N°d'ordre NNT : 2021LYSEI113

THESE de DOCTORAT DE L'UNIVERSITE DE LYON
opérée au sein de
(Institut National des Sciences Appliquées, INSA - Lyon)

Ecole Doctorale N° 160
(ÉLECTRONIQUE, ÉLECTROTECHNIQUE, AUTOMATIQUE)

Spécialité/ discipline de doctorat :
Traitement du Signal et de l'Image

Soutenue publiquement le 14/12/2021, par :
Antonio Tomas Lorente Mur

Single-pixel imaging : compressed video acquisition and reconstruction using Deep learning

---

Devant le jury composé de :

Giovannelli, Jean-François	Professeur des universités Université de Bordeaux 1	Rapporteur
Arridge, Simon	Professeur des universités University College London	Rapporteur
Blu, Thierry	Professeur des universités The Chinese University of Hong Kong	Examinateur
Gigan, Sylvain	Professeur des universités Sorbonne	Examinateur
Peyrin, Françoise	Directrice de recherche, INSERM, Lyon	Directeur de thèse
Ducros, Nicolas	Maître de conférence, INSA, Lyon	Co-directeur de thèse

Département FEDORA – INSA Lyon - Ecoles Doctorales

SIGLE	ECOLE DOCTORALE	NOM ET COORDONNEES DU RESPONSABLE
CHIMIE	**CHIMIE DE LYON** **https://www.edchimie-lyon.fr** Sec. : Renée EL MELHEM Bât. Blaise PASCAL, 3e étage secretariat@edchimie-lyon.fr	**M. Stéphane DANIELE** C2P2-CPE LYON-UMR 5265 Bâtiment F308, BP 2077 43 Boulevard du 11 novembre 1918 69616 Villeurbanne **directeur@edchimie-lyon.fr**
E.E.A.	**ÉLECTRONIQUE, ÉLECTROTECHNIQUE, AUTOMATIQUE** **https://edeea.universite-lyon.fr** Sec. : Stéphanie CAUVIN Bâtiment Direction INSA Lyon Tél : 04.72.43.71.70 secretariat.edeea@insa-lyon.fr	**M. Philippe DELACHARTRE** INSA LYON Laboratoire CREATIS Bâtiment Blaise Pascal, 7 avenue Jean Capelle 69621 Villeurbanne CEDEX Tél : 04.72.43.88.63 **philippe.delachartre@insa-lyon.fr**
E2M2	**ÉVOLUTION, ÉCOSYSTÈME, MICROBIOLOGIE, MODÉLISATION** **http://e2m2.universite-lyon.fr** Sec. : Sylvie ROBERJOT Bât. Atrium, UCB Lyon 1 Tél : 04.72.44.83.62 secretariat.e2m2@univ-lyon1.fr	**M. Philippe NORMAND** Université Claude Bernard Lyon 1 UMR 5557 Lab. d'Ecologie Microbienne Bâtiment Mendel 43, boulevard du 11 Novembre 1918 69 622 Villeurbanne CEDEX **philippe.normand@univ-lyon1.fr**
EDISS	**INTERDISCIPLINAIRE SCIENCES-SANTÉ** **http://ediss.universite-lyon.fr** Sec. : Sylvie ROBERJOT Bât. Atrium, UCB Lyon 1 Tél : 04.72.44.83.62 secretariat.ediss@univ-lyon1.fr	**Mme Sylvie RICARD-BLUM** Institut de Chimie et Biochimie Moléculaires et Supramoléculaires (ICBMS) - UMR 5246 CNRS - Université Lyon 1 Bâtiment Raulin - 2ème étage Nord 43 Boulevard du 11 novembre 1918 69622 Villeurbanne Cedex Tél : +33(0)4 72 44 82 32 **sylvie.ricard-blum@univ-lyon1.fr**
INFOMATHS	**INFORMATIQUE ET MATHÉMATIQUES** **http://edinfomaths.universite-lyon.fr** Sec. : Renée EL MELHEM Bât. Blaise PASCAL, 3e étage Tél : 04.72.43.80.46 infomaths@univ-lyon1.fr	**M. Hamamache KHEDDOUCI** Université Claude Bernard Lyon 1 Bât. Nautibus 43, Boulevard du 11 novembre 1918 69 622 Villeurbanne Cedex France Tél : 04.72.44.83.69 **hamamache.kheddouci@univ-lyon1.fr**
Matériaux	**MATÉRIAUX DE LYON** **http://ed34.universite-lyon.fr** Sec. : Yann DE ORDENANA Tél : 04.72.18.62.44 yann.de-ordenana@ec-lyon.fr	**M. Stéphane BENAYOUN** Ecole Centrale de Lyon Laboratoire LTDS 36 avenue Guy de Collongue 69134 Ecully CEDEX Tél : 04.72.18.64.37 **stephane.benayoun@ec-lyon.fr**
MEGA	**MÉCANIQUE, ÉNERGÉTIQUE, GÉNIE CIVIL, ACOUSTIQUE** **http://edmega.universite-lyon.fr** Sec. : Stéphanie CAUVIN Tél : 04.72.43.71.70 Bâtiment Direction INSA Lyon mega@insa-lyon.fr	**M. Jocelyn BONJOUR** INSA Lyon Laboratoire CETHIL Bâtiment Sadi-Carnot 9, rue de la Physique 69621 Villeurbanne CEDEX **jocelyn.bonjour@insa-lyon.fr**
ScSo	**ScSo*** **https://edsciencessociales.universite-lyon.fr** Sec. : Mélina FAVETON INSA : J.Y. TOUSSAINT Tél : 04.78.69.77.79 melina.faveton@univ-lyon2.fr	**M. Christian MONTES** Université Lumière Lyon 2 86 Rue Pasteur 69365 Lyon CEDEX 07 **christian.montes@univ-lyon2.fr**

*ScSo : Histoire, Géographie, Aménagement, Urbanisme, Archéologie, Science politique, Sociologie, Anthropologie

INSA LYON

Résumé

University of Lyon

Doctor of Philosophy

Single-pixel real time video reconstruction via Deep learning

by Antonio Tomas Lorente Mur

La caméra mono-pixel est une caméra qui permet de faire l'acquisition d'images bi-dimensionnelles à partir d'un capteur ponctuel. Elle mesure au niveau du détecteur le produit scalaire d'image de la scène avec des fonctions définies par l'utilisateur. L'image est alors récupérée par le biais d'algorithmes de reconstruction dédiés. La caméra mono-pixel peut être utilisée dans des problèmes d'imagerie où il serait impossible d'utiliser des méthodes d'imagerie conventionnelle par matrice de capteurs. En particulier, la caméra mono-pixel peut être couplée avec un spectromètre pour en faire une caméra hyperspectrale. De telles caméras mono-pixel permettent notamment de réaliser l'analyse de la signature spectrale de certaines molécules.

La principale limitation de l'imagerie mono-pixel sont les temps d'acquisition et de reconstruction, qui sont trop lents pour l'application en temps réel. L'objectif de cette thèse est la réalisation d'algorithmes de reconstruction et de sous-échantillonage pour permettre l'acquisition et reconstruction d'images à hautes fréquences.

Dans cette thèse nous avons étudié l'usage d'algorithmes d'apprentissage profond en imagerie mono-pixel. Plus concrètement, nous avons introduit l'usage de solutions de régularisation généralisée de Tikhonov dans des reconstructeurs par réseaux de neurones dans le but de les appliquer à des données expérimentales. Dans un deuxième temps, nous avons développé des architectures de réseaux neuronaux qui combinent des réseaux neuronaux avec l'algorithme d'espérance-maximisation afin d'estimer le maximum à posteriori de notre problème inverse. Enfin, nous avons étudié des schémas de sous-échantillonage évoluant au cours du temps selon l'évolution prédite au cours du temps de la variance. Nous avons combiné ce sous-échantillonage avec des schémas de reconstruction prenant en compte les frames reconstruites précédemment pour estimer la frame actuelle.

Par rapport aux approches classiques, notre approche permet l'acquisition et la reconstruction en temps réel, avec une cadence de 10 images par seconde. Enfin, nous avons montré l'application des méthodes proposées à des données issues d'une caméra mono-pixel hyperspectrale.

INSA LYON

Abstract

University of Lyon

Doctor of Philosophy

Single-pixel real time video reconstruction via Deep learning

by Antonio Tomas Lorente Mur

Single-pixel cameras allow acquisition of two-dimensional images using only a single-point detector. These use hardware to measure the inner product of the scene under view, along with the user-defined functions, and then the image is recovered using reconstruction algorithms. Single-pixel cameras are well suited for imaging problems where it might not be possible to apply traditional arrays of detectors. In particular, single-pixel cameras can be coupled with a spectrometer, to provide hyperspectral cameras. Such single-pixel hyperspectral cameras can be used, for instance, to analyze spectral signatures of certain molecules.

The main limitations of single-pixel imaging are the speeds of acquisition and reconstruction, as these are both too slow for real-time imaging applications. The aim of this project was to design reconstruction algorithms and down-sampling strategies to enable high-frequency image acquisition and reconstruction for single-pixel cameras.

The study described in this thesis investigated the use of deep-learning algorithms for single-pixel imaging. More precisely, we focused on introducing generalized Tikhonov-regularized solutions within deep-learned reconstructors for experimental data. Secondly, we worked on neural-network architectures that can combine neural networks, and the expectation-maximization algorithm to estimate the maximum a posteriori of our inverse problem. Finally, we investigated time-adaptive down-sampling algorithms based on projected variance estimates over time. To reconstruct a current frame, we combined these temporal down-sampling algorithms with reconstruction approaches that take into account previous reconstructed frames.

With respect to classic approaches, our approach demonstrates real-time acquisition and reconstruction times that can provide videos with 10 frames per second. Finally, we demonstrate the applicability of these approaches to experimental data from a single-pixel hyperspectral camera.

Remerciements

Je voudrais dans un premier temps remercier mes encadrants de thèse, Françoise Peyrin et Nicolas Ducros. Ils m'ont accueilli au laboratoire lors de mon stage de master, et m'ont ensuite permis de poursuivre l'aventure de la thèse à leur côté.

J'exprime ma gratitude à Françoise pour accepter de devenir directrice de thèse. Son expérience, sa patience et sa capacité à prendre du recul a sans aucun doute participé grandement à la réussite de cette thèse. Les points thèse mensuels étaient précieux pour se rassurer aux moments de doute et recentrer la thèse sur les questions importantes.

Je tiens à remercier Nicolas pour avoir été là au quotidien à CREATIS. Son enthousiasme, sa curiosité scientifique et sa rigueur ont été d'une grande aide tout au cours de la thèse. Nos échanges réguliers ont grandement contribué aux développements proposés pendant cette thèse. Grâce à sa bonne humeur, il aura en particulier su me motiver lors des moments de vide et des coups de mou qui sont arrivés pendant la thèse. Il m'a également permis de faire des heures en tant qu'enseignant vacataire au département de GE de l'INSA.

Je remercie également Olivier Beuf. En tant que directeur de CREATIS, il a fourni un environnement de travail accueillant et propice au travail.

J'exprime mes remerciements à tous les membres qui ont accepté de participer à mon jury de thèse. En particulier Jean-François Giovanelli et Simon Arridge m'ont fait l'honneur d'être les rapporteurs de mes travaux de thèse. Je remercie également Thierry Blu d'avoir assumé la présidence de mon jury, et Sylvain Gigan d'avoir fait le déplacement malgré la période de pandémie.

Je remercie tous les membres du laboratoire CREATIS avec lesquels j'ai pu échanger. Leur bonne humeur et bienveillance pendant la thèse ont été fortement appréciés. La bonne ambiance qu'on peut retrouver dans la salle de pause est l'une des raisons principales pour lesquelles ce labo est le meilleur!

Je remercie tous ceux qui travaillent ou ont travaillé avec moi au niveau de la manip (Bruno, Laurent, Pierre (Leclerc), Guilherme) ou au niveau de la toolbox spyrit (Sebastien, Thomas, Claire).

Je remercie les différents camarades bureau que j'ai pu avoir. Je remercie donc mes camarades du bureau des stagiaires : Suzanne, Valeriya, Maxime, Eloise et Sixiang. En particulier Suzanne m'a aidé tout au long de la thèse avec les procédures administratives. La présence de Valeriya a également fortement impacté ma thèse puisqu'on a eu des échanges très productifs, et on a en particulier rédigé un article de journal ensemble.

Un grand merci à mes camarades du bureau des doctorants à Blaise Pascal : Kenny, Goulven, Florian, Yanis, Maryam, Nicolas (Loiseau) et Ludmilla. Ainsi que mes camarades du bureau des doctorants à Léonard de Vinci : Maxime (encore lui!), Nina, Anne-Lise, Yunlong, et les petits nouveaux Nolann, Mohammed et Flora. Ils auront tous eu la patience de me supporter d'une façon ou d'une autre.

Je remercie également tous ceux avec qui j'ai pu échanger régulièrement en salle de pause pendant ces trois années et demi : Anthony, Fabienne, Marion, Frank, Nicolas (Pinon), Audrey, Sophie, Charles, Sofiane, Mathieu, Tom, Pierre-Jean, Pierre, Louise,

Elisabeth, François et Juan. Il est toujours bon de prendre une pause et discuter sur des thèmes scientifiques (ou non) pour se rafraîchir les idées.

Je remercie également les intervenants du département de GE qui m'ont proposé de faire des heures en tant qu'enseignant vacataire. En particulier, je remercie (encore une fois) Nicolas, ainsi que Thomas Grenier, Pierre-Jean Viverge et Jean-François Mogniotte.

Je remercie mes amis de Centrale Lyon avec lesquels je me suis réuni régulièrement pour discuter, découvrir des restos et m'amuser : Lucas, Thierry et Antoine. Merci pour ces bons moments! Et je remercie également mes amis de Centrale avec lesquels on organise régulièrement des séances de jeux en ligne : Thomas, Natan, Hugues, Thibaut, Clément.

