Alma Mater Studiorum–Università di Bologna

DOTTORATO DI RICERCA IN MATEMATICA
Ciclo XXXIV

Settore Concorsuale: 01/A5 - ANALISI NUMERICA
Settore Scientifico Disciplinare: MAT/08 - ANALISI NUMERICA

VARIATIONAL AND LEARNING MODELS FOR IMAGE AND TIME SERIES INVERSE PROBLEMS

Presentata da: Pasquale Cascarano

Coordinatore Dottorato:
Prof.ssa Valeria Simoncini

Supervisore:
Prof.ssa Elena Loli Piccolomini

Co-supervisori:
Prof. Andrea Pascucci
Prof. Stefano Gandolfi

Esame finale anno 2022

Al mio papà, alla mia mamma,
a Lara, a Mari, a Gigi...

"We can only see a short distance ahead, but we can see plenty there that needs to be done"
- Alan Turing (1950), father of modern computer science.

ABSTRACT

Inverse problems are at the core of many challenging applications arising in the field of natural sciences, medicine, engineering and industry. Variational and learning models provide estimated solutions of inverse problems as the outcome of specific reconstruction maps. More precisely, in the variational approach, the result of the reconstruction map is the solution of a regularized minimization problem designed to encode information on the acquisition process and prior knowledge on the solution. Whereas, in the learning approach, the reconstruction map is a parametric function whose parameters are identified by solving a minimization problem depending on a large set of data.

In this thesis, we go beyond this apparent dichotomy between variational and learning models and we show they can be harmoniously merged in unified hybrid frameworks preserving their main advantages. We develop several highly efficient methods based on both these model-driven and data-driven strategies, for which we provide in most cases a detailed convergence analysis.
The arising algorithms are applied to solve real inverse problems involving images and time series. For each task, we show the proposed schemes improve the performances of many other existing methods in terms of both computational burden and quality of the solution.
In the first part, we focus on gradient-based regularized variational models which are shown to be effective for segmentation purposes and for thermal and medical image enhancement. We consider gradient sparsity-promoting regularized models for which we develop different strategies to estimate the regularization strength. Furthermore, we introduce a novel gradient-based Plug-and-Play convergent scheme considering a deep learning based denoiser trained on the gradient domain.
In the second part, we address the tasks of natural image deblurring, image and video super resolution microscopy and positioning time series prediction, through deep learning based methods.
We boost the performances of widely used supervised, such as end-to-end trained convolutional and recurrent neural networks, and unsupervised deep learning strategies, such as Deep Image Prior, by penalizing the training losses with those handcrafted regularization terms typically used in the variational framework.

Keywords: inverse problems, variational models, learning models, convolutional neural networks, deep image prior, image super resolution, image deblurring, time series prediction.

Contents

List of Acronyms

ADMM	Alternating Direction Method of Multipliers
AWGN	Additive White Gaussian Noise
BCCB	Block Circulant with Circulant Block
CG	Conjugate Gradient
CNN	Convolutional Neural Network
DIP	Deep Image Prior
DNN	Deep Neural Network
FBS	Forward Backward Splitting
FFT	Fast Fourier Transform
FISTA	Fast-iterative shrinkage-thresholding
GNSS	Global Navigation Satellite System
HQS	Half-Quadratic Splitting
IR	Image Restoration
JAC	Jaccard index
LSTM	Long Short-Term Memory
MAP	Maximum a-posteriori
MISR	Multiple Image Super Resolution
MLE	Maximum likelihood estimation
PnP	Plug-and-Play
PSF	Point Spread Function
PSNR	Peak Signal-to-Noise Ratio
ReLu	Rectified Linear Unite
RNN	Recurrent Neural Newtork
SI	System Identification
SISR	Single Image Super Resolution
SNR	Signal to Noise Ration
SR	Super Resolution
SSIM	Structural Similarity Index Measure
TV	Total Variation

Introduction

Various tasks in signal processing can be casted as inverse problems, namely they rely on the estimation of unknown variables from indirect observations. The inverse problems are typically hard to solve since the observations present a low level of information about the unknowns.
In the last decades, variational and learning paradigms have been adopted to address this issue and to provide approximate solutions of such problems.
Approaches based on the variational framework estimate the unknowns as the solutions of unconstrained and constrained minimization problems including fidelity and regularization terms. The fidelity term is task-specific and encodes information about the degradation affecting the measurements. The regularization term is typically defined to induce prior knowledge on the estimate. Designing effective variational models is not a straightforward issue, since it requires deep knowledge about the physics of the acquisition process and a complex modeling of the statistics of the unknowns.
The availability of a huge amount of data, the development of more advanced algorithms capable to handle highly complex models and the rise of the graphics processing units (GPUs) heightening the processing power, has led the growth of deep learning which has revolutioned the field of inverse problems in signal processing. More specifically, the learning approach considers non-linear parametric networks whose parameters are calibrated to the task at hand by exploiting a varied set of training examples. These highly representative networks extract high-level features from a training set and then transfer this information to unseen data. Conversely to variational approaches, these learning approaches reach state-of-the-art performances but they are not theoretically well-understood and, moreover, they can perform poorly for data whose statistics is not well described by the fixed training set.
The aim of this thesis is to design highly efficient methods merging the advantages of both variational and deep learning frameworks. The arising schemes are then adapted to solve several image related tasks in the field of medicine, biology and remote sensing, and time series related tasks for structural monitoring in engineering.
Due to the COVID-19 pandemic and the consequent travel restrictions, for the author was not possible to go abroad. However, this thesis is the result of his remote collaboration with various research teams working on different fields of applied mathematics and engineering.
The main research group is affiliated to the Mathematics Department of the University of Bologna, where the author joined the PhD program in Mathematics. The other groups are based in the Department of Civil, Chemical, Environmental and Materials Engineering of the University of Bologna, in the Department of Physics, Informatics and Mathematics of the University of Modena

and Reggio Emilia, in the Department of Electronic Engineering of Tor Vergata in Rome, in the Institute of Computer Graphics and Vision of the Graz University of Technology and in the I3S Laboratory at the Université Côte d'Azur in Sophia-Antipolis.

Contribution and Outline

This thesis is organized in three parts and considers eight chapters. In Part I we develop some preliminaries and we set the notations used in the following dissertation. In Part II we focus on variational models, whereas Part III relies on learning models, for real image and time series inverse problems.

Part I is divided in two chapters:

- In Chapter 1 we describe the inverse problems of reference, namely system identification and image restoration.
- In Chapter 2 we first motivate the reason why these two tasks can be regarded as ill-posed problems. Then, we show that estimated solutions can be provided by adopting variational and learning models. Furthermore, we show how understanding both in a Bayesian framework helps us to provide useful insights into the design of proper energy functions and training losses for the variational and learning framework, respectively. As far as variational models are concerned, we first highlight merits and drawbacks of handcrafted regularization and, in particular, we focus on Tikhonov and Total Variation based regularizers. Then, we introduce learning-based regularizers. More precisely, we focus on the so-called deep Plug-and-Play regularizers and the Regularization by Denoising functional. As far as learning models are concerned, we first refer to pure learning models exploiting large datasets and high representative functions, like deep neural networks. Then, we go beyond standard end-to-end trained models by introducing the so-called Deep Image Prior framework. According to this framework, high-level features can be captured by the sole structure of a neural network fitted to a single degraded measurement, thus avoiding the need of a training set.

Part II is divided in three chapters in which we show how to solve different image related tasks through variational models.

- In Chapter 3 we propose a new super resolution algorithm which can handle either single or multiple images. It is based on a Total Variation regularization approach and implements a fully automated choice of all the parameters. The proposed algorithm is used to mitigate the problem of relatively poor spatial resolution in thermal remote sensing applications.
- In Chapter 4 we consider constrained and unconstrained variational models for single image super resolution based on the assumption that the gradient of the target image is sparse. We enforce this assumption by considering both an isotropic and an anisotropic ℓ_0 regularization on the image gradient for promoting piecewise constant solutions. Thus, we propose algorithms addressing the problem of joint single image super resolution and image partitioning.

The algorithms rely on novel efficient Alternating Direction Method of Multipliers schemes whose substeps are solved efficiently by means of hard-thresholding and standard conjugate-gradient solvers or, upon suitable assumptions, admit closed-form solutions. Finally, we prove a fixed-point convergence theorem for the arising schemes.

- In Chapter 5 we propose a novel gradient-based Plug-and-Play algorithm and we apply it to restore CT images. The plugged denoiser is implemented as a deep Convolutional Neural Network trained on the gradient domain and not on the image one, as in state-of-the-art works on Plug-and-Play. We further consider a hybrid algorithm combining the gradient-based denoiser with the Total Variation functional. The proposed frameworks rely on the Half-Quadratic Splitting scheme for which we prove a general fixed-point convergence theorem, under weak assumptions on both the involved denoisers.

Part III is divided in three chapters in which we show how to solve several image and time series related tasks by exploiting deep neural networks.

- In Chapter 6 we propose a deep learning-based architecture for super resolution in fluorescence microscopy which aims at localizing the molecules in high density frames acquired by single molecule localization techniques. The neural network is trained considering a loss function composed by an ℓ_2-norm based term regularized by non-negative and ℓ_0-based constraints. The ℓ_0 term is relaxed through its continuous ℓ_0 counterpart. The arising approach, is parameter-free, more flexible, faster and provides more precise molecule localization maps if compared to the other state-of-the-art methods.

- In Chapter 7 we propose a regularized Deep Image Prior unconstrained model combining an ℓ_2-norm based term with separable space-variant regularizers whose local regularization parameters are automatically estimated. In order to enable the application of a broad range of regularizers, we propose a novel Morozov's discrepancy principle constrained formulation of Deep Image Prior. The proposed unconstrained and constrained models are solved via the Alternating Direction Method of Multipliers. The arising algorithms are shown to be robust with respect to the choice of hyperparameters.
  Furthermore, we present a new deep learning-based algorithm that extends Deep Image Prior to super resolution for time-lapse microscopy videos. The arising algorithm introduces some novelties: the weights of the Deep Image Prior network architecture are initialized for each of the frames according to a new recursive updating rule combined with an efficient early stopping criterion and, finally, the Deep Image Prior loss function is penalized by the handcrafted Total Variation-based term.

- In Chapter 8 we propose to solve three tasks in Global Navigation Satellite Systems time series analysis, namely denoising, prediction and jump detection, by using the LSTM-Full architecture, based on long short-term memory recurrent neural networks.

Related Publications

This thesis mainly refers to the following published papers and pre-prints.

Part II is based on:

- Cascarano, P., Corsini, F., Gandolfi, S., Loli Piccolomini, E., Mandanici, E., Tavasci, L., Zama, F. *Super-resolution of thermal images using an automatic total variation based method.* Remote Sensing, 2020.
- Cascarano, P., Calatroni, L., Loli Piccolomini, E. *Efficient ℓ^0 gradient-based Super Resolution for simplified image segmentation.* IEEE Transactions on Computational Imaging, 2021.
- Mylonopoulos, D., Cascarano, P., Calatroni, L., Loli Piccolomini, E. *Constrained and unconstrained inverse Potts modeling for joint image super-resolution and segmentation.* Image Processing On Line, 2022.
- Cascarano, P., Loli Piccolomini, E., Morotti, E., Sebastiani, A. *Plug-and-Play gradient-based denoisers applied to CT image enhancement.* Applied Mathematics and Computation, 2022.

Part III is based on:

- Cascarano, P., Sebastiani, A., Comes, M.C, Franchini, G., Porta, F. *Combining Weighted Total Variation and Deep Image Prior for natural and medical image restoration via ADMM.* To appear on IEEE book of conference proceedings of International Conference on Computational Science and Its Applications (ICSSA 2021).
- Cascarano, P., Comes, M. C., Mencattini, A., Parrini, M. C., Loli Piccolomini, E., Martinelli, E. *Recursive Deep Prior Video: a Super Resolution algorithm for Time-Lapse Microscopy of organ-on-chip experiments.* Medical Image Analysis, 2021.
- Cascarano, P., Franchini, G., Kobler, E., Porta, F., Sebastiani, A. *Constrained and unconstrained Deep Image Prior optimization models with automatic regularization.* Submitted in October 2021.
- Cascarano, P. and Comes, M.C., Sebastiani, A., Mencattini, A. and Loli Piccolomini, E. and Martinelli, E. *DeepCEL0 for 2D single-molecule localization in fluorescence microscopy.* Bioinformatics, 2021.
- Loli Piccolomini, E., Gandolfi, S., Poluzzi, L., Tavasci, L., Cascarano, P., Pascucci, A. *Recurrent Neural Networks Applied to GNSS Time Series for Denoising and Prediction.* Book of proceedings of 26th International Symposium on Temporal Representation and Reasoning (TIME 2019).

Part I

Preliminaries

Chapter 1

Understanding the world through signals

The world we live in is full of *signals* which are the most common ways of communication. Among the most widespread form of signals we mention sounds, videos and images. However, any physical quantity exhibiting variation in space or time can be regarded as a signal.

Understanding the Earth's physical phenomena has always been the primary goal for scientists from all over the world and signals play a fundamental role for the purpose since they convey information about the object of study. The latter can be interpreted as an emitting *source* whose emitted energy is then acquired by a *system* endowed with a *receiver*. The system serves as a converter transforming the emitted signal, in the following referred to as *input signal* or *object*, into another form of energy which is referred to as *output signal* or *measurement*.
For instance, a microphone (system) converts an acoustic signal (input signal) to a voltage waveform (output signal), a camera (system) converts light energy (input signal) into electrical energy (output signal) or a GPS device (system) converts a radio wave (input signal) into real-time positioning data (output signal).
The output signal represents the response of the system to a given external stimulus, thus revealing important information about the object of study.
As an example, in structural monitoring applications, real-time positioning data, provided by the Global Navigation Satellite Systems technology, can be used to infer the stability of a building. Moreover, in bio-medicine, images recorded from sources of radiation, as it happens in Computed Tomography, Magnetic Resonance Imaging or fluorescence microscopy, are used to support the diagnosis or to reveal important biomedical insights.

Sounds, videos and images, are analog signals meaning that they are defined on continuous domains and they have continuous values. This continuous representation is not compatible with the computers used to process such data. Therefore, a digital representation with discrete time or space domain and amplitude quantized values is usually provided by a *recorder*. Such digital representation stems from the need to efficient process and store the information carried by the signals. Consequently, whilst analog signals can be mathematically described as functions defined on a continuous domain, digital signals can be viewed as vectors whose components represent

Figure 1.1: A schematic representation of the emitting, transforming, recording and storing phases in a signal acquisition process.

samples of such analog data.

The signal acquisition process can be mainly divided in four phases: emitting, transforming, recording and storing phases (see Figure 1.1 for a schematic representation). Unfortunately, during these phases, many sources of error due to the digitization and compression of the signal and/or to other impairments happening during the transmission, may corrupt the recorded measurement with aberrations, thus negatively affecting a subsequent signal analysis and limiting the understanding of the phenomenon at hand. Digital signal processing strategies aim at manipulating the discrete corrupted signals to improve their quality, storage efficiency and/or also to extract interesting and reliable insights.

In real applications, either the system or its input or output may be unknown and their identification represents an important challenge. The level of knowledge permits classifying the problems in two broad classes, namely *forward problems* and *inverse problems*. In the following part of this chapter, we describe these two classes of problems and we introduce two famous tasks in signal processing, namely *system identification* (SI) and *image restoration* (IR) both belonging to the class of inverse problems.

1.1 Forward and inverse problems in signal processing

Given an open set $\Omega_y \subset \mathbb{R}^{d_y}$, in the following we refer to an analog signal y as the function $y : \Omega_y \to \mathbb{R}^{c_y}$ lying in an Hilbert space, where Ω_y is called *acquisition domain*.
By setting $d_y = 1, 2, 3$, the function y is referred to as 1-dimensional (1D), 2-dimensional (2D) and 3-dimensional (3D) signal, respectively. Furthermore, when $c_y = 1$ or $c_y > 1$ the signal y is referred to as single-channel or multiple channel signal, respectively. In particular, when $d_y = 2$ and $c_y = 1$, y is referred to as gray scale image, or single-channel image, whereas when $c_y > 1$, y is referred to as multi-channel image. As an example, when $c_y = 3$, y assumes values on three color channels, as in the case of YCbCr or RGB images.

The first step of any signal processing strategy is to develop a model describing the behaviour of the real system at hand using mathematical tools. We assume $u \in \mathcal{X}$ is the input signal convey-

ing information about a physical object, whereas we denote by $b_0 \in \mathcal{Y}$ the output of the system. The physical process mapping u to b_0 is modeled as a continuous map $\mathcal{A}$ between the Hilbert spaces $\mathcal{X}$ and $\mathcal{Y}$. In formula:

$$b_0 = \mathcal{A}(u), \tag{1.1}$$

where the map $\mathcal{A}$ is referred to as *system* in the following.
Numerous applications and scientific disciplines rely on the prediction of the effect b_0 of a given cause u, or conversely, on the estimation of the unknown cause u producing the observed effect b_0. These two tasks are referred to as *forward problem* and *inverse problem*, respectively.

Forward Problem

$$\begin{array}{ccc} \textit{data} & & \textit{unknown} \\ u & \xrightarrow[\mathcal{A}]{} & b_0 \end{array}$$

Inverse Problem

$$\begin{array}{ccc} \textit{data} & & \textit{unknown} \\ b_0 & \xrightarrow[\mathcal{A}^{-1}]{} & u \end{array}$$

The solution of a forward problem is usually a physical quantity with a low level of information. Conversely, the inverse problem aims at retrieving the information lost during the acquisition. However, in real applications, we only dispose of b, a degraded counterpart of b_0. The presence of these measurement errors, which are mostly related to the emitting and recording phases of the signal acquisition process, is typically modeled as a random noise operator $\mathcal{N}$. The *acquisition model* of reference, relating the object u to the corrupted measurement b, reads:

$$b = \mathcal{N}\left(\mathcal{A}(u)\right). \tag{1.2}$$

Inverse problems are usually more challenging than forward problems and this can be somehow explained by recalling a typical mathematical property which is known as *ill-posedness*.
According to the definition given by the mathematician J. Hadamard in the early 20th century [1], a *well-posed problem* is a problem whose solution satisfies the following requirements:

- existence
- uniqueness
- continuous dependence on the initial data.

On the contrary, the solution of an *ill-posed problem* does not fulfil at least one of the previous conditions.
It is well-known that forward problems are usually well-posed problems whereas the associated inverse problems are ill-posed problems [2, 3]. The well-posedness of forward problems is mainly due to the continuity of the system $\mathcal{A}$. Nevertheless, sometimes the range of $\mathcal{A}$ does not coincide with the whole space $\mathcal{Y}$. Therefore, as far as inverse problems are concerned, the solution u may not exist since, in general, $b \notin \mathbf{range}(\mathcal{A})$. Moreover, as a consequence of the loss of information happening during the acquisition process, two very distant objects can have measurements which

are very close, hence the solution of an inverse problem could be not unique. Finally, the inverse of the map $\mathcal{A}$ is, in general, not continuous, hence we may have two similar measurements such that the corresponding objects are very distant, that is the solution of the inverse problem does not depend continuously on the data and is very sensitive to measurement errors.

The acquisition model (1.2) considers b and u as analog signals, namely they are thought as uninterrupted observations (e.g. on a time or space domain) of some physical quantities. However, this analog representation is not compatible with the current instrumentation and computer technology. In the following, the vector $\mathbf{u} \in \mathbb{R}^n$ denotes the discretization of the object u, the vector $\mathbf{b}_0, \mathbf{b} \in \mathbb{R}^m$ denote the discretizations of b_0 and b, respectively. Finally, by $\mathbf{A} : \mathbb{R}^n \to \mathbb{R}^m$ we refer to the discrete counterpart of the system $\mathcal{A}$.

In real applications, $\mathbf{A}$ and $\mathbf{u}$ may be treated either as data or unknowns. The task of identifying them can be casted as an inverse problem. In the following sections, we will introduce the two inverse problems of interest, namely *system identification* and *image restoration*. The following table summarises the substantial differences between the two tasks.

	System Identification	Image Restoration
Input signal ($\mathbf{u}$)	Known	Unknown
System ($\mathbf{A}$)	Unknown	Known
Output signal ($\mathbf{b}$)	Known	Known

1.1.1 Aliasing: the curse of digitization

In this thesis we deal with 1D and 2D signals, and more precisely with time series and images. We now briefly show some issues occurring when moving from their analog to their digital formulation and we discuss about the *aliasing* phenomena causing two different signals to become indistinguishable after sampling.

A time series u is a 1D analog signal defined on a time domain $\Omega_u \subset \mathbb{R}$, representing the time interval during which the variable u is observed. The discrete counterpart of u is simply a finite sequence of values separated in time.

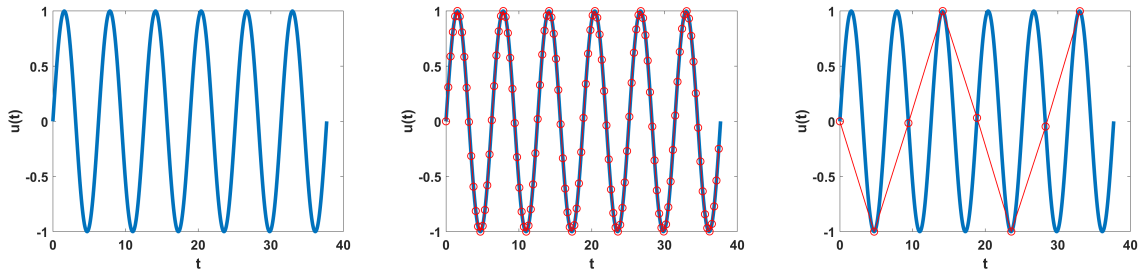
In Figure 1.2a we show a sinusoidal time series $u(t) = sin(t)$ for $t \in \Omega_u := [0, 12\pi]$. In Figure 1.2b we represent its discrete counterpart $\mathbf{u} \in \mathbb{R}^n$ sampled at a frequency $f_s := \dfrac{1}{\Delta t}$, where $\Delta t = \dfrac{\pi}{10}$, that is u is sampled in $n = 120$ equally spaced points. Therefore, the discrete signal can be thought as a vector whose i-th component is defined as $\mathbf{u}_i = u(t_i)$, where $t_i = i \cdot \Delta t$ for $i = 1 \dots n$.

However, sampling the analog signal may cause profound effects that must be either avoided or accounted for. We discuss about these implications by using a numerical example.

The analog signal u in Figure 1.2a has period $T_0 = 2\pi$. However, in Figure 1.2c we can observe that when the sampling rate Δt is high the signal is undersampled and its periodicity appears aliased into a signal of lower frequency content. It can be shown that to avoid this issue the sampling frequency f_s should exceed the Nyquist frequency f_{Nyq} [3]:

$$f_s > f_{Nyq} := \frac{2}{T_0}.$$

We now consider an image u, that is an analog 2D signal defined on space domain $\Omega_u \subset \mathbb{R}^2$. The discrete counterpart of the image u is a matrix $\mathbf{U} \in \mathbb{R}^{N_r \times N_c}$ whose elements are referred to



(a) (b) (c)

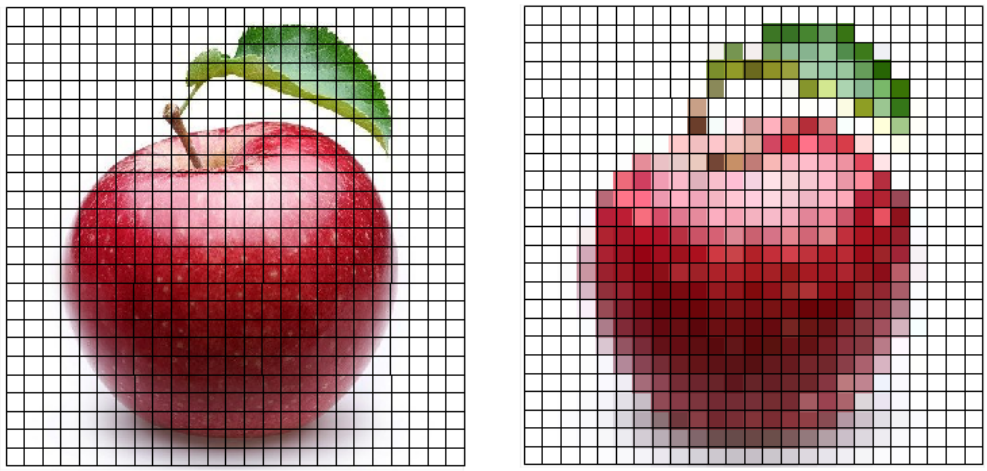
Figure 1.2: (a) analog sinusoidal signal of period $T_0 = 2\pi$. (b) analog signal (blue line) and digital signal sampled at $\Delta t = \frac{\pi}{10}$ (red markers). (c) analog signal (blue line) and aliased signal sampled at $\Delta t = \frac{3}{2}\pi$ (red markers).

as pixels. We stress that it is a common practice to refer to the discrete image $\mathbf{U}$ also as a vector $\mathbf{u} \in \mathbb{R}^n$ obtained by simply stacking the columns of $\mathbf{U}$. Hence, $n = N_r \cdot N_c$.
In Figure 1.3a we depict the `apple` image having an high pixel count (2500×1500 pixels) so that it can be considered as an analog signal u. In Figure 1.3b we report its sampled version $\mathbf{U}$ corresponding to a very rough discretization grid (25×25 pixels).
As it happens for 1D signals, the sampling should be done by avoiding the aliasing artifacts. Indeed, the aliasing phenomenon can also occur in spatially sampled signals causing the so-called Moiré patterns.
As an example, in Figure 1.4a we depict the `stripes` image (600×600 pixels) having an high spatial frequency and in Figure 1.4b we show its sampled counterpart (150×150 pixels) where the Moirè pattern is clearly visible. In this case, aliasing is generally avoided by applying low-pass filters to the analog signal before sampling. In Figure 1.4c we report the image (150×150 pixels) obtained by applying to the `stripes` image a low pass filter before sampling and, in this case, we can observe the Moirè artifacts are clearly suppressed.

1.1.2 Noise models

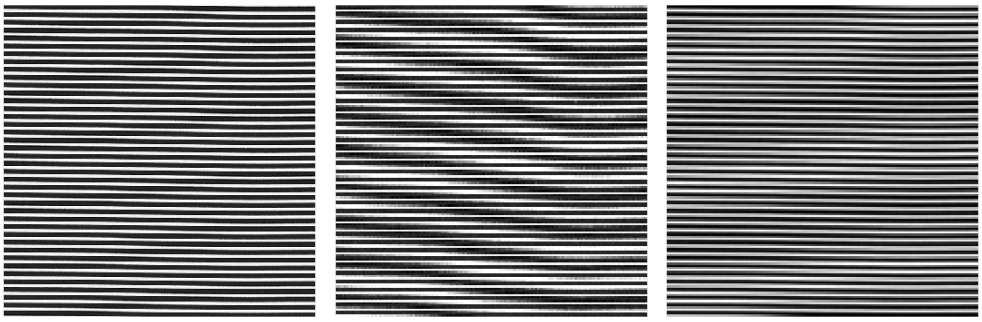
The noise corrupting the acquired data is the result of the contribution of different sources. For simplicity, in (1.2) we have assumed it is modeled as the random operator $\mathcal{N}$. We now list the noise models we will use throughout the following chapters.

Gaussian noise. A noise arising in many applications is the Additive White Gaussian Noise (AWGN). Hence, we assume $\mathcal{N}$ is a zero-mean m-dimensional multivariate random variable normally distributed with covariance matrix $\Sigma = \sigma^2 \mathbf{I}_m$, where by $\mathbf{I}_m$ we denote the identity matrix with m rows and columns. In particular, we denote by $\mathbf{e} \in \mathbb{R}^m$ the additive component, sampled from $\mathcal{N}$, affecting $\mathbf{b}$. More specifically, the entries $\mathbf{e}_i$ of $\mathbf{e}$, for $i = 1 \dots m$, are identically independently distributed (i.i.d.) realizations of the distribution. Hence, the probability density function of the random vector $\mathbf{e}$, reads:



(a) (b)

Figure 1.3: Spatial discretization for 2D signals. (a) analog `apple`. (b) sampled `apple`.



(a) (b) (c)

Figure 1.4: Aliasing and Moirè patters for 2D signals. (a) analog `stripes`. (b) aliased sampled `stripes` with Moirè patterns. (c) non aliased sampled `stripes`.

$$p(\mathbf{e}) = \prod_{i=1}^{m} p(\mathbf{e}_i) = \prod_{i=1}^{m} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{\mathbf{e}_i^2}{2\sigma^2}\right) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{\|\mathbf{e}\|_2^2}{2\sigma^2}\right).$$

The AWGN model is widely used since, in most cases, it produces a reasonable and satisfying approximation of the observed noise.

Poisson noise. In bio-medical imaging problems a more effective noise model follows a Poisson distribution. Differently from AWGN, Poisson noise is a signal-dependent noise, since its standard deviation σ depends on the intensity of the signal. Hence, we assume $\mathbf{b}_i$, the i-th component of $\mathbf{b}$, is the realization of a Poisson random variable whose probability density function reads:

$$p(\mathbf{b}_i) = \frac{\mu_i^{\mathbf{b}_i} e^{-\mu_i}}{\mathbf{b}_i!}$$

where by μ_i we refer to the mean value for $i = 1 \dots m$.

The mean value μ_i is directly proportional to the the number of photons hitting the sensors and, as a consequence, the same holds for the noise standard deviation σ_i which is related to μ_i according to this formula $\sigma_i = \sqrt{\mu_i}$. We remark that when the number of photons is sufficiently large, the Poisson distribution can be approximated by a Gaussian distribution.

Brownian noise. When treating signals defined on a time domain the AWGN model is usually combined with the Brownian noise model, which is produced by a Brownian motion and, in the literature, it is also known as random walk noise model. Differently from AWGN which can be produced by randomly choosing each sample independently, Brownian noise is produced by adding a random offset (usually a realization of a univariate normal distribution) to each sample to obtain the next one, hence two consecutive samples are not independent and identically distributed as it happens for AWGN.

1.2 System identification

A system can be thought as a transducer converting an input signal into an output signal which is, sometimes, easier to observe/read and to handle [4]. *System Identification* (SI) aims at estimating a real system using a dataset of input-output pairs describing its behaviour. Based on the literature, SI is used in many fields of industry and science, such as astronomy, biology, medicine and engineering. Particularly, in this thesis we are interested in SI applied to design and implement control systems or to plan and monitor engineering projects, as for example proposed in [5, 6]. The efficiency of SI strictly relies on the design of the dataset which can be either obtained from a stored database or collected by exciting the real system with predefined inputs. A pre-processing of the given data, applied to ensure they are within an acceptable range, to remove meaningless trends or noise, to fill missing data or to simply adjust the size of the data, can increase the performances of the method used to solve the SI task. Among the most used pre-processing techniques we mention: *magnitude scaling* which is particularly useful to adjust the magnitude of the data, *filtering* employed to remove noise and trends, *interpolation* which aims at estimating missing points in small datasets and *resampling* that is typically used to reduce the number of data points to a more manageable value.

In the next section we show how the SI task can be read as an inverse problem, whereas in Chapter 2 we show how an estimate of the real system can be provided.

1.2.1 System identification as inverse problem

A mathematical model for a real system can be derived from first principles which have a relatively small amount of parameters to be identified, all of them having a precise physical interpretation. This happens, for example, with those real systems described by partial differential equations which define the connection between the involved variables.
In real applications, we usually have available input and output of the real system and not physical laws describing its behaviour. Let $\mathcal{D} := \{(\mathbf{u}^j, \mathbf{b}^j)\}_{j=1\dots N}$ with $\mathbf{u}^j \in \mathbb{R}^n$ and $\mathbf{b}^j \in \mathbb{R}^m$, be a dataset of input-output pairs describing an unknown system $\mathbf{A} : \mathbb{R}^n \to \mathbb{R}^m$. By assuming additive noise,

we have:

$$\mathbf{b}^j = \mathbf{A}(\mathbf{u}^j) + \mathbf{e}^j \quad \text{for} \quad j = 1 \dots N, \tag{1.3}$$

where the component $\mathbf{e}^j \in \mathbb{R}^m$ corrupting $\mathbf{b}^j$ includes both Brownian noise and Gaussian noise.
The strategy is to consider a general purpose family of models containing many parameters which are adapted to the real system at hand by exploiting the set of data $\mathcal{D}$. Therefore, we assume $\mathbf{A}$ belongs to a parametric space of operators $T_\Theta := \{\mathbf{T}(\cdot, \boldsymbol{\theta}) : \mathbb{R}^n \to \mathbb{R}^m \mid \boldsymbol{\theta} \in \Theta\}$, where by $\Theta \subset \mathbb{R}^p$ we denote the space of parameters. In other words, we assume there exists $\boldsymbol{\theta} \in \Theta$ such that $\mathbf{A} = \mathbf{T}(\cdot, \boldsymbol{\theta})$.
Upon these assumptions, the model in (1.3) is said *parametric*, whereas when minimal or no prior assumptions on the structure of $\mathbf{A}$ are made, the model is referred to as *non-parametric*. Although the non-parametric approach is more practical and flexible in cases where system's physical insights are unavailable, it makes the modeling task significantly more complex, as the model class options are extremely rich and there are more unknowns that need to be considered.
Therefore, the SI task of reference reads as the following inverse problem:

$$\text{find} \quad \mathbf{A} \in T_\Theta, \quad \text{s.t} \quad \mathbf{b}^j = \mathbf{A}(\mathbf{u}^j) + \mathbf{e}^j \quad \text{for} \quad j = 1 \dots N. \tag{1.4}$$

We remark that differently from first principles the parameters have no physical or logical interpretation. However, looking for the parameters of the system is an ill-posed inverse problem (see Chapter 2), making the identification task (1.4) extremely challenging in real applications.

For example, we can assume T_Θ is a parametric space of linear operators with respect to the vector of parameters $\boldsymbol{\theta}$, hence $\mathbf{A}$ is assumed to be linear.
Among the most widespread parametric linear models we mention the finite impulse response (FIR) model [7], which can be regarded as a particular instance of (1.3). More precisely, the FIR model reads:

$$b^j = \boldsymbol{\theta} \cdot \mathbf{u}^j + e^j \quad \forall j = 1 \dots N, \tag{1.5}$$

where we assume the output b^j are scalars, the input $\mathbf{u}^j$ are n-dimensional vectors, for $j = 1 \dots N$, and $\boldsymbol{\theta} \in \mathbb{R}^n$ represents the vector of parameters to be identified. The N equations in (1.5) can be written as an $N \times n$ linear system. Therefore, assuming a FIR model entails that estimating $\boldsymbol{\theta}$ is equivalent to solve a linear system. Whenever a non linear model is considered, which is the most interesting case, other strategies must be used to identify the parameters. We remark that linear models fails when the real system exhibits a strong non-linear behavior, as it happens for time series prediction. Indeed, in this case, the systems involved are highly non-linear and, in terms of representativeness of the phenomenon, a non-linear model is more advantageous. The interested reader can refer to [7] for further information on linear and non-linear models.
In the following chapters we will assume $\mathbf{A}$ belongs to the class of Neural Networks as non-linear models [7] and we will show how to estimate the set of parameters.

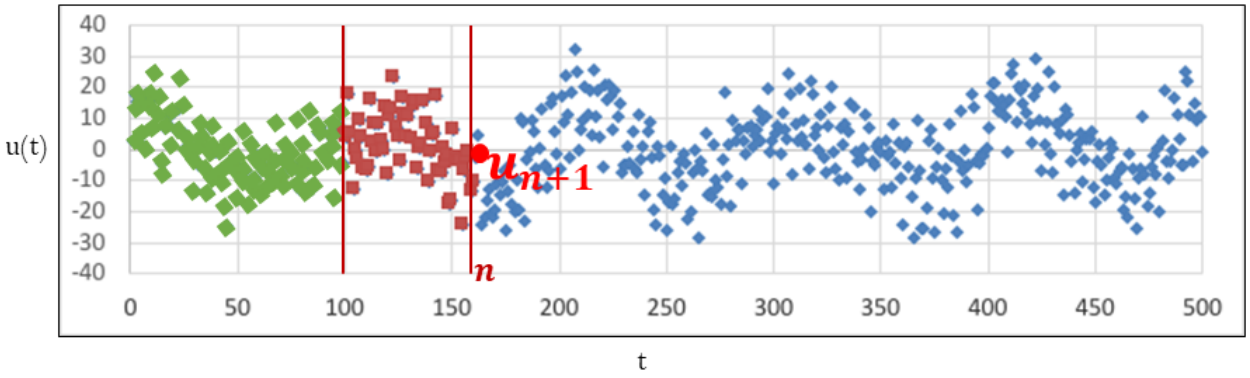


Figure 1.5: In time series prediction, a batch of h samples (red squares) is used for the prediction of $\mathbf{u}_{n+1}$ (red dot). Finally, the predicted value can be compared to the measured one (blue dot), for example to detect some anomalies.

An example of SI task: time series prediction

Let us consider a time series u defined on the time domain $\mathcal{T} := [0, T] \subset \mathbb{R}$, with $T > 0$. For each $t \in [0, T]$, we assume $u(t) \in \mathbb{R}$ describes a measurement of a phenomenon evolving in the time interval $\mathcal{T}$. We consider $\mathbf{u} \in \mathbb{R}^n$ such that $\mathbf{u}_i = u(t_i)$ where $t_i \in \mathcal{T}$ for $i = 1 \dots n$.
Prediction is one of the most important tasks in time series analysis. More precisely, from the knowledge of $\mathbf{u}$ the prediction task aims at estimating the value $\mathbf{u}_{n+1}$ which is sampled at a time step $t > T$. The model describing the behaviour of the time series is unknown. Therefore, in practice, we consider a general purpose non-linear parametric model such that the prediction task can be regarded as a particular instance of the SI inverse problem (1.4). Therefore, we need to estimate a non-linear map $\mathbf{A} \in T_\Theta$ such that:

$$\mathbf{u}_i = \mathbf{A}(\mathbf{u}_{i-1}, \dots, \mathbf{u}_{i-h}) \quad \text{for} \quad i = h+1, \dots, n. \tag{1.6}$$

where we assume the predicted values depends on a batch of $h < n$ previous samples. If we are able to provide a reliable on-line prediction of the time series behaviour, by comparing the predicted and the measured signals, we can extract meaningful insights, such as an abnormal behaviour of the phenomenon of study. In Figure 1.5 we report a schematic representation of the time series prediction task.

1.3 Image restoration

Nowadays, several applications such as digital photography, positron emission tomography (PET), magnetic resonance imaging (MRI) and computed tomography (CT), make use of signals which can be somehow visualized as 2D or 3D images, so as to allow a better subsequent analysis of the object under study. The image acquisition process can be mainly divided in two sub-processes: the *image formation* and the *image recording* processes.

The image formation process is performed by an imaging system being able to convert a particular radiation (visible light, X-rays, microwaves or ultrasounds) into an electrical impulse which

can be measured by a recorder. Typical sources of the degradation occurring during the image formation are: scattering of light between the object and the imaging system or their sudden movements during the acquisition, light diffraction phenomenon limiting the resolution to features of the order of illuminating wavelength, lenses limited spatial extent and imperfections and, finally, the digitization since an average of the illumination over regions rather than a sampling at discrete points is performed.
The image recording process is performed by the recorder measuring the electrical impulse and placing the measurements on an array of pixels. The degradation introduced in this case, is usually referred to as *noise* which is mainly due to the following causes: the usage of finite exposure time introducing stochastic perturbations from the random arrival of photons, the optical imperfections, the aliasing of high-frequency during the sampling, the digitization causing quatization errors, and finally, any further compression action.
Therefore, the image recorded is not only a discrete representation of the original object but it is, generally, corrupted by these sources of degradation.
The task of *image restoration* (IR) aims at inverting the degradation process, thus estimating the unknown clean image being given the corrupted data.

1.3.1 Image restoration as inverse problem

In the following discussion we consider gray-level images, however everything could be extended to the multi-channel case. Furthermore, we assume AWGN as noise model since it will be the one mainly used in the next chapters.
We describe the whole image acquisition process through the following linear model:

$$\mathbf{b} = \mathbf{b}_0 + \mathbf{e}, \quad \text{with} \quad \mathbf{b}_0 = \mathbf{A}\mathbf{u}, \tag{1.7}$$

where the forward operator $\mathbf{A} \in \mathbb{R}^{m\times n}$ is a known linear operator describing the image formation process, $\mathbf{b} \in \mathbb{R}^m$ and $\mathbf{u} \in \mathbb{R}^n$ represent the vectorized degraded and the unknown clean images and $\mathbf{b}_0 \in \mathbb{R}^m$ is the noise-free observation, whereas, upon AWGN assumptions, $\mathbf{e} \in \mathbb{R}^m$ is sampled from a zero-mean Gaussian distribution of standard deviation $\sigma_{\mathbf{e}}$.
Therefore, the IR task reads as the following inverse problem:

$$\text{find} \quad \mathbf{u} \in \mathbb{R}^n, \quad \text{s.t} \quad \mathbf{b} = \mathbf{b}_0 + \mathbf{e} \quad \text{with} \quad \mathbf{b}_0 = \mathbf{A}\mathbf{u}. \tag{1.8}$$

We remark that, differently from the SI task (1.4), the sole unknown of the IR problem is the clean image. In real applications, the imaging system $\mathbf{A}$ is typically dependent on few parameters which can be thought as unknowns of the IR problem (*blind deconvolution*). However, in this thesis, we assume $\mathbf{A}$ is fully known or, at least, these parameters can be can be manually set.
Finally, it is well-known that the discrete inverse problem in (1.8) is ill-posed (see Chapter 2), making the IR task extremely challenging in real applications.

1.3.2 Forward operators

The definition of the imaging system $\mathbf{A}$ in (1.7) strictly depends on the IR task considered. In this thesis we focus on three important tasks arising in the field of image restoration, namely *image*

denoising, *image deblurring* and *image super resolution*. The acquisition models for all these tasks can be obtained from (1.7) by simply changing the forward linear operator $\mathbf{A}$.
In this section we briefly introduce the *blurring* and *downsampling* operators.

Blurring operator

In the continuous setting, the blurring operator is usually described by a Fredholm integral equation of the first kind:

$$b_0(x) = \int_\Omega k(x,y)u(y)dy, \quad \forall x \in \Omega, \tag{1.9}$$

where we assume the noise-free observation b_0 and the object u are defined on the same space-domain Ω, while $k(x,y)$ represents the kernel function which is referred to as *blur kernel* or *point spread function* (PSF).
More precisely, the action of k on an image consisting of a single point source would lead to a spread of the intensity around the point, that is the reason why it is referred to as PSF.
The PSF k is assumed to have a compact support [8] and to satisfy the following constraints:

$$\begin{aligned} k(x,y) &\geq 0 \quad \forall x,y \in \Omega, \\ \int_\Omega k(x,y)dy &= 1 \quad \forall x \in \Omega. \end{aligned}$$

For simplicity, we further consider the PSF to be space invariant, i.e., invariant with respect to translations, namely:

$$k(x,y) = k(x-y) \quad \forall x,y \in \Omega.$$

Hence, (1.9) takes the form of a convolutional product, that is:

$$b_0(x) = \int_\Omega k(x-y)u(y)dy = (k * u)(x) \quad \forall x \in \Omega. \tag{1.10}$$

As an example of a space invariant PSF, we mention the Gaussian blur kernel defined as:

$$k(x-y) = \frac{1}{\sigma_k\sqrt{2\pi}} \exp(-\frac{\|x-y\|_2^2}{2\sigma_k^2}),$$

where σ_k is the standard deviation of the Gaussian related to the width of the blur.
As an example, in Figure 1.6a we depict the clean `flower` image and in the upper left corner a Gaussian kernel, whereas in Figure 1.6b we report the blurred data.

Upon the assumption of a space invariant PSF, the deblurring task reads as the inverse problem (1.8) by setting the imaging system $\mathbf{A}$ as a discretization of the convolution in (1.10), e.g. obtained by applying standard quadrature formulas [2, 9]. We refer to this discrete operator as *blur matrix*, which is denoted by $\mathbf{H}$ in the following. We assume that the convolution is performed without stride, then $n = m$ in (1.8) and $\mathbf{H} \in \mathbb{R}^{n\times n}$. We finally remark that $\mathbf{H}$ is defined as an operator acting on vectorized images.

(a) (b)

Figure 1.6: (a) Gaussian kernel and the clean `flower` image. (b) blurred `flower` image. The Gaussian kernel has been obtained by using the Matlab function `fspecial`. The blurred image has been obtained by using the Matlab function `imfilter`.

The structure of $\mathbf{H}$ depends both on the definition of k and on the boundary conditions fixed. More precisely, as shown in Figure 1.6a, the blur kernel acts also on the boundary of the image, hence what is outside the acquisition domain Ω influences the blur inside Ω. The choice of boundary conditions depends on the image considered and has a drastic influence on the quality of the restored image. We now revise two particular cases of boundary conditions which will be exploited in the following chapters.

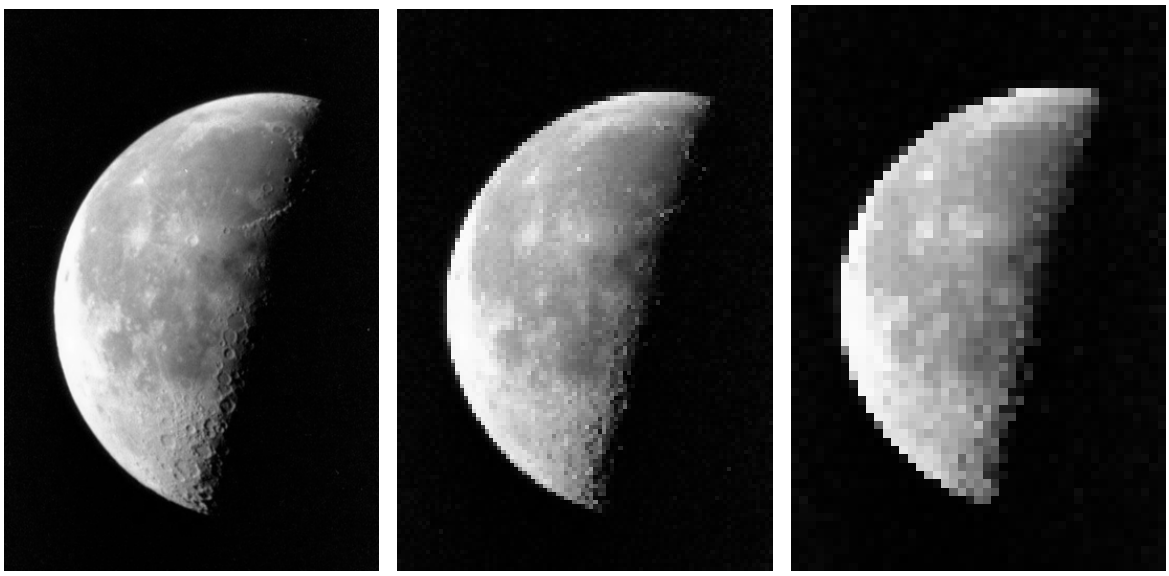
If the data have a dark background, it is natural to assume Dirichlet boundary conditions, namely to suppose the original image to be zero outside the acquisition domain. By adopting Dirichlet boundary conditions the arising matrix $\mathbf{H}$ has the structure of a block Toeplitz matrix with Toeplitz blocks [2, 9].
Periodic boundary conditions are among the most widespread choices. In this case, the blur matrix $\mathbf{H}$ is a block circulant matrix with circulant blocks (BCCB) [2, 9]. A nice property of BCCB matrices is that they can be diagonalized by the 2D Fourier transform $\mathbf{F} \in \mathbb{R}^{n\times n}$, hence:

$$\mathbf{H} = \mathbf{F}^H \mathbf{\Lambda} \mathbf{F}, \tag{1.11}$$

where $\mathbf{\Lambda} \in \mathbb{R}^{n\times n}$ is a diagonal matrix, $\mathbf{F}^H$ is the adjoint of $\mathbf{F}$.
The latter equality is an interesting result since that it allows to compute a blurred image without directly constructing $\mathbf{H}$ [2, 9]. Indeed, this can be cumbersome and computationally intractable expecially when dealing with large-scale images since the number of rows and columns of $\mathbf{H}$ equals the number of pixels of an image. Upon periodic boundary conditions, computing a blurred data, which is equivalent to the computation of matrix-vector products of the form $\mathbf{Hz}$ or $\mathbf{H}^H\mathbf{z}$ with $\mathbf{z} \in \mathbb{R}^n$, can be performed element-wise in the Fourier domain.

Downsampling operator

In some contexts there are also physical limitations in the acquisition process. The sensor is composed of a limited number of photodiodes whose density is directly proportional to the quality of the computed images. One might thinks that it could be possible to improve the quality of the data, keeping the same size of the sensor, by increasing the density of photodiodes.
However, little photodiodes get a smaller amount of light, thus worsening the quality of the acquired image. Furthermore, the greater the density of photodiodes the more expensive is the hardware. Differently from the deblurring task, the super resolution task takes into account this limitation and introduces it in the acquisition model.
In super resolution the system $\mathbf{A} \in \mathbb{R}^{m\times n}$ in (1.7) is usually defined as the product of a blur matrix $\mathbf{H} \in \mathbb{R}^{n\times n}$ and a *downsampling matrix* $\mathbf{S} \in \mathbb{R}^{m\times n}$.
Formally, given an integer $L > 1$ we define the downsampling matrix as the linear operator $\mathbf{S} \in \mathbb{R}^{m\times n}$ linking high resolution (HR) images to low resolution (LR) ones, where we denote by m and n the sizes of the LR and HR vectorized images, respectively, such that $n = L^2 m$. The integer L is usually referred to as *downsampling factor*.
In Figure 1.7 we report the action of a downsampling operator on an image of dimension 530×350 (Figure 1.7a) while changing $L = 4, 8$ (Figure 1.7b and 1.7c, respectively). The clear effect of the downsampling is the loss of details of an image.



(a) 530×350 (b) 135×90 (c) 68×45

Figure 1.7: Action of a downsampling operator while changing the downsampling factor. In (a) the original HR image. In (b) a LR image obtained setting $L = 4$. In (c) a LR image obtained setting $L = 8$.

There are plenty of downsampling operators. The first we mention is the *averaging operator*. Let L be the downsampling factor and $\mathbf{X} \in \mathbb{R}^{N_r\times N_c}$ an HR image whose vectorized counterpart is denoted by $\mathbf{x} \in \mathbb{R}^n$ where $n = N_r \cdot N_c$. The averaging operator averages pixels by patches of size

$L \times L$, so that it returns $\mathbf{y} \in \mathbb{R}^m$ with $m = n_r \cdot n_c$, which is the vectorized counterpart of the LR image $\mathbf{Y} \in \mathbb{R}^{n_r \times n_c}$ such that $N_r = L \cdot n_r$ and $N_c = L \cdot n_c$. In formula:

$$\mathbf{S}(\mathbf{x}) = \mathbf{y}, \tag{1.12}$$

where $\mathbf{Y} = \mathbf{M}\mathbf{X}\mathbf{N}^H$ and the matrices $\mathbf{M} \in \mathbb{R}^{n_r \times N_r}$ and $\mathbf{N} \in \mathbb{R}^{n_c \times N_c}$ read:

$$M = \begin{bmatrix} \mathbf{1}_L & \mathbf{0}_L & \cdots & \mathbf{0}_L \\ \mathbf{0}_L & \mathbf{1}_L & \cdots & \vdots \\ \vdots & & \ddots & \vdots \\ \mathbf{0}_L & \mathbf{0}_L & \cdots & \mathbf{1}_L \end{bmatrix}, \qquad N = \begin{bmatrix} \mathbf{1}_L & \mathbf{0}_L & \cdots & \mathbf{0}_L \\ \mathbf{0}_L & \mathbf{1}_L & \cdots & \vdots \\ \vdots & & \ddots & \vdots \\ \mathbf{0}_L & \mathbf{0}_L & \cdots & \mathbf{1}_L \end{bmatrix}, \tag{1.13}$$

where by $\mathbf{1}_L$, $\mathbf{0}_L \in \mathbb{R}^{1 \times L}$ we denote a vector of ones divided by L and a vector of zeros, respectively. Moreover, the adjoint $\mathbf{S}^H \in \mathbb{R}^{n \times m}$ of $\mathbf{S}$ reads:

$$\mathbf{S}^H(\mathbf{y}) = \mathbf{x}, \tag{1.14}$$

where by $\mathbf{x} \in \mathbb{R}^n$ we refer to the vectorized counterpart of $\mathbf{X} = \mathbf{M}^H\mathbf{Y}\mathbf{N}$.
Another widespread downsampling operator is the *decimation operator* which removes selected rows and columns. In Figure 1.8 we provide an example of the action of a decimation operator $\mathbf{S}$ on an HR image of size $n = 16$ ($N_r = 4$ and $N_c = 4$), by setting $L = 2$.
The adjoint operator $\mathbf{S}^H \in \mathbb{R}^{n \times m}$ interpolates the decimated image with zeros (see Figure 1.8).

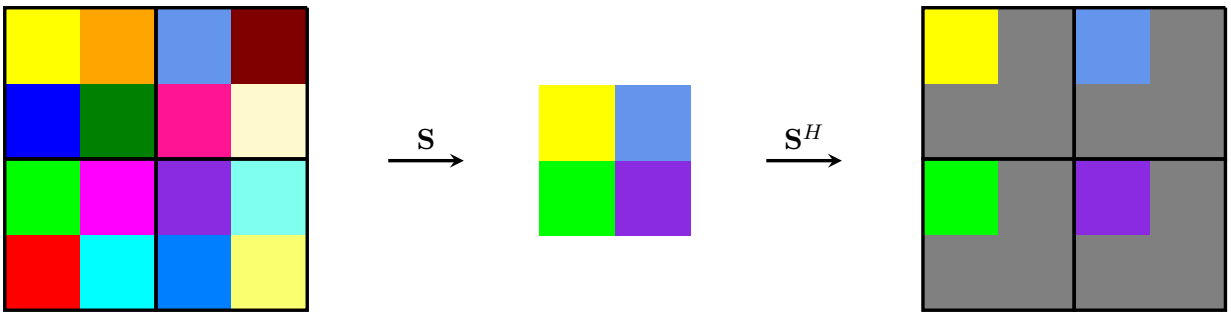


Figure 1.8: Action of the decimation operator $\mathbf{S}$ and its adjoint $\mathbf{S}^H$. $\mathbf{S}$ acts on a 4×4 image with $L = 2$, $\mathbf{S}^{\mathrm{H}}$ acts on a 2×2 image with $L = 2$.

In general, the downsampling operators are unstructured and, in particular, they cannot be diagonalized by the 2D discrete Fourier Transform, thus we cannot exploit the nice properties highlighted for the blurring operator in (1.11). As a consequence, the computation of matrix-vector products of the form $\mathbf{S}\mathbf{z}$ and $\mathbf{S}^H\mathbf{w}$ for $\mathbf{z} \in \mathbb{R}^n$ and $\mathbf{w} \in \mathbb{R}^m$ cannot be a-priori fastly computed.
Finally, we remark that the downsampling operators are not an injective function. For example, in Figure 1.9 we consider two chess-like images of dimension 4×4 and we apply the averaging operator setting $L = 2$. It is evident how these images have the same LR counterpart making the process of retrieving the HR image more challenging due to the lack of uniqueness.

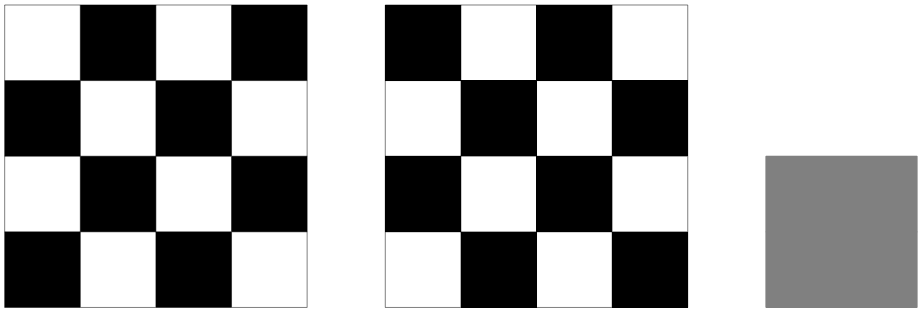


Figure 1.9: Two different chess-like 4×4 images and their avereged downsampling with $L = 2$.

Some examples of IR tasks: denoising, deblurring and super resolution

Image denoising. The denoising task aims at removing noise from the image preserving the main content of the data. The presence of noise is mainly due to atmospheric disturbances which can negatively affect the transmission of the signal, to the digitization process when photon counting of the camera sensor occurs or periodic fluctuations in the emitted electromagnetic radiation intensity. Hence, a clean image $\mathbf{U} \in \mathbb{R}^{N_r \times N_c}$ and a noisy image $\mathbf{B} \in \mathbb{R}^{N_r \times N_c}$ are related by the following acquisition model:

$$\mathbf{b} = \mathbf{u} + \mathbf{e}, \tag{1.15}$$

which is a particular instance of (1.7) by assuming $\mathbf{b}$ and $\mathbf{u}$ are the vectorizations of $\mathbf{B}$ and $\mathbf{U}$, respectively, where $n = m = N_r \cdot N_c$, and $\mathbf{A}$ is equal to the identity operator of $\mathbb{R}^n$.

Image deblurring. The deblurring task aims at providing sharper data by removing the blur which can be due to an intrinsic limit of the acquisition device, to the movement of the subject, to an improper holding of the device by the user, to dirty lens or wrong focusing. The acquisition model that relates a clean image $\mathbf{U} \in \mathbb{R}^{N_r \times N_c}$ and a noisy and blurred image $\mathbf{B} \in \mathbb{R}^{N_r \times N_c}$ reads:

$$\mathbf{b} = \mathbf{H}\mathbf{u} + \mathbf{e}, \tag{1.16}$$

which is a particular instance of (1.7) where we assume $\mathbf{A} = \mathbf{H} \in \mathbb{R}^{n \times n}$ which is the *blur matrix* described in the section 1.3.2. In the following we consider only blur matrices derived by the convolution with a zero-mean Gaussian kernels of standard deviation $\sigma_{\mathbf{H}}$.

Image super resolution. The task of image *super resolution* aims at enhancing the spatial resolution by representing on finer pixel grid the acquired low resolution data in order to highlight small but meaningful details. The spatial resolution, namely the pixel density of an image, is usually limited due to physical and cost constraints. For example in fluorescence microscopy the resolution is physically limited by the light diffraction limit, whereas in thermal imaging high resolution cameras require an extreme waste of money. Hence, a clean high resolution image $\mathbf{U} \in \mathbb{R}^{N_r \times N_c}$ and a noisy, blurred and low resolution image $\mathbf{B} \in \mathbb{R}^{n_r \times n_c}$ are related by the following acquisition model:

$$\mathbf{b} = \mathbf{S}\mathbf{H}\mathbf{u} + \mathbf{e}, \tag{1.17}$$

which is a particular instance of (1.7) by assuming $\mathbf{b} \in \mathbb{R}^m$ is the vectorization of the low resolution image $\mathbf{B}$ where $m = n_r \cdot n_c$, $\mathbf{u} \in \mathbb{R}^n$ is the vectorization of the unknown high resolution image $\mathbf{U}$ where $n = N_r \cdot N_c$ and $\mathbf{A} \in \mathbb{R}^{m \times n}$ is the product of a blur matrix $\mathbf{H} \in \mathbb{R}^{n \times n}$ and the downsampling matrix $\mathbf{S} \in \mathbb{R}^{m \times n}$ described previously.

In Figure 1.10 we provide a schematic representation of the acquisition models of these three tasks for the `cat` image. For all these tasks we aim at estimating the unknown clean image by using the sole degraded acquisition and assuming the imaging system is known.

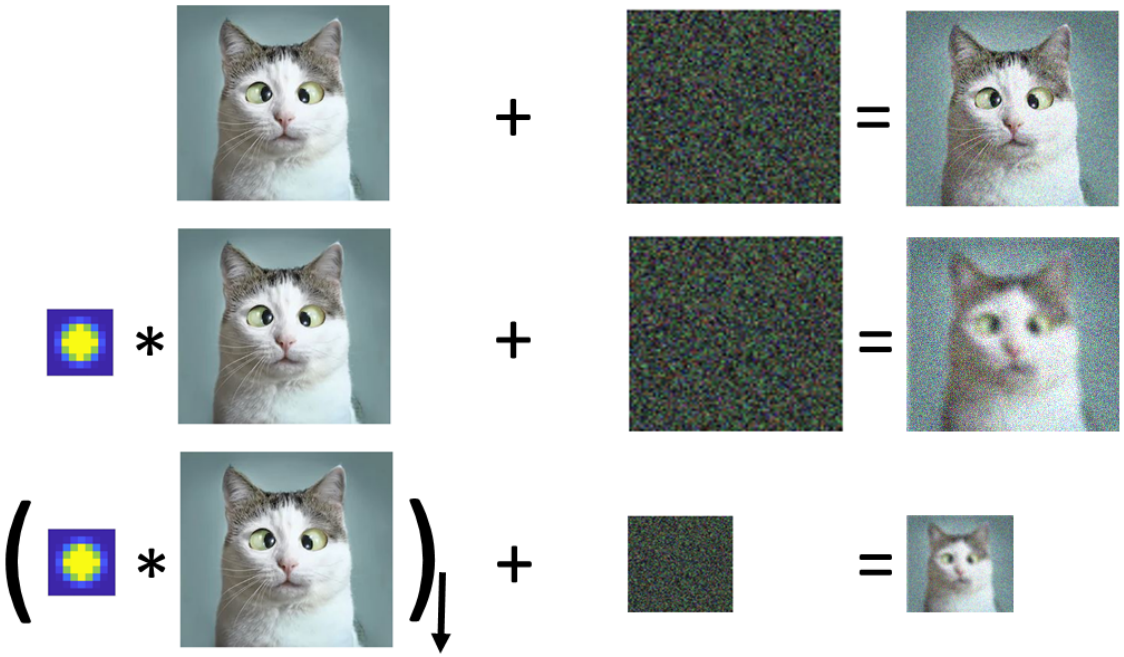


Figure 1.10: Upper panel: sketch of the image denoising acquisition model for the `cat` image. Middle panel: sketch of the image deblurring acquisition model for `cat` image. Lower panel: sketch of the super resolution acquisition model for the `cat` image. The convolution is performed with a Gaussian kernel, the black arrow denote the action of a downsampling operator. The aim of the these three tasks is to retrieve the clean `cat` image from the given degraded data.

Chapter 2

To ill-posedness and beyond

In Chapter 1 we have introduced the inverse problems of interest, namely system identification and image restoration. In this chapter, based on linear algebra considerations, we first motivate why they are both referred to as ill-posed problems. Then, we show how to provide reliable estimates of their solutions through *variational* and *learning* approaches which can be harmoniously merged in unified hybrid frameworks so as to overcome their respective limits.

2.1 Ill-posedness and regularization

Upon linearity assumptions, both the SI task (1.4) and the IR task (1.8) require the solution of a linear system:

$$\mathbf{y} = \mathbf{K}\mathbf{x}, \tag{2.1}$$

where $\mathbf{K} \in \mathbb{R}^{m \times n}$ is the matrix defining the linear system, $\mathbf{x} \in \mathbb{R}^n$ is the vector of unknowns and $\mathbf{y} \in \mathbb{R}^m$ represent the data.
Depending on the task, the matrix $\mathbf{K}$ could be either a tall, wide or square matrix, that is $m > n$, $m < n$ or $m = n$, respectively. As an example, in the SI task we usually have $m > n$ since the number of observations is higher than the number of parameters, hence in this case the matrix defining the linear system is tall. On the contrary, if we consider the super-resolution or deblurring tasks, as we have seen, the matrix is wide or square, respectively.

Let us consider the singular value decomposition (SVD) of $\mathbf{K}$:

$$\mathbf{K} = \mathbf{W}\mathbf{\Sigma}\mathbf{V}^T, \tag{2.2}$$

where $\mathbf{W} \in \mathbb{R}^{m \times m}$ and $\mathbf{V} \in \mathbb{R}^{n \times n}$ are orthonormal matrices, and $\mathbf{\Sigma} \in \mathbb{R}^{m \times n}$ reads:

$$\Sigma = \begin{pmatrix} \sigma_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_n \end{pmatrix} \quad \text{if } m = n, \quad \Sigma = \begin{pmatrix} \sigma_1 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & \sigma_m & 0 & \cdots & 0 \end{pmatrix} \quad \text{if } m < n,$$

$$\Sigma = \begin{pmatrix} \sigma_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_n \\ 0 & \cdots & 0 \\ \vdots & \cdots & \vdots \\ 0 & \cdots & 0 \end{pmatrix} \quad \text{if } m > n. \tag{2.3}$$

The scalars $\sigma_1 \geq \sigma_2 \geq \ldots \sigma_{\min(m,n)} \geq 0$ denote the singular values of $\mathbf{K}$. When $\mathbf{K}$ is not square, and hence not invertible, we can still consider its pseudo-inverse matrix $\mathbf{K}^+$ defined through its SVD decomposition:

$$\mathbf{K}^+ = \mathbf{W}\mathbf{\Sigma}^+\mathbf{V}^T,$$

where $\mathbf{\Sigma}^+$ is formed from $\mathbf{\Sigma}$ by taking the reciprocal of all the singular values.
Since $\mathbf{K}$ is not invertible, one may thus think to solve the linear system (2.1) by considering the following "pseudo-inverse solution":

$$\tilde{\mathbf{x}} = \mathbf{K}^+\mathbf{y} = \mathbf{V}\mathbf{\Sigma}^+\mathbf{W}^T\mathbf{y} = \sum_{i=1}^{r} \frac{\mathbf{w}_i^T\mathbf{y}}{\sigma_i}\mathbf{v}_i, \tag{2.4}$$

with $\mathbf{w}_i$, $\mathbf{v}_i$ denoting the i-th column of $\mathbf{W}$ and $\mathbf{V}$, respectively, and $r := \text{rank}(\mathbf{K})$.

Typically, the matrix $\mathbf{K}$ presents very small singular values and, in general, the data $\mathbf{y}$ is affected by small perturbations. As an example, when dealing with image deblurring, in [9] it has been observed that, in the continuous domain, the blur kernel (a compact operator) does not have a bounded inverse which gets reflected in an high condition number of its finite dimensional approximation. Therefore, by (2.4) we deduce that these small singular values lead to an amplification of the perturbations in the data. This somehow suggests an explanation to the sensitiveness of inverse problems solution to measurement errors, i.e. small errors in the measurements induce large errors in the resulting estimated solution.

We remark that when $m \leq n$ the linear system (2.1) is underdetermined, therefore it has infinite solutions. In particular $\tilde{\mathbf{x}}$ solves the linear system (2.1) and is the solution with the minimum norm. In formula:

$$\tilde{\mathbf{x}} = \underset{\mathbf{x}\in\mathbb{R}^n}{\arg\min} \|\mathbf{x}\|_2^2 \quad \text{s.t.} \quad \mathbf{K}\mathbf{x} = \mathbf{y}. \tag{2.5}$$

If $m > n$ the linear system is ovedetermined with, in general, no solutions. However, a least-square solution can still be provided. If the matrix $\mathbf{K}$ is full column rank, then $\tilde{\mathbf{x}}$ equals the least-square solution. In formula:

$$\tilde{\mathbf{x}} = \underset{\mathbf{x}\in\mathbb{R}^n}{\arg\min} \|\mathbf{K}\mathbf{x} - \mathbf{y}\|_2^2. \tag{2.6}$$

The instability to noise has stimulated the researchers to work both on the theory of inverse problems and on the development of efficient strategies to provide approximate (and reliable) solutions so as to overcome the ill-posedness. There exist several strategies preventing the amplification

of the noisy perturbations. A common approach in the theory of inverse problems is *regularization*, which is referred to as the process of replacing the ill-posed inverse problem of reference with an approximate stable and well-posed one, whose solution exists and continuously depends on the observation.
Among the possible regularization methods we mention the truncated SVD approach, which consists in cutting the smallest $r-k$ singular values, with $k \in \mathbb{N}$ properly fixed such that $k < r$, and truncate the sum in (2.4), so as to prevent noise amplification. In formula:

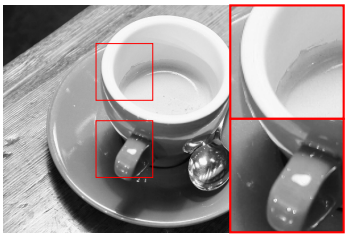
$$\tilde{\mathbf{x}}_k = \sum_{i=1}^{k} \frac{\mathbf{w}_i^T \mathbf{y}}{\sigma_i} \mathbf{v}_i.$$

An equivalent strategy considers an iterative algorithm solving the least-square optimization problem, as the one in (2.6). To avoid the noise amplification, the idea is to early stop the iterative process solving (2.6) by fixing a maximum number of iterations (itr). As an example, in Figure 2.1a we report the `coffee` image and in Figure 2.1b we report its blurred and noisy simulated acquisition obtained by applying to the original image the model (1.16) assuming AWGN with standard deviation $\sigma_{\mathbf{e}} = 0.01$ and setting $\sigma_{\mathbf{H}} = 1.2$. In Figure 2.1f we plot the PSNR behaviour as function of the iterations of a gradient-descent method solving the least-square problem. We can observe the PSNR first increases then it starts worsening, reaching the maximum at itr= 4. In Figures 2.1c, 2.1d and 2.1e we depict the iterates at itr= $2, 4, 11$, respectively. As a general comment, the solution at itr= 2 looks too smooth, whereas in the one computed at itr= 11 the noise is amplified. The best solution both in terms of PSNR and visual inspection is reached at itr= 4.
For further details concerning these methods we refer the interested reader to the books [2, 9, 10].

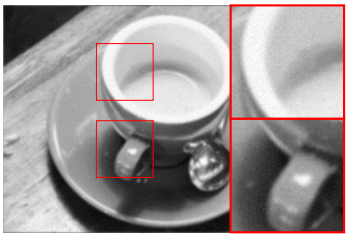
The most efficient regularization strategies, are *variational* and *learning* methods. So far, a plethora of literature has been devoted to the development of such strategies [11, 12, 13, 14, 15]. The common idea of variational and learning approaches is to develop a reconstruction map, whose definition depends on the inverse problem considered, to recover the unknowns from the data. According to the learning framework the reconstruction map can be interpreted as a parametric function whose parameters are identified by solving a minimization problem, whereas in the variational approach the outcome of the reconstruction function is a minimum of a penalized minimization problem. The challenge of variational methods is the design of specific regularized objectives reflecting prior knowledge on the solutions and tailored on the task considered. The learning formulation is attractive because it overcomes these limitations. However, it requires a training set whose collection can be costly, and moreover, the minimization problem to be solved is typically more difficult than the one considered by variational methods. In this thesis we exploit both variational and learning models to solve the IR task (1.8), whereas we consider the sole learning approach for the SI task (1.4).

2.2 Variational models

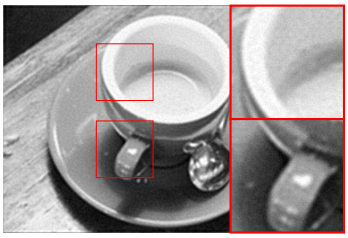
The variational approach for the IR task (1.8) defines the reconstruction map $\mathcal{E}_{var} : \mathbb{R}^m \to \mathbb{R}^n$, such that computing $\mathcal{E}_{var}(\mathbf{b})$ is equivalent to solve the following unconstrained and constrained minimization problems:



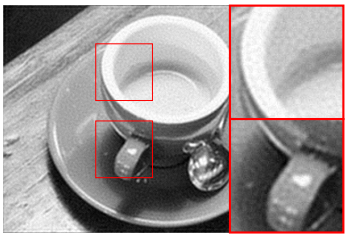
(a) original



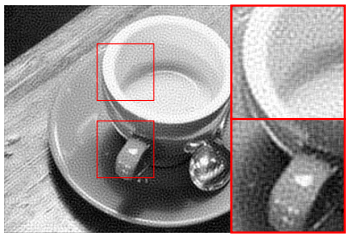
(b) degraded



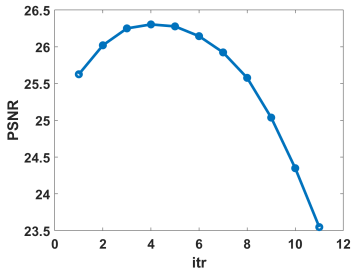
(c) itr=2



(d) itr=4



(e) itr=11



(f) PSNR plot

Figure 2.1: (a) original `coffee` image. (b) blurred and noisy `coffee` image obtained by applying the model (1.16) setting $\sigma_{\mathbf{H}} = 1.2$ and considering AWGN with $\sigma_{\mathbf{e}} = 0.01$. (c)-(d)-(e) solutions at iterations 2,4 and 11, respectively. (f) PSNR behaviour as functions of the iterations of the gradient-descent algorithm solving the least-square problem.

$$\mathcal{E}_{var}(\mathbf{b}) := \begin{cases} \mathbf{u}^* \in \underset{\mathbf{u}\in\mathbb{R}^n}{\arg\min}\ D(\mathbf{u},\mathbf{A},\mathbf{b}) + \lambda R(\mathbf{u}), & (2.7) \\ \mathbf{u}^* \in \underset{\mathbf{u}\in\mathbb{R}^n}{\arg\min}\ D(\mathbf{u},\mathbf{A},\mathbf{b}) \quad \text{s.t.} \quad R(\mathbf{u}) \leq c_R, & (2.8) \\ \mathbf{u}^* \in \underset{\mathbf{u}\in\mathbb{R}^n}{\arg\min}\ R(\mathbf{u}) \quad \text{s.t.} \quad D(\mathbf{u},\mathbf{A},\mathbf{b}) \leq c_D. & (2.9) \end{cases}$$

where $\mathbf{u}^* \in \mathbb{R}^n$ is an estimate of the real solution $\mathbf{u} \in \mathbb{R}^n$, the positive scalar λ is referred to as *regularization parameter* and balances the contribution of the two functionals D and R which are usually called *fidelity term* and *regularization term*, respectively. The functional D measures the discrepancy between the measurement $\mathbf{b}$ and the noise free observation $\mathbf{Au}$. The regularization term R encodes prior knowledge on the solution by penalizing undesired properties. The positive scalars c_R and c_D represent the strength of the constraints. Differently from the parameter λ, they usually have a physical meaning. For example, the value of c_D in (2.9) usually depends on the

amount of noise, whereas the value of c_R in (2.8) can reflect the number of non-zero pixel or the maximum value of the norm of the desired image.
The grand challenges of variational approaches are: (i) the definition of the fidelity term D depending on the assumptions on the noise, (ii) the design of regularizers R capturing the complexity of the image statistics and (iii) the proper setup of the parameter λ, c_D and c_R.

