that

$$\lim_{n\to\infty}\alpha_n = \alpha, \quad \lim_{n\to\infty}\beta_n = \beta. \tag{5.38}$$

From inequality (5.37), we deduce that $\{P_n\}_{n\in\mathbb{N}} \subseteq \mathcal{D}_\varsigma^{\Lambda^2\|P_{-1}\|}$; likewise, we have $\{D_n\}_{n\in\mathbb{N}} \subseteq \mathcal{D}_{1-\epsilon}^{\Lambda^2\|D_{-1}\|}$. Thus, by Lemma 5.2.10, there exists $P \in \mathcal{D}_\varsigma$ and $D \in \mathcal{D}_{1-\epsilon}$ such that

$$\lim_{n\to\infty}P_n = P, \quad \lim_{n\to\infty}D_n = D. \tag{5.39}$$

Since $\{(\mathbf{x}_n, \mathbf{y}_n^0)\}_{n\in\mathbb{N}}$ is bounded, there exists $(\mathbf{x}^\dagger, \mathbf{y}^\dagger) \in \mathbb{R}^d \times \mathbb{R}^{d'}$ and $I \subseteq \mathbb{N}$ such that

$$\lim_{n\in I}\mathbf{x}_n = \mathbf{x}^\dagger, \quad \lim_{n\in I}\mathbf{y}_n^0 = \mathbf{y}^\dagger. \tag{5.40}$$

Let us show that $(\mathbf{x}^\dagger, \mathbf{y}^\dagger)$ is a solution of problem (4.36). To this aim, we consider inequality (5.26) with $\mathbf{x}' = \hat{\mathbf{x}}$, $\mathbf{y}' = \hat{\mathbf{y}}$, and by observing that $\mathcal{L}(\mathbf{x}_{n+1}, \hat{\mathbf{y}}) \geq \mathcal{L}(\hat{\mathbf{x}}, \tilde{\mathbf{y}}_n)$, applying relation $\bar{\mathbf{x}}_n = \mathbf{x}_n + \gamma(\mathbf{x}_n - \mathbf{x}_{n-1})$ and the Cauchy-Schwarz inequality, we obtain

$$\begin{aligned}
&\frac{1}{2}\left(\frac{1}{\alpha_n} - L_n\right)\|\mathbf{x}_{n+1}-\bar{\mathbf{x}}_n\|^2_{P_n} + \frac{1}{k_{\max}}\sum_{k=0}^{k_{\max}-1}\frac{\alpha_n^2}{2}\|\mathbf{y}_n^k - \mathbf{y}_n^{k+1}\|^2_{D_n} \\
&\leq \frac{\beta_n}{2}(\|\mathbf{x}_n-\hat{\mathbf{x}}\|^2_{P_n} - \|\mathbf{x}_{n+1}-\hat{\mathbf{x}}\|^2_{P_n}) + \frac{\alpha_n^2}{2k_{\max}}(\|\mathbf{y}_n^0-\hat{\mathbf{y}}\|^2_{D_n} - \|\mathbf{y}_{n+1}^0-\hat{\mathbf{y}}\|^2_{D_n}) \\
&+ \frac{\beta_n\gamma_n^2}{2}\|\mathbf{x}_n-\mathbf{x}_{n-1}\|^2_{P_n} + \beta_n\gamma_n\|\mathbf{x}_n-\hat{\mathbf{x}}\|_{P_{n-1}}\|\mathbf{x}_n-\mathbf{x}_{n-1}\|_{P_{n-1}}.
\end{aligned} \tag{5.41}$$

Since $\{(\mathbf{x}_n, \mathbf{y}_n^0)\}_{n\in\mathbb{N}}$ is bounded, there exist constants $M > 0$ and $\tilde{M} > 0$ such that

$$\|\mathbf{x}_n - \hat{\mathbf{x}}\|_{P_n} \leq M, \quad \|\mathbf{y}_n^0 - \hat{\mathbf{y}}\|_{D_n} \leq \tilde{M}, \quad \forall\, n \geq 0. \tag{5.42}$$

Additionally, by Step 7 of Algorithm 7, we observe that

$$\frac{1}{\alpha_n} - L_n = L_n\left(\frac{1}{\epsilon} - 1\right) \geq L_0\left(\frac{1}{\epsilon} - 1\right) > 0, \quad \forall\, n \geq 0. \tag{5.43}$$

By employing bounds (5.42)-(5.43), conditions (5.19)-(5.22), the fact that $\alpha_n \leq \alpha_{n-1}$ and

$\beta_n \leq (1+\zeta_{n-1})\beta_{n-1} \leq (1+\chi_{n-1})\beta_{n-1} \leq \Lambda\beta_{-1}$ as seen in (5.12) and (5.21), respectively, and the lower bounds on the eigenvalues of $\{P_n\}_{n\in\mathbb{N}}$ and $\{D_n\}_{n\in\mathbb{N}}$, we derive from (5.41) the following inequality

$$\begin{aligned}
&\frac{\varsigma L_0}{2}\left(\frac{1}{\epsilon}-1\right)\|\mathbf{x}_{n+1}-\bar{\mathbf{x}}_n\|^2 + \frac{\min\left\{\alpha_0^2, \frac{\epsilon^2\delta^2\varsigma^2}{L^2}\right\}(1-\epsilon)}{2k_{\max}}\sum_{k=0}^{k_{\max}-1}\|\mathbf{y}_n^k-\mathbf{y}_n^{k+1}\|^2\\
&\leq \frac{1}{2}(\beta_{n-1}\|\mathbf{x}_n-\hat{\mathbf{x}}\|^2_{P_{n-1}} - \beta_n\|\mathbf{x}_{n+1}-\hat{\mathbf{x}}\|^2_{P_n})\\
&+ \frac{1}{2k_{\max}}(\alpha_{n-1}^2\|\mathbf{y}_n^0-\hat{\mathbf{y}}\|^2_{D_{n-1}} - \alpha_n^2\|\mathbf{y}_{n+1}^0-\hat{\mathbf{y}}\|^2_{D_n}) + \frac{1}{2}\left(\Lambda\beta_{-1}M^2 + \frac{\alpha_{-1}^2\tilde{M}^2}{k_{\max}}\right)\chi_{n-1}\\
&+ \frac{\Lambda\beta_{-1}\gamma_n^2(1+\chi_{n-1})}{2}\|\mathbf{x}_n-\mathbf{x}_{n-1}\|^2_{P_{n-1}} + \Lambda\beta_{-1}M\gamma_n\|\mathbf{x}_n-\mathbf{x}_{n-1}\|_{P_{n-1}}.
\end{aligned}$$

We sum the previous relation over $n=0,\ldots,N$, thus obtaining

$$\begin{aligned}
&\frac{\varsigma L_0}{2}\left(\frac{1}{\epsilon}-1\right)\sum_{n=0}^{N}\|\mathbf{x}_{n+1}-\bar{\mathbf{x}}_n\|^2 + \frac{\min\left\{\alpha_0^2, \frac{\epsilon^2\delta^2\varsigma^2}{L^2}\right\}(1-\epsilon)}{2k_{\max}}\sum_{n=0}^{N}\sum_{k=0}^{k_{\max}-1}\|\mathbf{y}_n^k-\mathbf{y}_n^{k+1}\|^2\\
&\leq \frac{1}{2}(\beta_{-1}\|\mathbf{x}_0-\hat{\mathbf{x}}\|^2_{P_{-1}} - \beta_N\|\mathbf{x}_{n+1}-\hat{\mathbf{x}}\|^2_{P_N})\\
&+ \frac{1}{2k_{\max}}(\alpha_{-1}^2\|\mathbf{y}_0^0-\hat{\mathbf{y}}\|^2_{D_{-1}} - \alpha_N^2\|\mathbf{y}_{n+1}^0-\hat{\mathbf{y}}\|^2_{D_N}) + \frac{1}{2}\left(\Lambda\beta_{-1}M^2 + \frac{\alpha_{-1}^2\tilde{M}^2}{k_{\max}}\right)\sum_{n=0}^{N}\chi_{n-1}\\
&+ \sum_{n=0}^{N}\frac{\Lambda\beta_{-1}\gamma_n^2(1+\chi_{n-1})}{2}\|\mathbf{x}_n-\mathbf{x}_{n-1}\|^2_{P_{n-1}} + \sum_{n=0}^{N}\Lambda\beta_{-1}M\gamma_n\|\mathbf{x}_n-\mathbf{x}_{n-1}\|_{P_{n-1}}.
\end{aligned}$$

Taking the limit for $N\to\infty$ and using conditions (5.22) and (5.37) yields

$$\sum_{n=0}^{\infty}\|\mathbf{x}_{n+1}-\bar{\mathbf{x}}_n\|^2 < \infty, \quad \sum_{n=0}^{\infty}\sum_{k=0}^{k_{\max}-1}\|\mathbf{y}_n^k-\mathbf{y}_n^{k+1}\|^2 < \infty,$$

which trivially leads to

$$\lim_{n\to\infty}\|\mathbf{x}_{n+1}-\bar{\mathbf{x}}_n\|^2 = 0, \quad \lim_{n\to\infty}\|\mathbf{y}_n^k-\mathbf{y}_n^{k+1}\| = 0, \quad k=0,\ldots,k_{\max}-1. \tag{5.44}$$

Since $\lim_{n\in I}\mathbf{y}_n^0 = \mathbf{y}^\dagger$, the latter limits in (5.44) imply that

$$\lim_{n\in I}\mathbf{y}_n^k = \mathbf{y}^\dagger, \quad k=1,\ldots,k_{\max}. \tag{5.45}$$

Note also that, based on condition (5.18), we have

$$\lim_{n\to\infty}\|\bar{\mathbf{x}}_n-\mathbf{x}_n\| = \lim_{n\to\infty}\gamma_n\|\mathbf{x}_n-\mathbf{x}_{n-1}\| = 0,$$

which means that

$$\lim_{n\in I}\bar{\mathbf{x}}_n = \mathbf{x}^\dagger. \tag{5.46}$$

In turn, the above limit and the former one in (5.44) imply

$$\lim_{n\in I} \mathbf{x}_{n+1} = \mathbf{x}^\dagger. \tag{5.47}$$

By plugging limits (5.38), (5.39), (5.40), (5.45), (5.46) and (5.47) inside relation (5.27), we obtain

$$\nabla f(\mathbf{x}^\dagger) + W^T \mathbf{y}^\dagger = 0. \tag{5.48}$$

Likewise, by inserting the same limits inside relation (5.29), we get

$$\lim_{n\in I} \mathbf{x}_n^{k+1} = \mathbf{x}^\dagger, \quad k = 0, \dots, k_{\max}-1.$$

In particular, for $k = 0$, we have $\lim_{n\in I} \mathbf{x}_n^1 = \mathbf{x}^\dagger$. By combining this last fact with (5.38), (5.45) and the continuity of the operator $\mathrm{prox}_{\beta\alpha^{-1}h^*}$ inside the dual step (5.10), we conclude that

$$\mathbf{y}^\dagger = \mathrm{prox}_{\beta\alpha^{-1}h^*}(\mathbf{y}^\dagger + \beta\alpha^{-1} W\mathbf{x}^\dagger). \tag{5.49}$$

Thanks to equations (5.48) and (5.49), we can apply Lemma 4.2.40 and conclude that $(\mathbf{x}^\dagger, \mathbf{y}^\dagger)$ is a solution of problem (4.36). Hence, it follows from item (ii) that the sequence $\{\beta_{n-1}k_{\max}\|\mathbf{x}^\dagger - \mathbf{x}_n\|^2_{P_{n-1}} + \alpha_{n-1}^2\|\mathbf{y}^\dagger - \mathbf{y}_n^0\|^2_{D_{n-1}}\}_{n\in\mathbb{N}}$ converges and, by definition of limit point, it admits a subsequence converging to zero. Then the sequence $\{(\mathbf{x}_n, \mathbf{y}_n^0)\}_{n\in\mathbb{N}}$ converges to $(\mathbf{x}^\dagger, \mathbf{y}^\dagger)$. □

5.3 The NPD Iterated Tikhonov method (NPDIT)

In this section we focus our attention on the deblurring problem that will be also considered in the numerical test. The selected model is described by the following

$$\arg\min_{\mathbf{x}\in\mathbb{R}^d} \frac{1}{2}\|A\mathbf{x} - \mathbf{b}^\delta\|^2 + \lambda TV(\mathbf{x}), \tag{5.50}$$

where TV stands for the Total Variation operator, $\lambda > 0$ is the regularization parameter, $A \in \mathbb{R}^{d\times d}$ is the blurring operator and $\mathbf{b}^\delta \in \mathbb{R}^d$ is the observed image affected by white Gaussian noise $\boldsymbol{\eta}_\delta$ with a δ level of intensity. Problem (5.50) can be reformulated as (4.34) by setting

$$f(\mathbf{x}) = \frac{1}{2}\|A\mathbf{x} - \mathbf{b}^\delta\|^2, \quad h(W\mathbf{x}) = TV(\mathbf{x}) = \sum_{i=1}^{n^2} \|\nabla_i \mathbf{x}\|,$$

where $W = (\nabla_1^T, \dots, \nabla_{d^2}^T)^T \in \mathbb{R}^{2d^2\times d}$ represents the discrete gradient operator and $h\colon \mathbb{R}^{2d^2} \to \bar{\mathbb{R}}$ is defined as

$$h(\mathbf{t}) = \sum_{i=1}^{d^2} \left\| \begin{pmatrix} t_{2i-1} \\ t_{2i} \end{pmatrix} \right\|.$$

Furthermore, it is straightforward to conclude that the minimization problem (5.50) satisfies Assumption 4.2.39. In this section, we describe how to appropriately select the parameters in Algorithm 7 when applied to problem (5.50). Additionally, we outline the experimental

setup used in the numerical examples presented in the next section.

5.3.1 Parameter choice

We equip Algorithm 7 with scaling matrices $\{P_n\}_{n\in\mathbb{N}}$ of the form

$$P_n = A^T A + \nu_n I_d, \quad \forall\, n \geq 0, \tag{5.51}$$

where $I_d \in \mathbb{R}^{d\times d}$ is the identity matrix and $\{\nu_n\}_{n\in\mathbb{N}}$ is either a positive constant sequence $\nu_n \equiv \nu > 0$ or a monotone sequence of positive real numbers converging to a positive value $\nu^* > 0$. These scaling matrices draw inspiration from the Iterated Tikhonov method described in Section §1.4.2. Hence, from now on, we will refer to Algorithm 7 as *NPDIT - Nested Primal–Dual Iterated Tikhonov method*.

Note that the matrices $\{P_n\}_{n\in\mathbb{N}}$ defined in (5.51) can be diagonalized and inverted by two fast Fourier transforms (FFTs) since A is a convolution operator that can be diagonalized by FFT. Moreover, they comply with Assumption 5.2.6 (ii), as stated in the following result.

Lemma 5.3.1. *Let $\{\nu_n\}_{n\in\mathbb{N}} \subseteq \mathbb{R}_{++}$ be a sequence that is either constant or monotone convergent to a positive value $\nu^* > 0$. Then, there exists a summable sequence of positive scalar $\{\zeta_n\}_{n\in\mathbb{N}}$ such that the sequence of matrices $\{P_n\}_{n\in\mathbb{N}}$ satisfies Assumption (5.2.6) (ii) that is*

$$P_n \preceq (1+\zeta_{n-1})P_{n-1}, \quad \forall\, n \geq 0.$$

Proof. If $\{\nu_n\}_{n\in\mathbb{N}}$ is a constant sequence such that $\nu_n \equiv \nu > 0$, then Assumption 5.2.6 (ii) trivially follows with $\zeta_n \equiv 0$.

If $\{\nu_n\}_{n\in\mathbb{N}}$ is a nonincreasing sequence converging to a positive value $\nu^* > 0$, then $P_n \in \mathcal{D}_{\nu_n}$ for all $n \geq 0$ and $P_n \preceq P_{n-1}$ because $\nu_n \leq \nu_{n-1}$. Hence, the sequence $\{P_n\}_{n\in\mathbb{N}}$ satisfies Assumption 5.2.6 (ii) with $\zeta_n \equiv 0$.

Finally, if $\{\nu_n\}_{n\in\mathbb{N}}$ is a nondecreasing sequence converging to a positive value $\nu^* > 0$, then $P_n \in \mathcal{D}_{\nu_0}$ for all $n \geq 0$ and it follows that

$$\begin{aligned}
\mathbf{x}^T P_n \mathbf{x} &= \mathbf{x}^T (A^T A + \nu_n I_d)\mathbf{x} \\
&= \mathbf{x}^T (A^T A + \nu_{n-1} I_d)\mathbf{x} + (\nu_n - \nu_{n-1})\mathbf{x}^T\mathbf{x} \\
&\leq \mathbf{x}^T P_{n-1}\mathbf{x} + (\nu_n - \nu_{n-1})\nu_0^{-1}\mathbf{x}^T P_{n-1}\mathbf{x} \\
&= (1 + (\nu_n - \nu_{n-1})\nu_0^{-1})\mathbf{x}^T P_{n-1}\mathbf{x}, \quad \forall\, \mathbf{x} \in \mathbb{R}^d,
\end{aligned}$$

where the inequality follows from the left-hand inequality in (4.19). By setting $\zeta_{n-1} = (\nu_n - \nu_{n-1})\nu_0^{-1} \geq 0$, we note that

$$\sum_{n=1}^{N} \zeta_{n-1} = \nu_0^{-1}(\nu_N - \nu_0) \quad \Rightarrow \quad \sum_{n=1}^{\infty} \zeta_{n-1} < \infty.$$

Hence, the sequence $\{P_n\}_{n\in\mathbb{N}}$ satisfies Assumption 5.2.6 (ii).

□

According to the selection of the parameter ν_n in (5.51), we consider three different implementations of NPDIT. In the first one, we let

$$\nu_n \equiv \nu, \quad \nu > 0, \tag{5.52}$$

where $\nu > 0$ is a constant that must be prefixed in advance by the user. The selection of ν significantly influences the performance and stability of the method throughout the iterations. Estimating it accurately can be challenging, which is why we have also explored two non-stationary strategies. More precisely, a non-stationary implementation - denominated NPDIT$_{\mathrm{D}}$ - adopts a decreasing geometric sequence for ν_n defined as

$$\nu_n = a_0 q^n + \nu_{\text{final}}, \quad \forall\ n \geq 0, \tag{5.53}$$

where we set $a_0 = \frac{1}{2}$, $q = 0.85$, and $\nu_{\text{final}} = 10^{-2}$ in the numerical results. In a second non-stationary implementation - denominated NPDIT$_{\mathrm{I}}$ - employs an increasing sequence for ν_n defined as

$$\nu_n = 1 - \frac{1}{n+1} + \nu_{\text{initial}}, \quad \forall\ n \geq 0, \tag{5.54}$$

with $\nu_{\text{initial}} = 10^{-2}$ in the numerical examples.

Initially, we also considered the trivial choice $P_n \equiv I_n$ for our tests; however, we noted that such a choice yields inferior performances when compared with the preconditioners (5.51). For this reason, we did not report the related results in the manuscript.