Enfin, je souhaite exprimer le plus grand des remerciements à ma famille qui a toujours été à mes côtés. Ma mère, mon père et ma soeur ont sans aucun doute été les personnes qui m'ont le plus épaulé pendant cette période de thèse. Je vous aime très fort.

Contents

List of Figures

List of Tables

List of Symbols

Symbols

$\boldsymbol{f}$	Ground truth image
$\boldsymbol{m}$	Measurements
N	Number of pixels in the ground truth image
M	Number of acquired measurements
$\boldsymbol{S}$	Downsampling matrix
$\boldsymbol{H}$	2-D Hadamard transform matrix
α	Intensity of the image of the scene (in photons)
μ_{dark}	Dark current mean
σ_{dark}	Dark current standard deviation
K	Photodetector gain

Linear Algebra

$\mathbb{R}$	Set of real numbers
$\mathbb{R}^N$	Real n-space
$\mathbb{R}^{M\times N}$	Space of M-by-N real valued matrices
$\boldsymbol{x}$	Vector x
$\boldsymbol{x}^\top$	Transpose of vector x
$\boldsymbol{X}$	Matrix x
$\mathbf{x}$	Multidimensional random variable x
$\|\boldsymbol{x}\|_{\boldsymbol{A}}^2 = \boldsymbol{x}^\top \boldsymbol{A}\boldsymbol{x}$	weighed ℓ_2 norm ($\boldsymbol{A}$ positive definite)
x_i	i^{th} component of vector $\boldsymbol{x}$
X_{ij}	j^{th} component of the i^{th} row of matrix $\boldsymbol{X}$

Time series

$\boldsymbol{f}_{1:t} = (\boldsymbol{f}_1, \ldots, \boldsymbol{f}_t)$	frames 1 to t of the ground truth video
$\boldsymbol{m}_{1:t} = (\boldsymbol{m}_1, \ldots, \boldsymbol{m}_t)$	measurements acquired from frames 1 to t
M^t	Number of acquired measurements from the t^{th} frame

Random variables

$\mathbf{f}$	Random variable from which $\boldsymbol{f}$ is drawn
$\mathbf{m}$	Random variable from which $\boldsymbol{m}$ is drawn
$p(\boldsymbol{f})$	Probability density function of $\mathbf{f}$ [when it is not ambiguous, we define the probability density functions by the arguments in the function alone]
$p(\boldsymbol{f}\|\boldsymbol{m})$	Marginal probability density function of $\boldsymbol{f}$ given $\boldsymbol{m}$
$\mathbf{f}\|\boldsymbol{m}$	Conditional random variable $\mathbf{f}$ given $\mathbf{m} = \boldsymbol{m}$
$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$	Multidimensional normal distribution with mean $\boldsymbol{\mu}$, and covariance $\boldsymbol{\Sigma}$
$\mathcal{P}(\lambda)$	Poisson distribution with mean $\lambda > 0$
$\mathcal{P}(\boldsymbol{\lambda})$	Multidimensional Poisson distribution - every component i of the vector follows an independent Poisson distribution with mean $\lambda_i > 0$
$\mathcal{S}_k(\boldsymbol{\mu_1}, \boldsymbol{\mu_2})$	Multidimensional Skellam distribution obtained as the difference of $\mathcal{P}(\boldsymbol{\mu}_1)$ and $\mathcal{P}(\boldsymbol{\mu}_2)$
$\mathbb{E}(\mathbf{f}\|\boldsymbol{m})$	Conditional expectation of $\mathbf{f}$ given $\mathbf{m} = \boldsymbol{m}$
$\mathbb{V}\text{ar}(\mathbf{f})$	Variance of $\mathbf{f}$
$\mathbb{C}\text{ov}(\mathbf{f})$	Covariance of $\mathbf{f}$

Introduction

Single-pixel imaging allows the reconstruction of two-dimensional images from a single-point detector. This technology has historically been used in imaging modalities where each single detector is either too expensive or too cumbersome to allow for arrays of detectors. The camera acquires a compressed version of the image of the scene via the hardware: a spatial light modulator modulates the image of the scene, which we then focus into a single-point detector via a system of lenses. By sequentially loading a series of masks into the spatial light modulator, we can therefore acquire a sequence of measurements from which we wish to recover the original image of the scene.

The idea of a single-pixel camera has been credited to a research group at Rice University (Houston, TX, USA) (Duarte et al., 2008). Their initial set-up was used as hardware implementation of the principles of compressive sensing (Donoho, 2006) that recovered the image through an ℓ_1-minimization algorithm from down-sampled measurements. This down-sampling at sub-Nyquist frequencies leads to under-determined inverse problems, but opens the door to real-time acquisition of single-pixel images.

While the idea of single-pixel imaging might be unappealing for conventional imaging techniques that rely on arrays of sensors (such as CCD or CMOS cameras), they are particularly interesting for modalities where a single sensor is either too expensive or too cumbersome to be displayed in an array. As such they have been used for infrared imaging (Radwell et al., 2014), hyperspectral imaging (Magalhaes et al., 2012), and microscopy (Radwell et al., 2014; Rodríguez et al., 2016), among others.

The initial compressed-sensing-based single-pixel imaging allowed for fast acquisition. However, the time-consuming ℓ_1-minimization algorithm made real-time image reconstruction challenging. The advent of deep-learning algorithms for image reconstruction inverse problems (Arridge et al., 2019) has, in particular, opened the path for real-time image reconstruction (Higham et al., 2018).

The goal of this thesis is to investigate strategies for fast single-pixel image reconstruction. Furthermore, we lack a framework for single-pixel imaging where the temporal redundancy of images is exploited to avoid acquiring the coefficients that do not vary across time. Such a framework would allow for a further reduction in the number of measurements required to acquire images for real-time video acquisition.

To meet the goals of this thesis, we developed algorithms based on deep-learning, to reconstruct images from experimental data from a single-pixel camera. One of the main limitations of deep-learning algorithms are their lack of interpretability, which make them harder to apply to experimental data. More precisely, we proposed a mapping from the measurement domain to the image domain based on Tikhonov regularisation. This mapping is then corrected by a deep-neural network. The applicability of this mapping is demonstrated for grayscale and hyperspectral measurements. Furthermore, we proposed an interpretable deep-learning architecture based on the expectation maximization algorithm (Dempster, Laird, and Rubin, 1977). Finally, we explore the framework of Kalman filtering for single-pixel video reconstruction, to exploit the temporal redundancy of natural videos. We further propose a time-adaptive down-sampling scheme alongside with

This thesis was performed as part of the ARMONI ANR (Ducros, 2017) project that aims to develop hyperspectral imaging tools for image-guided surgery. As such, the application of the single-pixel camera is aimed at hyperspectral imaging. The algorithms and methodologies were conducted in the CREATIS laboratory. Most of our methods have been tested on experimental data from the hyperspectral single-pixel camera prototype developed at Pilot platform of CREATIS.

The first four chapters explore the state of the art, and show the key concepts used during the final four chapters that explain our contributions.

In Chapter 1, the concept of single-pixel imaging is explained, with all its challenges. We further explain the core choices made during this thesis to tackle the aforementioned challenges.

In Chapter 2, we explain the core concepts of Bayesian approaches to solve inverse problems. These concepts are at the heart of single-pixel image reconstruction.

In Chapter 3, we present deep-learning and the theory behind it, alongside the more practical aspects of deep-learning.

In Chapter 4, we present the basic concepts of recursive Bayesian state estimation that can be used to recover signals over time from a series of noisy measurements.

In Chapter 5, we focus on the use of Tikhonov-based mapping for deep-learning reconstruction. This work is based on an article published in *Optics Express*, and aims, in particular, to set the framework to apply deep neural networks to data from an experimental single-pixel camera set-up.

In Chapter 6, we further develop the concepts of chapter 5, by applying the Tikhonov-based mapping for hyperspectral data combined with higher dimension convolutions. This work is based on an article submitted to *Optics Express*, and it is the result of a joint collaboration with V. Pronina from Skolkovo Institute of Science and Technology.

In Chapter 7, we propose expectation-maximisation-based deep-learning architecture for single-pixel image reconstruction. This neural network offers a highly interpretable framework for the deep-learning-based reconstruction. This neural network is specifically designed to tackle measurements corrupted by approximately normally distributed signal-dependent noise. This work is based on a conference proceeding in *IEEE ISBI 2021* further developed in an article submitted to *IEEE Transactions on Computational Imaging*.

In Chapter 8, we explore the use of recursive state estimation for single-pixel video reconstruction. Based on this, we propose a variance-based time-adaptive down-sampling scheme. Based on this down-sampling scheme, we show that we can reduce the number of acquired measurements for video reconstruction. We further present some of the perspectives presented by the single-pixel video reconstruction problem.

Part I

State of the art

Chapter 1

Single-pixel imaging: concept and applications

Most traditional cameras acquire images through arrays of photo-detectors. On the other hand, single-pixel cameras are designed to recover images with a single-point detector. In this chapter, we aim to cover the basic concepts of single-pixel imaging. This encompasses the hardware implementation, design of the mask patterns, and the mathematical model behind the single-pixel camera, along with an explanation of the choices made that will impact upon the rest of this manuscript.

1.1 Basic concepts of single-pixel imaging

1.1.1 System description

While many hardware implementations of single-pixel imaging systems have been proposed throughout the years (Gibson, Johnson, and Padgett, 2020; Edgar, Gibson, and Padgett, 2019), the principles are still the same in most cases. Depending on the applications, some technologies might be better suited than others. This section covers the conventional single-pixel camera set-up that is illustrated in Fig. 1.1.

The common components for all single-pixel set-ups are a set of lenses, a spatial light modulator (SLM), and a single-point photo-detector.

The image of the object $\boldsymbol{f} \in \mathbb{R}^N$ is formed in the SLM plane. The SLM modulates the image of the object through user-defined patterns $\boldsymbol{h}_i \in \mathbb{R}^N$ (e.g., in Fig. 1.1 the image is modulated by a Hadamard basis function). A lens then focuses the output of the SLM into the single-point detector. The numerical data is then digitized via a numerical converter, to acquire the inner product $\langle \boldsymbol{f}, \boldsymbol{h}_i \rangle$ of the object to be acquired $\boldsymbol{f}$, and the pattern on the SLM $\boldsymbol{h}_i$. When acquiring M sequential measurements, the measurement vector $\boldsymbol{m} \in \mathbb{R}^M$ is said to acquire the product

$$\boldsymbol{m} = \boldsymbol{H}\boldsymbol{f}, \tag{1.1}$$

with $\boldsymbol{H} = [\boldsymbol{h}_1, \ldots, \boldsymbol{h}_m]^\top \in \mathbb{R}^{M \times N}$.

The SLM chosen for single-pixel imaging is usually a digital micromirror device (DMD) (Duarte et al., 2008; Takhar et al., 2006), as this offers a good compromise between the refresh rate ($\approx$ 32 kHz), the number of pixels (>1000 pixels in resolution in both dimensions), and the active area ($\approx$ mm^2). The idea behind DMDs was initially patented by Nathalson (Nathanson, US patent 3746911, 1973-7-17), and it consists of an array matrix with thousands of small mirrors. Each of the mirrors can be independently tilted into two positions: the so-called ON state, and the OFF state. The ON state corresponds to a +12° tilt, and the mirrors reflect the light towards the detector. The OFF state, on the other hand, corresponds to a -12° tilt and does not reflect the light towards the detector. Every mirror in the ON state represents

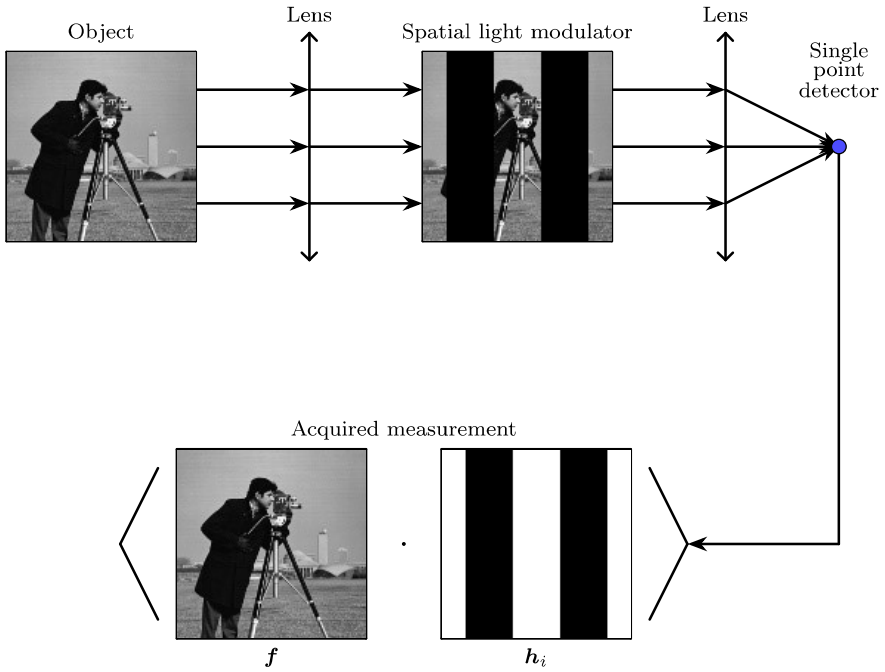


FIGURE 1.1 – The principle of the single-pixel camera. The measurements acquired are the inner product between the object to be acquired and the user-defined patterns on the spatial light modulator.

a '1', and every mirror in the OFF state represents a '0'. The option to implement gray scale patterns also exists, and most DMDs offer the option to display 8-bit gray-level patterns. The 8-bit digits of the gray-scale patterns $\boldsymbol{h}_i$ are separated into the corresponding 8 binary bit planes,

$$\boldsymbol{h}_i = \sum_{j=1}^{8} 2^j - 1 \boldsymbol{h}_{i,j} \tag{1.2}$$

and for every bit state $\boldsymbol{h}_{i,j}$, the display time is increased by the corresponding multiple of 2, to produce the resulting measurement

$$m_i = \sum_{j=1}^{8} 2^j - 1 \langle \boldsymbol{f}, \boldsymbol{h}_{i,j} \rangle. \tag{1.3}$$

The nature of the single-point photo-detector depends on the application design for the single-pixel camera. The original design by Duarte used a silicon-based photo-detector. Since then, many variants have been applied to suit the needs of the application. Some examples include infrared detectors, spectrometers, and TeraHertz detectors.