We now briefly review the interesting link between a Bayesian framework [16] and the variational model (2.7) which helps us to provide a rigorous framework to derive the data fidelity and the regularization terms from probability density functions (pdfs).
In the Bayesian framework we interpret data and unknowns of (1.8) as random variables, although in this part we use the same notation. According to the Bayes' theorem, given the observation $\mathbf{b}$, the posterior probability $p(\mathbf{u}|\mathbf{b}, \mathbf{A})$ is defined as the ratio of the product between the data likelihood $p(\mathbf{b}|\mathbf{u}, \mathbf{A})$ and the prior $p(\mathbf{u})$, and the evidence term $p(\mathbf{b}, \mathbf{A})$. In formula:

$$p(\mathbf{u}|\mathbf{b}, \mathbf{A}) = \frac{p(\mathbf{b}|\mathbf{u}, \mathbf{A})p(\mathbf{u})}{p(\mathbf{b}, \mathbf{A})}. \tag{2.10}$$

The data likelihood $p(\mathbf{b}|\mathbf{u}, \mathbf{A})$ defines a probability density on $\mathbf{b}$ given $\mathbf{u}$ and $\mathbf{A}$. Then, by considering the forward model (1.7), it corresponds to the pdf of the noise:

$$p(\mathbf{b}|\mathbf{u}, \mathbf{A}) = p(\mathbf{b} - \mathbf{A}\mathbf{b}) = p(\mathbf{e}).$$

Finally, the prior term $p(\mathbf{u})$ defines a distribution on the unknown of the problem encoding all the knowledge about the desired solution.
Therefore, the posterior distribution $p(\mathbf{u}|\mathbf{b}, \mathbf{A})$ encodes both information about the acquisition model (1.7) and prior assumptions on the solution. To extract meaningful points from the posterior probability distribution a popular choice is to use the mode of the posterior, namely the solution that maximizes the posterior probability, which is known as the maximum a-posteriori (MAP) estimator. In formula:

$$\mathbf{u}_{\text{MAP}} \in \underset{\mathbf{u}\in\mathbb{R}^n}{\arg\max}\ p(\mathbf{u}|\mathbf{b}, \mathbf{A}). \tag{2.11}$$

By definition (2.10) and by neglecting constant terms and taking the negative logarithm, (2.11) is equivalent to the following minimization problem:

$$\mathbf{u}_{\text{MAP}} \in \underset{\mathbf{u}\in\mathbb{R}^n}{\arg\min}\ -\log\left(p(\mathbf{b}|\mathbf{u}, \mathbf{A})\right) - \log\left(p(\mathbf{u})\right). \tag{2.12}$$

By comparing (2.12) and the variational model in (2.7), we deduce that:

$$D(\mathbf{u}, \mathbf{A}, \mathbf{b}) := -\log\left(p(\mathbf{b}|\mathbf{u}, \mathbf{A})\right), \quad \lambda R(\mathbf{u}) := -\log\left(p(\mathbf{u})\right). \tag{2.13}$$

Thus, the fidelity encodes information about the acquisition model whereas the regularizer allows to incorporate statistical knowledge about the solution.

Fidelity term

In (1.7), due to the AWGN assumptions with zero-mean and variance $\sigma_{\mathbf{e}}^2$, the likelihood reads:

$$p(\mathbf{b}|\mathbf{u}, \mathbf{A}) = \frac{1}{2\pi\sigma_{\mathbf{e}}^2}\exp\left(-\frac{1}{2\sigma_{\mathbf{e}}^2}\|\mathbf{A}\mathbf{u} - \mathbf{b}\|_2^2\right).$$

By considering the negative logarithm and neglecting the constant terms we get the corresponding ℓ_2-norm based data fidelity term:

$$D(\mathbf{u}, \mathbf{A}, \mathbf{b}) = \frac{1}{2\sigma_{\mathbf{e}}^2}\|\mathbf{Au} - \mathbf{b}\|_2^2. \tag{2.14}$$

Upon Poisson noise model assumptions, the corresponding fidelity equals the Kullback Leibler (KL) divergence, namely:

$$D(\mathbf{u}, \mathbf{A}, \mathbf{b}) = \sum_{i=1}^{m} \left(\mathbf{b}_i \ln \left(\frac{\mathbf{b}_i}{(\mathbf{Au})_i} \right) + (\mathbf{Au})_i - \mathbf{b}_i \right).$$

In conclusion, the likelihood and the corresponding data fidelity term can be often directly deduced from the problem at hand, whereas defining the prior, and consequently the regularizer, requires knowledge about the structure of the solution thus being a more challenging problem.

Handcrafted regularization terms

Among the first regularizers proposed in the literature we mention the handcrafted Tikhonov regularization. In his pioneering work [17] Tikhonov defines the following ℓ_2-norm based regularizer:

$$R(\mathbf{u}) = \|\mathbf{Lu}\|_2^2, \tag{2.15}$$

where $\mathbf{L}$ equals the identity operator (zero-order Tikhonov regularization). Later, (2.15) was extended to both a gradient- and a Laplacian-based ℓ_2-norm terms (first-order and second-order Tikhonov regularization) by considering $\mathbf{L}$ equal to the discrete gradient and to the discrete Laplacian operator, respectively.
In Bayesian terms, these regularizers come from a zero-mean white Gaussian prior with variance γ^2, namely:

$$p(\mathbf{u}) \sim \exp\left(-\frac{1}{2\gamma^2} \sum_{i}^{n} \|(\mathbf{Lu})_i\|_2^2 \right), \tag{2.16}$$

thus according to (2.13) the regularization parameter is somehow related to the inverse of γ^2. The Tikhonov regularizers are convex and differentiable, hence, for example, upon Gaussian noise assumptions on $\mathbf{b}$, by replacing (2.14) and (2.15) in (2.7), the arising objective is convex and can be minimized through standard gradient-based algorithms [18]. Moreover, we remark that, in this case, the solution of the variational model is unique if the intersection of the null space of both operators $\mathbf{A}$ and $\mathbf{L}$ only includes the zero vector, i.e. $\mathbf{Ker}(\mathbf{A}) \cap \mathbf{Ker}(\mathbf{L}) = \{\mathbf{0}\}$.
However, Tikhonov-based regularization is not effective since it favors dark images (when $\mathbf{L}$ is fixed as the identity operator) or smooth images with unsharp edges (when $\mathbf{L}$ is equal to either the discrete gradient or Laplacian) while the objects to be restored usually manifest sharp features.

Thus, it is essential to define regularizers tailored to encode information about the transitions of color intensities in different areas of the image. These regularizers should reflect some properties we expect the discrete gradient of the unknown image has, in order to preserve discontinuities and edges. It is well-known that natural images often admit very sparse approximations in the gradient domain and the theory of *compressed sensing* [19] provides many gradient sparsity-promoting

regularizers. Among them, we mention the Total Variation (TV) regularizer, introduced by Rudin, Osher, and Fatemi in their seminal paper [20], which has been the most investigated regularizer and the landmark for the development of modern regularization techniques in imaging [13].
Given a vectorized image $\mathbf{u} \in \mathbb{R}^n$, the TV regularizer is defined as follows:

$$\mathrm{TV}(\mathbf{u}) := \|\mathbf{D}\mathbf{u}\|_1 = \sum_{i=1}^{n} \left(|(\mathbf{D_h}\mathbf{u})_i|^2 + |(\mathbf{D_v}\mathbf{u})_i|^2 \right)^{1/2}, \tag{2.17}$$

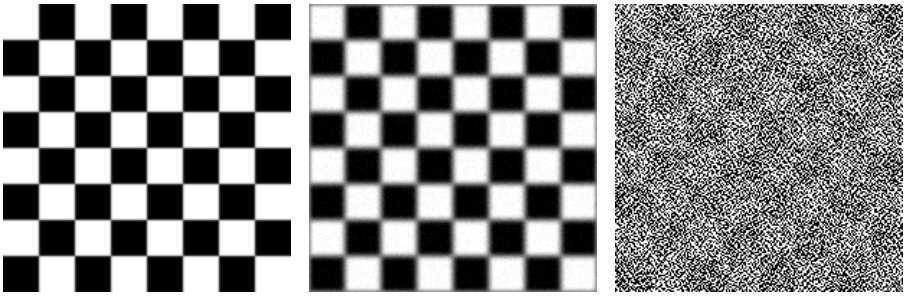
where by $\mathbf{D} = (\mathbf{D}_h; \mathbf{D}_v) \in \mathbb{R}^{2n \times n}$ we denote the discrete gradient such that $\mathbf{D}_h \in \mathbb{R}^n$, $\mathbf{D}_v \in \mathbb{R}^n$ are the first order finite difference discrete operators along the horizontal and vertical axes, respectively. The TV regularizer is still convex but non-differentiable. However, this is not a limitation since many optimization algorithms for non-smooth functionals can be still exploited [18].
We stress that TV can be derived from a prior distribution. Indeed, it is natural to consider the unknown image as a Markov random field (MRF), namely a particular property of the image at the i-th pixel depends on the neighboring pixels. The prior distribution for a MRF is the so-called Gibbs prior, and the TV functional is derived as the negative logarithm of the Laplace distribution pdf, which reads as a particular instance of Gibbs priors.

In Figure 2.2a we report the original `checkboard` image and in Figure 2.2b we depict a simulated corrupted noisy and blurred counterpart. In order to highlight the importance of the selection of a proper regularizer, we depict in Figure 2.2c the solution of the standard least-square problem, that is we solve (2.7) for a deblurring task assuming no prior is available ($\lambda = 0$). In Figure 2.2d and 2.2e we compare the first order Tikhonov with the TV restoration. The former struggles to retrieve sharp edges and to efficiently promote piecewise constant patches, whereas the latter provides a largely better solution.

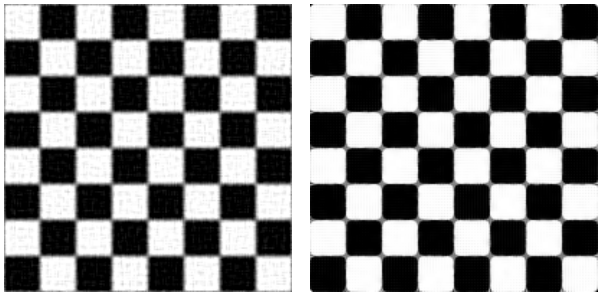
Even if, the TV-based regularizer has clear mathematical properties and can be interpreted in a Bayesian framework, due to its formulation, it struggles to reflect the complexity of image statistics. For example in Figure 2.3a we consider the degraded `skyscraper` image and we report the TV restoration Figure 2.3b. The restored image is denoised and sharp edges are preserved, however, homogeneous regions of the image such as the sky are not well restored and looks patchy. This issue is known in the literature as staircasing phenomenon and originates from the fact that TV favors piecewise constant solutions.
The TV regularizer has been extended in various ways by considering, for example, higher-order derivatives [21, 22] to overcome staircasing. Moreover, in order to induce better sparsity on the gradient domain the ℓ_1-norm in (2.17) has been replaced with non-convex ℓ_p-norms with $0 < p < 1$ [23, 24] or with the non-convex non-continuous ℓ_0 pseudo norm [25].

Beyond handcrafted regularization terms

Nowadays, it has been recognized that among the possible strategies to model efficient regularizers, learning from data represents the most efficient alternative. The fast development of learning-based techniques in the field of imaging inverse problems has allowed the definition of data-driven regularizers which have largely outperformed the handcrafted ones. Thus, variational and learning strategies are no more perceived as separate methods but can be easily combined in an hybrid



(a) original (b) degraded (c) least-square solution

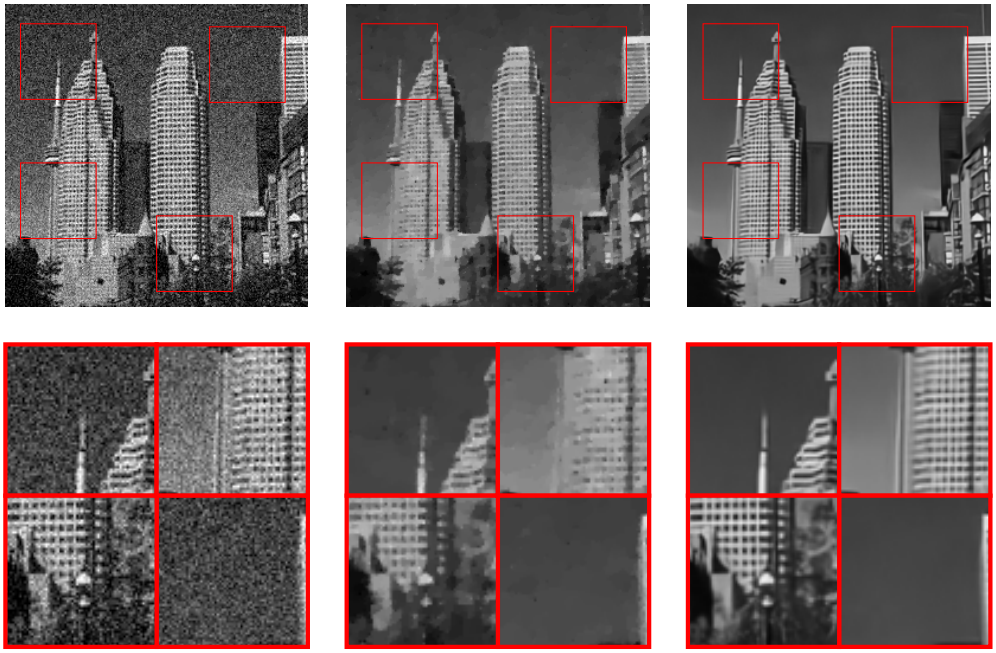


(d) Tikhonov restoration (e) TV restoration

Figure 2.2: Comparing Tikhonov and TV restorations. (a) original `checkboard` image. (b) blurred and noisy simulated acquisition obtained by applying (1.16) setting $\sigma_{\mathbf{H}} = 1$ and AWGN with $\sigma_{\mathbf{e}} = 0.01$. (c) least-square solution. (d) First order Tikhonov restoration ($\lambda = 0.05$). (e) TV restoration ($\lambda = 0.01$).

framework exploiting the merits of both. Given a dataset $\{(\mathbf{u}_i, \mathbf{b}_i)\}_{i=1\ldots N}$ of clean-degraded images, the core idea of these hybrid methods is to build flexible high-parametrized regularizers $R_{\boldsymbol{\theta}}$ whose parameters $\boldsymbol{\theta}$ are learnt by exploiting the considered dataset, e.g. in [26] or more recently in [27]. Once a proper set of parameters $\boldsymbol{\theta}^*$ has been identified, a possible strategy is to replace R with $R_{\boldsymbol{\theta}^*}$ in (2.7) and to apply standard optimization techniques to find the estimated solution. For a complete review of these strategies the reader can refer to [12, 28].

Among these hybrid regularization techniques in this thesis we are mostly interested in the so-called Plug-and-Play (PnP) regularizers [29] and their later development known as Regularization by Denoising (RED) functional [30]. The idea behind PnP is to exploit the modular structure of standard iterative proximal algorithms, such as the Alternating Direction Method of Multipliers (ADMM) or the Half-Quadratic Splitting (HQS) [31], solving the variational model in (2.7) when considering a generic regularizer R. The arising iterative scheme requires the solution of two or more proximal sub-problems related to the fidelity and to the regularization terms. Then, according to the PnP framework, the proximal mapping of the generic regularizer R is replaced with existing denoising algorithms like non-local means [32] and BM3D [33] which have been found capable to induce effective priors on the solution. Very recently, in [34, 35, 36, 37] also deep learning-based denoising algorithms have been used in the PnP framework. To show the effectiveness of such approach if compared to TV restoration, in Figure 2.3c we report the restored image provided by applying the PnP method [34] to the degraded `skyscraper` image depicted in Figure 2.3a. How-



(a) noisy and blurred (b) TV estimation (c) PnP estimation

Figure 2.3: Comparing between TV and PnP restorations. The starting image is obtained corrupting the original image according to the model (1.16) setting $\sigma_{\mathbf{H}} = 0.5$ and AWGN with $\sigma_{\mathbf{e}} = 0.1$.

ever, as we will see in Chapter 5, only upon restrictive assumptions, the PnP iterative scheme comes from a variational model.
In [30] the authors have introduced the RED regularizer which exploits the regularization by denoising principle, i.e. the capability of denoisers to induce regularization on the solution. Differently from PnP, the RED framework solves the variational model (2.7) by setting the regularizer as:

$$R(\mathbf{u}) = \frac{1}{2}\mathbf{u}^T(\mathbf{u} - \mathsf{D}(\mathbf{u})), \tag{2.18}$$

where $\mathsf{D}(\cdot)$ can be chosen as any off-the-shelf denoiser. By assuming the differentiability, local homogeneity, Jacobian symmetry and filter passivity of $\mathsf{D}(\cdot)$, in [30] the authors prove that R is convex and $\nabla R(\mathbf{u}) = \mathbf{u} - \mathsf{D}(\mathbf{u})$. However, although the two approaches exploit the same principle the link between RED and PnP, is not still well-understood.

2.3 Learning models

In the previous section, we have introduced the variational framework considering a reconstruction map $\mathcal{E}_{var}$ defined through a regularized minimization problem, namely the approximate solution of an inverse problem is seen as a minimizer of a particular energy functional encoding both prior knowledge on the solution and information about the acquisition process.
The availability of large datasets with enough samples to approximate real-world data distributions, the incredible boost in computer technology providing GPUs and their increased memory, have

paved the way to the recent success of learning. In the following sections we show how pure learning models can be used to solve both the SI and IR tasks.

Learning models for system identification

Let $\mathcal{D} := \{(\mathbf{u}^j, \mathbf{b}^j)\}_{j=1\dots N}$ be a set of input-output signals describing an unknown real system $\mathbf{A}$. The learning approach for the SI task (1.4) defines the reconstruction map $\mathcal{E}^{\text{SI}}_{learn} : \mathbb{R}^n \to \mathbb{R}^m$ such that:

$$\mathcal{E}^{\text{SI}}_{learn} := \mathbf{T}(\cdot, \boldsymbol{\theta}^*) \in \underset{\mathbf{T}_{\boldsymbol{\theta}}, \boldsymbol{\theta}\in\Theta}{\arg\min}\ \mathcal{L}(\boldsymbol{\theta}, \mathcal{D}, \mathbf{T}_{\boldsymbol{\theta}}) + \lambda\mathcal{R}(\boldsymbol{\theta}), \tag{2.19}$$

where $\mathcal{L}$ is a measure of error, $\mathcal{R}$ is a regularizer on the parameters which is considered to avoid overfitting and $\boldsymbol{\theta}^*$ is the learnt set of parameters such that $\mathcal{E}^{\text{SI}}_{learn} = \mathbf{T}(\cdot, \boldsymbol{\theta}^*)$ is an estimate of $\mathbf{A}$. Once the learning step is complete, $\mathcal{E}^{\text{SI}}_{learn}$ can be directly used to predict the output of the system given new unseen (not in the dataset) inputs.

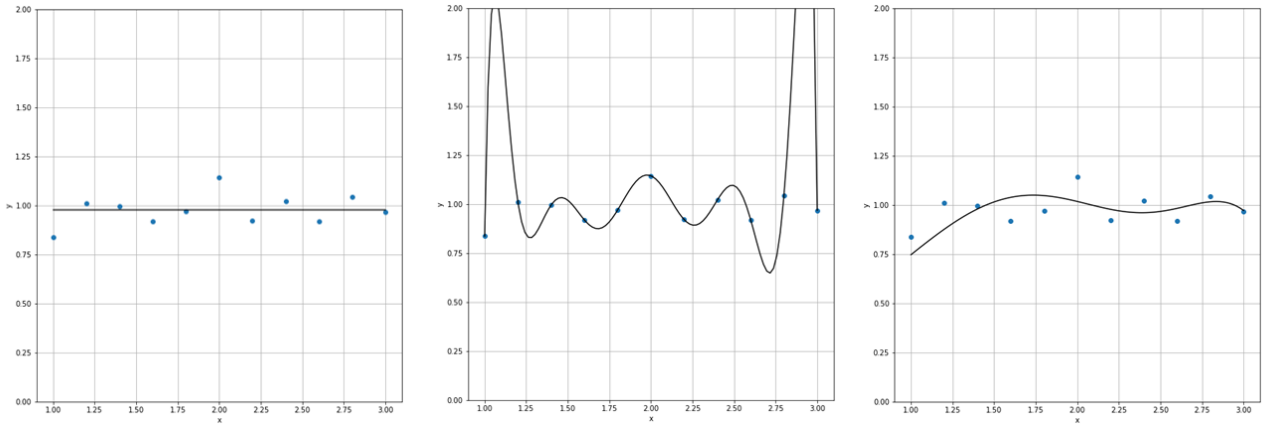
Understanding the learning minimization problem (2.19) as a statistical inference can provide useful insights into the selection of $\mathcal{L}$ and $\mathcal{R}$. Let us define $\mathcal{B} := \{\mathbf{b}^1, \dots, \mathbf{b}^N\}$ and $\mathcal{U} := \{\mathbf{u}^1, \dots, \mathbf{u}^N\}$. By assuming that the N example pairs $(\mathbf{b}^1, \mathbf{u}^1), \dots (\mathbf{b}^N, \mathbf{u}^N)$ are i.i.d., the likelihood distribution reads:

$$p(\mathcal{B}|\mathcal{U}, \boldsymbol{\theta}) = \prod_{j=1}^{N} p(\mathbf{b}^j|\mathbf{u}^j, \boldsymbol{\theta}). \tag{2.20}$$

We remark that in this case we have a family of pdfs parametrized by $\boldsymbol{\theta}$. The i.i.d. assumption entails that each pdf in the product (2.20) has the same distribution and shares the same parameters. We can extract useful points from the likelihood distribution defined in (2.20) by considering the maximum likelihood estimator (MAE). Hence, by neglecting constant terms the functional $\mathcal{L}$ can be defined as the negative log-likelihood of (2.20):

$$\mathcal{L}(\boldsymbol{\theta}, \mathcal{D}, \mathbf{T}_{\boldsymbol{\theta}}) = -\log\, p(\mathcal{B}|\mathcal{U}, \boldsymbol{\theta}) = -\sum_{j=1}^{N} \log(p(\mathbf{b}^j|\mathbf{u}^j, \boldsymbol{\theta})) = \frac{1}{2\sigma^2}\sum_{j=1}^{N}(\mathbf{b}^j - \mathbf{T}(\mathbf{u}^j, \boldsymbol{\theta}))^2, \tag{2.21}$$

where the last equality follows assuming $\mathbf{b}^j$ is affected by the sole Gaussian noise of variance σ^2.
We stress that in this field prior knowledge on the solution $\mathbf{A}$ is implicitly provided when assuming the system belongs to the parametric space of operators T_Θ and when considering the set of examples $\mathcal{D}$ for the parameter identification.
However, in Figure 2.4 we show that in real applications the MAE is prone to overfit. We consider a distribution of points $\{(x^j, y^j)\}_{j=1\dots 11} \subset \mathbb{R}^2$ which are depicted in Figure 2.4a (blue dots). We assume the dependence between the variables is described by a polynomial model of degree M. Namely, we assume $\mathbf{A} \in T_\Theta := \{\mathbf{T}(x, \boldsymbol{\theta}) = \theta_0 + \theta_1 x + \dots + \theta_M x^M, \forall x \in \mathbb{R}|\boldsymbol{\theta} \in \mathbb{R}^{M+1}\}$. Upon these assumptions, we solve the optimization problem (2.19) where $\mathcal{L}$ is fixed as in (2.21) and $\lambda = 0$. In Figure 2.4a we report the model (black line) obtained setting $M = 1$, whereas in Figure 2.4b we depict the model obtained considering $M = 7$. In the latter case, we can observe the model interpolates the points (overfitting), whereas we remark that the goal of SI is to achieve good



(a) M=1 (b) M=7 (c) M=7 with Gaussian prior

Figure 2.4: Comparisons between MAE and MAP estimation for SI task. (a) MAE when considering a family of polynomials of degree $M = 1$. (b) MAE when considering a family of polynomials of degree $M = 7$. (c) MAP when considering a family of polynomials of degree $M = 7$ and a Gaussian prior on the parameters of the model.

generalization by making accurate predictions for new (unseen) data.
By analyzing the computed parameters, we observe that their magnitudes are too high. We can further define a prior on the distribution of these parameters $\boldsymbol{\theta}$ to mitigate the effect of huge parameter values. Then, we compute the posterior distribution $p(\boldsymbol{\theta}|\mathcal{U}, \mathcal{B})$, which according to the Bayes' theorem reads:

$$p(\boldsymbol{\theta}|\mathcal{U}, \mathcal{B}) = \frac{p(\mathcal{B}|\mathcal{U}, \boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathcal{B}|\mathcal{U})}. \tag{2.22}$$

By considering a MAP estimator we get the minimization problem in (2.19), where we set $\mathcal{R}(\boldsymbol{\theta}) = -\log\left(p(\boldsymbol{\theta})\right)$. If we now assume a Gaussian prior with variance γ^2, we obtain $\mathcal{R}(\theta) = \frac{1}{2\gamma^2}\|\boldsymbol{\theta}\|_2^2$. In Figure 2.4c we show the smoothing effect of this Gaussian prior when considering the family of polynomials with degree $M = 7$.

Learning models for image restoration

Another interesting strategy to provide a reliable solution of the linear IR task (1.8) is to directly solve the inverse problem by computing $\hat{\mathbf{A}}$, the "regularized" counterpart of the inverse of $\mathbf{A}$. Similarly to the SI task, we assume the structure of $\hat{\mathbf{A}}$ is known. Hence, $\hat{\mathbf{A}}$ belongs to a parametric space of operators $\hat{T}_\Theta = \{\hat{\mathbf{T}}(\cdot, \boldsymbol{\theta}) : \mathbb{R}^m \to \mathbb{R}^n | \ \boldsymbol{\theta} \in \Theta\}$ where $\Theta \subset \mathbb{R}^p$ is the space of parameters. Let $\mathcal{D} := \{(\mathbf{u}^j, \mathbf{b}^j)\}_{j=1\ldots N}$ be a set of ground-truth images and their corresponding measurements. The learning approach for the IR task (1.8) defines the following reconstruction map $\mathcal{E}^{\text{IR}}_{learn} : \mathbb{R}^m \to \mathbb{R}^n$ defined as:

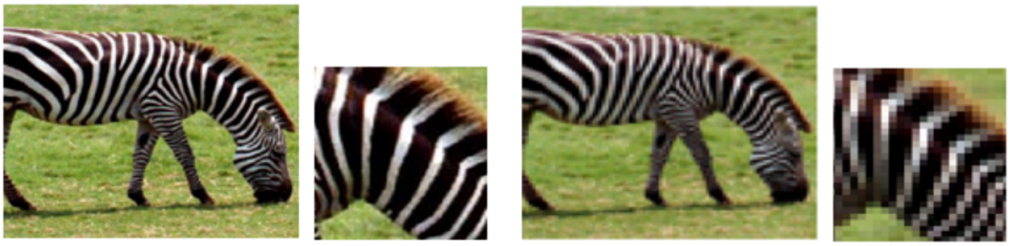
$$\mathcal{E}^{\text{IR}}_{learn} := \hat{\mathbf{T}}(\cdot, \boldsymbol{\theta}^*) \in \underset{\hat{\mathbf{T}}_{\boldsymbol{\theta}}, \boldsymbol{\theta} \in \Theta}{\arg\min} \ \mathcal{L}(\boldsymbol{\theta}, \mathcal{D}, \hat{\mathbf{T}}_{\boldsymbol{\theta}}) + \lambda \mathcal{R}(\boldsymbol{\theta}), \tag{2.23}$$

where by $\mathcal{L}$ we denote a measure of error whose definition depends on the dataset $\mathcal{D}$ and on the IR task considered and by $\mathcal{R}$ we denote a regularizer defined on the set of parameters. For example the loss function and the regularizer are usually defined as:

$$\mathcal{L}(\boldsymbol{\theta}, \mathcal{D}, \hat{\mathbf{T}}_{\boldsymbol{\theta}}) = \sum_{i=1}^{N} \|\hat{\mathbf{T}}(\mathbf{A}^{\dagger}(\mathbf{b}_i), \boldsymbol{\theta}) - \mathbf{u}_i\|_2^2, \quad \mathcal{R}(\boldsymbol{\theta}) = \|\boldsymbol{\theta}\|_2^2, \tag{2.24}$$

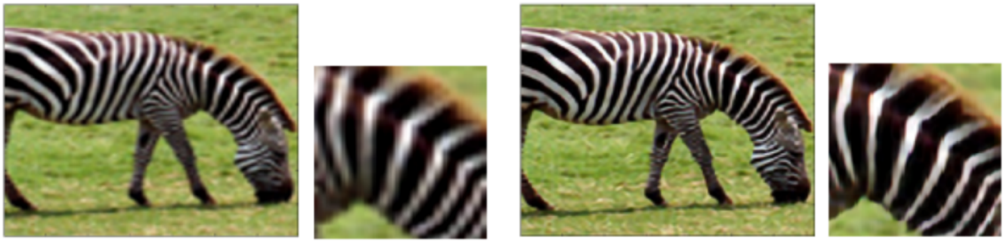
where $\mathbf{A}^{\dagger}$ can be chosen as the identity operator or a coarse approximation of the inverse of $\mathbf{A}$. We remark that in the variational approach the outcome of the reconstruction map $\mathcal{E}^{\mathrm{IR}}_{var}$ in (2.7) is equivalent to a regularized minimization problem, in the learning approach the reconstruction map $\mathcal{E}^{\mathrm{IR}}_{learn}$ is the solution of a minimization problem and can be directly used to invert the degradation process. More precisely, given an unseen degraded image $\mathbf{b}$, $\mathcal{E}^{\mathrm{IR}}_{learn}(\mathbf{b}) := \mathbf{T}(\mathbf{A}^{\dagger}\mathbf{b}, \boldsymbol{\theta}^*)$ is an estimate of the unknown clean image $\mathbf{u}$.

As an example, in Figure 2.5 we show how, exploiting the learning framework, we can efficiently solve the super resolution task. In particular, in Figure 2.5a we show the original HR `zebra` image (580×380), whereas in Figure 2.5b we show its LR counterpart (145×95). In Figure 2.5c we show the HR outcome of the Bicubic interpolation algorithm. We consider the Bicubic interpolation as coarse approximation $\mathbf{A}^{\dagger}$ and we estimate the parameters $\boldsymbol{\theta}^*$ by solving (2.23) once defined $\mathcal{L}$ and R as in (2.24) setting $\lambda = 0.001$. We consider $\hat{T}_{\Theta}$ as the class of Neural Networks whose structure is equal to the VDSR neural network described in [38]. Then, our result is depicted in Figure 2.5d.



(a) HR

(b) LR



(c) Bicubic HR estimation

(d) VDSR HR estimation

Figure 2.5: (a) original HR `zebra` image. (b) LR `zebra` image. (C) HR image obtained applying the Bicubic algorithm. (d) HR image obtained applying a learning model using the VDSR neural network.

Beyond learning models for image restoration

The idea of learning an approximation of the inverse of $\mathbf{A}$ to solve an IR task is interesting and has several advantages but also many limitations. The approach does not require neither the definition of a suitable regularizer inducing desired properties on the solution or information on the acquisition process, which are key points of the variational approach. However, the learning approach requires a large dataset of degraded-clean images, and in practice, this is not always possible (e.g. in medical imagery). Moreover, it has been proved to be largely dependent on the dataset fixed, thus being not efficient for data which are far from the distribution described by the fixed dataset.
Very recently, another approach called Deep Image Prior (DIP) has been proposed in [39] for solving many IR tasks such as denoising, deblurring, super resolution and inpaiting. In particular, the idea is to parametrize the solution $\mathbf{u}$ through a randomly initialized CNN f depending on a set of parameters $\boldsymbol{\theta} \in \mathbb{R}^p$ and to consider the following optimization problem:

$$\arg\min_{\boldsymbol{\theta}\in\mathbb{R}^p} \|\mathbf{A} f(\boldsymbol{\theta}, z) - \mathbf{b}\|_2^2, \tag{2.25}$$

where $\mathbf{z}$ is a fixed input and $\mathbf{A}$ is the forward operator defining the IR task. Then, an estimate $\mathbf{u}^*$ of the solution $\mathbf{u}$ is given by computing $f(\boldsymbol{\theta}^*, z)$ where $\boldsymbol{\theta}^*$ is obtained by early-stopping the iterative algorithm solving (2.25). The DIP approach is somewhat surprising since shows that the sole parametrization through a CNN is able to capture most of the low-level image statistics, differently from other standard learning based strategies which require large sets of training data. In practice, what has been observed is that during the optimization, more and more details are added (see Figure (2.6)). However, an appropriate stopping criterion is essential to avoiding overfitting. Thus, we stress that, as well as the PnP framework, the DIP approach can be regarded as an hybrid approach in-between variational and learning models. More precisely, similarly to learning models the optimization is performed on the set of parameters (of a CNN), but as well as for standard variational models the optimization step must be repeated when considering a different data $\mathbf{b}$. We will inspect in details the DIP framework in Chapter 7.

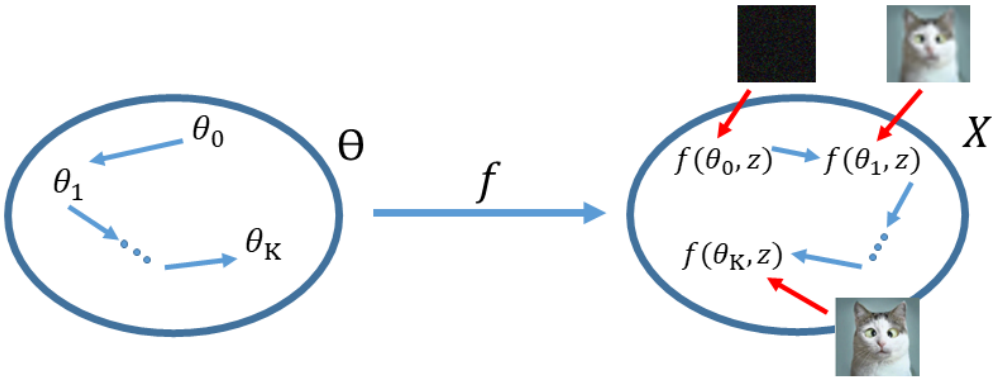


Figure 2.6: Deep Image Prior scheme. The unknown clean image is parametrized by a CNN f and the optimization is performed in the space of parameters. The iterative process is early-stopped at the iterate K-th to avoid overfitting.

Part II

Variational models for image inverse problems

Chapter 3

Super resolution for thermal imaging

Thermal imaging is the process of converting infrared radiation (heat) into visible images that depict the spatial distribution of temperature differences in a scene viewed by a thermal camera. The continuous growth of thermal remote sensing applications, for example, to investigate energy leak detection in buildings [40] or to support precision agriculture [41], has been determining an increasing demand for better spatial resolution, in order to analyse finer details of surface temperature patterns. The spatial resolution affects temperature readings, especially for objects of a comparable or smaller size than the pixel footprint. This feature is crucial for thermal remote sensing applications, where the measurement of the surface temperature of hot spots and cold spots, as well as average values on homogeneous surfaces, is often the primary goal.
However, the limited spatial resolution of the sensors operating in the thermal band (wavelengths in the range 8-14 microns) [42] makes the use of SR methods particularly appealing to overcome hardware technological limitations. In the literature, many approaches based on different criteria (interpolation or regularized variational) have been proposed to solve the single image super resolution (SISR) and the multiple image super resolution (MISR) tasks in the context of thermal images. In the field of SR for thermal remote sensing, although all the existing approaches provide satisfying performances, they have some downsides [43], namely some are not fully automatic and require a costly parameter set-up, others apply either to single or to multiple image super resolution.
A possible strategy is to estimate the unknown HR image $\mathbf{u} \in \mathbb{R}^n$ through variational models where the TV functional is considered as regularizer. This is equivalent to solve the following unconstrained optimization problem:

$$\mathbf{u}^*(\lambda) \in \underset{\mathbf{u}\in\mathbb{R}^n}{\arg\min}\ \Phi(\mathbf{u},\lambda) \equiv \frac{1}{2}\|\mathbf{A}\mathbf{u}-\mathbf{b}\|_2^2 + \lambda \mathrm{TV}(\mathbf{u}), \tag{3.1}$$

where by $\mathbf{u}^*(\lambda)$ we denote the solution corresponding to the fixed choice of $\lambda > 0$ and by $\mathbf{A}$ we refer to the super resolution forward operator. However, it is well-known that a critical issue of this model is the correct selection of the parameter λ balancing the strength of the regularization.

Contribution

This chapter is based on the publication [43] where we propose an automatic ℓ_2-TV based variational method for thermal images SR. Unlike many other existing methods [44, 45, 46, 47, 48], the algorithm is fully automatic, e.g. all the parameters are adaptively chosen, it is suited for both SISR and MISR of thermal images, preserves a limited computational cost and its reconstruction show better radiometric quality. For these features, it is possibly implementable in the thermal camera firmware. More precisely, inspired by [49], we consider a fully automatic method developed on two loops where the ℓ_2-TV problems arising in the inner loop are solved by means of the Forward Backward Splitting (FBS) algorithm (see Appendix B). We propose a new updating rule for the regularization parameter which is estimated iteratively along the optimization process. Furthermore, we show a decreasing behaviour of the sequence of regularization parameter which guarantees the convergence of our approach. Our proposal improves both the quality of the results and the convergence speed of the algorithm in [49] when tested on thermal images.

3.1 On the choice of the regularization parameter

In this section we review some standard strategies for the automatic estimation of the parameter parameter λ in (3.1) for the generic IR task (1.7). We focus on those strategies exploiting only the information encoded in $\mathbf{b}$, thus avoiding learning-based strategies relying on a data set of examples.

Tipically, the selection of a proper value of λ is done through trial-and-error procedures, namely several values of the regularization parameter are sampled from a fixed bounded interval $[\lambda_{\min}, \lambda_{\max}]$ and then used in (3.1). The optimal parameter is selected as the one providing the best solution in terms of a fixed metric or visual inspection. Other widely investigated approaches for automatically tuning the regularization parameter are the Generalized CrossValidation (GCV) [50] and the L-curve method [51]. More precisely, the GCV defines the optimal regularization parameter as the solution of the following optimization problem:

$$\lambda^* \in \underset{\lambda \in \mathbb{R}^+}{\arg\min} \|\mathbf{A}\mathbf{u}^*(\lambda) - \mathbf{b}\|_2^2. \tag{3.2}$$

The L-curve method selects several regularization parameters and builds the so-called L-curve plot by placing the values $\mathrm{TV}(\mathbf{u}^*(\lambda))$ on the x-axis and the values of the residual norm $\|\mathbf{A}\mathbf{u}^*(\lambda) - \mathbf{b}\|_2^2$ on the y-axis. The optimal λ is the one corresponding to a corner in the L-curve plot.
However, in spite of their good performances, the computational cost of both GCV and L-curve is still too high for real-time applications. Indeed, they require the solution of several optimization problems as the one in (3.1) which are usually solved through iterative schemes [18] and do not admit closed-form solutions, thus being cumbersome especially when dealing with high-dimensional data.
To overcome this limitation, in [52, 53, 54, 55], to name a few, the authors develop some strategies to estimate the regularization strength along the iterations of the optimization algorithm chosen for solving the ℓ_2-TV problem (3.1). All of them exploit the so-called Morozov discrepancy principle [56]. By considering the acquisition model (1.7), the core assumption of this principle is that the optimal solution $\mathbf{u}$ should satisfy the equation $\|\mathbf{A}\mathbf{u} - \mathbf{b}\|_2^2 = \|\mathbf{e}\|_2^2$, where the right hand side is

equal, in expectation, to the quantity $\sigma_{\mathbf{e}}^2 m$.
Therefore, the discrepancy principle looks for an estimate of $\mathbf{u}$ belonging to the set:

$$D_{\sigma_{\mathbf{e}}} = \{\mathbf{u} \in \mathbb{R}^n | \ \|\mathbf{A}\mathbf{u} - \mathbf{b}\|_2 < \delta = \tau\sigma_{\mathbf{e}}\sqrt{m}\}, \tag{3.3}$$

where τ is chosen such that $\tau > 1$ (over-estimating the noise level) or $\tau < 1$ (under-estimating the noise level).
The Morozov discrepancy principle provides good restorations but strongly depends on the knowledge of the noise variance and can perform poorly when a bad estimate is considered.

An interesting idea for an automatic estimation of the strength of the regularization was developed in [49] where the authors propose a novel adaptive rule for an ℓ_2-TV model solving the deblurring task. Their method is inspired by the continuation strategies used in [57, 58] to solve problems of form (3.1) in a compressed sensing framework.
A common practice in this framework is to make use of the empirical reduction rule:

$$\lambda_k = r \cdot \lambda_{k-1}, \tag{3.4}$$

where $r \in [0, 1]$ is called reduction factor and is usually chosen as a constant value and k represents the index of the iterative optimization process.
In [49] the authors assume the sequence of regularization parameters $\{\lambda_k\}_{k\in\mathbb{N}}$ follows the evolution of the objective functional Φ in (3.1) and reads:

$$\lambda_k = \frac{\Phi(\mathbf{u}^*(\lambda_{k-1}), \lambda_{k-1})}{\Phi(\mathbf{u}^*(\lambda_{k-2}), \lambda_{k-2})} \cdot \lambda_{k-1}, \tag{3.5}$$

thus without requiring any information about the noise level.
More precisely, the method described in [49] is developed on two loops. The outer loop updates the regularization parameter according to the formula reported in (3.5). The inner loop, for a fixed λ_k finds $\mathbf{u}^*(\lambda_k)$ by solving (3.1) through the iterative FBS method whose starting guess is chosen equal to $\mathbf{u}^*(\lambda_{k-1})$. The latter initialization is also known as warm start and guarantees a faster convergence speed of the FBS algorithm. The close connection between the regularization parameter definition and the functional values let the iterative algorithm to automatically stop when the objective functional Φ maintains the same value for two successive outer iterations.
In other words, upon suitable initializations, the method in [49] solves a sequence of ℓ_2-TV minimization problems which only differ by the magnitude of the regularization parameter defined according to the rule in (3.5).

3.2 ASRp: an automatic super resolution algorithm

In the Section 2.1 we have shown the link between the ill-conditioning of the operators defining the IR tasks and the ill-posedness of the related inverse problem. The SR acquisition model (1.17) considers the blurring operator which is known to be an ill-conditioned operator and the downsampling operator which is non injective (see Figure 1.9). Therefore, the SR task which aims at retrieving an HR image from one or more than one LR image, is an ill-posed inverse problem.

When only one LR image is available the SR task refers to as single image super resolution task, whereas when more LR images are available the SR task refers to as multiple image super resolution task. A possible estimate of the unknown HR image can be provided through variational models.

Let $\mathcal{G} = \{\mathbf{B}^j\}_{j=1}^r$ be a set of LR images of dimension $n_r \times n_c$, representing the same scene. An estimate of the HR image $\mathbf{U} \in \mathbb{R}^{N_r \times N_c}$, such that $N_r = L \cdot n_r$ and $N_c = L \cdot n_c$ with $L > 1$, can be provided by solving the following constrained minimization problem:

$$\mathbf{u}^* \in \arg\min_{\mathbf{u}\in\mathbb{R}^n} \mathrm{TV}(\mathbf{u}), \quad \text{s.t.} \quad \frac{1}{2r}\sum_{j=1}^{r} \|\mathbf{S}^j\mathbf{H}^j\mathbf{u} - \mathbf{b}^j\|_2^2 \leq \delta, \quad \delta > 0 \tag{3.6}$$

where δ is related to the noise standard deviation, $\mathbf{u} \in \mathbb{R}^n$, with $n = L^2 \cdot n_r \cdot n_c$, and $\mathbf{b}^j \in \mathbb{R}^m$, with $m = n_r \cdot n_c$, are vectors obtained by lexicographically reordering the 2D HR and LR images $\mathbf{U}$ and $\mathbf{B}^j$, respectively, for $j = 1 \dots r$. The matrices $\mathbf{S}^j$ and $\mathbf{H}^j$ represent the downsampling and the blur operators modelling the data acquisition process of each LR image. We assume that the same zero-mean Gaussian blur, of variance $\sigma^2_{\mathbf{H}}$, and downsampling operators, with downsampling factor L, apply to each LR image in the acquisition set $\mathcal{G}$; therefore, $\mathbf{H} = \mathbf{H}^j$ and $\mathbf{S} = \mathbf{S}^j$ and $\forall j = 1 \dots r$.

We now describe the Adaptive Super Resolution algorithm introduced in [43], which is referred to as ASRp, in the following.

We solve the constrained minimization problem (3.6) through a sequence of unconstrained penalized problems of the form:

$$\mathbf{u}^*(\lambda_k) = \arg\min_{\mathbf{u}\in\mathbb{R}^n} \Phi(\mathbf{u}, \lambda_k) \equiv \frac{1}{2r}\sum_{j=1}^{r} \|\mathbf{SHu} - \mathbf{b}^j\|_2^2 + \lambda_k \mathrm{TV}(\mathbf{u}), \ k = 0, 1, \dots \tag{3.7}$$

where we assume the sequence $\{\lambda_k\}_{k\in\mathbb{N}}$ is monotonically decreasing. This property ensures that the sequence $\{\mathbf{u}^*(\lambda_k)\}_{k\in\mathbb{N}}$ converges to the solution of (3.6) as demonstrated in [59].

Moreover, we observe that $\Phi(\cdot, \lambda)$ is a strictly convex function therefore, for each $\lambda > 0$, there exists a unique minimizer $\mathbf{u}^*(\lambda)$.

In the following Theorem 3.1 and Theorem 3.2 we describe the proposed rule for choosing a suitable decreasing sequence $\{\lambda_k\}_{k\in\mathbb{N}}$.

Theorem 3.1. *Let be λ_t and λ_s two positive regularization parameters satisfying $\lambda_t < \lambda_s$ and $\mathbf{u}^*(\lambda_t)$ and $\mathbf{u}^*(\lambda_s)$ the minimizers of $\Phi(\cdot, \lambda_t)$ and $\Phi(\cdot, \lambda_s)$, respectively. Then,*

$$\Phi(\mathbf{u}^*(\lambda_t), \lambda_t) < \Phi(\mathbf{u}^*(\lambda_s), \lambda_s).$$

Proof. Let us consider $\lambda_t < \lambda_s$. By definition of Φ we have:

$$\frac{1}{2r}\sum_{j=1}^{r} \|\mathbf{SHu}^*(\lambda_t) - \mathbf{b}^j\|_2^2 + \lambda_t \mathrm{TV}(\mathbf{u}^*(\lambda_t)) < \frac{1}{2r}\sum_{j=1}^{r} \|\mathbf{SHu}^*(\lambda_s) - \mathbf{b}^j\|_2^2 + \lambda_t \mathrm{TV}(\mathbf{u}^*(\lambda_s)) < \tag{3.8}$$

$$< \frac{1}{2r}\sum_{j=1}^{r} \|\mathbf{SHu}^*(\lambda_s) - \mathbf{b}^j\|_2^2 + \lambda_s \mathrm{TV}(\mathbf{u}^*(\lambda_s)). \tag{3.9}$$

The inequality (3.9) follows from the assumption $\lambda_t < \lambda_s$. Hence, $\Phi(\mathbf{u}^*(\lambda_t), \lambda_t) < \Phi(\mathbf{u}^*(\lambda_s), \lambda_s)$. □

Theorem 3.2. *Let $\mathbf{u}_{init}$ be the warm start for the iterative process solving (3.7) when $k = 0$, and let λ_0 be a proper initialization of the sequence of regularization parameters $\{\lambda_k\}_{k\in\mathbb{N}}$. Let $\mathbf{u}^*(\lambda_k)$ for $k = 0, 1, \dots$ be defined as in (3.7) and set $p \geq 1$. If we define:*

$$\lambda_k = \begin{cases} \dfrac{\Phi(\mathbf{u}^*(\lambda_{k-1}), \lambda_{k-1})}{\Phi(\mathbf{u}_{init}, \lambda_0)} \cdot \lambda_{k-1}, & k \leq p \\ \dfrac{\Phi(\mathbf{u}^*(\lambda_{k-1}), \lambda_{k-1})}{\Phi(\mathbf{u}^*(\lambda_{k-1-p}), \lambda_{k-1-p})} \cdot \lambda_{k-1}, & k \geq p+1, \end{cases} \tag{3.10}$$

then $\lambda_k < \lambda_{k-1}, \forall k \geq 1$.

Proof. By induction on k.
Case $k = 1$. By definition $\Phi(\mathbf{u}^*(\lambda_0), \lambda_0) < \Phi(\mathbf{u}_{\text{init}}, \lambda_0)$, therefore the formula in (3.10) entails $\lambda_1 < \lambda_0$.
Moreover, from Theorem 3.1 it follows that:

$$\Phi(\mathbf{u}^*(\lambda_1), \lambda_1) < \Phi(\mathbf{u}^*(\lambda_0), \lambda_0) < \Phi(\mathbf{u}_{\text{init}}, \lambda_0). \tag{3.11}$$

Case $k = 2$. By (3.10) and (3.11) follows:

$$\frac{\Phi(\mathbf{u}^*(\lambda_1), \lambda_1)}{\Phi(\mathbf{u}^*(\lambda_0), \lambda_0)} < 1, \frac{\Phi(\mathbf{u}^*(\lambda_1), \lambda_1)}{\Phi(\mathbf{u}_{\text{init}}, \lambda_0)} < 1,$$

hence, we have $\lambda_2 < \lambda_1$. Therefore, from Theorem 3.1:

$$\Phi(\mathbf{u}^*(\lambda_2), \lambda_2) < \Phi(\mathbf{u}^*(\lambda_1), \lambda_1) < \Phi(\mathbf{u}^*(\lambda_0), \lambda_0) < \Phi(\mathbf{u}_{\text{init}}, \lambda_0).$$

Case $k > 2$. Let us assume that $\lambda_{k-1} < \lambda_{k-2} < \cdots < \lambda_0$. Then:

$$\Phi(\mathbf{u}^*(\lambda_{k-1}), \lambda_{k-1}) < \Phi(\mathbf{u}^*(\lambda_{k-2}), \lambda_{k-2}) < \cdots < \Phi(\mathbf{u}_{\text{init}}, \lambda_0).$$

Hence

$$\frac{\Phi(\mathbf{u}^*(\lambda_{k-1}), \lambda_{k-1})}{\Phi(\mathbf{u}^*(\lambda_{k-1-p}), \lambda_{k-1-p})} < 1, \frac{\Phi(\mathbf{u}^*(\lambda_{k-1}), \lambda_{k-1})}{\Phi(\mathbf{u}_{\text{init}}, \lambda_0)} < 1.$$

For $p \geq 1$, by applying (3.10) we get $\lambda_k < \lambda_{k-1}$. □

Implementation notes

In the following we illustrate the implementation details of the proposed ASRp algorithm where p refers to the parameter for the computation of the sequence $\{\lambda_k\}_{k\in\mathbb{N}}$ according to (3.10).

The starting guess $\mathbf{u}_{\text{init}}$ of the iterative process is essential for the convergence speed of the method. Many existing variational methods use bicubic or bilinear interpolated HR images, but their quality is often not satisfactory. We propose an initial artefact free HR image obtained by a smoothing filter:

$$\mathbf{u}_{\text{init}} = \frac{1}{r} \sum_{j=1}^{r} (\mathbf{H}^T \mathbf{S}^T \mathbf{b}^j). \tag{3.12}$$

We observe that, since the acquired images are not constant, we avoid the trivial case $\mathbf{u}_{\text{init}}$ constant; hence $\text{TV}(\mathbf{u}_{\text{init}})$ is different from zero. Concerning the starting value λ_0, we set:

$$\lambda_0 = \frac{1}{2r}\sum_{j=1}^{r}\frac{\|\mathbf{S}\mathbf{H}\mathbf{u}_{\text{init}} - \mathbf{b}^j\|_2^2}{2\text{TV}(\mathbf{u}_{init})}. \tag{3.13}$$

We note that this choice guarantees that the value of the objective function $\Phi(\mathbf{u}_{\text{init}}, \lambda_0)$ is the average of the initial residual norms, i.e. :

$$\Phi(\mathbf{u}_{\text{init}}, \lambda_0) = \frac{1}{r}\sum_{j=1}^{r}\|\mathbf{S}\mathbf{H}\mathbf{u}_{\text{init}} - \mathbf{b}^j\|_2^2. \tag{3.14}$$

As in [49], for each value $\lambda_k,\ k = 0, 1, \ldots$, the vector $\mathbf{u}^*(\lambda_k)$ solving the minimization problem (3.7) is computed by applying the Accelerated Forward Backwards algorithm (AFB), reported in detail in Appendix B. The AFB method implements the Fast Iterative Shrinkage Thresholding strategy (FISTA) [60], while solving the arising TV-regularized denoising subproblem with the Chambolle algorithm [61].

The application of (3.12)-(3.13) to compute the starting guess and (3.10) to update the regularization parameters λ_k yields the ASRp algorithm whose steps are listed in Algorithm 1.
The algorithm iterations (index k) are stopped on the basis of a relative error tolerance τ_e, while τ_f represents the tolerance for the inner iterations of the AFB algorithm. Both the tolerances are fixed equal to 10^{-5}.

Algorithm 1 – ASRp: Adaptive Super-Resolution algorithm

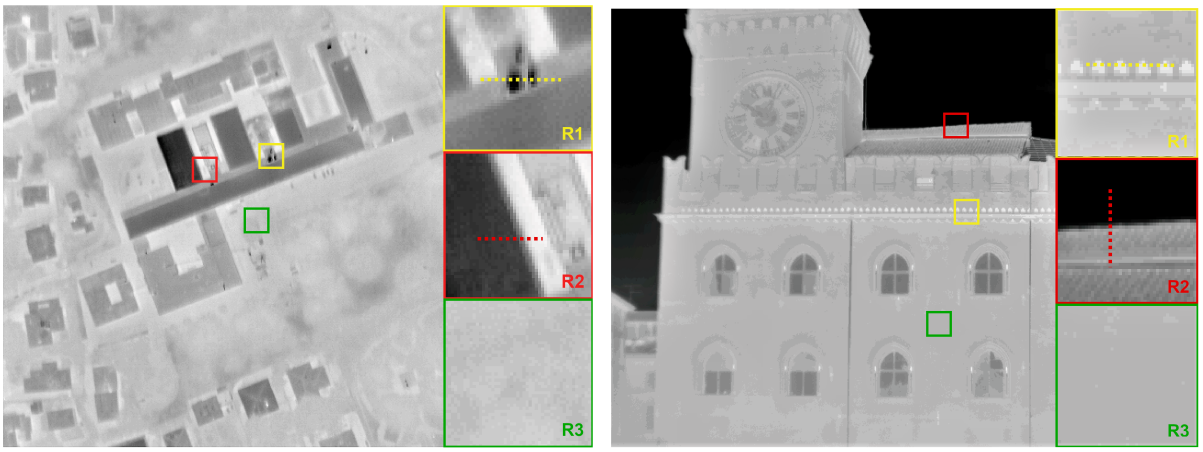
input: $\mathbf{H} \in \mathbb{R}^{n\times n},\ \mathbf{S} \in \mathbb{R}^{m\times n},\ \mathcal{G},\ p,\ r,\ \tau_e,\ \tau_f$
output: $\mathbf{u}^*$

$k = 0$; $\mathbf{u}_{\text{init}} = \left(\sum_{j=1}^{r}(\mathbf{H}^T\mathbf{S}^T\mathbf{b}^j)\right)/r$;
$\lambda_0 = \left(\sum_{j=1}^{r}\|\mathbf{S}\mathbf{H}\mathbf{u}_{\text{init}} - \mathbf{b}^j\|_2^2\right) / (2r\text{TV}(\mathbf{u}_{\text{init}}))$
$[\mathbf{u}^*(\lambda_0)] = \text{AFB}(\mathbf{u}_{\text{init}}, \lambda_0, \mathbf{H}, \mathbf{S}, \mathcal{G}, r, \tau_f)$ (see Algorithm 13)
repeat
  $k = k + 1$
  `if` $k \geq p + 1$, $\text{R} = \Phi(\mathbf{u}^*(\lambda_{k-1}), \lambda_{k-1})/\Phi(\mathbf{u}^*(\lambda_{k-1-p}), \lambda_{k-1-p})$
  `else` $\text{R} = \Phi(\mathbf{u}^*(\lambda_{k-1}), \lambda_{k-1})/\Phi(\mathbf{u}_{\text{init}}, \lambda_0)$
  $\lambda_k = \text{R} \cdot \lambda_{k-1}$
  $[\mathbf{u}^*(\lambda_k)] = \text{AFB}(\mathbf{u}^*(\lambda_{k-1}), \lambda_k, \mathbf{H}, \mathbf{S}, \mathcal{G}, r, \tau_f)$ (see Algorithm 13)
until $\|\mathbf{u}^*(\lambda_k) - \mathbf{u}^*(\lambda_{k-1})\| \leq \tau_e\|\mathbf{u}^*(\lambda_k)\|$
$\mathbf{u}^* = \mathbf{u}_k$

3.3 Numerical results

We now apply the proposed ASRp on the dataset described in the previous section in order to show its effectiveness in the field of thermal imagery.

Initialization, parameters and evaluation metrics. All the parameters and all the variables are initialized as described in Algorithm 1. For what concerns the blurring matrix $\mathbf{H}$, based on technical information of the acquisition camera, we fix $\sigma_{\mathbf{H}} = 1.4$. Morever, as downsampling matrix $\mathbf{S}$ we consider the decimation operator whose upsampling factor L is fixed equal to 4 in all the experiments. For the simulated data, we evaluate the quality of the SR reconstructions by means of Peak-Signal-to-Noise-Ratio (PSNR) and Structure Similarity index (SSIM), while when no ground-truth image is available the quality of the results is assessed by visual inspection.
Comparisons. We carry on some comparisons with respect to interpolation approaches such as Nearest Neighbour (NN), Bicubic and the SR method proposed in [62] which is termed as RISR. Furthermore, we compare our ASRp with the variational approach proposed in [49] and the Enhanced Deep Super Resolution (EDSR) neural network [63].



(a) (b)

Figure 3.1: (a) High Resolution 480 × 640 I1 image with regions of interest (ROI) R1, R2 and R3 for the quantitative analysis of super-resolution results. (b) High Resolution 480 × 640 I2 image with regions of interest (ROI) R1, R2 and R3 for the quantitative analysis of super resolution results.

Materials

In order to evaluate the performance of the proposed ASRp algorithm, in [43], we have tested two different situations, by considering thermal images acquired both from an airborne platform and from the ground.

Aerial thermal image. The first test image is constituted by an aerial thermal image of size 480 × 640 pixels, representing the School of Engineering of Bologna University. It is part of the thermal aerial survey over the city of Bologna, held on 14 March 2016. The aerial image used for the simulations is denoted by I1 in the following and is shown in Figure 3.1a. The flight height was set to approximately 800 metres above the ground, in order to obtain a ground sampling distance of about 0.5 m. More than 2,500 frames were acquired to cover a total area of about 11 km^2. In this scenario, SR offers both the opportunity to appreciate finer details of the imaged surfaces and

to increase flight height and consequently reduce the number of frames to cover the same area. For more details the interested reader can refer to [43] and references therein.

Terrestrial thermal image. The second test image is taken from a set of 50 terrestrial thermal images of size 480×640 pixels. The subject of the thermal image is the façade of Palazzo D'Accursio, an historical building in Bologna. The terrestrial image used for the simulations is denoted by I2 in the following and it is shown in Figure 3.1b. In this scenario, capturing the whole subject in a single image implies the setting of the camera to a considerable distance or the use of an optic with a wider field of view. In both cases, the increased ground sampling distance produces a detail loss, which may be partially recovered through the application of SR. For more details the interested reader can refer to [43] and references therein.

Simulated thermal image datasets. We use I1 and I2 as ground-truth and we build simulated datasets of 32 LR thermal images by first applying small rotations and translations to I1 and I2 and then downsampling by a factor $L = 4$. The interested reader can refer to [43] for more details. In the following, we call I1 and I2 both the single images and the relative datasets; it will be clear from the context if we refer to one or more images.

Experimentation

Results by varying the hyperparameter p. We first analyze the performance of the ASRp method on the simulated data by varying the value of the parameter p involved in the update rule reported in (3.10). In Figure 3.2 we plot the results obtained with $p = 1$,$p = 2$ and $p = 3$ for SISR of I1 and I2 images. We remark that when choosing $p = 1$ our implementation equals the one proposed in [49]. The plots show how the choice of the positive integer p of the method, introduced in (3.10), affects the quality of the HR reconstruction, underlying the importance of a proper choice for the regularization parameter. In all the tests both for SISR and MISR carried out, we observe the same steeper decreasing behaviour of the objective function, together with the faster increase of the PSNR when we use $p = 2$ instead of $p = 1$. Moreover, we find out that the value $p \geq 3$ does not improve the PSNR. Hence, (3.10) represents a substantial improvement of the update rule introduced in [49]. In the following we fix $p = 2$ and we refer to our method as ASR2.

Comparisons on simulated datasets. In the following, we name our method ASR2_s when a single LR image is available, whereas we name our method as ASR2_m when more than one LR images are available. Our ASR2_m assumes the starting LR images are acquired from the same point of view, hence we adopt the same registration procedure described in [62].

In Table 3.1, we compare the performances of our ASR2_s and ASR2_m methods with the other competitors. The ASR2_s and ASR2_m methods show the best performances in terms of the SSIM and PSNR metrics, respectively, while their execution time is higher than the one provided by the EDSR, which is the second best method. However, we remark that the EDSR algorithm required a couple of days long training phase on a fixed set of examples and, moreover, the model trained fits only for a specific magnification factor.

In Figures 3.3a and 3.3b we plot the trends of the PSNR parameter of ASR2_m and the computational time while the number of input images increases. We deduce that an increasing number of

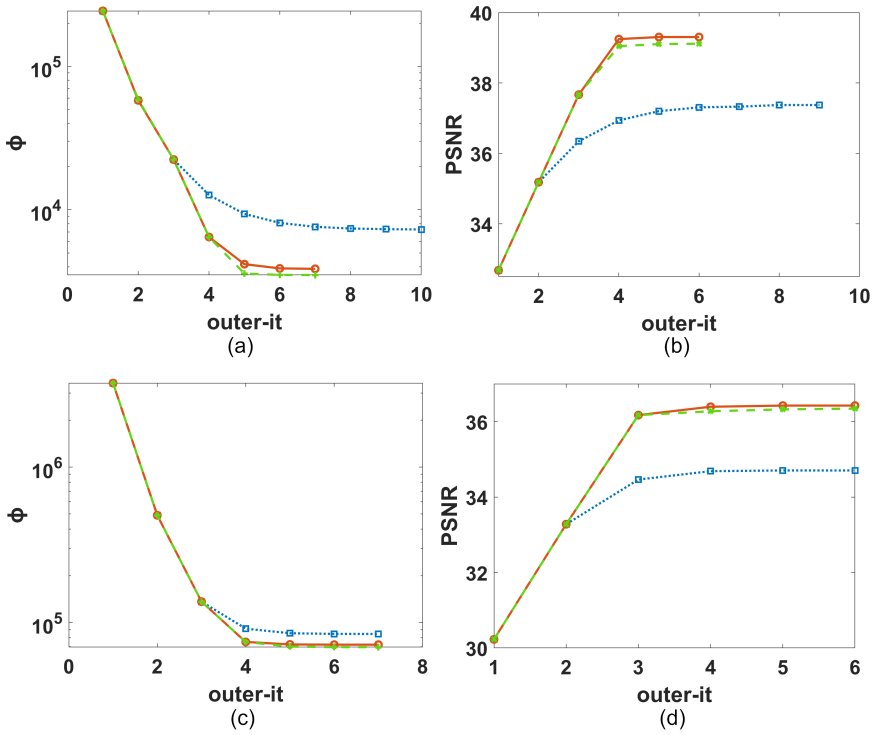


Figure 3.2: Comparison of ASRp results on SISR for $p = 1$ (blue dotted line), $p = 2$ (red continuous line) and $p = 3$ (green dashed line). On the left: $\Phi(\mathbf{u}^*(\lambda_k), \lambda_k)$ as a function of the outer iteration number for I1 (a) and I2 (c). On the right: PSNR values as a function of the outer iteration number for I1 (b) and I2 (d).

LR images positively affect the PSNR while the workload remains almost unchanged.
In Figure 3.4 we compare the reconstructed details of the I1 image (all the crops are represented in the same colour map). The ASR2 methods (both in case of SISR and MISR), which use the TV regularization, produce images with the sharpest contours; in particular, the black circle corresponding to a cold spike is better detected by the ASR2_s algorithm.

Evaluating the radiometric quality. A detailed radiometric evaluation of the HR images obtained with the SR algorithms (both in the case of single and multiple SR) is performed by identifying in I1 and I2 some Regions Of Interest (ROI) since in thermal images, the PSNR and SSIM parameters are not sufficient to evaluate the HR reconstructions. The ROIs chosen for our analysis are highlighted by the yellow, red and green boxes in Figure 3.1a and 3.1b. They contain different interesting features for the quantitative analysis. We distinguish between the ROIs R1 and R2, containing hot-cold spots or sharp edges, and R3 representing flat surfaces. In R1 and R2, we compute the following absolute errors:

$$\Delta T_M = |T_M(Y) - T_M(X)|, \quad \Delta T_m = |T_m(Y) - T_m(X)|, \tag{3.15}$$

where T_M and T_m are the maximum and minimum temperature values, respectively. By Y and X we denote the ground truth image and the HR computed image. Conversely, we evaluate the quality of the flat region R3 through the standard deviation $std = 1/(n-1)\sum_{i=1}^{n}(X_i - \bar{X})^2$ where X_i are the values of the pixels in R3, and $\bar{X}$ is their mean. The values for each ROI are reported

Table 3.1: Comparison among the proposed ASR2_s and ASR2_m methods, the NN, the Bicubic, the RISR and the EDSR methods in case of SR for the test images I1 and I2. ASR2_m and RISR consider 32 LR images. The column "time" represents the computation time in seconds.

Test	Method	PSNR	SSIM	time (s)
I1	Bicubic	37.803	0.470	**0.08**
	NN	36.473	0.447	0.25
	EDSR	39.499	0.524	5.998
	RISR	39.193	0.528	1894.63
	ASR2_s	39.408	**0.566**	254.09
	ASR2_m	**40.482**	0.549	187.11
I2	Bicubic	34.236	0.540	**0.004**
	NN	32.000	0.504	0.01
	EDSR	37.414	0.564	6.299
	RISR	35.204	0.545	2031.61
	ASR2_s	36.638	**0.605**	221.78
	ASR2_m	**37.483**	0.555	155.66

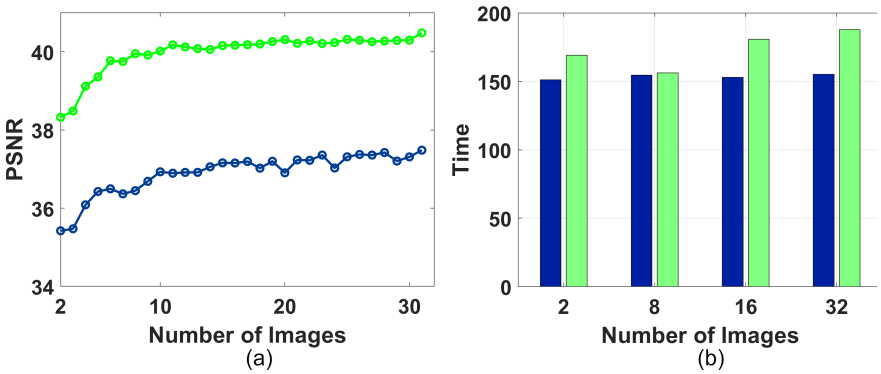


Figure 3.3: Results of the ASR2_m algorithm using an increasing number of input images for the test images I1 and I2. (a) PSNR versus the number of input images. I1: green dashed line; I2: blue continuous line. (b) Execution times in seconds. I1: green bars; I2: blue bars.

in Tables 3.2 and 3.3 where we highlight the best results according to the following observations. Column R1 in Table 3.2 is relative to a single cold spike; hence a small value of ΔT_m represents the ability of the method to capture such feature. The minimum value ΔT_m, represented in boldface, is obtained by the ASR2_s. On the contrary, in Table 3.3 column R1 is relative to several hot spikes. In this case, small value ΔT_M represents the capacity of the method to reproduce such situation and the best result, highlighted in column ΔT_M, is obtained by the ASR2_s. ASR2_s outperforms the other methods of more than 60% on both the cold spike (for I1) and hot spike (for I2). Column R2 of both tables represents an edge and the average value, between ΔT_M and ΔT_m, evaluates its quality. In both cases, ASR2_m reaches the best value. Finally the R3 columns of tables 3.2 and 3.3 are relative to homogeneous surfaces. In this case, the best results are obtained by the minimum value of the standard deviation which is reached by ASR2_m for I2 and by the RISR method for I1 (we observe that in Table 3.3 all the methods show a very similar performance).

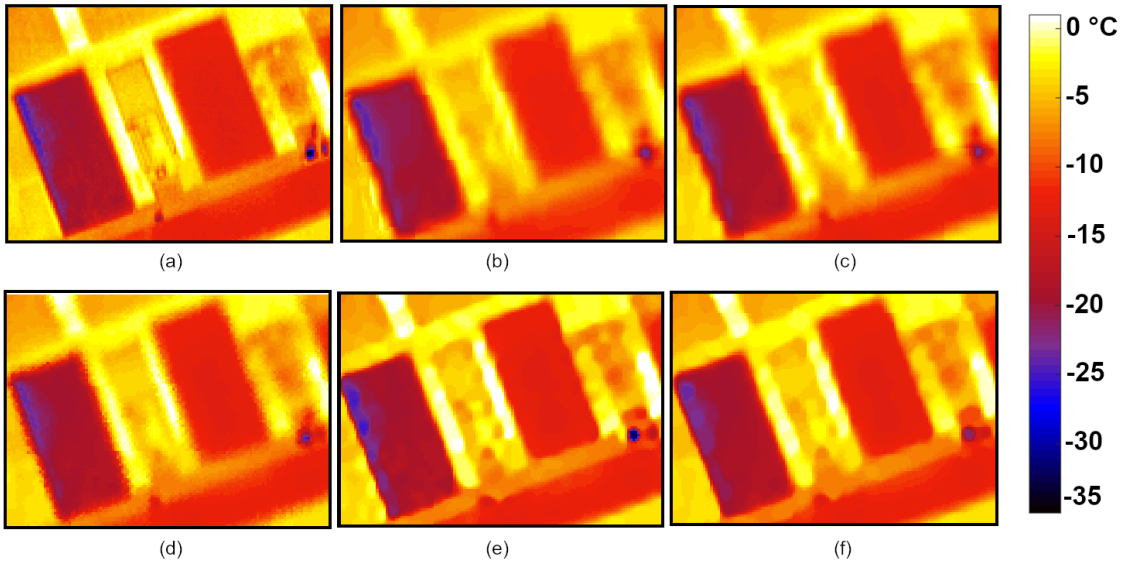


Figure 3.4: Comparisons among HR images obtained by different methods (detail of I1 image). All the images are depicted in the same color map. (a) Ground truth image. (b) Bicubic HR image. (c) EDSR HR image. (d)RISR HR image. (e) ASR2_s HR image. (f) ASR2_m HR image.

Table 3.2: Results of the tests on the I1 image with different methods. ΔT_M and ΔT_m are defined in (3.15) and are expressed in Celsius degrees; *std* is the standard deviation. For each ROI, highlighted in bold font are the values that best reproduce the regions specific features.