In the NPDIT algorithm and its two non-stationary variants, a backtracking strategy is employed to compute the steplength α_n. Particularly, at each step n, we compute $\alpha_n = \epsilon/L_n$, where $\epsilon = 0.99$ and $L_n > 0$ is such that

$$f(\tilde{\mathbf{x}}_n) \leq f(\bar{\mathbf{x}}_n) + \nabla f(\bar{\mathbf{x}}_n)^T(\tilde{\mathbf{x}}_n - \bar{\mathbf{x}}_n) + \frac{L_n}{2}\|\tilde{\mathbf{x}}_n - \bar{\mathbf{x}}_n\|^2_{P_n}, \tag{5.55}$$

where $\tilde{\mathbf{x}}_n$ and $\bar{\mathbf{x}}_n$ are defined as in (5.11) and (5.8), respectively. We recall that the resulting sequence $\{\alpha_n\}_{n\in\mathbb{N}}$ is monotonically nonincreasing. We start with an initial value of $L_0 = 0.1$ and iteratively increase it by a factor of $\delta = 0.8$ until condition (5.55) is satisfied, or $L_n > 1$, in which case we simply set $L_n = 1$ and $\alpha_n = \epsilon$. This choice is justified by the following remark.

Remark 5.3.2. *If $f(\mathbf{x}) = \frac{1}{2}\|A\mathbf{x} - \mathbf{b}^\delta\|^2$ and $P_n = A^TA + \nu_n I$, then condition (5.55) is satisfied for $L_n = 1$. Indeed, using a second-order Taylor expansion at $\mathbf{x} = \bar{\mathbf{x}}_n$, we have*

$$f(\tilde{\mathbf{x}}_n) = f(\bar{\mathbf{x}}_n) + \nabla f(\bar{\mathbf{x}}_n)^T(\tilde{\mathbf{x}}_n - \bar{\mathbf{x}}_n) + \frac{1}{2}(\tilde{\mathbf{x}}_n - \bar{\mathbf{x}}_n)A^TA(\tilde{\mathbf{x}}_n - \bar{\mathbf{x}}_n)^T.$$

From the fact that the matrix A^TA is positive semidefinite, we obtain

$$\begin{aligned} f(\tilde{\mathbf{x}}_n) &\leq f(\bar{\mathbf{x}}_n) + \nabla f(\bar{\mathbf{x}}_n)^T(\tilde{\mathbf{x}}_n - \bar{\mathbf{x}}_n) + \frac{1}{2}(\tilde{\mathbf{x}}_n - \bar{\mathbf{x}}_n)(A^TA + \nu_n I)(\tilde{\mathbf{x}}_n - \bar{\mathbf{x}}_n)^T \\ &= f(\bar{\mathbf{x}}_n) + \nabla f(\bar{\mathbf{x}}_n)^T(\tilde{\mathbf{x}}_n - \bar{\mathbf{x}}_n) + \frac{1}{2}\|\tilde{\mathbf{x}}_n - \bar{\mathbf{x}}_n\|_{P_n}^2. \end{aligned}$$

The choice of the β_n parameter must be coherent with Step 3 of Algorithm 7. Therefore, in both nonstationary versions of the NPDIT method, we select β_n at each iteration $n \in \mathbb{N}$ as follows:

$$\beta_n = \frac{\epsilon\nu_n}{8} = \frac{\epsilon\nu_n}{\|W\|^2},$$

where $\epsilon \in (0, 1)$. Finally, the value of γ_n for the inertial step was chosen following the strategy proposed in [26] and detailed below:

$$\gamma_n = \begin{cases} 0, & n = 0 \\ \min\left\{\gamma_n^{\text{FISTA}}, \frac{C\rho_n}{\|\mathbf{x}_n - \mathbf{x}_{n-1}\|}\right\}, & n = 1, 2, \ldots. \end{cases} \tag{5.56}$$

In all of the three considered examples, we chose $C = 0.1\|\mathbf{x}_1 - \mathbf{x}_0\|$ for the stationary version of NPDIT and for NPDIT$_{\text{D}}$, while we select $C = \|\mathbf{x}_1 - \mathbf{x}_0\|$ for the increasing case. Then, for all NPDIT implementations, we use the same value for ρ_n, which is $\rho_n = \frac{1}{n^{1.1}}$. In (5.56), the parameter γ_n^{FISTA} is defined by the usual FISTA rule (4.32), that is

$$t_0 = 1, \quad \begin{cases} t_{n+1} = \frac{1+\sqrt{1+4t_n^2}}{2}, \\ \gamma_n^{\text{FISTA}} = \frac{t_{n-1}}{t_{n+1}}, \end{cases} \qquad n = 0, 1, \ldots. \tag{5.57}$$

The number of primal–dual iterations is set for simplicity to $k_{\max} = 1$. This choice is coherent with the numerical experimentation carried out in [26] for a special instance of Algorithm 7, where it was observed that $k_{\max} = 1$ could be seen as a good trade-off between accuracy and complexity.

Finally, we remark that an automated method for computing the regularization parameter λ has not been implemented in the tests. Consequently, we will specify the value or values used for each example.

5.3.2 Experimental setup

The following numerical results were obtained using MATLAB R2023a on a MacBook Pro equipped with the M2 Pro chip and a 10-core CPU (6 performance cores and 4 efficiency cores), along with 16GB of RAM.

We provide three image deblurring examples in which we compare the three implementations of our NPDIT method with the following competitors:

- the NPD method described in Section §4.2.4 and proposed in [26], which can be seen as a special instance of Algorithm 7 obtained by setting $P_n \equiv I_d$, $\varsigma_n \equiv 1$, $\alpha_0 = 1$,

$\beta_n \equiv \beta < 1/8$ and γ_n computed according to (5.56) with $C = 10\|u_1 - u_0\|$; unlike Algorithm 7, NPD chooses a fixed steplength $\alpha_n \equiv \alpha_0$ without performing any backtracking procedure; note that the chosen value for α_0 is such that the backtracking condition at Step 7 of Algorithm 7 is automatically satisfied for all $n \geq 0$ with $\alpha_n = \alpha_0$;

- the preconditioned version of the Chambolle-Pock method (CP_{prec}), the popular primal–dual algorithm for composite convex optimization proposed in [121];
- a first order primal–dual method with linesearch (PD–LS) proposed by Malitsky and Pock in [106];
- the inexact version of FISTA with approximate proximal evaluations proposed in [27], whose practical implementation resembles Algorithm 7, in the sense that it can be seen as a nested primal–dual algorithm with extrapolation for solving problems of the form (4.34); however, unlike Algorithm 7, the number of inner iterations varies with the outer iteration according to an appropriate stopping criterion, the extrapolation parameter is computed as proposed by Chambolle and Dossal in [43], and there is no variable metric.

The blurred image is obtained by circular two-dimensional convolution, ensuring that the deblurring model is not affected by boundary ringing effects. Consequently, the numerical results only depend on the applied numerical method. Of course, in real applications, appropriate boundary conditions affecting the structure of the matrix A should be adopted, see [82].

In order to evaluate the performance of the different methods, we use the usual Relative Reconstruction Error (RRE) functional, defined as

$$\text{RRE}(\mathbf{x}_n) = \frac{\|\mathbf{x}_{\text{gt}} - \mathbf{x}_n\|}{\|\mathbf{x}_{\text{gt}}\|},$$

where $\mathbf{x}_{\text{gt}}$ represents the original image. The relative decrease of the objective function is computed by the quantity

$$\frac{R(\mathbf{x}_n) - R(\mathbf{x}^*)}{R(\mathbf{x}^*)},$$

where R is the objective function and $\mathbf{x}^*$ is the best reconstruction obtained among all the methods considered, achieving the minimum value of the objective function. This solution was precomputed by running all the methods for 10^3 iterations.

5.4 Numerical experiments

5.4.1 Example 1: Cameraman

In this example, we considered an image of a Cameraman with dimensions of 256×256. The image was blurred using a 10×10 Gaussian PSF with a standard deviation of 2. Additionally, white noise with an intensity level of 1% was added to the blurred image. This implies that

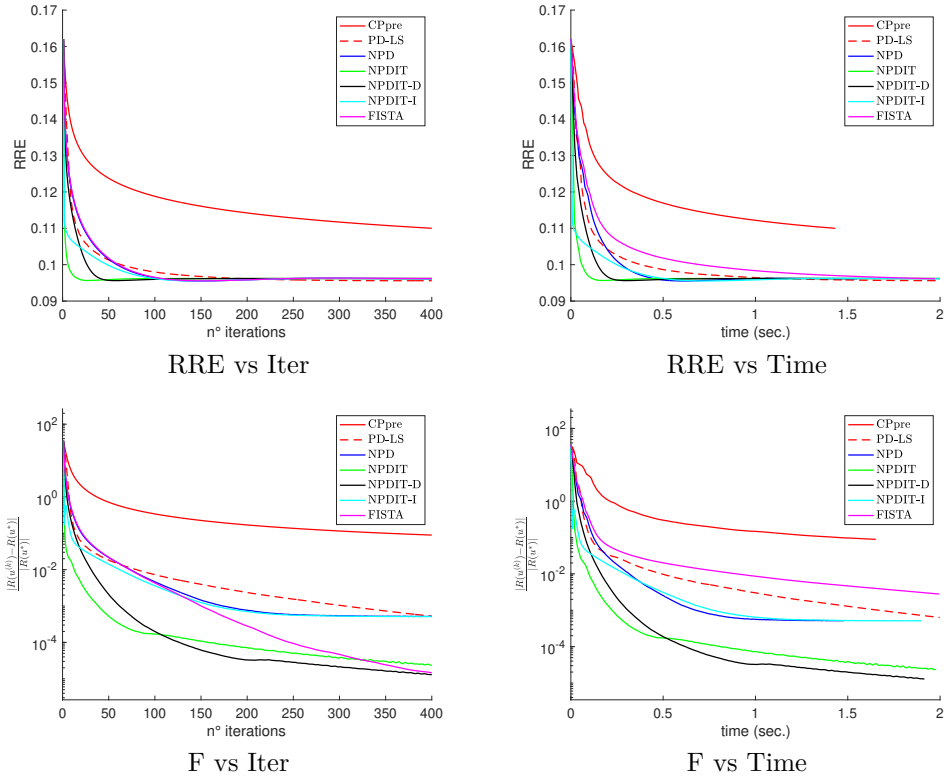


Figure 5.1: Example 1 – First row: RRE functional over 400 iterations and over 2 seconds of time. Second row: Relative decrease of the objective function over the number of iterations and time.

the norm of the noise is equal to 1% of the norm of the blurred image. For this example, we set $\lambda = 10^{-4}$ as the regularization parameter.

In the first row of Figure 5.1, we depict the behavior of the RRE functional over 400 iterations (left) and over 2 seconds of time (right) for all the methods considered. For the stationary NPDIT case, we set $\nu = 0.01$. We observe that our proposal, both stationary and non-stationary, outperforms all the other competitors. Indeed, we achieve lower values of the RRE in fewer iterations, which are also faster to compute compared to all the other cases. For NPDIT_I, i.e., the version of our proposal where ν_n is increasing, we note a significant speed-up in the initial iterations due to the small value of ν_n. However, as iterations progress, the performance worsens because ν_n approaches 1 and consequently slows down the convergence of the method.

In the second row of Figure 5.1, we compare the relative decrease of the objective function. Similarly, the stationary method and the non-stationary method NPDIT-D achieve better results than all the other competitors. However, there are two observations worth mentioning. The inexact version of FISTA proposed in [27] seems to perform comparably to NPDIT-D as the iterations progress, but the time required to achieve these results is significantly greater than that required by our proposal. Secondly, the performance of NPDIT_I is similar to the standard NPD algorithm. This is mainly due to the choice of the regular-

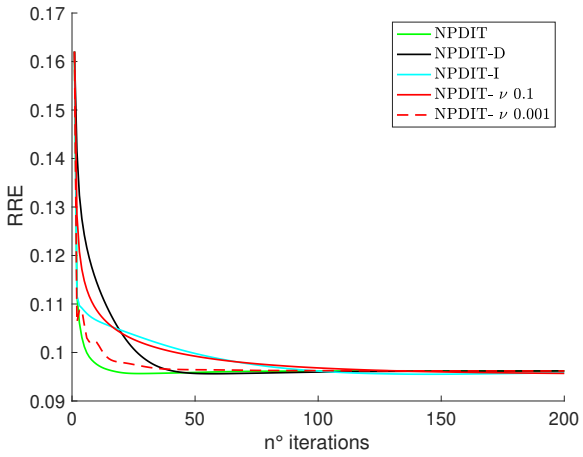


Figure 5.2: Example 1 – RRE for different values of ν for $\lambda = 10^{-4}$.

ization parameter λ. As we will show in the last part of the numerical examples, NPDIT$_{\text{I}}$ works better when the regularization parameter λ is underestimated.

Figure 5.2 shows the convergence behavior of the RRE functional for NPDIT$_I$, NPDIT$_D$, and the stationary variant of NPDIT equipped with three different values of the parameter ν. The plots for the non-stationary methods and the case with $\nu = 0.01$ are identical to those shown in Figure 5.1. For $\nu = 10^{-3}$, it is noticeable that the method introduces some instability and the speed of convergence is also reduced. When $\nu = 0.1$, the algorithm is stable again but it is slower than in the other cases. This demonstrates that the convergence speed and the stability of the method are affected by the choice of ν for stationary values $\nu_n = \nu$. Therefore, the decreasing sequence of ν_n in equation (5.53) provides a good trade-off without requiring the estimation of ν.

The reconstructed images shown in Figure 5.3 are obtained at various iterations for both the NPD and the NPDIT method with $\nu = 0.01$. For completeness, the upper part of the figure includes the true image, the PSF, and the observed image $\mathbf{b}^\delta$. Additionally, the achieved RRE value is reported for each reconstruction. Notably, the NPDIT method achieves satisfactory reconstructions with just 20 iterations, while the NPD method requires at least 100 iterations to achieve a similar level of accuracy.

5.4.2 Example 2: Peppers

In this second example, we examined the image of Peppers with dimensions 256×256 and applied an out-of-focus PSF with dimensions 10×10 for blurring. Additionally, we introduced white Gaussian noise with an intensity level of 1% to the image. For this case, we choose the regularization parameter $\lambda = 10^{-4}$.

The comparison between all the methods follows the same structure as in the previous case. In the first row of Figure 5.4, we present the RRE functional, while in the second row we focus on the relative decrease of the objective function. In this example, we observe that

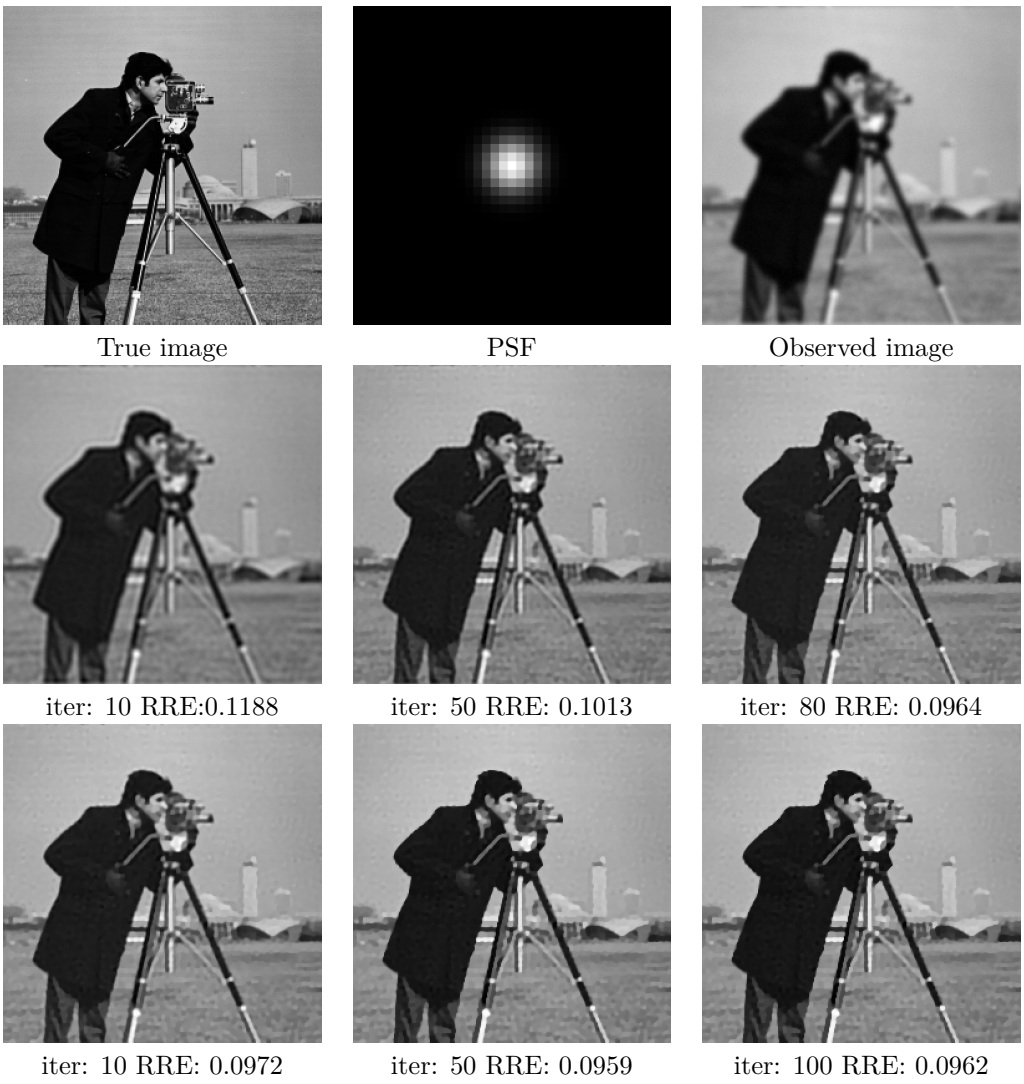


Figure 5.3: Example 1 – Reconstruction obtained at different iterations. First row: original data. Second row: images obtained with NPD. Third row: images obtained with NPDIT ($\nu = 0.01$).

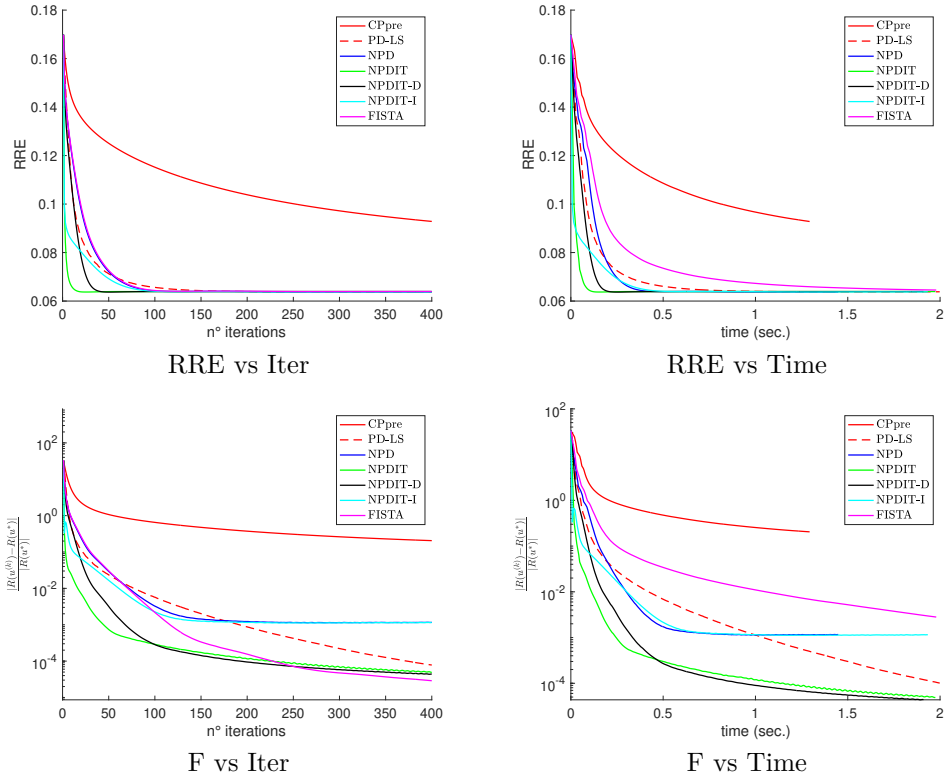


Figure 5.4: Example 2 – First row: RRE functional over 400 iterations and over 2 seconds of time. Second row: Relative decrease of the objective function over the number of iterations and time.

both the NPDIT method with $\nu = 0.01$ and the non-stationary version NPDIT_D achieve excellent results. Concerning the NPDIT_I algorithm, we notice a rapid decrease in the first iterations for the objective function, followed by a deterioration in performance as the iterations increase. Again, we note that the FISTA method outperforms all of our proposals in the relative decrease of the objective function after 400 iterations. However, as in Example 1, the time required is significantly greater than those of the stationary NPDIT and the NPDIT_D method.