1.1.2 Applications

The advantages and versatility of the single-pixel camera have lead to its application to many imaging problems. Some examples of single-pixel camera applications are shown in Fig. 1.2.

Infrared imaging (Radwell et al., 2014) designed a microscope based on the single-pixel camera that they used to image in the infrared and visible spectrum. To achieve

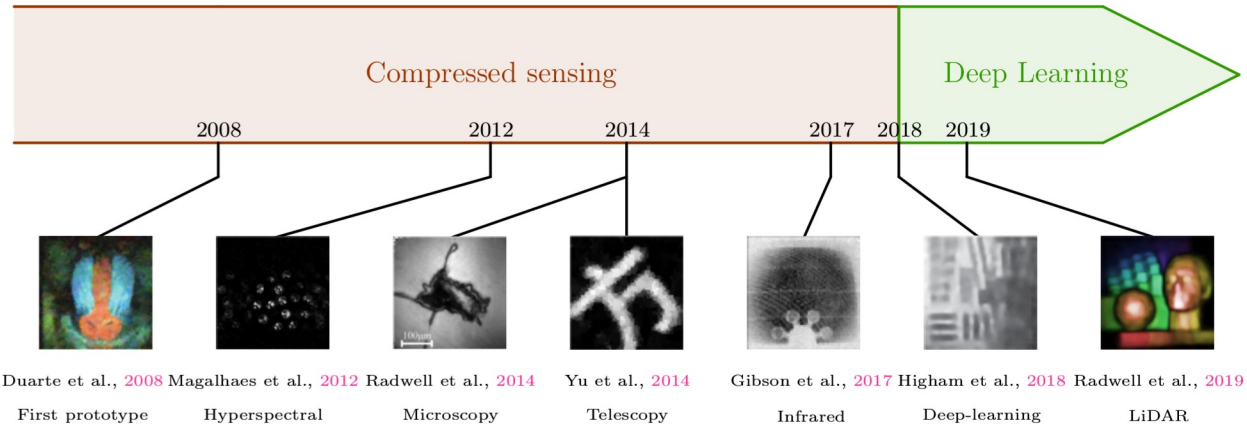


FIGURE 1.2 – Timeline of single-pixel imaging hardware and software developments. Publications are shown by year, and show the reconstruction method used alongside the imaging application.

such an imager, an infrared detector and a visible-light detector was used to gather the light from the ON and OFF states of the DMD. This has further been used to image methane gas leaks (Gibson et al., 2017), demonstrating the potential of single-pixel imaging in infrared imaging.

Telescopic imaging In (Yu et al., 2014) a single-pixel-camera-based telescope was used to view an object from 2 km away. Their set-up had a large field of view over large distances, and showed promising results for remote-target imaging.

Light detection and ranging imaging (Radwell et al., 2019) combined a single-pixel camera with a light detection and ranging (LiDAR) system that allowed them to recover three-dimensional (3D) maps of depth over long distances, which demonstrated the potential applicability of single-pixel cameras for self-driving cars, and several other robotics industries.

Microscopy The single-pixel camera has been used for many types of microscopy imaging. Their imaging at wavelengths where silicon-based detectors are blind has made them widely popular in non-visible-light microscopy (Radwell et al., 2014). In (Wu et al., 2021), a beam splitter was used for microscopy holographic imaging of biological tissues. (Rodríguez et al., 2016) proposed a dual-mode inverted microscope to exploit reflection and transmission information. This was further used for Fourier microscopy (Peng et al., 2021), where the Fourier spectrum was captured in the back focal plane of the objective.

Hyper-spectral imaging (Magalhaes et al., 2012) used a hyperspectral imaging system by replacing the punctual detector with a spectral analyser with spectral resolution of 10 ppm. This gave birth to hyperspectral single-pixel imaging, which is a flourishing research field due to its low price and its production of two-dimensional (2D) images with spectral resolutions that other hyperspectral imagers cannot attain. This is particularly important for biomedical applications, where high spectral resolution imagers can be used to analyze spectral signatures of certain molecules.

1.2 Advanced single-pixel imaging concepts

1.2.1 Compressed sensing

The ideas behind single-pixel imaging came to light inspired by the theory of compressed sensing (Donoho, 2006). This means that the single-pixel camera was designed to acquire a down-sampled version of the image subject to some noise

$$\boldsymbol{m} = \boldsymbol{H}\boldsymbol{f} + \boldsymbol{\epsilon}, \tag{1.4}$$

where $\boldsymbol{H} \in \mathbb{R}^{M\times N}$ is the matrix that contains the M chosen patterns, and $\boldsymbol{\epsilon} \in \mathbb{R}^M$ is random additive noise. This time however, the purpose is to reconstruct the image with $M \ll N$ measurements. This ensures faster acquisition rates, and reduced memory requirements.

The concept of single-pixel was first introduced by (Takhar et al., 2006), and was later perfected by (Duarte et al., 2008), who provided hardware implementation compressed sensing.

Compressed sensing relies on a transform operator $\boldsymbol{\Lambda} \in \mathbb{R}^{N\times N}$ that sparcifies $\boldsymbol{f}$

$$\boldsymbol{f} = \boldsymbol{\Lambda}\boldsymbol{s}, \tag{1.5}$$

where $\boldsymbol{s} \in \mathbb{R}^N$ is L-sparse (i.e., has L non-zero components).

Under the theory of compressed sensing, if the matrix $\boldsymbol{H}$ satisfies the so-called restricted isometry property (Donoho, 2006), then the original image can be perfectly recovered. The image is then recovered by solving the following inverse problem:

$$\hat{\boldsymbol{s}} = \underset{\boldsymbol{s}\in\mathbb{R}^N}{\operatorname{argmin}} \|\boldsymbol{s}\|_1 \quad \text{such that } \boldsymbol{H}\boldsymbol{\Lambda}\boldsymbol{s} = \boldsymbol{m}. \tag{1.6}$$

Matrices that verify the restricted isometry condition with a high likelihood can be drawn from a Bernoulli distribution,

$$(\boldsymbol{H})_{i,j} \sim \mathcal{B}(0, 1/2), \tag{1.7}$$

provided that

$$M \geq cL\log(\frac{D}{L}). \tag{1.8}$$

for some values of c (typically in the range of $[1; 100]$ (Donoho, 2006; J. Candès, K. Romberg, and Tao, 2006; Takhar et al., 2006)). This choice is binary, and therefore it is well suited to be implemented on a SLM.

This approach allows for a perfect reconstruction of the sparse signal, assuming noiseless acquisitions. Under noisy conditions, the problem to solve is

$$\hat{\boldsymbol{s}} = \underset{\boldsymbol{s}\in\mathbb{R}^N}{\operatorname{argmin}} \|\boldsymbol{s}\|_1 \quad \text{subject to } \|\boldsymbol{H}\boldsymbol{\Lambda}\boldsymbol{s} - \boldsymbol{m}\|_2 \leq \|\boldsymbol{\epsilon}\|_2. \tag{1.9}$$

The recovery error can then be bounded above by $C_0\|\boldsymbol{\epsilon}\|_2$, where C_0 is a constant.

A difficulty in compressed sensing is finding the sparcifying transform $\boldsymbol{\Lambda}$. In practice most approaches use the wavelet (Mallat, 1999) basis, or learned dictionaries. Neither of these approaches manages to get a perfect sparse representation of all natural images, which often led to reconstructions with artifacts for high compression rates. A popular alternative to the ℓ_1-minimization algorithm was the total-variation solution (Li, 2010), which is obtained by solving

$$\hat{\boldsymbol{f}} = \underset{\boldsymbol{f}\in\mathbb{R}^N}{\operatorname{argmin}} \|\boldsymbol{f}\|_{\mathrm{TV}} \quad \text{such that } \boldsymbol{H}\boldsymbol{f} = \boldsymbol{m}, \tag{1.10}$$

which corresponds to choosing $\boldsymbol{\Lambda}$ as the discrete gradient operator.

1.2.2 Deep learning

(Higham et al., 2018) proposed a deep-learning framework for single-pixel imaging. This framework paved the way for real-time single-pixel video reconstruction. The main ideas were to consider an auto-encoder structure where the encoding part was modeled by a matrix of binary patterns $\boldsymbol{H}$, alongside a deep neural reconstructor $\mathcal{G}_\theta$, such that

$$\hat{\boldsymbol{f}} \approx \mathcal{G}_\theta(\boldsymbol{H}\boldsymbol{f}). \tag{1.11}$$

The neural network is trained to minimize

$$\boldsymbol{\theta}, \boldsymbol{H} = \underset{\boldsymbol{\theta}^*, \boldsymbol{H}^*}{\operatorname{argmin}} \frac{1}{S} \sum_{j=1}^{S} \|\mathcal{G}_{\theta^*}(\boldsymbol{H}^* \boldsymbol{f}^{(i)}) - \boldsymbol{f}^{(i)}\|_2^2 + \beta(\|\boldsymbol{H}^* - \mathbf{1}\|_2^2 + \|\boldsymbol{H}^* + \mathbf{1}\|_2^2), \tag{1.12}$$

where $\{\boldsymbol{f}^{(i)}\}_{1 \leq i \leq S}$ are image samples from the STL-10 (Coates, Ng, and Lee, 2011) image dataset, and β is a regularization parameter that balances the loss, to ensure that the patterns are binary, while trying to minimize the mean-squared error. This framework not only learned the measurement matrix $\boldsymbol{H}$ as a linear fully connected layer, but it also offered a fast reconstructor for single-pixel measurements.

1.2.3 Down-sampling from an analytical basis

1.2.3.1 Analytical basis functions

While the methods that we have presented so far used patterns specifically designed with the reconstruction algorithm in mind, a popular alternative is to make the down-sampled acquisitions on an analytical orthogonal basis, such as the Hadamard, Fourier, or wavelet basis. These alternatives might be preferable, as they allow for higher flexibility in the choice of our down-sampling schemes and reconstruction algorithms. Furthermore, they also make full use of the properties of the analytical basis, which allows for easier understanding of the reconstruction algorithms.

The choice of the patterns implemented in the DMD determines the down-sampling ratio at which the single-pixel camera can operate, the sensitivity to additive noise, and the photon-based Poisson noise, and it determines the operating speed of the different components of the single-pixel camera. Real-time single-pixel imaging can only be attained if the pattern choice tackles all of these aspects adequately.

Hadamard patterns The Hadamard transform is a popular pattern choice (Yu et al., 2020a). It has been used for Hadamard transform optics (Sloane and Harwit, 1976), and helped to pave the way to single-pixel optics. One of the main advantages of Hadamard patterns is that they reduce the variance of additive noise by $1/N$ when applying the Moore-Penrose pseudo-inverse (Penrose, 1955)(this is the highest reduction we can obtain with real-valued patterns between $[-1; 1]$)(Hadamard, 1893). One dimensional Hadamard matrices are typically constructed iteratively over powers of 2, through the Sylvester construction

$$\boldsymbol{H}_1^{(1)} = (1)\,, \tag{1.13}$$

$$\boldsymbol{H}_2^{(1)} = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{1.14}$$

$$\boldsymbol{H}_{2^k}^{(1)} = \begin{pmatrix} \boldsymbol{H}_{2^{k-1}}^{(1)} & \boldsymbol{H}_{2^{k-1}}^{(1)} \\ \boldsymbol{H}_{2^{k-1}}^{(1)} & -\boldsymbol{H}_{2^{k-1}}^{(1)} \end{pmatrix} \quad \forall k > 0. \tag{1.15}$$

The 2D Hadamard transform $\boldsymbol{H}_n^{(2)} \in \mathbb{R}^{n^2 \times n^2}$ is based on the one-dimensional (1D) transform $\boldsymbol{H}_n^{(1)} \in \mathbb{R}^{n \times n}$. It applies the 1D Hadamard transform to the lines and columns of an image. Using the Kronecker product, indicated as $\otimes$, we obtain

$$\boldsymbol{H}_n^{(2)} = \boldsymbol{H}_n^{(1)} \otimes \boldsymbol{H}_n^{(1)}. \tag{1.16}$$

The Sylvester construction guarantees binary elements all across the matrix, as well as the essential property of Hadamard matrices that

$$|\det(\boldsymbol{H}_n)| = n^{\frac{n}{2}}. \tag{1.17}$$

Fourier patterns The Fourier transform has also been widely used for single-pixel imaging (Zhang et al., 2017d). The Fourier transform is known for its high compression, where the coefficients of natural images drop in energy with a speed of $1/\omega^2$, and where ω is the spacial frequency in the Fourier domain (van der Schaaf and van Hateren, 1996). Like the Hadamard patterns, the Fourier patterns reduce the original covariance of any additive noise by a factor of $1/N$.

With respect to Fourier transform implementation, the gray scale patterns have for a long time limited the acquisition speed due to the DMD mirror flip frequency of 256 Hz for gray-scale patterns. To solve this issue, the Fourier patterns can mostly be binarized through the hardware implementation, which consists in considering for small images the Fourier basis functions at a greater size, then binarizing these via dithering algorithms (Zhang et al., 2017c).