Method	R1	R2	R3
	ΔT_m	$(\Delta T_M + \Delta T_m)/2$	std
NN	13.405	0.732	0.136
Bicubic	17.605	2.123	0.135
EDSR	13.300	1.015	0.143
ASR2_s	**4.382**	0.698	0.179
RISR	15.813	1.688	**0.131**
ASR2_m	12.795	**0.346**	0.132

Experiments on real data. The last experiment considers the set of 50 thermal images of size 480×640 representing the same scene of I2 described previously. Those images are given as registered LR input to the super-resolution methods. Two portions of the HR images, obtained by the methods NN, Bicubic, EDSR, RISR, ASR2_s and ASR2_m are shown in Figure 3.5.
We observe that in the ASR2_s images (Figure 3.5c) the edges are well enhanced but the noise is still visible in the background, while the ASR2_m method produces very well defined HR image details compared to the other methods. The NN, Bicubic and EDSR methods show many artifacts when they reconstruct small details such as the round arches (Figures 3.5a, 3.5b, 3.5c). The RISR method returns an high quality HR image but with an artificial texture pattern (Figure 3.5d). We point out that all the crops are represented with the same colour map.

Table 3.3: Results of the tests on the I2 image with different methods. ΔT_M and ΔT_m are defined in (3.15) and are expressed in Celsius degrees; *std* is the standard deviation. For each ROI, highlighted in bold font are the values that best reproduce the regions specific features.

Method	R1	R2	R3
	ΔT_M	$(\Delta T_M + \Delta T_m)/2$	std
NN	1.295	11.644	0.036
Bicubic	0.336	2.133	0.026
EDSR	1.191	0.672	0.035
ASR2_s	**0.434**	2.261	0.028
RISR	1.557	6.004	0.020
ASR2_m	1.139	**0.646**	**0.017**

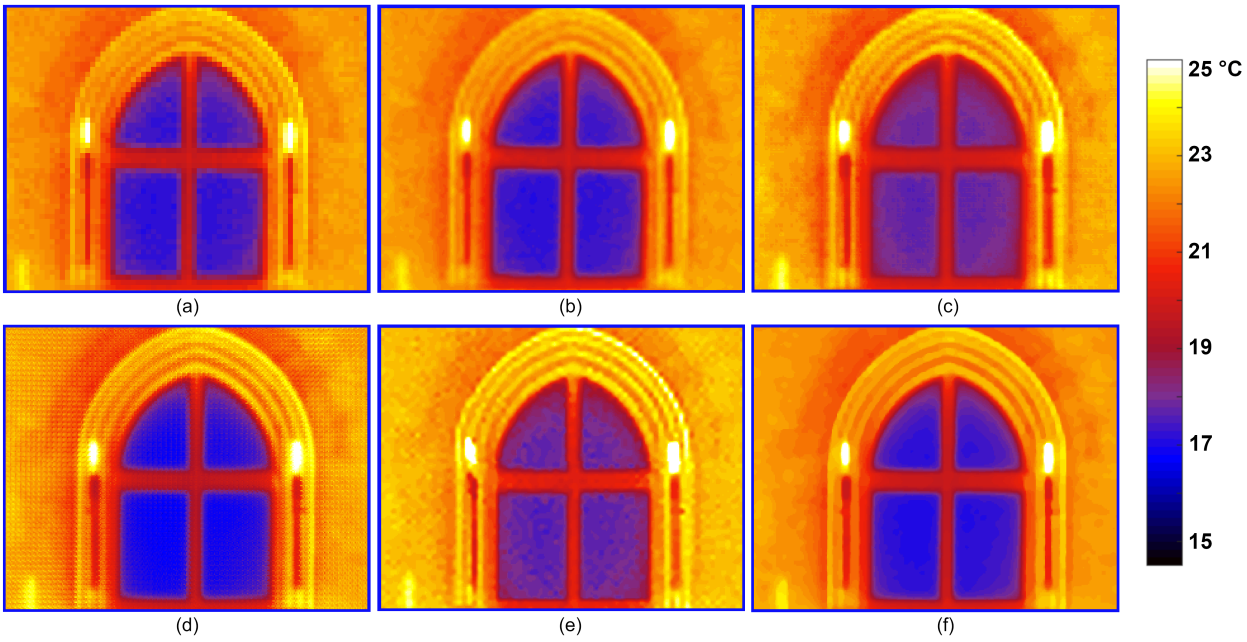


Figure 3.5: Window detail of the SR reconstructed images by different methods. (a) NN. (b) Bicubic. (c)EDSR. (d) RISR. (e) ASR2_s. (f) ASR2_m.

Chapter 4

Inverse Potts models for joint image super resolution and segmentation

Single image super resolution and Image Partitioning (IP) are two very popular tasks in the field of image processing. We recall that in SISR, the objective is to enhance the spatial resolution of a given LR and possibly blurred and noisy image so as to retrieve a HR version enhancing the quality of the LR data. The LR data acquisition model of reference is the one reported in (1.17). In this chapter we assume AWGN with standard deviation $\sigma_{\mathbf{e}}$. In IP (a technique which is often referred to as image segmentation), the objective is to extract, from a given digital image, regions of interest on the basis of geometric/semantic information. Such task is typically performed to facilitate subsequent data classification and labeling.
A standard approach to extract a suitable IP from LR data consists in solving the SISR and the IP tasks in a disjoint sequential manner. A clear limitation of such sequential strategy is that the quality of the partitioning obtained as final result depends on the quality of the super-resolved image obtained after performing the former reconstruction step. Typically, this depends on the SISR model used and on the accurate choice of its hyperparameters.
To overcome these limitations, a joint SR and IP approach performing both tasks at the same time has been proposed, e.g., in [64, 65] based on a Bayesian approach and, more recently, in [66] based on learning strategies.

In a variety of works [67, 68, 69, 70], Storath et al. consider the inverse Potts regularization model for joint image restoration and segmentation. Such approach belongs to the class of variational models and uses a regularizer defined in terms of the ℓ_0-type gradient smoothing prior introduced in [25], where the latter has been shown to favour a significant image smoothing which preserves salient image edges and eliminate insignificant details, thus favouring simplified and almost-partitioned reconstructions that can be easily used for subsequent segmentation purposes. Furthermore, a constrained ℓ_0-gradient based variational model was proposed in [71] and applied to image denoising problems. This alternative approach can be used in place of the unconstrained ones proposed by Storath et al., whenever information on the number of jumps (i.e. image discontinuities) of the desired solution is available.

Therefore, the inverse Potts modeling, based on ℓ_0-gradient regularization, suggests an alternative approach to Bayesian and learning strategies. To estimate an HR partitioned image $\mathbf{u}^*$ we can solve the following unconstrained variational model:

$$\mathbf{u}^* \in \underset{\mathbf{u}\in\mathbb{R}^n}{\arg\min}\ \frac{1}{2}\|\mathbf{SHu}-\mathbf{b}\|_2^2+\lambda R_{\ell_0}(\mathbf{u}), \tag{4.1}$$

or the following constrained variational model:

$$\mathbf{u}^* \in \underset{\mathbf{u}\in\mathbb{R}^n}{\arg\min}\ \frac{1}{2}\|\mathbf{SHu}-\mathbf{b}\|_2^2 \quad \text{s.t.} \quad R_{\ell_0}(\mathbf{u}) \leq c_{R_{\ell_0}} \in \mathbb{N}, \tag{4.2}$$

where $R_{\ell_0} : \mathbb{R}^n \to \mathbb{R}$ is the ℓ_0-based term promoting gradient-sparsity and the positive scalars λ and $c_{R_{\ell_0}}$ define the strength of the regularization.

Contribution

This chapter is mainly based on the publication [72] and on the preprint [73].
Inspired by the works [67, 68, 69, 70], in [72] we extend the inverse Potts models to SISR.
In particular, We consider two unconstrained variational models where an ℓ^0-gradient regularization term is considered both in a coupled (isotropic) and decoupled (anisotropic) form, the latter being better suited for directionally-biased images. To solve the model efficiently, we propose an ADMM algorithm which decomposes the original problem into substeps cheaply solved by means of direct hard-thresholding and standard iterative Conjugate Gradient (CG) linear solvers. Our variable splitting differs from the one introduced by Storath et al. in [67, 68, 69], where the non-convex substeps are solved by means either of approximate graph-cuts approaches [74] or dynamic programming algorithms. For the proposed ADMM algorithms, fixed-point convergence is proved. In the proposed methods, the most expensive step was shown to be the CG type solver required to solve the linear system arising in the quadratic substeps of the ADMM scheme which, in the case of large-scale data, can of course be a computational burden preventing the model to be used in practice. Therefore, in [73] under specific structural assumptions on the down-sampling operator $\mathbf{S}$, we make use of the strategy introduced in [75] where the authors show that the quadratic substep can be solved in closed-form by using Fourier-based techniques combined with the application of the Woodbury's formula. Analogously, we extend to the SISR problem the constrained ℓ_0-based formulation proposed in [71] for image denoising only. Also in this case, we consider the latter efficient ADMM algorithm for solving the constrained model.

4.1 On isotropic and anisotropic ℓ_0-gradient SR models

Before introducing the definition of the ℓ_0-gradient regularizer, we recall the definition of a partition.

Definition 4.1 (Partition)**.** Given $n, K \in \mathbb{N}$ such that $1 \leq K \leq N$. A K-partition of $\{1,\dots,n\}$ is the set $\Gamma = \{\Gamma_1,\dots,\Gamma_K\}$ of non-empty subsets Γ_i for $i = 1\dots K$, such that for all $\Gamma_i \in \Gamma$:

- $\Gamma_i \subset \{1,\dots,N\}$, for $i = 1\dots K$,
- $\Gamma_i \cap \Gamma_j = \emptyset$, for $i, j = 1\dots K$, with $i \neq j$,

- $\cup_{i=1}^{K}\Gamma_i = \{1,\dots,N\}$.

In the following, given a generic $\mathbf{x}\in\mathbb{R}^n$ and an element Γ_i of a K-partition Γ of $\{1,\dots,n\}$, we denote by $\mathbf{x}_{\Gamma_i}$ the subvector extracted from $\mathbf{x}$ whose entries are specified by the indexes in Γ_i. Then, we define the ℓ_0 functional with respect to the a K-partition Γ as the function $\|\cdot\|_0^\Gamma:\mathbb{R}^n\to\mathbb{R}_+$ such that:

$$\|\mathbf{x}\|_0^\Gamma := \sum_{i=1}^{K}|\|\mathbf{x}_{\Gamma_i}\||_0, \tag{4.3}$$

where $|\cdot|_0$ denotes the non-zero counting scalar function which is equal to 0 whenever the argument is zero and one otherwise and $\|\cdot\|$ stands for any ℓ_p norm, $p\geq 1$.
As a particular instance of (4.3), if $K=n$ and $\Gamma_i=\{i\}$ for all $i=1\dots n$, then the ℓ_0 functional counts the number of non-zero entries of the vector $\mathbf{x}\in\mathbb{R}^n$.

By choosing the standard Euclidead norm (i.e. $p=2$) in (4.3) and by recalling $\mathbf{D}\in\mathbb{R}^{2n\times n}$ denotes the discrete finite difference operator defined as the block matrix $(\mathbf{D}_h;\mathbf{D}_v)$, with $\mathbf{D}_h,\mathbf{D}_v\in\mathbb{R}^{n\times n}$ being the first order difference operator along the horizontal and vertical axes, respectively, we define the ℓ_0-gradient regularizer as the function $R_{\ell_0}:\mathbb{R}^n\to\mathbb{R}_+$ as:

$$R_{\ell_0}(\mathbf{u}) = \|\mathbf{D}\mathbf{u}\|_0^\Gamma. \tag{4.4}$$

According to the choice of Γ we can define our *isotropic* and *anistropic* ℓ_0-gradient regularizers.

Isotropic ℓ_0-gradient regularizer. By choosing Γ as a n-partition of $\{1,\dots,2n\}$ such that each $\Gamma_i=\{i,i+n\}$ for $i=1\dots n$, i.e. Γ_i is the set containing the indices corresponding to the vertical and horizontal differences at the i-th pixel, the function R_{ℓ_0} in (4.4) penalizes the number of jumps in $\mathbf{u}$ in terms of the non-zero values of its gradient magnitudes, jointly accounted for each pixel. In the following we denote by $R^{\mathrm{I}}_{\ell_0}$ the ℓ_0-gradient regularizer (4.4) upon the aforementioned choice for Γ and we refer to it as *isotropic* ℓ_0-gradient regularizer. Notice that $0\leq R^{\mathrm{I}}_{\ell_0}(\mathbf{x})\leq n$ for all $\mathbf{x}\in\mathbb{R}^n$.
To improve the readability of the following sections, when considering the isotropic regularization, by Γ we will always refer to the aforementioned partition. Thus, in this context, given a generic $\mathbf{x}\in\mathbb{R}^{2n}$ we will denote by $\mathbf{x}_{\Gamma_i}$ the subvector of $\mathbf{x}$ whose entries are $(\mathbf{x}_i,\mathbf{x}_{i+n})$ for $i=1\dots n$ and we will omit Γ when using the "norm-notation" of the ℓ_0 functional in (4.3). Hence, when referring to the function $R^{\mathrm{I}}_{\ell_0}$ we have:

$$R^{\mathrm{I}}_{\ell_0}(\mathbf{u}) := \|\mathbf{D}\mathbf{u}\|_0.$$

Anisotropic ℓ_0-gradient regularizer. By choosing Γ as a $2n$-partition $\{1,\dots,2n\}$ such that each $\Gamma_i=\{i\}$ for $i=1\dots 2n$, the function R_{ℓ_0} in (4.4) counts the number of non-zero components of the vector $\mathbf{D}\mathbf{u}$. In the following we denote by $R^{\mathrm{A}}_{\ell_0}$ the ℓ_0-gradient regularizer (4.4) upon the aforementioned choice for Γ and we refer to it as *anisotropic* ℓ_0-gradient regularizer. Notice that $0\leq R^{\mathrm{A}}_{\ell_0}(\mathbf{x})\leq n$, for all $\mathbf{x}\in\mathbb{R}^n$.
We will omit Γ when using the "norm-notation" in (4.3). Hence, when referring to the function $R^{\mathrm{A}}_{\ell_0}$ we have:

$$R^{\mathrm{A}}_{\ell_0}(\mathbf{u}) := \|\mathbf{D}\mathbf{u}\|_0 = \|\mathbf{D}_h\mathbf{u}\|_0 + \|\mathbf{D}_v\mathbf{u}\|_0,$$

where the second equality follows by the assumption on Γ and by the definition of $\mathbf{D}$.
For the sake of readability, in the anisotropic context, for each generic vector $\mathbf{x}$ the ℓ_0 functional in (4.3) will refer to a functional counting the non-zero entries of $\mathbf{x}$.

The isotropic ℓ_0-gradient unconstrained/constrained models

For solving (1.17) we introduce two $R^{\mathrm{I}}_{\ell_0}$-regularized SISR models, in an unconstrained and in a constrained fashion. Due to the non-convexity of the regularization under consideration, we remark that such formulations are indeed not equivalent, hence they deserve a separate discussion.

In [72] we have considered the unconstrained inverse Potts model for the problem of SISR. It computes solutions $\mathbf{u}^*$ as:

$$\mathbf{u}^* \in \underset{\mathbf{u}\in\mathbb{R}^n}{\arg\min}\ \frac{1}{2}\|\mathbf{SHu}-\mathbf{b}\|_2^2+\lambda\|\mathbf{Du}\|_0, \tag{4.5}$$

where the ℓ_2 fidelity term describes the presence of additive white Gaussian noise statistics and $\lambda > 0$ denotes the regularization parameter.

In [73] we have defined the analogous constrained model, inspired by the formulation proposed in [71] for simple image denoising problems (i.e. with no forward operator) and adapted here to the SISR task. It computes solutions $\mathbf{u}^*$ as:

$$\mathbf{u}^* \in \underset{\mathbf{u}\in\mathbb{R}^n}{\arg\min}\ \frac{1}{2}\|\mathbf{SHu}-\mathbf{b}\|_2^2 \quad \text{s.t.} \quad \|\mathbf{Du}\|_0 \leq \alpha \in \mathbb{N}. \tag{4.6}$$

The parameter α can be interpreted as the number of expected jumps in the desired solution. Choosing a proper value of α in (4.6) may be more practical than choosing λ in (4.5) whenever edge-maps specifying the number of edges for adjusting the flatness of the output are available. On the other hand, the choice of λ may be driven by standard *a-posteriori* parameter rules as the ones described in Chapter 3.

The anisotropic ℓ_0-gradient unconstrained model

In [72] we have further considered also the anistropic ℓ_0-gradient regularized model for SISR, which computes solutions $\mathbf{u}^*$ as:

$$\mathbf{u}^* \in \underset{\mathbf{u}\in\mathbb{R}^n}{\arg\min}\ \frac{1}{2}\|\mathbf{SHu}-\mathbf{b}\|_2^2+\lambda\left(\|\mathbf{D}_h\mathbf{u}\|_0+\|\mathbf{D}_v\mathbf{u}\|_0\right), \tag{4.7}$$

where we consider the same regularization parameter $\lambda > 0$ for both the gradient directions.

We stress that, by definition, both the isotropic and anisotropic ℓ_0-gradient regularizers penalize low-amplitude structures while preserving edges in the images, thus favouring sharp piecewise constant reconstructions which are particularly desirable for image segmentation problems. Thereore, the outcomes of the models (4.5), (4.6) and (4.7) are simplified, piecewise constant images which are amenable for precise IP. We finally remark the connection of these ℓ_0-gradient regularized models with the piecewise constant Mumford-Shah model, as analyzed, for example in [76].

4.2 ADMM optimization for the ℓ_0-gradient SR models

To numerically solve problems (4.5), (4.6) and (4.7) we propose to use the ADMM, a common optimization strategy which has been largely studied both in convex [77] and in non-convex [78] regimes. The ADMM iterations are defined in terms of a suitable variable splitting which differs from the one introduced by Storath et al. in [67, 68, 69] and allows us to efficiently solve the arising ADMM sub-problems either by standard fast solvers or via closed forms, as described in the following.

Variable splitting: the isotropic unconstrained case

Following [72], the optimization problem in (4.5) can be equivalently reformulated in terms of an auxiliary variable $\mathbf{z} \in \mathbb{R}^{2n}$ defined by $\mathbf{z} := \mathbf{D}\mathbf{u}$ as:

$$\begin{aligned} \underset{\mathbf{u}\in\mathbb{R}^n,\ \mathbf{z}\in\mathbb{R}^{2n}}{\arg\min}\ & \frac{1}{2}\|\mathbf{S}\mathbf{H}\mathbf{u}-\mathbf{b}\|_2^2 + \lambda\|\mathbf{z}\|_0 \\ \text{s.t.}\quad & \mathbf{z} = \mathbf{D}\mathbf{u}. \end{aligned} \tag{4.8}$$

By choosing an increasing sequence $\{\beta^k\}_{k\in\mathbb{N}}$ of the penalty parameter, for a given initialization $\mathbf{u}^0 \in \mathbb{R}^n$ and $\mathbf{z}^0 \in \mathbb{R}^{2n}$ we seek for minimisers of (4.5) by iterating the following scheme:

$$\begin{cases} \mathbf{z}^{k+1} & \in \underset{\mathbf{z}\in\mathbb{R}^{2n}}{\arg\min}\ \lambda\|\mathbf{z}\|_0 + \dfrac{\beta^k}{2}\|\mathbf{z} - (\mathbf{D}\mathbf{u}^k + \dfrac{\boldsymbol{\lambda}^k}{\beta^k})\|_2^2 \\ \mathbf{u}^{k+1} & = \underset{\mathbf{u}\in\mathbb{R}^n}{\arg\min}\ \dfrac{1}{2}\|\mathbf{S}\mathbf{H}\mathbf{u}-\mathbf{b}\|_2^2 + \dfrac{\beta^k}{2}\|\mathbf{D}\mathbf{u} - (\mathbf{z}^{k+1} - \dfrac{\boldsymbol{\lambda}^k}{\beta^k})\|_2^2 \\ \boldsymbol{\lambda}^{k+1} & = \boldsymbol{\lambda}^k - \beta^k(\mathbf{z}^{k+1} - \mathbf{D}\mathbf{u}^{k+1}). \end{cases} \tag{U-SISR-I}$$

In Theorem 4.3, under suitable assumptions on the gradient operator $\mathbf{D}$ and on the growth of $\{\beta^k\}_{k\in\mathbb{N}}$, we show a fixed-point convergence result for the sequence $\{\mathbf{u}_k\}_{k\in\mathbb{N}}$ generated. In particular, we will assume $\beta^k = k(1+\epsilon)^k, \epsilon > 0$ for each k (see Remark 4.4).

Variable splitting: the isotropic constrained case

Similarly, we can reformulate the constrained optimization problem (4.6) as:

$$\begin{aligned} \underset{\mathbf{u}\in\mathbb{R}^n,\ \mathbf{z}\in\mathbb{R}^{2n}}{\arg\min}\ & \frac{1}{2}\|\mathbf{S}\mathbf{H}\mathbf{u}-\mathbf{b}\|_2^2 + i_{\{\|\cdot\|_0\le\alpha\}}(\mathbf{z}) \\ \text{s.t.}\quad & \mathbf{z} := \mathbf{D}\mathbf{u}, \end{aligned} \tag{4.9}$$

where $i_{\{\|\cdot\|_0\le\alpha\}}(\cdot) : \mathbb{R}^{2n} \to \{0, +\infty\}$ denotes the indicator function of the non-convex set $\{\mathbf{z} \in \mathbb{R}^{2n} : \|\mathbf{z}\|_0 \le \alpha\}$. By following [71] and considering a sequence of increasing penalty parameters $\{\beta^k\}_{k\in\mathbb{N}}$, we thus seek for an approximation of an optimal solution of (4.9) by iterating the following scheme for initial $\mathbf{u}^0 \in \mathbb{R}^n$ and $\mathbf{z}^0 \in \mathbb{R}^{2n}$:

$$\begin{cases} \mathbf{z}^{k+1} & \in \underset{\mathbf{z}\in\mathbb{R}^{2n}}{\arg\min}\ i_{\{\|\cdot\|_0\le\alpha\}}(\mathbf{z}) + \dfrac{\beta^k}{2}\|\mathbf{z} - (\mathbf{D}\mathbf{u}^k + \dfrac{\boldsymbol{\lambda}^k}{\beta^k})\|_2^2 \\ \mathbf{u}^{k+1} & = \underset{\mathbf{u}\in\mathbb{R}^n}{\arg\min}\ \dfrac{1}{2}\|\mathbf{S}\mathbf{H}\mathbf{u}-\mathbf{b}\|_2^2 + \dfrac{\beta^k}{2}\|\mathbf{D}\mathbf{u} - (\mathbf{z}^{k+1} - \dfrac{\boldsymbol{\lambda}^k}{\beta^k})\|_2^2 \\ \boldsymbol{\lambda}^{k+1} & = \boldsymbol{\lambda}^k - \beta^k(\mathbf{z}^{k+1} - \mathbf{D}\mathbf{u}^{k+1}). \end{cases} \tag{C-SISR-I}$$

Variable splitting: the anisotropic unconstrained case

We can reformulate the unconstrained optimization problem (4.7) as:

$$\begin{aligned} \underset{\mathbf{u}\in\mathbb{R}^n,\ \mathbf{t},\mathbf{s}\in\mathbb{R}^n}{\arg\min} \quad & \frac{1}{2}\|\mathbf{SHu}-\mathbf{b}\|_2^2 + \lambda\left(\|\mathbf{t}\|_0+\|\mathbf{s}\|_0\right) && (4.10)\\ \text{s.t.} \quad & \mathbf{t}=\mathbf{D}_h\mathbf{u} \\ & \mathbf{s}=\mathbf{D}_v\mathbf{u}. && (4.11) \end{aligned}$$

In particular, for a given initialization $\mathbf{u}^0\in\mathbb{R}^n$, $\mathbf{t}^0\in\mathbb{R}^n$ and $\mathbf{s}^0\in\mathbb{R}^n$, under suitable assumptions on the gradient operator $\mathbf{D}$, a fixed-point convergence result (see Theorem 4.2) for the sequence $\{\mathbf{u}_k\}_{k\in\mathbb{N}}$ generated by the iteration:

$$\begin{cases} \mathbf{t}^{k+1} & \in \underset{\mathbf{t}\in\mathbb{R}^n}{\arg\min}\ \lambda\|\mathbf{t}\|_0+\dfrac{\beta_t^k}{2}\|\mathbf{t}-(\mathbf{D}_h\mathbf{u}^k+\dfrac{\boldsymbol{\lambda}^k}{\beta^k})\|_2^2 \\ \mathbf{s}^{k+1} & \in \underset{\mathbf{s}\in\mathbb{R}^n}{\arg\min}\ \lambda\|\mathbf{s}\|_0+\dfrac{\beta_s^k}{2}\|\mathbf{s}-(\mathbf{D}_v\mathbf{u}^k+\dfrac{\boldsymbol{\lambda}^k}{\beta^k})\|_2^2 \\ \mathbf{u}^{k+1} & = \underset{\mathbf{u}\in\mathbb{R}^n}{\arg\min}\ \dfrac{1}{2}\|\mathbf{SHu}-\mathbf{b}\|_2^2+\dfrac{\beta_t^k}{2}\|\mathbf{D}_h\mathbf{u}-(\mathbf{t}^{k+1}-\dfrac{\boldsymbol{\lambda}^k}{\beta_t^k})\|_2^2+ \\ & \qquad\qquad + \dfrac{\beta_s^k}{2}\|\mathbf{D}_v\mathbf{u}-(\mathbf{s}^{k+1}-\dfrac{\boldsymbol{\lambda}^k}{\beta_s^k})\|_2^2 \\ \boldsymbol{\lambda}_t^{k+1} & = \boldsymbol{\lambda}_t^k-\beta_t^k(\mathbf{t}^{k+1}-\mathbf{D}_h\mathbf{u}^{k+1}) \\ \boldsymbol{\lambda}_s^{k+1} & = \boldsymbol{\lambda}_s^k-\beta_s^k(\mathbf{s}^{k+1}-\mathbf{D}_v\mathbf{u}^{k+1}) \end{cases} \qquad \text{(U-SISR-A)}$$

is given for two increasing sequences of penalty parameters $\{\beta_t^k\}_{k\in\mathbb{N}}$ and $\{\beta_s^k\}_{k\in\mathbb{N}}$ such that $\beta_t^k=\beta_s^k=k(1+\epsilon)^k, \epsilon>0$ for each k.

We now provide more details on how to solve the different substeps for both ADMM schemes (U-SISR-I), (C-SISR-I) and (U-SISR-A).

Solving the ℓ_0 sub-steps

Due to the structure of the ℓ_0 functional defined in (4.3), the objective function corresponding to the $\mathbf{z}$-subproblem in (U-SISR-I) is separable. Hence, denoting by $\mathbf{v}^k$ the vector $\mathbf{Du}^k+\frac{\boldsymbol{\lambda}^k}{\beta^k}$ a solution $\mathbf{z}^{k+1}\in\mathbb{R}^{2n}$ can thus be computed by solving n 2D-optimization problems of the form:

$$\underset{\mathbf{x}\in\mathbb{R}^2}{\arg\min}\ \delta^k\|\mathbf{x}\|_0+\|\mathbf{x}-\mathbf{v}_{\Gamma_i}^k\|_2^2, \quad i=1\dots n, \tag{4.12}$$

where $\delta^k=\frac{2\lambda}{\beta^k}$. To solve (4.12), we apply the 2D hard-thresholding operator HT_{δ^k} to $\mathbf{v}^k$ as in [25]. Then, for $i=1\dots n$:

$$\mathbf{z}_{\Gamma_i}^{k+1}=\mathrm{HT}_{\delta^k}^2(\mathbf{v}_{\Gamma_i}^k)=\begin{cases}0 & \text{if } \|\mathbf{v}_{\Gamma_i}^k\|_2^2<2\delta^k,\\ \mathbf{v}_{\Gamma_i}^k, & \text{if } \|\mathbf{v}_{\Gamma_i}^k\|_2^2\geq 2\delta^k.\end{cases}$$

The steps for the solution of the ℓ_0 subproblem in (U-SISR-I) are summarized in Algorithm 2.

Concerning the constrained algorithm (C-SISR-I), a solution of the corresponding ℓ_0 substep associated to $\mathbf{z}$ can be computed by projecting $\mathbf{v}^k$ by following [71, Proposition 1]:

$$\mathbf{z}^{k+1}=\begin{cases}\mathbf{v}^k, & \text{if } \|\mathbf{v}^k\|_0\leq\alpha,\\ \tilde{\mathbf{v}}^k & \text{otherwise,}\end{cases}, \qquad \text{with} \qquad \tilde{\mathbf{v}}_{\Gamma_i}^k:=\begin{cases}\mathbf{v}_{\Gamma_i}^k & i\in\{(1),\dots,(\alpha)\}\\ 0 & i\in\{(\alpha+1),\dots,(n)\}\end{cases},$$

Algorithm 2 – ℓ_0 sub-step for (U-SISR-I)

input: $\mathbf{Du}^k \in \mathbb{R}^{2n}$, $\boldsymbol{\lambda}^k > 0$, $\beta^k > 0$, $\lambda > 0$
output: $\mathbf{z}^{k+1}$

1: $\mathbf{v}^k \leftarrow \mathbf{Du}^k + \frac{\lambda^k}{\beta^k}$
2: **for** $i = 1 \dots n$ **do**
3: $\quad \mathbf{z}_{\Gamma_i}^{k+1} \leftarrow \begin{cases} 0 & \text{if } ||\mathbf{v}_{\Gamma_i}^k||_2^2 < \frac{2\lambda}{\beta^k} \\ \mathbf{v}_{\Gamma_i}^k & \text{otherwise} \end{cases}$
4: **end for**

and the indexes $(1), \dots (n)$ are computed by sorting in descending order the ℓ_2 norms of the subvectors $\mathbf{v}_{\Gamma_i}^k$ for $i = 1 \dots n$, and relabelling them accordingly. In other words, $\mathbf{z}^{k+1}$ is computed by replacing by zero the $n - \alpha$ subvectors $\mathbf{v}_{\Gamma_i}^k$ of $\mathbf{v}^k$ with the smallest ℓ_2 norm.
The steps for the solution of the ℓ_0 subproblem in (C-SISR-I) case are summarized in Algorithm 3.

Algorithm 3 – ℓ_0 sub-step for (C-SISR-I)

input: $\mathbf{Du}^k \in \mathbb{R}^{2n}$, $\boldsymbol{\lambda}^k > 0$, $\beta^k > 0$, $\alpha \in \mathbb{N}$
output: $\mathbf{z}^{k+1}$

1: $\mathbf{v}^k \leftarrow \mathbf{Du}^k + \frac{\lambda^k}{\beta^k}$
2: Compute $\{(1), \dots, (n)\}$ by sorting the subvectors of $\mathbf{v}^k$ in descending order in terms of their ℓ_2 norm and compute $\tilde{\mathbf{v}}^k$.
3: Compute:

$$\mathbf{z}^{k+1} = \begin{cases} \mathbf{v}^k, & \text{if } \|\mathbf{v}^k\|_0 \leq \alpha \\ \tilde{\mathbf{v}}^k & \text{otherwise.} \end{cases}$$

Finally, due to decomposability of the ℓ^0 functional (4.3), the objective functions in (U-SISR-A) relative to the $\mathbf{t}$-subproblem and $\mathbf{s}$-subproblem are separable. Hence, this corresponds to solve the n 1D-optimization problems of the form:

$$\arg\min_{\mathbf{x}\in\mathbb{R}} \delta_t^k |\mathbf{x}|_0 + (\mathbf{x} - (\mathbf{f}_t^k)_i)_2^2, \quad i = 1 \dots n, \tag{4.13}$$

$$\arg\min_{\mathbf{x}\in\mathbb{R}} \delta_s^k |\mathbf{x}|_0 + (\mathbf{x} - (\mathbf{f}_s^k)_i)_2^2, \quad i = 1 \dots n, \tag{4.14}$$

where $\delta_t^k = \frac{2\lambda}{\beta_t^k}$, $\delta_s^k = \frac{2\lambda}{\beta_s^k}$ and $(\mathbf{f}_t^k)_i = (\mathbf{D}_h\mathbf{u}^k + \frac{\boldsymbol{\lambda}_t^k}{\beta_t^k})_i$, $(\mathbf{f}_s^k)_i = (\mathbf{D}_v\mathbf{u}^k + \frac{\boldsymbol{\lambda}_s^k}{\beta_s^k})_i$. Solving the problems (4.13) and (4.14) correspond to compute the proximal mapping of $|\cdot|_0$ with parameter δ_t^k and δ_s^k evaluated in $(\mathbf{f}_t^k)_i$ and $(\mathbf{f}_s^k)_i$, respectively, which is nothing but the 1D hard-thresholding operator [79]. Then:

$$\mathbf{t}_i^{k+1} = \mathrm{HT}_{\delta_t^k}^1((\mathbf{f}_t^k)_i) = \begin{cases} 0 & \text{if } |(\mathbf{f}_t^k)_i| < 2\delta_t^k \\ (\mathbf{f}_t^k)_i, & \text{if } |(\mathbf{f}_t^k)_i| \geq 2\delta_t^k \end{cases}$$

$$\mathbf{s}_i^{k+1} = \mathrm{HT}_{\delta_s^k}^1((\mathbf{f}_s^k)_i) = \begin{cases} 0 & \text{if } |(\mathbf{f}_s^k)_i| < 2\delta_s^k \\ (\mathbf{f}_s^k)_i, & \text{if } |(\mathbf{f}_s^k)_i| \geq 2\delta_s^k. \end{cases}$$

The steps for the solution of the ℓ_0 subproblems in (U-SISR-A) are summarized in Algorithm 4.

Algorithm 4 – ℓ_0 sub-step for (U-SISR-A)

input: $\mathbf{D}_h\mathbf{u}^k \in \mathbb{R}^n$,$\mathbf{D}_v\mathbf{u}^k \in \mathbb{R}^n$, $\boldsymbol{\lambda}_t^k > 0$,$\boldsymbol{\lambda}_s^k > 0$, $\beta_t^k > 0$,$\beta_s^k > 0$, $\lambda > 0$

output: $\mathbf{t}^{k+1}, \mathbf{s}^{k+1}$

$\mathbf{f}_t^k \leftarrow \mathbf{D}\mathbf{u}^k + \frac{\lambda^k}{\beta_t^k}$

$\mathbf{f}_s^k \leftarrow \mathbf{D}\mathbf{u}^k + \frac{\lambda^k}{\beta_s^k}$

for $i = 1 \dots n$ **do**

$$\mathbf{t}_i^{k+1} \leftarrow \begin{cases} 0 & \text{if } |(\mathbf{f}_t^k)_i| < \frac{2\lambda}{\beta_t^k} \\ (\mathbf{f}_t^k)_i & \text{otherwise} \end{cases}$$

$$\mathbf{s}_i^{k+1} \leftarrow \begin{cases} 0 & \text{if } |(\mathbf{f}_s^k)_i| < \frac{2\lambda}{\beta_s^k} \\ (\mathbf{f}_s^k)_i & \text{otherwise} \end{cases}$$

end for

Solving the ℓ_2-ℓ_2 sub-steps: exploiting the structure of H, S and D

We now consider the quadratic problems in (U-SISR-I), (C-SISR-I) and (U-SISR-A). The first order optimality conditions lead to the solution of large-size linear systems, whose coefficient matrix is symmetric and positive definite. As we have seen in Chapter 1, due to the presence of the downsampling operator $\mathbf{S}$, the use of the discrete Fourier transforms to solve the system is here not possible, as the product matrix $\mathbf{SH}$ does not have a block-circulant structure. To solve these quadratic problems efficiently, we have two alternatives:

- CG algorithm with a warm-start initialisation at every iteration.
- Closed form solution as in [75] provided suitable assumptions on $\mathbf{S}$.

We now discuss about the latter approach and we show how it can be implemented.

On the structure of H. By recalling section 1.3.2 we remark that due to the space-invariant assumption on the blur kernel, the matrix $\mathbf{H} \in \mathbb{R}^{n\times n}$ is a 2D convolution matrix. Assuming periodic boundary conditions, $\mathbf{H}$ thus takes the form of a BCCB, hence it can be easily diagonalized by the 2D discrete Fourier Transform, whose unitary matrix is denoted by $\mathbf{F} \in \mathbb{R}^{n\times n}$, as:

$$\mathbf{H} = \mathbf{F}^H\boldsymbol{\Lambda}\mathbf{F} \qquad \text{with} \qquad \mathbf{F}^H\mathbf{F} = \mathbf{F}\mathbf{F}^H = \mathbf{I}_n, \tag{4.15}$$

where $\boldsymbol{\Lambda} \in \mathbb{R}^{n\times n}$ is diagonal.

On the structure of S. We consider a down-sampling matrix $\mathbf{S} \in \mathbb{R}^{m\times n}$ in the form of a decimation operator removing selected rows and columns from the vectorized image it is applied to (see 1.3.2). Following [75], we assume in particular that the operator $\mathbf{S}^H \in \mathbb{R}^{n\times m}$ interpolates the decimated image with zeros. The matrix $\mathbf{S}$ is thus unstructured and, in particular, it cannot be diagonalized by the 2D discrete Fourier Transform. However, following [75] some considerations

can still be done to improve the computational efficiency. For an integer d, denoting by $\mathbf{J}_d \in \mathbb{R}^{d\times d}$ a matrix of all ones, by $\mathbf{1}_d$ the d-dimensional vector of ones and by $\mathbf{I}_d \in \mathbb{R}^{d\times d}$ the identity matrix, the following chain of identities holds:

$$\begin{aligned}\mathbf{F}\mathbf{S}^H\mathbf{S}\mathbf{F}^H :=& \frac{1}{L^2}\left(\mathbf{J}_L \otimes \mathbf{I}_{n_r}\right) \otimes \left(\mathbf{J}_L \otimes \mathbf{I}_{n_c}\right) && (4.16)\\ =& \frac{1}{L^2}\left(\mathbf{1}_L\mathbf{1}_L^T \otimes \mathbf{I}_{n_r}\mathbf{I}_{n_r}\right) \otimes \left(\mathbf{1}_L\mathbf{1}_L^T \otimes \mathbf{I}_{n_c}\mathbf{I}_{n_c}\right) && (4.17)\\ =& \frac{1}{L^2}\left(\left(\mathbf{1}_L \otimes \mathbf{I}_{n_r}\right)\left(\mathbf{1}_L^T \otimes \mathbf{I}_{n_r}\right)\right) \otimes \left(\left(\mathbf{1}_L \otimes \mathbf{I}_{n_c}\right)\left(\mathbf{1}_L^T \otimes \mathbf{I}_{n_c}\right)\right) && (4.18)\\ =& \frac{1}{L^2}\left(\left(\mathbf{1}_L \otimes \mathbf{I}_{n_r}\right) \otimes \left(\mathbf{1}_L \otimes \mathbf{I}_{n_c}\right)\right)\left(\mathbf{1}_L^T \otimes \mathbf{I}_{n_r}\right) \otimes \left(\mathbf{1}_L^T \otimes \mathbf{I}_{n_c}\right), && (4.19)\end{aligned}$$

where we recall that n_r and n_c are the number of rows and columns of the LR image such that $n_r \cdot n_c = m$ and $\otimes$ denotes the standard Kronecker product.

On the structure of D. As far as the operator $\mathbf{D}$ is concerned, assuming periodic boundary conditions, we have that both the matrices $\mathbf{D}_h$ and $\mathbf{D}_v$ are BCCB, hence they can be diagonalized by Fourier transforms as:

$$\mathbf{D}_h = \mathbf{F}^H\boldsymbol{\Sigma}_{\boldsymbol{h}}\mathbf{F} \quad \text{and} \quad \mathbf{D}_v = \mathbf{F}^H\boldsymbol{\Sigma}_{\boldsymbol{v}}\mathbf{F}, \tag{4.20}$$

with $\boldsymbol{\Sigma}_h \in \mathbb{R}^{n\times n}$ and $\boldsymbol{\Sigma}_v \in \mathbb{R}^{n\times n}$ diagonal matrices. As a consequence, the product $\mathbf{D}^H\mathbf{D} = \mathbf{D}_h^H\mathbf{D}_h + \mathbf{D}_v^H\mathbf{D}_v$, corresponding to the Laplace operator, can be expressed by:

$$\mathbf{D}^H\mathbf{D} = \mathbf{F}^H\left(\boldsymbol{\Sigma}_{\boldsymbol{h}}^H\boldsymbol{\Sigma}_{\boldsymbol{h}} + \boldsymbol{\Sigma}_{\boldsymbol{v}}^H\boldsymbol{\Sigma}_{\boldsymbol{v}}\right)\mathbf{F}. \tag{4.21}$$

That said, we now focus our attention on the solution of the ℓ_2-ℓ_2 subproblems required for the computation of $\mathbf{u}^{k+1}$ both in (U-SISR-I) and (C-SISR-I). For the sake of brevity we omit the discussion about the ℓ_2-ℓ_2 subproblem arising in (U-SISR-A). However we remark that in such a case we can proceed analogously.

By optimality, the desired $\mathbf{u}^{k+1}$ is the solution of the following linear system:

$$\left(\mathbf{H}^H\mathbf{S}^H\mathbf{S}\mathbf{H} + \beta^k\mathbf{D}^H\mathbf{D}\right)\mathbf{u}^{k+1} = \left(\mathbf{H}^H\mathbf{S}^H\mathbf{b} + \beta^k\mathbf{D}^H(\mathbf{z}^{k+1} - \frac{\boldsymbol{\lambda}^k}{\beta^k})\right). \tag{4.22}$$

To reduce the computational complexity of the linear system in (4.22) a direct solver is desirable. We remark that due to the structure of the decimation operator $\mathbf{S}$, the product $\mathbf{S}\mathbf{H}$ cannot be diagonalized in the frequency domain, thus preventing any direct computation of $\mathbf{u}^{k+1}$ in terms of fast Fourier solvers. However, following [75], upon the aforementioned considerations on the operators $\mathbf{H}$,$\mathbf{S}$ and $\mathbf{D}$ we can manipulate (4.22) in terms of $\mathbf{F}$ and $\mathbf{F}^H$ to deduce the following chain of equalities:

$$\begin{aligned}\left(\mathbf{F}^H\boldsymbol{\Lambda}^H\mathbf{F}\mathbf{S}^H\mathbf{S}\mathbf{F}^H\boldsymbol{\Lambda}\mathbf{F} + \beta^k\mathbf{F}^H\left(\boldsymbol{\Sigma}_{\boldsymbol{h}}^H\boldsymbol{\Sigma}_{\boldsymbol{h}} + \boldsymbol{\Sigma}_{\boldsymbol{v}}^H\boldsymbol{\Sigma}_{\boldsymbol{v}}\right)\mathbf{F}\right)\mathbf{u}_{k+1} &= \mathbf{r}^k && (4.23)\\ \left(\mathbf{F}^H\boldsymbol{\Lambda}^H\frac{1}{L^2}\left(\mathbf{J}_L \otimes \mathbf{I}_{n_r}\right) \otimes \left(\mathbf{J}_L \otimes \mathbf{I}_{n_c}\right)\boldsymbol{\Lambda}\mathbf{F} + \beta^k\mathbf{F}^H\left(\boldsymbol{\Sigma}_{\boldsymbol{h}}^H\boldsymbol{\Sigma}_{\boldsymbol{h}} + \boldsymbol{\Sigma}_{\boldsymbol{v}}^H\boldsymbol{\Sigma}_{\boldsymbol{v}}\right)\mathbf{F}\right)\mathbf{u}_{k+1} &= \mathbf{r}^k && (4.24)\\ \left(\boldsymbol{\Lambda}^H\frac{1}{L^2}\left(\mathbf{J}_L \otimes \mathbf{I}_{n_r}\right) \otimes \left(\mathbf{J}_L \otimes \mathbf{I}_{n_c}\right)\boldsymbol{\Lambda}\mathbf{F} + \beta^k\left(\boldsymbol{\Sigma}_{\boldsymbol{h}}^H\boldsymbol{\Sigma}_{\boldsymbol{h}} + \boldsymbol{\Sigma}_{\boldsymbol{v}}^H\boldsymbol{\Sigma}_{\boldsymbol{v}}\right)\mathbf{F}\right)\mathbf{u}_{k+1} &= \mathbf{F}\mathbf{r}^k && (4.25)\end{aligned}$$

$$\left(\frac{1}{L^2}\underline{\mathbf{\Lambda}}^H\underline{\mathbf{\Lambda}} + \beta^k\mathbf{\Sigma}_{\boldsymbol{h}}^H\mathbf{\Sigma}_{\boldsymbol{h}} + \mathbf{\Sigma}_{\boldsymbol{v}}^H\mathbf{\Sigma}_{\boldsymbol{v}}\right)\mathbf{F}\mathbf{u}_{k+1} = \mathbf{F}\mathbf{r}^k, \tag{4.26}$$

where we define:

$$\mathbf{r}^k := \mathbf{H}^H\mathbf{S}^H\mathbf{b} + \beta^k\mathbf{D}^H\left(\mathbf{z}^{k+1} - \frac{\boldsymbol{\lambda}^k}{\beta^k}\right), \qquad \underline{\mathbf{\Lambda}} := \left(\mathbf{1}_L^T \otimes \mathbf{I}_{n_r}\right) \otimes \left(\mathbf{1}_L^T \otimes \mathbf{I}_{n_c}\right)\mathbf{\Lambda}. \tag{4.27}$$

From (4.26), we can thus deduce:

$$\mathbf{u}^{k+1} = \mathbf{F}^H\left(\frac{1}{L^2}\underline{\mathbf{\Lambda}}^H\underline{\mathbf{\Lambda}} + \beta^k\mathbf{\Sigma}_{\boldsymbol{h}}^H\mathbf{\Sigma}_{\boldsymbol{h}} + \mathbf{\Sigma}_{\boldsymbol{v}}^H\mathbf{\Sigma}_{\boldsymbol{v}}\right)^{-1}\mathbf{F}\mathbf{r}^k. \tag{4.28}$$

By using now the Woodbury formula [80] to determine the expression of $\mathbf{u}_{k+1}$ we have:

$$\mathbf{u}^{k+1} = \frac{1}{\beta^k}\mathbf{F}^H\mathbf{\Psi}\mathbf{F}\mathbf{r}^k - \frac{1}{\beta^k}\mathbf{F}^H\mathbf{\Psi}\underline{\mathbf{\Lambda}}^H\left(\beta^k L^2\mathbf{I}_m + \underline{\mathbf{\Lambda}}\mathbf{\Psi}\underline{\mathbf{\Lambda}}^H\right)^{-1}\underline{\mathbf{\Lambda}}^H\mathbf{\Psi}\mathbf{F}\mathbf{r}^k, \tag{4.29}$$

where $\mathbf{\Psi} := \mathbf{F}\left(\mathbf{D}^H\mathbf{D}\right)^{-1}\mathbf{F}^H$. In order to overcome the fact that the discrete Laplace operator $\mathbf{D}^H\mathbf{D}$ may not invertible, we can follow [75] and add a regularization term $\sigma_{\boldsymbol{\mathcal{L}}}\|\mathbf{u}\|_2^2$ depending on a small constant $0 < \sigma_{\boldsymbol{\mathcal{L}}} \ll 1$ to make the operator $\mathbf{\Psi}$ invertible and the iteration (4.29) well-defined, so that $\mathbf{\Psi}_{\sigma_{\boldsymbol{\mathcal{L}}}} := \left(\mathbf{\Sigma}_h^H\mathbf{\Sigma}_h + \mathbf{\Sigma}_v^H\mathbf{\Sigma}_v + \sigma_{\boldsymbol{\mathcal{L}}}\mathbf{I}_n\right)^{-1}$.
Expression (4.29) provides now an efficient formula to compute at each $k \geq 1$ the quantity $\mathbf{u}^{k+1}$ since it only requires the inversion of diagonal matrices through standard FFT evaluations. We remark that such update is not possible for a general down-sampling operator $\mathbf{S}$ (such as, e.g., the Lanczos interpolation operator, often employed in the context of SISR problems), as the chain of equalities (4.16)-(4.19) is no longer true. Algorithm 5 reports the main step for the solution of the ℓ_2-ℓ_2 subproblem (4.15).

Algorithm 5 – Fast solution of ℓ_2-ℓ_2 problems

input: $\mathbf{b} \in \mathbb{R}^m$, $\mathbf{S} \in \mathbb{R}^{m\times n}$, $\mathbf{H} \in \mathbb{R}^{n\times n}$, $\mathbf{D}_h \in \mathbb{R}^{n\times n}$, $\mathbf{D}_v \in \mathbb{R}^{n\times n}$, $\mathbf{z}^{k+1} \in \mathbb{R}^{2n}$, $\beta^k > 0, 0 < \sigma_{\boldsymbol{\mathcal{L}}} \ll 1$
output: $\mathbf{u}^{k+1}$

1: $\mathbf{H} = \mathbf{F}^H\mathbf{\Lambda}\mathbf{F}$
2: $\mathbf{D}_h = \mathbf{F}^H\mathbf{\Sigma}_{\boldsymbol{h}}\mathbf{F}$
3: $\mathbf{D}_v = \mathbf{F}^H\mathbf{\Sigma}_{\boldsymbol{v}}\mathbf{F}$
% Compute $\underline{\mathbf{\Lambda}}$ and $\mathbf{\Psi}$
4: $\underline{\mathbf{\Lambda}} \leftarrow \left(\mathbf{1}_L^T \otimes \mathbf{I}_{n_r}\right) \otimes \left(\mathbf{1}_L^T \otimes \mathbf{I}_{n_c}\right)\mathbf{\Lambda}$
5: $\mathbf{\Psi} \leftarrow \left(\mathbf{\Sigma}_x^H\mathbf{\Sigma}_x + \mathbf{\Sigma}_y^H\mathbf{\Sigma}_y + \sigma_{\boldsymbol{\mathcal{L}}}\mathbf{I}_n\right)^{-1}$
% Compute the solution of the linear system
6: $\mathbf{r}^k \leftarrow \mathbf{H}^H\mathbf{S}^H\mathbf{b} + \beta^k\mathbf{D}^H(\mathbf{z}^{k+1} - \frac{\boldsymbol{\lambda}^k}{\beta^k})$
7: $\mathbf{u}^{k+1} \leftarrow \frac{1}{\beta^k}\mathbf{F}^H\mathbf{\Psi}\mathbf{F}\mathbf{r}^k - \frac{1}{\beta^k}\mathbf{F}^H\mathbf{\Psi}\underline{\mathbf{\Lambda}}^H\left(\beta^k L^2\mathbf{I}_m + \underline{\mathbf{\Lambda}}\mathbf{\Psi}\underline{\mathbf{\Lambda}}^H\right)^{-1}\underline{\mathbf{\Lambda}}^H\mathbf{\Psi}\mathbf{F}\mathbf{r}^k$.

4.3 Convergence analysis

In [72] we prove a fixed-point convergence theorem for the iterative schemes (U-SISR-A) and (U-SISR-I). We first recall Theorem 4.1 addressing the existence of minimizers for the optimization problems (4.7) and (4.5).

Theorem 4.1. *The solution sets of both the anisotropic* (4.7) *and isotropic* (4.7) *unconstrained problems are non-empty.*

Proof. The proof can be found in [67, Theorem 1], in the case of a general forward operator $\mathbf{A}$. □

Theorem 4.2. *Let the ADMM iterations* (U-SISR-A) *be defined under the following conditions:*

A.1 $\{\beta_t^k\}_{k\in\mathbb{N}}$,$\{\beta_s^k\}_{k\in\mathbb{N}}$ *are increasing sequences such that* $\sum_{k=1}^{+\infty}\sqrt{\frac{k}{\beta_t^k}} < +\infty$, $\sum_{k=1}^{+\infty}\sqrt{\frac{k}{\beta_s^k}} < +\infty$ *and* $\frac{\beta_s^k}{\beta_t^k} \to c \neq 0$.

A.2 $\mathbf{D}_h$ *and* $\mathbf{D}_v$ *are full rank.*

Then, the sequences $\{\mathbf{t}^k\}_{k\in\mathbb{N}}, \{\mathbf{s}^k\}_{k\in\mathbb{N}}, \{\mathbf{u}^k\}_{k\in\mathbb{N}}$ *converge, i.e.:*

$$\mathbf{t}^k \longrightarrow \mathbf{t}^*, \quad \mathbf{s}^k \longrightarrow \mathbf{s}^*, \quad \mathbf{u}^k \longrightarrow \mathbf{u}^*,$$

with $\mathbf{t}^* = \mathbf{D}_h\mathbf{u}^*$ *and* $\mathbf{s}^* = \mathbf{D}_v\mathbf{u}^*$.

Proof. We consider the ADMM sequences $\{\mathbf{u}^k\}_{k\in\mathbb{N}}, \{\mathbf{t}^k\}_{k\in\mathbb{N}}, \{\mathbf{s}^k\}_{k\in\mathbb{N}}$, defined for (U-SISR-A). We want to show that there exists $\mathbf{u}^*$ such that:

$$\mathbf{u}^k \to \mathbf{u}^*, \quad \mathbf{t}^k \to \mathbf{D}_h\mathbf{u}^*, \quad \mathbf{s}^k \to \mathbf{D}_v\mathbf{u}^*.$$

To shorten the proof, we remark that everything proved for the sequences $\{\mathbf{t}^k\}_{k\in\mathbb{N}}$,$\{\beta_t^k\}_{k\in\mathbb{N}}$, $\{\boldsymbol{\lambda}_t^k\}_{k\in\mathbb{N}}$ and $\{\mathbf{D}_h\mathbf{u}^k\}_{k\in\mathbb{N}}$ can be deduced for $\{\mathbf{s}^k\}_{k\in\mathbb{N}}$, $\{\beta_s^k\}_{k\in\mathbb{N}}$, $\{\boldsymbol{\lambda}_s^k\}_{k\in\mathbb{N}}$ and $\{\mathbf{D}_v\mathbf{u}^k\}_{k\in\mathbb{N}}$ in the same way.

We start defining the following functionals:

$$G_k^h(\mathbf{t}) := \lambda\|\mathbf{t}\|_0 + \frac{\beta_t^k}{2}\|\mathbf{t} - \left(\mathbf{D}_h\mathbf{u}^k + \frac{\boldsymbol{\lambda}_t^k}{\beta_t^k}\right)\|_2^2,$$
$$F_k(\mathbf{u}) := \frac{1}{2}\|\mathbf{S}\mathbf{H}\mathbf{u} - \mathbf{b}\|_2^2 + \frac{\beta_t^k}{2}\|\mathbf{D}_h\mathbf{u} - \left(\mathbf{t}^{k+1} - \frac{\boldsymbol{\lambda}_t^k}{\beta_t^k}\right)\|_2^2 + \frac{\beta_s^k}{2}\|\mathbf{D}_v\mathbf{u} - \left(\mathbf{s}^{k+1} - \frac{\boldsymbol{\lambda}_s^k}{\beta_s^k}\right)\|_2^2.$$

Step 1 There holds:

$$\|\mathbf{t}^{k+1} - \mathbf{D}_h\mathbf{u}^k - \frac{\boldsymbol{\lambda}_t^k}{\beta_t^k}\|_2 \leq \sqrt{\frac{2\lambda n}{\beta_t^k}}. \tag{4.30}$$

This inequality can be trivially shown by the minimality of $\mathbf{t}^{k+1}$ for the ℓ_0 subproblem in (U-SISR-A) which entails $G_k^h(\mathbf{t}^{k+1}) \leq G_k^h(\mathbf{D}_h\mathbf{u}^k + \frac{\boldsymbol{\lambda}_t^k}{\beta_t^k})$, therefore we get:

$$\lambda\|\mathbf{t}^{k+1}\|_0 + \frac{\beta_t^k}{2}\|\mathbf{t}^{k+1} - (\mathbf{D}_h\mathbf{u}^k + \frac{\boldsymbol{\lambda}_t^k}{\beta_t^k})\|_2^2 \leq \lambda\|\mathbf{D}_h\mathbf{u}^k + \frac{\boldsymbol{\lambda}_t^k}{\beta_t^k}\|_0 \leq \lambda n,$$

by definition of $\|\cdot\|_0$ in (4.3), where we recall n is the dimension of the vector $\mathbf{u}^k$. By neglecting the first term on the Left Hand Side (LHS) of the above inequality, we deduce (4.30).

Step 2 From the minimality of $\mathbf{u}^{k+1}$ in square problem of (U-SISR-A) we have: $F_k(\mathbf{u}^{k+1}) \leq F_k(\mathbf{u}^k)$ for every k. By definition of F_k and applying (4.30) and its analogous related to the sequences $\{\mathbf{s}^k\}_{k\in\mathbb{N}}$, $\{\beta_s^k\}_{k\in\mathbb{N}}$, $\{\boldsymbol{\lambda}_s^k\}_{k\in\mathbb{N}}$ and $\{\mathbf{D}_v\mathbf{u}^k\}_{k\in\mathbb{N}}$, we deduce:

$$\frac{1}{2}\|\mathbf{S}\mathbf{H}\mathbf{u}^{k+1}-\mathbf{b}\|_2^2+\frac{\beta_t^k}{2}\|\mathbf{D}_h\mathbf{u}^{k+1}-\mathbf{t}^{k+1}+\frac{\boldsymbol{\lambda}_t^k}{\beta_t^k}\|_2^2+\frac{\beta_s^k}{2}\|\mathbf{D}_v\mathbf{u}^{k+1}-\mathbf{s}^{k+1}+\frac{\boldsymbol{\lambda}_s^k}{\beta_s^k}\|_2^2 \leq \leq \frac{1}{2}\|\mathbf{S}\mathbf{H}\mathbf{u}^{k}-\mathbf{b}\|_2^2+2\lambda n. \tag{4.31}$$

Since the all the terms on the LHS of (4.31) are nonnegative, the following inequality holds:

$$\frac{1}{2}\|\mathbf{S}\mathbf{H}\mathbf{u}^{k+1}-\mathbf{b}\|_2^2 \leq \frac{1}{2}\|\mathbf{S}\mathbf{H}\mathbf{u}^{k}-\mathbf{b}\|_2^2+2\lambda n \leq \ldots \leq \frac{1}{2}\|\mathbf{S}\mathbf{H}\mathbf{u}^{0}-\mathbf{b}\|_2^2+2\lambda n k.$$

From (4.31) and by the sub-additivity property of the square root we can also derive the following inequality:

$$\|\mathbf{D}_h\mathbf{u}^{k+1}-\mathbf{t}^{k+1}+\frac{\boldsymbol{\lambda}_t^k}{\beta_t^k}\|_2 \leq \sqrt{\frac{1}{\beta_t^k}}\|\mathbf{S}\mathbf{H}\mathbf{u}^{0}-\mathbf{b}\|_2+\sqrt{4\lambda n\frac{k}{\beta_t^k}}. \tag{4.32}$$

Step 3 We show that the sequences $\{\mathbf{D}_h\mathbf{u}^k\}_{k\in\mathbb{N}}$ and $\{\mathbf{D}_v\mathbf{u}^k\}_{k\in\mathbb{N}}$ are Cauchy sequences, hence they converge. We prove this for $\{\mathbf{D}_h\mathbf{u}^k\}_{k\in\mathbb{N}}$, the proof for $\{\mathbf{D}_v\mathbf{u}^k\}_{k\in\mathbb{N}}$ is identical.

$$\|\mathbf{D}_h\mathbf{u}^{k+1}-\mathbf{D}_h\mathbf{u}^{k}\|_2 \leq \|\mathbf{D}_h\mathbf{u}^{k+1}-\mathbf{t}^{k+1}+\frac{\boldsymbol{\lambda}_t^k}{\beta_t^k}\|_2+\|\mathbf{D}_h\mathbf{u}^{k}-\mathbf{t}^{k+1}+\frac{\boldsymbol{\lambda}_t^k}{\beta_t^k}\|_2.$$

By assumption **A.1** applied on the RHS of (4.32) we deduce:

$$\|\mathbf{D}_h\mathbf{u}^{k+1}-\mathbf{t}^{k+1}+\frac{\boldsymbol{\lambda}_t^k}{\beta_t^k}\|_2 \to 0, \tag{4.33}$$

which, combined with (4.30) entails that $\{\mathbf{D}_h\mathbf{u}^k\}_{k\in\mathbb{N}}$ is a Cauchy sequence. Hence it converges to a point $\mathbf{t}^*$. Similarly, $\{\mathbf{D}_v\mathbf{s}^k\}_{k\in\mathbb{N}}$ converges to a point $\mathbf{s}^*$.

Step 4 We prove now the convergence to the same point of the sequences $\{\mathbf{t}^k\}_{k\in\mathbb{N}}$ and $\{\mathbf{D}_h\mathbf{u}^k\}_{k\in\mathbb{N}}$. By writing the Lagrangian parameter formulas (U-SISR-A) as:

$$\frac{\boldsymbol{\lambda}_t^{k+1}}{\beta_t^k}=\mathbf{D}_h\mathbf{u}^{k+1}-\mathbf{t}^{k+1}+\frac{\boldsymbol{\lambda}_t^k}{\beta_t^k}, \tag{4.34}$$

and from (4.33) we deduce that $\frac{\|\boldsymbol{\lambda}_t^{k+1}\|_2}{\beta_t^k} \to 0$. By monotonicity of the $\{\beta_t^k\}_{k\in\mathbb{N}}$ we then deduce that $\frac{\|\boldsymbol{\lambda}_t^{k}\|_2}{\beta_t^k} \to 0$. Hence:

$$\|\mathbf{D}_h\mathbf{u}^{k+1}-\mathbf{t}^{k+1}\|_2 \leq \frac{\|\boldsymbol{\lambda}_t^{k+1}\|_2+\|\boldsymbol{\lambda}_t^{k}\|_2}{\beta_t^k},$$

where both quantities on the RHS tend to 0 as $k\to\infty$. Therefore, by the uniqueness of the limit, $\mathbf{t}^k \longrightarrow \mathbf{t}^*$ and $\mathbf{D}_h\mathbf{u}^k \longrightarrow \mathbf{t}^*$. Similarly, $\mathbf{s}^k \longrightarrow \mathbf{s}^*$ and $\mathbf{D}_v\mathbf{u}^k \longrightarrow \mathbf{s}^*$.

Step 5 We can now prove convergence of the sequence $\{\mathbf{u}^k\}_{k\in\mathbb{N}}$. For simplicity, let us define the quantities $\mathbf{A} := \mathbf{SH}$ and $\mathbf{M}_k := \frac{1}{\beta_t^k}\mathbf{A}^T\mathbf{A} + \mathbf{D}_h^T\mathbf{D}_h + \frac{\beta_s^k}{\beta_t^k}\mathbf{D}_v^T\mathbf{D}_v$, for every k. By **A.2**, we observe that the matrix $\mathbf{M}_k$ is invertible for all k and that the optimality condition of the square problem in (U-SISR-A) reads:

$$\mathbf{M}_k\mathbf{u}^k = \mathbf{D}_h^T(\mathbf{t}^{k+1} - \frac{\boldsymbol{\lambda}_t^k}{\beta^k}) + \frac{\beta_s^k}{\beta_t^k}\mathbf{D}_v^T(\mathbf{s}^{k+1} - \frac{\boldsymbol{\lambda}_s^k}{\beta^k}) + \frac{1}{\beta_t^k}\mathbf{A}^T\mathbf{b}.$$

Since $\mathbf{t}^{k+1} \to \mathbf{t}^*$, $\mathbf{s}^{k+1} \to \mathbf{s}^*$, $\frac{\boldsymbol{\lambda}_t^k}{\beta_t^k} \to 0$, $\frac{\boldsymbol{\lambda}_s^k}{\beta_s^k} \to 0$, and by Assumptions **A.1** and **A.2**, we have that $\frac{1}{\beta_t^k}\mathbf{A}^T\mathbf{b} \to \mathbf{0}$ so that the RHS converges pointwise to $\mathbf{z}^* = \mathbf{D}_h^T t^* + c\mathbf{D}_v^T s^*$. Additionally, the sequence $\mathbf{M}_k^{-1}$ converges pointwise to $\mathbf{M}^*$. We thus have that $\mathbf{u}^k = \mathbf{M}_k^{-1}\mathbf{M}_k\mathbf{u}^k \to \mathbf{M}^*\mathbf{z}^* := \mathbf{u}^*$. We now want to show that $\mathbf{t}^* = \mathbf{D}_h\mathbf{u}^*$ and, similarly, that $\mathbf{s}^* = \mathbf{D}_v\mathbf{u}^*$. We show the details only for the former case. By the triangle inequality we get:

$$\|\mathbf{t}^* - \mathbf{D}_h\mathbf{u}^*\|_2 \leq \|\mathbf{t}^* - \mathbf{D}_h\mathbf{u}^k\|_2 + \|\mathbf{D}_h\mathbf{u}^k - \mathbf{D}_h\mathbf{u}^*\|_2 \leq \|\mathbf{t}^* - \mathbf{D}_h\mathbf{u}^k\|_2 + \|\mathbf{D}_h\|_2\|\mathbf{u}^k - \mathbf{u}^*\|_2,$$

where both terms tend to $\mathbf{0}$ since $\mathbf{D}_h\mathbf{u}^k \to \mathbf{t}^*$ and $\mathbf{u}^k \to \mathbf{u}^*$. □

Theorem 4.3. *Let the ADMM iterations* (U-SISR-I) *be defined under the following conditions:*

I.1 $\{\beta^k\}_{k\in\mathbb{N}}$ *is an increasing sequence such that* $\sum_k^{+\infty}\sqrt{\frac{k}{\beta^k}} < +\infty$

I.2 $\mathbf{D}$ *is full rank.*

Then, $\{\mathbf{z}^k\}_{k\in\mathbb{N}} \longrightarrow \mathbf{z}^*$, $\{\mathbf{u}^k\}_{k\in\mathbb{N}} \longrightarrow \mathbf{u}^*$ *and* $\mathbf{z}^* = \mathbf{D}\mathbf{u}^*$.

Proof. The proof of Theorem (4.3) follows the same steps as the previous one. The only main difference in it is the definiton of $\mathbf{M}_k$, which reads in this case:

$$\mathbf{M}_k := \frac{1}{\beta^k}\mathbf{A}^T\mathbf{A} + \mathbf{D}^T\mathbf{D} = \frac{1}{\beta^k}\mathbf{A}^T\mathbf{A} + \mathbf{D}_h^T\mathbf{D}_h + \mathbf{D}_v^T\mathbf{D}_v.$$

By proceeding similarly as above the conclusion holds. □

Remark 4.4. We remark that the full rank assumption on the operators $\mathbf{D}_h$ and $\mathbf{D}_v$ is verified, for instance, if Dirichlet boundary conditions are assumed. Thus, an artificial image zero-padding of the image can be considered. As far as the growth condition on the penalty parameters is concerned, A sufficient condition which guarantees the required growth of the penalty sequences is $\beta_t^k = \beta_s^k = O(k(1+\epsilon)^k)$, $0 < \epsilon \ll 1$. We remark that in [67] a geometric growth was assumed. Unfortunately, this is not enough for our theoretical convergence result to hold. However, our numerical experiments, however, show numerical convergence even when periodic boundary conditions and less severe growth conditions on the penalty sequences are used. A theoretical convergence proof in this case is left for future research as well as a convergence analysis for the constrained scheme (C-SISR-I).

4.4 Numerical results

In this section, we evaluate the performances of the our unconstrained isotropic and anisotropic ℓ_0-gradient models. In particular, we refer to the implementations solving the unconstrained isotropic ℓ_0-gradient model (4.5) as TV0I, to the unconstrained anisotropic model in (4.7) as TV0A and to the constrained isotropic ℓ_0-gradient model in (4.6) as c-TV0I. We superscript "CG" or "F" to these terms, whether the implementation solves the ℓ_2-ℓ_2 substep by means of CG algorithm or the FFT-based formula inspired by [75].

Initialisation, parameters and evaluation metrics. We initialise $\mathbf{u}^0$ in our models as $\mathbf{u}^0 = \mathbf{S}^T\mathbf{b}$. Given the non-convexity of problem (4.5), (4.7) and (4.6) the choice of a wise initialisation is important. However, in [73] we tested several ones, namely the aforementioned one, the zero image and the I-TV initialisations and we have observed all the methods look stable w.r.t. this choice. The variables $\mathbf{t}^0, \mathbf{s}^0, \mathbf{z}^0$ as well as $\boldsymbol{\lambda}_t^0, \boldsymbol{\lambda}_s^0, \boldsymbol{\lambda}^0$ in (U-SISR-I), (C-SISR-I) and (U-SISR-A) are set to $\mathbf{0}$. To ensure the convergence results provided by Theorems 4.2 and 4.3, the penalty sequences are chosen such that $\beta^k = k(1+\epsilon)^k$ with $\epsilon = 10^{-4}$. Note that for such small choice of ϵ, $k(1+\epsilon)^k \approx k$, i.e. the growth of $\{\beta^k\}_{k\in\mathbb{N}}$ is almost linear. The process is stopped when the relative change between consecutive iterates $\mathbf{u}^k$ is lower than 10^{-3}. For simulated data, we evaluate the quality of the SR outputs by means of Peak-Signal-to-Noise-Ratio (PSNR) and Structure Similarity index (SSIM) as well as the Jaccard index (Jac), an evaluation metric in the range $[0, 1]$ measuring the ratio between correctly detected points and false detections. While PSNR and SSIM are good choices to quantify reconstruction quality, the Jaccard index is more appropriate for segmentation purposes as it assesses correct versus false pixel localisation.

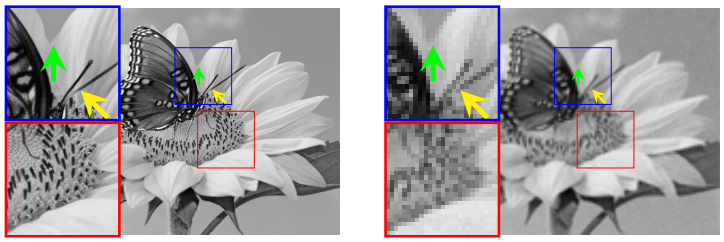
Comparisons. We compare our methods with several variational and learning methods. More precisely, we consider gradient-sparse regularization models such as convex isotropic TV (I-TV) [81], non-convex capped TV (c-TV) [82] and anisotropic fractional TV [83] (A-TV$^{1/2}$) which, for consistency, have been implemented within the same ADMM optimisation framework. Finally, we add comparisons with the results obtained by two state-of-the-art Deep Learning-based approaches. The former is the Content Adaptive Resampler (CAR) [84] convolutional neural network, which is characterised by a downsampler-upsampler structure. For that, we use a pre-trained model taking into account only the trained upsampler part. The latter is the Image Restoration Convolutional Neural Network (IRCNN) [85], which is a Plug-and-Play method based on Half Quadratic Splitting optimization.

Convergence analysis on synthetic data and computational cost comparisons

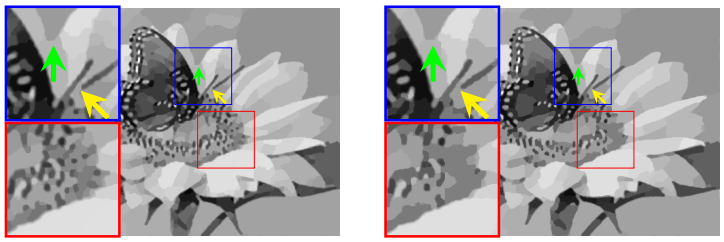
Based on 4.2 and 4.3, we validate the convergence properties of the proposed unconstrained isotropic and anisotropic ℓ_0-gradient ADMM algorithms and comment on their parameter sensitivity.

For this first example, LR data were generated by applying (1.17) to the HR 428×600 grayscale image in Figure 4.1a. Namely, Gaussian blur with $\sigma_{\mathbf{H}} = 1$, $L = 4$ down-sampling and AGWN with standard deviation $\sigma_{\mathbf{e}} = 0.01$ were applied to get the LR image in Figure 4.1b. In Figure 4.1c and 4.1d we report the results computed by TV0I$^{\text{CG}}$ for two different values of the regularization parameter $\lambda \in \{0.005, 0.01\}$. As expected, the jump-sparse regularization flattens out many de-

tails in the reconstruction, promoting a cartoon-like reconstruction: the higher the regularization parameter λ, the more simplified the reconstruction. We further add a close-up of two ROIs: the blue square contains both fine details (filaments, yellow arrows) and corner points (green arrows), the red one textured details. We report in the captions of Figure 4.1c and 4.1d the values $R^{\mathrm{I}}_{\ell_0}(\mathbf{u}^*)$ which correspond to the number of gradient jumps on the output image. Clearly, choosing a larger λ promotes more jump-sparsity, so that the number of jumps of $\mathbf{u}^*$ is smaller.