Finally, in Figure 5.5, we display the reconstructions obtained by the NPD method and the NPDIT with $\nu = 0.01$ fixed. Once again, in the first row, we present the true image, the PSF used for blurring, and the observation corrupted by noise. For this example, it is evident that the stationary version NDPIT requires only 10 iterations to achieve excellent results, while the NPD algorithm needs almost four times more iterations to obtain the same quality in the reconstruction.

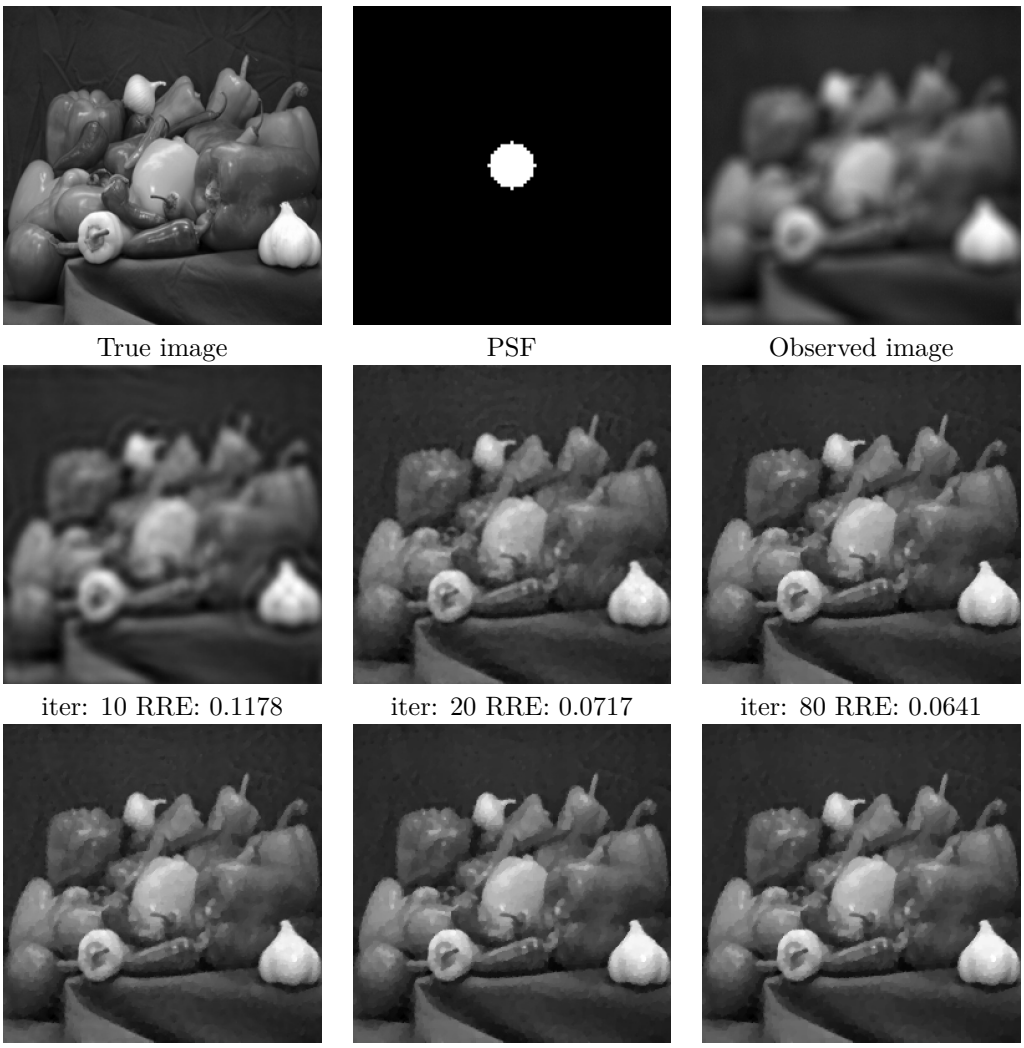


Figure 5.5: Example 2 – Reconstruction obtained at different iterations. First row: original data. Second row: images obtained with NPD. Third row: images obtained with NPDIT ($\nu = 0.01$).

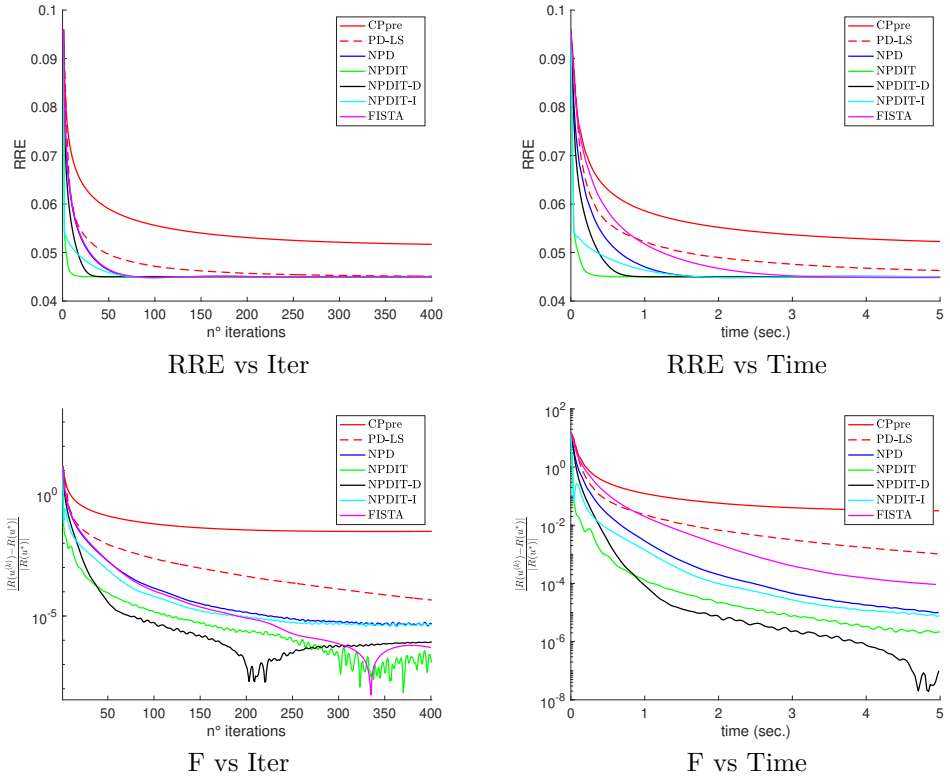


Figure 5.6: Example 3 – First row: RRE functional over 400 iterations and over 5 seconds of time. Second row: Relative decrease of the objective function over the same number of iterations and time as before.

5.4.3 Example 3:

In this final numerical example, we replace the TV term with the Overlapping Group sparsity TV (OG-TV) regularizer proposed in [102], which is defined as

$$h(W\mathbf{x}) = \sum_{i=1}^{d^2} \|B_i \nabla^h \mathbf{x}\| + \|B_i \nabla^v \mathbf{x}\|,$$

where $\nabla^h, \nabla^v \in \mathbb{R}^{d^2 \times d^2}$ are the horizontal and vertical discrete gradient operators, while $B_i \in \mathbb{R}^{m^2 \times d^2}$ is a unitary diagonal matrix such that $B_i\mathbf{x}$ is the vector whose entries are the pixel i of the image $\mathbf{x}$ and its $m^2 - 1$ first neighbors. In this case we have that $\|W\|^2 = 8m^2$. In order to test the performances of our methods on this new problem, we consider the jetplane image, which has been blurred and artificially corrupted with white Gaussian noise as in Example 1. For this case, we choose $\lambda = 5 \times 10^{-5}$ as the regularization parameter. The analysis of this experiment follows the same structure as in the previous examples. In the first row of Figure 5.6, we report the RRE functional as a function of the number of iterations (left side) and time (right side). The second row is devoted to the relative decrease of the objective function, again depending on the number of iterations and time. Even in

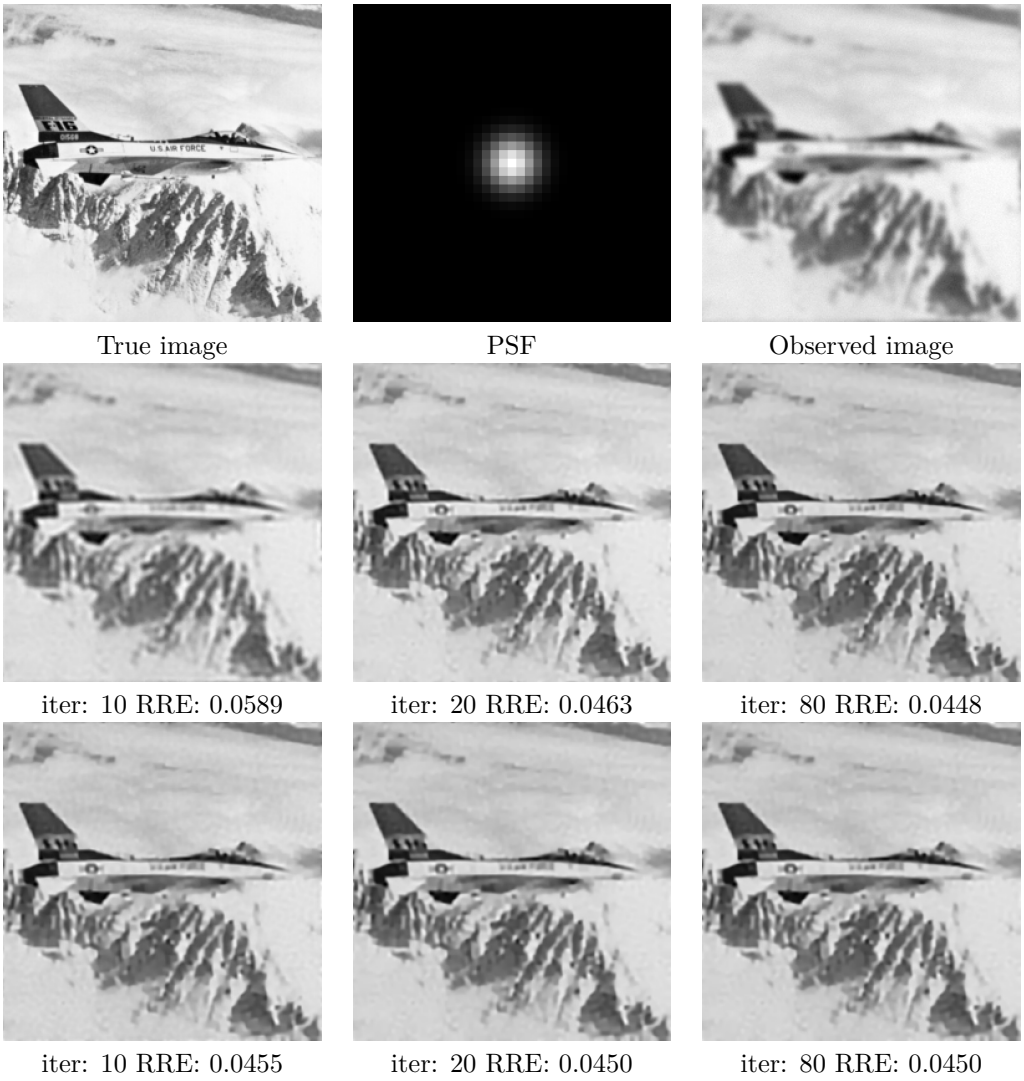


Figure 5.7: Example 3 – Reconstruction obtained at different iterations. First row: original data. Second row: images obtained with NPD. Third row: images obtained with NPDIT ($\nu = 0.01$).

this case, we can clearly state that all our proposals outperform the other methods in terms of both quality and speed.

Lastly, in Figure 5.7, we compare the reconstruction obtained with the NPD method and the stationary NPDIT method equipped with $\nu = 0.01$. The final images were computed after different numbers of iterations, and we can see that, even in this case, NPDIT achieves remarkable reconstructions even after just 10 iterations.

5.4.4 Stability test

As a final illustration, we investigated the robustness of the methods with respect to the regularization parameter λ under the condition of a stationary sequence $\nu_n = \nu$. To achieve this, we explored different values of this parameter for each example presented earlier. Fur-

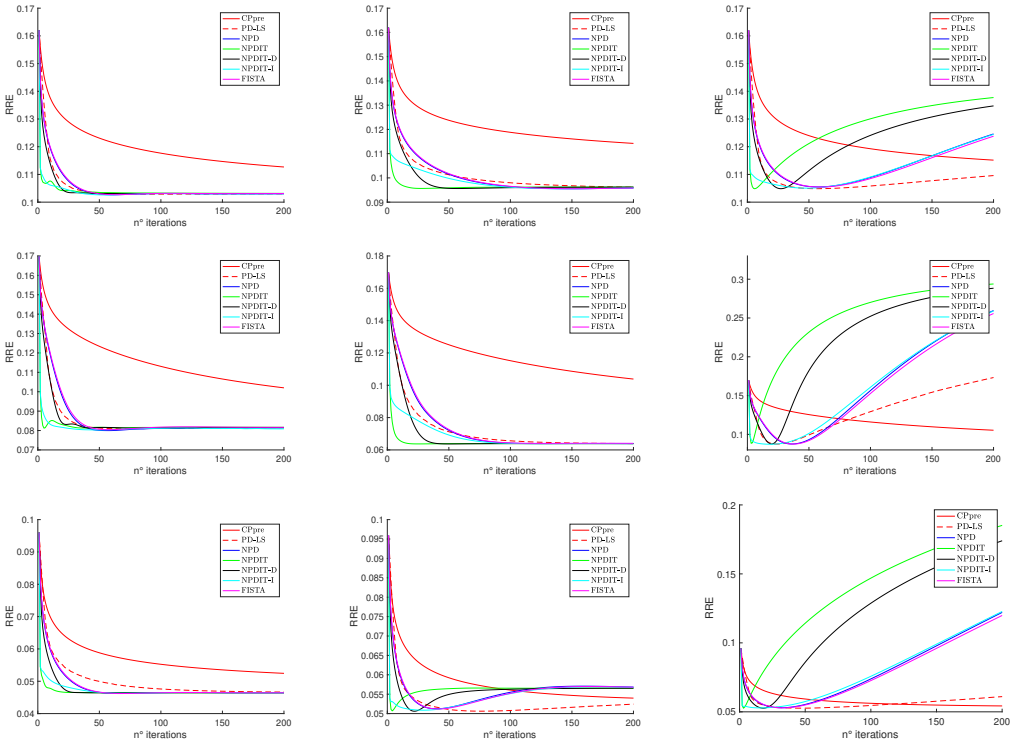


Figure 5.8: From top to bottom: RRE functional with respect to the number of iterations for example 1 (top row), example 2 (middle row), and example 3 (bottom row). From left to right: comparison for different values of λ, i.e., $\lambda \in \{10^{-3}, 10^{-4}, 10^{-5}\}$ for the first two rows, $\lambda \in \{10^{-4}, 10^{-5}, 10^{-6}\}$ for the third one.

thermore, we conducted a detailed analysis of the behavior of the proposed method with respect to the two non-stationary sequences employed in NPDIT$_D$ and NPDIT$_I$.

The comparison among the methods is carried out by assessing the quality of reconstructions using the RRE functional. Figure 5.8 presents a summary of the results as follows: each row corresponds to one of the mentioned examples. In particular, the i-th row corresponds to Example i, where $i = 1, 2, 3$. Each column is associated with a specific value of $\lambda \in \{10^{-3}, 10^{-4}, 10^{-5}\}$ for the first two rows and $\lambda \in \{10^{-4}, 10^{-5}, 10^{-6}\}$ for the last one.

In accordance with the previous results, the NPDIT$_D$ method exhibits behavior very similar to that of NPDIT, avoiding the selection of the stationary parameter ν. Regarding the NPDIT$_I$ method, it is observed to be more beneficial in cases of underestimation of the regularization parameter λ, as indicated by the theoretical analysis in [59] for the Iterated Tikhonov method. Specifically, from the rightmost column, we observe greater stability around the minimum point compared to other methods. However, whenever λ is underestimated, it is advisable to implement an effective stopping criterion (e.g., the discrepancy principle) to terminate the process before the RRE grows excessively.

5.5 Conclusions

We have formulated and analysed a nested primal–dual method tailored for solving regularized convex optimization problems. Our proposed approach approximates a variable metric proximal–gradient step with extrapolation by executing a fixed number of primal-dual iterates, while dynamically adjusting the steplength parameter through a specialized backtracking procedure. The convergence of the iterates sequence towards a solution of the problem has been established by assuming a relaxed monotonicity condition on the scaling matrices, as well as a specific shrinking criterion on the extrapolation parameters. Extensive numerical experiments have shown that our algorithm performs well in comparison to other similar competitors on some Total-Variation regularized least-squares problems arising in image deblurring, especially when it is equipped with scaling matrices inspired by the Iterated Tikhonov method.

Future work may include the application of our algorithm to other imaging problems such as computer tomography, its adaptation to more accurate boundary conditions for image deblurring, and its extension to more general optimization problems. In particular, for applications not addressed in this work, the matrix-vector product with the proposed scaling matrices might be computationally expensive. Therefore, we plan to investigate its approximation in suitable subspaces from both a theoretical and practical standpoint. Moreover, we will investigate the extension of our proposed algorithm to problems where constraints on the primal variable are imposed, and the definition of novel nested primal–dual algorithms where the backtracking procedure based on the Descent Lemma is replaced by a linesearch along the descent direction.

A Preconditioned version of NPD for image deblurring

This final chapter of the thesis discusses a reinterpretation of the NPDIT method applied to the image deblurring problem. Specifically, we will show that the variable metric matrix $P_n = A^T A + \nu_n I_d$ acts as a right preconditioner for the linear system of equations

$$A\mathbf{x} = \mathbf{b}^\delta, \tag{6.1}$$

where, as before, $A \in \mathbb{R}^{s\times d}$ represents the discretization of a space invariant convolution operator, $\mathbf{b}^\delta \in \mathbb{R}^s$ is the observed image corrupted by white Gaussian noise $\boldsymbol{\eta}_\delta$, and $\mathbf{x} \in \mathbb{R}^d$ denotes an unknown two–dimensional image with d pixels. Recall that we assume the vector $\mathbf{b}^\delta$ satisfies

$$\mathbf{b}^\delta = \mathbf{b} + \boldsymbol{\eta}_\delta, \qquad \|\boldsymbol{\eta}_\delta\| \leq \delta,$$

where $\mathbf{b}$ represents the unobserved noise-free blurred image, and $\delta > 0$ is an upper bound on the noise level. To address the ill-posed nature of A and the presence of noise, we adopt the same variational approach introduced in Section §5.4, which leads to solving

$$\arg\min_{\mathbf{x}\in\mathbb{R}^d} \frac{1}{2}\|A\mathbf{x} - \mathbf{b}^\delta\|^2 + h(W\mathbf{x}), \tag{6.2}$$

under the following assumptions:

(i) $h : \mathbb{R}^{d'} \to \overline{\mathbb{R}}$ is a proper, convex, lsc function;

(ii) $W \in \mathbb{R}^{d'\times d}$ and there exists $\mathbf{x}_0$ such that $W\mathbf{x}_0 \in \operatorname{relint}(\operatorname{dom}(h))$;

(iii) Problem (6.2) has at least one solution.