Wavelet patterns Wavelet basis functions have been widely used for many image-processing tasks, as they are well known for their compression. They were widely used for JPEG-2000 compression standards (Mallat, 1999). Their high compression rate, in particular, made the use of wavelets all the more popular, alongside their multi-scale approach (Rousset et al., 2017; Czajkowski, Pastuszczak, and Kotyński, 2018).

Their main drawback is that wavelets at high resolution levels produce patterns that select few photons from the scene. This is an issue when faced with Poisson noise, as the lower the amount of photons, the higher the noise.

Pattern splitting The implementation of many patterns needs to implement negative values in the DMD, and this is typically done by dividing the transform matrix $\boldsymbol{H}$ into positive and negative values, such that

$$\boldsymbol{H} = \boldsymbol{H}^+ - \boldsymbol{H}^-, \tag{1.18}$$

$$(\boldsymbol{H}^+)_{i,j} = \max((\boldsymbol{H})_{i,j}, 0), \quad \forall 0 \leq i,j \leq N, \tag{1.19}$$

$$(\boldsymbol{H}^-)_{i,j} = -\min((\boldsymbol{H})_{i,j}, 0), \quad \forall 0 \leq i,j \leq N. \tag{1.20}$$

This implementation means that to fully sample the image of N pixels, 2N measurements are required. This issue has, however, been widely solved by recovering the measurement from the OFF state of the DMD (Czajkowski, Pastuszczak, and Kotyński, 2018).

Other alternatives can also be imaged on a different basis, based on the specific properties. For instance, for the Fourier basis, the three-phase shifting algorithm (Creath, 1988; Zhang et al., 2017b) that is widely used in fringe analysis allows the nonpositivity and complex nature of our measurements to be dealt with. A more general alternative that includes semi-positive factorization methods has been considered. These require semi-negative matrix factorization, which can be computationally expensive. More details can be found in (Rousset, Peyrin, and Ducros, 2018) and (Lorente Mur et al., 2019).

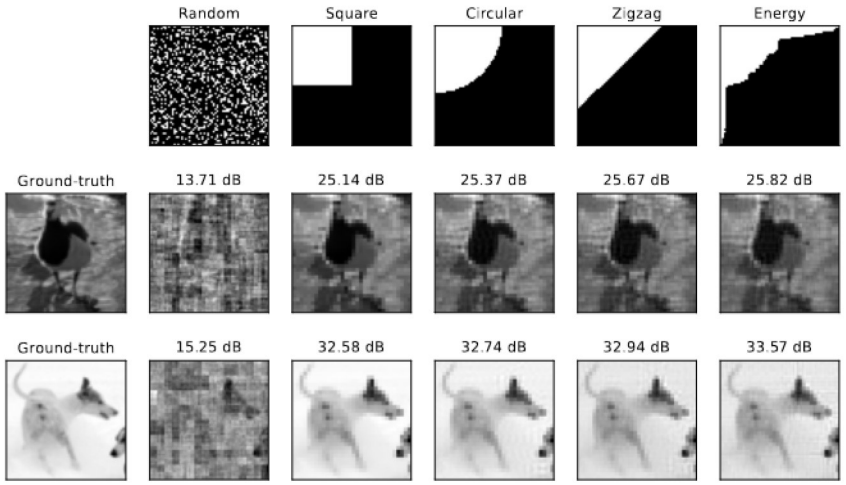


FIGURE 1.3 – Reconstructions from different down-sampling strategies on the Hadamard basis. The top row shows the coefficients acquired in the Hadamard domain to the images (Energy is the solution of 1.22). $\boldsymbol{H}$ is the Walsh-Hadamard basis. The images have $N = 64 \times 64$ pixels, and we kept $M = \frac{1}{4}N$ coefficients. The reconstructed images were obtained using the Moore-Penrose pseudo-inverse (Penrose, 1955). The signal-to-noise ratio of the reconstructed image is shown above each reconstruction.

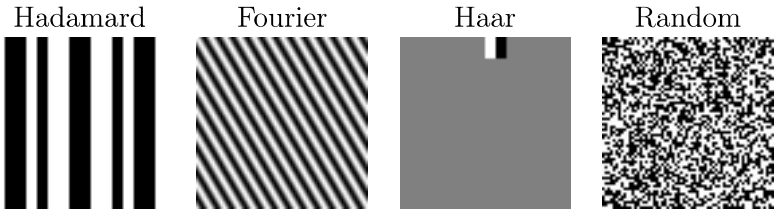
1.2.3.2 Down-sampling on an analytical basis

In the case of Hadamard, Fourier, or wavelet bases, one question remains: how to decide on the down-sampling mask? We aim to acquire the coefficients with highest energy, while ignoring those with lowest energy. The ideal down-sampling scheme would vary according to the acquired image, but the image is unknown prior to making the acquisition. Therefore, the down-sampling strategies can be based on the knowledge of each basis, and their properties when applied to natural images.

We will briefly explore some of the most popular down-sampling strategies for single-pixel acquisition, which are illustrated in Fig. 1.3.

Low frequency patterns The Fourier transform of natural images is known to decrease with a speed of $1/\omega^2$ on the spacial frequency ω domain (van der Schaaf and van Hateren, 1996). Therefore one of the most popular down-sampling strategies consists of using patterns that preserve only the low frequency coefficients. As shown in Fig. 1.3, low-frequency subsampling strategies can include keeping the frequencies in a circular, square, or zigzag shape in the transform domain (Zhang et al., 2017d).

Statistical sampling Assuming that natural images stem from a probability density function $p(\boldsymbol{f})$. Assuming that we have access to $(\mathbf{f}_1, \ldots, \mathbf{f}_S)$, S samples drawn from the same probability distribution as $\mathbf{f}$, the aim is to find the M indices selected through the matrix $\boldsymbol{H}$ that captures, on average, the highest proportion of the signal.



	Hadamard	Fourier	Haar	Random
Compression	++	+++	++++	++
η	$1/N$	$1/N$	$1/4$	N/A
γ	$1/2$	$1/2$	$1/\log_2(N)$	$1/2$
Binary	Yes	No	Yes	Yes

TABLE 1.1 – Comparisons of the Hadamard, Fourier, Haar, and random patterns. We consider the the maximum noise amplification coefficient η, the light throughput γ, and finally whether these four types of patterns can be considered as binary.

This is done by solving

$$\boldsymbol{H} = \underset{\boldsymbol{H}^*}{\operatorname{argmax}}\, \mathbb{E}(\|\boldsymbol{H}^*\boldsymbol{f}\|_2^2) = \underset{\boldsymbol{H}^*}{\operatorname{argmax}}\, \mathbb{V}\mathrm{ar}(\|\boldsymbol{H}^*\boldsymbol{f}\|_2) + \mathbb{E}(\|\boldsymbol{H}^*\boldsymbol{f}\|_2)^2. \tag{1.21}$$

$\boldsymbol{H}$ is indexed by the set Ω, which represents the subset of M retained indices from the original signal.

This choice of $\boldsymbol{H}$ is optimal for ℓ_2-based reconstructors, which for orthogonal transforms is equivalent to the inverse transform of the down-sampled signal (see e.g., (Baldassarre et al., 2016)).

As the probability distribution of $\mathbf{f}$ is not known, it is impossible to compute the solution to 1.21. However an estimation can be obtained via the S samples $(\mathbf{f}_1, \ldots, \mathbf{f}_S)$

$$\boldsymbol{H} = \underset{\boldsymbol{H}^*}{\operatorname{argmax}}\, \frac{1}{S}\sum_{j=1}^{S} \|\boldsymbol{H}^*\boldsymbol{f}_j\|_2^2. \tag{1.22}$$

(Baldassarre et al., 2016) showed that given enough samples, we can approach the true optimal down-sampling operator with any arbitrary degree of precision.

Adaptive sampling This adaptive sampling strategy was designed, in particular, for wavelet masks (Rousset et al., 2017), although we can imagine the same principles applied to other bases. This empiric method is aimed to perform the acquisition through several steps. For each step, some coefficients are acquired, and we estimate the highest nonacquired coefficients using up-sampling algorithms.

1.2.4 Comparisons of different patterns

When choosing an appropriate set of patterns to implement on a DMD, certain compromises need to be made. We summarise the different criteria of the different pattern types in Table 1.1.

Compression abilities To reduce acquisition times, we want patterns that allow the recovery of as good an image as possible with as few coefficients as possible. This has been studied empirically for the wavelet, Fourier, and Hadamard bases. The wavelet basis is known to show excellent compression (Mallat, 1999), which is far better than, for instance, the Fourier basis or the Hadamard basis (Yu et al., 2020b).

Further comparisons between Hadamard and Fourier were made in terms of their compression (Zhang et al., 2017c). The results appear to indicate that Fourier shows slightly better compression. (Chen et al., 2018) reported on a comparative study between random and Hadamard patterns, where for total variation based reconstructors, Hadamard patterns provided marginally better reconstructions at different compression rates.

Noise variance reduction We define η, the maximum noise amplification coefficient, as

$$\eta = \max_{\boldsymbol{f}} \frac{\|\boldsymbol{H}^{\dagger}\boldsymbol{f}\boldsymbol{H}^{\dagger\top}\|}{\|\boldsymbol{f}\|} \tag{1.23}$$

This coefficient tells us how much the variance of additive noise is reduced in the worst case when applying the Moore-Penrose pseudo-inverse denoted here as $\boldsymbol{H}^{\dagger}$. Assuming that the data is corrupted by an additive noise with a diagonal covariance matrix $\boldsymbol{\Sigma}_{\alpha}$, the covariance of the relative reconstruction error is lower than $\|\boldsymbol{H}^{\dagger}\boldsymbol{\Sigma}_{\alpha}\boldsymbol{H}^{\dagger\top}\|$. Therefore η offers an upper bound for the variance of the relative reconstruction error.

For the Hadamard, Fourier, and wavelet bases, η can be analytically determined. As shown in Table 1.1, for Hadamard and Fourier patterns, we get $\eta = 1/N$. For wavelet patterns based on the scaling of the first level of decomposition, we get $\eta = 1/2$. For random patterns this criterion is hard to apply, as they can vary substantially.

Light throughput The light throughput is a qualitative criterion to determine how much light will be collected for a given pattern. In the presence of Poisson noise (Foi et al., 2008), the higher the throughput, the better the signal-to-noise ratio. The least number of illuminated pixels per pattern can be defined as

$$\gamma = \min_{1\leq i\leq N} \frac{\|\boldsymbol{h}_i\|_1}{N} \tag{1.24}$$

Considering an acquired noisy measurement, $m_i^{\alpha} \sim \frac{1}{\alpha}\mathcal{P}(\alpha \boldsymbol{h}_i^{\top}\boldsymbol{f})$ (where α is the image intensity in photons), the signal-to-noise ratio for this coefficient is

$$\mathrm{SNR}(m_i^{\alpha}) = \alpha\frac{(\boldsymbol{h}_i^{\top}\boldsymbol{f})^2}{\boldsymbol{h}_i^{\top}\boldsymbol{f}} = \alpha\boldsymbol{h}_i^{\top}\boldsymbol{f} \geq \alpha N\gamma\|\boldsymbol{f}\|_1. \tag{1.25}$$

As N, α, and $\|\boldsymbol{f}\|_1$ are independent of the choice of patterns, in choosing patterns with higher values of γ, we make sure that the signal-to-noise ratio for every acquired measurement is higher.

As shown in Table 1.1 for random, Fourier, and Hadamard patterns, this number is set at $1/2$, as every single pattern averages to $1/2$. For Wavelet patterns on the other hand, the first decomposition level always produces patterns with an average of $1/\log_2(N)$.

Binary patterns The binary nature of patterns has an important role in ensuring that the DMD can transition between the different patterns at high frequency. For instance, most DMDs have a mirror flip frequency close to 20 kHz for binary patterns, whereas for 8-bit gray-scale patterns, teh mirror flip frequency is usually close to 256 Hz. Therefore, binary patterns are ideal for real-time single-pixel applications.

1.3 Objectives of the thesis

This PhD was financed by the ARMONI ANR (Ducros, 2017) project, which aims to promote the reduction of tumor margins by exploiting hyperspectral images acquired by a single-pixel camera. By analysing the spectral signature, it has been shown that we can differentiate between healthy and tumorous margins (Alston, 2017; Valdés et al., 2015).
In particular, this thesis aims to develop single-pixel algorithms for real-time single-pixel imaging. The fast acquisition implies compressive measurements with short integration times. This leads to an under-determined inverse problem with high levels of noise.
Furthermore, when considering the acquisition of videos, we want to exploit the temporal redundancy of natural videos to avoid reacquiring redundant coefficients over time.

1.4 The approach considered, and the choices

As discussed in the previous section, single-pixel imaging involves choosing a set of patterns, and to choose how to down-sample the chosen basis function. The approach that was taken during this thesis was to use the Hadamard basis down-sampled through statistical sampling. The use of the Hadamard basis seemed natural considering its properties of additive Gaussian noise reduction. These allow more light to be acquired, which increases the signal-to-noise ratio for our measurements with respect to the Poisson photon-counting noise. Finally, the binary nature of Hadamard patterns allows for a quick hardware implementation for single-pixel imaging applications.

The biggest downside of the Hadamard basis is the compression, which is lower than most of the other possibilities. While this point can be problematic, the noise-reduction means that we can reduce the integration time for each pattern. This noise-reducing property coupled with a fast micromirror flip frequency - due to the binary nature of our patterns - allows more measurements to be acquired. This helps to counter the comparatively lower compression.

While fast reconstruction has mainly been tackled through the use of deep-learning algorithms, their lack of interpretability is still an issue. We therefore aimed to combine deep learning with classic inverse problem theory. In particular, we aimed to develop methods that are interpretable and that can be adapted easily to specifically deal with the noise model specific to single-pixel imaging.