(a) HR (b) LR (x4)

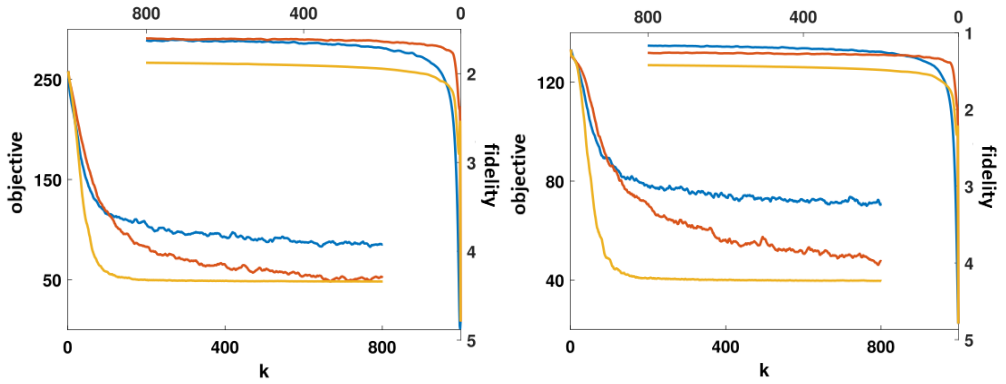


(c) $R^{\mathrm{I}}_{\ell_0}(\mathbf{u}^*)$=24067, $\lambda = 0.005$ (d) $R^{\mathrm{I}}_{\ell_0}(\mathbf{u}^*)$=18547, $\lambda = 0.01$

Figure 4.1: Results obtained for $\lambda \in \{0.005, 0.01\}$ by TV0I$^{\mathrm{CG}}$ on a synthetic image.

We then validate the algorithmic convergence behaviour w.r.t. the choice of the penalty sequences $\{\beta_t^k\}_{k\in\mathbb{N}}, \{\beta_s^k\}_{k\in\mathbb{N}}, \{\beta^k\}_{k\in\mathbb{N}}$. Namely, in Figure 4.2b and 4.2a we report the behaviour of the objective functions in (4.5) and (4.7) along the ADMM iterations for different choices of the penalty sequences (left). We choose $\beta^k = \beta_t^k = \beta_s^k \equiv 10$ for all k (blue line), $\beta^k = \beta_t^k = \beta_s^k = k^{0.5}$ (red line) and $\beta^k = \beta_t^k = \beta_s^k = k(1+\epsilon)^k$ with $\epsilon = 10^{-4}$ (yellow line). On the same plots we further show the decay of the quadratic data term (right). We observe that when the penalty sequence fulfil the required growth condition, then the convergence is nicely monotone, whereas for the other two choices, the decay exhibits oscillations while preserving a globally decreasing trend. Numerically, this suggests that possibly less severe growth conditions may be employed, such as a sufficiently large constant values of the penalty parameters. A further study on this is left for future research.

To confirm the improved computational performance of our ADMM algorithm w.r.t. to the one proposed in [68] and adapted to solve the SR problem (4.5), we report in Table 4.1 a comparison table both in terms of number of iterations-to-convergence and computational times. We stress that the poor performance of the ADMM algorithm in [68] is due here to the large computational cost required to solve the ℓ^0 gradient steps via inner optimization routines. This, combined with the use of CG solvers makes the overall cost much higher in comparison to our more explicit splitting. Furthermore, we report the iterations till convergence and the computational time of TV0I$^{\mathrm{F}}$ which



(a) (b)

Figure 4.2: Values of the objective functions (left y-axis) and of the fidelity term (right y-axis) in (4.7) (a) and (4.5) (b), along iterations, for $\lambda = 0.01$. The penalty sequences are chosen as $\beta^k = \beta_t^k = \beta_s^k \equiv 10$ (blue), $\beta^k = \beta_t^k = \beta_s^k = k^{0.5}$ (red), $\beta^k = \beta_t^k = \beta_s^k = k(1+\epsilon)^k$ with $\epsilon = 10^{-4}$ (yellow).

uses the closed-form Fourier-based strategy proposed in [75].

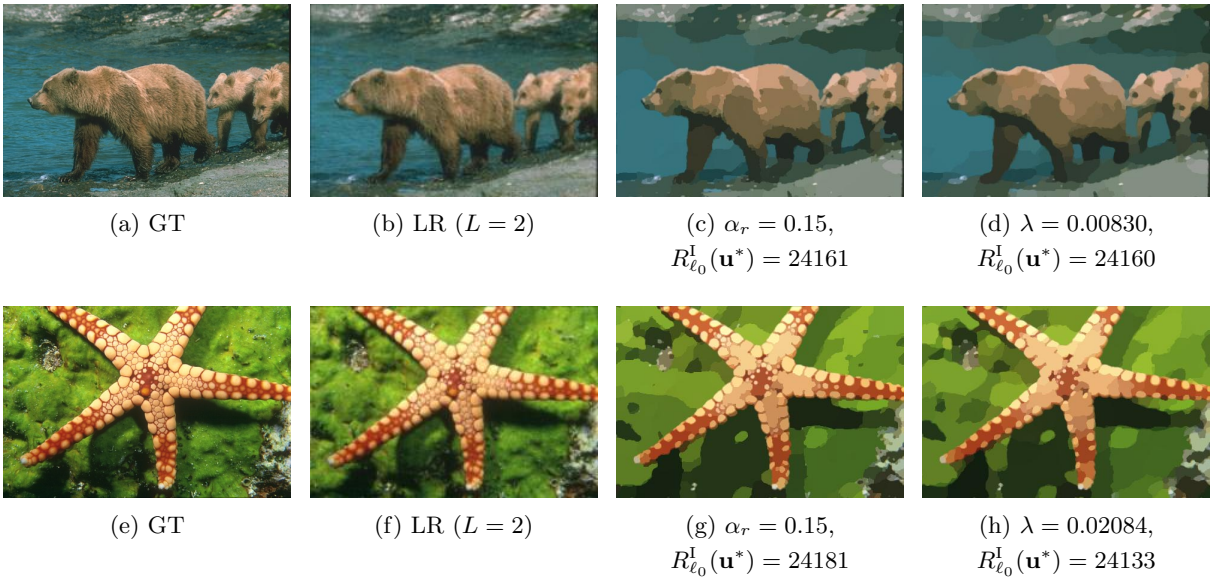
Method	[68]	TV0I$^{\mathrm{CG}}$	TV0I$^{\mathrm{F}}$
`iter`	1905	59	57
time (s)	2866.31	195.99	4.75

Table 4.1: Iterations till convergence (`iter`) and computational times (in seconds) for different methods solving (4.5) setting $\mathbf{b}$ equal the LR image depicted in Figure 4.1b.

On the choice of the regularization parameter

To favour comparisons between the constrained and the unconstrained isotropic ℓ_0-gradient regularized models, we fix the hyperparameter α_r related with the value α for c-TV0I$^{\mathrm{F}}$ and we estimate the regularization parameter λ of the unconstrained method so that the reconstruction computed by TV0I$^{\mathrm{F}}$ has (approximately) the same number of jumps as the solution of the constrained model. As an example, in Figure 4.3 we report two natural images of size 480×320 and their simulated LR acquisitions of size 240×160. We compute the HR solutions obtained by solving the constrained and the unconstrained models by setting $\alpha_r = \alpha/n = 0.16$ and choosing λ accordingly. We observe that the proposed choices of the hyperparamters α and λ produce comparable images. As a general comment, it is evident that there is no relation between the magnitude of λ and the level of partition, namely for different images different regularization parameters λ are required to obtain the same values of $R^{\mathrm{I}}_{\ell_0}(\mathbf{u}^*)$.

In the experiment reported in Figure 4.4 we increase the regularization strength in both algorithms. To do so, we decrease the value of the parameter α_r in the constrained proposal c-TV0I$^{\mathrm{F}}$ from 0.25 to 0.07 (so as to promote less and less jumps in the final image) and compare with the corresponding



(a) GT (b) LR ($L = 2$) (c) $\alpha_r = 0.15$, $R^{\mathrm{I}}_{\ell_0}(\mathbf{u}^*) = 24161$ (d) $\lambda = 0.00830$, $R^{\mathrm{I}}_{\ell_0}(\mathbf{u}^*) = 24160$

(e) GT (f) LR ($L = 2$) (g) $\alpha_r = 0.15$, $R^{\mathrm{I}}_{\ell_0}(\mathbf{u}^*) = 24181$ (h) $\lambda = 0.02084$, $R^{\mathrm{I}}_{\ell_0}(\mathbf{u}^*) = 24133$

Figure 4.3: Natural GT images of size 481×321 ((a) and (e)). Simulated LR acquisitions according to image formation model (1.17) by setting $L = 2$, $\sigma_{\mathbf{e}} = 0.01$, $\sigma_{\mathbf{H}} = 1.0$ ((b) and (f)). Reconstructions obtained by c-TV0I$^{\mathrm{F}}$ ((c) and (g)). Reconstructions obtained by TV0I$^{\mathrm{F}}$ ((d) and (h)).

choice for the unconstrained TV0I$^{\mathrm{F}}$ so as to have a similar value for $R^{\mathrm{I}}_{\ell_0}(\mathbf{u}^*)$ in the final image. As previously noticed, the effect of increasing the proposed regularization is a reduced number of partitions in the restored image corresponding to a lower number of classes in a possible later classification.

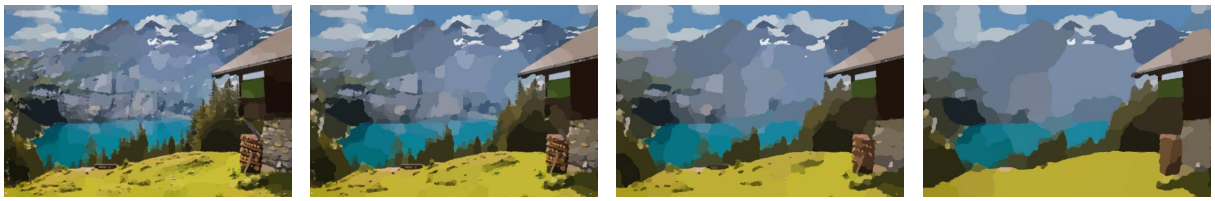
QR code recognition

The first application we consider is the problem of QR detection. As described, e.g., in [86], images of QR codes are often scanned by means of portable devices with limited resolution. Furthermore, QR scans are often taken from a distance and in non-optimal optical conditions so that blur and noise further limit the amount of visible information, thus making the use of artefact-free SR approaches crucial.

For the following tests, we first generate a binary QR code image of size 250×250, then we simulate several LR acquisitions according to the model (1.17) for different levels of degradation. We consider three test cases: $\sigma_{\mathbf{e}} = 0.01$ and $\sigma_{\mathbf{H}} = 1$ (TEST 1), $\sigma_{\mathbf{e}} = 0.05$ and $\sigma_{\mathbf{H}} = 1$ (TEST 2), and $\sigma_{\mathbf{e}} = 0.01$ and $\sigma_{\mathbf{H}} = 4$ (TEST 3). We compare the results obtained by our models with the competitors listed above. For each method, we select the model parameters maximising the Jaccard index. To avoid non-binary outputs (required for Jaccard index computations), for all models we post-process the SR results by means of an adaptive Otsu thresholding and re-compute the evaluation metrics on the binarised output, see Table 4.2.

In Figure 4.5 we report the results obtained by the different methods for the TEST 2 image before (red frame) and after (blue frame) binarization. We observe that due to the sharp nature of the the

(a) $\alpha_r = 0.25$ $(10.90s)$ (b) $\alpha_r = 0.17$ $(15.32s)$ (c) $\alpha_r = 0.10$ $(28.38s)$ (d) $\alpha_r = 0.07$ $(44.18s)$



(e) $\lambda = 0.005$ $(7.77s)$ (f) $\lambda = 0.01$ $(10.46s)$ (g) $\lambda = 0.02$ $(13.32s)$ (h) $\lambda = 0.04$ $(18.88s)$

Figure 4.4: Reconstructions with increasing regularization ($L = 2$). (a)-(d) Images obtained by c-TV0I$^{\mathrm{F}}$. (e)-(h) Images obtained by TV0I$^{\mathrm{F}}$.

ℓ_0-gradient regularizer, the TV0A$^{\mathrm{CG}}$ and TV0I$^{\mathrm{CG}}$ results are almost binary so they do not benefit much from the post-processing step in terms of Jaccard index values as much as the other methods do. In Figure 4.6 we report a zoom of the best results obtained before binarization by all methods starting from the highly corrupted TEST 3 LR image.

The quantitative evaluation of the results in terms of PSNR, SSIM and Jaccard index for the three different test cases is reported in Table 4.2. Without any binarization, the TV0A$^{\mathrm{CG}}$ model outperforms all the others as far as the PSNR, SSIM and Jaccard indices are concerned. The simplified geometry of the QR images considered (i.e. the sole presence of horizontal/vertical edges) makes in fact this kind of data tailored for such geometrically-biased regularizations. Furthermore, the highly non-convex jump-sparsification forces the ouptut to be almost binary, without the need of any further post-processing binarization, as it is required by all the other regularizations to achieve comparable (if not better) quality scores. This simple example shows that the image simplification intrinsically favoured by the use of ℓ_0-gradient regularizers shall limits the need of post-processing techniques in view of further segmentation analysis. As far as the deep-learning results are concerned, we remark that the CAR network in this experiment is used in a transfer learning mode, with no noisy nor blurred images observed in the training phase. For a fairer comparison, we thus consider the IRCNN PnP network which is capable to handle different levels of degradations, although it is shown to fail in the presence of highly-degraded data, see Figure 4.6.

Detail-preserving image cartoonisation

In [25] ℓ^0-gradient regularization was extensively shown to be effective on several image smoothing applications such as image cartoonisation and JPG compression artefact removal. Here, we consider a scenario where analogous tasks are performed along with a resolution improvement. To do so, we consider an RGB LR cartoon-type image of size 170×170 with not discernible details due to noise and blur artefacts caused by image compression and apply gradient-sparse SR models. As no ground

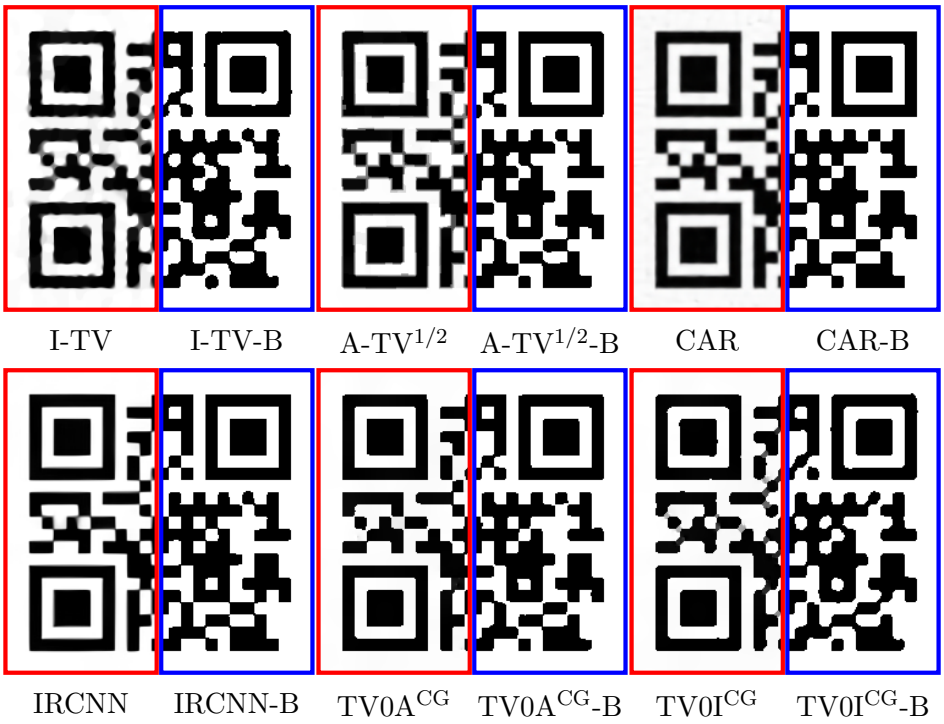


Figure 4.5: QR SR results obtained by different methods on TEST2 image before (red frames) and after (blue frames) binarisation.

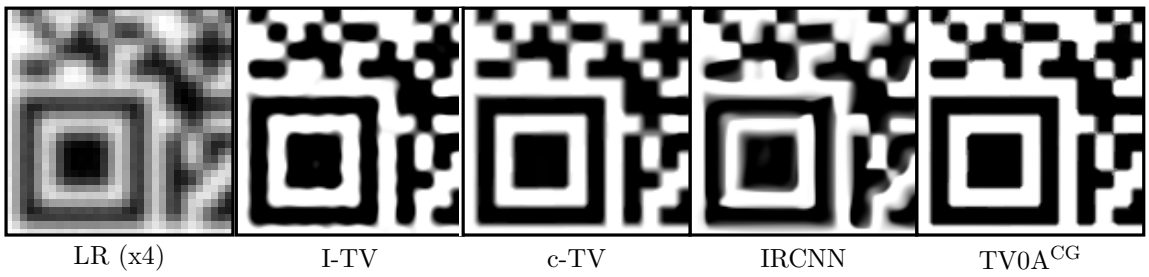


Figure 4.6: Details of QR code SR outputs for TEST 3 image.

truth is available for this example, for all models we empirically select the parameters producing the best visual output. In Figure 4.7 we report two close-ups of the computed SR reconstructions marked by blue and red boxes. The blue box highlights small details which are poorly discernible in the LR image, while the red box considers a patch of the face with some blunt edges and a small (but meaningful!) face mole (see green box). We see that both TV0A^{CG} and TV0I^{CG} reconstructions are sharper and more cartoonised than the ones obtained by the other models. However, we still observe anisotropic artifacts in the TV0A^{CG} reconstructions (the retrieved face-mole is squared). Furthermore, the well-known I-TV and c-TV loss of contrast reconstruction artefact makes small details hardly discernible. Due to the high-level of compression artefacts, both IRCNN and CAR perform poorly.

Table 4.2: Quantitative evaluation of SR models performance on QR for three different TEST images and methods. By "-B" we denote results after binarization. In each column we colour red the best method, blue the second-best.

LR	Method	PSNR	PSNR-B	SSIM	SSIM-B	Jac
TEST 1	TV0I^{CG}	22.5199	29.0809	0.9423	0.9873	0.9980
	TV0A^{CG}	32.5943	35.8478	0.9913	0.9989	0.9999
	I-TV	23.3845	26.3357	0.9489	0.9762	0.9963
	c-TV	19.4522	36.7496	0.8849	0.9977	0.9997
	A-TV$^{1/2}$	18.6328	36.7496	0.8594	0.9989	0.9997
	CAR	20.2460	27.8163	0.8159	0.9801	0.9966
	IRCNN	25.0589	35.3363	0.9622	0.9992	0.9995
TEST 2	TV0I^{CG}	19.3318	18.6308	0.8766	0.9156	0.9781
	TV0A^{CG}	22.6887	22.6256	0.9242	0.9653	0.9912
	I-TV	18.1101	18.9848	0.8012	0.9171	0.9798
	c-TV	18.7331	21.3473	0.8211	0.9595	0.9882
	A-TV$^{1/2}$	19.2182	22.5108	0.8664	0.9660	0.9910
	CAR	18.1320	26.7831	0.7493	0.9805	0.9906
	IRCNN	21.4314	26.3968	0.9057	0.9850	0.9902
TEST 3	TV0I^{CG}	18.3763	19.7532	0.8634	0.9294	0.9831
	TV0A^{CG}	19.2908	21.9341	0.8861	0.9556	0.9897
	I-TV	17.9552	20.1585	0.8222	0.9282	0.9846
	c-TV	16.9580	22.4648	0.7915	0.9605	0.9917
	A-TV$^{1/2}$	17.0785	20.6874	0.7706	0.9372	0.9863
	CAR	11.1809	11.5412	0.4057	0.6342	0.8887
	IRCNN	14.2915	12.5640	0.6342	0.6565	0.9133

On the effectiveness of a joint SR and IP approach

In this paragraph we show the effectiveness of the joint SR and IP approach as a pre-processing step before segmentation. In Figure 4.8 and 4.9 some results are reported. More precisely, for given unknown piecewise constant (Figure 4.8a) and natural (Figure 4.8g) HR image data, the super-resolved results obtained by applying TV0I^{CG} are reported in Figure 4.8c and Figure 4.8i, respectively. As it can be clearly seen, the method provides an accurate partitioning of both images which can be a precious pre-processing for further tasks. To motivate this further, we report in Figure 4.8f, in Figure 4.8l and in Figure 4.9d an example of how a good joint SR and IP process helps detecting the objects of interest which are there shown in terms of a given binary mask computed through standard segmentation algorithms or super-imposed to the image, respectively.

Cell detection

Table 4.3: Quantitative comparisons of SR and segmentation performance among different methods on the EVICAN dataset. Confidence intervals with 95% of confidence.

Method	PSNR	SSIM	Jaccard
A-TV$^{1/2}$	[32.0277,35.6011]	[0.7473,0.8775]	[0.7767,0.8541]
TV0I^{CG}	[31.9137,35.5359]	[0.7458,0.8778]	[0.7775,0.8622]
IRCNN	[32.9723,35.9698]	[0.7871,0.9070]	[0.5501,0.7531]

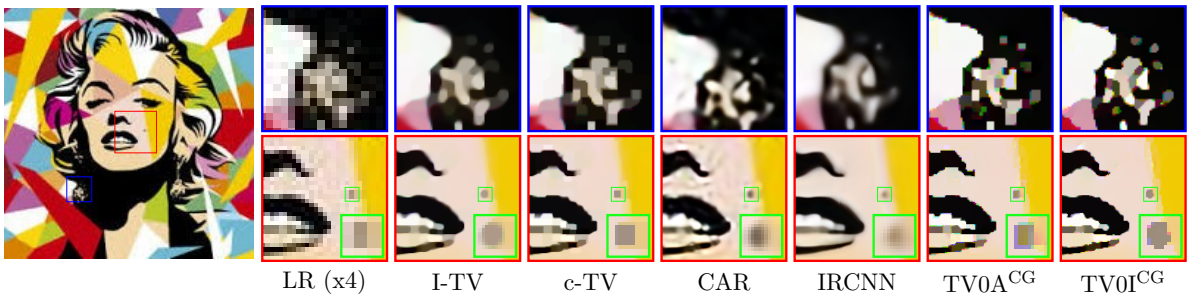
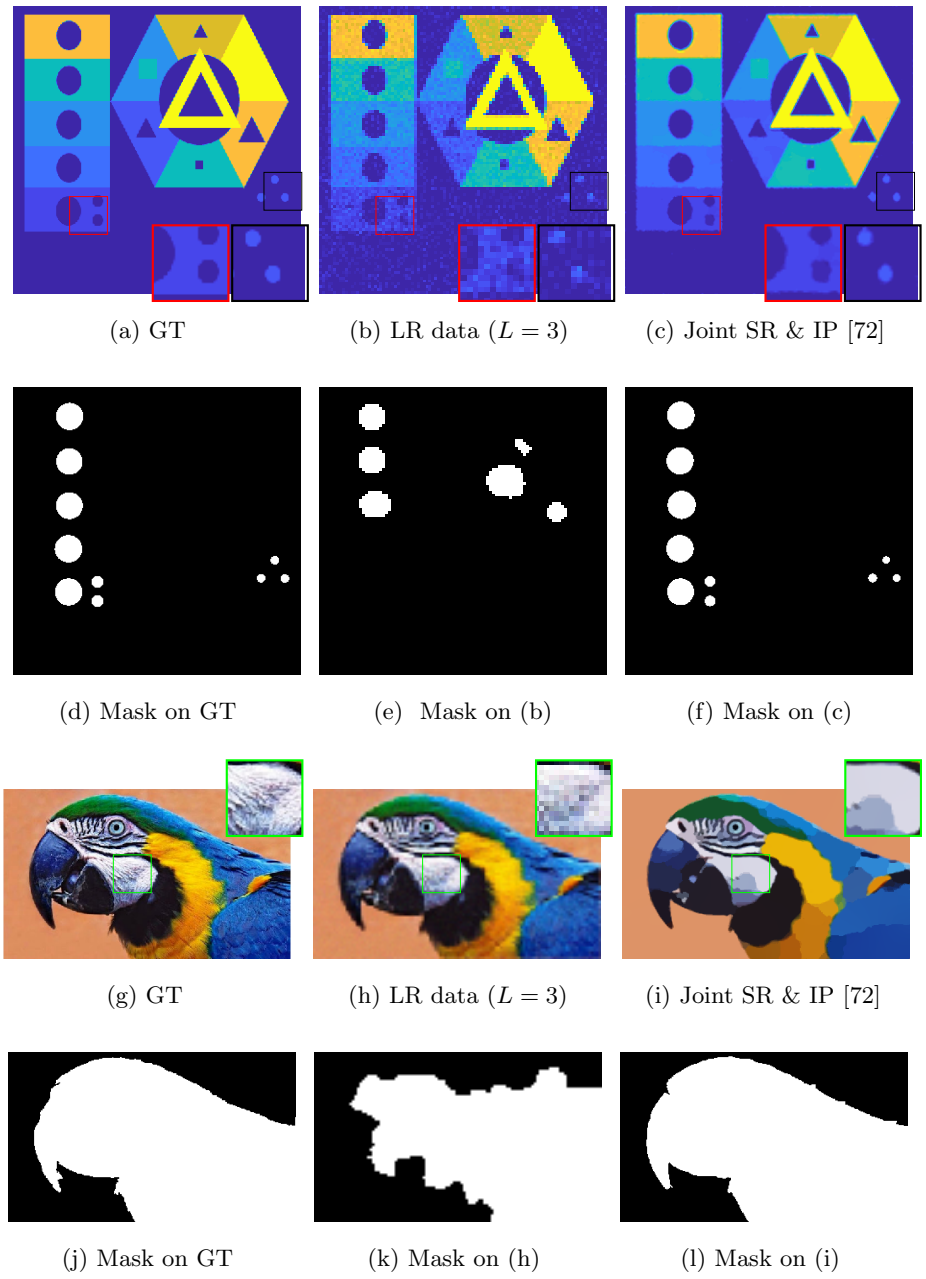


Figure 4.7: Detail-preserving image cartoonisation by SR models. For I-TV, c-TV, TV0A^{CG} and TV0I^{CG} the regularisation parameters are chosen as λ: 0.08, 0.05, 0.02, 0.02, respectively.

Standard light-microscopes suffer from a limited resolving power which often causes blur artefacts and limits spatial resolution in images. In such conditions, the performance of simple segmentation algorithms extracting isolated cells as well as cell clusters is often very limited and may benefit significantly from the use of a joint super-resolution image restoration pre-processing. We thus test the proposed ℓ^0-gradient SR model to segment a dataset of 30 light-microscope images extracted from the EVICAN dataset [87]. We apply the TV0I^{CG} algorithm and its competitors on LR acquisitions obtained by applying the linear degradation model (1.17) to the original images, considered here as Ground Truth (GT) with the following values of parameters: $L = 3$, $\sigma_{\mathbf{H}} = 2$ and $\sigma_{\mathbf{e}} = 0.02$. In Figure 4.10 we show the results for one test image in the dataset. Due to our interest in analysing the effectiveness of the proposed methods in pre-processing images for segmentation, we compute for each SR output image a binary mask by applying the cell-segmentation Matlab toolbox[1] based on edge detection and morphology. For the different methods, the segmented regions are shown in Figure 4.10c - 4.10e, while in Table 4.3 the confidence intervals (95% of confidence) of the PSNR, SSIM and Jaccard values computed on the whole dataset are reported. Thanks to its strong smoothing properties, the ℓ^0-gradient sparsity enforced by the TV0I^{CG} method allows for a better detection of the two isolated cells (green boxes) as well as the cell cluster (red boxes), resulting in higher Jaccard index values, as expected.

[1] https://www.mathworks.com/help/images/detecting-a-cell



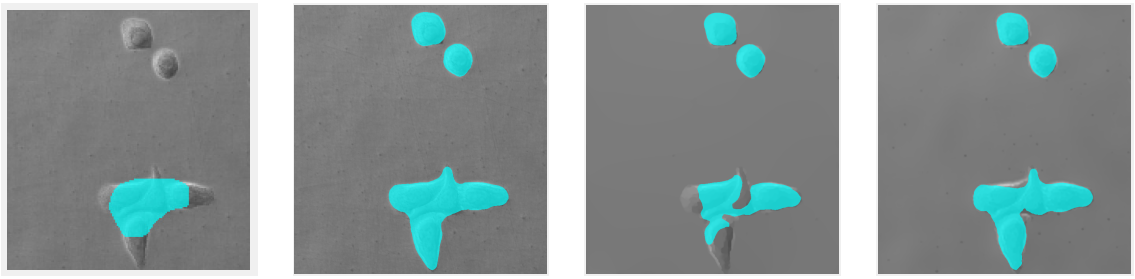
(a) GT (b) LR data ($L = 3$) (c) Joint SR & IP [72]

(d) Mask on GT (e) Mask on (b) (f) Mask on (c)

(g) GT (h) LR data ($L = 3$) (i) Joint SR & IP [72]

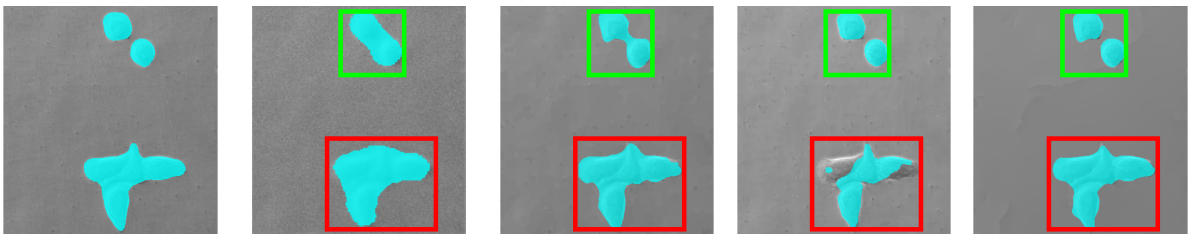
(j) Mask on GT (k) Mask on (h) (l) Mask on (i)

Figure 4.8: Joint SISR and IP of a piecewise constant (upper rows) and natural image (lower rows). The simulated LR acquisitions are corrupted according to model (1.17) by setting $L = 3$, $\sigma_{\mathbf{e}} = 0.01$, $\sigma_{\mathbf{H}} = 1.0$. The SR images have been obtained by appyling the algorithm TV0I$^{\mathrm{CG}}$ setting $\lambda = 0.0024$ and $\lambda = 0.023$, respectively.



(a) LR data (b) GT + mask (c) Sequential SR + IP (d) Joint SR & IP

Figure 4.9: Cell detection on low- and super-resolved data. Masks computed on the cells of interest are coloured cyan. (a) LR image ($L = 5$). (b) GT image with superimposed mask. (c) SR image computed by first applying the bicubic interpolation algorithm to the LR data and then the IP algorithm based on the gradient smoothing described in [25]. (d) SR image obtained from LR data by applying TV0I$^{\mathrm{CG}}$.



(a) GT (b) LR(x3) (c) A-TV$^{1/2}$ (d) IRCNN (e) TV0I$^{\mathrm{CG}}$

Figure 4.10: Cell detection results. In (b)-(c) the green and red squares indicate two isolate cells and a cell cluster, respectively. Computed masks are coloured cyan. In (c)-(e) the Jaccard index are 0.9013, 0.7007, 0.9320, respectively.

Chapter 5

Plug-and-Play gradient-based denoisers for CT image enhancement

Nowadays X-rays CT systems are designed to acquire images of almost every part of the human body. As a result, tomographic images are quite different from each other and they may contain several objects of various size, shape, and contrast with respect to the background. Moreover, the objective of the imaging task can be to identify small and low-contrasted objects, as a breast microcalcification, a tumoral mass, a larger bone with neat edges or a very thin vessel. In some cases, it is also necessary to subsequently segment the object or an area of interest in the restored image, to help the diagnosis. We assume the CT image $\mathbf{b}$ is corrupted according to the acquisition model (1.16) by considering AWGN of standard deviation $\sigma_{\mathbf{e}}$. An estimate $\mathbf{u}^* \in \mathbb{R}^n$ of $\mathbf{u}$ can be obtained by exploiting the variational framework, namely:

$$\mathbf{u}^* \in \underset{\mathbf{u}\in\mathbb{R}^n}{\arg\min}\ \frac{1}{2}\|\mathbf{A}\mathbf{u}-\mathbf{b}\|_2^2 + \lambda R(\mathbf{u}), \tag{5.1}$$

where by R we have denoted the regularization term which induces prior information on the estimate $\mathbf{u}^*$ by reflecting, for example, sparsity patterns, smoothness or geometric assumptions. It is well-known that regularizers defined on the gradient domain may enhance medical image reconstructions both in terms of shape recovering and noise removal [88, 89].

Very recently, in [90, 91] the authors proposed to adopt a Plug-and-Play scheme to restore CT images. The Plug-and-Play framework was firstly proposed in [29], where the authors strikingly showed that any off-the-shelf denoiser can be used to induce regularization on the desired solution. In particular, by adding an auxiliary variable $\mathbf{z} = \mathbf{u}$, the authors in [29] apply the ADMM scheme to solve (5.1). Hence, by setting $\beta > 0$ we obtain:

$$\begin{cases} \mathbf{u}^{k+1} & = \underset{\mathbf{u}\in\mathbb{R}^n}{\arg\min}\ \frac{1}{2}\|\mathbf{A}\mathbf{u}-\mathbf{b}\|_2^2 + \frac{\beta}{2}\|\mathbf{u}-(\mathbf{z}^k - \frac{\boldsymbol{\lambda}^k}{\beta})\|_2^2 \\ \mathbf{z}^{k+1} & \in \underset{\mathbf{z}\in\mathbb{R}^n}{\arg\min}\ \lambda R(\mathbf{z}) + \frac{\beta}{2}\|\mathbf{z}-(\mathbf{u}^k + \frac{\boldsymbol{\lambda}^k}{\beta})\|_2^2 \\ \boldsymbol{\lambda}^{k+1} & = \boldsymbol{\lambda}^k - \beta(\mathbf{z}^{k+1} - \mathbf{u}^{k+1}). \end{cases} \tag{5.2}$$

The modular structure of ADMM allows to deal with the fidelity and the regularization terms, separately. It is worth noting that the second step equals the proximal map of R with parameter $\frac{\lambda}{\beta}$

applied to $\mathbf{u}^k + \frac{\boldsymbol{\lambda}^k}{\beta}$ and can be interpreted as a denoising operation. Therefore, in [29] the authors proposed to replace the substep related to R with any off-the-shelf denoiser $\mathsf{D}_{\sigma_\mathsf{D}}(\cdot) : \mathbb{R}^n \to \mathbb{R}^n$, whose strength depends on the parameter $\sigma_\mathsf{D} := \frac{\lambda}{\beta}$, thus leading to the following iterative scheme:

$$\begin{cases} \mathbf{u}^{k+1} & \in \underset{\mathbf{u}\in\mathbb{R}^n}{\arg\min}\ \frac{1}{2}\|\mathbf{A}\mathbf{u}-\mathbf{b}\|_2^2 + \frac{\beta}{2}\|\mathbf{u} - (\mathbf{z}^k - \frac{\boldsymbol{\lambda}^k}{\beta})\|_2^2 \\ \mathbf{z}^{k+1} & = \mathsf{D}_{\sigma_\mathsf{D}}(\mathbf{u}^k + \frac{\boldsymbol{\lambda}^k}{\beta}) \\ \boldsymbol{\lambda}^{k+1} & = \boldsymbol{\lambda}^k - \beta(\mathbf{z}^{k+1} - \mathbf{u}^{k+1}). \end{cases} \tag{ADMM-PnP}$$

Hence, by means of splitting strategies the authors in [29] have created a flexible framework to easily force regularization by using denoising algorithms.

Contribution

We propose a PnP framework specifying a gradient-based CNN denoiser, to solve the deblurring task, through CNN networks trained to restore the corrupted image gradients. So far, among the wide literature of PnP, the embedded deep learning based denoisers have always considered only the image space. Moreover, motivated by the apparent complementarity of external and internal denoisers, we also propose a hybrid PnP scheme combining the Total Variation and our CNN-based denoiser. The considered PnP frameworks rely on the HQS algorithm for which we derive a fixed point convergence proof upon weak assumptions on the considered denoisers. This chapter is mainly based on the publication [92] in which we apply this novel PnP algorithm for the enhancement of CT images.

5.1 A short survey on Plug-and-Play

So far, a large number of papers on PnP have been published analyzing different aspects of the scheme, such as the proximal algorithm or the included denoiser. In particular, the considered proximal algorithms are the ADMM, the HQS or the FISTA [29, 34, 93, 94, 95]. Furthermore, in the last few years, PnP methods have been analyzed both in the consensus equilibrium (CE) and in the learning to optimize (L2O) frameworks [96, 97].

Focusing on the choice of the plugged denoiser, several proposals have already been successfully tested and they are usually labelled as internal or external denoisers [98]. Internal denoisers are tailored to define features onto the observed data. As consequence, they struggle to deal with several different image features simultaneously. Examples are the proximal maps of handcrafted regularizers, the BM3D [33] and the Non-Local Mean (NLM) filter [32]. External denoisers are related to an outer set of degraded-clean images, so they can fail when dealing with unseen noise variance and image patterns. Early studies made use of Gaussian Mixture Models (GMMs) [99] and trained nonlinear reaction diffusion based denoisers [100] as external denoisers. Since nowadays deep learning based architectures lead to outstanding performances for denoising images [101, 102], PnP frameworks are also equipped with pre-trained CNN-based denoisers in works such as [34, 35, 103]. The aforementioned approaches exploit either external or internal denoisers; very recently some

generalizations to handle multiple internal and/or external denoisers have been proposed in [104, 105]. In their seminal work the authors have not discussed theoretical convergence properties of the Plug-and-Play framework. The scheme ADMM-PnP is guaranteed to converge to the minimum $\mathbf{u}^*$ in (5.1) when $\mathsf{D}_{\sigma_\mathsf{D}}(\cdot)$ is the proximal map of a convex, closed and proper function R [77, 106]. Then, according to the Moreau's Theorem [107], $\mathsf{D}_{\sigma_\mathsf{D}}(\cdot)$ is required to be non-expansive, conservative and differentiable. However, in [93] the authors have shown that, in general, the most famous denoisers are expansive maps. Nevertheless, by considering an increasing sequence $\{\beta^k\}_{k\in\mathbb{N}}$ of penalty parameters and assuming $\mathsf{D}_{\sigma_\mathsf{D}}(\cdot)$ is a bounded denoiser, in [94], a fixed-point convergence theorem is proved for the scheme ADMM-PnP.

5.2 A hybrid Plug-and-Play scheme

We replace the optimization problem (5.1) by the following one:

$$\mathbf{u}^* \in \underset{\mathbf{u}\in\mathbb{R}^n}{\arg\min} \left\{ \frac{1}{2}\|\mathbf{A}\mathbf{u}-\mathbf{b}\|_2^2 + \lambda R_1(\mathbf{L}_1\mathbf{u}) + \eta R_2(\mathbf{L}_2\mathbf{u}) \right\}, \tag{5.3}$$

where we assume R_1 and R_2 are positive and convex real-valued maps:

$$R_1 : \mathbb{R}^{l_1} \to \mathbb{R}^+, \quad R_2 : \mathbb{R}^{l_2} \to \mathbb{R}^+, \tag{5.4}$$

with l_1 and l_2 positive integers, $\mathbf{L}_1 \in \mathbb{R}^{l_1\times n}$ and $\mathbf{L}_2 \in \mathbb{R}^{l_2\times n}$. Hence, instead considering one regularizer we consider two different regularization terms.
We now make use of the HQS iterative method described in [108, 109] as numerical solver to compute $\mathbf{u}^*$. By introducing the auxiliary variables $\mathbf{t} \in \mathbb{R}^{l_1}$ and $\mathbf{z} \in \mathbb{R}^{l_2}$ subject to $\mathbf{t} := \mathbf{L}_1\mathbf{u}$ and $\mathbf{z} := \mathbf{L}_2\mathbf{u}$, the following penalized half-quadratic function is taken into account:

$$\mathcal{L}(\mathbf{u},\mathbf{t},\mathbf{z};\rho^{\mathbf{t}},\rho^{\mathbf{z}}) := \frac{1}{2}\|\mathbf{A}\mathbf{u}-\mathbf{b}\|_2^2 + \lambda R_1(\mathbf{t}) + \eta R_2(\mathbf{z}) + \frac{\rho^{\mathbf{t}}}{2}\|\mathbf{L}_1\mathbf{u}-\mathbf{t}\|_2^2 + \frac{\rho^{\mathbf{z}}}{2}\|\mathbf{L}_2\mathbf{u}-\mathbf{z}\|_2^2. \tag{5.5}$$

At each iteration k, the HQS algorithm performs this alternated minimization scheme w.r.t. $\mathbf{t}$, $\mathbf{z}$ and the primal variable $\mathbf{u}$:

$$\begin{cases} \mathbf{t}_{k+1} \in \underset{\mathbf{t}\in\mathbb{R}^{l_1}}{\arg\min}\ \lambda R_1(\mathbf{t}) + \dfrac{\rho_k^{\mathbf{t}}}{2}\|\mathbf{L}_1\mathbf{u}_k-\mathbf{t}\|_2^2 & (5.6) \\ \mathbf{z}_{k+1} \in \underset{\mathbf{z}\in\mathbb{R}^{l_2}}{\arg\min}\ \eta R_2(\mathbf{z}) + \dfrac{\rho_k^{\mathbf{z}}}{2}\|\mathbf{L}_2\mathbf{u}_k-\mathbf{z}\|_2^2 & (5.7) \\ \mathbf{u}_{k+1} = \underset{\mathbf{u}\in\mathbb{R}^n}{\arg\min}\ \dfrac{1}{2}\|\mathbf{A}\mathbf{u}-\mathbf{b}\|_2^2 + \dfrac{\rho_k^{\mathbf{t}}}{2}\|\mathbf{L}_1\mathbf{u}-\mathbf{t}_{k+1}\|_2^2 + \dfrac{\rho_k^{\mathbf{z}}}{2}\|\mathbf{L}_2\mathbf{u}-\mathbf{z}_{k+1}\|_2^2, & (5.8) \end{cases}$$

where $\{\rho_k^{\mathbf{t}}\}_{k\in\mathbb{N}}$ and $\{\rho_k^{\mathbf{z}})_{k\in\mathbb{N}}$ are two non-decreasing sequences of positive penalty parameters.
The key feature of HQS is that the prior related sub-steps (5.6) and (5.7) are specified through the proximal maps of R_1 and R_2, respectively, which are mathematically equivalent to regularized denoising problems. The PnP framework exploits both this equivalence and the modular structure of the algorithm by replacing such proximal maps with any off-the-shelf denoiser.

To define our hybrid PnP scheme, we introduce a pre-trained learning-based denoiser $\mathsf{D}_\sigma^{\text{ext}}$ and an image-specific denoiser $\mathsf{D}_\gamma^{\text{int}}$. These denoisers depend on the positive parameters σ and γ which are related to the noise-level in the images to denoise, so that the greater σ and γ, the stronger the smoothing effect is. In particular, in our scheme we choose two sequences $\{\sigma_k\}_{k\in\mathbb{N}}$ and $\{\gamma_k\}_{k\in\mathbb{N}}$ such that, at step k, $\mathsf{D}_{\sigma_k}^{\text{ext}}$ and $\mathsf{D}_{\gamma_k}^{\text{int}}$ replace the sub-steps (5.6) and (5.7), respectively. A standard assumption in PnP is that σ_k and γ_k are both related with the penalty parameters $\rho_k^{\mathbf{t}}$ and $\rho_k^{\mathbf{z}}$ through these formulas:

$$\sigma_k := \sqrt{\frac{\alpha}{\rho_k^{\mathbf{t}}}}, \quad \gamma_k := \sqrt{\frac{\beta}{\rho_k^{\mathbf{z}}}}, \tag{5.9}$$

where α and β are chosen positive scaling factors. A sketch of the resulting hybrid PnP framework is reported in Algorithm 6.

Algorithm 6 Hybrid PnP HQS scheme

Input: α, β and $\{\rho_k^{\mathbf{t}}\}_{k\in\mathbb{N}}$, $\{\rho_k^{\mathbf{z}}\}_{k\in\mathbb{N}}$, $\mathbf{A}$, $\mathbf{L}_1$, $\mathbf{L}_2$, $\mathbf{b}$, $\mathbf{u}_1$, K.
Output: $\mathbf{u}_K$.
for k $= 1 \dots K$ **do**
$\mathbf{t}_{k+1} = \mathsf{D}_{\sigma_k}^{\text{ext}}(\mathbf{L}_1\mathbf{u}_k)$
$\mathbf{z}_{k+1} = \mathsf{D}_{\gamma_k}^{\text{int}}(\mathbf{L}_2\mathbf{u}_k)$
$\mathbf{u}_{k+1} = \arg\min_{\mathbf{u}\in\mathbb{R}^n} \frac{1}{2}\|\mathbf{A}\mathbf{u}-\mathbf{b}\|_2^2 + \frac{\rho_k^{\mathbf{t}}}{2}\|\mathbf{L}_1\mathbf{u}-\mathbf{t}_{k+1}\|_2^2 + \frac{\rho_k^{\mathbf{z}}}{2}\|\mathbf{L}_2\mathbf{u}-\mathbf{z}_{k+1}\|_2^2$
end for

We remark that under some quite general assumptions on the denoisers and on the sequences $\{\rho_k^{\mathbf{t}}\}_{k\in\mathbb{N}}$ and $\{\rho_k^{\mathbf{z}}\}_{k\in\mathbb{N}}$, the iterates defined in Algorithm 6 converge to a fixed-point $(\hat{\mathbf{u}}^*, \hat{\mathbf{t}}^*, \hat{\mathbf{z}}^*)$. In the following section, an in-depth discussion on the hypothesis and the fixed-point convergence theorem are reported.

5.2.1 A fixed-point convergence theorem for the hybrid Plug-and-Play scheme

To analyze the convergence properties of Algorithm 6, we start observing that if the denoisers $\mathsf{D}_\sigma^{\text{ext}}$ and $\mathsf{D}_\gamma^{\text{int}}$ are the proximal maps of two convex functions R_1 and R_2, respectively, then the scheme admits a variational structure and the convergence to a global minimum $\mathbf{u}^*$ of the objective function in (5.3) is guaranteed [108, 109]. However, in [93] the authors observe that a denoiser is a proximal map when it is non-expansive with symmetric gradient, thus limiting the set of suitable denoisers. In the effort of allowing less strict conditions on the involved denoisers, we show in this section that the proposed Algorithm 6 satisfies a fixed-point convergence theorem provided only their boundedness.

Definition 5.1 (Bounded Denoiser [94])**.** A *bounded denoiser* with parameter ϵ is a function $\mathsf{D}_\epsilon : \mathbb{R}^l \to \mathbb{R}^l$ such that for any $\mathbf{t} \in \mathbb{R}^l$ the following inequality holds:

$$\|\mathsf{D}_\epsilon(\mathbf{t}) - \mathbf{t}\|_2^2 \leq \epsilon^2 C_{\mathsf{D}}, \tag{5.10}$$

for a constant C_{D} independent of ϵ.

The previous definition entails that given the sequence $\{\epsilon_k\}_{k\in\mathbb{N}}$, D_{ϵ_k} converges to the identity function of $\mathbb{R}^l$ as $\epsilon_k \to 0$.
In order to state and prove the following fixed-point theorem, we make some assumptions.
Given $\{\rho_k^{\mathbf{t}}\}_{k\in\mathbb{N}}$ and $\{\rho_k^{\mathbf{z}}\}_{k\in\mathbb{N}}$ non-decreasing positive sequences, $\mathbf{L}_1 \in \mathbb{R}^{l_1\times n}$, $\mathbf{L}_2 \in \mathbb{R}^{l_2\times n}$ as input for Algorithm 6, then we assume:

1. $\mathsf{D}_\sigma^{\text{ext}}$ and $\mathsf{D}_\gamma^{\text{int}}$ are bounded denoisers.
2. $\mathbf{L}_1$ and $\mathbf{L}_2$ are full-rank matrices.
3. $\sum_{k=1}^{+\infty}\sqrt{\frac{k}{\rho_k^{\mathbf{z}}}} < +\infty, \quad \sum_{k=1}^{+\infty}\sqrt{\frac{k}{\rho_k^{\mathbf{t}}}} < +\infty$ and $\frac{\rho_k^{\mathbf{z}}}{\rho_k^{\mathbf{t}}} \to c$ where $c \in \mathbb{R}^+$.

Theorem 5.1 (Fixed-point convergence theorem for the hybrid PnP Algorithm 6)**.** *Given the assumptions 1-3, there exist* $\hat{\mathbf{t}}^* \in \mathbb{R}^{l_1}, \hat{\mathbf{z}}^* \in \mathbb{R}^{l_2}$ *and* $\hat{\mathbf{u}}^* \in \mathbb{R}^n$ *such that, for* $k \to \infty$*, the following relations hold:*

$$\mathbf{t}_k \to \hat{\mathbf{t}}^*, \quad \mathbf{L}_1\mathbf{u}_k \to \hat{\mathbf{t}}^*, \quad \mathbf{z}_k \to \hat{\mathbf{z}}^*, \quad \mathbf{L}_2\mathbf{u}_k \to \hat{\mathbf{z}}^*, \quad \mathbf{u}_k \to \hat{\mathbf{u}}^*,$$

where $\mathbf{t}_k, \mathbf{z}_k, \mathbf{u}_k$ *are computed as in Algorithm 6 at step* k*.*

Proof. By observing that $\mathbf{u}_{k+1}$ is the optimal solution of the minimization problem (5.8), and by using the relations in (5.9) and the assumption 1, we get the following chain of inequalities:

$$\begin{aligned}
&\frac{1}{2}\|\mathbf{A}\mathbf{u}_{k+1}-\mathbf{b}\|_2^2 + \frac{\rho_k^{\mathbf{t}}}{2}\|\mathbf{t}_{k+1}-\mathbf{L}_1\mathbf{u}_{k+1}\|_2^2 + \frac{\rho_k^{\mathbf{z}}}{2}\|\mathbf{z}_{k+1}-\mathbf{L}_2\mathbf{u}_{k+1}\|_2^2 \leq \\
&\leq \frac{1}{2}\|\mathbf{A}\mathbf{u}_k-\mathbf{b}\|_2^2 + \frac{\rho_k^{\mathbf{t}}}{2}\|\mathbf{t}_{k+1}-\mathbf{L}_1\mathbf{u}_k\|_2^2 + \frac{\rho_k^{\mathbf{z}}}{2}\|\mathbf{z}_{k+1}-\mathbf{L}_2\mathbf{u}_k\|_2^2 = \\
&= \frac{1}{2}\|\mathbf{A}\mathbf{u}_k-\mathbf{b}\|_2^2 + \frac{\rho_k^{\mathbf{t}}}{2}\|\mathsf{D}_{\sigma_k}^{\text{ext}}(\mathbf{L}_1\mathbf{u}_k)-\mathbf{L}_1\mathbf{u}_k\|_2^2 + \frac{\rho_k^{\mathbf{z}}}{2}\|\mathsf{D}_{\gamma_k}^{\text{int}}(\mathbf{L}_2\mathbf{u}_k)-\mathbf{L}_2\mathbf{u}_k\|_2^2 \leq \\
&\leq \frac{1}{2}\|\mathbf{A}\mathbf{u}_k-\mathbf{b}\|_2^2 + \frac{\rho_k^{\mathbf{t}}}{2}\sigma_k^2 C_{\mathsf{D}^{\text{ext}}} + \frac{\rho_k^{\mathbf{z}}}{2}\gamma_k^2 C_{\mathsf{D}^{\text{int}}} = \\
&= \frac{1}{2}\|\mathbf{A}\mathbf{u}_k-\mathbf{b}\|_2^2 + \frac{\alpha}{2}C_{\mathsf{D}^{\text{ext}}} + \frac{\beta}{2}C_{\mathsf{D}^{\text{int}}} \leq \\
&= \frac{1}{2}\|\mathbf{A}\mathbf{u}_k-\mathbf{b}\|_2^2 + \tilde{C},
\end{aligned} \tag{5.11}$$

with $\tilde{C} := \frac{\alpha}{2}C_{\mathsf{D}^{\text{ext}}} + \frac{\beta}{2}C_{\mathsf{D}^{\text{int}}}$.
Since all the considered terms in (5.11) are positive, the following inequalities hold:

$$\frac{1}{2}\|\mathbf{A}\mathbf{u}_{k+1}-\mathbf{b}\|_2^2 \leq \frac{1}{2}\|\mathbf{A}\mathbf{u}_k-\mathbf{b}\|_2^2 + \tilde{C} \leq \cdots \leq \frac{1}{2}\|\mathbf{A}\mathbf{u}_1-\mathbf{b}\|_2^2 + k\tilde{C}. \tag{5.12}$$

For the same reason, using (5.11) and (5.12) we get:

$$\|\mathbf{t}_{k+1}-\mathbf{L}_1\mathbf{u}_{k+1}\|_2 \leq \sqrt{\frac{1}{\rho_k^{\mathbf{t}}}}\|\mathbf{A}\mathbf{u}_1-\mathbf{b}\|_2 + \sqrt{\frac{2\tilde{C}k}{\rho_k^{\mathbf{t}}}}, \tag{5.13}$$

$$\|\mathbf{z}_{k+1} - \mathbf{L}_2\mathbf{u}_{k+1}\|_2 \leq \sqrt{\frac{1}{\rho_k^{\mathbf{z}}}}\|\mathbf{A}\mathbf{u}_1 - \mathbf{b}\|_2 + \sqrt{\frac{2\tilde{C}k}{\rho_k^{\mathbf{z}}}}. \tag{5.14}$$

We now prove that the sequences $\{\mathbf{t}_k\}_{k\in\mathbb{N}}$ and $\{\mathbf{z}_k\}_{k\in\mathbb{N}}$ are Cauchy sequences. Starting from the expressions of $\mathbf{t}_{k+1}$ and $\mathbf{z}_{k+1}$ in Algorithm 6, applying the definition of bounded denoiser and the estimates (5.13) and (5.14) the following inequalities hold:

$$\begin{aligned}\|\mathbf{t}_{k+1} - \mathbf{t}_k\|_2 &\leq \|\mathsf{D}_{\sigma_k}^{\text{ext}}(\mathbf{L}_1\mathbf{u}_k) - \mathbf{L}_1\mathbf{u}_k\|_2 + \|\mathbf{L}_1\mathbf{u}_k - \mathbf{t}_k\|_2 \leq \\ &\leq \sqrt{\frac{\alpha}{\rho_k^{\mathbf{t}}}}\sqrt{C_{\mathsf{D}^{\text{ext}}}} + \sqrt{\frac{1}{\rho_{k-1}^{\mathbf{t}}}}\|\mathbf{A}\mathbf{u}_1 - \mathbf{b}\|_2 + \sqrt{\frac{2\tilde{C}(k-1)}{\rho_{k-1}^{\mathbf{t}}}}\end{aligned} \tag{5.15}$$

$$\begin{aligned}\|\mathbf{z}_{k+1} - \mathbf{z}_k\|_2 &\leq \|\mathsf{D}_{\gamma_k}^{\text{int}}(\mathbf{L}_2\mathbf{u}_k) - \mathbf{L}_2\mathbf{u}_k\|_2 + \|\mathbf{L}_2\mathbf{u}_k - \mathbf{z}_k\|_2 \leq \\ &\leq \sqrt{\frac{\beta}{\rho_k^{\mathbf{z}}}}\sqrt{C_{\mathsf{D}^{\text{int}}}} + \sqrt{\frac{1}{\rho_{k-1}^{\mathbf{z}}}}\|\mathbf{A}\mathbf{u}_1 - \mathbf{b}\|_2 + \sqrt{\frac{2\tilde{C}(k-1)}{\rho_{k-1}^{\mathbf{z}}}}.\end{aligned} \tag{5.16}$$

By assumption 3 $\{\mathbf{z}_k\}_{k\in\mathbb{N}}$ and $\{\mathbf{t}_k\}_{k\in\mathbb{N}}$ are Cauchy sequences. Hence, there exist $\hat{\mathbf{t}}^*$ and $\hat{\mathbf{z}}^*$ such that $\mathbf{t}_k \to \hat{\mathbf{t}}^*$ and $\mathbf{z}_k \to \hat{\mathbf{z}}^*$.
Furthermore, the following inequalities (which use (5.13) and (5.14), respectively) state that $\mathbf{L}_1\mathbf{u}_{k+1} \to \hat{\mathbf{t}}^*$ and $\mathbf{L}_2\mathbf{u}_{k+1} \to \hat{\mathbf{z}}^*$:

$$\|\mathbf{L}_1\mathbf{u}_{k+1} - \hat{\mathbf{t}}^*\|_2 \leq \|\mathbf{L}_1\mathbf{u}_{k+1} - \mathbf{t}_{k+1}\|_2 + \|\mathbf{t}_{k+1} - \hat{\mathbf{t}}^*\|_2, \tag{5.17}$$

$$\|\mathbf{L}_2\mathbf{u}_{k+1} - \hat{\mathbf{z}}^*\|_2 \leq \|\mathbf{L}_2\mathbf{u}_{k+1} - \mathbf{z}_{k+1}\|_2 + \|\mathbf{z}_{k+1} - \hat{\mathbf{z}}^*\|_2. \tag{5.18}$$

Now, we prove the convergence of the sequence $\{\mathbf{u}_k\}_{k\in\mathbb{N}}$ computed as in Algorithm 6. At step k, $\mathbf{u}_{k+1}$ is the solution of the convex minimization problem (5.8), therefore the first order optimality conditions lead:

$$\left(\frac{1}{\rho_k^{\mathbf{t}}}\mathbf{A}^T\mathbf{A} + \mathbf{L}_1^T\mathbf{L}_1 + \frac{\rho_k^{\mathbf{z}}}{\rho_k^{\mathbf{t}}}\mathbf{L}_2^T\mathbf{L}_2\right)\mathbf{u}_{k+1} = \frac{1}{\rho_k^{\mathbf{t}}}\mathbf{A}^T\mathbf{b} + \mathbf{L}_1^T\mathbf{t}_{k+1} + \frac{\rho_k^{\mathbf{z}}}{\rho_k^{\mathbf{t}}}\mathbf{L}_2^T\mathbf{z}_{k+1}. \tag{5.19}$$

If we define $\mathbf{M}_k := \frac{1}{\rho_k^{\mathbf{t}}}\mathbf{A}^T\mathbf{A} + \mathbf{L}_1^T\mathbf{L}_1 + \frac{\rho_k^{\mathbf{z}}}{\rho_k^{\mathbf{t}}}\mathbf{L}_2^T\mathbf{L}_2$, then $\forall\ k > 1$, $\mathbf{M}_k$ is invertible for assumption 2. Hence, we can write for each k:

$$\mathbf{u}_{k+1} = \mathbf{M}_k^{-1}\left(\frac{1}{\rho_k^{\mathbf{t}}}\mathbf{A}^T\mathbf{b} + \mathbf{L}_1^T\mathbf{t}_{k+1} + \frac{\rho_k^{\mathbf{z}}}{\rho_k^{\mathbf{t}}}\mathbf{L}_2^T\mathbf{z}_{k+1}\right). \tag{5.20}$$

We observe that the two sequences in the right hand side of (5.20), represented by $\{\mathbf{M}_k^{-1}\}_{k\in\mathbb{N}}$ and by the term in parenthesis, are convergent pointwise (by assumption 3 and by considering the convergence of the sequences $\{\mathbf{t}_k\}_{k\in\mathbb{N}}$ and $\{\mathbf{z}_k\}_{k\in\mathbb{N}}$). By denoting as $\mathbf{u}^*$ the product of the two limits, we have proved that $\mathbf{u}_k \to \hat{\mathbf{u}}^*$.
This concludes the proof. □

Remark 5.2. We point out that this general proof applies also to the algorithm proposed in [34], for which no convergence results can be found in the literature. Moreover, we believe that with a small effort, our convergence result dealing with multiple denoisers can be extended to the ADMM scheme.

Remark 5.3. The fixed-point convergence Theorem 5.1 entails that the iterations enter in a steady-state and does not guarantee that the fixed-point $\hat{\mathbf{u}}^*$ is the minimum $\mathbf{u}^*$ of an implicit defined regularized objective as in (5.3). However, in the experimental part, we have shown that the reached fixed-point $\hat{\mathbf{u}}^*$ is a very good approximation of the desired image $\mathbf{u}$.

5.2.2 On the choice of external and internal denoisers

As regards the choice of the external denoiser, due to the state-of-the-art performances in denoising task reached by deep learning strategies [101, 102], we embed a deep CNN denoiser $\mathsf{D}_\sigma^{\mathrm{CNN}}$ as $\mathsf{D}_\sigma^{\mathrm{ext}}$. Previous studies have already successfully inspected a CNN-based PnP [34, 35] whose CNN denoisers act directly only on the image-domain.
Conversely, our denoiser acts on the image through an operator $\mathbf{L}_1$, which we set equal to the discrete gradient $\mathbf{D} = (\mathbf{D}_h; \mathbf{D}_v)$, where $\mathbf{D}_h, \mathbf{D}_v \in \mathbb{R}^{n\times n}$ are the finite differences discretization of first order derivative operators along the horizontal and vertical axes, respectively.
To investigate the effectiveness provided by the proposed learnt gradient-based denoiser, we consider the case where only the external denoiser is plugged in (thus excluding the internal prior): we label this scheme as GCNN. We will explain in the implementation notes how we have implemented the action of the CNN with respect to the choice of the operator $\mathbf{L}_1$. The general scheme of GCNN is reported in Algorithm 7.

Algorithm 7 GCNN.

Input: α and $\{\rho_k^{\mathbf{t}}\}_{k\in\mathbb{N}}$, $\mathbf{A}$,$\mathbf{D}$, $\mathbf{b}$, $\mathbf{u}_1$, K.
Output: $\mathbf{u}_K$.
for k $= 1 \dots K$ **do**
$\mathbf{t}_{k+1} = \mathsf{D}_{\sigma_k}^{\mathrm{CNN}}(\mathbf{D}\mathbf{u}_k)$
$\mathbf{u}_{k+1} = \underset{\mathbf{u}\in\mathbb{R}^n}{\arg\min} \left\{ \frac{1}{2}\|\mathbf{A}\mathbf{u} - \mathbf{b}\|_2^2 + \frac{\rho_k^{\mathbf{t}}}{2}\|\mathbf{D}\mathbf{u} - \mathbf{t}_{k+1}\|_2^2 \right\}$
end for

We fix as internal denoiser a scheme based on the TV. The properties of edge preserving and noise suppressing of the TV in many image processing applications are well-established. We recall the definition of the TV function:

$$\mathrm{TV}(\mathbf{u}) := \sum_{i=1}^{n} \|(\mathbf{D}\mathbf{u})_i\|_2 = \sum_{i=1}^{n} \left(\sqrt{(\mathbf{D}_h\mathbf{u})_i^2 + (\mathbf{D}_v\mathbf{u})_i^2} \right), \tag{5.21}$$

where $(\mathbf{D}\mathbf{u})_i := ((\mathbf{D}_h\mathbf{u})_i, (\mathbf{D}_v\mathbf{u})_i) \in \mathbb{R}^2$, for $i = 1 \dots n$ denotes the discrete image gradient computed at pixel i along the horizontal and vertical axes, separately. Hence, the function R_2 in (5.3) is set as:

$$R_2 : \quad \mathbb{R}^{2\times n} \to \mathbb{R}$$

$$\mathbf{x} \to \sum_{i=1}^{n} \|\mathbf{x}_i\|_2 \quad \text{with} \quad \mathbf{x}_i \in \mathbb{R}^2, \tag{5.22}$$

where we assume $\mathbf{L}_2 = \mathbf{D}$. Upon this assumption, we remark that, in Algorithm 6, $\mathsf{D}^{\text{int}}_{\gamma_k}$ reads as the proximal map of R_2 with parameter $\gamma_k^2 = \frac{\eta}{\rho_k^{\mathbf{z}}}$.
The method obtained with the described choices of CNN as external denoiser and TV functional as internal denoiser is reported in Algorithm 8. In the following, we will denote it as GCNN-TV.

Algorithm 8 GCNN-TV.

Input: α, β and $\{\rho_k^{\mathbf{t}}\}_{k\in\mathbb{N}}$, $\{\rho_k^{\mathbf{z}}\}_{k\in\mathbb{N}}$, $\mathbf{A}$, $\mathbf{b}$, $\mathbf{u}_1$, K.
Output: $\mathbf{u}_K$.
for k $= 1 \dots K$ **do**
$\mathbf{t}_{k+1} = \mathsf{D}^{\text{CNN}}_{\sigma_k}(\mathbf{D}\mathbf{u}_k)$
$\mathbf{z}_{k+1} = \text{prox}_{R_2}(\mathbf{D}\mathbf{u}_k)$
$\mathbf{u}_{k+1} = \arg\min_{\mathbf{u}\in\mathbb{R}^n} \frac{1}{2}\|\mathbf{A}\mathbf{u} - \mathbf{b}\|_2^2 + \frac{\rho_k^{\mathbf{t}}}{2}\|\mathbf{D}\mathbf{u} - \mathbf{t}_{k+1}\|_2^2 + \frac{\rho_k^{\mathbf{z}}}{2}\|\mathbf{D}\mathbf{u} - \mathbf{z}_{k+1}\|_2^2$
end for

Remark 5.4. By the way, we remark that when $\mathbf{L}_1 = \mathbf{I}$, Algorithm 7 is equivalent to the approach proposed in [34] and denoted as ICNN in the following, whereas we label ICNN-TV the algorithm obtained by adding the TV internal prior to ICNN (following the pattern of Algorithm 8).

Implementation notes

We now refer to particular implementation choices when the proposed algorithms are applied to image deblurring, as considered in our numerical experiments. Here, we refer GCNN-TV and ICNN-TV algorithms. At each iteration k, the minimization problem on the primal variable $\mathbf{u}$ is solved by applying the first order optimality conditions leading to the following linear system:

$$(\mathbf{A}^T\mathbf{A} + \rho_k^{\mathbf{t}}\mathbf{L}_1^T\mathbf{L}_1 + \rho_k^{\mathbf{z}}\mathbf{D}^T\mathbf{D})\mathbf{u}_{k+1} = \mathbf{A}^T\mathbf{b} + \rho_k^{\mathbf{t}}\mathbf{L}_1^T\mathbf{t}_{k+1} + \rho_k^{\mathbf{z}}\mathbf{D}^T\mathbf{z}_{k+1}. \tag{5.23}$$

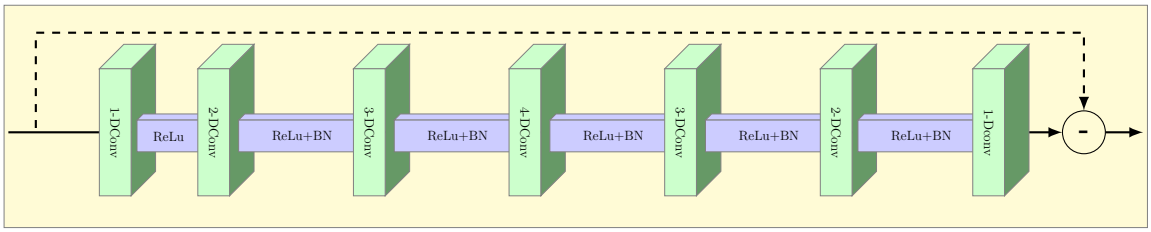
This linear system (5.23) is solvable if the coefficient matrix has full-rank, that is if the following condition holds:

$$\mathbf{Ker}(\mathbf{A}^T\mathbf{A}) \cap \mathbf{Ker}(\mathbf{D}^T\mathbf{D}) \cap \mathbf{Ker}(\mathbf{L}_1^T\mathbf{L}_1) = \{\mathbf{0}\}, \tag{5.24}$$

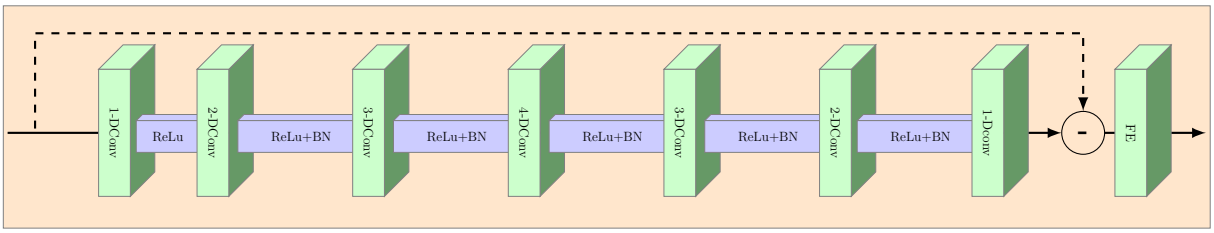
where by **Ker** we denote the null space of a matrix and $\mathbf{0}$ represents the n-dimensional null vector. The condition (5.24) is satisfied both for $\mathbf{L}_1 = \mathbf{I}$ and for $\mathbf{L}_1 = \mathbf{D}$. Indeed, $\mathbf{A}$ represents a blurring operator, which is a low-pass filter, whereas the regularization matrix $\mathbf{D}$ is a difference operator, i.e. a high-pass filter. The solution of (5.23) is given by:

$$\mathbf{u}_{k+1} = (\mathbf{A}^T\mathbf{A} + \rho_k^{\mathbf{t}}\mathbf{L}_1^T\mathbf{L}_1 + \rho_k^{\mathbf{z}}\mathbf{D}^T\mathbf{D})^{-1}(\mathbf{A}^T\mathbf{b} + \rho_k^{\mathbf{t}}\mathbf{L}_1^T\mathbf{t}_{k+1} + \rho_k^{\mathbf{z}}\mathbf{D}^T\mathbf{z}_{k+1}). \tag{5.25}$$

The direct computation of the analytical solution (5.25) requires the inversion of a high dimensional matrix. By assuming periodic boundary conditions $\mathbf{A}^T\mathbf{A}$, $\mathbf{D}^T\mathbf{D}$ and $\mathbf{L}_1^T\mathbf{L}_1$ are BCCB matrices which can be diagonalized by the 2D FFT. Hence, the solution of (5.23) can be efficiently computed. Similarly, $\mathbf{u}_{k+1}$ in Algorithm 7 can be computed by setting $\rho_k^{\mathbf{z}} = 0$.



(a) I-Net architecture scheme [34].



(b) G-Net architecture scheme.

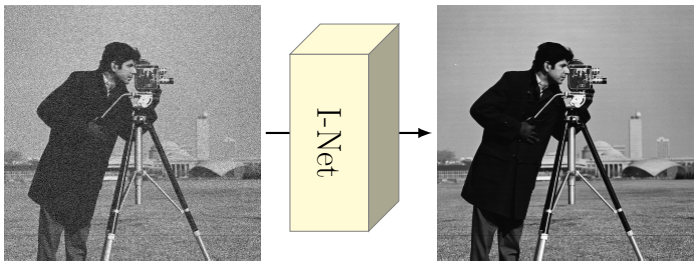
Figure 5.1: I-Net and G-Net architecture schemes. BN represents the batch normalization and m-DConv denotes m-dilated convolution.

Concerning the update of $\mathbf{z}_k$ in Algorithm 8, we observe that it reduces to the solution of n bi-dimensional optimization problems which can be computed in a closed form by using the proximal map of the L_2-norm.

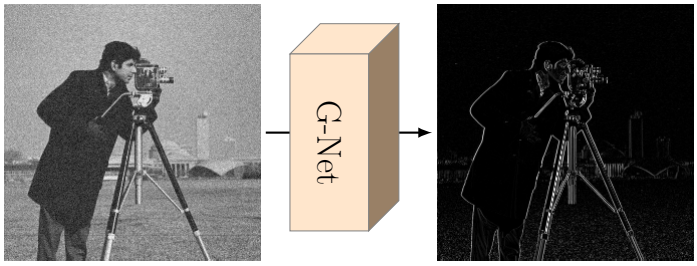
To implement the CNN based external denoiser $\mathsf{D}_\sigma^{\mathrm{CNN}}$ we adopt the widely used DnCNN architecture proposed in [34]. We refer to this architecture, which is shown in Figure 5.1a, as I-Net. It is constituted by seven dilated convolutional layers [110] activated by ReLu functions.
For the CNN training, we consider the *Train400* image dataset [100]. It contains 400 gray-scale natural images of size 180×180 obtained by cropping larger images from the Berkeley Segmentation dataset [111]. We make use in our implementation of the 25 denoisers used in [34], each one trained on a single noise level in the range [2, 50] with step 2. As represented in Figure 5.2a, the I-Net is trained to remove noise from the noisy input images.
Our proposal considers the case $\mathbf{L}_1 = \mathbf{D}$. In this case, we add the linear Feature Extractor (FE) computing the discrete image gradient at the end of the I-Net architecture, thus obtaining the G-Net network depicted in Figure 5.1b. Therefore, in order to compute the iterate $\mathbf{t}_{k+1}$ as in (5.6), the G-Net is trained to give as output the noisy-free gradient image taking as input the noisy images as I-Net (Figure 5.2b). We use the ADAM optimizer (see Appendix A) with the Tensorflow default parameters and we set the epochs number to 150. The correspondence between the iteration k of the algorithms and one of the 25 available networks is performed as in [34].

5.3 Numerical results

In this section, we describe the results obtained by testing the proposed schemes on the task of image denoising and deblurring. We validate our methods both on a synthetic image, characterized by elements of interest for CT medical purposes, and on real CT images. All the ground truth images have values in the range $[0, 255]$.



(a) CNN image denoiser on the image.



(b) CNN denoiser for gradient restoration.

Figure 5.2: Trained schemes for denoising.