Note that the data fidelity term $f(\mathbf{x}) = \frac{1}{2}\|A\mathbf{x} - \mathbf{b}^\delta\|^2$ trivially satisfies Hypothesis 4.2.39 (i). Given a matrix P that approximates A and is well-conditioned, left and right preconditioning of the linear system (6.1) are defined as follows:

$$\begin{aligned} P^{-1}A\mathbf{u} &= P^{-1}\mathbf{b}^\delta, && \text{(left preconditioning)},\\ AP^{-1}P\mathbf{u} &= \mathbf{b}^\delta, && \text{(right preconditioning)}. \end{aligned}$$

As mentioned earlier, we will reformulate the NPDIT method applied to problem (6.2) as a right preconditioning strategy. Consequently, in this chapter, we focus on the left preconditioning approach. This can be understood as replacing the gradient step in the proximal evaluation with a higher-order method, where the descent direction $-\nabla f(\mathbf{x})$ is premultiplied by a suitable preconditioner P. The key difference compared to the NPDIT method is that, in the left preconditioning approach, the matrix P only affects the data fidelity term and not the proximity operator of the non-smooth term $h \circ W$. By combining this with extrapolation techniques and allowing for inexact proximal evaluations, we derive the following iterative scheme

$$\begin{cases} \bar{u}_n = u_n + \gamma_n(u_n - u_{n-1}), \\ u_{n+1} \approx \operatorname{prox}_{\alpha_n h \circ W}\left(\bar{u}_n - \alpha_n P_n^{-1} A^T(A\bar{u}_n - \mathbf{b}^\delta)\right), \end{cases} \qquad \forall n \geq 0, \tag{6.3}$$

where $P_n \in \mathbb{R}^{d\times d}$ is the preconditioning matrix and $\gamma_n \geq 0$ is the extrapolation parameter.

In the following, we will show that iterative schemes like (6.3) result in faster convergence compared to standard variable metric approaches. Moreover, the numerical results provided in the final section demonstrate that the reconstruction quality is comparable to that achieved by the NPDIT algorithm. Additionally, we emphasize that a sufficiently accurate proximal evaluation is essential if the shifting parameter ν_n is small or if one aims to eliminate the extrapolation step. With appropriate choices of the non-smooth term $h \circ W$, the preconditioner P_n, and the extrapolation parameter γ_n, the scheme (6.3) aligns with well-known algorithms such as ISTA, FISTA, and ITTA [56, 12, 84].

6.1 Preconditioned Nested Primal–Dual (PNPD)

This section will be devoted to deriving and analyzing our proposal and discussing possible strategies for choosing a suitable preconditioner matrix $P \in \mathcal{S}_+(\mathbb{R}^d)$. As an initial step, in the first part we prove that the variable metric approach can be indeed reformulated as a right preconditioning strategy. Hence, in the remaining part we show that the left preconditioning approach result in an iterative scheme faster than NPD requiring less computational effort at each iteration than NPDIT. However, a possible drawback of our approach is that it changes the norm of the data fidelity term computing a solution potentially different from the one of (6.2). This issue will be addressed in the final part of this section, where we introduce non–stationarity in the preconditioner letting the sequence of preconditioners P_n converge to the identity operator. This will allow us to compare our proposal with standard approaches that use the ℓ^2-norm in the data fidelity term.

Throughtout this chapter we will always refer to the variational model (6.2) since we will focused only on the image deblurring problem.

6.1.1 Variable metric approach as right preconditioning

Let $R \in \mathbb{R}^{d\times d}$ be invertible, applying the right preconditioning strategy to the original linear system of equation (6.1), we obtain the equivalent formulation of the variational problem (6.2), that is

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}\in\mathbb{R}^d} \frac{1}{2}\|AR^{-1}R\mathbf{x} - \mathbf{b}^\delta\|^2 + h(W\mathbf{x}). \tag{6.4}$$

Setting $\tilde{A} = AR^{-1}$, $\mathbf{z} = R\mathbf{x}$, and $\tilde{W} = WR^{-1}$, the solution $\hat{\mathbf{x}}$ of (6.2) is given by

$$\hat{\mathbf{x}} = R^{-1}\hat{\mathbf{z}},$$

where

$$\hat{\mathbf{z}} = \arg\min_{\mathbf{z}\in\mathbb{R}^d} \frac{1}{2}\|\tilde{A}\mathbf{z} - \mathbf{b}^\delta\|^2 + h(\tilde{W}\mathbf{z}) = \arg\min_{\mathbf{z}\in\mathbb{R}^d} \tilde{f}(\mathbf{z}) + h(\tilde{W}\mathbf{z}). \tag{6.5}$$

The NPD method (4.43) applied to problem (6.5) results in

$$\begin{cases} \bar{\mathbf{z}}_n = \mathbf{z}_n + \gamma_n(\mathbf{z}_n - \mathbf{z}_{n-1}), \\ \mathbf{z}_{n+1} = \operatorname{prox}_{\alpha_n h\circ\tilde{W}}(\bar{\mathbf{z}}_n - \alpha_n\nabla\tilde{f}(\bar{\mathbf{z}}_n)), \end{cases} \tag{6.6}$$

where, for simplicity of notation, the "$\approx$" symbol has been replaced with the "$=$" symbol assuming that the proximity operator of the non-differentiable part can be computed exactly. In the end, we will revert to the approximation notation considering the same approximation scheme as the inexact variable case.

Let $\mathbf{x}_n = R^{-1}\mathbf{z}_n$, clearly $\lim_n \mathbf{z}_n = \hat{\mathbf{z}}$ implies $\lim_n \mathbf{x}_n = \hat{\mathbf{x}}$. Therefore, we want to write the scheme in equation (6.6) in terms of the sequence $\{\mathbf{x}_n\}_{\in\mathbb{N}}$.

Lemma 6.1.1. *Let h be a convex lower semicontinuous function. Let $W \in \mathbb{R}^{d'\times d}$, $R \in \mathbb{R}^{d\times d}$ invertible, and $\tilde{W} = WR^{-1}$, then*

$$\operatorname{prox}_{\alpha h\circ\tilde{W}}(\mathbf{z}) = R\operatorname{prox}^{R^TR}_{\alpha h\circ W}(R^{-1}\mathbf{z}).$$

Proof. We observe that, for a function $g : \mathbb{R}^d \to \mathbb{R}$, it holds

$$\hat{\mathbf{z}} \in \arg\min_{\tilde{z}} g(R^{-1}\tilde{\mathbf{z}}) \;\Leftrightarrow\; R^{-1}\hat{z} \in \arg\min_{\tilde{\mathbf{x}}} g(\tilde{\mathbf{x}}) \;\Leftrightarrow\; \hat{\mathbf{z}} \in R\arg\min_{\tilde{\mathbf{x}}} g(\tilde{\mathbf{x}}),$$

Therefore, by fixing $\mathbf{x} = R^{-1}\mathbf{z}$, we have

$$\begin{aligned} \operatorname{prox}_{\alpha h\circ\tilde{W}}(\mathbf{z}) &= \arg\min_{\tilde{\mathbf{z}}} h(WR^{-1}\tilde{\mathbf{z}}) + \frac{1}{2\alpha}\|\mathbf{z} - \tilde{\mathbf{z}}\|^2 \\ &= R\arg\min_{\tilde{\mathbf{x}}}\; h(W\tilde{\mathbf{x}}) + \frac{1}{2\alpha}\|R(\mathbf{x} - \tilde{\mathbf{x}})\|^2 \\ &= R\operatorname{prox}^{R^TR}_{\alpha h\circ W}(R^{-1}\mathbf{z}). \end{aligned}$$

□

Recalling that $f(\mathbf{x}) = \frac{1}{2}\|A\mathbf{x} - \mathbf{b}^\delta\|^2$ and $\tilde{f}(\mathbf{x}) = \frac{1}{2}\|\tilde{A}\mathbf{x} - \mathbf{b}^\delta\|^2$, it holds

$$\nabla \tilde{f}(R\mathbf{x}) = R^{-T}\nabla f(\mathbf{x}). \tag{6.7}$$

Therefore, using Lemma 6.1.1 and equation (6.7), from the second equation in (6.6), for $\bar{\mathbf{x}} = R^{-1}\bar{\mathbf{z}}$ we have

$$\begin{aligned}\mathbf{x}_{n+1} &= R^{-1}\mathbf{z}_{n+1}\\ &= \mathrm{prox}^{R^TR}_{\alpha_n h\circ W}(R^{-1}\bar{\mathbf{z}}_n - \alpha_n R^{-1}\nabla\tilde{f}(\bar{\mathbf{z}}_n))\\ &= \mathrm{prox}^{R^TR}_{\alpha_n h\circ W}(\bar{\mathbf{x}}_n - \alpha_n (R^TR)^{-1}\nabla f(\bar{\mathbf{x}}_n)).\end{aligned}$$

Finally, adding also the extrapolation step and replacing back the approximation symbol in the second equation of (6.6), the NPD method applied to problem (6.4) is given by the iterative scheme

$$\begin{cases}\bar{\mathbf{x}}_n = \mathbf{x}_n + \gamma_n(\mathbf{x}_n - \mathbf{x}_{n-1}),\\ \mathbf{x}_{n+1} \approx \mathrm{prox}^{R^TR}_{\alpha_n h\circ W}(\bar{\mathbf{x}}_n - \alpha_n (R^TR)^{-1}\nabla f(\bar{\mathbf{x}}_n)).\end{cases} \tag{6.8}$$

Note that, given a stationary preconditioner $P_n = P \in \mathcal{D}_\varsigma$, by choosing $R = P^{\frac{1}{2}}$ such that $R^TR = P$, then the iteration (6.8) is exactly the variable metric scheme (6.3).

6.1.2 Preconditioned Nested Primal-Dual (PNPD)

In the previous section we discussed how NPDIT applied to problem (6.2) can be interpreted as a right preconditioning strategy. Hence, we can now analyze the left preconditioning approach.

Given $S \in \mathcal{S}_+(\mathbb{R}^s)$, we consider the optimization problem

$$\arg\min_{\mathbf{x}\in\mathbb{R}^d} f_S(\mathbf{x}) + h(W\mathbf{x}) = \arg\min_{\mathbf{x}\in\mathbb{R}^d} \frac{1}{2}\|A\mathbf{x} - \mathbf{b}^\delta\|^2_{S^{-1}} + h(W\mathbf{x}). \tag{6.9}$$

In the data fidelity term

$$f_S(\mathbf{x}) = \frac{1}{2}\|S^{-\frac{1}{2}}(Au - \mathbf{b}^\delta)\|^2,$$

the linear operator $S^{\frac{1}{2}}$ can be interpreted as a left preconditioner for the linear system (6.1). If we further assume that there exists $P \in \mathcal{S}_+(\mathbb{R}^d)$ such that

$$P^{-1}A^T = A^TS^{-1}, \tag{6.10}$$

then it holds

$$\nabla f_S(\mathbf{x}) = A^TS^{-1}(A\mathbf{x} - \mathbf{b}^\delta) = P^{-1}\nabla f(\mathbf{x}), \tag{6.11}$$

where $f(\mathbf{x}) = \frac{1}{2}\|A\mathbf{x} - \mathbf{b}^\delta\|^2$ as in equation (6.2).

Therefore, applying the inexact proximal gradient approach (4.43) to problem (6.9), under

the assumption (6.10), we obtain a two-steps iterative scheme, named *Preconditioned Nested Primal-Dual* (PNPD), defined as

$$\begin{cases} \bar{\mathbf{x}}_n = \mathbf{x}_n + \gamma_n(\mathbf{x}_n - \mathbf{x}_{n-1}), \\ \mathbf{x}_{n+1} \approx \operatorname{prox}_{\alpha_n h \circ W}(\bar{\mathbf{x}}_n - \alpha_n P^{-1}\nabla f(\bar{\mathbf{x}}_n)). \end{cases} \tag{6.12}$$

The first difference between the PNPD method and NPDIT is that we use the standard definition of the proximity operator for the non–differentiable part $h \circ W$. As a result, the scaling matrix P no longer affects the dual variable in the nested iteration nor the dual step length β_n, which now depends only on the norm of the operator $W^T W$, as in NPD. This implies that in PNPD the preconditioner acts only on the primal part, whereas in NPDIT it also influences the dual part. Consequently, as the number of inner iterations increases, the computational cost of PNPD becomes lower compared to that of NPDIT. However, the main difference between the PNPD method and NPDIT lies in the fact that they compute solutions to different variational problems. Specifically, the PNPD strategy solves problem (6.9), while the NPDIT method solves problem (6.2).

Algorithm 8 reports the pseudocode of the proposed PNPD method.

Algorithm 8: PNPD

Choose $k_{\max} \in \mathbb{N}$, $\mathbf{x}_{-1} \in \mathbb{R}^d$, $\mathbf{x}_0 = \mathbf{x}_{-1}$, $\mathbf{y}_{-1}^{k_{\max}} \in \mathbb{R}^{d'}$, $P \in \mathcal{S}_+(\mathbb{R}^d)$, $0 < \beta < \frac{1}{\|W\|^2}$, $0 < \alpha < \frac{1}{L_S}$.

For $n = 0, 1, \ldots$

1. Choose $\gamma_n \geq 0$ and compute the extrapolated point
$$\bar{\mathbf{x}}_n = \mathbf{x}_n + \gamma_n(\mathbf{x}_n - \mathbf{x}_{n-1}). \tag{6.13}$$
2. Set $\mathbf{y}_n^0 = \mathbf{y}_{n-1}^{k_{\max}}$.
3. Compute $k_{\max}$ primal-dual iterates:
   For $k = 0, 1, \ldots, k_{\max} - 1$
$$\mathbf{x}_n^k = \bar{\mathbf{x}}_n - \alpha_n P^{-1}\nabla f(\bar{\mathbf{x}}_n) - \alpha_n W^T \mathbf{y}_n^k \tag{6.14}$$
$$\mathbf{y}_n^{k+1} = \operatorname{prox}_{\beta\alpha_n^{-1} h^*}(\mathbf{y}_n^k + \beta\alpha_n^{-1} W \mathbf{x}_n^k). \tag{6.15}$$
4. Compute $\mathbf{x}_n^{k_{\max}} = \bar{\mathbf{x}}_n - \alpha_n P^{-1}\nabla f(\bar{\mathbf{x}}_n) - \alpha_n W^T \mathbf{y}_n^{k_{\max}}$.
5. Compute the next iterate as
$$\tilde{\mathbf{x}}_n = \frac{1}{k_{\max}} \sum_{k=1}^{k_{\max}} \mathbf{x}_n^k. \tag{6.16}$$

Currently, the preconditioner P is fixed for each iteration n, but later, we will show how it can be chosen non-stationary varying with n.

To conclude this first part regarding the PNPD method, we prove a convergence result toward the solution of the initial problem (6.9) under suitable assumptions.

Theorem 6.1.2 (Convergence of PNPD)**.** *Let $f(\mathbf{x}) = \frac{1}{2}\|A\mathbf{x} - \mathbf{b}^\delta\|^2$. Suppose that h and W*

satisfy Assumption 4.2.39. Let $\{(\mathbf{x}_n, \mathbf{y}_n^0)\}_{n\in\mathbb{N}}$ *be the primal–dual sequence generated by the PNPD method (Algorithm 8) with* $\alpha_n = \alpha \in \left(0, \|P^{-1}A^TA\|^{-1}\right]$ *and* $\beta_n = \beta \in (0, \|W\|^{-2})$ *for all* $n \in \mathbb{N}$*. Suppose also that the inertial parameters* $\{\gamma_n\}_{n\in\mathbb{N}}$ *satisfy the condition (4.44) and that* $S \in \mathcal{S}_+(\mathbb{R}^s)$ *and* $P \in \mathcal{S}_+(\mathbb{R}^d)$ *satisfy equation (6.10).*

Then, the following statements hold:

(i) the sequence $\{(\mathbf{x}_n, \mathbf{y}_n^0)\}_{n\in\mathbb{N}}$ *is bounded;*

(ii) the primal sequence $\{\mathbf{x}_n\}_{n\in\mathbb{N}}$ *converges to a solution of the initial problem (6.9).*

Proof. As anticipated, the NPD method applied to the problem (6.9) gives exactly the iteration of PNPD. We only have to prove that ∇f_S is L_S-Lipschitz continuous with $L_S = \|P^{-1}A^TA\|$. Thanks to equation (6.11), we have

$$\|\nabla f_S(\mathbf{x}) - \nabla f_S(\mathbf{z})\| = \|P^{-1}\nabla f(\mathbf{x}) - P^{-1}\nabla f(\mathbf{z})\| \leq \|P^{-1}A^TA\|\|\mathbf{x} - \mathbf{z}\|.$$

The thesis follows applying Theorem 4.2.43 to problem (6.9). □

6.1.3 A polynomial choice for P

A reasonable choice for the preconditioner P and the associated matrix S, that satisfies the condition (6.10), is

$$P = A^TA + \nu I, \qquad S = AA^T + \nu I, \tag{6.17}$$

with $\nu > 0$. This is inspired by the iterated Tikhonov method, which coincides with the Levenberg–Marquardt method applied to linear problems [62]. More in general, we can show that the identity in equation (6.10) is satisfied whenever P is in the set of some particular polynomials of A^TA and S is a corresponding polynomial of AA^T.

Defined the set of polynomials with non-negative coefficients as

$$\mathcal{Q} = \left\{ \sum_{i=0}^{k} c_i x^i \mid c_i \geq 0,\ c_0 > 0,\ k \in \mathbb{N} \right\}, \tag{6.18}$$

and the set of matrices

$$\mathcal{P} = \left\{ P \in \mathbb{R}^{d\times d} \mid P = p(A^TA) \ \wedge\ p \in \mathcal{Q} \right\}, \tag{6.19}$$

if $P \in \mathcal{P}$, then, by definition $P = p(A^TA)$ for some $p \in \mathcal{Q}$, and thus $P \in \mathcal{S}_+(\mathbb{R}^d)$ since $c_0 > 0$. We also observe that for each $P = p(A^TA) \in \mathcal{P}$, we can define a corresponding $S = p(AA^T) \in \mathcal{S}_+(\mathbb{R}^s)$, which satisfies the equation (6.10). Then, the following results hold.