Mathematical formulation of the inverse problem considered

Let $\boldsymbol{f} \in [0,1]^N$ be the image to acquire. We measure $\boldsymbol{y} = \boldsymbol{H}\boldsymbol{f}$ using the hardware, and recover $\boldsymbol{f}$ using the software. The matrix $\boldsymbol{H} = [\boldsymbol{h}_1, \dots, \boldsymbol{h}_N] \in \mathbb{R}^{N \times N}$ collects the patterns that are sequentially uploaded in a SLM. These patterns can be chosen as seen in Section 1.2.3.2, and in our case, this refers to the Hadamard basis ordered according the energy-based statistical maps, as described in Section 1.2.3.2. To accelerate the acquisition times, not all of the coefficients are acquired.

The acquisition is corrupted by a mixed Poisson-Gaussian noise (Foi et al., 2008; Rosenberger et al., 2016), and can be modeled as

$$\hat{\mathbf{m}}^{\alpha}_{+,-} \sim K\,\mathcal{P}(\alpha \boldsymbol{S}\boldsymbol{H}^{+,-}\boldsymbol{f}) + \mathcal{N}(\mu_{\text{dark}}, \sigma^2_{\text{dark}}) \tag{1.26}$$

where $\mathcal{P}$ and $\mathcal{N}$ are the Poisson and Gaussian distributions, $\boldsymbol{S} = [\boldsymbol{I}_M, \boldsymbol{0}] \in \mathbb{R}^{M \times N}$ is a down-sampling matrix (where $M \leq N$), K is a constant that represents the overall system gain (in counts/electron), α is the intensity (in photons) of the image (which is proportional to the integration time), μ_{dark} is the dark current (in counts), and σ_{dark} is the dark noise (in counts). We further hypothesize that μ_{dark} and σ_{dark} are independent of the image intensity α.

The difference between two Poisson variables leads to normalized measurements $\mathbf{m}^\alpha$ corrupted by mixed Skellam-Gaussian (Skellam, 1946) noise:

$$\mathbf{m}^\alpha = (\hat{\mathbf{m}}_+^\alpha - \hat{\mathbf{m}}_-^\alpha)/(\alpha K). \tag{1.27}$$

Our goal was to design algorithms to estimate $\boldsymbol{f}$ from $\boldsymbol{m}^\alpha$. We will approximate the noise as a normally distributed noise leading to

$$\mathbf{m}^\alpha = \boldsymbol{S}\boldsymbol{H}\boldsymbol{f} + \boldsymbol{\epsilon}^\alpha, \quad \text{with} \quad \boldsymbol{\epsilon}^\alpha \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma}_\alpha). \tag{1.28}$$

We will discuss the reasons and the foundations behind this normal approximation in Chapter 7.

Conclusion

This chapter presents the core principles of single-pixel imaging. It introduces the fundamental aspects behind the hardware implementation of the single-pixel camera, the choice of pattern designs, and the down-sampling strategies. In this thesis, we chose to down-sample the image on the Hadamard basis, with statistical down-sampling to reduce acquisition times. This leads to an under-determined inverse problem that is corrupted with mixed Skellam-Gaussian noise. We will explore methods to solve this inverse problem in the following chapters. Both the static and dynamic cases will be explored.

Chapter 2

Bayesian reconstruction for under-determined linear inverse problems

This chapter provides a brief introduction to the philosophy and mathematical foundations of Bayesian inference for inverse problems in computational imaging. In particular, we show their use for reconstruction of images from under-determined linear inverse problems.

2.1 Bayesian inversion for linear inverse problems

2.1.1 Problem to solve

We consider linear inverse problems

$$\boldsymbol{m} = \boldsymbol{A}\boldsymbol{f} + \boldsymbol{\epsilon}, \tag{2.1}$$

$$\boldsymbol{\epsilon} \sim p(\boldsymbol{\epsilon}), \tag{2.2}$$

where $\boldsymbol{m} \in \mathbb{R}^M$ are the measurements, $\boldsymbol{f} \in \mathbb{R}^N$ is the original data we aim to recover from the measurements, and $\boldsymbol{\epsilon} \in \mathbb{R}^M$ is the noise following a probability density function. For independent measurements (as is often the case), through the chain rule, $p(\boldsymbol{\epsilon}) = \prod_{i=1}^{M} p(\epsilon_i)$. We consider the specific case where $M < N$. This family of problems is commonly known as under-determined linear inverse problems, and can be used to model a wide variety of imaging problems, such as super-resolution, and of course, the single-pixel camera problem considered here (simply by considering $\boldsymbol{A} = \boldsymbol{S}\boldsymbol{H}$ in Eq. 1.28, and $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma}_\alpha)$).

2.1.2 Statistical approach to inverse problems

Bayesian inference aims to use probability distributions and the theory of probability calculus as a tool to solve inverse problems. The uncertainty produced by the random nature of noise, and the loss of information from the forward operator $\boldsymbol{A}$, are modeled by the use of probability density functions.

The Bayesian approach assumes that both the measurements and the unknown image are samples from a probability distribution. Mathematically, $\boldsymbol{m}$ stems from a random variable $\mathbf{m}$ with a probability density function $p(\boldsymbol{m})$. Similarly, it assumes that $\boldsymbol{f}$ stems from a random variable $\mathbf{f}$ with a probability density function - known as the prior distribution - $p(\boldsymbol{f})$.

Furthermore, Bayesian approaches also assume that we have access to the distribution of the observation $\mathbf{m}$, given $\mathbf{f}$ through the measurement model of Eq. 2.2. This is also known as the likelihood of the model $p(\boldsymbol{m}|\boldsymbol{f}, \boldsymbol{A})$, which is directly linked to the

probability density function of $\boldsymbol{\epsilon}$. For a given $\boldsymbol{A}$ and $\boldsymbol{f}$, $p_{\boldsymbol{\epsilon}}(\boldsymbol{\epsilon}) = p_{\boldsymbol{\epsilon}}(\boldsymbol{m} - \boldsymbol{A}\boldsymbol{f}|\boldsymbol{A}, \boldsymbol{f})$. If $\boldsymbol{A}$ and $\boldsymbol{f}$ are held constant, then $p_{\boldsymbol{\epsilon}}(\boldsymbol{m} - \boldsymbol{A}\boldsymbol{f}|\boldsymbol{A}, \boldsymbol{f})$ only depends on $\boldsymbol{m}$. This allows us to define the probability density function of $p(\boldsymbol{m}|\boldsymbol{A}, \boldsymbol{f}) = p_{\boldsymbol{\epsilon}}(\boldsymbol{m} - \boldsymbol{A}\boldsymbol{f}|\boldsymbol{A}, \boldsymbol{f})$. Note that the dependency on $\boldsymbol{A}$ always exists, but for the sake of simplicity it is mostly omitted in the notations, and we only note $p(\boldsymbol{m}|\boldsymbol{f})$ for the likelihood.

The Bayesian solution to the inverse problem is defined as the probability density function of $\mathbf{f}$ given $\mathbf{m}$, this is also called the posterior distribution $p(\boldsymbol{f}|\boldsymbol{m})$. This solution not only provides estimators of $\boldsymbol{f}$, but also allows us to determine how likely the estimators are to be true for a given measurement.

Assuming that $p(\boldsymbol{f})$ is known, the target distribution can then be computed for a given sample $\boldsymbol{m}$ using the Bayes rule

$$p(\boldsymbol{f}|\boldsymbol{m}) = \frac{p(\boldsymbol{f})p(\boldsymbol{m}|\boldsymbol{f})}{p(\boldsymbol{m})}. \tag{2.3}$$

While at first, it might appear that $p(\boldsymbol{m})$ is also needed, it is worth noting that for a given value $\boldsymbol{m}$, this is a constant, and can be deduced as the normalising parameter, to normalise the probability density function generated by $p(\boldsymbol{f})p(\boldsymbol{m}|\boldsymbol{f})$.

Unfortunately, this solution is impractical for high dimensional problems. Indeed, $p(\boldsymbol{f}|\boldsymbol{m})$ is a mapping from $\mathbb{R}^N$ to $\mathbb{R}$, so even small-sized discrete problems require a large amount of memory and are intractable. For instance, storing the probability density function of 64×64 images using single-precision floating point variables would require $32^{64\times64} = 1.2 \cdot 10^{6165}$ bytes of memory. This explains why for imaging inverse problems, Bayesian approaches tend to limit themselves to point-wise estimators of the posterior, which we will discuss in the following section.

2.2 Point-wise estimators of the posterior distribution

As discussed in the previous section, as obtaining the full posterior distribution is impossible for the imaging of most inverse problems, we tend to limit ourselves to point-wise estimators of $p(\boldsymbol{f}|\boldsymbol{m})$. In this section, we cover the most popular estimators.

2.2.1 Maximum a-posteriori estimation and links to deterministic approaches to inverse problems

The maximum a posteriori (MAP) aims to compute the value that maximises $p(\boldsymbol{f}|\boldsymbol{m})$. This estimator gives the most likely $\boldsymbol{f}$ for a given $\boldsymbol{m}$ by minimizing

$$\boldsymbol{f}^* = \underset{\boldsymbol{f}}{\operatorname{argmin}}\{-\log(p(\boldsymbol{f}|\boldsymbol{m}))\}, \tag{2.4a}$$

$$\boldsymbol{f}^* = \underset{\boldsymbol{f}}{\operatorname{argmin}}\{-\log p(\boldsymbol{m}|\boldsymbol{f}) - \log p(\boldsymbol{f}) + \log p(\boldsymbol{m})\}, \tag{2.4b}$$

$$\boldsymbol{f}^* = \underset{\boldsymbol{f}}{\operatorname{argmin}}\{-\log p(\boldsymbol{m}|\boldsymbol{f}) - \log p(\boldsymbol{f})\}, \tag{2.4c}$$

where Eq. 2.4b follows through the Bayes rule, and Eq. 2.4c follows as $p(\boldsymbol{m})$ does not depend on $\boldsymbol{f}$. This minimization does indeed guarantee that we maximise $p(\boldsymbol{f}|\boldsymbol{m})$, but it also simplifies the expression of exponential (Duda and Hart, 1973) prior distributions (see Section 2.3.1) and Gaussian and Laplace noise distributions, and it converts multiplications into additions, which tends to simplify the minimization problem.

2.2.2 Conditional expectation

The conditional expectation is another popular point estimator of the posterior distribution. It gives the expected value of $\boldsymbol{f}$ given $\boldsymbol{m}$:

$$\boldsymbol{f}^* = \mathbb{E}(\mathbf{f}|\boldsymbol{m}) = \int_{\mathbb{R}^N} p(\boldsymbol{f}|\boldsymbol{m})\boldsymbol{f}\ d\boldsymbol{f}. \tag{2.5}$$

Computing this estimator is challenging, as it requires integration over $\mathbb{R}^N$. However, as we will show in Section 2.2.3, this estimator can be computed by generating samples from the posterior. As shown in Section 3.1, it can also be approximated by using deep-learned reconstructors.

2.2.3 Sampling-based estimation

Sampling-based estimation aims to generate K samples $\{\boldsymbol{f}^i\}_{1\leq i\leq K}$ from a probability density function $p(\boldsymbol{f}|\boldsymbol{m})$. Then $p(\boldsymbol{f}|\boldsymbol{m})$ is analytically approximated as

$$p(\boldsymbol{f}|\boldsymbol{m}) \approx \sum_{i=1}^{K} w^{(i)}\delta(\boldsymbol{f}-\boldsymbol{f}^i), \tag{2.6}$$

where the weights $w^{(i)}$ need to be computed based on the likelihood of $\boldsymbol{f}^i$.

This approximation further allows computation of estimators in the form $\mathbb{E}(\boldsymbol{f}|\boldsymbol{m})$, through a Monte-Carlo approximation (Giovannelli, 2008)

$$\mathbb{E}(\boldsymbol{f}|\boldsymbol{m}) \approx \frac{1}{K}\sum_{i=1}^{K} w^{(i)}\boldsymbol{f}^i. \tag{2.7}$$

This approximation will converge with a speed $\mathcal{O}(K^{-1/2})$ due to the central limit theorem, and it is independent of the dimension of the treated problem.

The only remaining question is: how do we simulate the samples $\boldsymbol{f}^i \sim p(\boldsymbol{f}|\boldsymbol{m})$? We further describe two popular options for Monte-Carlo reconstructions.

Metropolis-Hastings sampling The Metropolis-Hastings algorithm (Hastings, 1970) generates a Markov chain with a stationary probability distribution that follows $p(\boldsymbol{f}|\boldsymbol{m})$. It requires that $p(\boldsymbol{f}|\boldsymbol{m})$ can be evaluated for any $\boldsymbol{f}$ up to a multiplicative constant, and a probability distribution π from where samples can easily be drawn (e.g., a Gaussian distribution).

The algorithm is presented in section Algorithm 1. This generates the different iterations based solely on the previous iteration, to generate a Markov chain. Every new iteration can be kept or rejected with a certain probability. The rejection probability is based on $p(\boldsymbol{f})p(\boldsymbol{m}|\boldsymbol{f})$, and after enough iterations (usually referred to as the number of burnout iterations, n_b), the iterations are generated according to $p(\boldsymbol{f}|\boldsymbol{m})$.

The number of burn-in iterations (Boissy, Giovannelli, and Minvielle, 2020) depends on the choice of π. For Gaussian distributions, it is often necessary to adjust the variance of this probability distribution to ensure that we do not reject all of the samples, or alternatively to ensure that we do not accept all of the samples. For N-dimensional random variables $\boldsymbol{f}$, desirable acceptance rates would lie between 20% and 50% (Gelman, Gilks, and Roberts, 1997). The potentially high burn-in iterations can make this algorithm very time consuming, and unsuitable for real-time applications.