Comparisons. Our methods are compared with the baseline TV regularization implemented in the standard ADMM algorithm, which uses the discrepancy principle [112] for the estimation of the regularization parameter, the approach proposed in [34] which is referred to as ICNN in the following, the standard PnP with BM3D and NLM chosen as denoisers and a very recent method [105] which combines a truncated L_1-norm computed on the wavelet operator applied to the signal and BM3D (BM3D-WL1), therefore two internal denoisers. To complete our comparison, we also consider the ICNN-TV algorithm.

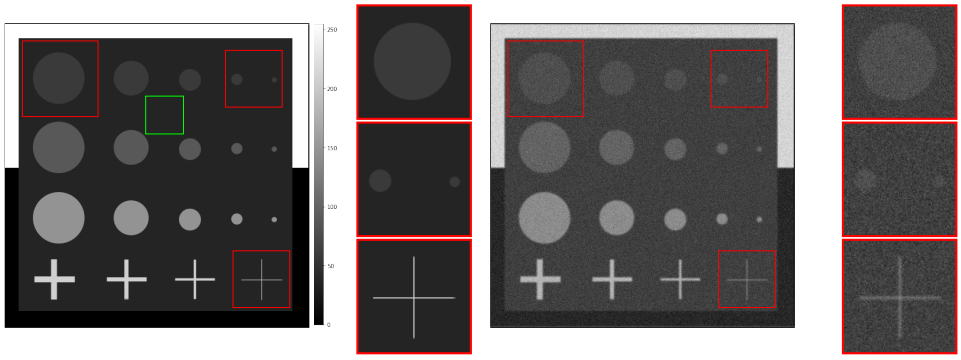
Initialization, parameter and evaluation metrics. For a quality assessment of the results, we create artificially blurred and noisy images from a ground truth (GT) image and we compute the Structural Similarity Index Measure (SSIM) and the Peak Signal-to-Noise-Ratio (PSNR) [113] between the restored image and the ground-truth. Moreover, to quantify noise removal, we compute the standard deviation on uniform regions of interest of the restored images.

For all the proposed algorithms the input parameters α and β are heuristically chosen to compute a solution satisfying the discrepancy principle. The algorithms perform at most $K = 30$ iterations. The first iterate $\mathbf{u}_1$ is initialized as a vector of zeros. Concerning the choice of $\{\rho_k^{\mathbf{t}}\}_{k\in\mathbb{N}}$ and $\{\rho_k^{\mathbf{z}}\}_{k\in\mathbb{N}}$, we have set $\rho_k^{\mathbf{t}} = \rho_k^{\mathbf{z}} = k(1+\epsilon)^k$, with $\epsilon > 0$, satisfying the conditions required in the fixed-point convergence theorem stated in section 5.2.1. All the hyperparameters of the competitors have been fixed in order to provide a solution which satisfies the discrepancy principle.

Results on a synthetic test problem

We start our experiments by considering the numerical simulation acting on the gray-scale 512×512 synthetic image reported in Figure 5.3a. The image is designed to test the algorithms performance in the case of low and high contrast objects, with curved and straight borders: the ground truth

image contains many circles of different diameter but uniform intensity; each row has homogeneous circles, with enhancing contrast, from top to bottom, with respect to the uniform background. The fourth row contains crosses of different thickness and high contrast. To build our test problems we consider the acquisition model in (1.16). Thus, we blur the ground truth image using a Gaussian 15×15 kernel with zero mean and standard deviation $\sigma_{\mathbf{H}} = 1.2$, then we introduce AWGN with standard deviation $\sigma_{\mathbf{e}}$ in $\{\frac{10}{255}, \frac{15}{255}, \frac{20}{255}\}$. In Figure 5.3b we show the corrupted image obtained with $\sigma_{\mathbf{e}} = \frac{15}{255}$. In Figure 5.3a and 5.3b, we also depict three close-ups on the regions bounded by red squares.



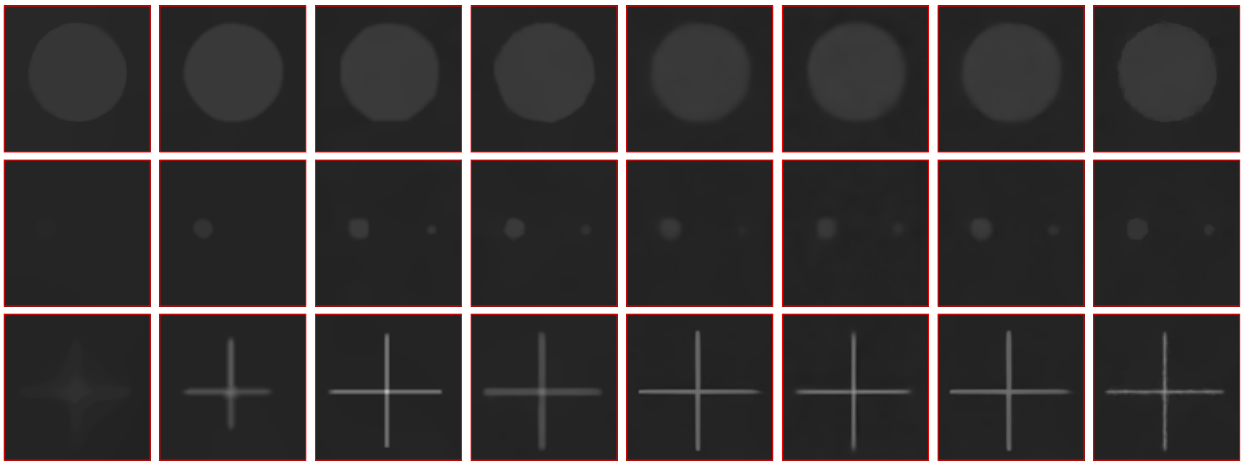
(a) Ground truth (b) Corrupted

Figure 5.3: Ground truth gray-scale test image and a simulated degraded acquisition. In (a) the green square highlights the uniform patch used to evaluate ROI-std. In (a) and (b) three close-ups (red boxes) are depicted alongside the images.

In Figure 5.4, for each method we report the three restored zooms in the same range of gray levels. For what concerns the low-contrast circles, reported in the first two rows, it is evident that the hybrid approaches (such as BM3D-WL1, ICNN-TV and GCNN-TV) outperform the other algorithms which exploit only one denoiser (TV, NLM, BM3D,ICNN, GCNN). Indeed, TV (Figure 5.4a) and NLM (Figure 5.4b) struggle to retrieve the small circles, whereas BM3D deforms the shape of the objects (Figure 5.4c). We highlight that the smallest circle is visible in the ICNN reconstruction (Figures 5.4e) and it is further enhanced in the GCNN restoration 5.4f. Focusing on the restoration of an object, the one-pixel thick cross, with a different shape and contrast, we observe that BM3D, ICNN and GCNN achieve the highest enhancement (see the last row of Figure 5.4). However we remark that, even in this case, TV and NLM tend to suppress very thin details.

In Figure 5.5 we plot the pixel intensities of a horizontal image row passing through all the lowest-contrasted circles, to better inspect the effects of adding the TV internal prior to the ICNN and GCNN schemes on the most challenging objects. The plot in Figure 5.5a reflects the typical loss-of-contrast drawback of the TV prior, oversmoothing the two smallest circles. Adding the TV prior to ICNN and GCNN algorithms removes the residual noise, especially visible in the largest circle, while enhancing the edges.

To test the robustness of the proposed models with respect to the noise, we analyze the results, reported in Table 5.1, obtained by the considered methods when different variances of the AWGN are considered. We observe that, in terms of PSNR, the GCNN method gets the best values in



(a) TV (b) NLM (c) BM3D (d) BM3D-WL1 (e) ICNN (f) GCNN (g) ICNN-TV (h) GCNN-TV

Figure 5.4: Three close-ups for each reconstruction by different methods obtained for the synthetic image.

all the cases, thus confirming the effectiveness of the proposed CNN denoiser defined on the image gradient domain. When we introduce the contribution of the TV-based internal prior, the PSNR values decrease, even if the global denoising effect due to TV is visually evident, as previously underlined. To confirm this, we report in Table 5.1 the standard deviation (ROI-std) computed on the constant region marked by the green bounding square in Figure 5.3a. The TV and NLM methods always have the lowest values, whereas the proposed hybrid approaches ICNN-TV and GCNN-TV are more effective in case of high noise.

Results on real CT medical images

We now consider two X-ray Computed Tomography images to compare the effectiveness of the proposed schemes. In order to illustrate the advantages of our proposals, according to their features highlighted in the synthetic case, we examine a head and chest CT images containing small and low-contrasted details.

CT head image for epidural hemorrhage detection

The considered head tomographic image is downloaded from an open source dataset[1]. It shows an intracranial hemorrhage, which requires a rapid and intensive medical treatment based on the accurate localization of the blood in the CT image obtained by segmentation algorithms (represented as the red region in Figure 5.6a). If the image is severely corrupted, the segmentation procedure may fail. As an example, after blurring the ground truth image with a Gaussian kernel of size 15×15 and standard deviation 0.5 and adding AWGN with standard deviation 25, we compute the segmentation mask by an online open source software [2]. The segmented region is shown in red in

[1] `https://www.kaggle.com/vbookshelf/computed-tomography-ct-images`

[2] `http://brain.test.woza.work/`

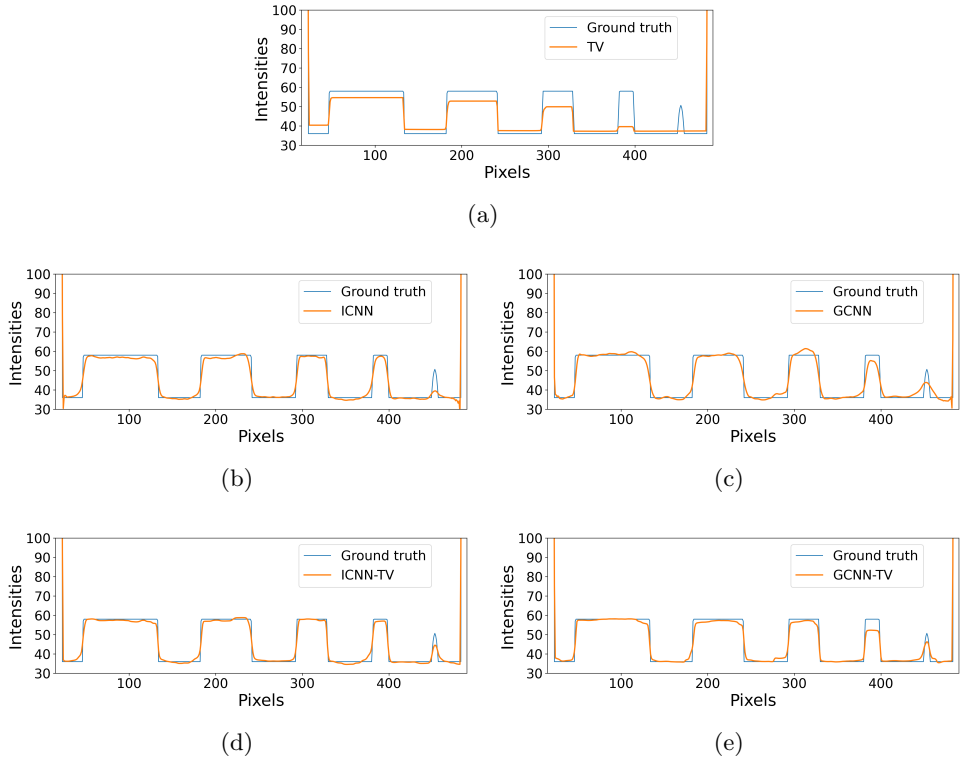


Figure 5.5: Intensity line profiles on the 90th row cutting the lowest contrasted circles. The blue and orange lines represent the ground truth and the restored image profiles for different methods, respectively.

Figure 5.6b. To highlight the importance of deblurring and denoising the image before segmenting it, we show the red mask computed on one restored image in Figure 5.6c.

In Figure 5.7 we report three close-ups for each method. The first one highlights the central part of the head CT image containing blood vessels, whereas the second zoom shows a portion of the cerebral cortex with sulci. The third zoom of the figure focuses onto the epidural hemorrhage (pointed by the magenta arrow). In Table 5.2 we report the PSNR computed between the restored image and the ground truth, and the Jaccard similarity coefficient (Jac) between the masks computed on the ground truth and the restored images. By a visual comparison, we observe that TV, NLM, BM3D-WL1 output images look too smooth and blocky whereas the BM3D deforms the anatomical contours. We highlight that the GCNN method accurately restores the vessels and sulci borders and it gets the highest PSNR value, reflecting the effectiveness of the gradient-based regularization. As regard the Jaccard values, the best ones are achieved by the hybrid frameworks (i.e. ICNN-TV and GCNN-TV), where the smoothing effect of the TV-based denoiser improves the border detectability, making the restored images suitable for segmentation tasks.

	AWGN of $\sigma_e = \frac{10}{255}$		AWGN of $\sigma_e = \frac{15}{255}$		AWGN of $\sigma_e = \frac{20}{255}$	
	PSNR	ROI-std	ROI-PSNR	ROI-std	PSNR	ROI-std
TV	30.8085	**0.0271**	28.6664	**0.0507**	27.4028	**0.0772**
NLM	32.6266	0.0896	31.3772	0.1122	30.1042	**0.1382**
BM3D	32.1221	0.3657	31.3283	0.5785	30.4806	0.7281
BM3D-WL1	31.7616	0.3974	30.7724	0.5802	30.2951	0.7779
ICNN	34.1838	0.4398	33.0519	0.5555	32.9788	0.7851
GCNN	**34.7078**	0.4749	**33.9640**	0.6568	**33.2446**	0.8189
ICNN-TV	32.3531	0.4081	31.3775	0.4798	30.4499	0.5553
GCNN-TV	33.2648	0.1706	31.7743	0.2512	30.6453	0.3129

Table 5.1: Measures computed on restored images varying the standard deviation of the AWGN. The two best PSNR and ROI-std (standard deviation computed inside the green square in Figure 5.3a) values for each AWGN are highlighted in blue and green, respectively. The first best is highlighted in bold.

(a) GT + mask (b) Corrupted + mask (c) Restored + mask

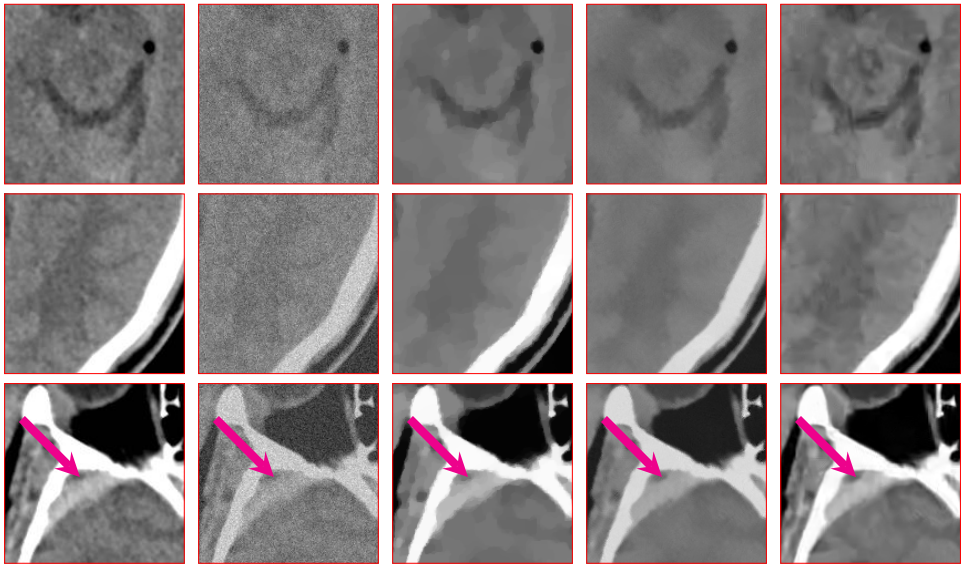
Figure 5.6: Head tomographic image with epidural hemorrhage. Computed masks are coloured red.

Restoration of low-dose CT real chest image

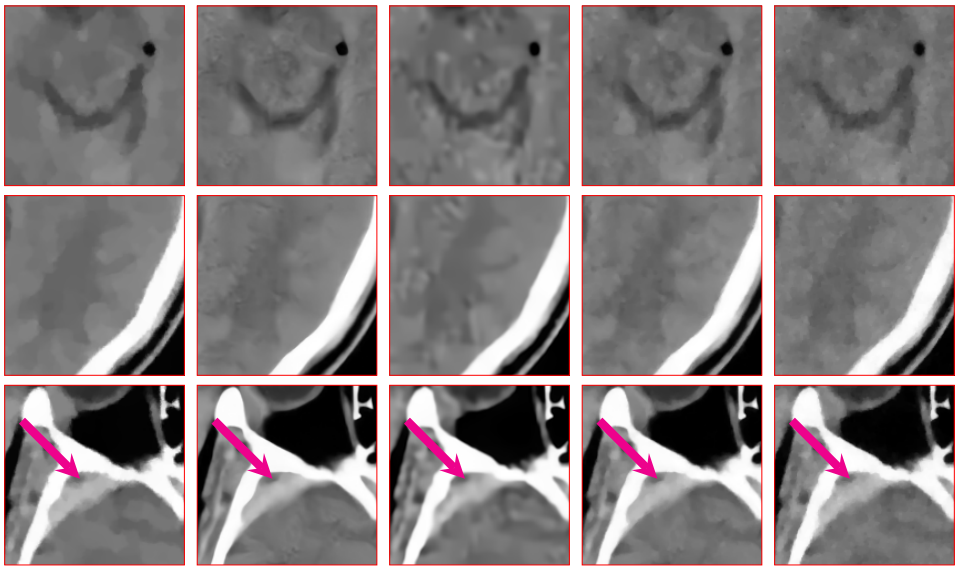
We now consider a Computed Tomography open source dataset[3] of real chest images. In Figure 5.8a we focus on one image (ID: 0005) of the dataset. We point out that it contains many different objects, varying in size, dimension and gray intensity. To simulate a low-dose CT reconstructed image, which is characterized by high noise, after blurring the image by using a Gaussian kernel of dimension 15×15 with standard deviation 0.5, we add AWGN with high standard deviation equals to 25. In Figure 5.8b we show the very noisy corrupted image where small and low-contrasted details are not well detectable.

In Figure 5.9 we report three close-ups of the restorations showing different details of the image. In the first close-up we observe that in some cases the borders of the ascending aorta and superior vena cava sections pointed by the arrow are not well distinguishable as in the ground truth image. In particular, we notice that the GCNN method produces the best image. The second crop contains thin vessels immersed in the dark pulmonary background. The images obtained with TV, NML and BM3D-WL1 algorithms are too smooth and some details are hardly visible. In the BM3D and

[3] `https://www.kaggle.com/kmader/siim-medical-images`



(a) GT (b) corrupted (c) TV (d) NLM (e) BM3D



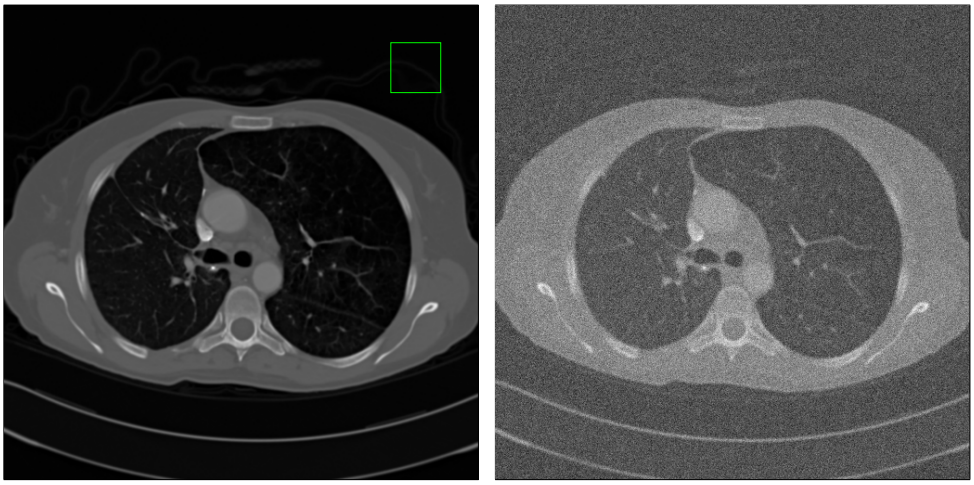
(f) BM3D-WL1 (g) ICNN (h) GCNN (i) ICNN-TV (j) GCNN-TV

Figure 5.7: Three close-ups for each reconstruction by different methods obtained for the head CT image. The magenta arrows highlight the epidural hemorrhages.

ICNN-based output images the circular sections of the vessels are distorted into triangular shapes, whereas the images obtained with gradient-based CNN restore very well the path of the main vessels, without oversmoothing. In the third row, the close-ups show that only GCNN and GCNN-TV well recover the circular shape of the vertebral canal and GCNN outperforms the competitors in identifying the transverse process edges (Figures 5.9h and 5.9j).
To deeper analyse the improvement given by the gradient-based CNN over the image-based one, we plot in Figure 5.10 the profiles relative to the green segments depicted in Figure 5.8a over the first and third crops. The first plot (Figure 5.10a) refers to a large homogeneous object and it is evident

	TV	NLM	BM3D	BM3D-WL1	ICNN	GCNN	ICNN-TV	GCNN-TV
PSNR	30.6781	28.4723	31.7320	31.9917	33.0800	**33.5881**	32.2843	32.0872
Jac	0.9471	0.8827	0.9313	0.9500	0.9387	0.9398	**0.9557**	0.9504

Table 5.2: PSNR and Jaccard computed on restored image. The two best PSNR and Jaccard values are highlighted in green and blue, respectively. The first best is highlighted in bold.



(a) Ground truth (b) Corrupted

Figure 5.8: Low-dose CT chest image (`ID: 0005`). In (a) the green square highlights the uniform patch used to evaluate ROI-std.

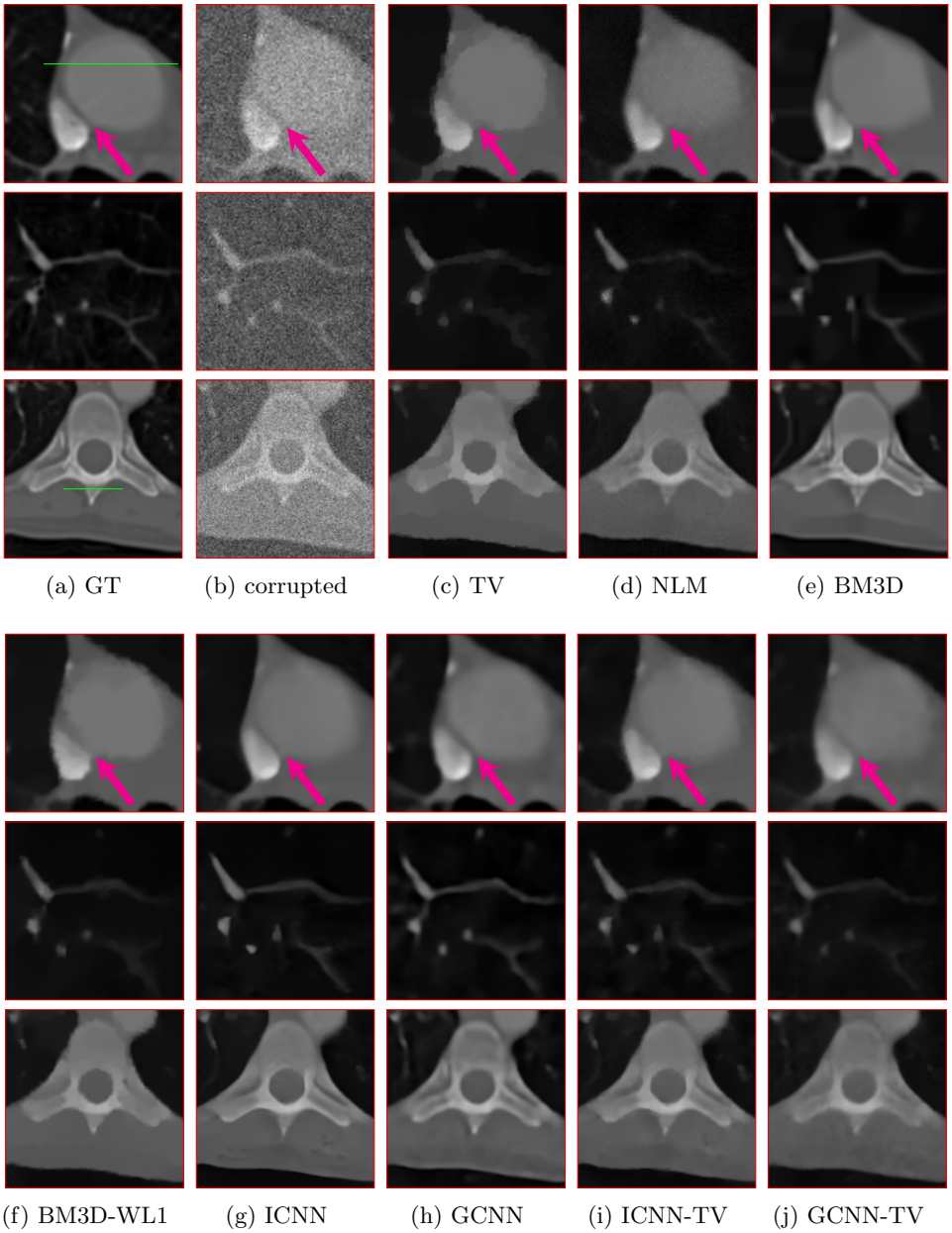
that the GCNN red line better fits the blue line corresponding to the ground truth and that the orange ICNN profile oversmoothes. The profile over the spinous process (Figure 5.10b) highlights that GCNN better restores thin objects. We can conclude for the restoration of this image that the use of a gradient-based CNN denoiser has advantages such as a better enhancing of the objects contours and the preservation of small details, over the use of an image-based CNN denoiser.

Finally, to measure the reconstruction quality and the residual noise, we compute the PSNR and SSIM measures on the whole image and the standard deviation on a flat region indicated by the green square in Figure 5.8a. From the Table 5.3, we observe that the GCNN method attains both the best PSNR and SSIM. The BM3D algorithm achieves the second best PSNR but it often deformates the curve boundary contours of the objects (as in Figure 5.9e). Regarding the ROI-std measure, as expected, the TV method gets the lowest standard deviation on the region of interest. Moreover, we observe that the addition of TV as internal prior in the CNN-based methods considerably lowers the standard deviation values, as confirmed by ICNN-TV and GCNN-TV columns.

At last, Figure 5.11 generalises the results of Table 5.2. We have in fact executed the GCNN and ICNN algorithms on 100 images from the whole chest dataset and computed the boxplots relative to the PSNR (Figure 5.11a) and the SSIM (Figure 5.11b) metrics. These statistics validate the results discussed on one single image and confirm that GCNN outperforms ICNN.

	TV	NLM	BM3D	BM3D-WL1	ICNN	GCNN	ICNN-TV	GCNN-TV
PSNR	32.1727	30.9899	34.7675	32.9104	34.1673	**35.0309**	34.0946	33.5789
SSIM	0.9297	0.9129	0.9499	0.9358	0.9474	**0.9546**	0.9466	0.9443
ROI-std	**0.1746**	0.3017	0.6569	0.5816	1.1136	1.2366	0.2844	0.3460

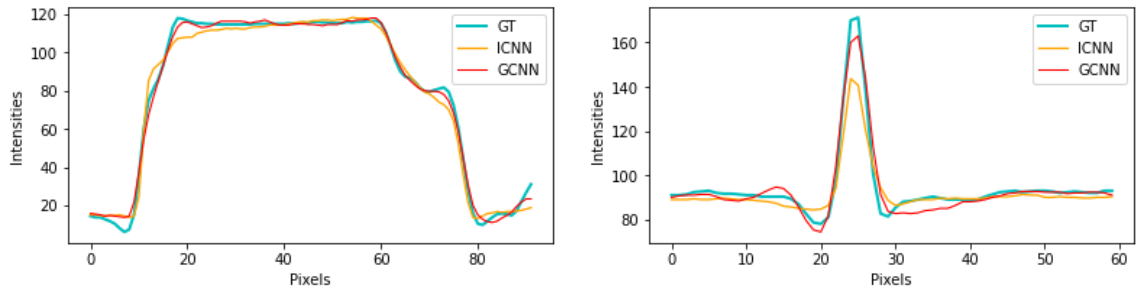
Table 5.3: Standard deviation computed on the region of interest inside the green square in Figure 5.8a, for the Low-Dose CT chest images.



(a) GT (b) corrupted (c) TV (d) NLM (e) BM3D

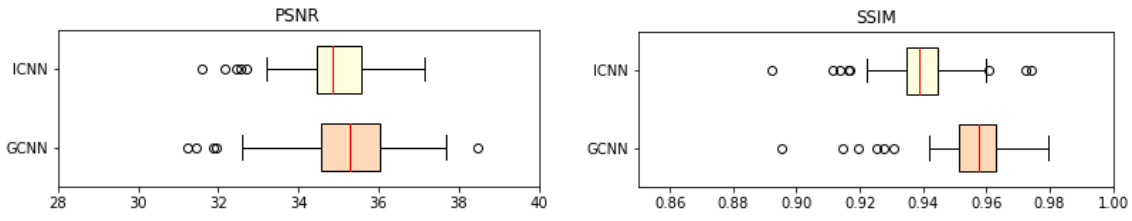
(f) BM3D-WL1 (g) ICNN (h) GCNN (i) ICNN-TV (j) GCNN-TV

Figure 5.9: Three close-ups for each reconstruction by different methods obtained for the chest low-dose CT image. The magenta arrows highlight a region of interest.



(a) (b)

Figure 5.10: Intensity line profiles on the horizontal lines depicted in Figure 5.9a, over the aorta (left) and on the spinous process of the vertebra (right). The blue, orange and red lines represent the ground truth, the ICNN and the GCNN restored image profiles, respectively.



(a) (b)

Figure 5.11: Boxplots of the PSNR values (a) and SSIM values (b) computed on 100 chest images by ICNN algorithm (yellow ones) and GCNN algorithm (orange ones).

Part III

Learning models for image and time series inverse problems

Chapter 6

DeepCEL0 for super resolution in fluorescence microscopy

The spatial resolution of images acquired by fluorescence microscopy refers to the shortest distance at which two fluorescent entities are perceived separately by the camera system.
As a consequence of the light diffraction phenomena, lens with a uniformly illuminated circle aperture generate patterns known as Airy disks [114], namely the fluorescent emitters to be imaged are represented as blobs and not as isolated spots of light. The ability of the microscope to distinguish two relatively close entities is bounded by the well-known diffraction limit [115] that represents an intrinsic constraint of the optical acquisition device. More precisely, according to the Abbe's criterion, the smallest resolvable distance by a light microscope corresponds roughly to half the optical wavelength, that is about 200 nanometers (nm) [116], thus compromising the direct observation of structures at nanoscale such as proteins, microtubules, mitochondria and less complex molecules.
In the last decades, super-resolution microscopy techniques have revolutionized light microscopy biological imaging allowing biologists to see beyond the diffraction limit [117]. Among them we mention Single-Molecule Localization Microscopy (SMLM) strategies, which retrieve the localization maps by sequentially activating and imaging a small percentage of photoswitchable fluorophores (emitters) in the Field of View (FOV). More precisely, these SMLM techniques provide a stack of diffraction-limited frames, containing blobs (Airy disks), modeled by Gaussian PSFs [118]. Each frame is then analyzed separately with the aim of providing high precision localization maps of the emitters. Finally, after the individual processing is performed, all the frames are re-combined together to finally obtain a unique super-resolved image overcoming the diffraction limit.
The sparser the set of activated emitters per frame, the more precise the localization is. However, considering sparse frames takes a longer acquisition time thus limiting the ability to capture fast dynamics within live specimens. Conversely, a high density of activated emitters negatively affects the quality of the super-resolved image in terms of localization precision. Indeed, localization in high-density settings, that are characterized by overlapping PSFs, represents a challenging task for all the existing sophisticated localization software tools [119, 120, 121].

In this chapter, by $\mathbf{b} \in \mathbb{R}_+^m$ we refer to the vectorized frame acquired by PALM/STORM techniques representing a sparse set of activated molecules on a coarse pixel-grid of dimension

$n_r \times n_c$ such that $m = n_r \cdot n_c$, whereas by $\mathbf{u} \in \mathbb{R}^n_+$ we denote the localization map referred to $\mathbf{b}$, that is a vectorized image defined on a L-time thinner pixel-grid of dimension $N_r \times N_c$, such that $N_r = L \cdot n_r$ and $N_c = L \cdot n_c$ and $n = N_r \cdot N_c$.
The acquisition of the PALM/STORM diffraction-limited image $\mathbf{b}$ reads:

$$\mathbf{b} = \mathcal{N}(\mathbf{SHu}), \tag{6.1}$$

where by $\mathcal{N}$ we denote the random variable modeling the presence of both data dependent Poisson noise and AWGN with mean zero and standard deviation $\sigma_{\mathbf{e}}$, which are typical aberrations in fluorescence imaging [122, 123, 124]. The Gaussian blurring operator $\mathbf{H}$ of standard deviation $\sigma_{\mathbf{H}}$ models the presence of the typical Airy disk patterns (blobs) in the LR acquisition and the operator $\mathbf{S}$ refers to the downsampling operator linking the HR localization map, represented on a fine pixel grid, to the PALM/STORM acquired image, defined on a L-time coarser pixel grid.

Let $\mathcal{D} := \{(\mathbf{b}_k, \mathbf{u}_k)\}_{k=1\dots K}$ be a set of K 2D-image pairs, where $\mathbf{b}_k \in \mathbb{R}^m$ denotes the simulated diffraction-limited frame and $\mathbf{u}_k \in \mathbb{R}^n$ denotes its HR localization map. In this chapter we develop a learning-based strategy solving (6.1) by training a particular deep architecture $f_{\boldsymbol{\theta}} : \mathbb{R}^n \to \mathbb{R}^n$ with weights $\boldsymbol{\theta}$. We consider the following regularized loss function:

$$\boldsymbol{\theta}^* \in \underset{\boldsymbol{\theta}\in\mathbb{R}^p}{\arg\min} \frac{1}{K} \left(\sum_{k=1}^{K} \mathcal{L}(f_{\boldsymbol{\theta}}, \mathbf{b}_k, \mathbf{u}_k, \mathbf{A}^\dagger) + \mathcal{R}(f_{\boldsymbol{\theta}}, \mathbf{b}_k, \mathbf{A}^\dagger) \right), \tag{6.2}$$

where $\mathcal{L}$ denotes the penalty function, $\mathcal{R}$ the regularizer and by $A^\dagger : \mathbb{R}^m \to \mathbb{R}^n$ we refer to a coarse super resolution algorithm. Once provided the final set of trained weights $\boldsymbol{\theta}^*$ and a PALM/STORM acquisition $\mathbf{b}$, an approximation $\mathbf{u}^*$ of the HR localization map $\mathbf{u}$ is obtained by computing $f_{\boldsymbol{\theta}^*}(A^\dagger(\mathbf{b}))$.

Contribution

This chapter is based on the publication [125] where we propose a learning-based method termed DeepCEL0, to perform high precision molecule localization. The method merges the main advantages of [126, 127], two state-of-the-art approaches belonging to the class of learning and variational methods, respectively. More specifically, we consider the encoder-decoder architecture proposed in [126]. We add positivity constraints to the model through the insertion of a ReLU layer in the network architecture. Moreover, we propose to include in the loss the CEL0 penalty [128] which has been shown to be an effective regularizer for variational methods in SR fluorescence microscopy [127]. Thus, DeepCEL0 supplies a quite fast and parameter-free deep-learning method capable to provide high precision localization maps.

6.1 A short survey on super resolution in fluorescence microscopy

Approaches based on diverse rationale, from blinking statistics [129, 130, 131], standard Gaussian fitting or centroid estimation [132] and subtraction of the model PSF [133, 134, 135] to on-grid [136, 137, 138, 127, 139] and off-the-grid [140, 141] methods with sparsity-promoting regularizer, have been proposed to retrieve the localization map $\mathbf{u}$ from the high-density PALM/STORM acquisition

b.
More specifically, the idea behind on-grid approaches is the creation of a fine-grained grid to model the locations of activated emitters. Among them, we find regularized variational approaches such as CEL0 [127] which exploits the Continuous Exact ℓ_0 regularizer [128] as regularizer.
In contrast to on-grid methods, the so-called off-the-grid super resolution methods, such as ADCG [140] and TVSTORM [141], work in a greedy way over the space of measures adding new molecules at each iteration and then optimize their positions and/or amplitudes in the continuum.
In the last few years, many deep learning-based approaches exploiting CNNs have been proposed. Among the methods belonging to this class we mention DeepSTORM [126] which is trained on artificially generated frames, and tested directly on experimental data, thus avoiding to collect a huge amount of training samples related to the particular experiment under study. Another deep learning-based approach named DECODE [142] has been recently developed. DECODE is able to simultaneously detect and localize the emitters and to predict both the probability of detection and the uncertainity of localization for each emitter in high-density data.
For the sake of clarity, in the following paragraphs, we briefly review the CEL0 and the DeepSTORM models for super resolution in fluorescence microscopy.

Super resolution fluorescence microscopy using CEL0. It is worth noting that the unknown high-precision localization map $\mathbf{u}$ in (6.1) is sparse. Therefore, a quite common approach, to provide an estimate $\mathbf{u}^*$ of the unknown $\mathbf{u}$, exploits the variational framework by considering sparsity constraints through the ℓ_0 penalization. In [127] the authors consider the continuous relaxation $\Phi_{\text{CEL0}} : \mathbb{R}^n \to \mathbb{R}$ of the ℓ_0 penalization, which is named as CEL0 penalizer [128] and reads:

$$\Phi_{\text{CEL0}}(\mathbf{u}) := \sum_{i=1}^{n} \lambda_{\text{CEL0}} - \frac{\|\mathbf{c}_i\|}{2}\left(|\mathbf{u}_i| - \frac{\sqrt{2\lambda_{\text{CEL0}}}}{\|\mathbf{c}_i\|}\right)^2 \mathbf{1}_{V_i}, \tag{6.3}$$

where by $\mathbf{1}_{V_i}$ we denote the characteristic function of the set $V_i := \{\mathbf{u}_i \in \mathbb{R} \big| |\mathbf{u}_i| < \frac{\sqrt{2\lambda_{\text{CEL0}}}}{\|\mathbf{c}_i\|}\}$, by c_i we denote the i-th column of the matrix $\mathbf{SH}$ for $i = 1 \dots n$, whereas the positive scalar λ_{CEL0} balances the strength of the sparsity induced by the CEL0 penalizer.
Therefore, the CEL0-based method retrieve the high precision localization map $\mathbf{u}^*$ by solving the following unconstrained optimization problem:

$$\mathbf{u}^* \in \underset{\mathbf{u}\in\mathbb{R}^n}{\arg\min} \frac{1}{2}\|\mathbf{SHu} - \mathbf{b}\|_2^2 + \Phi_{\text{CEL0}}(\mathbf{u}) + \mathbf{1}_{\geq 0}(\mathbf{u}), \tag{6.4}$$

where $\mathbf{1}_{\geq 0}(\cdot)$ is the characteristic function of the positive octant of $\mathbb{R}^n$ constraining the computed estimation $\mathbf{u}^*$ to have positive entries.
For its numerical solution, in [127] the authors make use of the iterative reweighted ℓ_1 (IRL1) strategy [143] which is tailored to handle non-smooth non-convex optimization problems and ensures the convergence to a critical point of (6.4) [128].
Despite being among the most effective methods in the field of SMLM, the accuracy of CEL0 reconstructions strictly depends on the choice of the regularization parameter balancing the ℓ_2 and the sparsity-promoting terms, thus drastically limiting its usage in real experiments. Moreover, the IRL1 strategy makes the whole reconstruction process much slower than learning-based methods.

Super resolution fluorescence microscopy using DeepSTORM. In [126] the authors have introduced a fast and parameter-free deep learning-based method termed as DeepSTORM, which makes use of a CNN to provide the estimate $\mathbf{u}^*$. More specifically, the training loss is a particular instance of the one reported in (6.2) and reads:

$$\boldsymbol{\theta}^* \in \underset{\boldsymbol{\theta}\in\mathbb{R}^p}{\arg\min} \frac{1}{K} \left(\sum_{k=1}^{K} \|\mathbf{G} f_{\boldsymbol{\theta}}(\mathrm{NN}(\mathbf{b}_k)) - \mathbf{G}\mathbf{u}_k\|_2^2 + \|f_{\boldsymbol{\theta}}(\mathrm{NN}(\mathbf{b}_k))\|_1 \right), \tag{6.5}$$

where by NN we denote the coarse Nearest Neighbour super resolution algorithm, by $\mathbf{G} \in \mathbb{R}^{n\times n}$ we refer to a Gaussian blurring with standard deviation $\sigma_{\mathbf{G}} = 1$. Once provided the final set of trained weights $\boldsymbol{\theta}^*$ and a PALM/STORM acquisition $\mathbf{b}$, an approximation $\mathbf{u}^*$ of the HR localization map $\mathbf{u}$ is obtained by computing $f_{\boldsymbol{\theta}^*}(\mathrm{NN}(\mathbf{b}))$.
As far as the training set is concerned, one of the main novelties of DeepSTORM is that it can provide good performances on real images even if it is trained only on a synthetic dataset created using an ImageJ plug-in called ThunderStorm [144]. Despite being a fast and parameter-free super-resolution microscopy algorithm able to manage high-density data, DeepSTORM is not capable to reconstruct the localization maps with high precision.

6.2 The proposed DeepCEL0

The proposal developed in [125] combines the learning-based approach DeepSTORM and the sparsity-constrained variational approach CEL0. In particular, we aim at providing a method which preserves the main advantages of both, namely the ability of CEL0 to retrieve high precision localization maps and the fast and parameter-free computation provided by DeepSTORM. Therefore, following the idea of [126], we train a CNN architecture $f_{\boldsymbol{\theta}} : \mathbb{R}^n \to \mathbb{R}^n$ based on the following regularized loss function:

$$\boldsymbol{\theta}^* \in \underset{\boldsymbol{\theta}\in\mathbb{R}^p}{\arg\min} \frac{1}{K} \left(\sum_{k=1}^{K} \|\mathbf{G} f_{\boldsymbol{\theta}}(\mathrm{NN}(\mathbf{b}_k)) - \mathbf{G}\mathbf{u}_k\|_2^2 + \Phi_{\mathrm{CEL0}}(f_{\boldsymbol{\theta}}(\mathrm{NN}(\mathbf{b}_k))) \right). \tag{6.6}$$

This loss presents two main differences, if compared to (6.5). First, the ℓ_1-regularizer is replaced by the CEL0 regularizer in order to guarantee better sparse reconstructions. Second, we consider a non-negativity promoting CNN architecture $f_{\boldsymbol{\theta}}$ which is a slightly modified version of the CNN architecture used in [126], in order to force non-negativity constraints on the reconstructed solutions.

A non-negativity promoting deep architecture

The considered CNN architecture, denoted as $f_{\boldsymbol{\theta}}$ in (6.6), is a modified version of the one proposed in [126]. For the sake of brevity, we call convolutional layer the composition of convolutional filters with a batch normalization layer followed by a ReLu non-linearity as activation function (see Appendix A). The original architecture is an encoder-decoder and it is composed of seven convolutional layers. Each layer uses 3×3 kernels of different depth equal to 32, 64, 128 and 512,

respectively. In the encoder part, the filters' depth increases, and a 2×2 max-pooling is used as downscaling operator to compress the features. In the decoder part, the layers are interleaved with a nearest neighbour upsampling operator and the filters' depth decreases. At the end of the network, another layer is added to compute the pixel-wise prediction. This layer is a 1×1 convolutional filter. In the original implementation, this last layer uses a linear activation function, i.e., the identity. In order to induce non-negativity constraints to the computed solution, we replace the activation layer with ReLu.

Training set and implementation notes

In the experimental section, we evaluate how our method performs when dealing with an upsampling factor L equal to 4. According to this choice, we now describe the considered synthetic training set $\mathcal{D} = \{(\mathbf{b}_k, \mathbf{u}_k)\}_{k=1\dots K}$. In the following, we refer to $\mathbf{b}_k$ as the input image and to $\mathbf{u}_k$ as the target image, respectively.
As well as for DeepSTORM [126], we generate a synthetic dataset made up of 20 high density images. The emitters are positioned on a FOV of size 64×64 pixels such that each pixel has size of 100 nm. We extract from these high density images, $K = 10000$ patches of size 26×26. By projecting the emitter positions on a 4-times thinner pixel grid, we build the target images $\mathbf{u}_k$ of size 104×104.
The input images $\mathbf{b}_k$ for $k = 1 \dots K$ are constructed corrupting the synthetic images in order to simulate the experimental conditions. More precisely, we first blur all the 26×26 patches with a discrete Gaussian kernel and add Poisson and Gaussian noisy components. Then, these corrupted patches are upsampled by a factor equals 4 through the NN interpolation algorithm.
We stress that, in real applications, the standard deviation of the PSF considered, if unknown, can be estimated using the Abbe's criterion which requires the light wavelength and the numerical aperture of the optical device used for acquiring the experimental data under study. Furthermore, the amount of noise can be either calculated directly from the microscope and detector characteristics or even through several mathematical techniques [145, 146].

We use this synthetic dataset to train the proposed DeepCEL0 by minimizing the loss function defined in (6.6). We train the network for 100 epochs on batches containing 16 samples using Adam optimizer and setting the learning rate equal to 0.001.
In (6.6), we set $\mathbf{G}$ as a fixed Gaussian blurring operator, whose standard deviation $\sigma_{\mathbf{G}}$=1. Moreover, we remark that the contribution of the CEL0 regularization term can be weighted using the parameter λ_{CEL0}. It is worth noting that in this deep learning-based framework the parameter λ_{CEL0} has not the same meaning of the regularization parameter in the variational approach (6.4). Indeed, in the former case it does not directly correspond to a degree of sparsity of the computed solution. In our experiments, we tested different values for λ_{CEL0} but we observed that setting it to 100 provides outstanding results for all the tests.

6.3 Numerical results

We now apply the proposed DeepCEL0 both on simulated and real diffraction-limited data, in order to highlight its effectiveness in the field of fluorescence microscopy.

Evaluation metrics. We assess the quality of the high resolution localization maps provided by our method and the competitors both on synthetic and real PALM/STORM images through quality metrics and visual inspections. When the ground-truth (GT) images are available, the performances are evaluated by pairing the GT molecules with the estimated ones: a match between a GT and an estimated molecule is created when the distance between their localizations is lower than a set tolerance δ, whose standard values, if expressed in terms of pixels, are 2, 4 and 6. Such a tolerance δ is chosen lower than the Full-width at Half Maximum (FWHM) of the estimated Gaussian PSF modeling the Airy pattern. In the following, the matched estimated molecules up to the given tolerance are defined as True Positive (TP) molecules; the remaining estimated molecules are referred to as False Positive (FP) molecules; and finally, the GT molecules with no match are categorized as False Negative (FN) molecules. Beyond the estimated molecules, it is important to take into account the pixels of the GT images not corresponding to any molecule, which are labelled as True Negative (TN) molecules. The performances are assessed by computing the following evaluation metrics:

$$\text{Jaccard}(\%) = \frac{\text{TP}}{\text{TP} + \text{FP} + \text{FN}} \times 100, \tag{6.7}$$

$$\text{Sensitivity}(\%) = \frac{\text{TP}}{\text{TP} + \text{FN}} \times 100, \tag{6.8}$$

$$\text{Specificity}(\%) = \frac{\text{TN}}{\text{TN} + \text{FP}} \times 100. \tag{6.9}$$

In order to quantify the level of corruption in the simulated data, we consider the signal-to-noise ratio (SNR) in decibel (dB).

Comparisons. We compare our approach with the state-of-the-art algorithms CEL0 and DeepSTORM on both high density synthetic and real SMLM PALM/STORM low resolution images whose level of corruption ranges from SNR = 15dB to SNR = 10 dB.

Finally, in order to inspect the role of the non-negativity constraints, we train the non-negativity promoting CNN used in DeepCEL0 by considering the ℓ_1 regularized loss function in (6.5). In the following, the method is referred to as DeepSTORM-ReLu.

Furthermore, we run DeepSTORM and DeepSTORM-ReLu by selecting the regularization parameters values in the range [1,200]. In brief, DeepSTORM does not achieve any significant improvement in the reconstructions of the localization maps. Therefore, for all the experiments, the regularization parameter is fixed equal to 1, as in its standard formulation [126]. Instead, for what concerns DeepSTORM-ReLu, we found the best results by setting the regularization parameter equal to 10. So, in the following sections, only the best results obtained for DeepSTORM-ReLu are reported.

Localization on synthetic test images with theoretical PSF

DeepSTORM algorithm could provide a fast SR image reconstruction, although it is not tailored to provide high precision localization maps of the emitters. Conversely, CEL0 is designed to localize the molecules but presents two main flaws: the method is strongly dependent on the choice of the regularization parameter and the overall computation process is largely slower than the one provided by DeepSTORM. Once the network has been trained, the proposed DeepCEL0 and DeepSTORM share a comparable computational time for SR image reconstruction. Moreover, as we demonstrate in the following, DeepCEL0 is also able to effectively localize the molecules.

First of all, to validate the ability of the proposed method to discriminate two or more neighboring emitters, we construct three different synthetic test scenarios as GT images, simulating high resolution fluorescence microscopy localization maps on a FOV of 512×512 pixels of size 25 nm. The two first scenarios, referred as Test 1a and Test 2a, represent two molecules, arranged on the 256th column, at distance of 25 nm and 75 nm, respectively. The other scenario, referred as Test 3a, shows four molecules disposed on a circle of a radius equal to 125 nm. Two molecules are arranged on the 256th column whereas the others are arranged on the 256th row. Once set $L = 4$, by down-sampling the GT images according to the model in (6.1), we simulate the acquisition of 128×128 LR diffraction-limited frames, where the PSF is modeled by a Gaussian function whose FWHM is equal to 258.21 nm, that is $\sigma_{\mathbf{H}} \approx 110$ nm.
In the lower panel of Figure 6.1, we report the close-ups ($\times 10$ zooming) of the Region of Interest (ROI), namely the central zone of the 512×512 images where the synthetic spots are located, for the GT and Nearest Neighbour (NN), DeepSTORM, DeepCEL0 and CEL0 super resolved reconstructions. All the reconstructions reported have been normalized in the interval [0,1].
The zooms related to these methods are highlighted by blue, purple, green, red and yellow boxes, respectively. It is noteworthy that in the purple box the NN $\times 4$ upsampled image is reported in order to visualize the LR image at the same dimensions of the GT image. For the CEL0 method, we depict two different reconstructions obtained by setting two different regularization parameters for each test case. In the upper panel of Figure 6.1, we draw the line profiles corresponding to the 256th column of the 512×512 GT, NN and DeepCEL0 images. For all the three tests, the two molecules arranged on the 256th column correspond to the two peaks in the line profile of the GT image (see the blue line and blue box), whereas we observe they are completely overlapped and indistinguishable in the diffraction limited NN upsampled image's line profile (see purple line and purple box).

As qualitative results, in Test 1a, Test 2a and Test 3a, DeepCEL0 is able to well approximate the positions of the two molecules (see the red dashed line and red box). In all the three tests considered, the synthetic molecules are placed at a distance which is largely smaller than FWHM. In particular, Test 3a is the most challenging scenario since the number of molecules to estimate is greater than two. DeepCEL0 provides broadly better performances than DeepSTORM and more stable performances with respect to the CEL0 method. Indeed, DeepSTORM (see green box) does not separate the two synthetic molecules and provides a large number of FP molecules, thus confirming it is not designed to produce accurate localization maps. CEL0 (see yellow boxes), as expected, provides highly accurate localization maps, but, as we can observe, its performances

are highly sensitive to the choice of the regularization parameter. Furthermore, it is remarkable how different scenarios require different optimal parameters. On the contrary, the results provided by DeepCEL0 have been computed by using the same architecture trained setting $\lambda_{\text{CEL0}} = 100$. Therefore, these tests prove the effectiveness of our approach retrieving highly precise localization maps and more stability with respect to the choice of the hyperparameter λ_{CEL0}, if compared to the CEL0 method.

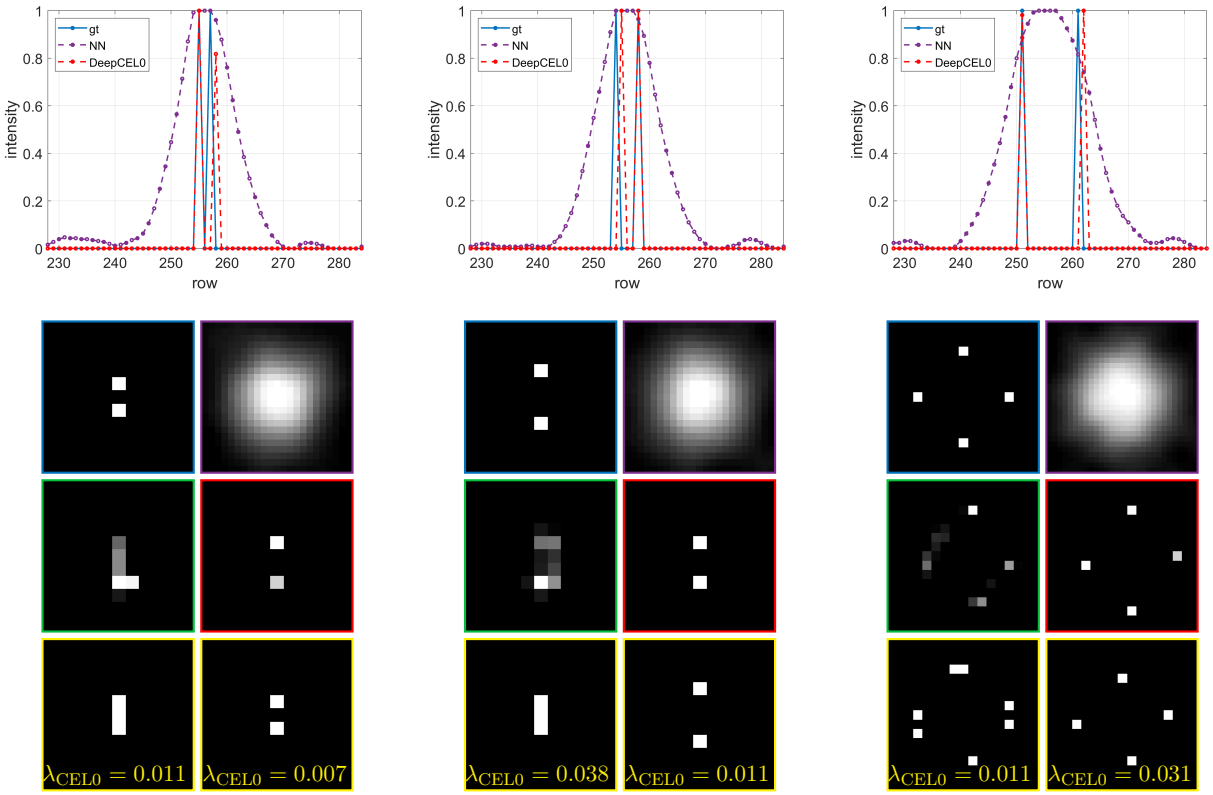


Figure 6.1: Localization on synthetic test images Line profiles crossing the 256th column of GT, NN and DeepCEL0 images for Test 1a, Test 2a, Test 3a (upper panel from left to right). Close-ups (x10) on the ROI of GT (blue box), NN (purple box), DeepSTORM (green box), DeepCEL0 (red box) and CEL0 with two different regularization parameters (yellow box) for Test 1a, Test 2a, Test 3a (lower panel from left to right).

Localization on a realistic dataset with theoretical PSF

We now consider a more realistic dataset provided by the 2013 SMLM challenge[1]. The dataset is a stack of 361 different frames used as GT localization maps in our analysis. A total number of 81049 emitters are counted: for each frame 217 fluorophores are activated on average. The frames simulate realistic high density acquisitions of 8 tubes of diameter size equal to 30 nm depicted on a FOV of 256×256 pixels of size 25 nm. We simulate LR diffraction-limited acquisitions by applying the image formation model in (6.1) setting $L = 4$, thus representing the GT scenarios on

[1] `https://srm.epfl.ch/Challenge/ChallengeSimulatedData`

a coarser grid of 64×64 pixels of size 100 nm. The PSF is modeled by using a Gaussian function of FWHM = 258.21 nm ($\approx \sigma_{\mathbf{H}} = 110$ nm). We further corrupt the LR frames by adding Poisson noise and Gaussian noise with different standard deviations, such that SNR = 10 dB, 12 dB, 15 dB, respectively referred to as Test 2b, Test 3b, Test 4b. Finally, we consider a more realistic scenario, referred to as Test 1b, where the frames are corrupted by Poisson noise and Gaussian noise components of different standard deviations.

In Table 6.1, we report the results expressed in terms of Jaccard index for δ set as 2,4,6 pixels, corresponding to a tolerance of 50,100,150 nm, respectively. The sensitivity and specificity values, instead, are computed only with respect to the tolerance $\delta = 2$, which expresses a more faithful compliance of the reconstructions compared to the GT.
DeepSTORM, DeepSTORM-ReLu and DeepCEL0 models are trained on the synthetic images described previously such that the PSF is set to satisfy the experimental conditions described above and the noise aberration leads to an SNR value equals 15 dB on average. These three trained models are used for all the four tests considered. For the sake of a fair comparison, for each test we estimate the CEL0 regularization parameter by a trial and error procedure sampling 30 values over the range $[1e\text{-}3, 1]$ on randomly chosen 8 frames among the stacked ones. In the following, we refer to the best regularization parameter for CEL0 as the one maximazing the Jaccard index with tolerance δ equals 2.
In Table 6.1, we report for each test (Test 1b, 2b, 3b, 4b) the results of CEL0, DeepSTORM, DeepSTORM-ReLu and DeepCEL0. For what concerns CEL0, in the first row referred to each test, we draw the results obtained with respect to the best regularization parameters, whereas, in the second row, we show the results obtained by choosing as regularization parameter the best one obtained for Test 4b. We stress that for the trained methods the regularization parameter is fixed as described in the previous sections.

We inspect in depth Test 2b, the most challenging one due to a low SNR value. In Figure 6.2, we report the sum of all the stacked reconstructed SR frames provided by the competing methods alongside the stacked GT and LR frames. In particular, for GT and all the methods, above the magenta line we report the activated molecules as white spots, whereas below the magenta line we depict the normalized images. The three close-ups ($\times 4$ zooming) show three region of interest, where the tubulins seems completely overlapped in the LR acquisition.
As a general comment, on the one side, DeepSTORM struggles to provide high precision localization maps: very low Jaccard values for all the tests and tolerances are obtained. For all the tests, the sensitivity reaches the highest possible value but at the expense of a very low specificity, meaning that this result is completely biased since it provides a very huge number of FP molecules (more than 50000). As an example, for Test 2b, this aspect is evident from Figure 6.2d.
On the other side, CEL0 reaches the best performances on Test 4b and competing performances on Test 3b, but poor results on Test 1b and Test 2b that represent the most realistic scenarios.
Conversely to the above mentioned competing methods, DeepCEL0 provides satisfying results for all the tests with respect to the Jaccard, sensitivity and specificity indexes, thus confirming the effectiveness of the novelties introduced, namely the combination of non-negativity constraints and the usage of a CEL0 regularized loss function. In particular, Figure 6.2f confirms once again how

our proposal can better provide more faithful localization maps with less FP molecules if compared to the other competing methods. We investigate the addition of the sole non-negativity constraints by showing the evaluation metrics (Table 6.1) and the visual reconstructions (Figure 6.2e) obtained by DeepSTORM-ReLu. Our experiments confirm that the non-negativity promoting CNN is the component of DeepCEL0 which contributes the most to an improvement of the evaluation metrics. Indeed, Table 6.1 highlights how DeepSTORM-ReLu clearly outperforms DeepSTORM, although with globally slightly lower performances than DeepCEL0. Moreover, a visual comparison between the reconstructions of DeepSTORM-ReLu and DeepCEL0 emphasises how the contribution of the CEL0 penalty in the loss of DeepCEL0 allows the method to better induce sparsity, by suppressing false positives, and to retrieve more faithful structures.

Finally, an interesting aspect to underline is that, even if DeepCEL0 is trained on images with SNR equals 15 dB on average and uses for all the experiments the same value of the regularization parameter, the trained models seem stable at varying the level of corruption (Test 1b, Test 2b, Test 3b, Test 4b). This stability is not still valid for CEL0 which shows a strong dependence on the choice of the regularization parameter as highlighted by the results in Table 6.1.

Table 6.1: Performance evaluation for localization on 2013 SMLM challenge realistic dataset with theoretical PSF in terms of Jaccard Index, sensitivity and specificity. The first and second best Jaccard Index values are highlighted in red and blue, respectively.

Test	**Method**	**Jaccard** (%)			**Sensitivity**(%)	**Specificity** (%)
		$\delta = 2$	$\delta = 4$	$\delta = 6$	$\delta = 2$	$\delta = 2$
1b	CEL0	41.64	49.46	51.65	56.22	99.88
		35.94	43.93	46.70	67.74	99.71
	DeepSTORM	0.38	0.38	0.38	100.00	13.17
	DeepSTORM-ReLu	56.34	67.55	70.50	73.73	99.90
	DeepCEL0	58.96	68.94	71.55	70.89	99.95
2b	CEL0	42.29	50.21	51.48	49.08	99.95
		17.13	28.64	33.16	51.15	99.34
	DeepSTORM	0.38	0.38	0.38	100.00	13.34
	DeepSTORM-ReLu	51.86	65.93	69.70	70.51	99.88
	DeepCEL0	54.83	68.49	71.37	65.44	99.94
3b	CEL0	62.17	63.60	64.32	65.90	99.98
		48.19	50.46	51.38	73.73	99.82
	DeepSTORM	0.38	0.38	0.38	100.00	13.53
	DeepSTORM-ReLu	58.10	67.54	70.08	76.04	99.90
	DeepCEL0	61.13	69.66	71.55	69.59	99.95
4b	CEL0	68.27	70.33	71.02	78.34	99.95
	DeepSTORM	0.38	0.38	0.38	100.00	12.65
	DeepSTORM-ReLu	59.49	69.11	71.76	75.12	99.91
	DeepCEL0	61.48	69.80	72.33	70.12	99.96

Localization on a realistic dataset without theoretical PSF

In the previous paragraph, we have considered a realistic dataset for which all the simulated acquisitions were constructed applying to the ground truth images the forward model (6.1), thus the sources of degradation were exactly known, i.e., the PSF and the type of noise (mixed Poisson and Gaussian). Therefore, we now consider the Microtubules dataset from the 2016 SMLM challenge[2]. This experimental datasets consist of sequences of frames from realistic stained biological samples acquired according to a 2D modality. More precisely, these simulated data are created using a sophisticated simulator considering specificities of the fluo dyes and camera. The dataset is a stack of 2500 different frames depicting high density acquisitions (2 emitters per μm^2) of 3 microtubules on a FOV of $6.4 \times 6.4 \times 1.5$ μm (64×64 pixels of size 100 nm). Since in this case we do not know the theoretical PSF, the FWHM has been estimated through trial and error procedures and set equal to 447,45 nm ($\approx 190nm$).
The regularization parameter used for the CEL0 reconstruction, equals 1.85, and it has been heuristically selected in order to maximize the Jaccard ($\delta = 0$) index. The other models are trained on a synthetic training set constructed considering the estimated PSF.

In Figure 6.3, we depict the LR image (Figure 6.3a) alongside the GT image (Figure 6.3b) and the reconstructions (setting $L = 4$) obtained by all the considered methods. In particular, below the green line, we show the normalized images, whereas above the green line, we report the maps representing the activated molecules depicted in white. To highlight the differences between the competing methods we consider three close-ups ($\times 4$ zooming). All the considerations raised on the previous subparagraph hold also in this case. The DeepCEL0 reconstruction (Figure 6.3f) reaches 39.43 as Jaccard index setting $\delta = 0$, whereas CEL0 (Figure 6.3c), DeepSTORM (Figure 6.3d) and DeepSTORM-ReLu (Figure 6.3e) reconstructions achieve 35.73, 2.32 and 34.38, respectively. This proves once again DeepCEL0 is able to localize the emitters with high precision. Furthermore, the binary maps show that DeepCEL0 preserves better the shape of the tubulins if compared with CEL0 and DeepSTORM reconstructions. Moreover, by comparing the close-ups of the DeepCEL0 and DeepSTORM-ReLu solutions it is evident how the addition of the CEL0 based regularizer helps to improve the localization and to suppress FP molecules.

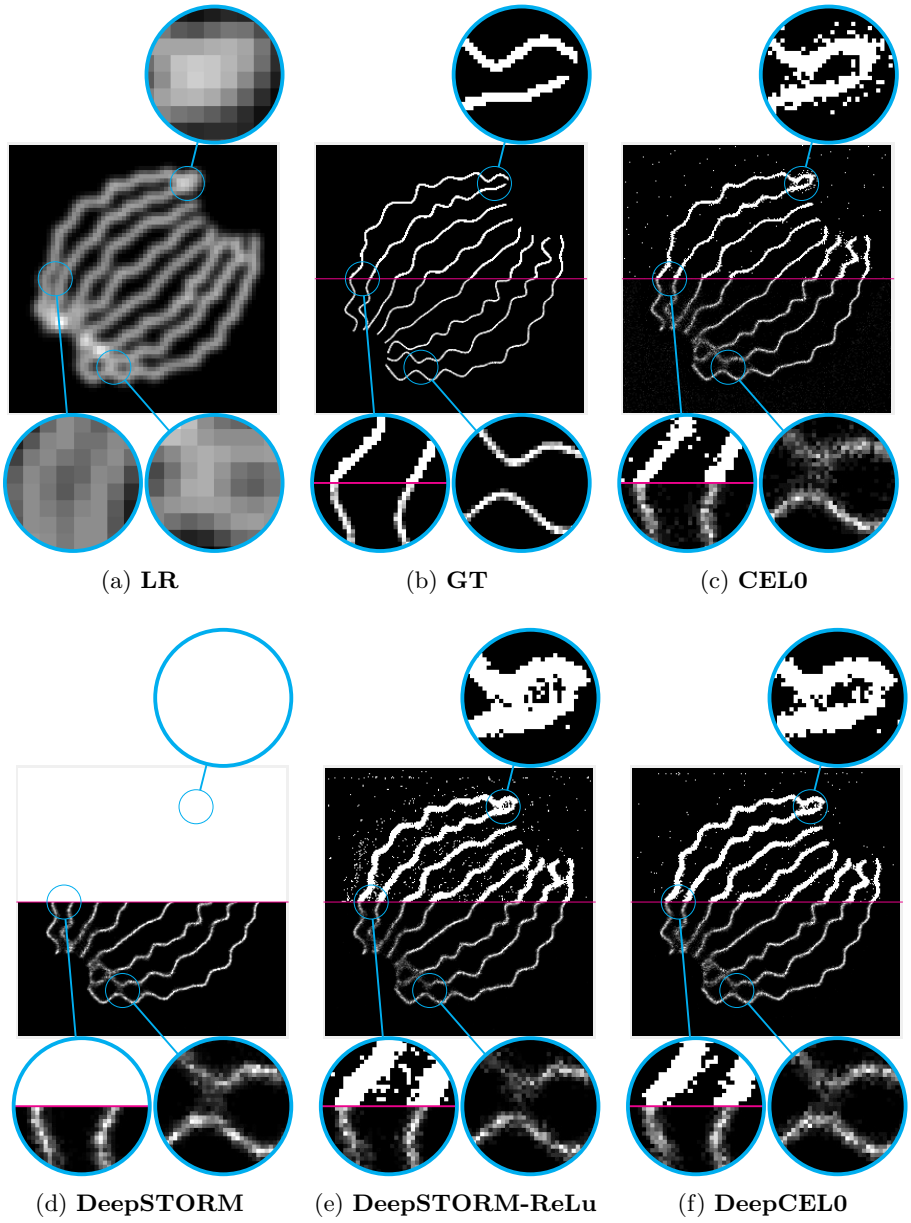
Localization on a real dataset

As final evaluation test, we compare DeepCEL0 with CEL0, DeepSTORM and DeepSTORM-ReLu on a real high density dataset of tubulins that is part of the 2013 IEEE ISBI SMLM challenge. The dataset refers to a real acquisition stack representing a field of view of 128×128 pixels of size 100 nm over 500 different consecutive frames. The PSF is modeled by using a Gaussian function of FWHM = 351.8 nm, that is $\sigma_{\mathbf{H}} \approx 150$ nm [147]. We aim at representing the acquired LR frames on a finer pixel grid of a factor $L = 4$. CEL0 reconstruction considers the regularization parameter equal to 0.5 as suggested by [148]. All the learning-based models, are trained by considering the above underlined PSF.

In Figure 6.4, we report the normalized LR image, i.e., the sum of all the stacked LR frames.

[2] `https://srm.epfl.ch/Challenge/ChallengeSimulatedData`

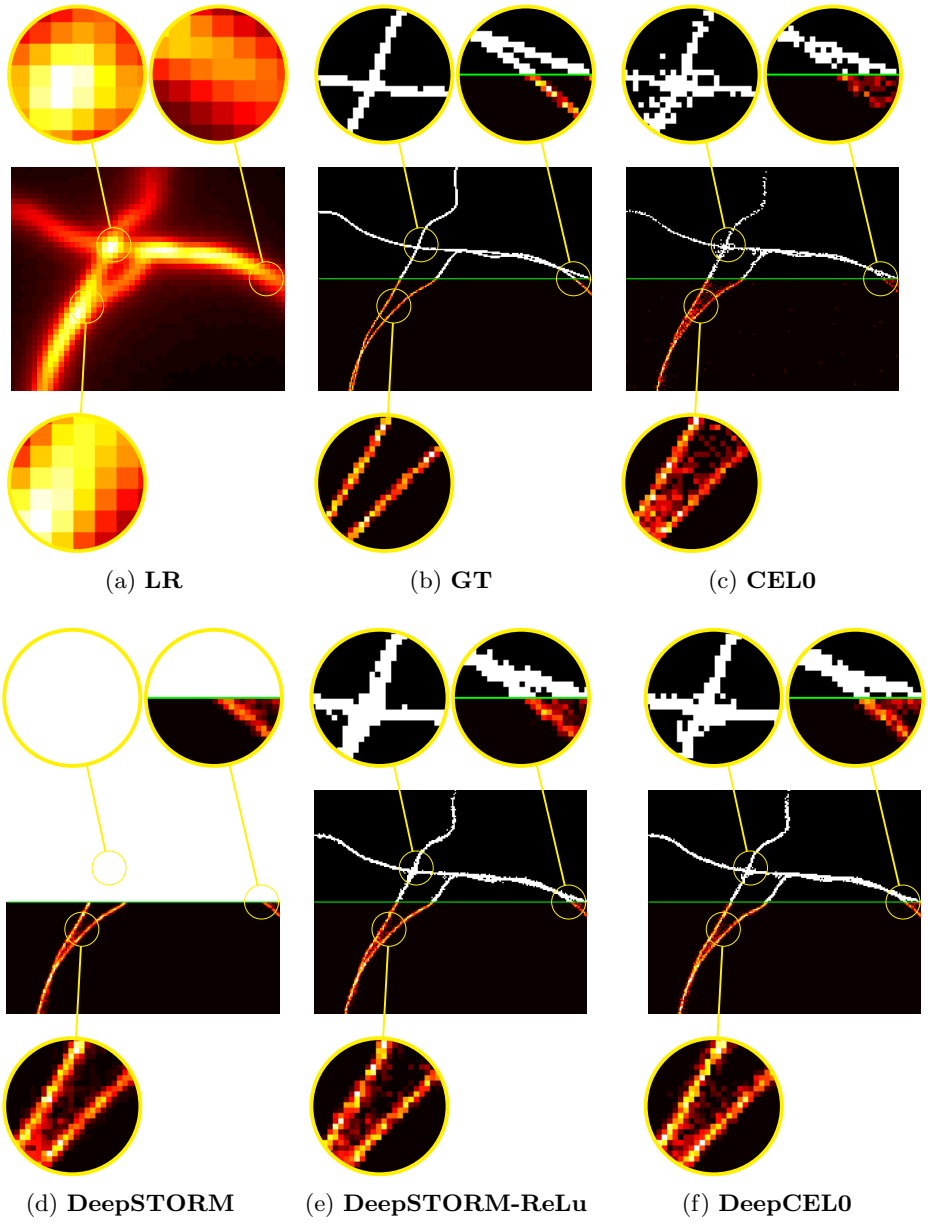
Moreover we depict the reconstructions provided by all the considered competitors. Above the green line we show the image with the activated molecules as white spots, whereas, below the green line the normalized images are reported. Finally, three close-ups are considered to better underline the differences between the competing methods. All the considerations highlighted for the simulated scenarios are still valid in this case. We here appreciate the influence of the L0 regularization both in CEL0 (Figure 6.4b) and DeepCEL0 (Figure 6.4e) reconstructions. Indeed, they look sharper than the solution provided by DeepSTORM (Figure 6.4c). Finally, if compared with CEL0, DeepCEL0 better separates the diverse tubulins and better preserves the shapes.