Proposition 6.1.3. *Let* $A \in \mathbb{R}^{s\times d}$*,* $P = p(A^TA) \in \mathcal{P}$ *and* $S = p(AA^T)$*. Then it holds*

$$P^{-1}A^T = A^TS^{-1}. \tag{6.20}$$

Proof. Since $P \in \mathcal{S}_+(\mathbb{R}^d)$ and $S \in \mathcal{S}_+(\mathbb{R}^s)$, they are invertible. Moreover, it holds

$$A^T S = A^T p(AA^T) = \sum_{i=0}^{k} c_i A^T (AA^T)^i = \sum_{i=0}^{k} c_i (A^T A)^i A^T = p(A^T A) A^T = P A^T,$$

which is equivalent to (6.20). □

Proposition 6.1.4. *Let $p(x) = \sum_{i=0}^{k} c_i x^i \in \mathcal{Q}$, $A \in \mathbb{R}^{s\times d}$, and $P = p(A^T A) \in \mathcal{P}$. Then, it holds*

$$\|P^{-1} A^T A\| \leq \left(c_1 + c_0 \|A\|^{-2}\right)^{-1}. \tag{6.21}$$

Proof. Using the singular value decomposition of $A = U\Sigma V^T$, we have $A^T A = V \Sigma^T \Sigma V^T$, and hence

$$P^{-1} A^T A = p(A^T A)^{-1} A^T A = p(V\Sigma^T \Sigma V^T)^{-1} V \Sigma^T \Sigma V^T = V D V^T,$$

where $D = p(\Sigma^T\Sigma)^{-1}\Sigma^T\Sigma \in \mathbb{R}^{d\times d}$ is a diagonal matrix. Let r be the rank of A and $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$ its positive singular values, then the diagonal entries of D are

$$d_j = \begin{cases} \frac{\sigma_j^2}{p(\sigma_j^2)}, & j = 1, \dots, r, \\ 0, & j = r+1, \dots, d. \end{cases} \tag{6.22}$$

Since $c_i \geq 0$ for $i = 0, \dots, k$, we have

$$p(\sigma_j^2) \geq c_1 \sigma_j^2 + c_0. \tag{6.23}$$

Dividing both sides of equation (6.23) by σ_j^2, we have

$$\frac{p(\sigma_j^2)}{\sigma_j^2} \geq c_1 + \frac{c_0}{\sigma_j^2} \geq c_1 + \frac{c_0}{\sigma_1^2} = c_1 + c_0 \|A\|^{-2},$$

which is equivalent to

$$\frac{\sigma_j^2}{p(\sigma_j^2)} \leq \left(c_1 + c_0 \|A\|^{-2}\right)^{-1}. \tag{6.24}$$

The thesis follows by observing that

$$\|P^{-1} A^T A\| = \|D\| = \max_{j=1,\dots,r} \frac{\sigma_j^2}{p(\sigma_j^2)} \leq \left(c_1 + c_0 \|A\|^{-2}\right)^{-1},$$

thanks to equation (6.24). □

From Proposition 6.1.4 and Theorem 6.1.2 follows that, if $P \in \mathcal{P}$, the convergence of PNPD is guaranteed whenever we choose $0 < \alpha < \left(c_1 + c_0 \|A\|^{-2}\right)$.

6.1.4 PNPD as an inexact version of FISTA and ITTA

When the proximity operator $\text{prox}_{\alpha_k h\circ W}$ in equation (6.12) is known in closed form, then the PNPD method (6.12) can be rewritten replacing the "$\approx$" with the "$=$" symbol. This is the case explored in this subsection, where we consider

$$h(Wu) = \lambda\|u\|_1. \tag{6.25}$$

Recalling Example 4.2.31 described in Section §4.1, we have that

$$\text{prox}_{\alpha_n h\circ W}(\mathbf{x}) = \text{prox}_{\alpha_n\lambda\|\cdot\|_1}(\mathbf{x}) = \mathcal{T}(\alpha_n\lambda, \mathbf{x}), \tag{6.26}$$

where we recall that $\mathcal{T}(\alpha, \mathbf{x})$ is the soft-thresholding function defined component–wise as

$$(\mathcal{T}(\alpha, \mathbf{x}))_i = \text{sign}(x_i)\max\{|x_i| - \alpha, 0\}.$$

Note that in the regularization term (6.25), usually $\mathbf{x}$ represents the wavelet coefficients of the image, and the matrix A in equation (6.2) is given by the product between the convolution operator and the inverse wavelet transform.

Setting $P = I$, equation (6.12) becomes

$$\begin{cases} \bar{\mathbf{x}}_n = \mathbf{x}_n + \gamma_n(\mathbf{x}_n - \mathbf{x}_{n-1}), \\ \mathbf{x}_{n+1} = \mathcal{T}(\alpha_n\lambda, \bar{\mathbf{x}}_n - \alpha_n\nabla f(\bar{\mathbf{x}}_n)), \end{cases}$$

that is the PNPD iterations coincide with the FISTA iterative scheme (4.33) applied to the optimization problem

$$\arg\min_{\mathbf{x}\in\mathbb{R}^d} f(\mathbf{x}) + \lambda\|\mathbf{x}\|_1. \tag{6.27}$$

Of course, removing the extrapolation step, i.e., by setting $\gamma_n = 0$, we fall back on the ISTA algorithm [56].

Consider now the variational model (6.2) and fix $P = A^TA + \nu I$, we obtain

$$P^{-1}\nabla f(\mathbf{x}) = P^{-1}A^T(A\mathbf{x} - \mathbf{b}^\delta) = A^T(AA^T + \nu I)^{-1}(A\mathbf{x} - \mathbf{b}^\delta). \tag{6.28}$$

Therefore, choosing $\alpha_n = 1$ and $\gamma_n = 0$ in (6.12), the exact version of PNPD without extrapolation becomes

$$\mathbf{x}_{n+1} = \mathcal{T}\left(\lambda, \mathbf{x}_n - A^T(AA^T + \nu I)^{-1}(A\mathbf{x}_n - \mathbf{b}^\delta)\right),$$

which is the *Iterated Tikhonov Thresholding Algorithm* (ITTA) proposed in [84] for solving the optimization problem

$$\arg\min_{\mathbf{x}\in\mathbb{R}^d}\frac{1}{2}\|A\mathbf{x} - \mathbf{b}^\delta\|^2_{(AA^T+\nu I)^{-1}} + \lambda\|\mathbf{x}\|_1 = \arg\min_{\mathbf{x}\in\mathbb{R}^d}\frac{1}{2}\|A\mathbf{x} - \mathbf{b}^\delta\|^2_{S^{-1}} + \lambda\|\mathbf{x}\|_1,$$

with $S = AA^T + \nu I$. Note the similarity with our variational problem (6.9).

6.1.5 PNPD with a non-stationary preconditioner

The preconditioner $P = A^T A + \nu I$ used in [2, 84], requires the estimation of the parameter ν that affects the convergence speed and stability of the method. Therefore, as done for non–stationary iterated Tikhonov, ν could be chosen as a non-stationary sequence as in [2, 84, 39, 59]. In a more general framework, P_n could change at each iteration and the PNPD method (6.12) becomes

$$\begin{cases} \bar{\mathbf{x}}_n = \mathbf{x}_n + \gamma_n(\mathbf{x}_n - \mathbf{x}_{n-1}), \\ \mathbf{x}_{n+1} \approx \operatorname{prox}_{\alpha_n h \circ W}(\bar{\mathbf{x}}_n - \alpha_n P_n^{-1} \nabla f(\bar{\mathbf{x}}_n)). \end{cases} \tag{6.29}$$

In what follows, we propose a class of preconditioners P_n of the form

$$P_n = (1 - \nu_n) A^T A + \nu_n I, \tag{6.30}$$

with $\{\nu_n\}_{n\in\mathbb{N}}$ such that $0 < \nu_n \leq 1$ for all $n \in \mathbb{N}$. We propose three possible choices for the sequence $\{\nu_n\}_{n\in\mathbb{N}}$ that are

$$\nu_n = \frac{0.85^n}{2} + \nu_\infty \quad \text{with} \quad \nu_\infty \in \left(0, \frac{1}{2}\right); \tag{6.31}$$

$$\nu_n = \left(1 - \frac{1}{\sqrt{n+1}}\right)(1 - \nu_0) + \nu_0 \quad \text{with} \quad \nu_0 \in (0, 1); \tag{6.32}$$

$$\nu_n = \min\{c^{n-n_{\text{bt}}}, 1\} \quad \text{with} \quad n_{\text{bt}} \in \mathbb{N}, \quad c = \nu_0^{-\frac{1}{n_{\text{bt}}}}, \quad \text{and} \quad \nu_0 \in (0, 1). \tag{6.33}$$

The decreasing sequence (6.31) is largely used in the literature [84, 39, 2] and is inspired by the seminal paper on the convergence of the non-stationary iterated Tikhonov method [77]. On the other hand, nondecreasing sequences, if properly chosen, can provide fast and stable convergence as proved in [59]. This is the case of the sequence (6.32) proposed in [2] for NPDIT. The sequence (6.33) is also an increasing sequence but with a different growth rate than the sequence (6.32). Indeed, the latter sequence starts from ν_0 and increases exponentially until it reaches 1 for $n \geq n_{\text{bt}}$.

Note that, for large enough n, ν_n chosen as in equation (6.31) approaches ν_∞, while $\nu_n \to 1$ when chosen according to (6.32) or (6.33). Therefore, the convergence of the non–stationary PNPD iteration (6.29) may follow from Theorem 6.1.2 for the ν_n sequence in (6.31), and from Theorem 4.2.43 for the ν_n sequences in (6.32) and (6.33). This should be carefully proven in the general case, but it clearly holds if one assumes that the sequence ν_n reaches a fixed value after a certain number of iterations. Indeed, for the sequence in (6.33), we have $P_n = I$ for all $n \geq n_{\text{bt}}$, making the method equivalent to NPD for $n \geq n_{\text{bt}}$.

When we choose a non-stationarity preconditioner P_n, each iteration of the non-stationary

PNPD method (6.29) can be seen as a single step to solve the optimization problem

$$\arg\min_{\mathbf{x}\in\mathbb{R}^d} \frac{1}{2}\|A\mathbf{x}-\mathbf{b}^\delta\|^2_{S_n^{-1}} + \lambda\ h(W\mathbf{x}). \tag{6.34}$$

Since the data fidelity term is a function of S_n^{-1} while the regularization term does not depend on n, there is an implicit dependence on n of the regularization parameter λ. This can be noted, normalizing S_n^{-1} in (6.34), i.e., replacing it with $\frac{S_n^{-1}}{\|S_n^{-1}\|}$. With this change, the problem (6.34) becomes

$$\arg\min_{\mathbf{x}\in\mathbb{R}^d} \frac{1}{2}\|A\mathbf{x}-\mathbf{b}^\delta\|^2_{\frac{S_n^{-1}}{\|S_n^{-1}\|}} + \lambda\ h(W\mathbf{x}) = \arg\min_{\mathbf{x}\in\mathbb{R}^d} \frac{1}{2}\|A\mathbf{x}-\mathbf{b}^\delta\|^2_{S_n^{-1}} + \lambda_n\ h(W\mathbf{x}),$$

with $\lambda_n = \lambda\|S_n^{-1}\|$. Again, for n large enough, the convergence of S_n follows the analysis above for P_n, and in particular, S_n converges to the identity matrix for the ν_n sequences (6.32) and (6.33).

6.2 Numerical experiments

This section presents numerical results for the PNPD method (Algorithm 8). Specifically, we compare the results obtained with our proposed algorithm against those achieved with NPD (Algorithm 6) and NPDIT (Algorithm 7).

All the algorithms are implemented in Python 3.12.3 with NumPy 1.26.4 and all the code used to generate the plots in this section is available in a public Git repository[1]. The experiments below were run on a PC with Kubuntu 24.04, equipped with a 4.75 GHz AMD Ryzen 7 7735HS processor and 16 GB of RAM.

To assess the performance of the methods, we use the RRE and the SSIM, which is computed using the `skimage.metrics.structural_similarity` function from the *scikit-image* library. In the following, besides analyzing these metrics as functions of the number of iterations, we often present them based on the elapsed time. The time is measured using the Python function `time.perf_counter`. Initialization time is not included in the measurements, as it is comparable across all considered algorithms and negligible relative to the duration of a single NPD iteration.

Unless stated otherwise, we use the following parameter choices to ensure the convergence of the methods:

- For NPD, PNPD, and NPDIT, we set $\alpha_n = \alpha = 1$, and $\mathbf{x}_0 = \mathbf{b}^\delta$.
- For NPD and PNPD, we set $\beta_n = \beta = \frac{0.99}{8}$.
- For NPDIT, we set $\beta_n = \beta = \frac{0.99\ \nu}{8}$.

When comparing the two preconditioning strategies, NPDIT and PNPD, we use a stationary

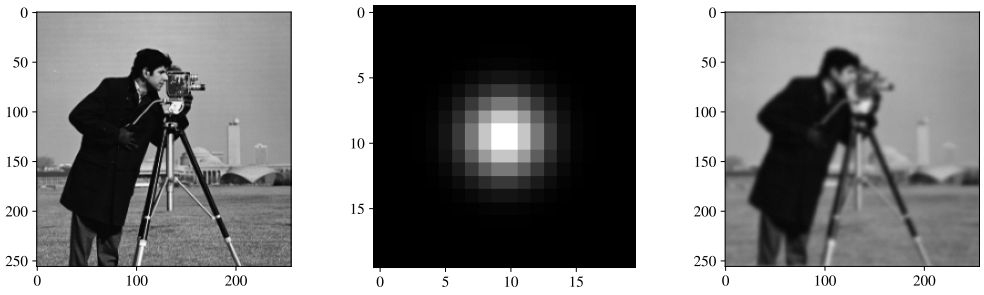
[1] https://github.com/Giuseppe499/PNPD

version of the preconditioner, $P = A^T A + \nu I$, where $\nu > 0$ will be specified for each example. To provide a comprehensive analysis of the performance of our left preconditioning strategy, we also consider the non-stationary preconditioner $P_n = (1 - \nu_n)A^T A + \nu_n I$, where the sequence $\{\nu_n\}_{n\in\mathbb{N}}$ is chosen as in equations (6.31)–(6.33).

6.2.1 Example 1

In this example, we considered a grayscale image of a cameraman with dimensions 256×256. The blurred image $\mathbf{b}$ was obtained using a Gaussian PSF with a $\sigma = 2$ pixels standard deviation. To generate the final observed image $\mathbf{b}^\delta$, we added white Gaussian noise $\boldsymbol{\eta}_\delta$ generated using the NumPy function `numpy.random.normal` and it is scaled such that $\delta = \|\boldsymbol{\eta}_\delta\| = 0.01\|\mathbf{b}\|$.

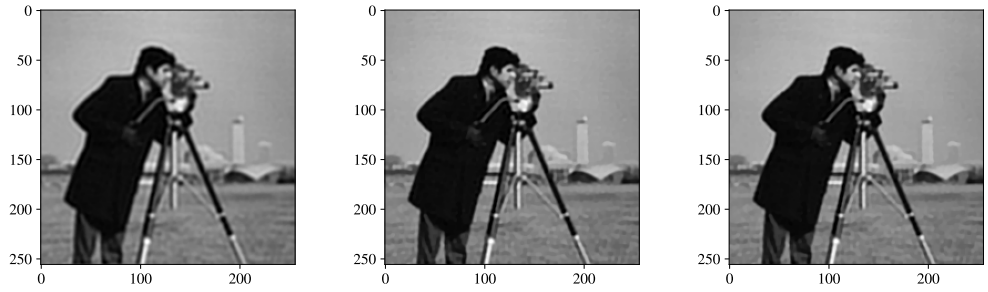
Figure 6.1 displays the ground truth image $\mathbf{x}_{\text{gt}}$, the PSF (shifted to the center and cropped for better visualization), and the observed image $\mathbf{b}^\delta$. In Figure 6.2 we show the reconstructions obtained with NPD, NPDIT, and PNPD after 10 iterations.



(a) (b) (c)

Figure 6.1: Example 1: (a) Ground truth image of a cameraman. (b) PSF used to blur the ground truth (center crop of size 20×20). (c) Observed image $\mathbf{b}^\delta$.

The comparison between the three different algorithms NPD, NPDIT, and PNPD is presented in Figure 6.3. The performances of each method were measured through the RRE and SSIM functionals. Iteration-wise, we observe that PNPD and NPDIT exhibit similar behaviors, both converging faster than NPD. This is due to the presence of the preconditioner P, which effectively enhances the speed of convergence of the algorithms. In terms of execution time, the PNPD strategy outperforms both the other methods. The gap between our proposal and NPDIT is due to the fact that as $k_{\max}$ increases, the NPDIT algorithm must compute more FFTs at each iteration. Indeed, the NPDIT method looks for approximate evaluations of $\text{prox}^P_{\alpha h \circ W}$ while PNPD approximates $\text{prox}_{\alpha h \circ W}$ in the same manner as NPD. Therefore, NPDIT has to perform an extra multiplication by P^{-1} for each extra nested iteration compared to PNPD. This behavior is underlined in Table 6.1, reporting the average time spent for one step of PNPD and NPDIT for different values of $k_{\max}$. We can see that Δ, which is the difference between the average time spent for the two methods,



(a) NPD (b) NPDIT (c) PNPD

Figure 6.2: Example 1: Comparison of the reconstructions obtained with NPD, PNPD, and NPDIT after 10 iterations. The preconditioner parameter is $\nu = 10^{-1}$. The number of nested loop iterations is $k_{\max} = 3$. The regularization parameter is $\lambda = 2 \cdot 10^{-4}$ for NPD and NPDIT, while $\lambda = 2 \cdot 10^{-3}$ for PNPD.

increases as $k_{\max}$ increases and, more interestingly, the same happens for the ratio of the execution time of one step of NPDIT and one of PNPD. The execution speed gap between the two methods increases to the point in which one iteration of PNPD with $k_{\max} = 8$ is fast almost as NPDIT with $k_{\max} = 1$.

k_{max}	PNPD	NPDIT	Δ	NPDIT/PNPD
1	0.0090	0.0109	0.0019	1.211
2	0.0096	0.0119	0.0022	1.275
4	0.0114	0.0153	0.0040	1.342
8	0.0149	0.0221	0.0072	1.483
16	0.0216	0.0350	0.0134	1.620
32	0.0374	0.0641	0.0267	1.714
64	0.0664	0.1207	0.0543	1.818

Table 6.1: Average time spent for one step of PNPD and NPDIT for different values of $k_{\max}$. The difference between the execution time of the two methods is shown in the Δ column, while the last column reports the ratio of the execution time of a step of NPDIT and one of PNPD.

Stability

While the previous analysis was dedicated to comparing PNPD with NPD, and NPDIT, we are now considering the stability properties of our proposal. In Figure 6.4 we report the RREs and SSIMs obtained for PNPD with different values of the preconditioner parameter ν in equation (6.17). We can observe that as ν decreases the method becomes faster but also unstable. This behavior is justified from a theoretical viewpoint. Indeed, as ν approaches zero, the preconditioner $P = A^T A + \nu I$ tends to the positive semidefinite linear operator $A^T A$. Therefore, computing P^{-1} we are trying to invert an almost singular operator and this introduces instability along the iterations. On the other hand, the preconditioner P is converging to the true Hessian matrix of the differentiable term in the model problem (6.2), resulting in faster convergence towards the minimum point. A trade-off between these two properties is clearly needed.