Algorithm 1 The Metropolis-Hastings sampling algorithm

Initialisation : randomly pick an initial value $\boldsymbol{f}^{(0)}$
for $i \in \{1, \dots, K + n_b\}$ **do**
  Generate a candidate sample according to π
  $\boldsymbol{f}_c^{(i)} = \boldsymbol{f}^{(i-1)} + p_i \quad \text{where} \quad p_i \sim \pi$
  Compute the acceptance probability
  $a^{(i)} = \min\left(1, \frac{p(\boldsymbol{f}_c^{(i)})p(\boldsymbol{m}|\boldsymbol{f}_c^{(i)})}{p(\boldsymbol{f}^{(i-1)})p(\boldsymbol{m}|\boldsymbol{f}^{(i-1)})}\right)$
  Accept or reject the new candidate
    Generate $u^{(i)}$, a random uniform number $\in [0, 1]$
    If $u^{(i)} \leq a^{(i)}$, accept $\boldsymbol{f}_c^{(i)}$ as the new iterate i.e. $\boldsymbol{f}^{(i)} = \boldsymbol{f}_c^{(i)}$
    If $u^{(i)} > a^{(i)}$, reject $\boldsymbol{f}_c^{(i)}$, and copy the previous iteration $\boldsymbol{f}^{(i)} = \boldsymbol{f}^{(i-1)}$.
end for
The iterations $\{\boldsymbol{f}^{(n_b)+1}, \dots, \boldsymbol{f}^{(n_b+K)}\}$ **are kept as K samples of** $p(\boldsymbol{f}|\boldsymbol{m})$.

Importance sampling Importance sampling (Doucet, 2001) uses a distribution $\pi(\boldsymbol{f}|\boldsymbol{m}))$ – called the importance distribution – and uses it to obtain an estimate of Eq. 2.7. The importance distribution needs to be easy to sample, and $\text{supp}(p(\boldsymbol{f}|\boldsymbol{m})) \subset \text{supp}(\pi(\boldsymbol{f}|\boldsymbol{m}))$. A typical choice of importance distribution is Gaussian distributions.

The importance distribution can then easily appear in the target estimator expression:

$$\mathbb{E}(\boldsymbol{f}|\boldsymbol{m}) = \int_{\mathbb{R}^N} \frac{p(\boldsymbol{f}|\boldsymbol{m})}{\pi(\boldsymbol{f}|\boldsymbol{m})} \pi(\boldsymbol{f}|\boldsymbol{m}) \boldsymbol{f} \; d\boldsymbol{f}. \tag{2.8}$$

By using the Bayes theorem, and by applying the Monte-Carlo approximation of Eq. 2.7, our estimator becomes

$$\mathbb{E}(\boldsymbol{f}|\boldsymbol{m}) \approx \sum_{i=1}^{K} \frac{\frac{p(\boldsymbol{f}^i)p(\boldsymbol{m}|\boldsymbol{f}^i)}{\pi(\boldsymbol{f}^i|\boldsymbol{m})}}{\sum_{j=1}^{K} \frac{p(\boldsymbol{f}^j)p(\boldsymbol{m}|\boldsymbol{f}^j)}{\pi(\boldsymbol{f}^j|\boldsymbol{m})}} \boldsymbol{f}^i. \tag{2.9}$$

This leads to the following importance sampling algorithm :

Algorithm 2 Importance sampling algorithm

Data: $\boldsymbol{m}$
Known functions: $p(\boldsymbol{f}), p(\boldsymbol{m}|\boldsymbol{f})$
1 - **Draw K samples** $\{\boldsymbol{f}^i\}_{1 \leq i \leq K} \sim \pi(\boldsymbol{f}|\boldsymbol{m})$
2 - **Compute the nonnormalized weights** $w^{*(i)} = \frac{p(\boldsymbol{f}^i)p(\boldsymbol{m}|\boldsymbol{f}^i)}{\pi(\boldsymbol{f}^i|\boldsymbol{m})}$.
3 - **Normalize the weights** $w^{(i)} = \frac{w^{*(i)}}{\sum_{j=1}^{K} w^{*(j)}}$.
4 - **Compute the estimator of the target expectation**
$\mathbb{E}(\mathcal{F}(\boldsymbol{f})|\boldsymbol{m}) \approx \sum_{i=1}^{K} w^{(i)} \mathcal{F}(\boldsymbol{f}^i)$

The convergence rate of importance sampling algorithms essentially depend on how closely the importance distribution approximates the posterior distribution, where the ideal case would be $\pi(\boldsymbol{f}|\boldsymbol{m}) = p(\boldsymbol{f}|\boldsymbol{m})$. This result means that the importance sampling algorithm can be time consuming, in particular if the choice of $\pi(\boldsymbol{f}|\boldsymbol{m})$ is ill-suited to our problem.

2.3 Choosing a prior distribution, and maximum a posteriori

As pointed out in Section 2.1.2, for Bayesian approaches to be used, it is necessary to know the prior distribution of the model parameters $p(\boldsymbol{f})$. As this is usually unknown, for most image reconstruction problems handcrafted priors are used for $p(\boldsymbol{f})$. We describe some examples and the corresponding MAP estimators here.

For all of these examples, we also assume (unless stated otherwise) that the likelihood follows a generalized Gaussian distribution

$$\epsilon \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_\alpha), \tag{2.10}$$

or, in other words,

$$p(\boldsymbol{m}|\boldsymbol{f}) \propto e^{-\frac{1}{2}\|\boldsymbol{A}\boldsymbol{f}-\boldsymbol{m}\|^2_{\boldsymbol{\Sigma}_\alpha^{-1}}}. \tag{2.11}$$

2.3.1 Exponential priors

Exponential priors are widespread in the image reconstruction community, and they are given by

$$p(\boldsymbol{f}) = \frac{e^{-g(\boldsymbol{f})}}{\int e^{-g(\boldsymbol{h})} d\boldsymbol{h}} \propto e^{-g(\boldsymbol{f})}. \tag{2.12}$$

g is usually chosen as a convex function, which results in MAP estimations that benefit from the theory of convex optimisation. Some of the most popular choices of g include the ℓ_2 norm (Tikhonov, 1995), the ℓ_1 norm (J. Candès, K. Romberg, and Tao, 2006), and the total variation (Rudin, Osher, and Fatemi, 1992). The ℓ_1 norm in particular has been widely identified as a norm that seeks sparse solutions. It has been mainly associated with the theory of compressed sensing. The total variation has mainly been identified as a prior that favours piece-wise linear functions. Exponential priors simplify the MAP expression

$$\boldsymbol{f}^* = \underset{\boldsymbol{f}}{\operatorname{argmin}} \|\boldsymbol{A}\boldsymbol{f} - \boldsymbol{m}\|^2_{\boldsymbol{\Sigma}_\alpha^{-1}} + g(\boldsymbol{f}), \tag{2.13}$$

In Sec. 2.3.2, we explore the ℓ_2 regularisation with its advantages, and in Sec. 2.3.3 we'll quickly explore the implications of other choices of g alongside with some examples of algorithms to find the MAP estimate.

2.3.2 ℓ_2 solutions: the Gaussian prior

The exponential distribution that results from a weighed ℓ_2 norm is a Gaussian distribution. Solving MAP Gaussian problems is equivalent to doing a Tikhonov regularisation. Tikhonov regularisation is widely used, as it is easy to use due to its exact solutions. Furthermore, many optimisation algorithms used for convex optimisation (see Section 2.3.3) require the resolution of several ℓ_2 optimisations. This means that the ℓ_2 regularisation has huge implications even if the chosen prior is not Gaussian.

2.3.2.1 Noiseless measurements: the pseudo-inverse

The Moore-Penrose (Penrose, 1955) pseudo-inverse is the solution of the noiseless, nonweighed ℓ_2 MAP regularisation. In a noiseless case, $p(\boldsymbol{m}|\boldsymbol{f}) = \delta(\boldsymbol{m} - \boldsymbol{A}\boldsymbol{f})$, where

δ denotes the Dirac distribution. This means that the solution to the inverse problem would be solved over the hyper-plane $\boldsymbol{m} = \boldsymbol{A}\boldsymbol{f}$. The MAP estimator is therefore

$$\boldsymbol{f}^* = \underset{\boldsymbol{f}}{\operatorname{argmin}} \|\boldsymbol{f}\|_2^2 \quad \text{such that} \quad \boldsymbol{m}^\alpha = \boldsymbol{A}\boldsymbol{f}. \tag{2.14}$$

Equation 2.14 has an analytical solution known as the Moore-Penrose pseudo-inverse

$$\boldsymbol{f}^* = \boldsymbol{A}^\top(\boldsymbol{A}\boldsymbol{A}^\top)^{-1}\boldsymbol{m}. \tag{2.15}$$

Furthermore, if $\boldsymbol{A}$ is a down-sampled orthogonal matrix (i.e., it can be written as $\boldsymbol{A} = \boldsymbol{S}\boldsymbol{H}$, where $\boldsymbol{H} \in \mathbb{R}^{N\times N}$ is an orthogonal transform matrix and $\boldsymbol{S} = [\boldsymbol{I}_M, \boldsymbol{O}] \in \mathbb{R}^{M\times N}$ is a down-sampling matrix), Eq. 2.15 is equivalent to

$$\boldsymbol{f}^* = \boldsymbol{H}^\top\boldsymbol{S}^\top(\boldsymbol{S}\boldsymbol{H}\boldsymbol{H}^\top\boldsymbol{S}^\top)^{-1}\boldsymbol{m} = \boldsymbol{H}^\top\boldsymbol{S}^\top(\underbrace{\boldsymbol{S}\boldsymbol{S}^\top}_{=\boldsymbol{I}_M})^{-1}\boldsymbol{m} = \boldsymbol{H}^\top\boldsymbol{S}^\top\boldsymbol{m}, \tag{2.16}$$

which can also be computed as

$$\boldsymbol{f}^* = \boldsymbol{H}^\top\boldsymbol{y}^*, \quad \text{with } \boldsymbol{y}^* = \begin{bmatrix}\boldsymbol{m}^\alpha \\ \boldsymbol{0}\end{bmatrix}. \tag{2.17}$$

2.3.2.2 Noisy measurements: the Tikhonov solution

The case where $\mathbf{f}$ and $\mathbf{m}|\boldsymbol{f}$ are both normally distributed is one of the few cases where full Bayesian inversion is possible. Assuming

$$\mathbf{f} \sim \mathcal{N}(\boldsymbol{\mu}, \tilde{\boldsymbol{\Sigma}}), \tag{2.18}$$
$$\mathbf{m}|\boldsymbol{f} \sim \mathcal{N}(\boldsymbol{A}\boldsymbol{f}, \boldsymbol{\Sigma}_\alpha), \tag{2.19}$$

we can show that (see Theorem B.0.3 in Annex B) $\mathbf{f}|\boldsymbol{m}$ is also normally distributed

$$\mathbf{f}|\boldsymbol{m} \sim \mathcal{N}(\boldsymbol{f}^*, \boldsymbol{P}^*), \tag{2.20}$$

and that the expectation and covariance are given by the analytical expressions

$$\boldsymbol{f}^* = \boldsymbol{\mu} + \tilde{\boldsymbol{\Sigma}}\boldsymbol{A}^\top(\boldsymbol{A}\tilde{\boldsymbol{\Sigma}}\boldsymbol{A}^\top + \boldsymbol{\Sigma}_\alpha)^{-1}(\boldsymbol{m} - \boldsymbol{A}\boldsymbol{\mu}), \tag{2.21}$$

and

$$\boldsymbol{P}^* = \tilde{\boldsymbol{\Sigma}} - \tilde{\boldsymbol{\Sigma}}\boldsymbol{A}^\top(\boldsymbol{A}\tilde{\boldsymbol{\Sigma}}\boldsymbol{A}^\top + \boldsymbol{\Sigma}_\alpha)^{-1}\boldsymbol{A}\tilde{\boldsymbol{\Sigma}}. \tag{2.22}$$

Furthermore, due to the properties of normal variables, the conditional expectancy $\boldsymbol{f}^*$ is also the solution to the MAP (see Theorem B.0.4 in Annex B for more details):

$$\boldsymbol{f}^* = \underset{\boldsymbol{f}}{\operatorname{argmin}} \|\boldsymbol{A}\boldsymbol{f} - \boldsymbol{m}\|^2_{\boldsymbol{\Sigma}_\alpha^{-1}} + \|\boldsymbol{f} - \boldsymbol{\mu}\|^2_{\tilde{\boldsymbol{\Sigma}}^{-1}}. \tag{2.23}$$

Additionally, Eq. 2.21 has a straightforward interpretation if $\boldsymbol{A} = \boldsymbol{S}\boldsymbol{H}$ is a down-sampled orthogonal matrix, with $\boldsymbol{H} \in \mathbb{R}^{N\times N}$ being an orthogonal transform matrix. If we consider the prior variance in the transform domain $\boldsymbol{\Sigma} = \boldsymbol{H}^\top\tilde{\boldsymbol{\Sigma}}\boldsymbol{H}$, which admits the following decomposition

$$\boldsymbol{\Sigma} = \begin{bmatrix}\boldsymbol{\Sigma}_1 & \boldsymbol{\Sigma}_{21}^\top \\ \boldsymbol{\Sigma}_{21} & \boldsymbol{\Sigma}_2\end{bmatrix} \tag{2.24}$$

with $\boldsymbol{\Sigma}_1 \in \mathbb{R}^{M\times M}$ and $\boldsymbol{\Sigma}_{21} \in \mathbb{R}^{(N-M)\times M}$, then 2.21 is equivalent to

$$\boldsymbol{f}^* = \boldsymbol{H}^\top \boldsymbol{y}^*, \quad \text{with } \boldsymbol{y}^* = \begin{bmatrix} \boldsymbol{y}_1^* \\ \boldsymbol{y}_2^* \end{bmatrix}, \tag{2.25a}$$

$$\boldsymbol{y}_1^* = \boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_1[\boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_\alpha]^{-1}(\boldsymbol{m}^\alpha - \boldsymbol{\mu}_1), \tag{2.25b}$$

$$\boldsymbol{y}_2^* = \boldsymbol{\mu}_2 + \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_1^{-1}[\boldsymbol{y}_1^*(\boldsymbol{m}^\alpha) - \boldsymbol{\mu}_1]. \tag{2.25c}$$

In the transform domain, Eq. (2.25b) corresponds to denoising the acquired measurements $\boldsymbol{m}^\alpha$, and Eq. (2.25c) to completing the missing coefficient from the denoised image.