(a) **LR** (b) **GT** (c) **CEL0**

(d) **DeepSTORM** (e) **DeepSTORM-ReLu** (f) **DeepCEL0**

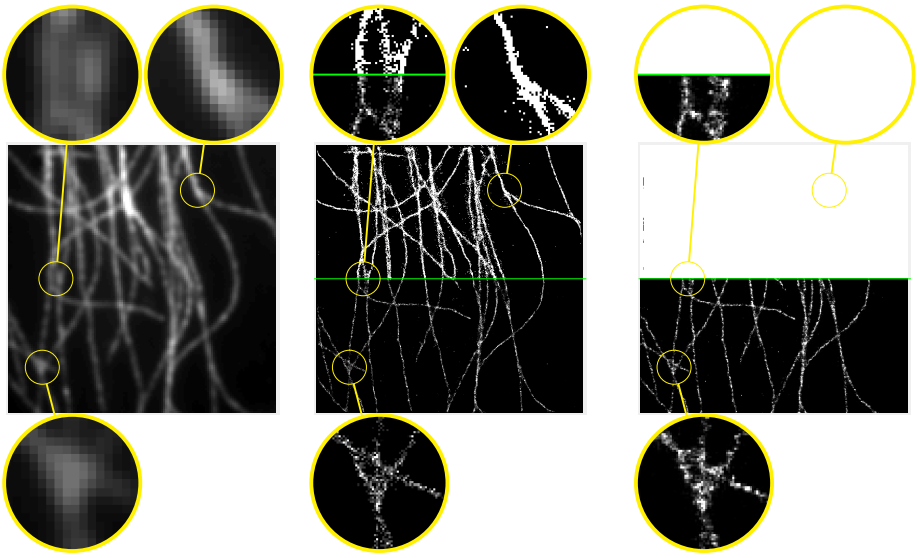
Figure 6.2: Image reconstruction results of the whole stack for the 2013 SMLM challenge realistic dataset with theoretical PSF. The first half of the images shows the binarized versions of the images. The second half of the images shows the normalized counterparts.



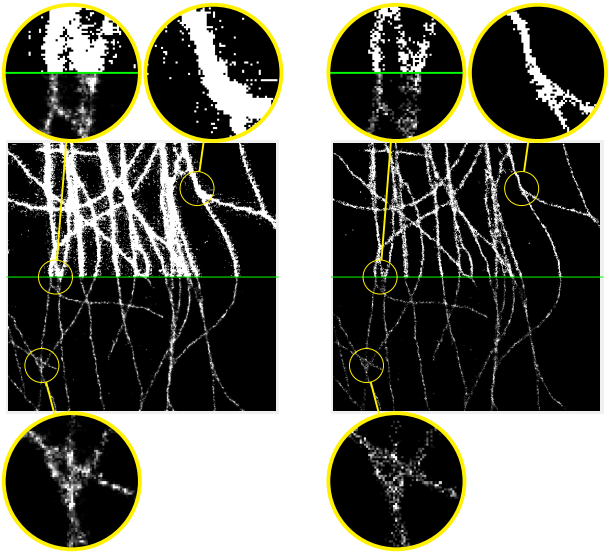
(a) **LR** (b) **GT** (c) **CEL0**

(d) **DeepSTORM** (e) **DeepSTORM-ReLu** (f) **DeepCEL0**

Figure 6.3: Image reconstruction results of the whole stack for the 2016 SMLM challenge realistic dataset without theoretical PSF. The first half of the images shows the binarized versions of the images. The second half of the images shows the normalized counterparts.



(a) **LR** (b) **CEL0** (c) **DeepSTORM**



(d) **DeepSTORM-ReLu** (e) **DeepCEL0**

Figure 6.4: Image reconstruction results of the whole stack for the IEEE ISBI tubulins dataset. The first half of the images shows the binarized versions of the images. The second half of the images shows the normalized counterparts.

Chapter 7

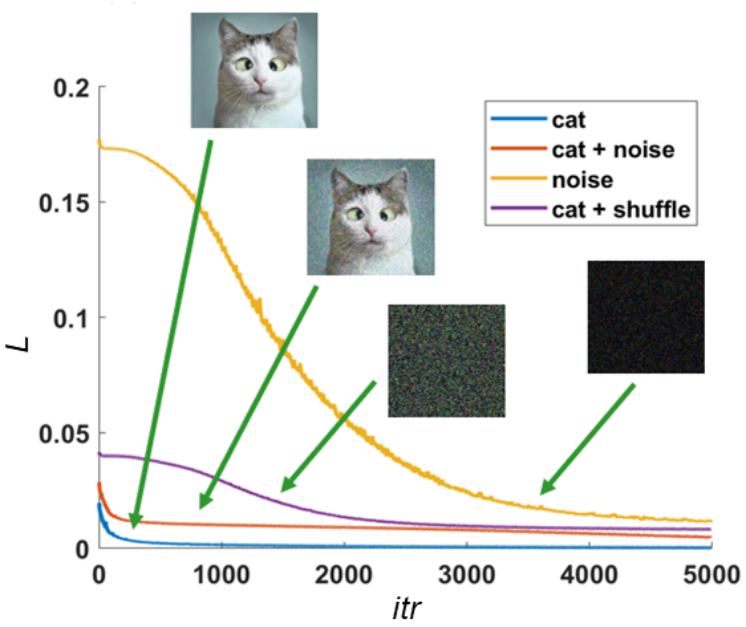
Deep Image Prior for image and video restoration

In the last decade, supervised deep learning-based approaches have shown state-of-the-art performances in the field of imaging inverse problems [11] due to their capability to learn the correlation between degraded images and their cleaned counterparts by exploiting high representative models, like Deep Neural Network (DNN) architectures, and an outer training set of degraded-clean example pairs. However, in general, they have several issues, including the lack of generalization when not trained with enough data. Moreover, in many real applications, such as medical imaging, it is practically impossible to build a labeled dataset with both ground truth and degraded data [149]. All these reasons have motivated researchers to inspect the so-called unsupervised deep learning approaches avoiding the usage of training sets [150, 151, 152, 153].
Deep Image Prior [39] is among the most promising methods belonging to this class. The DIP framework leverages the fact that the architecture of a deep Convolutional Neural Network generator reproduces natural images more easily than random noise, thus inducing an implicit regularization. More specifically, by recalling the acquisition model for the generic IR task (1.7) and considering a CNN $f : \mathbb{R}^p \times \mathbb{R}^N \to \mathbb{R}^n$, the DIP approach combines the following unconstrained minimization problem:

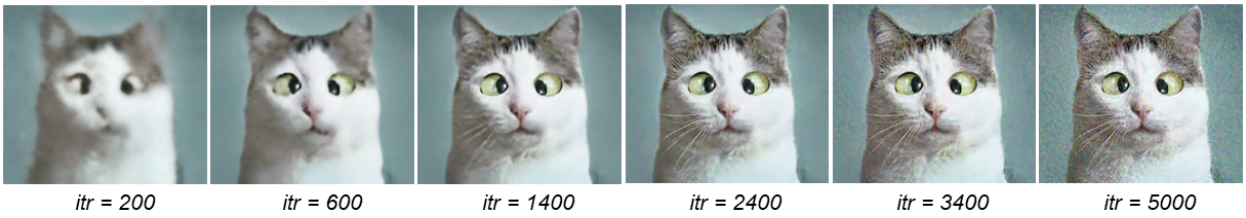
$$\arg\min_{\boldsymbol{\theta}\in\mathbb{R}^p} L(\boldsymbol{\theta}) := \frac{1}{2}\|\mathbf{A}f(\boldsymbol{\theta}, \mathbf{z}) - \mathbf{b}\|_2^2 \tag{7.1}$$

with a regularization by early-stopping procedure to compute a set of weights $\boldsymbol{\theta}^*$. An estimate $\mathbf{u}^*$ of the unknown $\mathbf{u}$ is then obtained computing $f(\boldsymbol{\theta}^*, \mathbf{z})$, where $\mathbf{z} \in \mathbb{R}^N$ represents a sample of a uniform distribution. We remark that the set of weights $\boldsymbol{\theta}^*$ is computed by applying standard gradient-based iterative algorithms to the problem (7.1) and early stopping the iterative process. The reason why we need to early-stop the optimization process will be clear in the following lines.

In Figure 7.1a we show the so-called DIP high impedance to noise property. We plot the behaviour of L as a function of the iterations (itr) of the gradient-based algorithm applied to solve (7.1). The operator $\mathbf{A}$ in (7.1) is set equal to the identity operator, whereas for $\mathbf{b}$ we consider four different choices: the `cat` natural image, the `cat` image affected by AWGN, the pure noise image sampled from a Gaussian distribution and, finally, the `cat` image after randomly permuting the



(a)



(b)

Figure 7.1: (a) Plot of L defined in (7.1) for different choices of $\mathbf{b}$: original `cat` image (blue line), noisy `cat` image (red line), Gaussian sample (yellow line), shuffled `cat` image (purple line). (b) Outputs at different iterations of the optimization process solving (7.1) when $\mathbf{b}$ is set as the noisy `cat` image and $\mathbf{A}$ as the identity operator.

pixels.

As a general comment, the optimization looks much easier for the first two choices, if compared to the last two for which it is evident a significant "impedance". In Figure 7.1b we depict the outputs at different iterations, when $\mathbf{b}$ is set as the noisy `cat` image. We can observe that in the limit the parametrization provided by the CNN can overfit the noise as well ($itr = 5000$), although it does so very reluctantly (Figure 7.1a) and at $itr = 1400$ we obtain an almost perfect restoration. In other words, in the first iterates of the optimization process the CNN parametrization offers high impedance to noise and low impedance to signal.

Up to now, researchers have mostly worked on a theoretical analysis of DIP [12, 154, 155] as well as to boost its performance. Inspired by the standard variational regularization methods, in [156, 157, 158, 159, 160, 161] the authors improved the DIP performance by adding an explicit penalization term R to the objective in (7.1) to avoid the issue of overfitting. Hence, the optimization

problem (7.1) is replaced by the following regularized one:

$$\arg\min_{\boldsymbol{\theta}\in\mathbb{R}^p} \frac{1}{2}\|\mathbf{A}f(\boldsymbol{\theta},\mathbf{z})-\mathbf{b}\|_2^2+\lambda R(f(\boldsymbol{\theta},\mathbf{z})). \tag{7.2}$$

More precisely, in [156, 159, 161] R is set as the standard Total Variation [20], whereas in [160] the authors consider the RED regularizer [30].
Furthermore, the selection of the regularization parameter λ in (7.2) is an essential issue that this approach inherits from the class of variational regularization methods [15]. A wise choice of regularization parameter is obviously crucial for obtaining useful approximate solutions to ill-posed problems. Indeed, replacing (7.1) with (7.2) induces better regularized solutions, provided a suitable value for λ depending both on the level of degradation of the acquired image and on the considered problem. As we have seen in Chapter 3, in the literature there exist various strategies for choosing the parameter λ, such as the Morozov's discrepancy principle, the generalized cross-validation (GCV) and the L-curve method. However, it is well-known that such strategies can present different limitations: they are not at all easy to apply for every regularizer; they can provide either over or under smoothed solutions; they may often require to solve (7.2) many times for different values of λ, making the overall procedure computationally expensive. For these reasons, in this context it is a common practice to manually tune the regularization parameter by trial-and-error procedures [156, 157, 159, 160, 161], thus leading to an high demanding workload.

Contribution

Section 7.1 is based on the publication [158] and on the pre-print [162], where we provided two different DIP-based optimization models which share the property of automatically balancing the effect of the regularization. First, we consider an unconstrained model as the one in (7.2) where the regularization term is additively separable. The strength of the regularization is pixelwise weighted by a set (one for each pixel) of automatically estimated regularization parameters whose definition is based on local patterns. Furthermore, we propose to reformulate the standard regularized unconstrained DIP optimization problem (7.2) as a constrained one, whose constraints are set in accordance to the discrepancy principle. Finally, both arising unconstrained and constrained optimization problems are solved via ADMM.
Section 7.2 is based on the publication [157] where we proposed a novel deep learning-based algorithm that extends the well-known DIP to Time-Lapse Microscopy Video Super Resolution, thus not requiring any training set. The proposed method, termed Recursive Deep Prior Video introduces some novelties: the weights of the DIP network architecture are initialized for each of the video frames according to a new recursive updating rule combined with an efficient early stopping criterion. Finally, for each frame the proposed method considers (7.2) by setting the regularizer equal to either the handcrafted isotropic or anisotropic Total Variation.

7.1 Constrained and unconstrained Deep Image Prior models for image restoration

In this section, we introduce two regularized unconstrained and constrained DIP optimization models and we solve them within the ADMM framework.

7.1.1 Unconstrained model

To address problem (7.2), the approaches described in [156, 159, 161] consider the handcrafted Total Variation regularizer with a single regularization parameter, which does not allow to adapt the regularization to the local image patterns. In our unconstrained model, we replace the regularizer in (7.2) and the parameter λ with a flexible space variant regularizer which considers a set of local regularization parameters λ_i, for $i = 1 \dots n$, weighting the strength of the regularization for each pixel. The resulting unconstrained regularized DIP model reads:

$$\underset{\boldsymbol{\theta}\in\mathbb{R}^p}{\arg\min}\ \frac{1}{2}\|\mathbf{A}f(\boldsymbol{\theta},\mathbf{z})-\mathbf{g}\|_2^2+\sum_{i=1}^{n}\lambda_i R_i((\mathcal{A}f(\boldsymbol{\theta},\mathbf{z}))_{I_i}), \tag{7.3}$$

where I_i for $i = 1 \dots n$ are a partition of the set $I := \{1, \dots, l\}$, R_i are real-valued functions representing the local components of the regularizer, $\mathcal{A} : \mathbb{R}^n \to \mathbb{R}^l$ is a generic operator and l is a positive integer such that $l \geq n$. The functions R_i and the local parameters λ_i usually represent local energies defined on a neighbourhood of the i-th pixel thus forcing prior information based on local patterns. Considering a vector $\mathbf{v}$ in $\mathbb{R}^l$, for every $i = 1, \dots n$ we denote $\mathbf{v}_{I_i} \in \mathbb{R}^{|I_i|}$ as the vector specified by the components of $\mathbf{v}$ whose indexes are in I_i.
Examples of regularization terms belonging to this class are the Tikhonov-like and the Total Variation ones. For instance, in the Tikhonov-based regularizers, $\mathcal{A}$ is usually chosen as the identity or the Laplacian operators, whereas $R_i : \mathbb{R} \to \mathbb{R}$ is chosen as the square function. Concerning the isotropic Total Variation, $\mathcal{A}$ represents the discrete gradient and $R_i : \mathbb{R}^2 \to \mathbb{R}$ is chosen as the ℓ_2-norm function.

By adding an auxiliary variable $\mathbf{v} := \mathcal{A}f(\boldsymbol{\theta}, \mathbf{z})$, the optimization problem (7.3) is equivalent to the following formulation:

$$\begin{aligned}\underset{\boldsymbol{\theta}\in\mathbb{R}^p,\mathbf{v}\in\mathbb{R}^l}{\arg\min}\ & \frac{1}{2}\|\mathbf{A}f(\boldsymbol{\theta},\mathbf{z})-\mathbf{g}\|_2^2+\sum_{i=1}^{n}\lambda_i R_i(\mathbf{v}_{I_i}) \\ \text{s.t.}\quad & \mathcal{A}f(\boldsymbol{\theta},\mathbf{z})=\mathbf{v}.\end{aligned} \tag{7.4}$$

As iterative solver we consider the ADMM scheme. The augmented Lagrangian function related to the problem (7.4) reads:

$$\begin{aligned}L(\boldsymbol{\theta},\mathbf{v},\boldsymbol{\mu}_{\mathbf{v}}) =& \frac{1}{2}\|\mathbf{A}f(\boldsymbol{\theta},\mathbf{z})-\mathbf{g}\|_2^2+\sum_{i=1}^{n}\lambda_i R_i(\mathbf{v}_{I_i}) \\ &+\frac{\beta_{\mathbf{v}}}{2}\|\mathcal{A}f(\boldsymbol{\theta},\mathbf{z})-\mathbf{v}\|_2^2+\langle\boldsymbol{\mu}_{\mathbf{v}},\mathcal{A}f(\boldsymbol{\theta},\mathbf{z})-\mathbf{v}\rangle,\end{aligned} \tag{7.5}$$

where $\beta_{\mathbf{v}}$ is a positive scalar, called penalty parameter, and $\boldsymbol{\mu}_{\mathbf{v}}$ is the Lagrangian parameter associated with the constraint $\mathcal{A}f(\boldsymbol{\theta}, \mathbf{z}) = \mathbf{v}$. According to the ADMM framework, we seek for its saddle point by minimizing with respect to the primal variables $\boldsymbol{\theta}$ and $\mathbf{v}$, alternatively, and by maximizing with respect to the dual variable $\boldsymbol{\mu}_{\mathbf{v}}$. Upon suitable initialization of the involved variables, the k-th iteration of the ADMM iterative algorithm reads as follows:

$$\begin{cases} \boldsymbol{\theta}^{k+1} \in \underset{\boldsymbol{\theta}\in\mathbb{R}^p}{\arg\min} \ \frac{1}{2}\|\mathbf{A}f(\boldsymbol{\theta}, \mathbf{z}) - \mathbf{g}\|_2^2 + \frac{\beta_{\mathbf{v}}}{2}\left\|\mathcal{A}f(\boldsymbol{\theta}, \mathbf{z}) - \mathbf{v}^k + \frac{\boldsymbol{\mu}_{\mathbf{v}}^k}{\beta_{\mathbf{v}}}\right\|_2^2 & (7.6) \\ \mathbf{v}^{k+1} \in \underset{\mathbf{v}\in\mathbb{R}^l}{\arg\min} \ \sum_{i=1}^{n} \lambda_i^k R_i(\mathbf{v}_{I_i}) + \frac{\beta_{\mathbf{v}}}{2}\left\|\mathbf{v} - \left(\mathcal{A}f(\boldsymbol{\theta}^{k+1}, \mathbf{z}) + \frac{\boldsymbol{\mu}_{\mathbf{v}}^k}{\beta_{\mathbf{v}}}\right)\right\|_2^2 & (7.7) \\ \boldsymbol{\mu}_{\mathbf{v}}^{k+1} = \boldsymbol{\mu}_{\mathbf{v}}^k + \beta_{\mathbf{v}}(\mathcal{A}f(\boldsymbol{\theta}^{k+1}, \mathbf{z}) - \mathbf{v}^{k+1}). & (7.8) \end{cases}$$

In our implementation, the first problem (7.6) is solved inexactly by applying one iteration of a gradient-based method using back-propagation to compute the gradient. We observe that this optimization problem is very similar to the one solved in the classical DIP framework (7.1). In this particular case, we force $\mathcal{A}f(\boldsymbol{\theta}^{k+1}, \mathbf{z})$ to be close to $\mathbf{v}^k - \frac{\boldsymbol{\mu}_{\mathbf{v}}^k}{\beta_{\mathbf{v}}}$. From a numerical point of view, this squared ℓ_2-norm term provides a stabilizing and robustifying effect to the DIP minimization.
Concerning the optimization problem in (7.7), if the proximal map of R_i can be easily computed for all $i = 1 \dots n$, then the problem can be efficiently solved in a closed form by applying the proximity operator of R_i to the n components of $\mathcal{A}f(\boldsymbol{\theta}^{k+1}, \mathbf{z}) + \frac{\boldsymbol{\mu}_{\mathbf{v}}^k}{\beta_{\mathbf{v}}}$. Such hypotheses on R_i are not so restrictive. For example both Tikhonov-like and isotropic Total Variation regularizers satisfy these assumptions since the R_i are set as the square or ℓ_2-norm functions. In our implementation, we chose to vary the set of local regularization parameters λ_i along the iterations. In particular, their formulation is inspired by [163] and reads:

$$\lambda_i^k = \frac{1}{2n} \frac{\|\mathbf{A}f(\boldsymbol{\theta}^{k+1}, \mathbf{z}) - \mathbf{g}\|_2^2}{R_i\left((\mathcal{A}f(\boldsymbol{\theta}^{k+1}, \mathbf{z}))_{I_i}\right)}. \tag{7.9}$$

This entails that the smaller is the value of the local component function the greater is the regularization provided at pixel i.

Remark 7.1. The value of the penalty parameter $\beta_{\mathbf{v}}$ is hand-tuned. However, in the experimental part we empirically show that the choice of this hyperparameter does not affect the performance of the method as much as the choice of the regularization parameter when dealing with model (7.2). In detail, we empirically demonstrate that the performance of the proposed model is not sensitive to this penalty parameter $\beta_{\mathbf{v}}$ for all considered test problems if chosen in a reasonable set.

7.1.2 Constrained model

The starting point of this approach is again the regularized DIP optimization problem in (7.2). Differently from the unconstrained model described in Section 7.1.1, we here assume $R : \mathbb{R}^n \to \mathbb{R}$ is a generic regularizer. The constrained model we refer to in the following reads as:

$$\underset{\boldsymbol{\theta}\in\mathbb{R}^p}{\arg\min} \ R(f(\boldsymbol{\theta}, \mathbf{z})) \quad \text{s.t.} \quad f(\boldsymbol{\theta}, \mathbf{z}) \in D_{\sigma_{\mathbf{e}}}, \tag{7.10}$$

where $D_{\sigma_{\mathbf{e}}}$ is defined as:

$$D_{\sigma_{\mathbf{e}}} := \{f(\boldsymbol{\theta}, \mathbf{z}) \in \mathbb{R}^n \mid \|\mathbf{A}f(\boldsymbol{\theta}, \mathbf{z}) - \mathbf{g}\|_2^2 \leq \tau\sigma_{\mathbf{e}}^2 m\} \tag{7.11}$$

with τ being a positive scalar and $\sigma_{\mathbf{e}}$ being the standard deviation of the noise affecting $\mathbf{g}$. The constrained model (7.10) exploits the Morozov's discrepancy principle by simply extending [55, 54]. We propose to solve (7.10) via ADMM. By introducing two auxiliary variables $\mathbf{t} := f(\boldsymbol{\theta}, \mathbf{z})$ and $\mathbf{r} := \mathbf{A}f(\boldsymbol{\theta}, \mathbf{z}) - \mathbf{g}$, two positive penalty parameters $\beta_{\mathbf{t}}$ and $\beta_{\mathbf{r}}$, the augmented Lagrangian functional is defined as:

$$\begin{aligned} L(\boldsymbol{\theta}, \mathbf{t}, \mathbf{r}; \boldsymbol{\mu}_{\mathbf{t}}, \boldsymbol{\mu}_{\mathbf{r}}) = R(\mathbf{t}) + i_{B_\delta}(\mathbf{r}) + \langle \boldsymbol{\mu}_{\mathbf{t}}, f(\boldsymbol{\theta}, \mathbf{z}) - \mathbf{t} \rangle + \frac{\beta_{\mathbf{t}}}{2}\|\mathbf{t} - f(\boldsymbol{\theta}, \mathbf{z})\|^2 \\ + \langle \boldsymbol{\mu}_{\mathbf{r}}, \mathbf{A}f(\boldsymbol{\theta}, \mathbf{z}) - \mathbf{g} - \mathbf{r} \rangle + \frac{\beta_{\mathbf{r}}}{2}\|\mathbf{r} - (\mathbf{A}f(\boldsymbol{\theta}, \mathbf{z}) - \mathbf{g})\|^2, \end{aligned} \tag{7.12}$$

where i_{B_δ} is the indicator function of the ball $B_\delta \subset \mathbb{R}^m$, centered in $\mathbf{0} \in \mathbb{R}^m$, of radius $\delta := \sqrt{\tau\sigma^2 m}$, and $\boldsymbol{\mu}_{\mathbf{t}}$, $\boldsymbol{\mu}_{\mathbf{r}}$ are the Lagrangian parameters related to the auxiliary variables. Hence, we consider the following iterative scheme:

$$\begin{cases} \boldsymbol{\theta}^{k+1} \in \underset{\boldsymbol{\theta} \in \mathbb{R}^p}{\arg\min} \ \dfrac{\beta_{\mathbf{r}}}{2} \left\| \mathbf{r}^k - \left(\mathbf{A}f(\boldsymbol{\theta}, \mathbf{z}) - \mathbf{g} + \dfrac{\boldsymbol{\mu}_{\mathbf{r}}^k}{\beta_{\mathbf{r}}} \right) \right\|_2^2 + \dfrac{\beta_{\mathbf{t}}}{2} \left\| f(\boldsymbol{\theta}, \mathbf{z}) - \mathbf{t}^k + \dfrac{\boldsymbol{\mu}_{\mathbf{t}}^k}{\beta_{\mathbf{t}}} \right\|_2^2 & (7.13) \\ \mathbf{t}^{k+1} = \underset{\mathbf{t} \in \mathbb{R}^n}{\arg\min} \ R(\mathbf{t}) + \dfrac{\beta_{\mathbf{t}}}{2} \left\| \mathbf{t} - \left(f(\boldsymbol{\theta}^{k+1}, \mathbf{z}) + \dfrac{\boldsymbol{\mu}_{\mathbf{t}}^k}{\beta_{\mathbf{t}}} \right) \right\|_2^2 & (7.14) \\ \mathbf{r}^{k+1} = \underset{\mathbf{r} \in \mathbb{R}^m}{\arg\min} \ i_{B_\rho}(\mathbf{r}) + \dfrac{\beta_{\mathbf{r}}}{2} \left\| \mathbf{r} - \left(\mathbf{A}f(\boldsymbol{\theta}^{k+1}, \mathbf{z}) - \mathbf{g} + \dfrac{\boldsymbol{\mu}_{\mathbf{r}}^k}{\beta_{\mathbf{r}}} \right) \right\|_2^2 & (7.15) \\ \boldsymbol{\mu}_{\mathbf{t}}^{k+1} = \boldsymbol{\mu}_{\mathbf{t}}^k + \beta_{\mathbf{t}}(f(\boldsymbol{\theta}^{k+1}, \mathbf{z}) - \mathbf{t}^{k+1}) & (7.16) \\ \boldsymbol{\mu}_{\mathbf{r}}^{k+1} = \boldsymbol{\mu}_{\mathbf{r}}^k + \beta_{\mathbf{r}}(\big(\mathbf{A}f(\boldsymbol{\theta}^{k+1}, \mathbf{z}) - \mathbf{g}\big) - \mathbf{r}^{k+1}). & (7.17) \end{cases}$$

This proposed ADMM scheme sequentially updates the primal variables $\boldsymbol{\theta}, \mathbf{t}$ and $\mathbf{r}$ and the dual variables $\boldsymbol{\mu}_{\mathbf{t}}$, $\boldsymbol{\mu}_{\mathbf{r}}$. Similarly to the standard DIP framework solving (7.1), the first step in (7.13) updates the network's weights performing one back-propagation step. The update of $\mathbf{t}$, provided by the second step reported in (7.14), strictly depends on the choice of the regularizer. However, problem (7.14) is mathematically equivalent to a proximal map, therefore it can admit a closed form solution or it can be solved through either fixed point or gradient descent strategies as in [160]. The update of $\mathbf{r}$ is a simple projection onto the ball B_δ.

We stress that our general approach (7.10), largely differs from model (7.2) proposed in [39], since it overcomes the problem of tuning the regularization parameter provided the noise standard deviation $\sigma_{\mathbf{e}}$. In practice, it is sufficient to consider a good estimate of $\sigma_{\mathbf{e}}$ which can be computed by applying the efficient algorithms described in [164, 165].
Finally, we point out that the penalty parameters $\beta_{\mathbf{t}}$ and $\beta_{\mathbf{r}}$ are hand-tuned. The considerations highlighted for the penalty $\beta_{\mathbf{v}}$ in Remark 7.1 also apply to these two hyperparameters.

7.1.3 Numerical results

In this section, we show the results of some numerical experiments carried out to highlight the main benefits of the suggested unconstrained and constrained models and to evaluate their effectiveness

in solving image deblurring and denoising tasks on synthetic natural images as well as real medical ones.

Implementation details. Regarding the choice of the regularizer, both models allow a certain freedom of choice. For the unconstrained model, we consider the handcrafted space variant Total Variation and in the following we refer to it as DIP-WTV. Upon this assumption the set of regularization parameters is defined according to the formula given by (7.9).
Concerning the constrained model (7.10), we set the regularizer R equal to the RED regularizer [30]. We refer to this approach as cDIP-RED in the following. The cDIP-RED approach requires the knowledge of the noise standard deviation affecting the acquired image which we estimate by applying the algorithm described in [164], even if for the simulated tests we do know it. The parameter τ in (7.10) is set equal to 1 for all the experiments. For both models and for all the experiments performed we stop the related ADMM iterative process after 5000 iterations.
As deep neural network architecture we consider a generative CNN encoder-decoder architecture, as suggested in [39], whose number of weights $\boldsymbol{\theta}$ is about 2 millions.
The input $\mathbf{z}$ is a 3D random input tensor sampled from a uniform distribution having the same size of the unknown image and 32 channels. In the experiments, we follow the common practice in the DIP framework [160, 39] and use Adam [166] to update the set of parameters $\boldsymbol{\theta}$ according to (7.6) and (7.13). As typically done, we also perturb in each iteration the input $\mathbf{z}$ by a component sampled from a Gaussian distribution with zero mean and standard deviation equal to $\frac{1}{30}$ and we compute the final output as the average of all iterates.
Evaluation metrics, comparisons and test set. We perform several tests by varying the level of degradation and evaluate the performances through qualitative visual comparisons and quantitatively by PSNR and SSIM metrics. The proposed approaches DIP-WTV and cDIP-RED are compared with the standard DIP [39] and the DeepRED [160] algorithms. We point out that in [160] the authors prove that DeepRED outperforms other several approaches as far as the deblurring and denoising tasks are concerned. Moreover, we underline that in their implementation, to enforce the regularization, the authors implement a strategy that increases the magnitude of the regularization parameter λ, along the iterations, when the computed solution starts overfitting the corrupted image. More precisely, when the PSNR value between the restored image and the degraded image is greater than a given threshold γ the regularization parameter is increased by adding a given constant.
In Figure 7.2, we depict the images used in the numerical simulations. We consider a test set of five red-green-blue (RGB) natural images belonging to the Set5 dataset [167], two black-white (BW) natural images and one chest CT image of a patient affected by COVID-19 already post-processed into a 2D image after the acquisition. We treat all the images belonging to Set5 as ground truths as well as the *watercastle* BW image. In our experiments, the simulated acquired images are created by applying the image formation model (1.16) to the related ground truths. In particular, to simulate blurred data we assume that $\mathbf{A} = \mathbf{H}$ represents the discretization of a convolutional product with a Gaussian kernel of standard deviation $\sigma_{\mathbf{H}}$. We remark that the level of degradation of the simulated acquisition is specified by the magnitudes of $\sigma_{\mathbf{e}}$ and $\sigma_{\mathbf{H}}$. Finally, we stress that

the *skyscraper* BW image and the real chest CT image are affected by artifacts. Since no ground truths are available for these images, the comparisons among the methods are carried out through visual inspection.

Butterfly *Bird* *Head* *Woman*

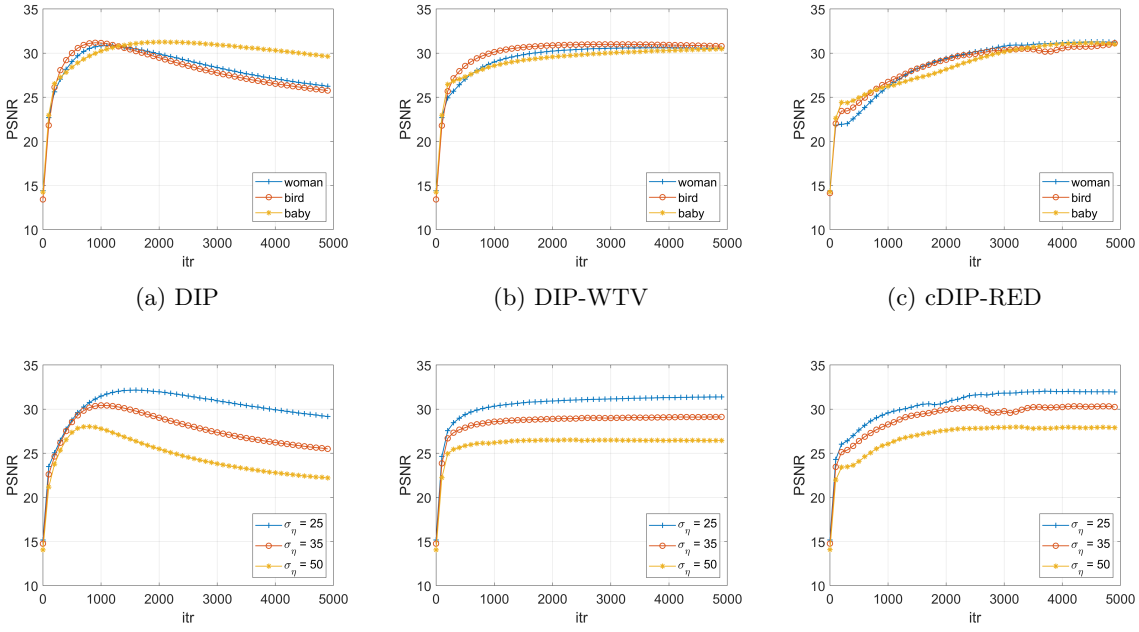
Baby *Watercastle* *Skyscraper* *chest CT*

Figure 7.2: The test images employed in the numerical experiments. *Butterfly*: RGB image 256×256 pixels, *Bird*: RGB image 288×288 pixels, *Head*: RGB image 256×256 pixels, *Woman*: RGB 224×320 pixels, *Baby*: RGB 512×512 pixels, *Watercastle*: BW 320×480 pixels, *Skyscraper*: BW 256×256 pixels, *chest CT*: BW 512×512 pixels.

Stability with respect to hyperparameters and empirical convergence

In this section, we describe the advantages which the models previously suggested bring over the considered competitors. In the first part, we empirically show the proposed approaches avoid the typical noise overfitting of DIP. Then, we underline how the suggested methods are more robust with respect to the choice of the hyperparameters than DeepRED. Finally, we empirically demonstrate that solutions of the proposed approaches satisfy the Morozov's discrepancy principle.

No overfitting. In the first test, we highlight the sensitivity of the standard DIP algorithm with respect to the choice of the optimal number of iterations to be performed and we compare it with DIP-WTV and cDIP-RED. For all experiments in this section we set the ADMM penalties $\beta_{\mathbf{v}} = 1$, $\beta_{\mathbf{t}} = 0.5$ and $\beta_{\mathbf{r}} = 1$. We consider the *woman*, *bird*, and *baby* images and we simulate the noisy acquisitions by corrupting the ground truths with an AWGN component of standard deviation $\sigma_{\mathbf{e}} = 35$. Then, we apply DIP, DIP-WTV, and cDIP-RED and in the upper panel of Figure 7.3 we depict the behaviour of the PSNR metric along the performed iterations. In order to analyze the relation between the noise level of the simulated acquisition and the optimal number of iterations to be performed by DIP, DIP-WTV and cDIP-RED, in lower panel of Figure 7.3, we report the behavior of the PSNR metric along the iterations while the level of corruption changes. In partic-



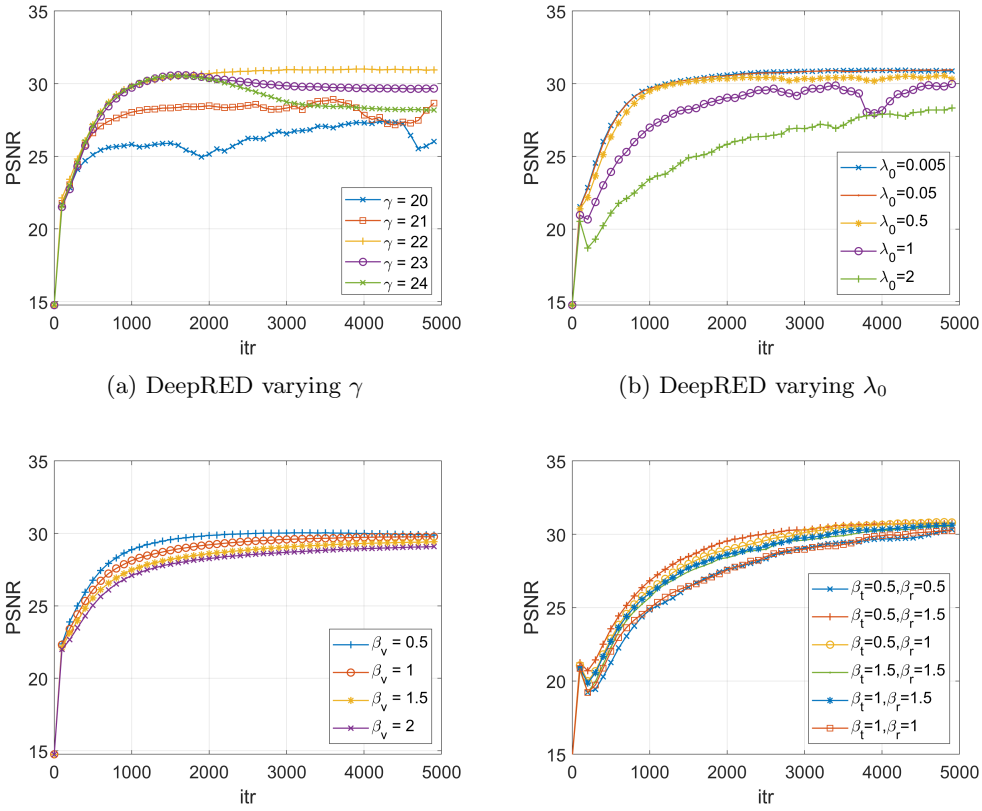
(a) DIP (b) DIP-WTV (c) cDIP-RED

(d) DIP (e) DIP-WTV (f) cDIP-RED

Figure 7.3: The PSNR values achieved by DIP, DIP-WTV and cDIP-RED along the iterations. In (a)-(b)-(c) the DIP, DIP-WTV and cDIP-RED are tested on three different RGB images degraded setting $\sigma_{\mathbf{e}} = 35$. In (d)-(e)-(f) DIP, DIP-WTV and cDIP-RED, respectively, are tested on the *butterfly* RGB image corrupted with different noise levels.

ular, we consider the *butterfly* test images and we corrupt it by AWGN with $\sigma_{\mathbf{e}} = 25, 35, 50$. As a general comment, Figure 7.3 shows that standard DIP starts overfitting the corrupted image along the iterative process. Moreover, for the DIP approach, this test highlights that the number of iterations to reach the best PSNR strongly depends on the image considered (Figure 7.3a), and on the level of corruption (Figure 7.3d). Conversely, the DIP-WTV (Figures 7.3b and 7.3e) and cDIP-RED (Figures 7.3c and 7.3f) schemes do not overfit the corrupted data while the PSNR does not decrease.

No regularization parameter is required. The DeepRED algorithm overcomes the problem of overfitting by adding the RED regularizer to the objective minimized by the standard DIP, provided a proper value for the regularization parameter λ. In this section, we highlight the sensitivity of the DeepRED algorithm with respect to the choice of the hyparameters defining the sequence of the regularization parameters, namely the threshold γ and the starting value of the regularization parameter λ_0. In Figure 7.4a, we show the behaviour of the PSNR for different values of the threshold γ. The degraded image is obtained by corrupting the *butterfly* image with an AWGN component setting $\sigma_{\mathbf{e}} = 35$. The parameter λ_0 is fixed equal to 0.005 while the increasing factor equals 0.03. For high values of the threshold, the regularization contribution is too weak thus the PSNR starts decreasing, which means DeepRED starts overfitting the degraded image. For low values of the threshold, the regularization parameter starts becoming high thus too much regular-



(a) DeepRED varying γ

(b) DeepRED varying λ_0

(c) DIP-WTV varying $\beta_{\mathbf{v}}$

(d) cDIP-RED varying $\beta_{\mathbf{t}}$ and $\beta_{\mathbf{r}}$

Figure 7.4: The PSNR values achieved by DIP, DIP-WTV and cDIP-RED along the iterations for the *butterfly* RGB image with $\sigma_{\mathbf{e}} = 35$.

ization is enforced. The best compromise for this *butterfly* test problem is $\gamma = 22$. However, in our experiments we observe that this value depends once again on both the image and the noise level considered and cannot be fixed a priori. In Figure 7.4b, we report the PSNR behaviour obtained by fixing $\gamma = 22$ and by changing the starting value of the regularization parameter λ_0. We observe the output of the DeepRED in 5000 iterates largely depends on the choice of this hyperparameter. We stress that DeepRED is implemented in the ADMM framework which requires the tuning of the penalty parameter. In our experiments, the DeepRED penalty parameter is set equal to 0.5 as suggested by the authors in [160].

The main feature of DIP-WTV and cDIP-RED is that the introduced regularization has no parameters to be estimated. In the case of DIP-WTV, the space variant regularization parameters are automatically estimated along the iterations, whereas the constrained formulation of cDIP-RED allows to automatically estimate the strength of the regularization by the Morozov's discrepancy principle.

Stability with respect to ADMM penalties $\beta_{\mathbf{v}}$, $\beta_{\mathbf{t}}$ and $\beta_{\mathbf{r}}$. We remark that for the DIP-WTV and cDIP-RED approaches we just need to hand tune the ADMM penalties $\beta_{\mathbf{v}}$, $\beta_{\mathbf{t}}$ and $\beta_{\mathbf{r}}$. However, in order to prove the stability of these methodologies with respect to the choice of these

parameters, in Figures 7.4c and 7.4d we depict the PSNR behaviour provided by DIP-WTV and cDIP-RED on the previous test image by setting different values for $\beta_{\mathbf{v}}$, $\beta_{\mathbf{t}}$ and $\beta_{\mathbf{r}}$. We stress that the range for the ADMM penalties for DIP-WTV and cDIP-RED has been deduced by the values suggested in [160] for their ADMM implementation.
From these figures we can conclude that for the DIP-WTV and the cDIP-RED methods the ADMM penalties affect the convergence speed, but the PSNR behaviour of both the approaches is stable along the iterations and no noise-overfitting is present for any of the configurations considered. Moreover, we also observe that these different configurations provide comparable restorations in terms of visual quality in 5000 iterations. Furthermore, to maximize the performances of the cDIP-RED method we should set $\beta_{\mathbf{t}} < \beta_{\mathbf{r}}$. This is due to the fact that a bigger value of $\beta_{\mathbf{r}}$ provides more consistency with the initial data. Finally, we stress that the PSNR behaviour reported in Figures 7.4c and 7.4d for the particular *butterfly* test problem are common to all the other tests performed. All these considerations allow us to state that DIP-WTV and cDIP-RED are more robust than DIP and DeepRED with respect to the choice of the hyperparameters values. Moreover, independently on the ADMM penalties parameters setting, if compared to the standard DIP, we can redstop DIP-WTV and cDIP-RED being confident that these methods do not overfit noise.

Satisfying the Morozov's discrepancy principle. In Figure 7.5, we consider once again the denoising test on the *butterfly* image described previously. We analyze the behaviour of the constraint ratio $\|f(\boldsymbol{\theta}^{(k)}, \mathbf{z}) - \mathbf{g}\|/\delta$ as a function of the iterations number. We remark that a constraint ratio equal to 1 entails the corresponding iterate is almost at the boundary of $D_{\sigma_{\mathbf{e}}}$ defined in (7.11). We observe that DIP and DeepRED (setting $\gamma = 24$) slowly overfit the simulated noisy acquisition and converge to an interior point of $D_{\sigma_{\mathbf{e}}}$. On the other hand, DeepRED (with $\gamma = 22$), DIP-WTV and cDIP-RED converge to a solution which lies on the boundary of $D_{\sigma_{\mathbf{e}}}$ and hence implicitly satisfy the discrepancy principle. We stress that we empirically observe the same behaviour for all the other experiments performed. As a general comment, this test confirms once again how the performances of DeepRED largely depend on the choice of the hyperparameter γ defining the strength of the regularization. Moreover, we empirically show a more robust convergent behaviour of DIP-WTV and cDIP-RED avoiding costly parameter tuning.

Denoising task

We validate DIP-WTV and cDIP-RED by comparing them with DIP and DeepRED on the Set5 [167] dataset for the denoising task. The starting noisy images are created by corrupting the ground truth images with an AWGN component of standard deviation equals to 25 and 50. We remark once again that for the cDIP-RED approach we estimate the noise standard deviation even if we know its value. The performances are evaluated by means of the PSNR metric and, in addition, by a visual comparison. In particular, Figures 7.6 and 7.7 report the restored *baby* and *butterfly* images starting from the data with the highest level of corruption considered. In Table 7.1 we report the mean values of the PSNR metric on Set5. For the DIP algorithm we have selected for each image the number of iteration maximizing the PSNR value. For DeepRED we set the ADMM penalty equal to 0.5, whereas we have selected the threshold γ and the starting regularization parameter λ_0

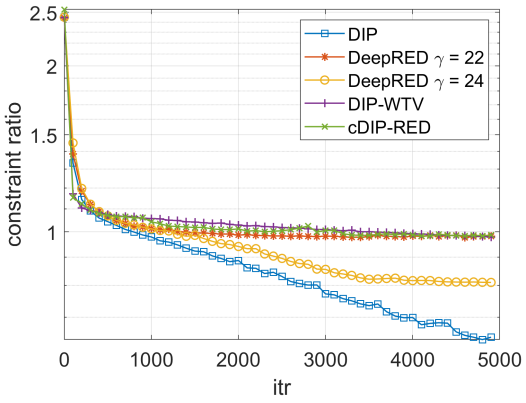
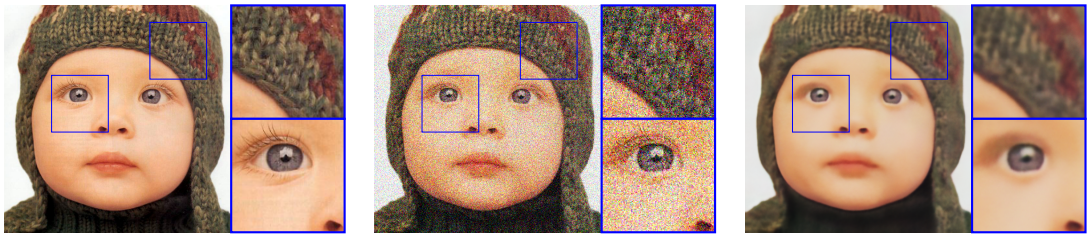


Figure 7.5: The constraint ratio's trend along the iterations obtained by applying DIP, DeepRED (with $\gamma = 22$ and $\gamma = 24$), DIP-WTV and cDIP-RED to the *butterfly* RGB image corrupted by AWGN with $\sigma_{\mathbf{e}} = 35$.

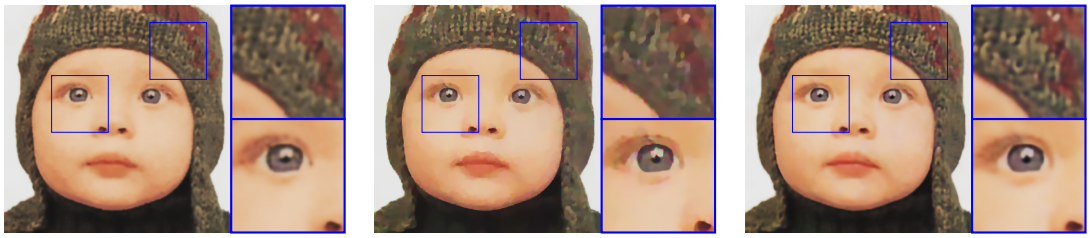
in order to maximize the PSNR for each image. For DIP-WTV and cDIP-RED we set for all the images $\beta_{\mathbf{v}} = 1$ and $\beta_{\mathbf{t}} = 0.5$ and $\beta_{\mathbf{r}} = 1$, respectively. For DeepRED, DIP-WTV and cDIP-RED the restored images have been obtained performing 5000 iterations.
The results reported in Table 7.1 show that cDIP-RED outperforms DIP and provides slightly better performances with respect to DeepRED in terms of PSNR metric. We remark that cDIP-RED does not require any hand-tuning of the regularization parameter. Concerning DIP-WTV, we observe that it provides better performances than DIP. Moreover, we stress that it has shown more robustness to the choice of the hyperparameters with respect to the DeepRED and it has the lowest number of hyperparameters to be set. Unfortunately, the handcrafted Total Variation regularizer is not as effective as RED regularization for natural images, which manifests in lower PSNR scores for DIP-WTV. In Figures 7.6 and 7.7, we report the simulated noisy acquisitions of the *baby* and *butterfly* images setting $\sigma_{\mathbf{e}} = 50$ and the restored images obtained by DIP, DeepRED, DIP-WTV and cDIP-RED. Moreover, in the captions, we highlight the PSNR values. As a general comment, the DIP algorithm struggles to recover the image texture. The cDIP-RED restorations look sharper and more faithful to the ground truth than the ones obtained by DeepRED and DIP-WTV as underlined by the close-ups.

Table 7.1: PSNR mean values for the Set5 for two level of noise. In blue we highlight the best PSNR value.

	$\sigma_{\mathbf{e}}$	Noisy	DIP	DeepRED	DIP-WTV	cDIP-RED
PSNR	25	25.46	32.29	32.91	32.48	**32.95**
	50	19.89	27.87	28.15	27.98	**28.34**



(a) GT (b) noisy (c) DIP



(d) DeepRED (e) DIP-WTV (f) cDIP-RED

Figure 7.6: Restored images for the *baby* test problem setting $\sigma_{\mathbf{e}} = 50$. The PSNR values are: Noisy: 19.84 dB, DIP: 27.85 dB, DeepRED: 28.32 dB, DIP-WTV: 28.26 dB, cDIP-RED: 28.43 dB.

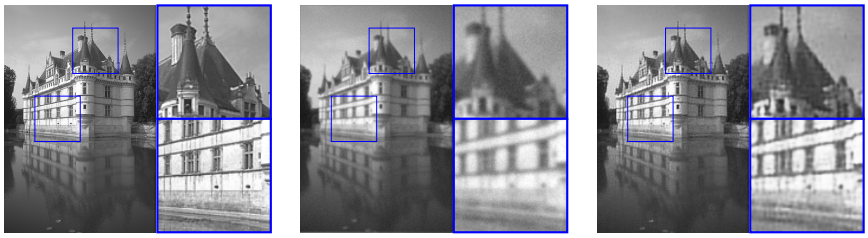
(a) GT (b) noisy (c) DIP

(d) DeepRED (e) DIP-WTV (f) cDIP-RED

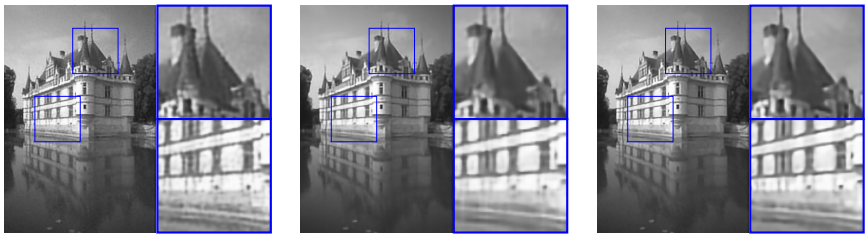
Figure 7.7: Restored images for the *butterfly* test problem setting $\sigma_{\mathbf{e}} = 50$. The PSNR values are: Noisy: 19.88 dB, DIP: 27.81 dB, DeepRED: 28.13 dB, DIP-WTV: 28.01 dB, cDIP-RED: 28.69 dB.

Deblurring task

In this section, we compare DIP-WTV and cDIP-RED with DIP and DeepRED on the Set5 [167] dataset for the deblurring task. The starting degraded images are constructed by setting the standard deviation of the noise $\sigma_{\mathbf{e}} = 10$ and the standard deviation of the Gaussian blur $\sigma_{\mathbf{H}} = 2$.



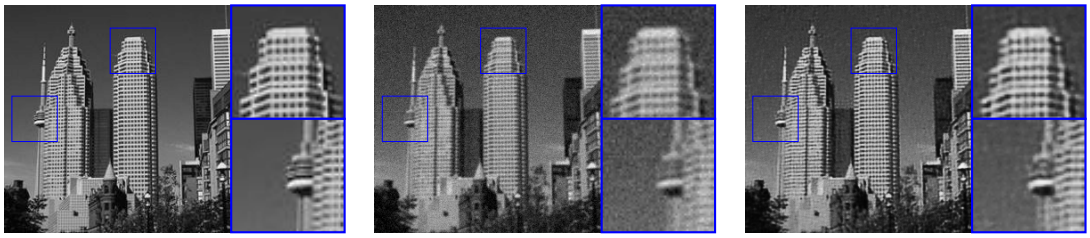
(a) GT (b) noisy (c) DIP



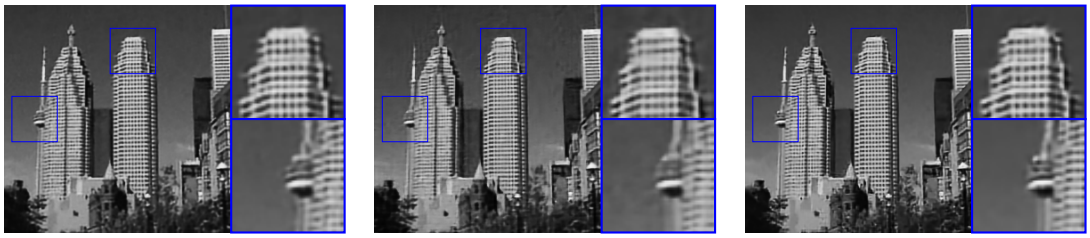
(d) DeepRED (e) DIP-WTV (f) cDIP-RED

Figure 7.8: Restored images for the *watercastle* test problem with $\sigma_{\mathbf{e}} = 5$ and $\sigma_{\mathbf{H}} = 1.6$. The PSNR and SSIM values are: Noisy: 22.87 dB - 0.76, DIP: 25.81 dB - 0.87, DeepRED: 26.23 dB - 0.89, DIP-WTV: 25.87 dB - 0.88, cDIP-RED: 26.28 dB - 0.89.

The performances have been evaluated by means of the PSNR and SSIM metrics. In Table 7.2, we report the mean values of the PSNR and SSIM metrics. Moreover, we consider the *skyscraper* and the *watercastle* images and we add blur and noise by setting $\sigma_{\mathbf{e}} = 10$ and $\sigma_{\mathbf{H}} = 0.8$ for the first image, $\sigma_{\mathbf{e}} = 5$ and $\sigma_{\mathbf{H}} = 1.6$ for the second. The simulated degraded acquisitions are drawn in Figures 7.8b and 7.9b, respectively. In Figures 7.9 and 7.8, we report the results obtained by applying DIP, DeepRED, DIP-WTV and cDIP-RED and in the caption we report the PSNR and SSIM metrics. For the DIP and DeepRED we set all the hyperparameters in order to maximize the PSNR. For DIP-WTV and cDIP-RED we set for all the tests $\beta_{\mathbf{v}} = 1.5$ and $\beta_{\mathbf{t}} = 1.5$ and $\beta_{\mathbf{r}} = 2$, respectively. For DeepRED, DIP-WTV, and cDIP-RED the restored images have been obtained performing 5000 iterations.
From Table 7.2 we observe again that DeepRED and cDIP-RED reach comparable performances on Set5. However, we stress that, differently from DeepRED, the cDIP-RED scheme does not require to fix the regularization parameter. Moreover, DIP-WTV outperforms the standard DIP. For the *watercastle* image, DeepRED, and cDIP-RED reach similar performances in terms of PSNR and SSIM metrics, however the DeepRED restoration looks noisier than the one provided by cDIP-RED. Finally, DIP-WTV always performs better than the standard DIP.
Concerning the *skyscraper* we do not have a ground truth available, therefore we can compare the results only through visual inspection. Indeed, it is clear from Figure 7.9a that the *skyscraper* image is affected by jpeg-compression artifacts. In order to simulate a more realistic acquisition, we further corrupt this compressed image with blur and noise (Figure 7.8b). The close-ups in Figure 7.9 highlight that the output cDIP-RED suppress the artifacts and outperforms the restorations provided by DIP, DeepRED and DIP-WTV in terms of visual quality. In particular, cDIP-RED



(a) Compressed acquisition (b) Blurry and noisy (c) DIP



(d) DeepRED (e) DIP-WTV (f) cDIP-RED

Figure 7.9: Restored images for the *skyscraper* test problem with with $\sigma_{\mathbf{e}} = 10$ and $\sigma_{\mathbf{H}} = 0.8$.

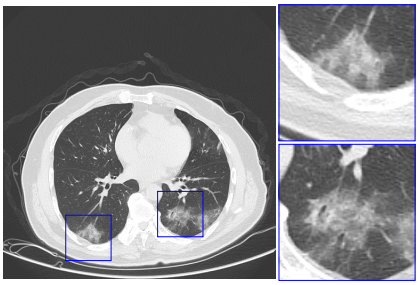
can retrieve better the details and remove the artifacts and the noise.

Table 7.2: PSNR and SSIM mean values for the Set5 considering Gaussian blur with $\sigma_{\mathbf{H}} = 2$ and the noise-level $\sigma_{\mathbf{e}} = 10$. In blue we highlight the best PSNR and SSIM values.

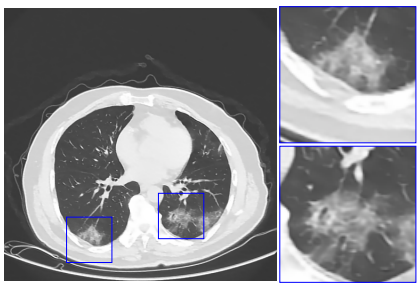
	Blurred	DIP	DeepRED	DIP-WTV	cDIP-RED
PSNR	25.93	30.08	30.81	30.56	**30.90**
SSIM	0.81	0.91	0.92	0.92	**0.93**

Artifact removal for a chest CT image

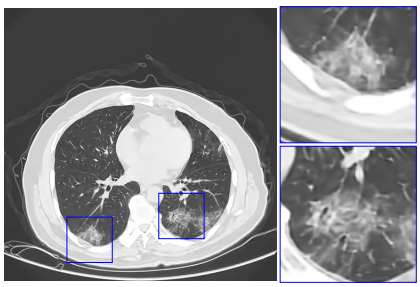
Finally, we show how our methods can be effective for retrieving one real medical chest CT image of a patient affected by COVID-19 [168]. In Figure 7.10a we report the acquired data together with the close-ups of two details (inflammation zones) in the lungs backside where are visible the effects of the interstitial pneumonia caused by COVID-19 disease. From these panels the standard artifacts related to the discrete angles sampling typical of the CT application are clearly visible. In Figures 7.10b and 7.10c, we show the restored images provided by our DIP-WTV and cDIP-RED approaches, respectively. Generally, all finer structures, such as the inflammation details, alveoli and bronchioles, are sufficiently well retrieved, as highlighted by the close-ups. Finally, it is evident that the restoration provided by cDIP-RED looks sharper than the one restored by DIP-WTV.



(a) Acquired chest CT



(b) DIP-WTV



(c) cDIP-RED

Figure 7.10: Reconstructed images for the real CT problem. In (a) we report the acquired data, in (b)-(c) we report the restored images obtained by DIP-WTV and cDIP-RED, respectively.

7.2 Deep Prior for video super resolution in time-lapse microscopy

Light Time-Lapse Microscopy (TLM) imaging is successfully used to acquire and record biological experiments based on Organ-On-Chip (OOC) platforms, which are miniaturized microfluidic devices mimicking in-vitro complex 3D cellular micro-environments [169], such as cell migration [170] or multi-cellular interaction [171, 172].

After acquisition by TLM, the live cell videos are analyzed by means of sophisticated computerized algorithms with the aim of finding biological insights embedded within cell motility. The way in which cells move, indeed, has been discovered meaningful to understand biological processes, such as wound healing [173] and morphogenesis [174] but also cancer growth and spread of metastasis [175]. Usually, the automated exploitation of such devices includes the localization and tracking of cells through increasingly cutting-edge single particle tracking software [176, 177].

Cell trajectories are then coded in time and space domain by extracting some motility descriptors useful to uncover and quantitatively evaluate the response to target therapeutic agents [178, 179]. However, the reached biological conclusions can be compromised by the experimental set-up of TLM in terms of spatiotemporal resolution [180, 181].

First of all, a suitably high frame rate is required to entirely capture and then to accurately estimate the duration of biological events such as apoptosis or cell-cell interaction. Not less relevant, a high spatial resolution implies many benefits for the video analysis. The higher pixel density, characterizing the so-called High Resolution images with respect to Low Resolution ones, reveals to be essential to better distinguish particles within a video and hence to effectively dissect intricate

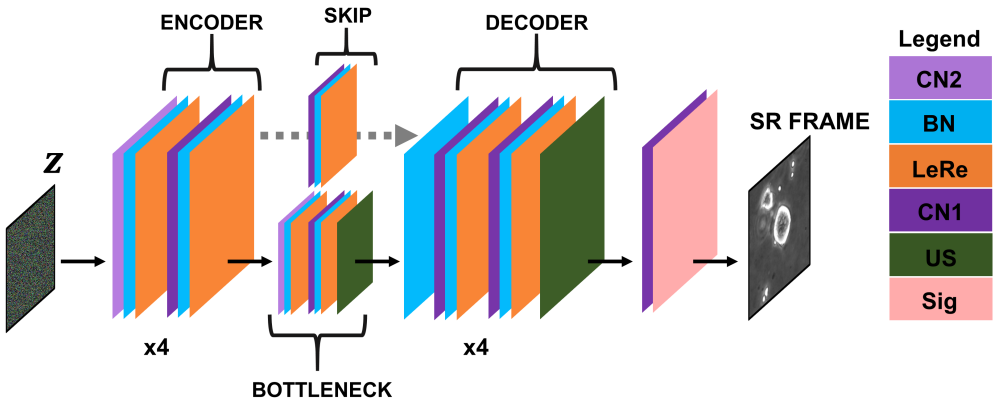


Figure 7.11: Encoder-Decoder architecture with long skip connections. In the legend on the right: CN2 stands for Convolutional layer with stride 2, BN for batch normalization, LeRe for Leaky ReLu activation, CN1 for Convolutional layer with stride 1, US for Lanczos Upsampling, Sig for sigmoid activation. The image z is the random input image of the RDPV method and the SR frame is the computed output.

biological phenomena involving multiple cell populations.
Multi-cellular interaction may constitute an example: during interaction, the distance among cells reduces until they overlap. A high spatial resolution may decrease cell detection and tracking errors showing more defined and detailed shapes and edges. As a result, a more reliable tracking positively affects the trustworthiness of the motility descriptors extracted from the trajectories thus uncovering unbiased biological findings.
Unfortunately, acquiring HR images is not always possible, due to the high cost of the high performing acquisition instruments and physical constraints, such as the optical zoom provided by the camera. However, if a higher magnification is feasible, it does not certainly guarantee a better image clarity. In addition, HR images related to long-term experiments, i.e., from hours to days, can reach a size from tens to hundreds of gigabytes, thus requiring very high processing capabilities. A fair compromise between high computational requirements and the preservation of the biological informative content may be represented by the adoption of super resolution algorithms: experiments can be acquired at a LR and then post-processed by means of SR algorithms.

7.2.1 The proposed Recursive Deep Prior for Videos

In the field of TLM videos it is not possible to have available large datasets of low resolution and high resolution sequences of frames, and moreover, even if the frames recorded belong to the same experiment, some conditions may change from one video to another, thus making supervised learning methods not suitable. Therefore, motivated by the interesting properties of DIP, in [157] we extended this unsupervised method to address the task of video super resolution in TLM. We now describe the proposed method.
First, we consider the TV-regularized DIP model adapted to the super resolution task, which reads:

$$\arg\min_{\boldsymbol{\theta}\in\mathbb{R}^p} \frac{1}{2}\|\mathbf{SH}f_{\boldsymbol{\theta}}(\mathbf{z}) - \mathbf{y}^t\|_2^2, + \lambda \mathrm{TV}_p(f_{\boldsymbol{\theta}}(\mathbf{z})), \tag{7.18}$$

where by $f_{\boldsymbol{\theta}}$ we denote the encoder-decoder CNN architecture sketched in Figure 7.11 whose set of weights is denoted by $\boldsymbol{\theta}$, $\mathbf{y}^t$ is the low resolution TLM video frame at time t and $\mathbf{z}$ is a random image sampled from a uniform distribution having the same spatial size of the unknown HR frame. By TV_p we denote the anisotropic Total Variation when $p = 1$ and the isotropic Total Variation when $p = 2$.

The proposed video super resolution method applies on one video frame at a time. Therefore, given $\mathbf{y}^t$, the LR frame at time t, the method reconstruct the super resolved (SR) frame $\mathbf{u}^t$ computing $f_{\boldsymbol{\theta}_t^*}(\mathbf{z})$, where $\boldsymbol{\theta}^*$ is obtained by early-stopping the optimization process solving (7.18) as for the standard DIP approach.

However, some novelties are introduced. A scheme of the proposed method is depicted in Figure 7.12. The process reconstructing the SR frame at time t is divided in two steps: the initialization and the computation steps. The computation step involves the iterative process solving the optimization problem (7.18), by means of standard iterative gradient-based algorithms, and the reconstruction process computing the SR counterpart of the LR frame $\mathbf{y}^t$. The output of the computation step are the reconstructed SR frame $\mathbf{u}^t$ and the set of weights $\boldsymbol{\theta}_t^*$ such that $\mathbf{u}^t = f_{\boldsymbol{\theta}_t^*}(\mathbf{z})$. The initialization step defines the starting point of the optimization process, namely the set of weights $\boldsymbol{\theta}$ initializing the fixed CNN architecture before optimization.

Given the TLM LR frame $\mathbf{y}^t$, in order to exploit the correlations among neighboring frames, before optimization, the fixed encoder-decoder architecture is initialized by the set of weights $\boldsymbol{\theta}_{t-1}^*$ which is the output of the computation phase at time $t-1$. Furthermore, the computation step, depicted in Figure 7.12 (green square), involves the iterative process solving (7.18) which is early-stopped by means of an adaptive criterion looking at a sliding window of a fixed size including the values of the objective function computed along the iterations. The process is stopped when these values start flattening and reach a steady-state. When $\lambda = 0$ in (7.18), the method is referred to as Recursive Deep Prior Video (RDPV), whereas when $\lambda > 0$, the method is referred to as RDPV-TVi when we choose the isotropic TV (p=2) or RDPV-TVa when we consider the anisotropic TV (p=1).

7.2.2 Numerical results

In this section we report some numerical results in order to validate the efficiency of the RDPV and its TV-based variants, on synthetic as well as real videos from OOC experiments related to tumor-immune interaction.

Materials: synthetic and real data. The dataset of synthetic videos analyzed is based on the work by [180], where the authors derive a stochastic particle model to artificially mimic the migration of immune cells towards a target tumor cell and their consequent interaction. A total amount of 100 synthetic videos is generated. Each video represents a region of interest of 288 $\times$ 288 pixels containing 16 immune cells and one target tumor cell. A total number of 100 frames is collected for each video emulating a video frame acquisition of 20 seconds. The theoretical

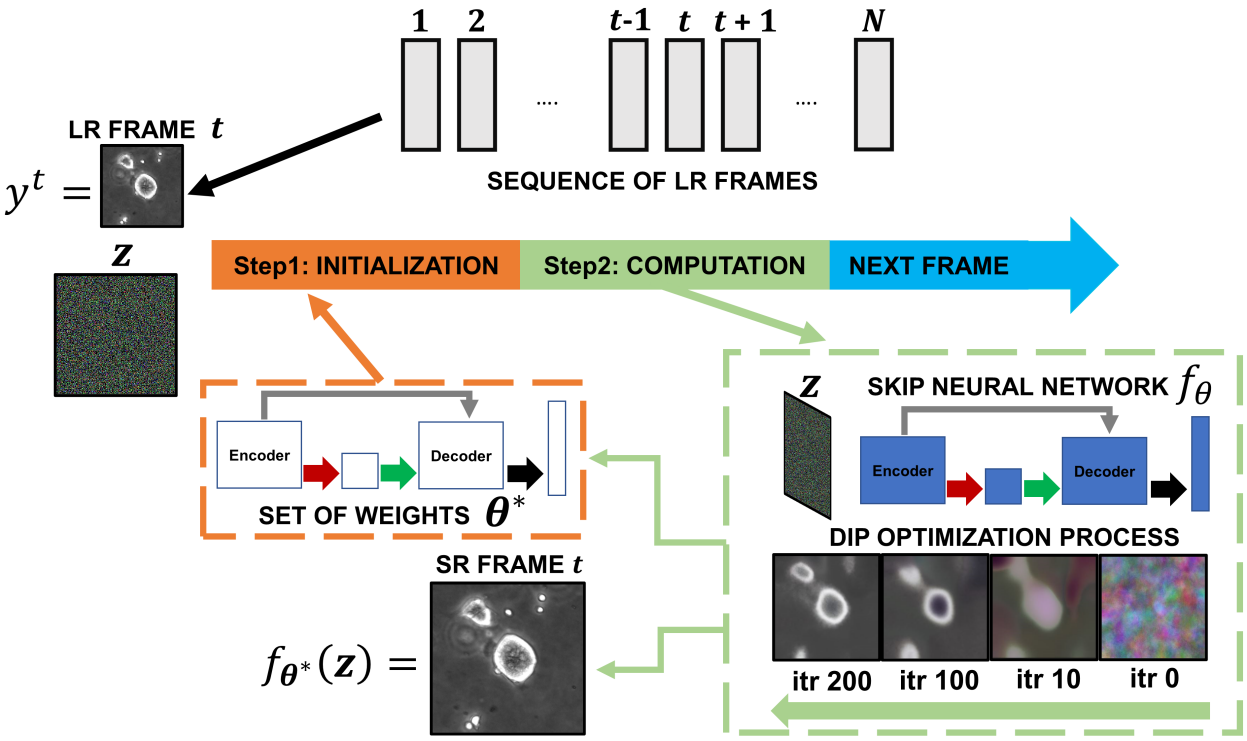


Figure 7.12: RDPV scheme.

trajectories of the total 1600 immune cells (16 for each of the 100 synthetic videos), obtained as result of the implementation of the stochastic particle model, are here considered as ground truth trajectories (GT trajectories). In the following, we will refer to the described atlas of videos as original or noise-free synthetic videos or simply as synthetic videos. The original videos are further corrupted by adding white Gaussian noise with standard deviation equal to 0.001.
The proposed approach is validated on two sets of real cell videos, namely Videos Type 1 and Videos Type 2, each one containing 10 videos, respectively. Each video represents a Region of Interest (ROI) of 288×288 pixels with a tumor cell occupying its center. Each video frame has been acquired every two minutes. Videos Type 1 counts 31 frames, while Videos Type 2 are composed by 57 frames. For the sake of brevity we here omit further information about the considered data. However, the interested reader can refer to [157] and references therein. We stress that the original synthetic and real videos are used as ground-truths in the following. In order to obtain LR frames, all the videos have been downsampled by a factor equal to 4 using the standard Bicubic interpolation and then reconstructed by SR methods setting $L = 4$.

Evaluation metrics. A two-fold evaluation process is carried on. Both synthetic and real videos are involved in a test to verify image similarity between original HR (described previously) and reconstructed SR video frames based on the computation of the PSNR metric. For both synthetic and real videos, the average PSNR values computed over all the video frames are calculated. The original synthetic videos are also utilized to evaluate how the performance of the tracking software is affected by the proposed SR algorithms. The tracking algorithm used is the Cell Hunter (CH) [180] software using the Circular Hough Transform (CHT) [182] for localizing the

cell and the Munkres' algorithm [183] for the assignment of the trajectories. More specifically, immune cell trajectories detected from the tracking algorithm on LR videos (LR trajectories), on original HR videos (HR trajectories) and on SR videos (SR trajectories) are compared with the theoretical/ground truth trajectories (GT trajectories), i.e., trajectories directly obtained from the implementation of the stochastic particle model in the creation phase of the synthetic videos [180]. This kind of analysis is accomplished only on synthetic videos because theoretical trajectories are not available in real contexts. The tracking algorithm could fail by missing trajectories or by associating tracts corresponding to different detected trajectories to a single ground truth trajectory. For this reason, tracking performances are assessed by computing the percentage of detected cell trajectories with respect to the total number of the ground truth ones and the so-called swapping error [184] that measures the average number of swaps per trajectory.

Comparisons. We compare the implemented RDPV, RDPV-TVi and RDPV-TVa with some of the best performing trained methods, ESRGAN [185], proSRGAN, proSR, proSRs [186], RCAN [187], EDSR [63], all of them applied one frame at time. Moreover, we consider also the Deep Prior Video (DPV) method, which applies the standard DIP frame by frame without considering any initialization.