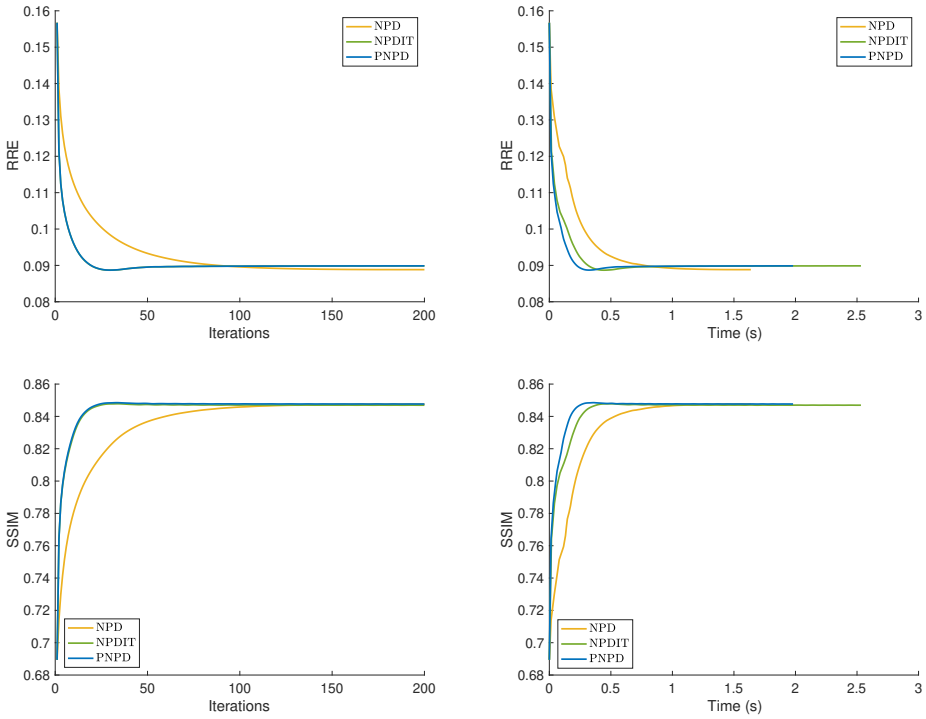


Figure 6.3: Example 1: Comparison of the RREs and SSIMs between PNPD, NPD, and NPDIT. The preconditioner parameter is $\nu = 10^{-1}$. The number of nested loop iterations is $k_{\max} = 1$ for NPD and $k_{\max} = 3$ for NPDIT and PNPD. The regularization parameter is $\lambda = 2 \cdot 10^{-4}$ for NPD and NPDIT, while $\lambda = 2 \cdot 10^{-3}$ for PNPD.

One possible strategy to overcome instability is to increase the number $k_{\max}$ of nested iterations. In Figure 6.5 we show the results obtained with PNPD with different values of ν, but in this case, for each different choice, we increased $k_{\max}$ enough to prevent instability. Even though a higher $k_{\max}$ reflects in a higher iteration execution time, a decrease in ν still results in a faster convergence.

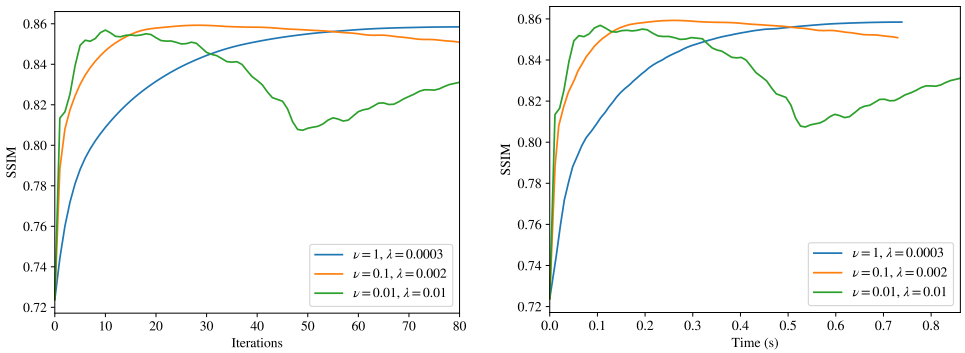


Figure 6.4: Example 1: Comparison of the SSIMs of PNPD with $k_{\max} = 1$ for different values of ν and λ.

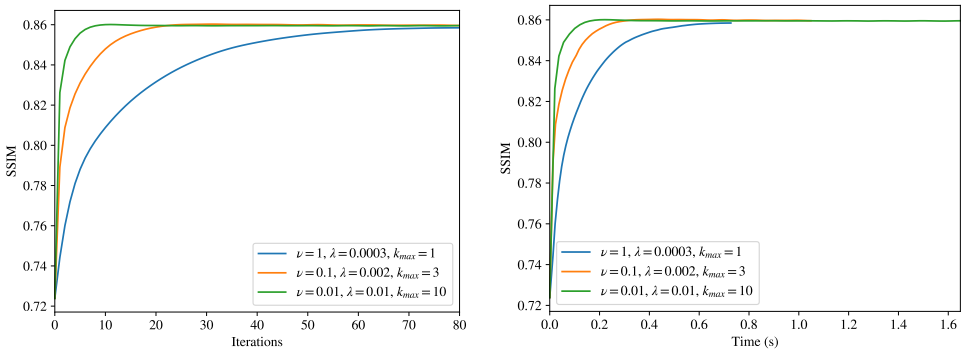


Figure 6.5: Example 1: Comparison of the SSIMs of PNPD for different values of ν, λ, and $k_{\max}$ set high enough to fix instability.

One other possible strategy to overcome instability along the iterations is to get rid of the extrapolation step obtaining the PNPD_NE method defined as

$$\mathbf{x}_{n+1} \approx \text{prox}_{\alpha_n h \circ W}(\mathbf{x}_n - \alpha_n P^{-1} \nabla f(\mathbf{x}_n)). \tag{6.35}$$

This method is less prone to instability than PNPD as depicted in Figure 6.6. Indeed, differently from Figure 6.4, even though $k_{\max} = 1$, PNPD_NE does not show worrying instability even for $\nu = 10^{-2}$. However, this improvement in stability comes at the cost of convergence speed.

Therefore, in the case in which PNPD shows instability for a certain value of ν, we have two different choices. We can increase $k_{\max}$ (which also comes at the cost of CPU time) or we can use PNPD_NE instead of PNPD. As an example, analyzing the case in which

$\nu = 0.01$, we can see from Figure 6.5 that PNPD with $k_{\max} = 10$ reaches an SSIM of 0.853 after 0.079 seconds or 4 iterations, while PNPD_NE (see Figure 6.6) after 0.093 seconds or 8 iterations.

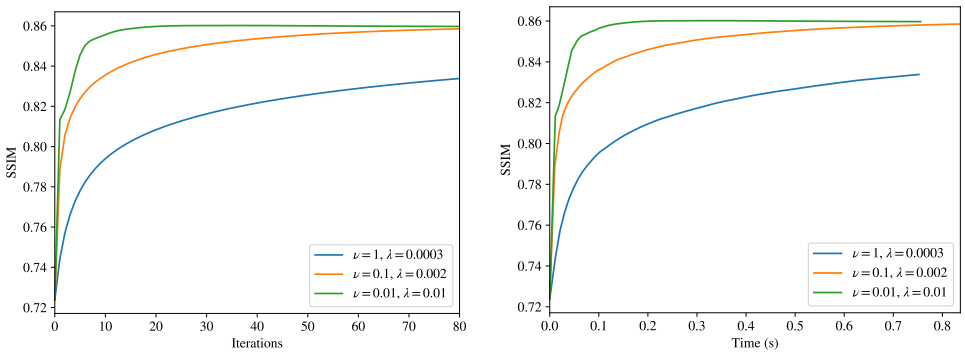


Figure 6.6: Example 1: Comparison of the SSIMs of PNPD_NE with $k_{\max} = 1$ for different values of ν and λ.

Non–stationary Preconditioning

Lastly, we compare the results obtained for PNPD with the preconditioner sequences $\{P_n\}$ of the form in equation (6.30) with the different sequences of $\{\nu_n\}_{n\in\mathbb{N}}$ discussed in Subsection 6.1.5. For the choice (6.31) we set $\nu_\infty = 10^{-2}$, for (6.32) and (6.33) we set $\nu_0 = 10^{-2}$, and for (6.33) we set $n_{\text{bt}} = 20$. The compared stationary PNPD, cf. (6.31)–(6.33), is obtained with $\nu_n = \nu = 10^{-1}$.

In Figure 6.7, we observe that the decreasing sequence (6.31) is slower at the beginning and then accelerates, overtaking its stationary version. On the other hand, the increasing sequence (6.32), has a fast convergence in the first iterations, but it slows down, and it falls behind the stationary PNPD. The bootstrap sequence (6.33) also exhibits an increasing behavior, resulting in fast convergence in the initial steps where the preconditioner acceleration is most noticeable. Furthermore, differently from (6.32), we can see that (6.33) can outperform the stationary PNPD.

In Figure 6.8, we tested the behavior of the bootstrap sequence (6.33) when the parameters ν_0 and n_{bt} are changed and are set unfavorably. We can see that by choosing ν_0 too small ($\nu_0 = 10^{-3}$) the method becomes unstable in the first iterations, but, since it eventually becomes equivalent to NPD after a finite number of iterations, the method still converges to a solution of equation (6.2). If we instead choose a reasonable ν_0, changing n_{bt} can affect convergence speed but has a large margin of error. In particular, we tested n_{bt} for the values, 5, 20 and 50. We can see that the convergence speed increases as n_{bt} increases. This is particularly noticeable when switching from 5 to 20, but is less impactful when changing from 20 to 50. Taking n_{bt} too high results in the switch to NPD happening later and, therefore, if ν_0 is chosen in a suboptimal way, the method could diverge at the beginning and will start to converge only when approaching n_{bt}.

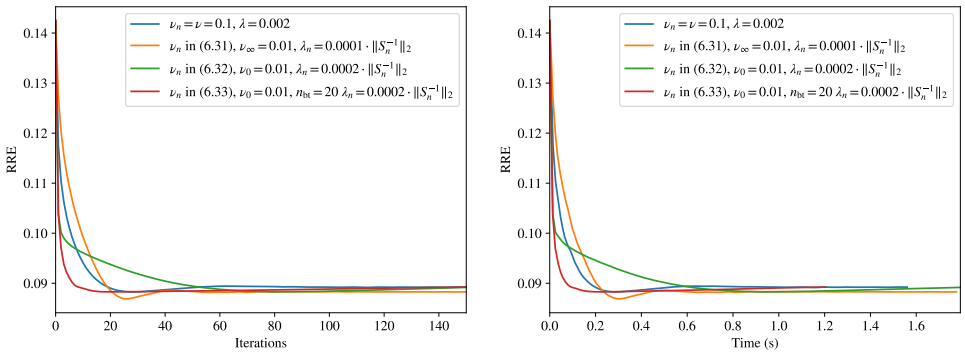


Figure 6.7: Example 1: Comparison of the RREs for the non-stationary version of PNPD and different sequences of ν_n. For this test, we set $k_{\max} = 3$.

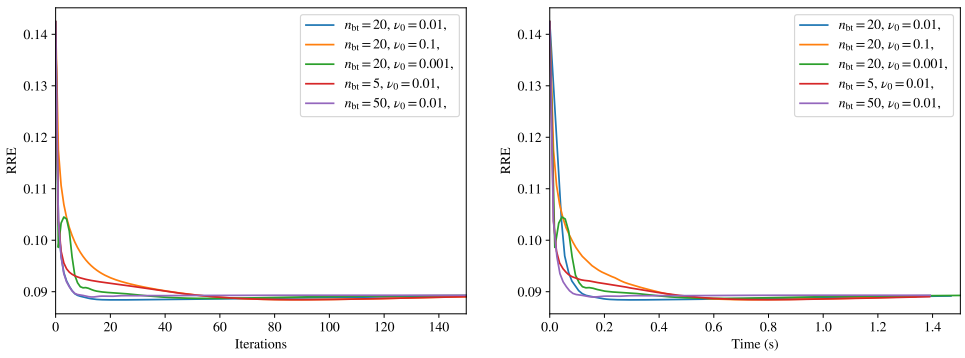


Figure 6.8: Example 1: Comparison of the RREs obtained with PNPD for the non-stationary bootstrap sequence of ν_n in (6.33). For this test, we set $\lambda_n = 2 \cdot 10^{-4} \cdot \|S_n^{-1}\|$ and $k_{\max} = 3$.

6.2.2 Example 2

In this example, we consider the same framework as in the previous one but increasing the noise level in the final observed image $\mathbf{b}^\delta$ from 1% to 2%.

To further analyze the performance of these algorithms, Figure 6.9 presents a comparison of the quality of the reconstructions at each step by computing the RRE and the SSIM. In both scenarios, the PNPD method outperforms the other strategies in terms of both the number of iterations required and CPU time.

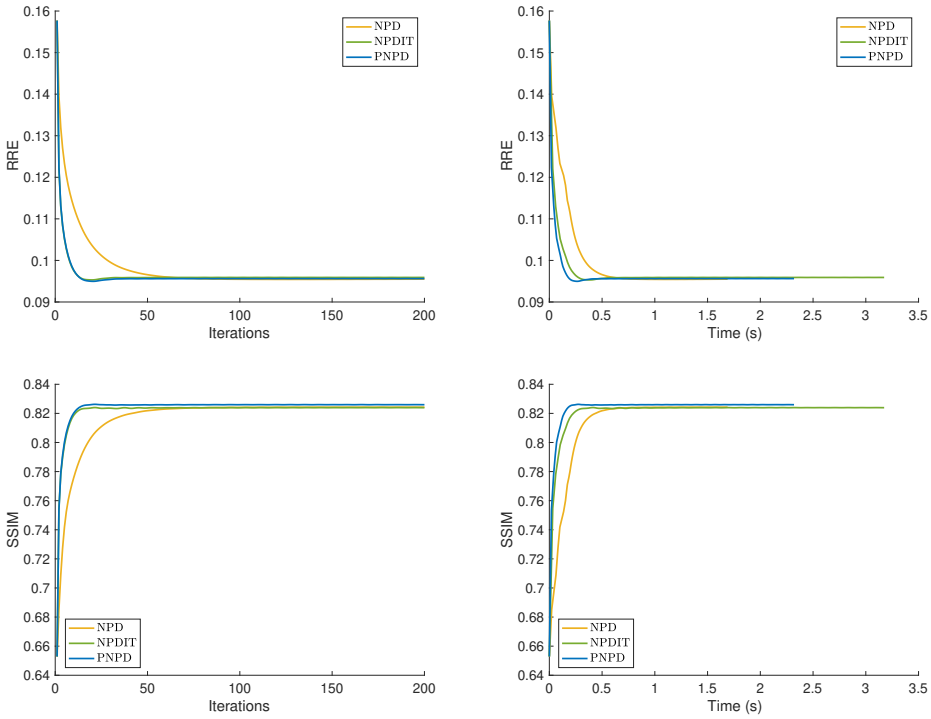


Figure 6.9: Example 2: Comparison of the RREs and SSIMs between PNPD, NPD, and NPDIT. The preconditioner parameter is $\nu = 10^{-1}$. The number of nested loop iterations is $k_{\max} = 2$ for NPD and $k_{\max} = 5$ for NPDIT and PNPD. The regularization parameter is $\lambda = 7 \cdot 10^{-4}$ for NPD and NPDIT, and is $\lambda = 6 \cdot 10^{-3}$ for PNPD.

A further improvement of PNPD can be obtained by employing the strategies discussed in the previous Example 1, as shown in Figure 6.10. Here, in addition to the optimal stationary case presented in Figure 6.9, we consider a smaller ν, increasing $k_{\max}$ or using PNPD_NE which is described in (6.35). In the same figure, we also consider the non-stationary PNPD with the bootstrap strategy, as defined in (6.33). As we already expected, the case without extrapolation is the slowest in terms of iterations. However, when there is uncertainty about the optimal parameters settings, the PNPD_NE method appears to be a reasonable choice. This is because the method remains stable without the need to increase the number of nested iterations, and it allows for smaller values of ν in the preconditioner. Furthermore, in terms of CPU time, all the proposed strategies shows similar performances.

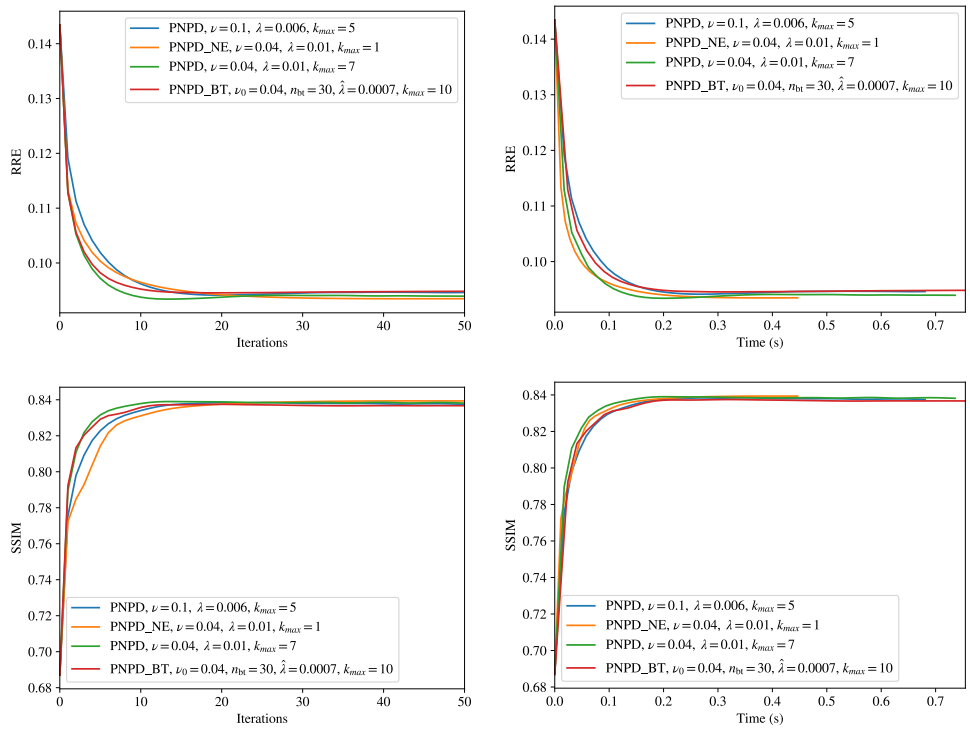
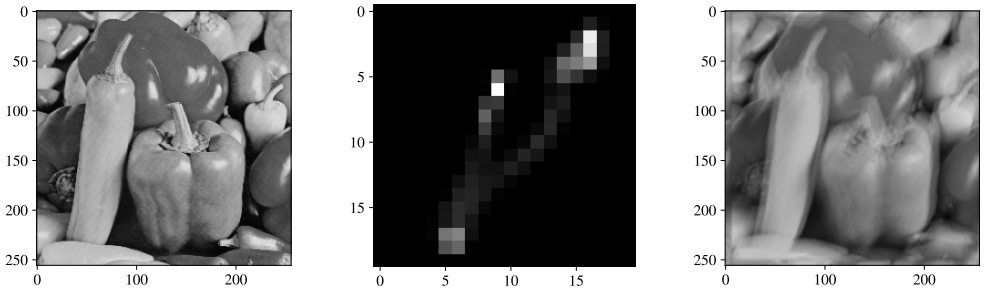


Figure 6.10: Example 2: Comparison of the RREs and SSIMs between the proposed variants of PNPD. In particular, the results are obtained with PNPD, PNPD_NE, and the bootstrap version of PNPD (PNPD_BT), which uses the non-stationary sequence ν_n in (6.33) and $\lambda_n = \hat{\lambda} \cdot \|S_n^{-1}\|$.