2.3.3 Non-Gaussian priors

In this section, we will consider the case of an exponential prior distribution where g is convex (like the ℓ_1 norm, or total variation, for instance).

We will provide some insight as to how the MAP estimator

$$\boldsymbol{f}^* = \underset{\boldsymbol{f}}{\operatorname{argmin}} \|\boldsymbol{A}\boldsymbol{f} - \boldsymbol{m}\|^2_{\boldsymbol{\Sigma}_\alpha^{-1}} + g(\boldsymbol{f}), \tag{2.26}$$

can be obtained by considering one popular algorithm for convex optimisation, the alternating direction method of multipliers (ADMM).

The ADMM (Boyd et al., 2011) algorithm is a convex optimisation algorithm based on the Karush–Kuhn–Tucker (KKT) conditions (Kuhn and Tucker, 1951). To apply the ADMM algorithm, it is important to note that solving Eq. 2.26 is equivalent to solving a problem where we introduce an auxiliary variable $\boldsymbol{z}$ that is constrained to be equal to $\boldsymbol{f}$, i.e.,

$$\boldsymbol{f}^* = \underset{\boldsymbol{f}}{\operatorname{argmin}} \|\boldsymbol{A}\boldsymbol{f} - \boldsymbol{m}\|^2_{\boldsymbol{\Sigma}_\alpha^{-1}} + g(\boldsymbol{z}) \quad \text{such that} \quad \boldsymbol{f} = \boldsymbol{z}. \tag{2.27}$$

The augmented Lagrangian function is

$$\mathfrak{L}(\boldsymbol{f}, \boldsymbol{z}, \boldsymbol{\alpha}) = \|\boldsymbol{A}\boldsymbol{f} - \boldsymbol{m}\|^2_{\boldsymbol{\Sigma}_\alpha^{-1}} + g(\boldsymbol{z}) + \boldsymbol{\beta}^\top(\boldsymbol{f} - \boldsymbol{z}) + \rho\|\boldsymbol{f} - \boldsymbol{z}\|_2^2, \tag{2.28}$$

where $\boldsymbol{\beta}$ are the Lagrangian multipliers, and ρ is a penalty parameter. The ADMM then alternatively optimizes $\{\boldsymbol{f}, \boldsymbol{z}, \boldsymbol{\beta}\}$ by solving the following subproblems :

$$\boldsymbol{f}^{(k+1)} = \underset{\boldsymbol{f}}{\operatorname{argmin}} \|\boldsymbol{A}\boldsymbol{f} - \boldsymbol{m}\|^2_{\boldsymbol{\Sigma}_\alpha^{-1}} + \rho\|\boldsymbol{f} + \frac{1}{\rho}\boldsymbol{\beta}^{(k)} - \boldsymbol{z}^{(k)}\|_2^2, \tag{2.29a}$$

$$\boldsymbol{z}^{(k+1)} = \underset{\boldsymbol{z}}{\operatorname{argmin}}\, g(\boldsymbol{z}) + \rho\|\boldsymbol{f}^{(k+1)} + \frac{1}{\rho}\boldsymbol{\beta}^{(k)} - \boldsymbol{z}\|_2^2, \tag{2.29b}$$

$$\boldsymbol{\beta}^{(k+1)} = \boldsymbol{\beta}^{(k)} + \nu(\boldsymbol{f}^{(k+1)} - \boldsymbol{z}^{(k+1)}), \tag{2.29c}$$

where (k) represents the index of the different iterates, and the parameter ν is an update rate for the multiplier $\boldsymbol{\beta}$. The ADMM algorithm converges to a global minimum assuming that $\boldsymbol{g}$ is a convex function (Eckstein and Yao, 2015).

As we can see, Eq. 2.29a is a Tikhonov regularized problem that admits the analytical solution given by Eq. 2.21. In the case where $\boldsymbol{A}$ is a down-sampled orthogonal transform matrix, the solution becomes Eq. 2.25, which can be computed efficiently with a fast transform.

Equation 2.29b is solved using the proximal mapping of g. For some convex functions, it has a closed form. For instance, if $g(\boldsymbol{f}) = \lambda\|\boldsymbol{f}\|_1$ ($\lambda > 0$), Eq. 2.29b can be solved as

$$\boldsymbol{z}^{(k+1)} = \mathcal{T}\left(\boldsymbol{f}^{(k+1)} + \frac{1}{\rho}\boldsymbol{\beta}, \frac{\lambda}{\rho}\right), \tag{2.30}$$

where $\mathcal{T}$ is the soft thresholding operator defined for any $\boldsymbol{f}$ by

$$\mathcal{T}\left(\boldsymbol{f}, \frac{\lambda}{\rho}\right) = \text{sign}\left(\boldsymbol{f}\right) \cdot \max\left(|\boldsymbol{f}| - \frac{\lambda}{\rho}, \boldsymbol{0}\right). \tag{2.31}$$

Similarly, the total variation's proximal mapping can be computed through a combination of shrinkage functions (Parikh, 2014). If $g(\boldsymbol{f}) = \lambda(\|\boldsymbol{D}_x\boldsymbol{f}\|_1 + \|\boldsymbol{D}_y\boldsymbol{f}\|_1)$ ($\lambda > 0$), Eq. 2.29b can be solved as

$$\boldsymbol{z}^{(k+1)} = \mathcal{T}(\boldsymbol{D}_x\boldsymbol{f}^{(k+1)} + \frac{1}{\rho}\boldsymbol{\beta}, \frac{\lambda}{\rho}) + \mathcal{T}(\boldsymbol{D}_y\boldsymbol{f}^{(k+1)} + \frac{1}{\rho}\boldsymbol{\beta}, \frac{\lambda}{\rho}). \tag{2.32}$$

Other popular proximal algorithms can also be used to solve Eq. 2.26, such as the primal-dual algorithm (Chambolle and Pock, 2010), or the split Bregman algorithm (Goldstein and Osher, 2009).

Conclusion

This chapter introduces the Bayesian approach and philosophy for solving linear inverse problems for image processing tasks. We showed the necessity for computing point estimators of the posterior distribution, such as the MAP and conditional expectations of the posterior. We also went through some examples of computing MAP estimates. Linear inverse problems have two main challenges. First, while Gaussian priors allow an analytical solution for Bayesian inversion, nonGaussian priors do not. Therefore, inverse problems with nonGaussian priors are usually limited to point estimations of the prior. Secondly, the probability density function of the set of natural images is not known analytically, and we usually need to approximate it. In Chapter 3, we will see how deep learning can be used to compute some estimators of the conditional expectation of the posterior distribution, given some training samples of the posterior. And in Chapter 4, we will explore some extension of these concepts to time series.

Chapter 3

Deep-learning reconstruction for linear inverse problems

Deep learning is being increasingly used to solve image-processing tasks. In particular, deep-learning methods have shown results that have exceeded most traditional methods for computing point-wise estimators of random variables. In this chapter, we review deep learning for the purpose of solving linear inverse problems. In particular, Section 3.1 shows how the choice of the cost function of deep neural networks allows us to compute a family of Bayesian point-wise estimators. In Section 3.2, we go through the different core structures that are common to most deep neural networks. Then in Section 3.3, we cover some of the theory behind the training of deep architectures. Finally in Section 3.4, we go through some of the empiric practices behind training deep neural networks.

3.1 Deep-learning-based reconstruction for inverse problems

We consider the inverse problem presented in Chapter 2

$$\boldsymbol{m} = \boldsymbol{A}\boldsymbol{f} + \boldsymbol{\epsilon}, \tag{3.1}$$

$$\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_\alpha). \tag{3.2}$$

This inverse problem can be used to model the single-pixel acquisition presented in Eq. 1.28. As indicated in Section 2.2.2, when full Bayesian inversion is not feasible, an estimator of $\boldsymbol{f}$ can be obtained via the conditional expectation $\mathbb{E}(\mathbf{f}|\boldsymbol{m})$.

Direct reconstruction methods using deep learning aim to provide a mapping $\mathcal{G}$ that estimates the conditional expectation

$$\mathcal{G}(\boldsymbol{m}) = \mathbb{E}(\mathbf{f}|\boldsymbol{m}), \tag{3.3}$$

To obtain this estimator, there is the need to solve (see annex C)

$$\underset{\mathcal{G}:\mathbb{R}^M \mapsto \mathbb{R}^N}{\operatorname{argmin}} \; \mathbb{E}\left[\|\mathcal{G}(\boldsymbol{m}) - \boldsymbol{f}\|^2\right]. \tag{3.4}$$

For general distributions of $\mathbf{f}$, the mapping $\mathcal{G}^*$ is nonlinear, and there are no closed-form solutions. Instead of solving Eq. (3.4), which is computationally intractable in general, deep-learning-based methods replace the expectation by the empirical mean over a database, and optimize a mapping $\mathcal{G}_{\boldsymbol{\theta}}$ within a family of mappings parameterized by some weights $\boldsymbol{\theta} \in \mathbb{R}^P$.

Feed-forward deep neural networks are a natural choice, as the universal approximation theorem (Hornik, Stinchcombe, and White, 1989) establishes that it can approximate any Borel-measurable function from a finite dimensional space to another

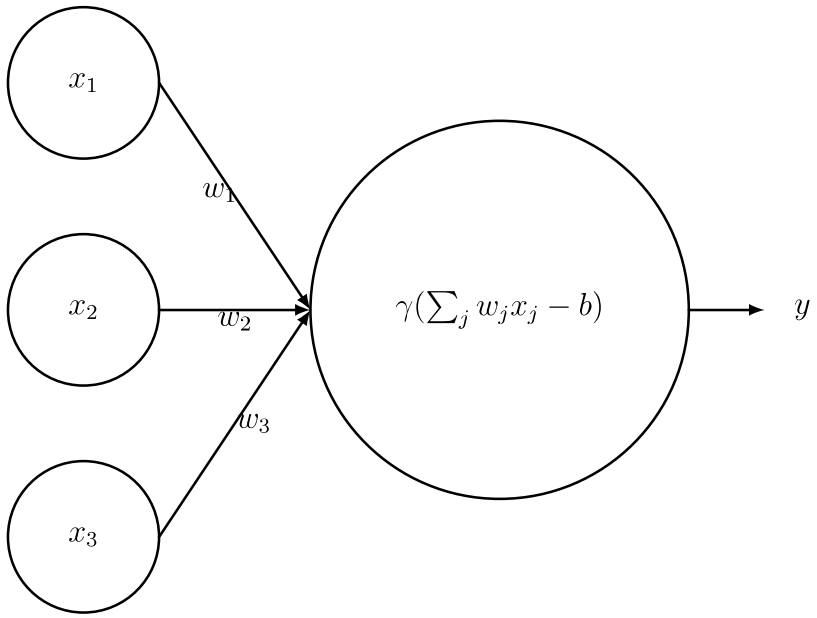


FIGURE 3.1 – Rosenblatt perceptron with three inputs x_1, x_2, x_3. The output is generated by applying a nonlinear activation function γ to the weighted sum $\sum_j w_j x_j - b$.

with any degree of precision. Replacing the expectation with its empirical counter part is justified by the law of large numbers, according to which the empirical expectancy converges to the actual expectation as the number of samples tends to infinity.

Therefore deep-learning estimators of the conditional expectation are approximated by solving

$$\mathcal{G}_\theta^* \in \underset{\theta \in \mathbb{R}^P}{\operatorname{argmin}} \frac{1}{S} \sum_{i=0}^{S-1} \left[\|\mathcal{G}_\theta(\boldsymbol{m}_i) - \boldsymbol{f}_i\|^2 \right] \tag{3.5}$$

where $(\boldsymbol{f}_i, \boldsymbol{m}_i)_{i \in \{0,\dots,S-1\}}$ are samples of $(p(\mathbf{f}), p(\mathbf{m}|\mathbf{f}))$ obtained from an image database.

3.2 Fundamental structures of deep learning

3.2.1 The perceptron

The perceptron was first introduced by Frank Rosenblatt (Rosenblatt, 1961) as a simple decision-making model. It takes several binary inputs $x_1, \dots, x_v$ and produces a single binary output y. Rosenblatt designed a simple rule to compute the output. By introducing weights $w_1, \dots, w_v$ that weigh the respective importance for each input, and a single bias b, the output is obtained by comparing the weighted sum to the bias :

$$y = \begin{cases} 0 & \text{if } \sum_{j=1}^{v} w_j x_j - b \leq 0, \\ 1 & \text{else.} \end{cases} \tag{3.6}$$

Despite its simplicity, stacking several layers of perceptrons allows complex operations to be achieved. For example, the NAND operation, which is a fundamental

structure of many complex electronic systems, can be explicitly implemented via perceptrons with fixed weights (each weight would have a value of -2, and the bias would be 3 to produce the NAND gate).

Due to the discontinuity of the thresholding operation that is applied to the weighed sum, this perceptron would not be suitable for algorithms that solve Eq. 3.5. The thresholding operation is replaced by a nonlinear continuous function γ, called the "activation function". Classic examples of this function would be the sigmoid function, the hyperbolic tangent, and the rectified linear unit (ReLU). As shown in Fig. 3.1, the output y for a perceptron would therefore be computed as :

$$y = \gamma(\sum_{j=1}^{v} w_j x_j - b). \tag{3.7}$$

3.2.2 Artificial neural networks

As shown in Fig. 3.2, the most general way of combining perceptrons is the fully connected layer. Fully connected layers are applied to the output of other fully connected layers, which leads to artificial neural networks with several layers.