Implementation details. The computation step requires the usage of a gradient-based algorithm to solve (7.18). As suggested in [39], in our experiments, we use the ADAM scheme by fixing the learning rate at 10^{-3}. The core idea of the automatic early stopping procedure, is to stop the iterations based on the behaviour of the objective function in (7.18). More precisely, this technique considers a window (patience) of consecutive values of the objective function and then, if there is not enough decrease during the iterative process, the algorithms are stopped before a fixed maximum number of iterations is reached.
For the first frame of synthetic videos, we impose a maximum of 1000 iterations with early stopping starting from 500 iterations (patience = 50). For the subsequent frames, due to the initialization step, the number of iterations can be reduced: we set a maximum of 500 iterations with early stopping starting from 300 iterations (patience = 50). For the first frame of real videos, we impose a maximum of 3000 iterations with early stopping starting from 2000 iterations (patience = 50). For the subsequent frames the number of iterations is reduced: we set a maximum of 2000 iterations with early stopping starting from 1000 iterations (patience = 50). For a fair comparison, we set the maximum number of iterations for DPV as the number of iterations performed by RDPV according to the aforementioned stopping criterion.
These hyperparameters, i.e., the maximum number of iterations and the starting of the early stopping rule, are chosen in order to provide a fair compromise between reconstruction quality and computational time. Since the best reconstructions are obtained when the loss starts flattening and reaches a steady-state, the starting of the early stopping rule is imposed at about half of the maximum number of iterations. Postponing the starting of the early stopping rule makes the network more prone to over-learn the image statistics also reproducing the noise affecting the LR frame in the SR reconstruction. Moreover, we remark that for all the processed synthetic and real frames, the early stopping ends the iterative process before the maximum number of iterations is

reached.
Concerning the proposed regularized methods (RDPV-TVa and RDPV-TVi), we set the trade-off parameter λ in (7.18) equal to 0.008 for all the reconstructed videos. In our experiments, comparable performances are obtained in terms of PSNR if the λ is chosen in the range $[10^{-3}, 10^{-2}]$.

The importance of super resolution on cell tracking

Video analysis usually starts with cell localization and tracking. The errors made by tracking algorithms in this phase can propagate and heavily compromise the extrapolated biological conclusions. When the spatial resolution is low, the tracking software could fail in its localization task.
In order to corroborate this statement, in Figure 7.13 we show how the localization algorithm (CHT) acts on LR and SR frames obtained by a coarse SR algorithm (Bicubic interpolation) and the proposed RDPV, respectively. More precisely, the left panel of Figure 7.13 depicts an example of cell localization on the LR synthetic video frame where localized cells are marked as red circles and missed cells are pointed out by black arrows. As a result, we observe that some cells are missed, and others are identified as unique entities because partially overlapped. The bicubic outcome reveals to be blurred with the contour of the cells not sharp. As a consequence, this smoothing effect damages the edge detection of the circular-shaped objects by the CHT. Indeed, as depicted in the central panel of Figure 7.13, the lack of fine details may lead to inaccurate localization. It is evident how the software is able to distinguish cells but misses some of them. As shown in the right panel of Figure 7.13 the proposed RDPV allows increasing the trustworthiness of tracking software in localizing cells so that overcoming the limitation of coarse results obtained by the standard bicubic algorithm. This simple test does not only highlight an interesting connection between pixel density and tracking performances, but also the importance of good quality SR reconstructions.

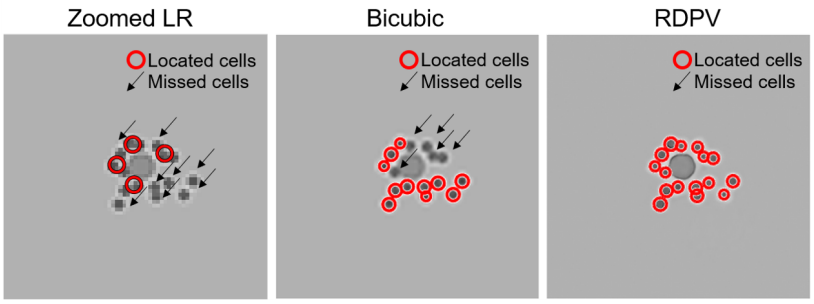


Figure 7.13: Immune cell localization on synthetic videos. Example of cell localization by means of Cell Hunter software on a LR frame (left panel) and the corresponding SR counterparts reconstructed by the Bicubic method (central panel) and the proposed RDPV method (right panel).

A visual comparison between GT, HR, SR, and LR trajectories on synthetic video frames is provided in Figure 7.14. A zoom is supplied for a better visualization of the LR trajectories. Conversely to LR trajectories, the appearance of HR and SR trajectories is very similar to that of GT ones. From a quantitative point of view, the percentage of total number of detected LR trajectories on the 100 synthetic videos with respect to the overall number of the GT ones (1600) is equal to 51%, while it reaches the 100% for the detection of both the HR and the SR trajectories. The swapping error

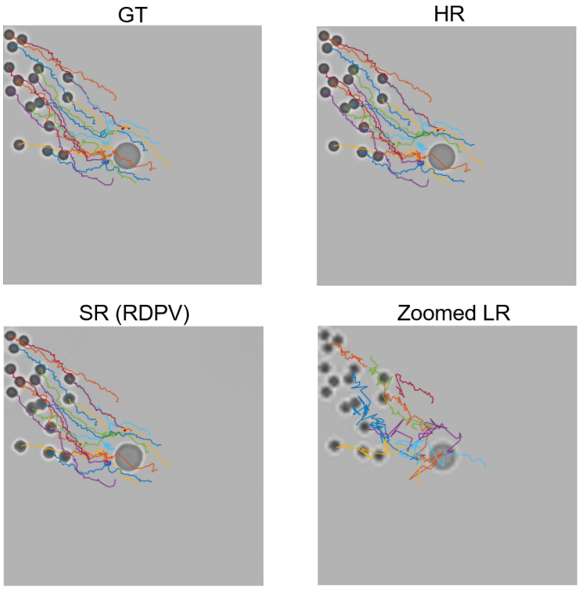


Figure 7.14: Visual representation of immune cell trajectories on synthetic videos. Ground-truth (GT) trajectories directly obtained from the implementation of the model in the creation phase of the synthetic videos (upper-left panel). High resolution (HR) trajectories identified by Cell Hunter software on the original HR video (upper-right panel). Super resolution (SR) trajectories identified by Cell Hunter software on the SR video, reconstructed by the proposed RDPV method (lower-left panel). Lower resolution (LR) trajectories identified by Cell Hunter software on the LR video (lower-right panel). A zoom of LR trajectories is provided for a better visualization (Zoomed LR).

counts 18.4, 3 and 3.5 swaps per trajectory, for the LR, HR and SR trajectories, respectively. This is a consequence of the previously underlined CHT errors made on LR video frames (Figure 7.13), since the estimation of trajectories is strictly correlated to the detection of cells frame by frame.

Finally, in Figure 7.15 an example of a Video Type 1 frame reconstructed with RDPV is shown. The favorable effect of super resolution on cell localization, edge map detection and cell tracking is also highlighted. The application of the RDPV algorithm allows us to reduce the false occurrences during the cell localization phase (Cell localization in Figure 7.15), to successfully separate cells in contact (Edge detector in Figure 7.15) and to effectively construct cell trajectories (Cell tracking in Figure 7.15).

Image quality evaluation on synthetic cell videos

So far, we have underlined the importance of the high spatial resolution for TLM videos in order to get a successful tracking. In Figure 7.13, we test the CHT localization algorithm on the SR output of the bicubic algorithm, highlighting that sophisticated SR algorithms are required in order to retrieve fine image details. In Figure 7.16, we show the distributions of the average PSNR values computed over all the frames of each of the 100 synthetic videos for all the above mentioned methods. Figure 7.16a refers to the original synthetic videos, whereas Figure 7.16b refers to the corrupted synthetic videos. The proposed RDPV and RDPV-TVi algorithms always outperform

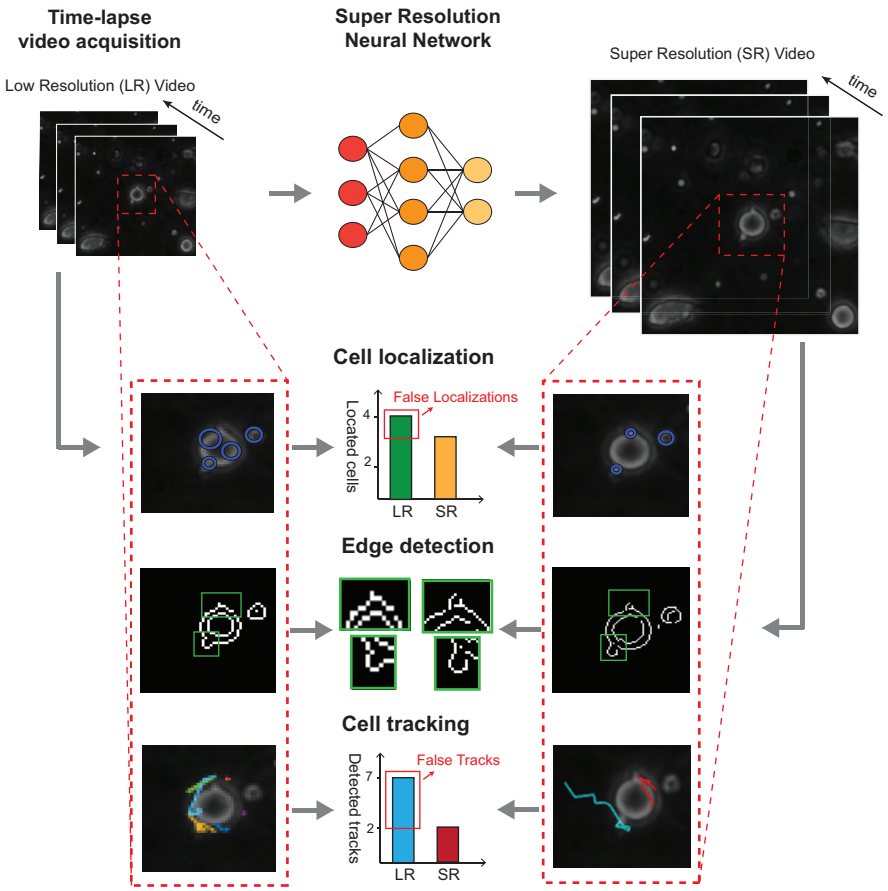


Figure 7.15: More effective cell motility evaluation moving from Lower Resolution to Super Resolution videos. The number of false occurrences during the cell localization phase is reduced (Cell localization); cells in contact are separated thanks to an efficient edge detection (Edge detector); cell trajectories are effectively constructed (Cell tracking).

the standard DPV. This is because RDPV implements the new recursive updating rule which takes into account the knowledge acquired by the previous reconstructed frames. Moreover, the regularization term within RDPV-TVi adds to the model prior information on the solution that are not completely captured by the fixed CNN architecture. Very promising results are achieved from the comparisons with the trained networks. Comparable performances are observed on noise-free artificial videos (Figure 7.16a). Better performances are achieved on corrupted artificial videos (Figure 7.16b). This is because one of the main drawbacks of trained architecture is the instability with respect to the presence of noise components in the input data. They are not able to filter out the added noise from test images if the training dataset does not present a considerable number of noisy-images at different levels of noise. Anyway, this necessity might be limiting in practical applications. The presented results stress the importance of developing an algorithm whose output is not dependent on a fixed set of image examples.

According to such quantitative results, we depict images obtained by the worst and the best trained methods (ESRGAN and RCAN, respectively) alongside GT, LR, DPV, RDPV and RDPV-TVi images. More specifically, Figure 7.17 highlights one of the video frames from the corrupted synthetic

videos. We observe RCAN achieves promising performances in terms of average PSNR but, as emerged from Figure 7.17, the immune cells shape appears distort (red and green arrows). The same warping also holds for the ESRGAN method. The standard DPV, instead, is not able to guarantee a fine result, conversely to RDPV which, on its part, has the drawback to not effectively separate some of the immune cells (red and green arrows). The addition of TV regularization (RDPV-TVi) leads to optimal results both in terms of immune cell shape and differentiation.

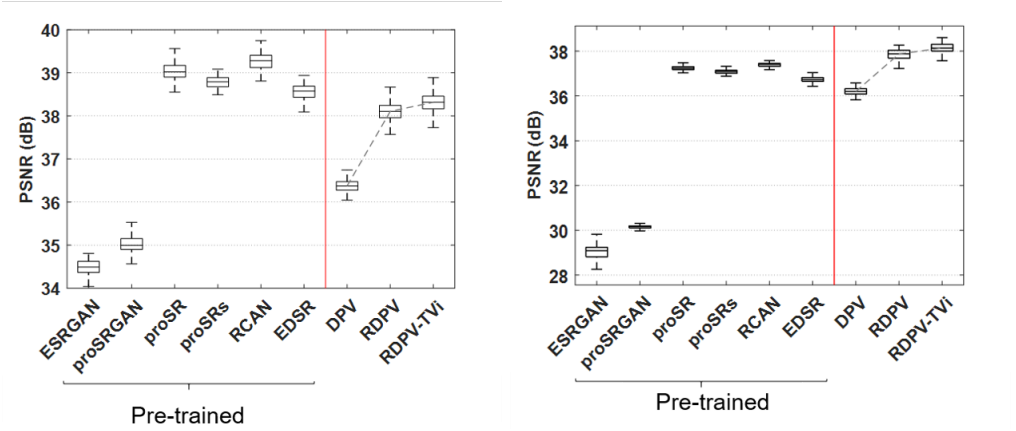


Figure 7.16: (a) Quantitative results in terms of PSNR on synthetic videos ($\sigma = 0$, upper panel) and (b) their corrupted counterparts ($\sigma = 0.001$, lower panel). Boxplots comprise the average values of PSNR computed over all the frames of each synthetic video.

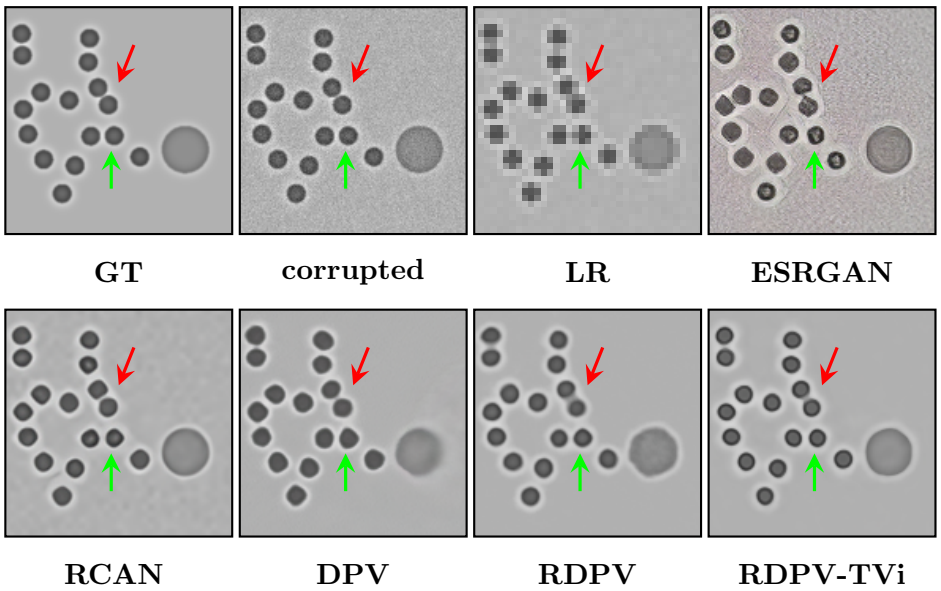


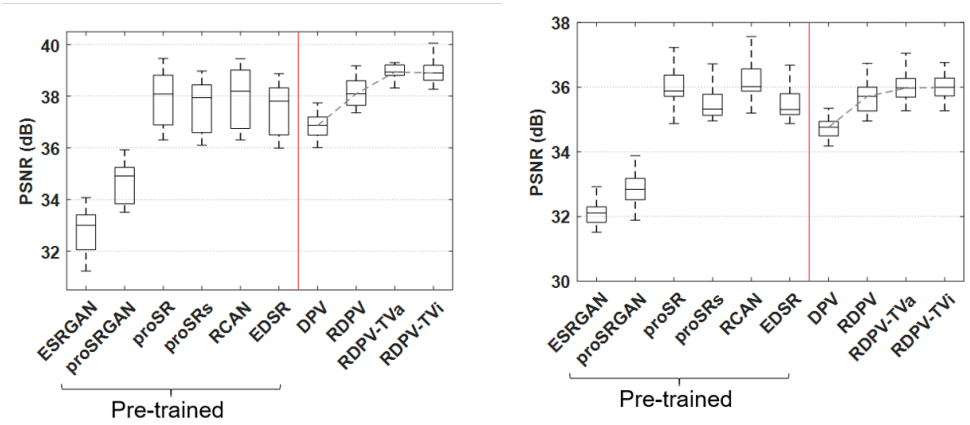
Figure 7.17: Synthetic Frame reconstruction close-ups (x2), $L = 4$, $\sigma_{\mathbf{e}} = 0.001$.

Image quality evaluation on real cell videos

We finally validate the proposed approach on videos from two diverse OOC experiments by exploiting tumor-immune interaction: Videos Type 1 and Videos Type 2. In Figure 7.18a and Figure 7.18b, we show the average PSNR values for Video Type 1 and Video Type 2, respectively. The average PSNR values are computed over all the frames of each of the real videos of the two types.

The proposed methods RDPV,RDPV-TVi and RDPV-TVa outperform the standard DPV both for Video Type 1 and Video Type 2 in terms of average PSNR. We stress that the addition of both isotropic and anisotropic Total Variation improve the performances of the RDPV, confirming once again the importance of these additional terms. For what concerns the comparisons with the trained architecture, RDPV, RDPV-TVi and RDPV-TVa outperform them in terms of average PSNR on Video Type 1 (Figure 7.18a) and reach comparable performances on Video Type 2 (Figure 7.18b). Not less relevant, from the PSNR distributions shown in the boxplots in Figure 7.18, it is remarkable to see that the distributions of the average PSNR values have a high variance whereas unsupervised methods have more stationary performances over all the tests executed on real videos. This is more evident for the average PSNR distributions on Video Type 1 (Figure 7.18a). This highlights a stronger sensitivity to the given input frames for the trained methods with respect to the unsupervised ones. We stress that, for each of the two video types, even if the frames recorded belong to the same experiment, some conditions may change from one video (or even frames) to another, such as the brightness of the FOV. Indeed, analyzing the acquired videos, we observe that sudden changes of luminosity are evident especially for Videos Type 1, thus negatively affecting the performances of trained methods in some cases. This confirms the need of unsupervised methods for TLM videos super resolution, since it not feasible to collect a dataset accounting all the possible real boundary conditions.

As qualitative evaluation, in Figure 7.19 we depict a video frame example (from Videos Type 2) obtained by the worst and the best trained methods (ESRGAN and RCAN, respectively) alongside GT, LR, DPV, RDPV and RDPV-TVi images. In Figure 7.19, three regions of interest are closed-up and highlighted by using three different coloured squares: red, green and blue. All the approaches increase the resolution of the starting LR frame and in particular all of them are able to separate the cells belonging to the cluster within the red square. As it is evident the trained approaches introduce artifacts alongside the cell cluster (red square). Such artifacts can dramatically affect the cell tracking software in the localization phase thus identifying false positive cell candidates. We observe that all the unsupervised methods reach quite similar results with respect to the ground-truth image. However, the cells within the cluster look better separated for the image obtained by RDPV-TVi, thus confirming that the addition of the Total Variation regularizer improves the results.



(a) (b)

Figure 7.18: (a) Quantitative results in terms of PSNR on real Videos Type 1 (upper panel) and Type 2 (lower panel). Boxplots comprise the average values of PSNR computed over all the frames of each real video.

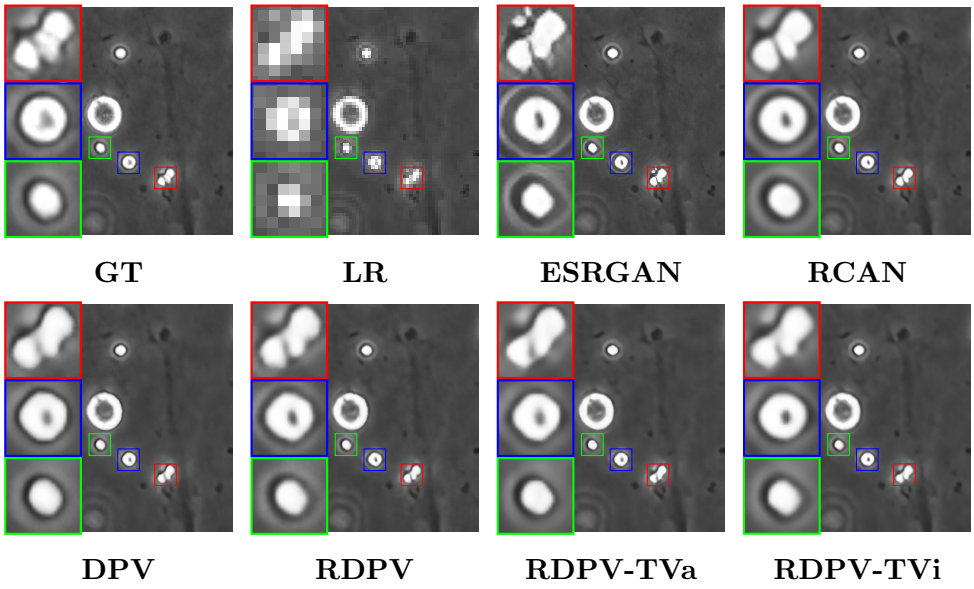


Figure 7.19: Real Frame of Videos Type 2, $L = 4$.Three regions of interest are also closed-up and highlighted by blue, green and red squares.

Chapter 8

Recurrent Neural Networks for GNSS time series modeling

Global Navigation Satellite System (GNSS) is the standard generic term for satellite navigation systems that provide autonomous geo-spatial positioning with global coverage. More precisely, GNSS refers to a constellation of satellites, constantly orbiting around the Earth, which transmit positioning and timing data to GNSS receivers. The receivers are endowed of small chips which are able to capture and analyze the signal emitted by a satellite in order to determine their location on the Earth. Examples of GNSS constellation include Europe's Galileo, the USA's NAVSTAR Global Positioning System (GPS), Russia's Global'naya Navigatsionnaya Sputnikovaya Sistema (GLONASS) and China's BeiDou Navigation Satellite System.

A common operation to find an object location using its distances to three other known points or stations is called 3D-trilateration. In Figure 8.1 we provide a scheme of the 3D-trilateration used by GNSS technology to find the position of a receiver R on the Earth. Given a satellite $S1$, the information sent to the receiver R are: the satellite position x_{S1} and the recorded time t_{S1}. The receiver has a quartz cristals internal clock, therefore it measures the time t_R the signal arrives. Then, the distance d_1 between the satellite $S1$ and the receiver R is approximated as the product of the speed of light c and the difference $t_{R1} - t_{S1}$. Unfortunately, one satellite is not enough to provide a precise location of R on the Earth since the intersection between the sphere of center $S1$ and ray d_1 and the Earth is a large area. Mathematically, in order to narrow down the area we need at least three satellites. In principle, three satellites should be enough to identify the exact location of R on the Earth but, in practice, more satellites are needed. This is due to the many reasons: the receiver uses quartz cristals clock which do not provide precise timing as much as the atomic clock used by satellites and the line of sight can be obstructed thus negatively affecting the transmission. All these sources of error compromise the accuracy of the computed distances, hence having access to more than three satellites is also a benefit to reduce the error. Generally, the more signals from satellites are used in the position estimation the more accurate the GNSS unit can estimate the position.

Nowadays GNSS, is mainly used for many monitoring applications. Thanks to the capability of these systems to provide continuously acquired data, which are converted in three-dimensional

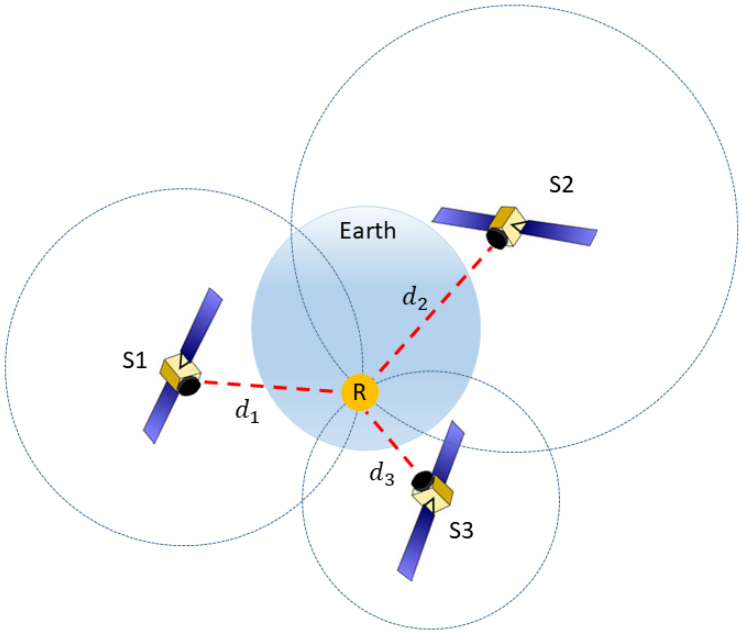


Figure 8.1: 3D-trilateration scheme in GNSS technology.

coordinates after data processing, the monitoring is usually performed through time series analysis. GNSS monitoring can have different purposes depending on the monitored object and the data processing varies consequently. Moreover, the analysis can have different goals: sometimes a signal-denoising is required in order to allow more accurate characterization of the signals, whereas in other cases the goal is a reliable prediction of the coordinates that will be obtained from the incoming data. For GNSS time series many forecasting models have been proposed, based on ARMA, ARIMA and Kalman methods [188, 189, 190, 191, 192], and many denoising models, based on moving averages, sequential filtering and, recently, based on deep learning [193, 194, 195]. The class of DNN algorithms provides a family of networks, suited for sequential data processing, called Recurrent Neural Networks (RNNs). Among them Long Short-Term Memory (LSTM) models, introduced at the beginning to solve vanishing gradient problems which affect all the RNNs [196, 197, 198], has showed its strength for almost all of the time dependent problems. The LSTM is characterized by a gated structure which allows it to store past sequence features in its memory block, bringing out them in the output and preserving long term dependences. LSTM approach has achieved grateful results in speech recognition [199, 200], hand-writing recognition [201] and recently in Traffic Flow Prediction [202], Real Time Autonomous Vehicle Navigation [203].

Contribution

In this chapter we consider a RNN for prediction and denoising of GNSS time series. The implemented network adds to the LSTM layer an activation hyperbolic tangent layer and a Fully Connected layer. We provide specific hints on the hyperparameters choice for solving the denoising and prediction tasks. This chapter is based on the conference paper [204].

8.1 The proposed LSTM-Full neural network

In this section we review the main notions about RNNs. In particular, we focus on the LSTM architecture which represents a powerful tool for sequential data processing. Finally, we describe our model LSTM-Full model.

Recurrent Neural Networks

RNNs are a family of Neural Networks (see Appendix A) suited for time series processing. In RNNs the output at previous time steps affects the output at the current time step so as they are able to catch long term dependencies in sequential data. Given a starting hidden layer vector $\mathbf{h}_0$, then for each time step t and for each input vector $\mathbf{x}_t$ the forward equations of a general RNN are:

$$\mathbf{h}_t = \mathbf{W}_{hh}\mathbf{h}_{t-1} + \mathbf{U}_{xh}\mathbf{x}_t + \mathbf{b}_h \tag{8.1}$$

$$\mathbf{y}_t = \Phi(\mathbf{h}_t) \tag{8.2}$$

where $\mathbf{W}_{hh}$ denotes the hidden-to-hidden weight matrix, $\mathbf{U}_{xh}$ the input-to-hidden weight matrix, $\mathbf{b}_h$ the bias vector and Φ is a pointwise non-linear activation function. The main difference between Feed Forward Neural Networks (FNNs) (see Appendix A) and RNN, is that the latter have a temporal structure. As for FFNNs, gradient based algorithms are used to optimize a loss function respect to the unknown weights and the Back Propagation Through Time (BPTT) algorithm is employed to compute the gradients. One drawback of RNNs is that BPTT algorithm computes gradients that tend to vanish or explode due to the fact that we are composing many times the same non linear function Φ [196, 197].
The most effective model used in applications are gated RNNs such as Long Short-Term Memory (LSTM) nets. A LSTM network [205] is made up of LSTM units showed in Figure 8.2b. Generally, a LSTM unit is composed of a cell which is able to record the main information over long time intervals [198] and three different gates: the forget gate, the input gate and the output gate. These gates have the task to supervise the flow of information and prevent vanishing gradient problems. The forget gate uses a sigmoid function to decide which information has to be taken into account in the previous cell state. The input gate decides which new information has to be stored in the cell memory. The output gate decides the contents of the output vector. Each gate takes the same input: $\mathbf{x}_t$ the current input and $\mathbf{h}_{t-1}$ the previous hidden layer vector. The LSTM unit forward equations are presented below:

$$\mathbf{f}_t = \sigma(\mathbf{U}_f\mathbf{x}_t + \mathbf{W}_f\mathbf{h}_{t-1} + \mathbf{b}_f) \tag{8.3}$$

$$\mathbf{i}_t = \sigma(\mathbf{U}_i\mathbf{x}_t + \mathbf{W}_i\mathbf{h}_{t-1} + \mathbf{b}_i) \tag{8.4}$$

$$\mathbf{g}_t = \tanh(\mathbf{U}_g\mathbf{x}_t + \mathbf{W}_g\mathbf{h}_{t-1} + \mathbf{b}_g) \tag{8.5}$$

$$\mathbf{c}_t = \mathbf{i}_t * \mathbf{g}_t + \mathbf{f}_t * \mathbf{c}_{t-1} \tag{8.6}$$

$$\mathbf{o}_t = \sigma(\mathbf{U}_o\mathbf{x}_t + \mathbf{W}_o\mathbf{h}_{t-1} + \mathbf{b}_o) \tag{8.7}$$

$$\mathbf{h}_t = \tanh(\mathbf{c}_t) * \mathbf{o}_t \tag{8.8}$$

where $*$ represents the pointwise product, $\mathbf{f}_t, \mathbf{i}_t, \mathbf{o}_t$ are respectively the forget gate, the input gate and the output gate vectors at time t, $\mathbf{c}_t$ is the cell vector at time t and the terms $\mathbf{U}$ and $\mathbf{W}$ in each equation are the weight matrices.

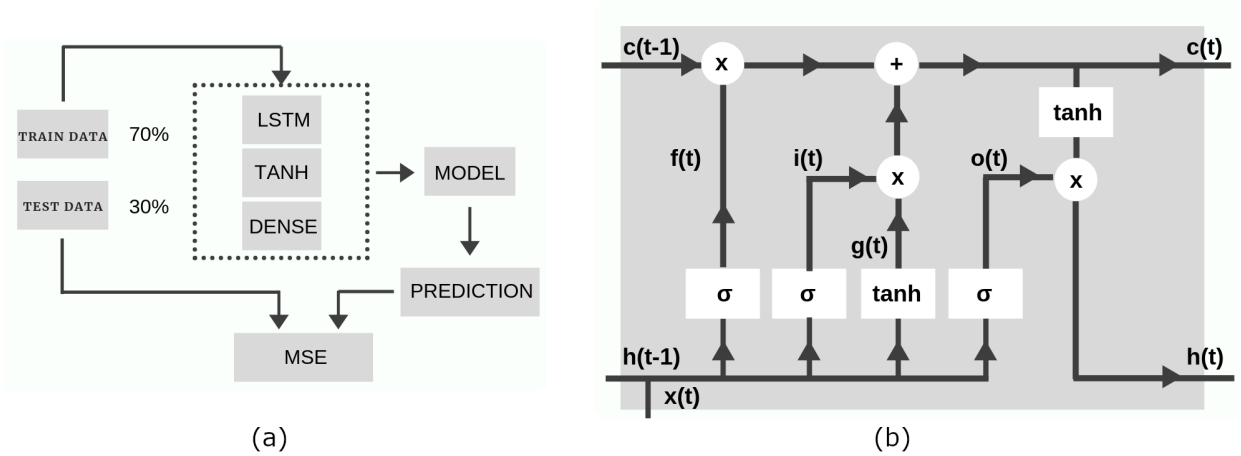


Figure 8.2: (a) Complete process used in this study. (b) LSTM unit.

The LSTM-Full architecture

Given the input sequence $\mathbf{x}_t \in \mathbb{R}^h$ at time t, the model we propose is a RNN-based model whose equations are:

$$\mathbf{h}_t^1 = \Phi(\mathbf{x}_t, \mathbf{h}_{t-1}^1) \tag{8.9}$$

$$\mathbf{h}_t^2 = \tanh(\mathbf{h}_t^2) \tag{8.10}$$

$$\mathbf{y}_t = \mathbf{W}_{h^2}\mathbf{h}_t^2 + \mathbf{b}_{h^2}, \tag{8.11}$$

where Φ is taken as the LSTM unit with the forward equations (8.3)-(8.8). We add after the classic LSTM unit: an activation tanh layer and then a full connected layer that realizes a product between the matrix of weights $\mathbf{W}_{h^2}$ and the hidden vector $\mathbf{h}_t^2$.
A crucial point of this architecture is the choice of the dimension of the vector $\mathbf{h}_t^1$ which represents the "memory" of our model (it depends on $\mathbf{h}_{t-1}^1$) and, from a computational perspective, influences the number of the weights of the whole architecture. We remark that for each input vector $\mathbf{x}_t$, the hidden layer vector $\mathbf{h}_t^1$ has the same dimension. In the following we indicate by H the length of the LSTM hidden vector and we refer to the model described by equations (8.9)-(8.11) as LSTM-Full.

We can estimate the weights on the LSTM-Full model within a supervised learning framework. Therefore, we need to define the form of the input and the form of the label vectors. We now consider an observed time series $\{\mathbf{u}_i\}_{i=1\ldots T}$ where T is the number of samples. By recalling the model of reference (1.6), we here assume the unknown system $\mathbf{A}$ is approximated by the LSTM-Full parametric model. The $T-h$ input vector of the model are defined as follows:

$$\mathbf{x}_{t-h} = [\mathbf{u}_{t-1}, \ldots, \mathbf{u}_{t-h}], \quad t \in [h+1, T], \tag{8.12}$$

where h is the so-called sliding window parameter representing the length of each vector $\mathbf{x}_t$. The $T-h$ label vector are defined as shown below:

$$\mathbf{y}_{t-h} = [\mathbf{u}_t, \mathbf{u}_{t+1}, \dots \mathbf{u}_{t+s-1}], \quad t \in [h+1, T], \tag{8.13}$$

where s represents the number of rows of the weights matrix $\mathbf{W}_{h^2}$. For example, if $s = 1$ the length of the label vector is one. In other words, we use h time steps to predict a single value of the time series.
In the following analysis the dataset is divided in two disjoint subsets: the training set (taken as 70% of the whole data set) and the test set (30% of the whole data set). The first is used to optimize the model parameters, whereas the other is employed to evaluate model accuracy (see Figure 8.2a).

8.2 Numerical results

In this section we carry out an objective analysis in order to investigate the properties of the proposed LSTM-Full method and demonstrate its effectiveness. We consider three sets of data in order to evaluate the efficiency of the designed LSTM-Full network in GNSS time series analysis. The first time series is a synthetic one, which allows to compare the output of the network to a known "ground truth" and has been created taking into account the well-known characteristics of a geodetic time series of daily positions. The second time series derives from a real GNSS permanent station that daily processes the data using a static approach and it is characterized by a comparatively high signal to noise ratio. The third time series comes from a monitoring GNSS station used for early warning monitoring and it is characterized by a low signal compared to the measurements.

8.2.1 A synthetic time series

To create a synthetic time series that simulates properly a real GNSS one, it is important to understand which are the main characteristics of the GNSS positioning. Using GNSS signals a receiver can estimate its position, with different levels of accuracy, measuring the transmitting time of GNSS signals emitted from four or more GNSS satellites and knowing the position of each satellite during the time. The signal crosses the atmosphere with a speed close to the speed of light depending on the physical characteristics of the atmosphere that change continuously. Therefore a GNSS receiver installed on a building roof or on a stable rock can measure during the time some periodical effects due to the seasonal thermal dilatation or solid earth and ocean tides. For all these reasons a GNSS time series can be composed by some periodical terms and, as literature shows, a noisy component which is the sum of a white noise that follows a normal distribution and a random walk noise. The theoretical synthetic time series is constructed as follows:

$$\mathbf{u}_t^0 = Asin(2\pi f t) + q \quad \text{for} \quad t = 1 \dots 4000, \tag{8.14}$$

where we choose $q = 0.01, A = 0.01, f = \dfrac{1}{365}$. We assume that u, q, A are expressed in meters (m) whereas t is expressed in days (d) and f is expressed in d^{-1}. In order to represent a realistic

case study we add: $\mathbf{e}$ a Gaussian noise component, with mean equal to zero and standard deviation $p = 2$ mm and $\mathbf{r}$ a random walk noise or red noise component generated using a normal distribution with mean zero and standard deviation equal to 0.03 mm. Hence, we construct the raw time-series as follows:

$$\mathbf{u}_t = \mathbf{u}_t^0 + \mathbf{e}_t + \mathbf{r}_t \quad \text{for} \quad t = 1 \dots 4000. \tag{8.15}$$

Since in our model we use the activation *tanh* layer, we scale all the time series values in $[-1, 1]$. This pre-processing increases the net performances. The scaled time series $\mathbf{u}$ and $\mathbf{u}^0$ are shown in Figure 8.3a using the green and blue dots, respectively.

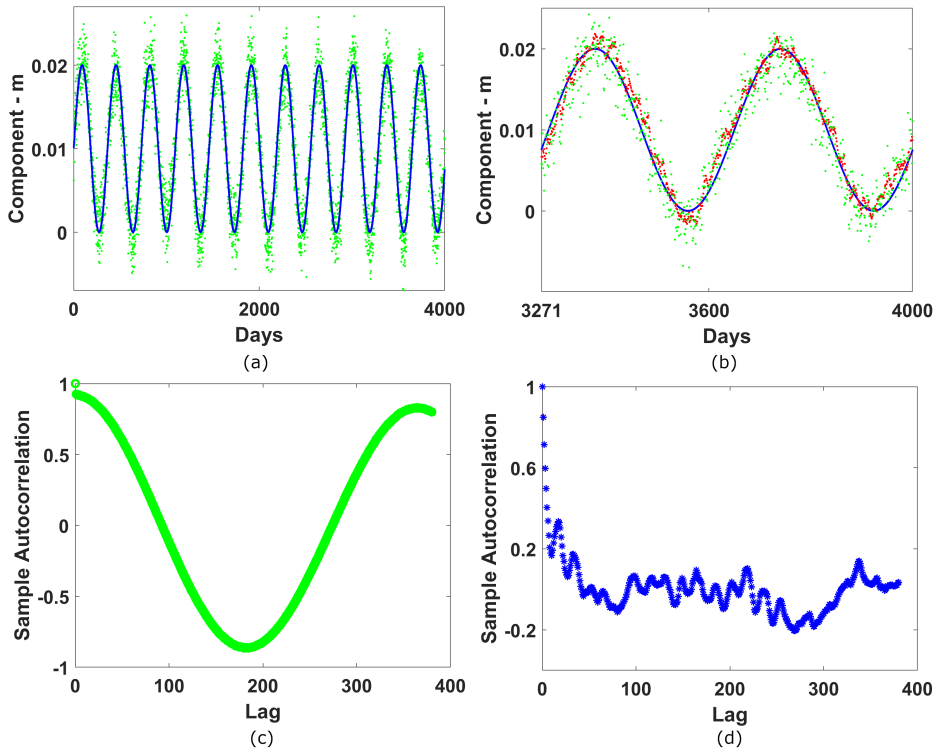


Figure 8.3: (a) Plot over 4000 days of the raw synthetic time series (green dots) and the theoretical sinusoidal signal (line blue). (b) Predicted time steps over 730 days (red dots) compared with the raw signal (green dots) and the theoretical sinuisoidal signal (blue line).(c) ACF over 400 lag of the raw time series. (d) ACF of the prediction error over 365 lags.

Synthetic time series analysis

The purposes of this analysis are to evaluate (i) if the model proposed can predict the theoretical time series $\mathbf{u}^0$ defined in (8.14) using only the raw signal $\mathbf{u}$ defined in (8.15), (ii) if the model proposed can filter the raw data, (iii) the sensitivity of the proposed method to the choice of the parameters H representing the complexity of the learning model used and h representing the information. Concerning the parameter s we fix it equal to 1.
In the following we indicate by $\mathbf{u}^*$ the solution constructed with the LSTM-Full algorithm. All of them represent a 730 days solution. In the first column of Table 8.1 we report the results of

MSE($\mathbf{u}^0, \mathbf{u}^*$), over different choices of the sliding window parameter h. The choice of the sliding window parameter is a crucial point and in practice it should be a trade off between the training time and the prediction performance. Here we suggest, when possible, to choose h as the lag with the highest Autocorrelation Function (ACF) value of raw data (see Figure 8.3c); this seems necessary if we want to compute a less scattered solution.

In Figure 8.3b (red-dot line) we depict the solution for $h = 365$, since that the time series considered has an annual seasonality. In the second column of Table 8.1 we report the STD and we observe that the scattering is reduced while the sliding window dimension increases. The best improvements in terms of MSE (48% less than raw data) is reached choosing $h = 183$ whereas the best improvements in denoising is reached choosing $h = 365$ (49% less than raw data). As we expect the computational training time increases when the input size, that is h, increases. In Figure 8.3d we report the ACF plot of the differences between the theoretical time steps and the predicted time steps. The plot highlights that there is no relevant correlation within prediction errors, as a good prediction is expected to be. All the previous results are computed setting $H = 30$.

h	MSE($\mathbf{u}^0$,$\mathbf{u}^*$) (mm)	STD($\mathbf{u}^0$,$\mathbf{u}^*$) (mm)	time (s)
7	1.276	0.718	7.862
14	1.212	0.687	9.953
30	1.245	0.693	15.115
183	1.198	0.690	59.482
365	1.206	0.678	105.21

Table 8.1: All the results are computed setting $H = 30$. Columns 1 and 2 show respectively the MSE($\mathbf{u}^0$,$\mathbf{u}^*$) and STD($\mathbf{u}^0$,$\mathbf{u}^*$) values, where $\mathbf{u}^0$ is the theoretical signal and $\mathbf{u}^*$ is the predicted signal, for different length of the input, both in millimeters. The raw time series has a MSE equal to 2.327 millimeters and a STD equal to 1.329 millimeters. Column 3 shows the computational training time, in seconds, choosing different input size.

In Table 8.2 we report the summary of the analysis carried out choosing different values for the parameter H and setting $s = 365$. As it emerges from the first and the second column of Table 8.2 the best results in terms of MSE and of STD are obtained setting $H = 5$ (50 % less for MSE and 51 % less for STD respect to raw data). In the third column we report the computational training time that again increases while the parameter H increases.

H	MSE($\mathbf{u}^0$,$\mathbf{u}^*$) (mm)	STD($\mathbf{u}^0$,$\mathbf{u}^*$) (mm)	time (s)
5	1.169	0.646	109.174
30	1.205	0.678	115.049
100	1.309	0.727	224.441
300	1.262	0.708	2801.995

Table 8.2: All the results are computed setting $s = 365$. Columns 1 and 2 show respectively the MSE($\mathbf{u}^0$,$\mathbf{u}^*$) and STD($\mathbf{u}^0$,$\mathbf{u}^*$) values, where $\mathbf{u}^0$ is the theoretical signal and $\mathbf{u}^*$ is the predicted signal, for different length of H. The raw time series has a MSE equal to 2.327 millimeters and a STD equal to 1.329 millimeters. Column 3 shows the computational training time, in seconds, choosing different values of H.

8.2.2 A real static time series: an offshore gas platform

The first real time series, hereafter called "static" due to the GNSS data processing used, is 11 years long and each time step represents a daily solution. It is acquired by a permanent station on an off-shore platform, which is maneged by Eni S.p.A., an Italian multinational oil and gas company. It has three different components: North, East and Up components. The North component is depicted in Figure 8.4a using green dots. It is for sure the most accurate time series obtainable by GNSS technology because each solution is the mean of the observations recorded with a sampling time of 30 seconds during 24 hours. Since these kind of time series are generally used for tectonic plate motion or landslide monitoring it is important to have accurate and no noise solutions. The analysis that we conduct refers to those applications where the goal is the characterization of the periodical signals and in this case a signal denoising can improve the results.

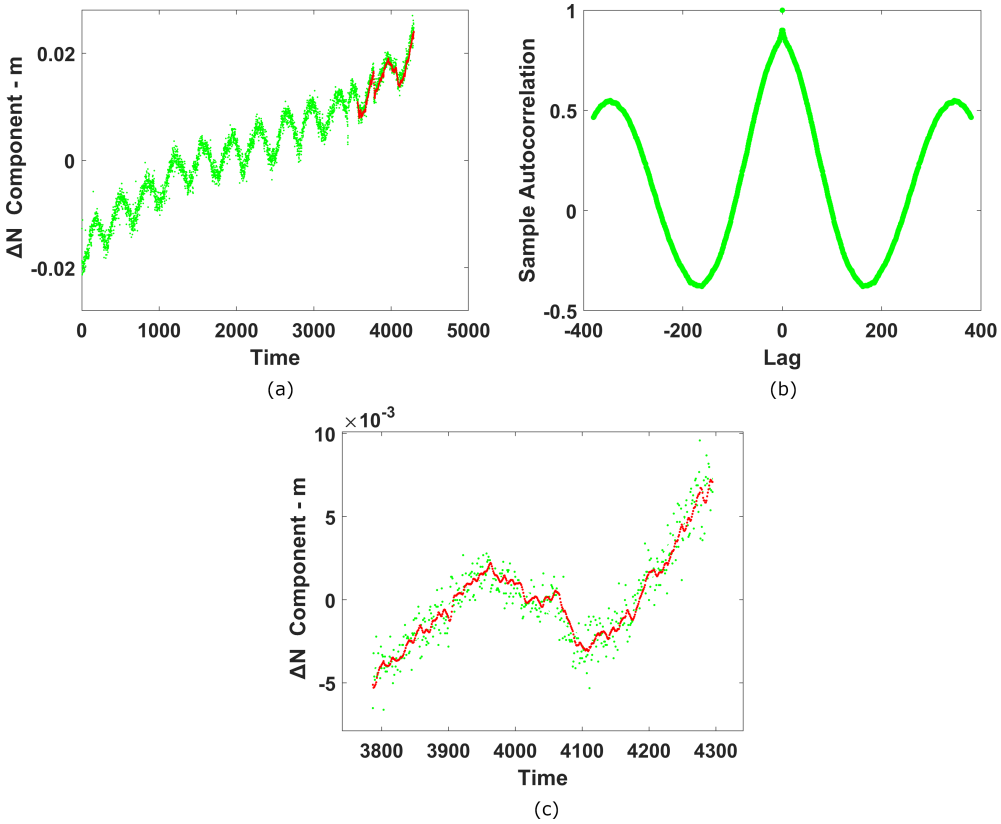


Figure 8.4: (a) Plot of the raw North Component of the Static Time Series (green dots) and the filtered time series (red dots). (b) ACF over 400 lag of the raw time series. (c) Plot of 509 filtered time steps (red dots), computed choosing s=365, compared with the raw time steps (green dots).

Static time series analysis

So far we have proved the effectiveness of the proposed LSTM-Full method on a test problem, we now want to evaluate our model on the real time series described in Section 8.2.2. Since the

ultimate goal for this time series is to remove the noisy components, we set the hidden vectors length parameter H equal to 5. In the following we indicate with $\mathbf{u}$ and $\mathbf{u}_h^*$ the raw time steps and the filtered solution respect to the sliding window parameter h, with p_0 and p_h the 5th-degree regression polynomials respect to $\mathbf{u}$ and $\mathbf{u}_h^*$. In order to evaluate the scattering of raw data and of $\mathbf{u}_h^*$, we consider the value $\text{STD}(p_h - \mathbf{u}_h^*)$. In the first column of Table 8.3 we compute the STD values of the filtered time series over different choices of the sliding window parameter s. The best result is achieved using h equal to 365, which is the lag with the highest autocorrelation function value, shown in Figure 8.4b. In Figure 8.4c we plot the filtered time series values obtained using $h = 365$ compared with the raw North Component of the static time series. In the second column of Table 8.3 we report the computational training time, which increases while h increases, as it is for the synthetic case. In Table 8.4 we report the STD values for the North Component, East Component and Up Component of our real static time series, using a sliding window parameter equal to 365.

s	$\text{STD}(p_s - \mathbf{u}_h^*)$ (mm)	time (s)
7	0.563	8.026
14	0.481	10.153
30	0.432	15.348
183	0.432	66.314
365	0.423	183.304

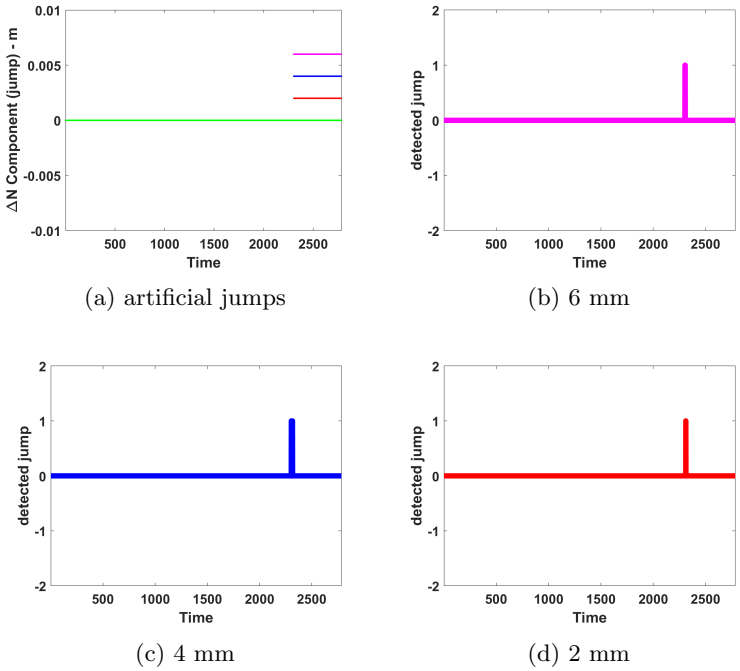
Table 8.3: Column 1 shows the STD values for the North Component, in millimeters, for different size of the sliding window parameter. The raw time series have a STD value equal to 0.975 millimeters. Column 2 shows the computational training time in seconds choosing different input size.

Component	$\text{STD}(p_s - \mathbf{u}_h^*)$ (mm)	$\text{STD}(p_0 - \mathbf{u})$ (mm)
N	0.423	0.975
E	0.559	1.031
U	1.221	2.808

Table 8.4: In the first column: STD values in millimeters for the North, East and Up component of the real static time series, using a sliding window parameter equal to 365. In the second column: the STD values of the raw signals for each component.

We consider the original offshore time series and we add artificial jumps of different magnitudes. In Figure 8.5a we report the differences plot between the raw time series and the one obtained by adding the jump, namely red line for the jump of magnitude 2 mm, blue line for the jump of magnitude 4 mm and purple line for the jump of magnitude 6 mm. In Figure 8.5b, 8.5c, 8.5d we report the plots of the differences between the predicted signal with LSTM-Full and the measured one. We point out that the hyperparameters for LSTM-Full have been set such that $H = 50$,

$h = 365$ and $s = 5$. Therefore, the model predict the time series values of the following 5 days with input vectors of length 365. We can observe that the LSTM-Full model is capable to detect the sole time steps with jumps.



(a) artificial jumps (b) 6 mm

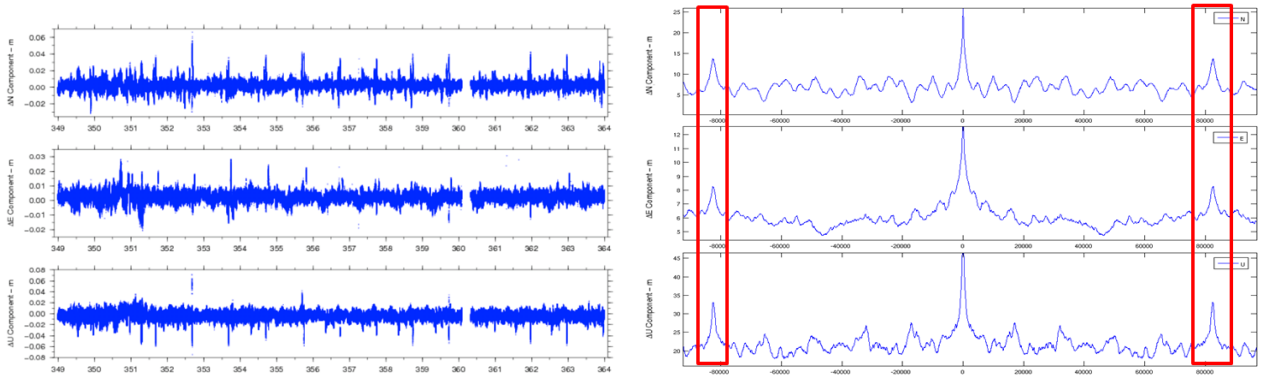
(c) 4 mm (d) 2 mm

Figure 8.5: Jump detection for the static time series. (a) Differences between the original time series and the one with artificial jumps. (b)-(c)-(d) differences between LSTM-Full outcomes and measured data with jumps.

8.2.3 A real kinematic time series: the Garisenda tower

The second real GNSS time series, hereafter called "kinematic", derives from higher frequency (1Hz) coordinates solutions, that were estimated using a kinematic approach during the data processing. The time span considered for the test is about 30 days and every sample represents a solution per second. As for the static time series, our data set is made up of North, East and Up components. The GNSS station is located on the top of the Garisenda tower of Bologna (Italy) which is one of the most important features of Bologna's cultural heritage, but it is notoriously affected by problems of stability. The adjacent Asinelli tower (50 meters taller than the Garisenda) constitutes an example of an unavoidable obstacle to satellite signals, which may affect GNSS solutions and have to be considered a habitual problem in these applications. This is an example of the so-called GPS multipath phenomena which is caused by the reception of signals arrived not only directly from satellites, but also reflected or diffracted from the local objects. At the beginning of the year 2011, a monitoring system was installed on the Garisenda tower in order to monitor its structural behavior. In 2013, the Department of Civil, Environmental and Materials Engineering of Bologna University installed a permanent GNSS station on the roof of the Garisenda for the double purpose of monitoring the building and testing the satellite technology for this type of application. The

interested reader can refer to [195] for further details. In Figure 8.6a we depict the components of this time series. This is a completely different scenario respect to the previous one because we have a solution every second and for this reason, the redundancy of the system and the accuracy are lower. This approach can be suitable for early warning applications where it is much more important to have as soon as possible information about unexpected changes respect to the normal movement. Since this kinematic time series is used for structure monitoring such as bridge monitoring or critical landslides where a fast movement of the object can be critical for the safety of life, here the ultimate goal of our analysis is to develop a method capable to give an accurate prediction of the incoming coordinates to improve the capability to detect some anomalies.



(a) Garisenda time series components

(b) ACF plots

Figure 8.6: (a) Plot of the raw Nord, East and Up Components of the Garisenda Time Series (blue dots). (b) ACF of the raw time series. The images are taken from [195].

	N	E	U
MM	3.3	2.2	5.1
LSTM-Full (s=1)	2.0	1.1	2.9
LSTM-Full (s=5)	2.6	1.8	4.4
Red. MM	42%	23%	27%
Red. LSTM-Full (s=1)	65%	62%	58%
Red. LSTM-Full (s=5)	38%	20%	25%

Table 8.5: Upper panel: STD($\tilde{\mathbf{x}} - \tilde{\mathbf{y}}$) values in millimeters for the N,E,U components of the Garisenda time series, in millimeters, for the MM and LSTM-Full with $s = 1$ and $s = 5$ approaches. Lower panel: percentage of reduction when comparing STD($\tilde{\mathbf{x}} - \tilde{\mathbf{y}}$) with STD($\tilde{x}$).

Kinematic time series analysis

In the case of the kinematic time series described in Section 8.2.3, the aim of the LSTM-Full model is to give an accurate prediction for the incoming time steps, since we want to apply this model for structural monitoring. Since we are not seeking for a filtered solution, we fix the dimension of the LSTM hidden vectors at 100. The lag with the highest value of the ACF is 86164 (the number

of seconds in a sideral day), shown Figure 8.6b, and it is impossible to use it as sliding window parameter, due to the computational cost of the algorithm. Therefore, in this case, we heuristically set $h = 300$. In order to evaluate the efficiency of the proposed method we use the STD value of the difference between the raw data $\tilde{\mathbf{x}}$ and the predicted time steps $\tilde{\mathbf{y}}$. In upper panel of Table 8.5 we report the STD values, in millimeters, by comparing our learning approach setting $s = 1$ and $s = 5$ and the method proposed in [195] using sequential filtering. Moreover, in the lower panel of Table 8.5 we report the percentage of reduction of STD($\tilde{\mathbf{x}} - \tilde{\mathbf{y}}$) with respect to STD($\tilde{\mathbf{x}}$) for all the three components. The latter values for the N,E and U components are 5.7 mm, 2.9 mm and 6.9 mm, respectively. As a general comment, in both cases the LSTM-Full approach outperforms the MM method, however we observe that its accuracy starts decreasing when s increases.

Conclusions

The common thread among the chapters of this thesis was to present efficient algorithmic solutions to solve image restoration and system identification tasks by using variational, learning and hybrid paradigms.

In Chapter 3 we have proposed ASR2, a flexible algorithm for SISR and MISR applied to thermal images. The proposed method is based on a ℓ_2-TV regularized approach and includes an adaptive and low-cost computation of the regularization parameter, thus resulting fully automatic and computationally efficient. A detailed analysis has been conducted, comparing the proposed algorithm with the EDSR [63], one of the most performing Deep Neural Networks for SR, and RISR [62], one of the most recent algorithms designed for thermal image SR. The experimental results prove the strength of the proposed approach in terms of visual evaluation, e.g. PSNR and SSIM. A careful radiometric analysis confirms the reliable reconstructions of sharp details and isolated hot or cold spots, which are essential in thermal imagery and many remote sensing applications. This proposal gains at least 60% accuracy in reconstructing temperature peaks compared to the EDSR and RISR, and removes at least 45% of noise. The flexibility of the approach makes it particularly suitable for both aerial and terrestrial thermal remote sensing, especially for applications involving the detection and delineation of hot spots or temperature anomalies at a finer spatial scale.

In Chapter 4 we proposed constrained and unconstrained variational models, for joint single-image super-resolution and image partitioning, based on the use of an ℓ_0-type jump penalizer, combined with ℓ_2 data fidelity, for favoring sharp gradient smoothing. The use of non-convex jump-sparse regularizations has been considered in [67, 68, 69]. To overcome the computational limitations required by the use of ADMM splitting strategies considered in these works, we have implemented a novel ADMM algorithm allowing for the efficient numerical solution of the models by means of direct hard-thresholding or standard CG solvers or, upon a specific assumption on the down-sampling operator, by diagonalization in the Fourier domain. Our implementations are 15-times faster when the CG solver is used, 700-times faster when using FFT, than the approach in [67].
The methods are validated on synthetic data and tested on real-world examples where gradient-sparse super-resolved outputs are required in view of a subsequent accurate detection/segmentation step (such as QR code recognition and cell detection). By numerous comparisons with convex and non-convex variational approaches, and with state-of-the-art deep learning methods [85, 84], we show that the proposed approaches significantly improves classification precision, while limiting at the same time smoothing and loss-of-contrast artifacts in comparison with classical convex regular-

izations. Further work should address the use of analogous algorithms for the joint modelling of SR and segmentation problems via, e.g., Mumford-Shah functionals [68]. Furthermore, the extension of the convergence results to other gradient discretisations and to less restrictive growth conditions for the sequence of penalty parameters is envisaged.

In Chapter 5 we have proposed a new PnP method using learnt gradient-based priors applied to CT medical image restoration. We considered a Half-Quadratic Splitting minimization algorithms where the denoising step is executed by a CNN acting on the image gradients (GCNN method). We also considered a hybrid regularization where we added a Total Variation functional in the GCNN scheme (GCNN-TV). The numerical experiments on synthetic and real CT medical images show that the proposed GCNN well recovers the curve contours of flat and low-contrast objects, as well as thin vessels. The GCNN-TV further smooths homogeneous area such as backgrounds and small low-contrast objects on very noisy images and its restoration appears suitable for segmentation. The obtained image enhancements confirm that the proposed PnP gradient-based regularization is effective for the restoration of medical CT images, since the competitors, namely other PnP algorithms considering different denoisers defined on the image-domain, get lower quality indices.

In Chapter 6 we presented a deep learning-based method called DeepCEL0 for precise single molecule localization in high density fluorescence microscopy settings. The proposed method brings together the benefits of two well-known standard methods in the field, i.e., DeepSTORM [126] and CEL0 [127], introducing a network architecture with two main novelties: a continuous ℓ_0-penalized training loss function and the adoption of non-negativity constraints on the solution through a ReLu layer. Compared to the standard methods, numerical results show the stability of DeepCEL0 at varying the level of corruption and its ability to provide very high precision localization maps, without detriment to computational cost. Moreover, the method is parameter-free and can be easily tested and applied on real data after a training phase on only synthetic data. The promising results make the methods easy to perform in disparate real applications exploiting fluorescence microscopy.

In the first part of Chapter 7 we proposed a constrained and an unconstrained DIP optimization models which automatically estimate the strength of the regularization. The unconstrained one uses a space variant handcrafted regularizer whose local regularization parameters are adaptively defined along the optimization process, whereas the constrained model is tailored for a generic regularizer and implicitly forces solutions satisfying the discrepancy principle. Particularly, we used the space variant Total Variation and the RED regularizer in the implementation for the unconstrained and the constrained models, respectively. The main strengths of the developed frameworks are threefold: it is not required to set proper values for the regularization parameter, the schemes implemented are more robust with respect to the selection of the hyperparameters than other state-of-the-art DIP-based methods [39, 160], and both schemes avoid the typical overfitting behaviour of the DIP framework. The numerical experiments on image denoising and deblurring show better results of the developed approaches with respect to state-of-the-art strategies with the great advantage of avoiding costly parameter tuning.

In the second part of Chapter 7 we presented a novel approach called Recursive Deep Prior Video to overcome the limitation of low resolution in time-lapse microscopy scenario for organ-on-chip applications, within the so-called super-resolution context. The main novelties of the approach refer to the recursive initialization of the weights of the DIP network architecture combined with an efficient early stopping criterion. In addition, the DIP loss function has been penalized by Total Variation-based terms. The method has been validated on synthetic, i.e., artificially generated, as well as real videos from OOC experiments related to tumor-immune interaction and compared to the most effective state-of-the-art approaches in the context of trained methods. The proposed approach demonstrated to be feasible to real-time applications due to the unsupervised architecture, robust to noise thanks to the regularization terms, and able to effectively work in combination with state-of-the-art edge localization and edge detection methods for the task of object recognition and biological experiment characterization. Future works will address the problem of improving the effectiveness of the approach in terms of parallelization and of implementation in the routine use of microfluidic devices to accelerate the uptake of OOC experiments.

In Chapter 8 we considered the LSTM-Full Network architecture which has been applied for the first time to address the denoising, prediction and jump-detection tasks of GNSS time series. The performances depend on the choice of the sliding window parameter and of the hidden vectors size of the LSTM layer. The experiments suggest to choose the sliding window parameter as the lag with the highest value of the autocorrelation function, when the computational resources allow it. The hidden vectors size should be choosen small if the task is denoising, otherwise, it should be higher to give an accurate prediction on incoming data, so as to better detect jumps. The method provide accurate prediction, if compared to [195], and is capable to detect discontinuities even when dealing with high noise data.

Appendix A

Deep Learning

In this appendix we give a short survey on Deep Learning, a branch of computer science, which develops algorithms that learn from the data. After a concise introduction to Machine Learning, we introduce the neural networks. We begin explaining the perceptron and subsequently we introduce DNNs. Next, we illustrate some algorithms commonly used to train DNNs, namely SGD and Adam. In the last section we introduce the CNNs, very suitable models for image related tasks. For a complete overview the interested reader can refer to [206].

Machine learning basics

Machine learning can be considered as a sort of applied statistics, because its main purpose is to statistically estimate functions which are complicated to evaluate with common programming paradigms. In particular, machine learning algorithms are able to learn from data or experience. The computer scientist Tom M. Mitchell in [207] gave a concise and widely accepted definition of learning algorithm, that we report here.

Definition A.1. A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.

To create a machine learning algorithm, we could consider a variety of experiences E, measures of performance P or tasks T. The functioning of this kind of machine learning algorithms consists of two parts:

- **Learning stage**: during this step, a set of examples is provided to the algorithm that develops its own logic to solve the task.
- **Inference stage**: in this step the algorithm performs the task on a new input not previously given in the set of examples.

The possible tasks T range from the easier one, like classification and regression, to the more challenging like transcription, translation and synthesis.

The performance measure P is closely related to the type of task and the type of experience used in the algorithm. In some cases we measure the accuracy of the model, that is the percentage of examples for which the model perform the task in the proper manner. The choice of P is often difficult and influences the learning capabilities of the method. In practice, we are also interested to determine how well a machine learning algorithm will work in the real world. Hence, the performance is measured on a test set of data, not previously seen during the learning phase.

Dependently on the form of experience E we identify three category of learning based algorithms:

- **Supervised learning**: each example in the dataset is linked with a target or label which is the result of the task. Many deep learning algorithms can be included in this class.
- **Unsupervised learning**: the input data is provided without the corresponding output label. Usually the goal of this kind of algorithms is to learn implicitly/explicitly the probability distribution that generated the dataset. This approach is useful for tasks like denoising, synthesis or density estimation.
- **Reinforcement learning**: the input data is not a fixed dataset. The algorithm is free to perform a task, only at the end it receives a feedback. If the action is wrong the model receives a negative feedback and, for this reason, is forced to change its strategy. Otherwise, if the action is correct the model receives a positive reward and it keeps on using its strategy.

For a comprehensive perspective on the fundamentals of machine learning the reader can refer to [208, 209].

A.1 Neural networks

Artificial neural networks are a class of learning algorithms, characterized by a computing system inspired by biological neural networks that form the brain. The key feature of these systems is the connection between a collection of computational nodes or units, commonly called neurons because they mimic biological neurons. Each connection transmit a signal or an information from a neuron to the following one, as synapses do. Formally a neural network is a directed and weighted graph: each neuron is a node and each connection is an edge. These networks learn from the data through a training process, as explained in the previous section. During the learning stage, the network adjusts its parameters, the weights of the edges, called synaptic weights.

Perceptron

The first neural network was proposed by the American psychologist Frank Rosenblatt in 1958, combining the biological findings of Donald Hebb and the idea of McCulloch-Pitts' neuron. This architecture, represented in Figure A.1, took the name of Rosenblatt Perceptron. Formally we provide this mathematical definition:

Definition A.2. Given an input $\mathbf{x} \in \mathbb{R}^D$, a weights vector $\mathbf{w} \in \mathbb{R}^D$, a bias (or threshold) $b \in \mathbb{R}$ and an activation function $\phi : \mathbb{R} \to \{0, 1\}$, the Perceptron is a function $f : \mathbb{R}^D \to \{0, 1\}$ defined as

$$f(\mathbf{x}) = \phi(\mathbf{w} \cdot \mathbf{x} + b) = \phi\left(\sum_{i=1}^{D} \mathbf{w}_i \mathbf{x}_i + b\right).$$

In his work, Rosenblatt considered the Heaviside step function as activation and showed the Perceptron model learns binary classifiers of separable data. Furthermore the Perceptron, with a suitable choice of weights and threshold, was proved to represent the logical operations AND, OR and NOT. However, as stated by the cognitive scientist Marvin Minsky and the matematician Seymour Papert in [210], this simple architecture has numerous drawbacks:

- a Perceptron is not able to determine if the number of active inputs is odd or even.
- a Perceptron is incapable to learn the XOR operation (exclusive disjunction).
- a Perceptron can approximate only linearly separable functions.

The interest in Neural Networks had steadily decreased due to the criticisms highlighted by Minsky and Papert, researchers had been overly optimistic and without results their funding was cut. This period is called First AI winter.

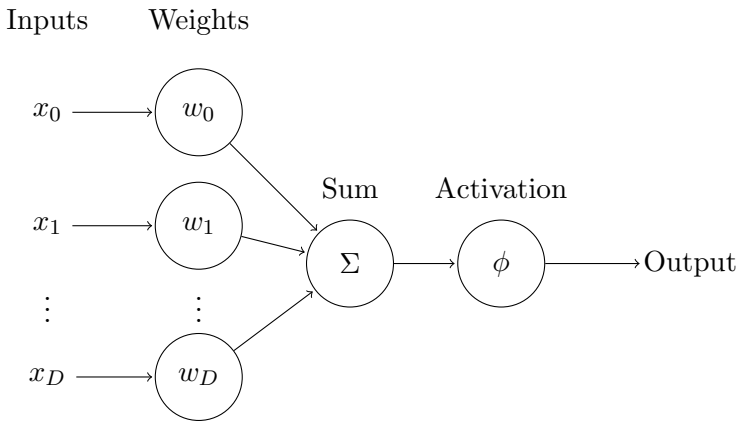


Figure A.1: Perceptron architecture

Deep neural networks

Machine Learning experts thought that a multilayer perceptron could have overcome the problems of perceptron. However, they were limited by the technology available in that period. An implementation of their ideas would have been too expensive. Only in the eighties, a series of innovations in the computer science field, led to Multilayer Feedforward Networks. The architecture of this kind of networks can be represented as a directed weighted acyclic graph. This means that there are no loops inside the network and the data is processed trough a sequence of neuron layers connected in series. There are no connections that bridge the output of the model into itself and for this reason

they are also called feedforward networks. A representative example of these architectures is the multilayer Perceptron (MLP). This kind of networks can either perform regression or classification tasks, depending on the choice of the activation function ϕ. In Figure A.2 we depict an example of MLP architecture.

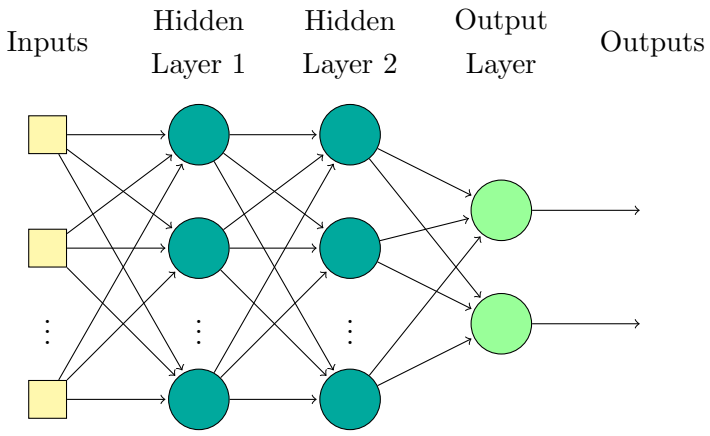


Figure A.2: Multilayer Perceptron architecture

Usually, the architecture of a deep neural network follows this structure:

- The input layer is made up of N units, where N is the dimension of the features vector. These input units do not perform any computation on the data, they only pass the information to the following layers.
- A series of L layers of neurons called hidden layers, where $L \geq 2$. The last layer, called output layer, is composed of K neurons, where $K \geq 1$.
- A group of synaptic weights that represent the strength of the connections between neurons of successive layers.

In order to illustrate more clearly how a deep neural network works, we introduce some notations. At first let us consider the input $\mathbf{x} \in \mathbb{R}^N$. Then let us denote the number of layers by L and the number of neurons by N_L. Obviously N_0 corresponds to the size of the input layer and N_L to the size of the output layer. Hence $N_0 = N$ and $N_L = K$. We organize the synaptic weights in weight matrices $\boldsymbol{W}^{(i)} \in \mathbb{R}^{N_i \times N_{i-1}}$ where an element $\boldsymbol{W}^{(i)}_{sr}$ encodes the weight between the r-th neuron in the $(i-1)$-th layer and the s-th neuron in the i-th layer. Thus each matrix $\boldsymbol{W}^{(i)}$ for $i = 1, \ldots, L$ contains the weights of the edges which connect the $(i-1)$-th layer to the i-th. In addition we denote the bias vectors $\boldsymbol{b}^{(i)} \in \mathbb{R}^{N_i}$ for $i = 1, \ldots, L$ and the activation function of the s-th neuron in the i-th layer as $g^{(i)}_s : \mathbb{R} \to \mathbb{R}$. In a more compact form, we define the activation function of the i-th layer as $\boldsymbol{f}^{(i)} : \mathbb{R}^{N_i} \to \mathbb{R}^{N_i}$ where $\boldsymbol{f}^{(i)}(\mathbf{a}) = (g^{(i)}_1(\mathbf{a}_1), \ldots, g^{(i)}_{N_i}(\mathbf{a}_{N_i}))$.
From a computational point of view, a deep neural network is a function $\boldsymbol{N} : \mathbb{R}^N \to \mathbb{R}^K$ whose output is computed using Algorithm 9. Commonly, the dimension of every layer, the activation functions of each neuron and the depth of the network are fixed and are called hyperparameters. This class of parameters is chosen before the learning stage following heuristic strategies. Thus,

Algorithm 9 – Feed Forward Neural Network $\boldsymbol{N}(\mathbf{x})$

input: $\mathbf{x}$, L, N_i, $\boldsymbol{W}^{(i)}$, $\boldsymbol{b}^{(i)}$, $\boldsymbol{f}^{(i)}$.
output: $\boldsymbol{N}(\mathbf{x})$.
  $\mathbf{h}^{(0)} = x$
  for $i = 1, \dots, L$ **do**
    $\mathbf{a}^{(i)} = \mathbf{b}^{(i)} + \boldsymbol{W}^{(i)} h^{(i-1)}$
    $\mathbf{h}^{(i)} = \boldsymbol{f}^{(i)}(\mathbf{a}^{(i)})$
  end for
  return $\mathbf{h}^{(L)} \to \boldsymbol{N}(\mathbf{x})$

during the actual training process, a Feed Forward Neural Network learns only the weight matrices and the bias vectors.