6.2.3 Example 3

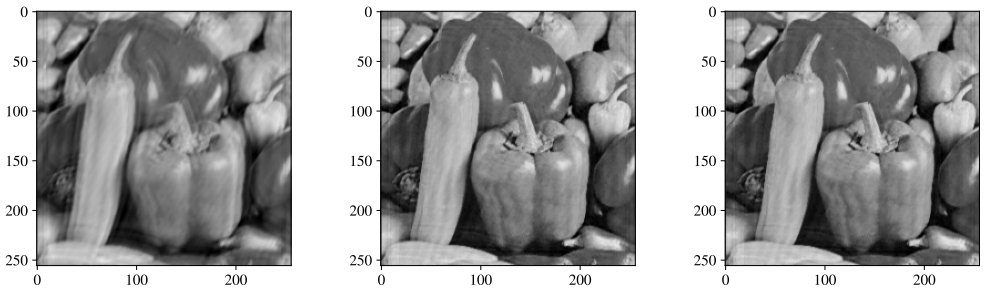
As a final example, we consider a different grayscale image of some peppers with dimensions 256×256, a motion blur PSF, and white Gaussian noise with an intensity level of 0.5%.

Figure 6.11 displays the ground truth $\mathbf{x}_{\text{gt}}$, the PSF, and the blurred image $\mathbf{b}^{\delta}$. Figure 6.12 shows the reconstructions obtained with NPD, NPDIT, and PNPD after 5 iterations. We can observe that, even though the noise level is lower than in previous examples, the PNPD and NPDIT methods outperform the standard NPD strategy after just 5 iterations. This



(a) (b) (c)

Figure 6.11: Example 3: (a) Ground truth image of some peppers. (b) PSF used to blur the ground truth (center crop of size 20×20). (c) Observed image $\mathbf{b}^{\delta}$.



(a) NPD (b) NPDIT (c) PNPD

Figure 6.12: Example 3: Comparison of the reconstructions obtained with NPD, PNPD, and NPDIT after 5 iterations. The preconditioner parameter is $\nu = 10^{-2}$. The number of nested loop iterations is $k_{\max} = 2$. The regularization parameter is $\lambda = 10^{-4}$ for NPD and NPDIT, and is $\lambda = 6 \cdot 10^{-3}$ for PNPD.

is particularly noticeable in Figure 6.13, where we compared the RREs and SSIMs obtained with PNPD, NPD, and NPDIT. Iteration-wise, PNPD and NPDIT exhibit similar behaviors, both converging faster than NPD. However, in terms of CPU time, PNPD shows a slight improvement over NPDIT. Similarly to Figure 6.10 in Example 2, Figure 6.14 compares different parameter settings for the PNPD method. Again, we observe that, when the optimal parameter choice is known, the stationary case remains the best among all possibilities. In

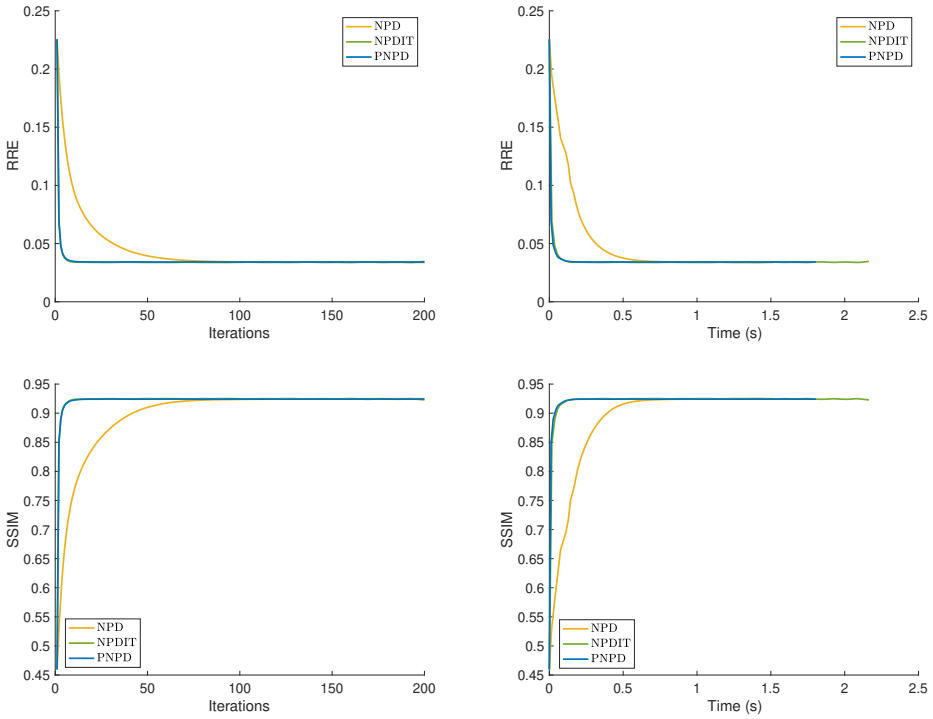


Figure 6.13: Example 3: Comparison of the RREs and SSIMs between PNPD, NPD, and NPDIT. The preconditioner parameter is $\nu = 10^{-2}$. The number of nested loop iterations is $k_{\max} = 1$ for NPD and $k_{\max} = 2$ for NPDIT and PNPD. The regularization parameter is $\lambda = 10^{-4}$ for NPD and NPDIT, and is $\lambda = 6 \cdot 10^{-3}$ for PNPD.

the stationary case, when choosing ν too small, PNPD becomes unable to achieve the same performance metrics as the optimal case. For example, PNPD with $\nu = 10^{-2}$ achieves an SSIM of 0.935 after 50 iterations, while PNPD and PNPD_NE with $\nu = 10^{-3}$ both achieve a lower SSIM of 0.92. Instead, the non-stationary PNPD with the bootstrap sequence (6.33) (PNPD_BT), achieves the same SSIM as the optimal stationary case. PNPD_BT, although slightly slower in the initial iterations compared to other considered cases, performs nearly as well as the optimal stationary case.

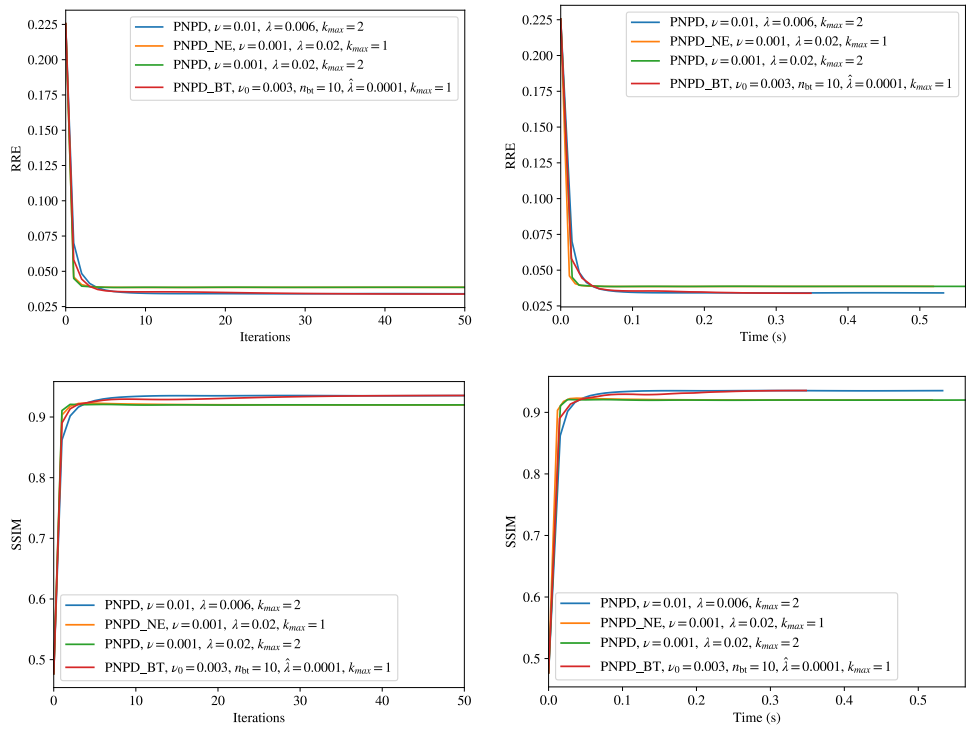


Figure 6.14: Example 3: Comparison of the RREs and SSIMs between the proposed variants of PNPD. In particular, we show results obtained with PNPD, PNPD_NE, and the bootstrap version of PNPD (PNPD_BT) which uses the non-stationary sequence ν_n in (6.33) and $\lambda_n = \hat{\lambda} \cdot \|S_n^{-1}\|$.

6.3 Conclusions

Inspired by the NPDIT method recently proposed in [2], we investigated preconditioning strategies for proximal gradient methods applied to image deblurring problems. We proved that, for this particular application, the NPDIT method can be interpreted as the right preconditioning. Therefore, we proposed a left preconditioning method to reduce the number of evaluations of the preconditioner, and thus the CPU time, at each iteration. Moreover, we explored some strategies to improve the stability of the proposed PNPD method preserving the fast convergence in the first iterations. Numerical results on image deblurring problems with white Gaussian noise confirm the advantages of PNPD over NPDIT.

Interesting future investigations concern other choices of the preconditioning matrix, par-

ticularly when our proposal might be computationally expensive to invert, like in computed tomography applications. Moreover, the role of preconditioning combined with other extrapolation strategies, see [28], deserves further investigation.

Conclusions and Future Work

In this concluding chapter, we provide a brief overview of the main results presented in this thesis and offer some insights into potential future directions we would like to pursue.

In the first part of this thesis, specifically in Chapters 2 and 3, we analyze in detail the behavior of the graph Laplacian operator and its fractional power. In particular, in Chapter 2, we examine the dependence of the graph Laplacian on an initial reconstruction computed using a reconstruction map Ψ. We demonstrate that, under very weak hypotheses on Ψ, the `graphLa+`Ψ method exhibits both convergence and stability properties. In Chapter 3, we consider the fractional power of the graph Laplacian operator within an $\ell^2 - \ell^q$ variational framework with $q \leq 1$. We apply it to both image deblurring and CT reconstruction problems showing that the fractional exponent can significantly improve the quality of the restored images.

Looking ahead, we would like to extend all the theoretical results proven in Chapter 2, which considered the simpler case with $q = 1$, to the more general case in Chapter 3 where $q \leq 1$. This extension would introduce additional complexity to the theoretical analysis. Firstly, since we maintain the dependence on the general reconstruction map Ψ. Secondly, since we can not fully describe the action of the fractional graph Laplacian, as our understanding of the behavior of the fractional exponent is limited to a relatively small Krylov subspace.

Another direction we would like to explore is the application of a bilevel optimization strategy to estimate the hyperparameters that define the graph Laplacian or, even better, its fractional counterpart. In Chapter 1, we showed that the edge–weight function used to define the graph Laplacian operator depends on two hyperparameters: the sparsity coefficient R and the σ coefficient. While we provided reasonable estimates that work well in practice, alternative choices could yield even better results.

Finally, with regard to the standard graph Laplacian, we aim to investigate the multifeature graph Laplacian, where the edge-weight function depends not only on the intensity of individual pixels but also on additional image features. For example, a straightforward extension could involve comparing the average neighborhood distribution of pixel intensity values rather than focusing solely on differences between individual pixels. Some work in this direction has already been considered in [40] for the image segmentation problem.

The second part of this thesis focused on optimization strategies for regularized convex problems. In Chapter 5, we described a variable metric approach combined with extrapolation that uses a fixed number of nested iterations to compute an approximation of the proximal gradient point. We demonstrated that, when applied to the image deblurring problem, an iterated Tikhonov–based approach significantly enhances the convergence speed of the method, outperforming classical approaches. In Chapter 6, we showed that, in the image deblurring framework, the variable metric strategy can be reformulated as a right preconditioning approach. Consequently we analyzed the left preconditioning strategy, showing that further improvements can be achieved by reducing computational costs while maintaining reconstruction quality. On the other hand, the main drawback of the left preconditioning approach is the modification of the data fidelity norm, which is replaced by the norm induced by the chosen preconditioner. However, in the final part of the chapter, we demonstrated that a non–stationary preconditioning can mitigate this issue by allowing the sequence of preconditioners to converge to the identity operator.

The first thing we would like to pursue, which we were unfortunately unable to include in this thesis due to time constraints, is to establish a connection between the two parts of this thesis. Specifically, we would like to replace the MM–GKS strategy used in Chapters 2 and 3 to solve the $\ell^2 - \ell^q$ variational model with either the NPDIT method from Chapter 5 or the PNPD algorithm from Chapter 6. This can be achieved in two main steps: the first step involves applying the NPDIT and PNPD methods to the CT reconstruction problem. The second step involves replacing the TV regularization with the graph Laplacian or its fractional power.

While the second step is relatively straightforward to implement, as we already have the necessary estimates for the graph Laplacian norm, the first step requires considerably more effort. CT problems have to be analyzed differently than image deblurring problems. Firstly, the preconditioner P needs to be approximated in some way. This is an issue that would also arise in image deblurring problems when considering boundary conditions other than the periodic ones. However, in [57], a strategy was proposed to compute the PSF associated with P for different BCs. Unfortunately, such an approach cannot be directly applied to CT problems. Moreover, the step length depends on the spectral norm of the discretized Radon transform operator, which must be approximated by some strategy.

Another direction we would like to explore is the application of an unrolling neural network to the RGB case in image deblurring problems. This neural network would automatically estimate all the relevant parameters of the PNPD method.

Finally, in [97], a multilevel approach applied to the FISTA method was recently proposed. The idea of combining proximal-gradient methods with multigrid approaches led to excellent reconstructions, supported by strong theoretical analysis. We would like to extend this to the more general case of variable metric approaches, where inexactness in the computation of the proximal gradient step is allowed.

Bibliography

[1] S. Aleotti, D. Bianchi, D. Evangelista, M. Donatelli, W. Li, and E. L. Piccolomini. Official GitHub repository for the `graphla+`ψ codes. https://github.com/devangelista2/GraphLaPlus, 2023. Online; accessed 26-July-2024.

[2] S. Aleotti, S. Bonettini, M. Donatelli, M. Prato, and S. Rebegoldi. A nested primal–dual iterated Tikhonov method for regularized convex optimization, manuscript. *Computational Optimization and Applications*, 2024.

[3] S. Aleotti, A. Buccini, and M. Donatelli. Fractional graph Laplacian for image reconstruction. *Applied Numerical Mathematics*, 200:43–57, 2024.

[4] A. Ali and R. J. Tibshirani. The generalized lasso problem and uniqueness. *Electronic Journal of Statistics*, 13:2307–2347, 2019.

[5] H. Antil and S. Bartels. Spectral approximation of fractional PDEs in image processing and phase field modeling. *Computational Methods in Applied Mathematics*, 17(4):661–678, 2017.

[6] V. Antun, F. Renna, C. Poon, B. Adcock, and A. C. Hansen. On instabilities of deep learning in image reconstruction and the potential costs of AI. *Proceedings of the National Academy of Sciences*, 117:30088–30095, 2020.

[7] P. Arias, V. Caselles, and G. Sapiro. A variational framework for non-local image inpainting. In *International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*, pages 345–358. Springer, 2009.

[8] F. Bach, R. Jenatton, J. Mairal, and G. Obozinski. Structured sparsity through convex optimization. *Stat. Sci.*, 27(4):450–468, 2012.

[9] A. B. Bakushinskii. Remarks on choosing a regularization parameter using the quasi-optimality and ratio criterion. *USSR Computational Mathematics and Mathematical Physics*, 24(4):181–182, 1984.

[10] H. H. Bauschke and P. L. Combettes. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. CMS Books in Mathematics. Springer, Cham, 2 edition, 2017.

[11] A. Beck and M. Teboulle. Fast gradient-based algorithms for constrained total varia-

tion image denoising and deblurring problems. *IEEE Transactions on Image Processing*, 18(11):2419–2434, 2009.

[12] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Imaging Sci.*, 2:183–202, 2009.

[13] A. Beck and M. Teboulle. *Gradient-based algorithms with applications to signal-recovery problems*, pages 42–88. Cambridge University Press, 2009.

[14] F. Benedetto, C. Estatico, and S. Serra-Capizzano. Superoptimal preconditioned conjugate gradient iteration for image deblurring. *SIAM Journal on Scientific Computing*, 26:1012–1035, 01 2005.

[15] M. Benzi, D. Bertaccini, F. Durastante, and I. Simunec. Non-local network dynamics via fractional graph laplacians. *Journal of Complex Networks*, 8(3):cnaa017, 2020.

[16] M. Bertero and P. Boccacci. *Introduction to inverse problems in imaging*. Institute of Physics Publishing, Bristol, 1998.

[17] M. Bertero and P. Boccacci. A simple method for the reduction of boundary effects in the richardson-lucy approach to image deconvolution. *http://dx.doi.org/10.1051/0004-6361:20052717*, 437, 07 2005.

[18] D. P. Bertsekas. Nonlinear programming. *Journal of the Operational Research Society*, 48(3):334–334, 1997.

[19] D. Bianchi, A. Buccini, M. Donatelli, and E. Randazzo. Graph laplacian for image deblurring. *Electronic Transaction on Numerical Analysis*, 55:169–186, 2022.

[20] D. Bianchi and M. Donatelli. Graph approximation and generalized Tikhonov regularization for signal deblurring. In *2021 21st International Conference on Computational Science and Its Applications (ICCSA)*, pages 93–100. IEEE, 2021.

[21] D. Bianchi, M. Donatelli, F. Durastante, and M. Mazza. Compatibility, embedding and regularization of non-local random walks on graphs. *Journal of Mathematical Analysis and Applications*, 511(1):126020, 2022.

[22] D. Bianchi, M. Donatelli, D. Evangelista, W. Li, and E. L. Piccolomini. Graph laplacian and neural networks for inverse problems in imaging: Graphlanet. In *International Conference on Scale Space and Variational Methods in Computer Vision*, pages 175–186, 2023.

[23] S. Bonettini, I. Loris, F. Porta, and M. Prato. Variable metric inexact line-search-based methods for nonsmooth optimization. *SIAM Journal on Optimization*, 26(2):891–921, 2016.

[24] S. Bonettini, F. Porta, and V. Ruggiero. A variable metric forward-backward method with extrapolation. *SIAM Journal on Scientific Computing*, 38:A2558–A2584, 2016.

[25] S. Bonettini, F. Porta, V. Ruggiero, and L. Zanni. Variable metric techniques for

forward-backward methods in imaging. *Journal of Computational and Applied Mathematics*, 385:113192, 2021.

[26] S. Bonettini, M. Prato, and S. Rebegoldi. A nested primal–dual FISTA-like scheme for composite convex optimization problems. *Computational Optimization and Applications*, 84:85–123, 2023.

[27] S. Bonettini, S. Rebegoldi, and V. Ruggiero. Inertial variable metric techniques for the inexact forward–backward algorithm. *SIAM Journal on Scientific Computing*, 40(5):A3180–A3210, 2018.

[28] A. Buccini, P. Dell'Acqua, and M. Donatelli. A general framework for ADMM acceleration. *Numerical Algorithms*, 85:829–848, 2020.

[29] A. Buccini and M. Donatelli. Graph Laplacian in $\ell^2 - \ell^q$ regularization for image reconstruction. In *2021 21st International Conference on Computational Science and Its Applications (ICCSA)*, pages 29–38. IEEE, 2021.

[30] A. Buccini and M. Donatelli. Graph Laplacian in $\ell^2 - \ell^q$ regularization for image reconstruction. In *2021 21st International Conference on Computational Science and Its Applications (ICCSA)*, pages 29–38. IEEE, 2021.