The key feature of each neuron is its activation function. Referring to the biological neural networks, an activation function is a formalization of the action potential firing in a neuron, triggered by an input signal. Mathematically, these functions regulate how the signal/information propagates through the network. As previously said, the activation functions have to be fixed. Frequently their choice is motivated by learning algorithms that we will describe later. Some functions prevent few problems during the learning phase. For this reason, we first define some activation layers commonly used.

Definition A.3 (Sigmoid)**.**

$$\phi(x) = \sigma(x) = \frac{1}{1 + e^{-x}}. \tag{A.1}$$

Definition A.4 (Hyperbolic Tangent)**.**

$$\phi(x) = \tanh(x) = \frac{(e^{x} - e^{-x})}{(e^{x} + e^{-x})}. \tag{A.2}$$

Definition A.5 (ReLu)**.**

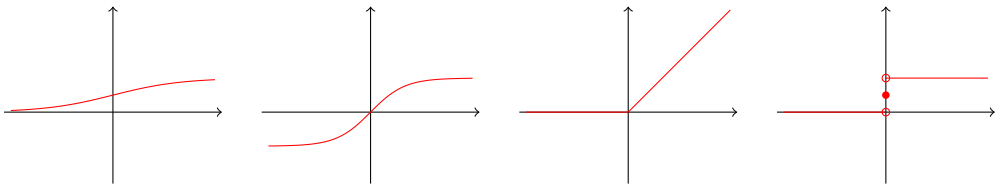
$$\phi(x) = ReLu(x) = \max(0, x) = \begin{cases} x & \text{for } x > 0 \\ 0 & \text{for } x < 0. \end{cases} \tag{A.3}$$

Definition A.6 (Heaviside step function)**.**

$$\phi(x) = H(x) = \begin{cases} 1 & \text{for } x > 0 \\ \frac{1}{2} & \text{for } x = 0 \\ 0 & \text{for } x < 0. \end{cases} \tag{A.4}$$

A.2 Training deep neural networks

A feed forward neural network can be considered as function depending on a set of weights and biases which are denoted by $\theta = \{(\boldsymbol{W}^{(i)}, \boldsymbol{b}^{(i)}) \text{ for } i = 1, \dots, L\}$, in the following. Thus, formally



(a) Sigmoid (b) tanh (c) ReLu (d) Heaviside

Figure A.3: Plots of the activation functions

a DNN is defined as a function $\boldsymbol{N}(x;\theta)$ parameterized by θ. To improve the behavior of the algorithm a performance measure P is considered. Unfortunately, in some cases P is intractable, therefore, we prefer to optimize the algorithm indirectly. More precisely, we shift the learning problem to the optimization of a cost function $J(\theta)$ which is linked to P. Hence, the optimization in learning problems differs from pure optimization: in the latter the minimization of a functional is the problem itself, in the former one, the initial learning problem is reduced to a straightforward optimization problem.
Commonly, the cost function is written as an average over dataset:

$$J(\theta) = \mathbb{E}_{\hat{p}}\left[L(\boldsymbol{N}(x;\theta), y)\right], \tag{A.5}$$

where L is a loss function, $\boldsymbol{N}(x;\theta)$ is the predicted output when x is the input, y the target output and $\hat{p}$ is the empirical distribution of the data. Despite its simple form, we only have a training set $\mathcal{M} = \{(x_i, y_i)\}_{i=1}^n$ and $\hat{p}$ is not given, thus we have to consider an estimation of $J(\theta)$. A common choice is the empirical risk defined as follows:

$$\bar{J}(\theta) = \frac{1}{n}\sum_{i=1}^{n} L(\boldsymbol{N}(x_i;\theta), y_i). \tag{A.6}$$

A plenty of optimization algorithms using first and second order derivatives could be used to minimize the functional $\bar{J}(\theta)$ in (A.6). A well-known and representative algorithm is *gradient descent*, which optimizes a function following the opposite direction of its gradient. This strategy is also known as *steepest descent*. See Algorithm 10 for a formal description. Obviously, gradient descent can be used only when the cost function to be minimized is differentiable.
Figure A.4 represents an example of several iterations performed by the gradient descent algorithm to find the minimum on an elliptic paraboloid. For the sake of clarity, we only depict the level sets of the paraboloid. For a complete overview on numerical optimization methods refer to [31].

Backpropagation

An analytical expression for the gradient can be computed in a straightforward manner using the chain rule. However, the evaluation of this type of expression is usually computationally expensive. The back-propagation algorithm, also called *backprop*, allows us to calculate the partial derivatives in an amount of time linearly proportional to the depth of the computational graph related to the neural network. A computational graph is a directed graph where each node represents a variable

Algorithm 10 – Gradient Descent algorithm for the function $\bar{J}(\theta)$

input: $\bar{J}$, θ_0, $\alpha \in \mathbb{R}^+$, $\theta_{old} = \theta_0$.
output: θ^*.

$\theta_{old} = \theta_0$
repeat
  $\theta_{new} = \theta_{old} - \alpha \nabla \bar{J}(\theta_{old})$
  $\theta_{old} = \theta_{new}$
until the stopping test is satisfied
$\theta^* = \theta_{old}$
return θ^*

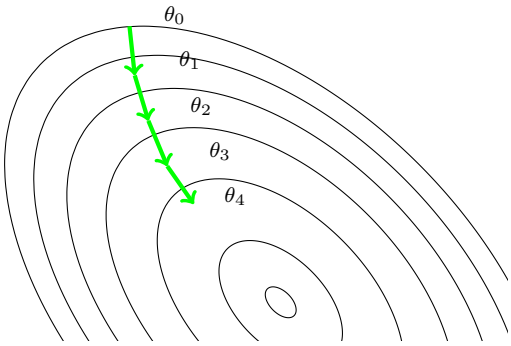


Figure A.4: Illustration of gradient descent on a series of level sets

or a function. Hence, functions can feed their output into other functions and variable can feed their value into functions. We can associate each network with the computational graph that represents $\bar{J}$ and use backprop to compute $\nabla \bar{J}$. This algorithm is often misunderstood as being a learning/optimization method. Instead, it is a clever algorithm very suitable for the computation of derivatives in a computational graph. Its application in a learning framework was initially proposed in [211]. The application independent version of this algorithm is also called reverse-mode differentiation.
For the sake of clarity, we present the backpropagation method only for the single layer perceptron in Figure A.1. The generalization to multi-layer networks takes advantage of the fact that complex architectures can be considered as compositions of simpler ones. The interested reader can refer to to [212] for a comprehensive description of the backpropagation algorithm. We now compute a partial derivative of only one term in $\bar{J}$, which is the sum over all the examples in the training set. By using the chain rule we have:

$$\frac{\partial L(\boldsymbol{N}(x;\theta),y)}{\partial w_i} = \frac{\partial L(\boldsymbol{N}(x;\theta),y)}{\partial \phi}\frac{\partial \phi}{\partial w_i} = \frac{\partial L(\boldsymbol{N}(x;\theta),y)}{\partial \phi}\frac{\partial \phi}{\partial \Sigma}\frac{\partial \Sigma}{\partial w_i}. \tag{A.7}$$

This equation provides us a decomposition for all the partial derivatives. From a computational point of view it is cheaper to save the first two factors in the decomposition. We observe the only term that changes, with respect to the choice of the index i, is the last one, which can be easily

computed. In formula:

$$\frac{\partial \Sigma}{\partial w_i} = \frac{\partial (w \cdot x)}{\partial w_i} = \frac{\partial \left(\sum_{j=1}^{D} w_j x_j \right)}{\partial w_i} = x_i. \tag{A.8}$$

The choice of the activation function ϕ influences the second term of the decomposition. For this reason, we restrict ourselves to a small set of activation functions (depicted in Figure A.3) whose derivatives has convenient properties. The equation (A.7) shows that only local derivatives are necessary, hence the nodes in the computational graph can be considered independently. Therefore, for example, Tensorflow builds a computational graph with extra nodes that compute the derivatives of the pre-existing nodes. Whereas, PyTorch implicitly does the same computations without exposing the computational graph.
Whatever the architecture of the net is, the backpropagation algorithm can be divided into two main steps:

1. **Feedforward step**: the examples in the training set are given to the computational graph which computes and store the outputs of the network and their related derivatives.

2. **Backprop step**: the computational graph is visited in the opposite direction to stack the local derivatives in order to obtain all the partial derivatives.

Stochastic Gradient Descent (SGD)

Usually the cost function $\bar{J}$ is the sum of a loss function over the training examples. For this reason, as the size of dataset increases, the computation of $\nabla \bar{J}$ becomes too expensive. Stochastic gradient descent is a modified version of steepest descent, that cuts down the computational time required to obtain the gradients. Instead of the exact gradient, SGD uses an approximation. Initially, given an integer $m \ll n$, on each iteration of the algorithm a sample of m examples is drawn from the initial dataset. This subset is called minibatch and is defined as $\mathcal{B} = \{(x_{s(i)}, y_{s(i)})\}_{i=1}^{m}$ where $s(i)$ is an index drawn uniformly from the set of indices. The size of minibatch is held fixed for all the iterations. Depending on the minibatch, we subsequently use an approximation of the gradient defined as:

$$g_{\mathcal{B}}(\theta) = \frac{1}{m} \sum_{i=1}^{m} \nabla L(\boldsymbol{N}(x_{s(i)}; \theta), y_{s(i)}). \tag{A.9}$$

A formal description of SGD is reported in Algorithm 11. The strategy of SGD has less theoretical guarantees than the gradient descent. However, SGD often obtains small values of the objective function in less computational time.
The momentum method is a modified version of SGD. It is physically inspired and it was proposed in order to accelerate learning in case of noisy gradients or losses with high curvature. A loss can be considered as a potential energy and the parameters initialization is equivalent to putting a particle in some location in the hyperspace. The optimization process is equivalent to simulating the movements of the particle due to the potential energy. Differently from SGD, in which we suppose that the particle has zero initial velocity, in momentum we give to the particle a random initial velocity and, at each iteration, we update the particle's velocity and its position. The presence of this additional term guarantees an improved convergence rate. For this reason the momentum

Algorithm 11 – Stochastic gradient Descent algorithm for the function $\bar{J}(\theta)$

input: $\bar{J}$, θ, $\alpha \in \mathbb{R}^+$, $m \in \mathbb{N} \cap [1, n]$, `stopping`.
output: θ.

repeat
  $\mathcal{B} \leftarrow$ sample a minibatch of size m from the training set
  $\theta = \theta - \alpha g_{\mathcal{B}}(\theta)$
until `stopping` is satisfied
return θ

algorithm is considered an accelerated variant of SGD. A formal description of the momentum method is reported in Algorithm 12.

Algorithm 12 – SGD with momentum update for the function $\bar{J}(\theta)$

input: $\bar{J}$, θ, $\alpha \in \mathbb{R}^+$, $\gamma \in \mathbb{R}^+$, $m \in \mathbb{N} \cap [1, n]$, v, `stopping`.
output: θ.

repeat
  $\mathcal{B} \leftarrow$ sample a minibatch of size m from the training set
  $v = \gamma v - \alpha g_{\mathcal{B}}(\theta)$
  $\theta = \theta + v$
until `stopping` is satisfied
return θ

Adaptive algorithms

The methods we previously mentioned use the same global learning rate at each iteration. In practice, the choice of a global learning rate is difficult, mainly because the magnitude of the gradients can change during the training. Motivated by these problems, researchers started developing a new class of adaptive methods. The most commons are RMSprop and Adam. For a complete review the interested reader can refer to [213].

- **RMSprop** at each iteration updates a variable v, commonly called running average of squared gradients, and uses it to compute the parameter update. In formulas:

$$v = \eta v + (1 - \eta) g_{\mathcal{B}}^2(\theta) \tag{A.10}$$

$$\theta = \theta - \frac{\alpha}{\sqrt{\delta + v}} g_{\mathcal{B}}(\theta), \tag{A.11}$$

  where η is the decay rate (usually set to 0.9), α the learning rate and δ is a small constant used to stabilize division by small numbers (usually set to $\approx 10^{-6}$).

- **Adam** in some sense, can be considered as an RMSprop with a momentum update. In addition to v, the running average of gradients is computed at each iteration and saved in

the variable m. In formulas:

$$m = \eta_1 m + (1 - \eta_1) g_{\mathcal{B}}(\theta) \tag{A.12}$$

$$v = \eta_2 v + (1 - \eta_2) g^2_{\mathcal{B}}(\theta), \tag{A.13}$$

where η_1 and η_2 are small numbers close to 1. In [166], the authors proposed to set $\eta_1 = 0.9$ and $\eta_2 = 0.999$. These two variables m and v are biased estimates of the first and the second moment. Before the update, a bias-corrected versions of m and v are considered:

$$\hat{m} = \frac{m}{1 - \eta_1^t}, \qquad \hat{v} = \frac{v}{1 - \eta_2^t}, \tag{A.14}$$

where t represents the number of iteration in which we are computing these quantities. Then $\hat{m}$ and $\hat{v}$ are used to update the parameters as follows:

$$\theta = \theta - \frac{\alpha}{\sqrt{\hat{v}} + \delta} \hat{m}. \tag{A.15}$$

Overfitting and underfitting

The functional $J(\theta)$ defined in (A.5), is the expectation on the training set. We would like to define a cost function which depends on the data-generating distribution $\hat{p}$, in order to have a neural network that generalizes its results also on unobserved data. However, in many application $\hat{p}$ is unknown and only a finite dataset of examples is available. Commonly, to study the generalization capabilities of a neural network, the dataset is splitted into two parts: the training set and the test set. Only the available examples in the training set are used in the learning phase. The data we do not use in the training stage, are then used in a testing phase to measure the performances of the network. Here, we define two problems that machine learning algorithms may encounter:

- **Underfitting** occurs when the model is unable to perform well on the training set.
- **Overfitting** occurs when the model is not capable to generalize to unseen data, namely, the difference between the performances on the training set and the test set is too large.

The tradeoff between these two issues can be found modifying the model's capacity. Roughly speaking, the capacity of a model is the model's ability to represent a wide sets of function, and in our case, the capacity corresponds to the number of hyper-parameters. The development of a new machine learning algorithm is time consuming. For this reason it is important to have a wide knowledge of machine learning systems, rather than blindly guessing. In some applications, there exist procedural algorithms that work much better than a black-box neural network. There are only a few heuristic rules which guide the design of an algorithm. Some researchers criticize the lack of rigorous criteria for choosing one architecture over another. In particular, in [214], the author compares the entire machine learning field to alchemy. In this thesis, we chose architectures with high performances in tasks similar to the one we want to address. Building new types of neural networks to solve image and time series related inverse problems is out of the scope of this thesis.

A.3 Convolutional networks

Convolutional neural networks, also known as CNNs, are a type neural networks specialized for processing data with a grid topology. In particular they are very suitable for computer vision tasks. The name indicates that in at least one layer the network uses a convolution in place of a matrix multiplication. Before explaining how a convolutional layer works, here we provide some useful definitions.

Definition A.7. Given two functions $k : \mathbb{R}^2 \to \mathbb{R}$ and $i : \mathbb{R}^2 \to \mathbb{R}$, the convolution of k and i is written as $k * i$ and it is defined as follows:

$$f(x, y) = k(x, y) * i(x, y) = \int_{\mathbb{R}^2} k(s, r) i(x - s, y - r)\, ds\, dr. \tag{A.16}$$

Definition A.8. Given two functions $K : \mathbb{Z}^2 \to \mathbb{R}$ and $I : \mathbb{Z}^2 \to \mathbb{R}$, the discrete convolution of K and I is written as $K * I$ and it is defined as follows:

$$F(x, y) = K(x, y) * I(x, y) = \sum_s \sum_r K(s, r) I(x - s, y - r). \tag{A.17}$$

Usually, following the convolutional network terminology:

- I is said input,
- K is called kernel,
- F is the output of the convolution operation and it is called feature map.

In practice, many deep learning libraries implement a sliding dot product or cross-correlation which works similarly to convolution. Developers use this strategy because convolution has the input flipped respect to the kernel. This means that as s or r increases, the index into the input decreases, but the index into the kernel increases.

Definition A.9. Given two functions $K : \mathbb{Z}^2 \to \mathbb{R}$ and $I : \mathbb{Z}^2 \to \mathbb{R}$, the cross-correlation of K and I is written as $K * I$ and it is defined as follows:

$$F(x, y) = I(x, y) \star K(x, y) = \sum_s \sum_r I(x + s, y + r) K(s, r). \tag{A.18}$$

When we work with images the domain is no more $\mathbb{Z}^2$ but the indexes are restricted to the image's pixels and the kernel's entries. Hence we consider $I \in \mathbb{R}^{h \times w}$ and $K \in \mathbb{R}^{k_1 \times k_2}$. Obviously, under these assumptions, some terms in (A.17) and (A.18) are not defined because their indexes are out of bounds. Hence we explain two methods to set the elements out of the boundary:

- **Valid padding**: the convolution kernel K is only allowed to visit positions where the kernel is entirely contained within the input I. Every pixel in the feature map is function of the same number of input's pixels. In this case, the height and width of the output are $h - k_1 + 1$ and $w - k_2 + 1$, respectively. An example is depicted on the left in Figure A.5 and in Figure A.6. The shrinkage of the output's dimension become a problem as the number of convolutional layers increases and is highly dependent to the kernel's size.

- **Same padding**: the input I is padded with zeros so that the feature map will have the same dimension of the input. In this case we can stack as much convolutional layer as we want. Hence, the architecture is no more limited by the kernel's size. However, this choice underrepresents the border's pixels respect to the ones in the middle of the input. An example this padding is depicted on the right in Figure A.5.

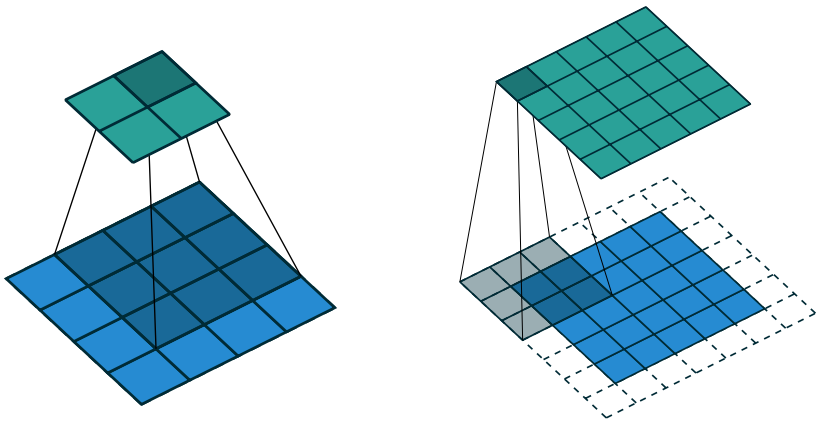


Figure A.5: On the left an example of valid padding and on the right an example of same padding

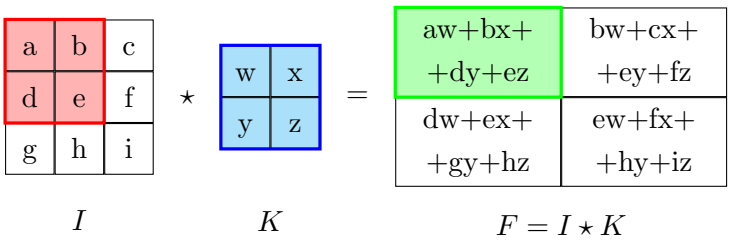
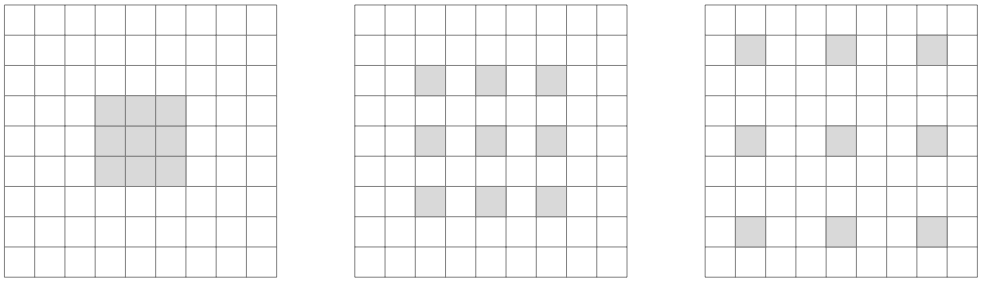


Figure A.6: An example of 2-D convolution without kernel flipping (cross-correlation) using valid padding

There exist a lot of modified convolutional layers. In this thesis we use only the dilated convolution, initially proposed in [110] and [215]. For an in-depth review of the arithmetic of convolution in deep learning refer to [216] or see Chapter 9 of [206].
Given the kernel $K \in \mathbb{R}^{k_1 \times k_2}$ and the dilation factor $d \in \mathbb{Z}^+$, the kernel of the dilated convolution layer is $K_d \in \mathbb{R}^{k_1(d) \times k_2(d)}$ is obtained inserting $d-1$ spaces between the elements of K, so $k_1(d) = k_1 d - d + 1$ and $k_2(d) = k_2 d - d + 1$. In view of the previous definition, a regular convolution corresponds to a dilated convolution with dilation factor $d = 1$. In Figure A.7 we represent the same kernel with different dilation factors. This type of convolution expands the receptive field of the layer without increasing the kernel size.

Another important part of convolutional networks are the pooling functions. Generally, pooling is used to make the representation invariant respect a set of transformations like rotation or scaling. The invariance property is useful in a classification tasks: we want the algorithm to recognize an object, independently from its orientation. In other tasks, like image reconstruction, preserving



(a) $d = 1$ (b) $d = 2$ (c) $d = 3$

Figure A.7: Dilated convolutional filters

the information in an exact location it is very important: we want the method to reconstruct also the finer details. Therefore, in this thesis we do not pay attention to this class of pooling functions and we will not explain formally how pooling works. The interested reader can refer to [206] for more details.
In a machine learning algorithm with convolutional layers, the kernel weights are obtained during the training phase. The use of convolution has some improvements compared to the traditional neural networks:

- Each entries of the kernel is used at every position of the input (except for the boundary pixels). Hence, the parameters are shared and used more than once. This property, called **parameter sharing**, makes the computation more efficient than a matrix multiplication and can be executed in parallel using a GPU. Moreover, the memory required to store a kernel K is smaller than the one needed for a full matrix.

- In a common neural network an output unit is influenced by all the input units in the previous layer (see Figure A.8). In a CNN an output unit depends on a small set of input units (see Figure A.9). The size of the set is related to the dimension of the convolutional kernel. This property, called **sparse interactions**, reduces the size of the features tensor and improves the model's efficiency. The interactions between larger portions of the input can be described stacking more convolutional layers.

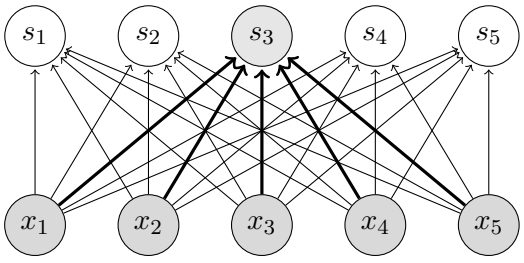


Figure A.8: Example of fully connected layer

- A convolutional layer is independent from the input dimensions. Hence, mathematically a CNN which works on images could be used without regard for the image size.

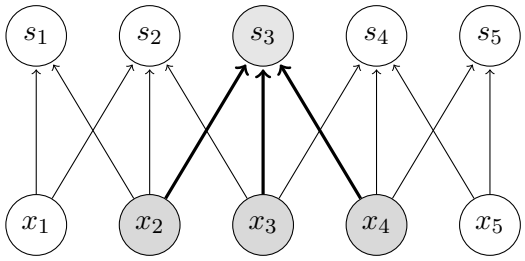


Figure A.9: Example of sparse connectivity

- The parameter sharing condition is linked to the **equivariance** property. In practice, let T be a translation and $\boldsymbol{N}$ a DNN we have $T(\boldsymbol{N}(x)) = \boldsymbol{N}(T(x))$. This means the feature map of a shifted input coincides with the shifted feature map of the initial input. This property characterizes only standard convolution, changing the stride of convolution equivariance does not hold. Moreover, convolution is not naturally equivariant to scaling or rotation.

Appendix B

Accelerated Forward Backward (AFB) algorithm

We describe the numerical procedure used to compute the solution of problem (3.7) for a fixed λ_k. Let us define the starting values $\mathbf{x}_0 = \mathbf{u}_{k-1}$ and $\mathbf{y}_0 = \mathbf{x}_0$, $t_0 = 1$, then by iterating on m until convergence:

$$\mathbf{v} = \mathbf{y}_m - \frac{\beta}{2r} \sum_{j=1}^{r} \mathbf{H}^T S^T (\mathbf{S}\mathbf{H}\mathbf{x}_m - \mathbf{b}^j) \tag{B.1}$$

$$\mathbf{x}_{m+1} = \arg\min_{\mathbf{u}\in\mathbb{R}^n} \left\{ \lambda_k \mathrm{TV}(\mathbf{u}) + \frac{1}{2\beta} \|\mathbf{y}_m - \beta(\mathbf{v} - \mathbf{u})\|^2 \right\} \tag{B.2}$$

$$t_{m+1} = 1 + \frac{\sqrt{1 + 4t_m^2}}{2},\ \alpha = \left(\frac{t_m - 1}{t_{m+1}} \right) \tag{B.3}$$

$$\mathbf{y}_{m+1} = \mathbf{x}_m + \alpha \left(\mathbf{x}_{m+1} - \mathbf{x}_m \right). \tag{B.4}$$

In order to guarantee the convergence of the FISTA iterations, we should choose the parameter β so that $0 < \beta < 2/\alpha$, where α is the unknown Lipschitz constant related to the data discrepancy. In our experiments we use $\beta = 10^{-2}$ which always heuristically ensured a convergent behavior. We observe that (B.2) is a denoising problem that is efficiently solved by the Chambolle algorithm [61], thus obtaining an Accelerated Forward-Backward (AFB) procedure.
Following [61], the solution of problem (B.2) is defined as:

$$\mathbf{x}_{m+1} = \mathbf{v} - \beta \mathbf{p}^{(N_m+1)}.$$

Setting $\mathbf{p}^{(0)} = 0$, each component (ℓ, μ) of $\mathbf{p}$ is iteratively computed as follows:

$$\mathbf{p}_{\ell,\mu}^{(n+1)} = \frac{\mathbf{p}_{\ell,\mu}^{(n)} + \tau \mathbf{W}_{\ell,\mu}^{(n)}}{1 + \tau \left| \mathbf{W}_{\ell,\mu}^{(n)} \right|}, \quad \mathbf{p}^{(0)} = 0, \quad n = 0, \ldots, N_m \tag{B.5}$$

where the components (ℓ, μ) of $\mathbf{W}^{(n)}$ are defined by means of the discrete gradient $\mathbf{D}$, using the forward finite differences, and by the discrete divergence operator div, i.e.:

$$\mathbf{W}_{\ell,\mu}^{(n)} = \mathbf{D} \left(\mathrm{div}(\mathbf{p}^{(n)}) - \frac{\mathbf{v}}{\beta} \right)_{\ell,\mu}.$$

The steps of the final AFB algorithm are reported in Algorithm 13.
As suggested in [61] the optimal condition for stability and convergence is $\tau \leq 1/4$; we set $\tau = 1/8$. The values of the indices (ℓ, μ) are the rows and columns of the HR image. The input $\mathbf{H}$ represents the blur matrix, $\mathbf{S}$ is the downsampling operator, $\mathcal{G}$ contains the set of low-resolution lexicographically reordered images $\mathbf{b}^j$, r is the number of low resolution images. The outer iterations (index m) of algorithm AFB in Algorithm 13 are stopped on the basis of the relative distance of the objective function Φ at two successive iterations and τ_f is the tolerance parameter.

Algorithm 13 – AFB: Accelerated Forward-Backward algorithm

input: $\mathbf{u}_{k-1} \in \mathbb{R}^n, \lambda_k, \mathbf{H} \in \mathbb{R}^{n \times n}, \mathbf{S} \in \mathbb{R}^{m \times n}, \mathcal{G}, r, \tau_f$

output: $\mathbf{x}_m$

$\mathbf{x}_0 = \mathbf{u}_{k-1}$, $\mathbf{y}_0 = \mathbf{x}_0$, $m = 0$, $\tau = 0.25$, $\beta = 10^{-2}$, $t_0 = 1$;
repeat
  $\mathbf{v} = \mathbf{y}_m - \frac{\lambda_k}{r} \sum_{j=1}^{r} \mathbf{H}^T \mathbf{S}^T (\mathbf{S}\mathbf{H}\mathbf{x}_m - \mathbf{b}^j)$
  $n = 0$, $\mathbf{p}_{\ell,\mu}^{(n)} = 0$, $\forall \ell, \mu$
  repeat
    compute $\mathbf{p}_{\ell,\mu}^{(n+1)}$ as in (B.5)
    $n = n + 1$
  until $\max_{\ell,\mu} |\mathbf{p}_{\ell,\mu}^{(n)} - \mathbf{p}_{\ell,\mu}^{(n-1)}| < 0.1 \cdot \max_{\ell,\mu} |\mathbf{p}^{(n)}|$
  $\mathbf{x}_{m+1} = \mathbf{v} - \beta \mathbf{p}$
  $t_{m+1} = 1 + \frac{\sqrt{1+4t_m^2}}{2}$
  $\mathbf{y}_{m+1} = \mathbf{x}_m + \frac{t_m - 1}{t_{m+1}} (\mathbf{x}_{m+1} - \mathbf{x}_m)$
  $m = m + 1$
until $|\Phi(\mathbf{x}_m, \lambda_k) - \Phi(\mathbf{x}_{m-1}, \lambda_k)| \leq \tau_f \Phi(\mathbf{x}_m, \lambda_k)$

Bibliography

[1] Jacques Hadamard. *Lectures on Cauchy's problem in linear partial differential equations*. Courier Corporation, 2003.

[2] Per Christian Hansen. *Discrete inverse problems: insight and algorithms*. SIAM, 2010.

[3] J Carlos Santamarina and Dante Fratta. *Discrete signals and inverse problems: an introduction for engineers and scientists*. John Wiley & Sons, 2005.

[4] Lennart Ljung. System identification. In *Signal analysis and prediction*, pages 163–173. Springer, 1998.

[5] Alexandre Gonçalves Evsukoff, Beatriz SLP De Lima, and Nelson FF Ebecken. Long-term runoff modeling using rainfall forecasts with application to the Iguaçu river basin. *Water resources management*, 25(3):963–985, 2011.

[6] Marcelo Matus, Doris Sáez, Mark Favley, Carlos Suazo-Martínez, José Moya, Guillermo Jiménez-Estévez, Rodrigo Palma-Behnke, Gabriel Olguín, and Pablo Jorquera. Identification of critical spans for monitoring systems in dynamic thermal rating. *IEEE Transactions on Power Delivery*, 27(2):1002–1009, 2012.

[7] Oliver Nelles. *Nonlinear System Identification: From Classical Approaches to Neural Networks, Fuzzy Models, and Gaussian Processes*. Springer Nature, 2020.

[8] Alan C Bovik. *Handbook of image and video processing*. Academic press, 2010.

[9] Mario Bertero, Patrizia Boccacci, and Christine De Mol. *Introduction to inverse problems in imaging*. CRC press, 2021.

[10] Per Christian Hansen. *Rank-deficient and discrete ill-posed problems: numerical aspects of linear inversion*. SIAM, 1998.

[11] Michael T McCann, Kyong Hwan Jin, and Michael Unser. Convolutional neural networks for inverse problems in imaging: A review. *IEEE Signal Processing Magazine*, 34(6):85–95, 2017.

[12] Simon Arridge, Peter Maass, Ozan Öktem, and Carola-Bibiane Schönlieb. Solving inverse problems using data-driven models. *Acta Numerica*, 28:1–174, 2019.

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

[14] Barbara Kaltenbacher, Andreas Neubauer, and Otmar Scherzer. *Iterative regularization methods for nonlinear ill-posed problems*. de Gruyter, 2008.

[15] Otmar Scherzer, Markus Grasmair, Harald Grossauer, Markus Haltmeier, and Frank Lenzen. *Variational methods in imaging*. Springer, 2009.

[16] Andrew M Stuart. Inverse problems: a bayesian perspective. *Acta numerica*, 19:451–559, 2010.

[17] Andrei Nikolaevich Tikhonov. On the solution of ill-posed problems and the method of regularization. In *Doklady Akademii Nauk*, volume 151, pages 501–504. Russian Academy of Sciences, 1963.

[18] Antonin Chambolle and Thomas Pock. An introduction to continuous optimization for imaging. *Acta Numerica*, 25:161–319, 2016.

[19] Emmanuel J Candès, Justin Romberg, and Terence Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on information theory*, 52(2):489–509, 2006.

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

[21] Antonin Chambolle and Pierre-Louis Lions. Image recovery via total variation minimization and related problems. *Numerische Mathematik*, 76(2):167–188, 1997.

[22] Kristian Bredies, Karl Kunisch, and Thomas Pock. Total generalized variation. *SIAM Journal on Imaging Sciences*, 3(3):492–526, 2010.

[23] Jianping Zhang and Ke Chen. A total fractional-order variation model for image restoration with nonhomogeneous boundary conditions and its numerical solution. *SIAM Journal on Imaging Sciences*, 8(4):2487–2518, 2015.

[24] Wenjuan Yao, Jie Shen, Zhichang Guo, Jiebao Sun, and Boying Wu. A total fractional-order variation model for image super-resolution and its SAV algorithm. *J. Sci. Comput.*, 82(3):1–18, 2020.

[25] Li Xu, Cewu Lu, Yi Xu, and Jiaya Jia. Image smoothing via l0 gradient minimization. *ACM Trans. Graph (SIGGRAPH Asia)*, 2011.

[26] Stefan Roth and Michael J Black. Fields of experts: A framework for learning image priors. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 860–867. IEEE, 2005.

[27] Erich Kobler, Alexander Effland, Karl Kunisch, and Thomas Pock. Total deep variation: A stable regularization method for inverse problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[28] Florian Knoll, Kerstin Hammernik, Chi Zhang, Steen Moeller, Thomas Pock, Daniel K Sodickson, and Mehmet Akcakaya. Deep-learning methods for parallel magnetic resonance imaging reconstruction: A survey of the current approaches, trends, and issues. *IEEE signal processing magazine*, 37(1):128–140, 2020.

[29] Singanallur V Venkatakrishnan, Charles A Bouman, and Brendt Wohlberg. Plug-and-play priors for model based reconstruction. In *2013 IEEE Global Conference on Signal and Information Processing*, pages 945–948. IEEE, 2013.

[30] Yaniv Romano, Michael Elad, and Peyman Milanfar. The little engine that could: Regularization by denoising (red). *SIAM Journal on Imaging Sciences*, 10(4):1804–1844, 2017.

[31] Jorge Nocedal and Stephen Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

[32] Antoni Buades, Bartomeu Coll, and J-M Morel. A non-local algorithm for image denoising. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 60–65. IEEE, 2005.

[33] Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Image denoising by sparse 3-d transform-domain collaborative filtering. *IEEE Transactions on image processing*, 16(8):2080–2095, 2007.

[34] Kai Zhang, Wangmeng Zuo, Shuhang Gu, and Lei Zhang. Learning deep cnn denoiser prior for image restoration. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3929–3938, 2017.

[35] Tim Meinhardt, Michael Moller, Caner Hazirbas, and Daniel Cremers. Learning proximal operators: Using denoising networks for regularizing inverse imaging problems. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1781–1790, 2017.

[36] JH Rick Chang, Chun-Liang Li, Barnabas Poczos, BVK Vijaya Kumar, and Aswin C Sankaranarayanan. One network to solve them all–solving linear inverse problems using deep projection models. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5888–5897, 2017.

[37] Kai Zhang, Yawei Li, Wangmeng Zuo, Lei Zhang, Luc Van Gool, and Radu Timofte. Plug-and-play image restoration with deep denoiser prior. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[38] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Accurate image super-resolution using very deep convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1646–1654, 2016.

[39] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9446–9454, 2018.

[40] Matthew Fox, Steve Goodhew, and Pieter De Wilde. Building defect detection: external versus internal thermography. *Building and Environment*, 105:317–331, aug 2016.

[41] Krishna Ribeiro Gomes, David Hernández López, José Ortega, Rocío Ballesteros, Tomás Poblete, and Miguel Moreno. Uncooled thermal camera calibration and optimization of the photogrammetry process for UAV applications in agriculture. *Sensors*, 17(10):2173, sep 2017.

[42] Thomas Luhmann, Johannes Piechel, and Thorsten Roelfs. Geometric calibration of thermographic cameras. In Claudia Kuenzer and Stefan Dech, editors, *Thermal Infrared Remote Sensing*, volume 17 of *Remote Sensing and Digital Image Processing*, chapter 2, pages 27–42. Springer Netherlands, 2013.

[43] Pasquale Cascarano, Francesco Corsini, Stefano Gandolfi, Elena Loli Piccolomini, Emanuele Mandanici, Luca Tavasci, and Fabiana Zama. Super-resolution of thermal images using an automatic total variation based method. *Remote Sensing*, 12(10):1642, 2020.

[44] Russell C. Hardie, Kenneth J. Barnard, John G. Bognar, Ernest E. Armstrong, and Edward A. Watson. High-resolution image reconstruction from a sequence of rotated and translated frames and its application to an infrared imaging system. *Optical Engineering*, 37(1):247, jan 1998.

[45] Gang Sun, Qinghui Li, and Lin Lu. MAP algorithm to super-resolution of infrared images. In Henri Maître, Hong Sun, Jianguo Liu, and Enmin Song, editors, *MIPPR 2007: Multispectral Image Processing*, volume 6787, pages 1–9. SPIE, nov 2007.

[46] Antigoni Panagiotopoulou and Vassilis Anastassopoulos. Super-resolution reconstruction of thermal infrared images. In *Proceedings of the 4th WSEAS International Conference on REMOTE SENSING*, 2008.

[47] Hui Yu, Fu-sheng Chen, Zhi-jie Zhang, and Chen-sheng Wang. Single infrared image super-resolution combining non-local means with kernel regression. *Infrared Physics & Technology*, 61:50–59, nov 2013.

[48] Pablo Meza, Guillermo Machuca, Sergio Torres, Cesar San Martin, and Esteban Vera. Simultaneous digital super-resolution and nonuniformity correction for infrared imaging systems. *Applied Optics*, 54(21):6508, jul 2015.

[49] L. B. Montefusco and D. Lazzaro. An iterative L1-based image restoration algorithm with an adaptive parameter estimation. *IEEE Transactions on Image Processing*, 21(4):1676–1686, apr 2012.

[50] Peter Craven and Grace Wahba. Smoothing noisy data with spline functions. *Numerische mathematik*, 31(4):377–403, 1978.

[51] Per Christian Hansen, Toke Koldborg Jensen, and Giuseppe Rodriguez. An adaptive pruning algorithm for the discrete l-curve criterion. *Journal of computational and applied mathematics*, 198(2):483–492, 2007.

[52] You-Wei Wen and Andy M Yip. Adaptive parameter selection for total variation image deconvolution. *Numer. Math. Theor. Meth. Appl*, 2(4):427–438, 2009.

[53] Joao P Oliveira, José M Bioucas-Dias, and Mário AT Figueiredo. Adaptive total variation image deblurring: a majorization–minimization approach. *Signal processing*, 89(9):1683–1693, 2009.

[54] Michael K Ng, Pierre Weiss, and Xiaoming Yuan. Solving constrained total-variation image restoration and reconstruction problems via alternating direction methods. *SIAM journal on Scientific Computing*, 32(5):2710–2736, 2010.

[55] Chuan He, Changhua Hu, Wei Zhang, and Biao Shi. A fast adaptive parameter estimation for total variation image restoration. *IEEE Transactions on Image Processing*, 23(12):4954–4967, 2014.

[56] Vladimir Alekseevich Morozov. On the solution of functional equations by the method of regularization. In *Doklady Akademii Nauk*, volume 167, pages 510–512. Russian Academy of Sciences, 1966.

[57] Stamatios Lefkimmiatis, John Paul Ward, and Michael Unser. Hessian schatten-norm regularization for linear inverse problems. *IEEE transactions on image processing*, 22(5):1873–1888, 2013.

[58] Yue Hu, Greg Ongie, Sathish Ramani, and Mathews Jacob. Generalized higher degree total variation (hdtv) regularization. *IEEE transactions on image processing*, 23(6):2423–2435, 2014.

[59] Nicholas Ian and Mark Gould. On the convergence of a sequential penalty function method for constrained minimization. *SIAM Journal on Numerical Analysis*, 26(1):107–128, 1989.

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

[61] Antonin Chambolle. An algorithm for total variation minimization and applications. *Journal of Mathematical Imaging and Vision*, 20(1):89–97, Jan 2004.

[62] Emanuele Mandanici, Luca Tavasci, Francesco Corsini, and Stefano Gandolfi. A multi-image super-resolution algorithm applied to thermal imagery. *Applied Geomatics*, feb 2019.

[63] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 136–144, 2017.

[64] Huanfeng Shen, Liangpei Zhang, Bo Huang, and Pingxiang Li. A MAP approach for joint motion estimation, segmentation, and super resolution. *IEEE Transactions on Image processing*, 16(2):479–490, 2007.

[65] Majdi Mansouri and Ali Mohammad-Djafari. Joint super-resolution and segmentation from a set of low resolution images using a Bayesian approach with a Gauss-Markov-Potts prior. *International Journal of Signal and Imaging Systems Engineering*, 3(4):211–221, 2010.

[66] Quentin Delannoy, Chi-Hieu Pham, Clément Cazorla, Carlos Tor-Díez, Guillaume Dollé, Hélène Meunier, Nathalie Bednarek, Ronan Fablet, Nicolas Passat, and François Rousseau. SegSRGAN: Super-resolution and segmentation using generative adversarial networks—Application to neonatal brain MRI. *Computers in Biology and Medicine*, 120:103755, 2020.

[67] Martin Storath, Andreas Weinmann, and Laurent Demaret. Jump-sparse and sparse recovery using potts functionals. *IEEE Transactions on Signal Processing*, 62(14):3654–3666, 2014.

[68] Martin Storath, Andreas Weinmann, Jürgen Frikel, and Michael Unser. Joint image reconstruction and segmentation using the Potts model. *Inverse Problems*, 31(2):025003, 2015.

[69] Martin Storath, Dennis Rickert, Michael Unser, and Andreas Weinmann. Fast segmentation from blurred data in 3D fluorescence microscopy. *IEEE Transactions on Image Processing*, 26(10):4856–4870, 2017.

[70] Lukas Kiefer, Martin Storath, and Andreas Weinmann. PALMS: Image Partitioning-A New Parallel Algorithm for the Piecewise Affine-Linear Mumford-Shah Model. *Image Processing On Line*, 10:124–149, 2020.

[71] S. Ono. l_0 gradient projection. *IEEE Trans. Image Process.*, 26(4):1554–1564, 2017.

[72] Pasquale Cascarano, Luca Calatroni, and Elena Loli piccolomini. Efficient ℓ^0 gradient-based super resolution for simplified image segmentation. *IEEE Transactions on Computational Imaging*, pages 1–1, 2021.

[73] Dario Mylonopoulos, Pasquale Cascarano, Luca Calatroni, and Elena Loli Piccolomini. Constrained and unconstrained inverse potts modeling for joint image super-resolution and segmentation. *Image Processing On Line (IPOL)*, 2022.

[74] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23(11):1222–1239, 2001.

[75] Ningning Zhao, Qi Wei, Adrian Basarab, Nicolas Dobigeon, Denis Kouamé, and Jean-Yves Tourneret. Fast Single Image Super-Resolution Using a New Analytical Solution for l2-l2 Problems. *IEEE Transactions on Image Processing*, 25(8):3683–3697, 2016.

[76] Lukas Kiefer, Martin Storath, and Andreas Weinmann. Iterative Potts minimization for the recovery of signals with discontinuities from indirect measurements: the multivariate case. *Foundations of Computational Mathematics*, 21(3):649–694, 2021.

[77] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine learning*, 3(1):1–122, 2011.

[78] Yu Wang, Wotao Yin, and Jinshan Zeng. Global Convergence of ADMM in Nonconvex Nonsmooth Optimization. *Journal of Scientific Computing*, 78(1):29–63, 2019.

[79] Thomas Blumensath and Mike E Davies. Iterative hard thresholding for compressed sensing. *Appl. Comput. Harmon. A.*, 27(3):265–274, 2009.

[80] Max A Woodbury. *Inverting modified matrices*. Statistical Research Group, 1950.

[81] S. J. Osher A. Marquina. Image super-resolution by TV-regularization and bregman iteration. *J. Sci. Comput.*, 37:367–382, 2008.

[82] Tong Zhang. Multi-stage convex relaxation for learning with sparse regularization. In *Adv. Neural. Inf. Process. Syst.*, pages 1929–1936, 2009.

[83] Feishe Chen, Lixin Shen, and Bruce W Suter. Computing the proximity operator of the ℓ_p norm with $0 < p < 1$. *IET Signal Processing*, 10(5):557–565, 2016.

[84] Wanjie Sun and Zhenzhong Chen. Learned image downscaling for upscaling using content adaptive resampler. *IEEE Trans. Image Process.*, 29:4027–4040, 2020.

[85] Kai Zhang, Wangmeng Zuo, Shuhang Gu, and Lei Zhang. Learning deep CNN denoiser prior for image restoration. In *Proceedings of the IEEE CVPR*, pages 3929–3938, 2017.

[86] Y. Kato, D. Deguchi, T. Takahashi, I. Ide, and H. Murase. Low resolution QR-code recognition by applying super-resolution using the property of qr-codes. In *ICDAR 2011*, volume IEEE ICDAR, pages 992–996.

[87] Mischa Schwendy, Ronald E Unger, and Sapun H Parekh. EVICAN–a balanced dataset for algorithm development in cell and nucleus segmentation. *Bioinformatics*, 2020.

[88] VL Coli, E Loli Piccolomini, E Morotti, and L Zanni. A fast gradient projection method for 3d image reconstruction from limited tomographic data. In *Journal of Physics: Conference Series*, volume 904, page 012013. IOP Publishing, 2017.

[89] Elena Loli Piccolomini and Elena Morotti. A model-based optimization framework for iterative digital breast tomosynthesis image reconstruction. *Journal of Imaging*, 7(2), 2021.

[90] Kaixuan Wei, Angelica Aviles-Rivero, Jingwei Liang, Ying Fu, Carola-Bibiane Schönlieb, and Hua Huang. Tuning-free plug-and-play proximal algorithm for inverse imaging problems. In *International Conference on Machine Learning*, pages 10158–10169. PMLR, 2020.

[91] Ji He, Yan Yang, Yongbo Wang, Dong Zeng, Zhaoying Bian, Hao Zhang, Jian Sun, Zongben Xu, and Jianhua Ma. Optimizing a parameterized plug-and-play admm for iterative low-dose ct reconstruction. *IEEE transactions on medical imaging*, 38(2):371–382, 2018.

[92] Pasquale Cascarano, Elena Loli Piccolomini, Elena Morotti, and Andrea Sebastiani. Plug-and-play gradient-based denoisers applied to ct image enhancement. *Applied Mathematics and Computation*, 2022.

[93] Suhas Sreehari, S Venkat Venkatakrishnan, Brendt Wohlberg, Gregery T Buzzard, Lawrence F Drummy, Jeffrey P Simmons, and Charles A Bouman. Plug-and-play priors for bright field electron tomography and sparse interpolation. *IEEE Transactions on Computational Imaging*, 2(4):408–423, 2016.

[94] Stanley H Chan, Xiran Wang, and Omar A Elgendy. Plug-and-play ADMM for image restoration: Fixed-point convergence and applications. *IEEE Transactions on Computational Imaging*, 3(1):84–98, 2016.

[95] Ulugbek S Kamilov, Hassan Mansour, and Brendt Wohlberg. A plug-and-play priors approach for solving nonlinear imaging inverse problems. *IEEE Signal Processing Letters*, 24(12):1872–1876, 2017.

[96] Rizwan Ahmad, Charles A Bouman, Gregery T Buzzard, Stanley Chan, Sizhuo Liu, Edward T Reehorst, and Philip Schniter. Plug-and-play methods for magnetic resonance imaging: Using denoisers for image recovery. *IEEE signal processing magazine*, 37(1):105–116, 2020.

[97] Tianlong Chen, Xiaohan Chen, Wuyang Chen, Howard Heaton, Jialin Liu, Zhangyang Wang, and Wotao Yin. Learning to Optimize: A Primer and A Benchmark, 2021.

[98] Inbar Mosseri, Maria Zontak, and Michal Irani. Combining the power of internal and external denoising. In *IEEE international conference on computational photography (ICCP)*, pages 1–9. IEEE, 2013.

[99] Daniel Zoran and Yair Weiss. From learning models of natural image patches to whole image restoration. In *2011 International Conference on Computer Vision*, pages 479–486. IEEE, 2011.

[100] Yunjin Chen and Thomas Pock. Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1256–1272, 2016.

[101] Junyuan Xie, Linli Xu, and Enhong Chen. Image denoising and inpainting with deep neural networks. In *Advances in neural information processing systems*, pages 341–349, 2012.

[102] Harold C Burger, Christian J Schuler, and Stefan Harmeling. Image denoising: Can plain neural networks compete with BM3D? In *2012 IEEE conference on computer vision and pattern recognition*, pages 2392–2399. IEEE, 2012.

[103] Kai Zhang, Yawei Li, Wangmeng Zuo, Lei Zhang, Luc Van Gool, and Radu Timofte. Plug-and-Play image restoration with Deep Denoiser Prior. *arXiv preprint*, 2020.

[104] Arie Rond, Raja Giryes, and Michael Elad. Poisson inverse problems by the plug-and-play scheme. *Journal of Visual Communication and Image Representation*, 41:96–108, 2016.

[105] Liangtian He, Yilun Wang, and Shaobing Gao. A support-denoiser-driven framework for single image restoration. *Journal of Computational and Applied Mathematics*, page 113495, 2021.

[106] Neal Parikh and Stephen Boyd. Proximal algorithms. *Foundations and Trends in optimization*, 1(3):127–239, 2014.

[107] Jean-Jacques Moreau. Proximité et dualité dans un espace hilbertien. *Bulletin de la Société mathématique de France*, 93:273–299, 1965.

[108] Donald Geman and Chengda Yang. Nonlinear image recovery with half-quadratic regularization. *IEEE transactions on Image Processing*, 4(7):932–946, 1995.

[109] Yilun Wang, Junfeng Yang, Wotao Yin, and Yin Zhang. A new alternating minimization algorithm for total variation image reconstruction. *SIAM Journal on Imaging Sciences*, 1(3):248–272, 2008.

[110] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015.

[111] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A Database of Human Segmented Natural Images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proc. 8th Int'l Conf. Computer Vision*, volume 2, pages 416–423, July 2001.

[112] Vladimir Alekseevich Morozov. *Methods for solving incorrectly posed problems*. Springer, 1984.

[113] Alain Hore and Djemel Ziou. Image quality metrics: Psnr vs. ssim. In *2010 20th international conference on pattern recognition*, pages 2366–2369. IEEE, 2010.

[114] Malte Renz. Fluorescence microscopy—A historical and technical perspective. *Cytometry Part A*, 83(9):767–779, 2013.

[115] Nikolay I Zheludev. What diffraction limit? *Nature materials*, 7(6):420–422, 2008.

[116] Ernst Abbe. Beiträge zur theorie des mikroskops und der mikroskopischen wahrnehmung. *Archiv für mikroskopische Anatomie*, 9(1):413–468, 1873.

[117] Steffen J Sahl and WE Moerner. Super-resolution fluorescence imaging with single molecules. *Current opinion in structural biology*, 23(5):778–787, 2013.

[118] Kurt Rossmann. Point spread-function, line spread-function, and modulation transfer function: tools for the study of imaging systems. *Radiology*, 93(2):257–272, 1969.

[119] Daniel Sage, Hagai Kirshner, Thomas Pengo, Nico Stuurman, Junhong Min, Suliana Manley, and Michael Unser. Quantitative evaluation of software packages for single-molecule localization microscopy. *Nature methods*, 12(8):717–724, 2015.

[120] Kristoffer Bernhem and Hjalmar Brismar. SMLocalizer, a GPU accelerated ImageJ plugin for single molecule localization microscopy. *Bioinformatics*, 34(1):137–138, 2018.

[121] Janel L Davis, Brian Soetikno, Ki-Hee Song, Yang Zhang, Cheng Sun, and Hao F Zhang. RainbowSTORM: an open-source ImageJ plug-in for spectroscopic single-molecule localization microscopy (sSMLM) data analysis and image reconstruction. *Bioinformatics*, 36(19):4972–4974, 2020.

[122] Y Garini, A Gil, I Bar-Am, D Cabib, and N Katzir. Signal to noise analysis of multiple color fluorescence imaging microscopy. *Cytometry: The Journal of the International Society for Analytical Cytology*, 35(3):214–226, 1999.

[123] Jennifer C Waters. Accuracy and precision in quantitative fluorescence microscopy, 2009.

[124] Anna Jezierska, Hugues Talbot, Caroline Chaux, Jean-Christophe Pesquet, and Gilbert Engler. Poisson-gaussian noise parameter estimation in fluorescence microscopy imaging. In *2012 9th IEEE International Symposium on Biomedical Imaging (ISBI)*, pages 1663–1666. IEEE, 2012.

[125] Pasquale Cascarano, Maria Colomba Comes, Andrea Sebastiani, Arianna Mencattini, Elena Loli Piccolomini, and Eugenio Martinelli. DeepCEL0 for 2D single-molecule localization in fluorescence microscopy. *Bioinformatics*, 12 2021. btab808.

[126] Elias Nehme, Lucien E Weiss, Tomer Michaeli, and Yoav Shechtman. Deep-STORM: super-resolution single-molecule microscopy by deep learning. *Optica*, 5(4):458–464, 2018.

[127] Simon Gazagnes, Emmanuel Soubies, and Laure Blanc-Féraud. High density molecule localization for super-resolution microscopy using CEL0 based sparse approximation. In *2017 IEEE 14th International Symposium on Biomedical Imaging (ISBI 2017)*, pages 28–31. IEEE, 2017.

[128] Emmanuel Soubies, Laure Blanc-Féraud, and Gilles Aubert. A continuous exact \ell_0 penalty (CEL0) for least squares regularized problem. *SIAM Journal on Imaging Sciences*, 8(3):1607–1639, 2015.

[129] Thomas Dertinger, Ryan Colyer, Gopal Iyer, Shimon Weiss, and Jörg Enderlein. Fast, background-free, 3D super-resolution optical fluctuation imaging (SOFI). *Proceedings of the National Academy of Sciences*, 106(52):22287–22292, 2009.

[130] Susan Cox, Edward Rosten, James Monypenny, Tijana Jovanovic-Talisman, Dylan T Burnette, Jennifer Lippincott-Schwartz, Gareth E Jones, and Rainer Heintzmann. Bayesian localization microscopy reveals nanoscale podosome dynamics. *Nature methods*, 9(2):195–200, 2012.

[131] Nils Gustafsson, Siân Culley, George Ashdown, Dylan M Owen, Pedro Matos Pereira, and Ricardo Henriques. Fast live-cell conventional fluorophore nanoscopy with ImageJ through super-resolution radial fluctuations. *Nature communications*, 7(1):1–9, 2016.

[132] Ricardo Henriques, Mickael Lelek, Eugenio F Fornasiero, Flavia Valtorta, Christophe Zimmer, and Musa M Mhlanga. QuickPALM: 3D real-time photoactivation nanoscopy image processing in ImageJ. *Nature methods*, 7(5):339–340, 2010.

[133] Matthew P Gordon, Taekjip Ha, and Paul R Selvin. Single-molecule high-resolution imaging with photobleaching. *Proceedings of the National Academy of Sciences*, 101(17):6462–6465, 2004.

[134] Xiaohui Qu, David Wu, Laurens Mets, and Norbert F Scherer. Nanometer-localized multiple single-molecule fluorescence microscopy. *Proceedings of the National Academy of Sciences*, 101(31):11298–11303, 2004.

[135] Arnauld Sergé, Nicolas Bertaux, Hervé Rigneault, and Didier Marguet. Dynamic multiple-target tracing to probe spatiotemporal cartography of cell membranes. *Nature methods*, 5(8):687–694, 2008.

[136] Seamus J Holden, Stephan Uphoff, and Achillefs N Kapanidis. Daostorm: an algorithm for high-density super-resolution microscopy. *Nature methods*, 8(4):279–280, 2011.

[137] Junhong Min, Cédric Vonesch, Hagai Kirshner, Lina Carlini, Nicolas Olivier, Seamus Holden, Suliana Manley, Jong Chul Ye, and Michael Unser. Falcon: fast and unbiased reconstruction of high-density super-resolution microscopy data. *Scientific reports*, 4(1):1–9, 2014.

[138] Siewert Hugelier, Johan J De Rooi, Romain Bernex, Sam Duwé, Olivier Devos, Michel Sliwa, Peter Dedecker, Paul HC Eilers, and Cyril Ruckebusch. Sparse deconvolution of high-density super-resolution images. *Scientific reports*, 6(1):1–11, 2016.

[139] Oren Solomon, Yonina C Eldar, Maor Mutzafi, and Mordechai Segev. Sparcom: Sparsity based super-resolution correlation microscopy. *SIAM Journal on Imaging Sciences*, 12(1):392–419, 2019.

[140] Nicholas Boyd, Geoffrey Schiebinger, and Benjamin Recht. The alternating descent conditional gradient method for sparse inverse problems. *SIAM Journal on Optimization*, 27(2):616–639, 2017.

[141] Jiaqing Huang, Mingzhai Sun, Jianjie Ma, and Yuejie Chi. Super-resolution image reconstruction for high-density three-dimensional single-molecule microscopy. *IEEE Transactions on Computational Imaging*, 3(4):763–773, 2017.

[142] Artur Speiser, Lucas-Raphael Müller, Philipp Hoess, Ulf Matti, Christopher J Obara, Wesley R Legant, Anna Kreshuk, Jakob H Macke, Jonas Ries, and Srinivas C Turaga. Deep learning enables fast and dense single-molecule localization with high accuracy. *Nature methods*, pages 1–9, 2021.

[143] Peter Ochs, Alexey Dosovitskiy, Thomas Brox, and Thomas Pock. On iteratively reweighted algorithms for nonsmooth nonconvex optimization in computer vision. *SIAM Journal on Imaging Sciences*, 8(1):331–372, 2015.

[144] Martin Ovesnỳ, Pavel Křížek, Josef Borkovec, Zdeněk Švindrych, and Guy M Hagen. ThunderSTORM: a comprehensive ImageJ plug-in for PALM and STORM data analysis and super-resolution imaging. *Bioinformatics*, 30(16):2389–2390, 2014.

[145] Perrine Paul, Heiko Duessmann, Tytus Bernas, Heinrich Huber, and Dimitrios Kalamatianos. Automatic noise quantification for confocal fluorescence microscopy images. *Computerized Medical Imaging and Graphics*, 34(6):426–434, 2010.

[146] Biagio Mandracchia, Xuanwen Hua, Changliang Guo, Jeonghwan Son, Tara Urner, and Shu Jia. Fast and accurate scmos noise correction for fluorescence microscopy. *Nature communications*, 11(1):1–12, 2020.

[147] Makhlad Chahid. *Echantillonnage compressif appliqué à la microscopie de fluorescence et à la microscopie de super résolution.* PhD thesis, Bordeaux, 2014.

[148] Arne Bechensteen, Laure Blanc-Féraud, and Gilles Aubert. New methods for L 2-L 0 minimization and their applications to 2D Single-Molecule Localization Microscopy. In *2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)*, pages 1377–1381. IEEE, 2019.

[149] Martin J Willemink, Wojciech A Koszek, Cailin Hardell, Jie Wu, Dominik Fleischmann, Hugh Harvey, Les R Folio, Ronald M Summers, Daniel L Rubin, and Matthew P Lungren. Preparing medical imaging data for machine learning. *Radiology*, 295(1):4–15, 2020.

[150] Weijie Gan, Cihat Eldeniz, Jiaming Liu, Sihao Chen, Hongyu An, and Ulugbek S Kamilov. Image Reconstruction for MRI using Deep CNN Priors Trained without Groundtruth. In *2020 54th Asilomar Conference on Signals, Systems, and Computers*, pages 475–479. IEEE, 2020.

[151] Yosef Gandelsman, Assaf Shocher, and Michal Irani. "Double-DIP": Unsupervised Image Decomposition via Coupled Deep-Image-Priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11026–11035, 2019.

[152] Alexander Krull, Tim-Oliver Buchholz, and Florian Jug. Noise2void-Learning Denoising from Single Noisy Images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2129–2137, 2019.

[153] Jaakko Lehtinen, Jacob Munkberg, Jon Hasselgren, Samuli Laine, Tero Karras, Miika Aittala, and Timo Aila. Noise2Noise: Learning Image Restoration without Clean Data. In *International Conference on Machine Learning*, pages 2965–2974. PMLR, 2018.

[154] Zezhou Cheng, Matheus Gadelha, Subhransu Maji, and Daniel Sheldon. A Bayesian perspective on the Deep Image Prior. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5443–5451, 2019.

[155] Sören Dittmer, Tobias Kluth, Peter Maass, and Daniel Otero Baguer. Regularization by architecture: A deep prior approach for inverse problems. *Journal of Mathematical Imaging and Vision*, 62(3):456–470, 2020.

[156] Daniel Otero Baguer, Johannes Leuschner, and Maximilian Schmidt. Computed tomography reconstruction using Deep Image Prior and learned reconstruction methods. *Inverse Problems*, 36(9):094004, sep 2020.

[157] Pasquale Cascarano, Maria Colomba Comes, Arianna Mencattini, Maria Carla Parrini, Elena Loli Piccolomini, and Eugenio Martinelli. Recursive deep prior video: a super resolution algorithm for time-lapse microscopy of organ-on-chip experiments. *Medical Image Analysis*, page 102124, 2021.

[158] Pasquale Cascarano, Andrea Sebastiani, Maria Colomba Comes, Giorgia Franchini, and Federica Porta. Combining Weighted Total Variation and Deep Image Prior for natural and medical image restoration via ADMM. *arXiv preprint arXiv:2009.11380*, 2021.

[159] Jiaming Liu, Yu Sun, Xiaojian Xu, and Ulugbek S Kamilov. Image restoration using Total Variation regularized Deep Image Prior. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7715–7719. IEEE, 2019.

[160] Gary Mataev, Peyman Milanfar, and Michael Elad. DeepRED: Deep Image Prior powered by RED. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 1–10, 2019.

[161] Dave Van Veen, Ajil Jalal, Mahdi Soltanolkotabi, Eric Price, Sriram Vishwanath, and Alexandros G Dimakis. Compressed sensing with deep image prior and learned regularization. *arXiv preprint arXiv:1806.06438*, 2018.

[162] Pasquale Cascarano, Giorgia Franchini, Erich Kobler, Federica Porta, and Andrea Sebastiani. Constrained and unconstrained deep image prior optimization models with automatic regularization. *arXiv*, 2021.

[163] V Bortolotti, RJS Brown, P Fantazzini, Germana Landi, and Fabiana Zama. Uniform Penalty inversion of two-dimensional NMR relaxation data. *Inverse Problems*, 33(1):015003, 2016.

[164] John Immerkaer. Fast noise variance estimation. *Computer vision and image understanding*, 64(2):300–302, 1996.

[165] Priyanka Kokil and Turimerla Pratap. Additive white gaussian noise level estimation for natural images using linear scale-space features. *Circuits, Systems, and Signal Processing*, 40(1):353–374, 2021.

[166] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[167] Marco Bevilacqua, Aline Roumy, Christine Guillemot, and Marie line Alberi Morel. Low-Complexity Single-Image Super-Resolution based on Nonnegative Neighbor Embedding. In *Proceedings of the British Machine Vision Conference*, pages 135.1–135.10. BMVA Press, 2012.

[168] Tao Yan, Pak Kin Wong, Hao Ren, Huaqiao Wang, Jiangtao Wang, and Yang Li. Automatic distinction between COVID-19 and common pneumonia using multi-scale convolutional neural network on chest CT scans. *Chaos, Solitons & Fractals*, 140:110153, 2020.

[169] Alessandro Polini, Ljupcho Prodanov, Nupura S Bhise, Vijayan Manoharan, Mehmet R Dokmeci, and Ali Khademhosseini. Organs-on-a-chip: a new tool for drug discovery. *Expert opinion on drug discovery*, 9(4):335–352, 2014.

[170] Alexandre J Kabla. Collective cell migration: leadership, invasion and segregation. *Journal of The Royal Society Interface*, 9(77):3268–3278, 2012.

[171] Luca Businaro, Adele De Ninno, Giovanna Schiavoni, Valeria Lucarini, Gabriele Ciasca, Annamaria Gerardino, Filippo Belardelli, Lucia Gabriele, and Fabrizio Mattei. Cross talk between cancer and immune cells: exploring complex dynamics in a microfluidic environment. *Lab on a Chip*, 13(2):229–239, 2013.

[172] Elena Agliari, Elena Biselli, Adele De Ninno, Giovanna Schiavoni, Lucia Gabriele, Anna Gerardino, Fabrizio Mattei, Adriano Barra, and Luca Businaro. Cancer-driven dynamics of immune cells in a microfluidic environment. *Scientific reports*, 4(1):1–15, 2014.

[173] Tanya J Shaw and Paul Martin. Wound repair at a glance. *Journal of cell science*, 122(18):3209–3213, 2009.

[174] Peter Friedl and Darren Gilmour. Collective cell migration in morphogenesis, regeneration and cancer. *Nature reviews Molecular cell biology*, 10(7):445–457, 2009.

[175] Peter Friedl and Katarina Wolf. Tumour-cell invasion and migration: diversity and escape mechanisms. *Nature reviews cancer*, 3(5):362–374, 2003.

[176] Elena Biselli, Elena Agliari, Adriano Barra, Francesca Romana Bertani, Annamaria Gerardino, Adele De Ninno, Arianna Mencattini, Davide Di Giuseppe, Fabrizio Mattei, Giovanna Schiavoni, et al. Organs on chip approach: a tool to evaluate cancer-immune cells interactions. *Scientific reports*, 7(1):1–12, 2017.

[177] Marie Nguyen, Adele De Ninno, Arianna Mencattini, Fanny Mermet-Meillon, Giulia Fornabaio, Sophia S Evans, Mélissande Cossutta, Yasmine Khira, Weijing Han, Philémon Sirven, et al. Dissecting effects of anti-cancer drugs and cancer-associated fibroblasts by on-chip reconstitution of immunocompetent tumor microenvironments. *Cell reports*, 25(13):3884–3893, 2018.

[178] Davide Di Giuseppe, Francesca Corsi, Arianna Mencattini, Maria Colomba Comes, Paola Casti, Corrado Di Natale, Lina Ghibelli, and Eugenio Martinelli. Learning cancer-related drug efficacy exploiting consensus in coordinated motility within cell clusters. *IEEE Transactions on Biomedical Engineering*, 66(10):2882–2888, 2019.

[179] Stefania Parlato, Adele De Ninno, Rosa Molfetta, Elena Toschi, Debora Salerno, Arianna Mencattini, Giulia Romagnoli, Alessandra Fragale, Lorenzo Roccazzello, Maria Buoncervello,

et al. 3d microfluidic model for evaluating immunotherapy efficacy by tracking dendritic cell behaviour toward tumor cells. *Scientific reports*, 7(1):1–16, 2017.

[180] M. C Comes, P. Casti, A. Mencattini, D. Di Giuseppe, F. Mermet-Meillon, A. De Ninno, M. C. Parrini, L. Businaro, C. Di Natale, and E. Martinelli. The influence of spatial and temporal resolutions on the analysis of cell-cell interaction: a systematic study for time-lapse microscopy applications. *Scientific Reports*, 9:1–11, 2019.

[181] Joost B Beltman, Athanasius FM Marée, and Rob J De Boer. Analysing immune cell migration. *Nature Reviews Immunology*, 9(11):789–798, 2009.

[182] E Roy Davies. *Machine vision: theory, algorithms, practicalities*. Elsevier, 2004.

[183] James Munkres. Algorithms for the assignment and transportation problems. *Journal of the society for industrial and applied mathematics*, 5(1):32–38, 1957.

[184] Johannes Huth, Malte Buchholz, Johann M Kraus, Martin Schmucker, Götz Von Wichert, Denis Krndija, Thomas Seufferlein, Thomas M Gress, and Hans A Kestler. Significantly improved precision of cell migration analysis in time-lapse video microscopy through use of a fully automated tracking system. *BMC cell biology*, 11(1):24, 2010.

[185] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. ESRGAN: Enhanced super-resolution generative adversarial networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 0–0, 2018.

[186] Yifan Wang, Federico Perazzi, Brian McWilliams, Alexander Sorkine-Hornung, Olga Sorkine-Hornung, and Christopher Schroers. A fully progressive approach to single-image super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 864–873, 2018.

[187] Yulun Zhang, Kunpeng Li, Kai Li, Lichen Wang, Bineng Zhong, and Yun Fu. Image super-resolution using very deep residual channel attention networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 286–301, 2018.

[188] Peter J. Brockwell and Richard A. Davis. *Introduction to Time Series and Forecasting*. Springer New York, 1996.

[189] Jingzhou Xin, Jianting Zhou, Simon Yang, Xiaoqing Li, and Yu Wang. Bridge structure deformation prediction based on GNSS data using Kalman – ARIMA – GARCH model. *Sensors (Basel, Switzerland)*, 18, 01 2018.

[190] X. Luo, M. Mayer, and B. Heck. Analysing time series of GNSS residuals by means of AR(I)MA processes. In *VII Hotine-Marussi Symposium on Mathematical Geodesy*, pages 129–134. Springer Berlin Heidelberg, 2012.

[191] Ping-Feng Pai and Chih-Sheng Lin. A hybrid ARIMA and support vector machines model in stock price forecasting. *Omega*, 33(6):497–505, 2005.

[192] Xiaoguang Luo. *GPS Stochastic Modelling - Signal Quality Measures and ARMA Processes.* Springer, 01 2013.

[193] José Lima and J Casaca. Smoothing GNSS Time Series with asymmetric simple moving averages. *Journal of Civil Engineering and Architecture*, 6, 06 2012.

[194] Changhui Jiang, Shuai Chen, Yuwei Chen, Boya Zhang, Ziyi Feng, Hui Zhou, and Yuming Bo. A MEMS IMU de-noising method using long short term memory recurrent neural networks (LSTM − RNN). *Sensors*, 18:3470, 10 2018.

[195] Stefano Gandolfi, Luca Poluzzi, and Luca Tavasci. Structural monitoring using GNSS technology and sequential filtering. In *Proceedings for FIG Working Week*, 05 2015.

[196] Y Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks / a publication of the IEEE Neural Networks Council*, 5:157–66, 02 1994.

[197] Sepp Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 6(2):107–116, 04 1998.

[198] Claudio Gallicchio. Short-Term Memory of Deep RNN. In *Proceedings for European Symposium on Artificial Neural Networks*, 02 2018.

[199] Alex Graves, Navdeep Jaitly, and Abdel-rahman Mohamed. Hybrid speech recognition with Deep Bidirectional LSTM. In *2013 IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU 2013 - Proceedings*, pages 273–278, 12 2013.

[200] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 38, 03 2013.

[201] Alex Graves. Generating sequences with recurrent neural networks. *ArXiv: 1308.0850*, 08 2013.

[202] Yuelei Xiao and Yang Yin. Hybrid LSTM neural network for short-term traffic flow prediction. *Information*, 10:105, 03 2019.

[203] Hee-Un Kim and Tae-Suk Bae. Deep learning-based GNSS network-based real-time kinematic improvement for autonomous ground vehicle navigation. *Journal of Sensors*, 2019:1–8, 03 2019.

[204] Elena Loli Piccolomini, Stefano Gandolfi, Luca Poluzzi, Luca Tavasci, Pasquale Cascarano, and Andrea Pascucci. Recurrent neural networks applied to gnss time series for denoising and prediction. In *26th International Symposium on Temporal Representation and Reasoning (TIME 2019)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2019.

[205] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 12 1997.

[206] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[207] Tom M Mitchell et al. Machine learning. 1997. *Burr Ridge, IL: McGraw Hill*, 45(37):870–877, 1997.

[208] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

[209] Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

[210] Marvin Minsky and Seymour Papert. *Perceptrons: An Introduction to Computational Geometry*. MIT Press, 1969.

[211] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[212] Raúl Rojas. *Neural networks: a systematic introduction*. Springer Science & Business Media, 2013.

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

[214] Matthew Hutson. Has artificial intelligence become alchemy? *Science*, 2018.

[215] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv preprint arXiv:1412.7062*, 2014.

[216] Vincent Dumoulin and Francesco Visin. A guide to convolution arithmetic for deep learning. *arXiv preprint arXiv:1603.07285*, 2016.