[31] A. Buccini, M. Donatelli, and L. Reichel. Iterated Tikhonov regularization with a general penalty term. *Numerical Linear Algebra with Applications*, 24(4):e2089, 2017.

[32] A. Buccini, O. De la Cruz Cabrera, M. Donatelli, A. Martinelli, and L. Reichel. Large-scale regression with non-convex loss and penalty. *Applied Numerical Mathematics*, 157:590–601, 2020.

[33] A. Buccini, O. De la Cruz Cabrera, C. Koukouvinos, M. Mitrouli, and L. Reichel. Variable selection in saturated and supersaturated designs via minimization. *Communications in Statistics - Simulation and Computation*, 52:1–22, 2021.

[34] A. Buccini, M. Pragliola, L. Reichel, and F. Sgallari. A comparison of parameter choice rules for $\ell^p - \ell^q$ minimization. *Annali dell'Università di Ferrara*, 68:441–463, 2022.

[35] A. Buccini and L. Reichel. An ℓ^2-ℓ^q regularization method for large discrete ill-posed problems. *Journal of Scientific Computing*, 78:1526–1549, 2019.

[36] A. Buccini and L. Reichel. Limited memory restarted $\ell^p - \ell^q$ minimization methods using generalized Krylov subspaces. *Advances in Computational Mathematics*, 49:26, 2023.

[37] C. T. H. Baker, L. Fox, D. F. Mayers, and K. Wright. *Numerical solution of Fredholm integral equations of first kind*, volume 7, issue 2. The computer Journal, 1964.

[38] J. F. Cai, R. H. Chan, and Z. Shen. A framelet-based image inpainting algorithm. *Applied and Computational Harmonic Analysis*, 24(2):131–149, 2008. Special Issue on Mathematical Imaging – Part II.

[39] Y. Cai, M. Donatelli, D. Bianchi, and T. Z. Huang. Regularization preconditioners for frame-based image deblurring with reduced boundary artifacts. *SIAM Journal on Scientific Computing*, 38(1):B164–B189, 2016.

[40] L. Calatroni, Y. van Gennip, C.-B. Schönlieb, H. M. Rowland, and A. Flenner. Graph clustering, variational image segmentation methods and Hough transform scale detection for object measurement in images. *Journal of Mathematical Imaging and Vision*, 57(2):269–291, 2017.

[41] D. Calvetti, B. Lewis, and L. Reichel. On the regularizing properties of the gmres method. *Numerische Mathematik*, 91:605–625, 06 2002.

[42] E. Candes and B. Recht. Simple bounds for recovering low-complexity models. *Mathematical Programming*, 141(1–2):577–589, 2013.

[43] A. Chambolle and C. Dossal. On the convergence of the iterates of the "fast iterative shrinkage/thresholding algorithm". *Journal of Optimization Theory and Applications*, 166(3):968–982, 2015.

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

[45] R. H. Chan, T. F. Chan, L. Shen, and Z. Shen. Wavelet algorithms for high-resolution image reconstruction. *SIAM Journal on Scientific Computing*, 24(4):1408–1432, 2003.

[46] R. H. Chan and H.-X. Liang. Half-quadratic algorithm for ℓ_p-ℓ_q problems with applications to TV-ℓ_1 image restoration and compressive sensing. In *Efficient Algorithms for Global Optimization Methods in Computer Vision*, pages 78–103. Springer, New York, 2014.

[47] T. F. Chan and J. (Jackie) Shen. *Image Processing and Analysis*. Society for Industrial and Applied Mathematics, 2005.

[48] J. Chen and I. Loris. On starting and stopping criteria for nested primal-dual iterations. *Numerical Algorithms*, 82:605–621, 2019.

[49] E. Chouzenoux, J.-C. Pesquet, and A. Repetti. Variable metric forward-backward algorithm for minimizing the sum of a differentiable function and a convex function. *Journal of Optimization Theory and Applications*, 162:107–132, 2014.

[50] P. G. Ciarlet. *Introduction to Numerical Linear Algebra and Optimization*. Cambridge University Press, Cambridge, 1989.

[51] M. J. Colbrook, V. Antun, and A. C. Hansen. The difficulty of computing stable and accurate neural networks: On the barriers of deep learning and smale's 18th problem. *Proceedings of the National Academy of Sciences*, 119:e2107151119, 2022.

[52] P. L. Combettes and J. C. Pesquet. Proximal splitting methods in signal processing. *Fixed-point algorithms for inverse problems in science and engineering*, 2011.

[53] P. L. Combettes and B. C. Vu. Variable metric quasi-Fejér monotonicity. *Nonlinear Analysis*, 78:17–31, 2013.

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

[55] J. W. Daniel, W. B. Gragg, L. Kaufman, and G. W. Stewart. Reorthogonalization and stable algorithms for updating the Gram-Schmidt QR factorization. *Mathematics of Computation*, 30(136):772–795, 1976.

[56] I. Daubechies, M. Defrise, and C. De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on Pure and Applied Mathematics*, 57(11):1413–1457, 2004.

[57] P. Dell'Acqua, M. Donatelli, C. Estatico, and M. Mazza. Structure preserving preconditioners for image deblurring. *Journal of Scientific Computing*, 72, 2017.

[58] D. di Serafino, G. Landi, and M. Viola. Directional TGV-based image restoration under Poisson noise. *Journal of Imaging*, 7(6):99, 2021.

[59] M. Donatelli. On nondecreasing sequences of regularization parameters for nonstationary iterated Tikhonov. *Numerical Algorithms*, 60:651–668, 2012.

[60] M. Donatelli, C. Estatico, A. Martinelli, and S. Serra-Capizzano. Improved image deblurring with anti-reflective boundary conditions and re-blurring. *Inverse Problems*, 22:2035, 10 2006.

[61] M. Donatelli and L. Reichel. Square smoothing regularization matrices with accurate boundary conditions. *Journal of Computational and Applied Mathematics*, 272:334–349, 2014.

[62] H. W. Engl, M. Hanke, and G. Neubauer. *Regularization of Inverse Problems*. Mathematics and Its Applications. Springer, Dordrecht, Netherlands, 1996 edition, July 1996.

[63] C. Estatico, S. Gratton, F. Lenti, and D. Titley-Peloquin. A conjugate gradient like method for p-norm minimization in functional spaces. *Numerische Mathematik*, 137(4):895–922, 2017.

[64] D. Evangelista, E. Morotti, and E. L. Piccolomini. COULE dataset. https://www.kaggle.com/datasets/loiboresearchgroup/coule-dataset, 2023. Accessed on 01/12/2023.

[65] D. Evangelista, E. Morotti, and E. L. Piccolomini. RISING: A new framework for model-based few-view CT image reconstruction with deep learning. *Computerized Medical Imaging and Graphics*, 103:102156, 2023.

[66] D. Evangelista, E. Morotti, E. L. Piccolomini, and J. Nagy. Ambiguity in solving imaging inverse problems with deep-learning-based operators. *Journal of Imaging*, 9(7):133, 2023.

[67] Y. W. Fan and J. G. Nagy. Synthetic boundary conditions for image deblurring. *Linear Algebra and its Applications*, 434(11):2244–2268, 2011. Special Issue: Devoted to the 2nd NASC 08 Conference in Nanjing (NSC).

[68] T. G. Feeman. *The Mathematics of Medical Imaging: A Beginner's Guide*. Springer Publishing Company, Incorporated, 2014.

[69] S. Gazzola, P. C. Hansen, and J. G. Nagy. IR Tools: a MATLAB package of iterative regularization methods and large-scale test problems. *Numerical Algorithms*, 81(3):773–811, 2019.

[70] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (6):721–741, 1984.

[71] H. Ghanbari and K. Scheinberg. Proximal quasi-Newton methods for regularized convex optimization with linear and accelerated sublinear convergence rates. *Computational Optimization and Applications*, 69:597–627, 2018.

[72] G. Gilboa and S. Osher. Nonlocal linear image regularization and supervised segmentation. *Multiscale Modeling & Simulation*, 6(2):595–630, 2007.

[73] G. Gilboa and S. Osher. Nonlocal operators with applications to image processing. *Multiscale Modeling & Simulation*, 7(3):1005–1028, 2009.

[74] G. H. Golub and C. F. Van Loan. *Matrix Computations, 4th edition*. Johns Hopkins University Press, Baltimore, 2013.

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

[76] M. Hanke. *Conjugate Gradient Type Methods for Ill-Posed Problems*. Chapman and Hall/CRC, 2017.

[77] M. Hanke and C. W. Groetsch. Nonstationary iterated Tikhonov regularization. *Journal of Optimization Theory and Applications*, 98:37–53, 1998.

[78] M. Hanke and P. C. Hansen. Regularization methods for large-scale problems. *Surveys on Mathematics for Industry*, pages 253–315, 1993.

[79] M. Hanke, J. Nagy, and R. Plemmons. *Preconditioned iterative regularization for ill—posed problems*, pages 141–164. De Gruyter, Berlin, New York, 1993.

[80] P. C. Hansen. *Rank-Deficient and Discrete Ill-Posed Problems*. Society for Industrial and Applied Mathematics, 1998.

[81] P. C. Hansen. *Discrete Inverse Problems: Insight and Algorithms*. Society for Industrial and Applied Mathematics, USA, 2010.

[82] P. C. Hansen, J. G. Nagy, and D. P. O'Leary. *Deblurring Images: Matrices, Spectra, and Filtering*. SIAM, Philadelphia, 2006.

[83] G. Huang, A. Lanza, S. Morigi, L. Reichel, and F. Sgallari. Majorization-minimization generalized Krylov subspace methods for ℓ_p-ℓ_q optimization applied to image restoration. *BIT Numerical Mathematics*, 57:351–378, 2017.

[84] J. Huang, M. Donatelli, and R. Chan. Nonstationary iterated thresholding algorithms for image deblurring. *Inverse Problems in Imaging*, 7(3):717–736, 2013.

[85] A. K. Jain. *Fundamentals of digital image processing*. Prentice-Hall, Inc., USA, 1989.

[86] K. Jiang, D. Sun, and K.-C. Toh. An inexact accelerated proximal gradient method for large scale linearly constrained convex sdp. *SIAM Journal on Optimization*, 22, 2012.

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

[88] A. C. Kak and M. Slaney. *Principles of Computerized Tomographic Imaging*. Society for Industrial and Applied Mathematics, USA, 2001.

[89] C. Kanzow and T. Lechner. Efficient regularized proximal quasi-Newton methods for large-scale nonconvex composite optimization problems. *arXiv:2210.07644*, 2022.

[90] M. Keller, D. Lenz, and R. K. Wojciechowski. *Graphs and Discrete Dirichlet Spaces*. Grundlehren der mathematischen Wissenschaften. Springer, Cham, 2021.

[91] D. Krishnan and R. Fergus. Fast image deconvolution using hyper-Laplacian priors. In *Advances in Neural Information Processing Systems*, pages 1033–1041, 2009.

[92] J. Lampe, L. Reichel, and H. Voss. Large-scale tikhonov regularization via reduction by orthogonal projection. *Linear Algebra and its Applications*, 436(8):2845–2865, 2012. Special Issue dedicated to Danny Sorensen's 65th birthday.

[93] L. Landweber. An iteration formula for fredholm integral equations of the first kind. *American Journal of Mathematics*, 73(3):615–624, 1951.

[94] H. Lantéri, M. Roche, O. Cuevas, and C. Aime. A general method to devise maximum likelihood signal restoration multiplicative algorithms with non-negativity constraints. *Signal Processing*, 81(5):945–974, May 2001.

[95] A. Lanza, S. Morigi, L. Reichel, and F. Sgallari. A generalized Krylov subspace method for ℓ_p-ℓ_q minimization. *SIAM Journal on Scientific Computing*, 37:S30–S50, 2015.

[96] A. Lanza, M. Pragliola, and F. Sgallari. Residual whiteness principle for parameter-free image restoration. *Electronic Transactions on Numerical Analysis*, 53:329–351, 2020.

[97] G. Lauga, E. Riccietti, N. Pustelnik, and P. Gonçalves. Multilevel fista for image restoration. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5, 2023.

[98] C. Lee and S. J. Wright. Inexact successive quadratic approximation for regularized optimization. *Computational Optimization and Applications*, 72:641–674, 2019.

[99] J. D. Lee, Y. Sun, and M. A. Saunders. Proximal Newton-type methods for minimizing composite functions. *SIAM Journal on Optimization*, 24(3):1420–1443, 2014.

[100] F. Li and M. K. Ng. Image colorization by using graph bi-Laplacian. *Advances in Computational Mathematics*, 45(3):1521–1549, 2019.

[101] H. Li, J. Schwab, S. Antholzer, and M. Haltmeier. NETT: Solving inverse problems with deep neural networks. *Inverse Problems*, 36(6):065005, 2020.

[102] J. Liu, T.-Z. Huang, I. W. Selesnick, X.-G. Lv, and P.-Y. Chen. Image restoration using total variation with overlapping group sparsity. *Information Sciences*, 295:232–246, 2015.

[103] Y. Lou, X. Zhang, S. Osher, and A. Bertozzi. Image recovery via nonlocal operators. *Journal of Scientific Computing*, 42(2):185–197, 2010.

[104] D. G. Luenberger and Y. Ye. *Linear and Nonlinear Programming*. Springer, 2015.

[105] F. T. Luk and D. Vandevoorde. Reducing boundary distortion in image restoration. In F. T. Luk, editor, *Advanced Signal Processing: Algorithms, Architectures, and Implementations V*, volume 2296 of *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, pages 554–565, October 1994.

[106] Y. Malitsky and T. Pock. A first-order primal-dual algorithm with linesearch. *SIAM Journal on Optimization*, 28(1):411–432, 2018.

[107] F. G. Meyer and X. Shen. Perturbation of the eigenvectors of the graph Laplacian: Application to image denoising. *Applied and Computational Harmonic Analysis*, 36(2):326–334, 2014.

[108] T. R. Moen, B. Chen, D. R. 3rd Holmes, X. Duan , Z. Yu, L. Yu, S. Leng, J. G. Fletcher G, and C. H. McCollough. Low-dose CT image and projection dataset. *Medical physics*, 48(2):902–911, 2021.

[109] J. J. Moreau. Fonctions convexes duales et points proximaux dans un espace hilbertien. *Comptes rendus hebdomadaires des séances de l'Académie des sciences*, 255:2897–2899, 1962.

[110] E. Morotti, D. Evangelista, and E. L. Piccolomini. A green prospective for learned post-processing in sparse-view tomographic reconstruction. *Journal of Imaging*, 7(8):139, 2021.

[111] E. Morotti, D. Evangelista, and E. L. Piccolomini. Increasing noise robustness of deep

learning-based image processing with model-based approaches. *Numerical Computations: Theory and Algorithms NUMTA 2023*, page 155, 2023.

[112] J. L. Mueller and S. Siltanen, editors. *Linear and Nonlinear Inverse Problems with Practical Applications*, volume 10. Society for Industrial and Applied Mathematics, USA, 2012.

[113] F. Natterer. *The Mathematics of Computerized Tomography*. Society for Industrial and Applied Mathematics, USA, 2001.

[114] Y. Nesterov. A method for solving a convex programming problem with convergence rate $\mathcal{O}(1/k^2)$. *Doklady Akademii Nauk SSSR*, pages 27:372–376, 1983.

[115] M. K. Ng, R. H. Chan, and W.-C. Tang. A fast algorithm for deblurring models with neumann boundary conditions. *SIAM Journal on Scientific Computing*, 21(3):851–866, 1999.

[116] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, New York, NY, USA, 2e edition, 2006.

[117] P. Ochs, Y. Chen, T. Brox, and T. Pock. iPiano: Inertial proximal algorithm for non-convex optimization. *SIAM Journal on Imaging Sciences*, 7(2):1388–1419, 2014.

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

[119] G. Peyré, S. Bougleux, and L. Cohen. Non-local regularization of inverse problems. In D. Forsyth, P. Torr, and A. Zisserman, editors, *Computer Vision – ECCV 2008*, pages 57–68. Springer Berlin Heidelberg, 2008.

[120] M. Piana and M. Bertero. Projected landweber method and preconditioning. *Inverse Problems*, 13(2):441, apr 1997.

[121] T. Pock and A. Chambolle. Diagonal preconditioning for first order primal-dual algorithms in convex optimization. In *Proceedings of the 2011 International Conference on Computer Vision*, volume 1, pages 1762–1769, 2011.

[122] B. Polyak. *Introduction to Optimization*. Optimization Software - Inc., Publication Division, N.Y., 1987.

[123] S. Rebegoldi and L. Calatroni. Scaled, inexact and adaptive generalized FISTA for strongly convex optimization. *SIAM Journal on Optimization*, 32(3):2428–2459, 2022.

[124] L. Reichel and G. Rodriguez. Old and new parameter choice rules for discrete ill-posed problems. *Numerical Algorithms*, 63:65–87, 2013.

[125] L. Reichel, F. Sgallari, and Q. Ye. Tikhonov regularization based on generalized Krylov subspace methods. *Applied Numerical Mathematics*, 62(9):1215–1228, 2012.

[126] R. T. Rockafellar. *Convex Analysis*. Princeton Mathematical Series. Princeton University Press, Princeton, N. J., 1970.

[127] R. T. Rockafellar and R. J. B. Wets. *Variational Analysis*. Grundlehren der Mathematischen Wissenschaften, 1998.

[128] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015.

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

[130] M. Schmidt, N. Le Roux, and F. Bach. Convergence rates of inexact proximal-gradient methods for convex optimization. *arXiv:1109.2415v2*, 2011.

[131] S. Serra-Capizzano. A note on antireflective boundary conditions and fast deblurring models. *SIAM Journal on Scientific Computing*, 25(4):1307–1325, 2004.

[132] L. A. Shepp and Y. Vardi. Maximum likelihood reconstruction for emission tomography. *IEEE transactions on medical imaging*, 1(2):113–122, 1982.

[133] J. Stoer, R. Bulirsch, R. Bartels, W. Gautschi, and C. Witzgall. *Introduction to numerical analysis*, volume 1993. Springer, 1980.

[134] A. Susnjara, N. Perraudin, D. Kressner, and P. Vandergheynst. Accelerated filtering on graphs using Lanczos method. *arXiv preprint arXiv:1509.04537*, 2015.

[135] A. N. Tikhonov. Solution of incorrectly formulated problems and the regularization method. *Soviet Mathematics Doklady*, 4:1035–1038, 1963.

[136] E. Tjoa and C. Guan. A survey on explainable artificial intelligence (xai): Toward medical xai. *IEEE Transactions on Neural Networks and Learning Systems*, 32(11):4793–4813, 2020.

[137] S. Villa, S. Salzo, L. Baldassarre, and A. Verri. Accelerated and inexact forward-backward algorithms. *SIAM Journal on Optimization*, 23(3):1607–1633, 2013.

[138] H. Voss. An Arnoldi method for nonlinear eigenvalue problems. *BIT Numerical Mathematics*, 44(2):387–401, 2004.

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

[140] A. C. Yağan and M. T. Özgen. A spectral graph wiener filter in graph fourier domain for improved image denoising. In *2016 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 450–454. IEEE, 2016.

[141] Q. Yang, D. Chen, T. Zhao, and Y. Chen. Fractional calculus in image processing: a review. *Fractional Calculus and Applied Analysis*, 19(5):1222–1249, 2016.

[142] M.-C. Yue, Z. Zhou, and A. M. C. So. A family of inexact SQA methods for non-smooth convex minimization with provable convergence guarantees based on the Luo–Tseng error bound property. *Mathematical Programming*, 174:327–358, 2019.