A fully connected layer may take as an input a vector $\boldsymbol{a}^l \in \mathbb{R}^{F_l}$, and it outputs a vector in $\boldsymbol{a}^{l+1} \in \mathbb{R}^{F_{l+1}}$. This operation $\boldsymbol{a}^l$ is usually referred to as the hidden state of the l^{th} layer. A fully connected layer is a combination of a linear layer and a nonlinear function. Every component of the output vector is the result of a perceptron applied to the input vector.

$$\boldsymbol{a}^{l+1} = \gamma(\boldsymbol{W}^{l+1}\boldsymbol{a}^l + \boldsymbol{b}^{l+1}), \tag{3.8}$$

where $\boldsymbol{W}^{l+1}$ is the weight matrix that contains the different weights $(w_{j,k})_{1\leq j\leq F_{l+1}, 1\leq k\leq F_l}$ of all of the different neurons, and $\boldsymbol{b}^{l+1}$ is the bias vector that contains all of the biases $(b_j^{l+1})_{1\leq j\leq \leq F_{l+1}}$. This means that a fully connected layer with F_{l+1} outputs and F_l inputs has $F_{l+1} \times F_l$ weights, and F_{l+1} biases.

For feed-forward neural networks, we will note for the rest of the chapter w_{jk}^l, as the value of the k^{th} weight of the j^{th} neuron of the l^{th} layer, and b_j^l, as the bias of the j^{th} neuron in the l^{th} layer.

3.2.3 Convolutional layers

Convolutional neural networks (LeCun, 1989) have shown outstanding performance when solving imaging inverse problems, such as denoising (Zhang et al., 2017a), super-resolution (Dong et al., 2015), and deconvolution (Xu et al., 2014), as well as for many biomedical imaging inverse problems, such as computed tomography (Kang, Min, and Ye, 2017) and accelerated magnetic resonance imaging (MRI) (Jin et al., 2017). These are therefore the backbone of many neural networks. The theory behind convolutional layers is linked with the theory of framelets (Yin et al., 2017; Yoo, Wahab, and Ye, 2018), whereby the convolutional layers serve as a linear transform to a linear space where the inverse problem is easier to solve.

The two main ideas for the implementation of convolutional layers are the use of local receptive fields and the use of shared weights and biases across the different receptive fields.

— **Local receptive fields**: For fully connected layers, inputs and hidden states are represented as a vector of neurons all displayed in a vertical line. In convolutional networks, it is more helpful to represent the inputs as squared matrices

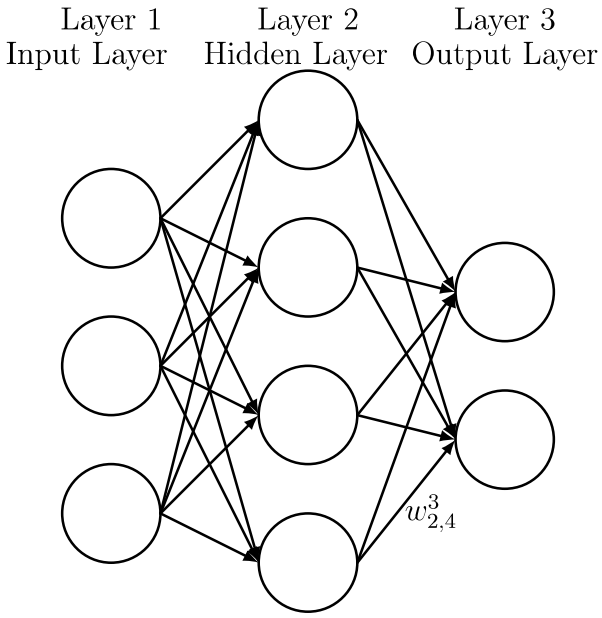


FIGURE 3.2 – Neural network with two fully connected layers. The input is of dimension 3, and the output is of dimension 2. $w^l_{j,k}$ is the weight from the k^{th} neuron in the $(l-1)^{\text{th}}$ layer to the j^{th} neuron in the l^{th} layer.

of neurons. As shown in Fig. 3.3, the inputs are then connected to a layer of hidden neurons, but unlike fully connected layers, each neuron is only connected to a few neurons in a square region in the layer (called receptive fields). The local perceptive field is moved throughout the whole image to produce the hidden layer.

— **Shared weights and biases**: Every neuron in a layer is connected to a limited number of pixels from the previous layer through its local receptive field, and its weights and biases are shared by all of the neurons in the hidden layer. The output of the $(j,k)^{\text{th}}$ neuron of the $(l+1)^{\text{th}}$ layer is

$$\boldsymbol{a}^{l+1} = \gamma(b^l + \boldsymbol{w}^l * \boldsymbol{a}^l), \tag{3.9}$$

where $*$ is the two-dimensional convolution operation. This weight sharing property of the convolutional layers also means that all of the neurons detect the same feature. Therefore the hidden layers of convolutional neural networks are called *feature maps*.

As shown in Fig. 3.4, convolutional layers can produce several feature maps by using several sets of convolutional kernels. To simplify their representation, hidden states of convolutional layers are often represented as stacks of two-dimensional images, where each layer is a different feature map, as shown. We note the hidden state of the l^{th} layer $\boldsymbol{a}^l \in \mathbb{R}^{M^l \times h \times w}$, where M^l represents the number of feature maps in the l^{th} hidden state, and h and w represent the height and width of every feature map. Finally, the output of a convolutional layer that takes a hidden state with M^l feature

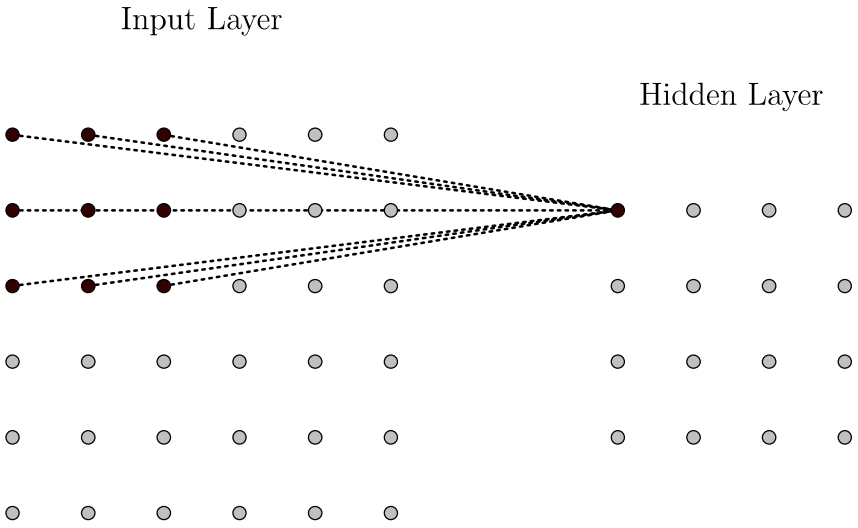


FIGURE 3.3 – Illustration of the concept of local perceptive fields for convolutional layers between an input layer and a hidden layer. In this case, we show an 8×8 input image, and our local perceptive field is of size 3×3.

maps as an input, and has M^{l+1} feature maps as an output, is

$$\boldsymbol{a}^{l+1,m} = \gamma(b^{l,m} + \sum_{\tilde{m}=0}^{M^l} \boldsymbol{w}^{l,m,\tilde{m}} * \boldsymbol{a}^{l,\tilde{m}}), \quad \forall 0 \leq m \leq M^{l+1}. \tag{3.10}$$

3.2.4 Pooling layers

In addition to convolutional layers, convolutional neural networks can also have pooling layers. Their purpose is to simplify the information contained within the feature maps, by reducing the dimension of the feature maps.

The pooling layer will therefore output a feature map of reduced dimension, where each element is computed from a region of the input feature map. The most popular pooling layer is the max-pooling layer. As illustrated in Fig. 3.5. A max-pool layer is defined by two parameters: the kernel size and the stride. The kernel size determines how big the region considered within the feature map is. The stride determines how many pixels we skip before reapplying the max-pooling layer. These layers allow us to emulate multi-scale analysis: the image is analyzed on different scales through deep neural networks, similarly to the theory of wavelet analysis.

3.2.5 U-net

Amongst the many possible combinations of fully connected layers, convolutional layers and pooling layers, the U-net architecture is arguably the most popular. Initially introduced by (Ronneberger et al., 2015) for MRI segmentation, it has been used successfully to solve many inverse problems, such as computed tomography (Jin et al., 2017), compressed MRI reconstruction (Sun et al., 2019), and denoising (Gurrola-Ramos, Dalmau, and Alarcón, 2021).

The success of this architecture lies in the two core concepts of the architecture: skipping connections, and multi-scale analysis (see Fig. 3.6). The skipped connections

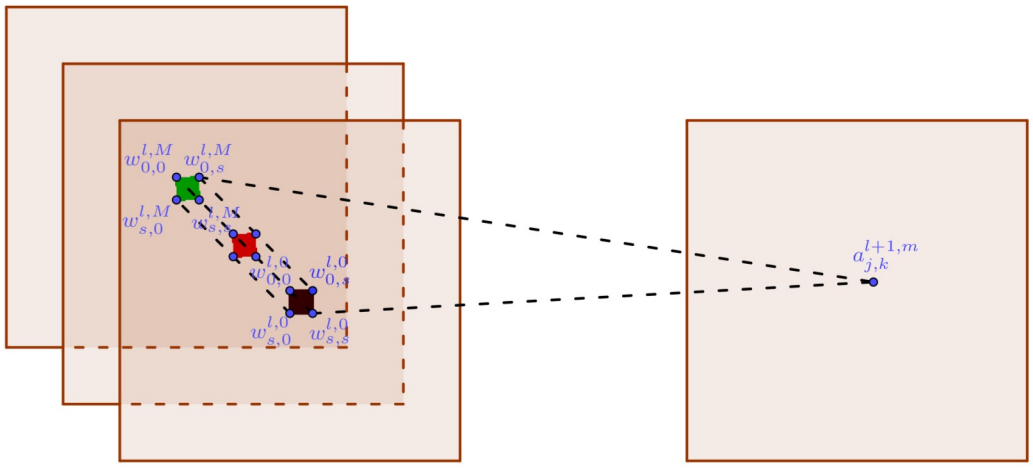


FIGURE 3.4 – Visual representation of a convolutional layer with a kernel size of $s \times s$. $a^{l+1,m}_{j,k}$ represents the output of the j,k^{th} hidden neuron of the l^{th} layer, on the m^{th} feature map.

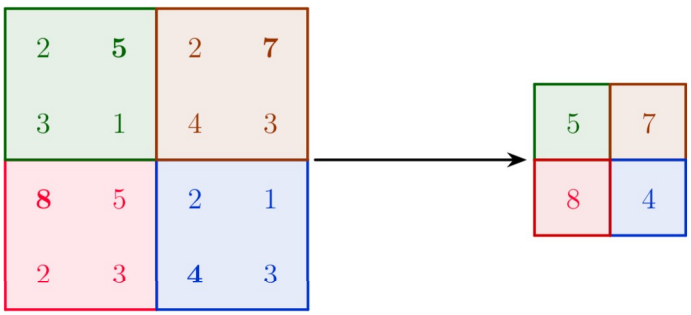


FIGURE 3.5 – Example of a max-pool layer with a kernel size of 2, and a stride of 2. We see that it considers regions within the input image, and considers the maximum of these regions to compute the output down-sampled image.

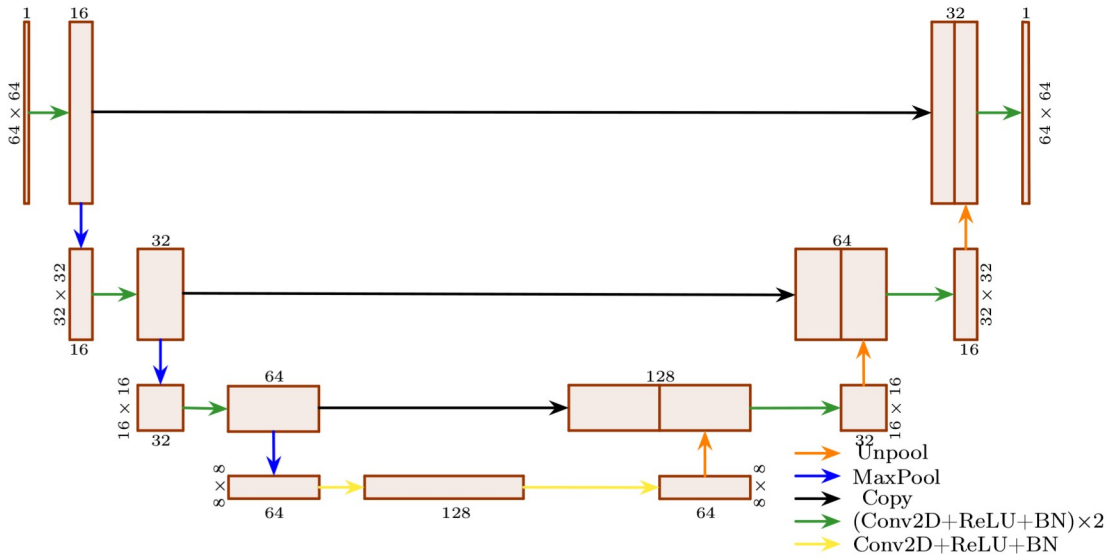


FIGURE 3.6 – Example of a U-net for 64×64 images, with one input channel and one channel output. Each rectangle represents a multi-channel feature map, the depths of which are reported at the top of the feature map.

within the neural network allow the vanishing gradient problem to be avoided (see Section 3.4.1), as it reduces the distance from any given layer to the end of the network. Furthermore, the pooling layers that down-sample the image allow for the image to be analyzed in several levels of detail. The intuition to the down-sampling is that it allows details to be analyzed at several scales, and the skipped connections also make sure that the information on the higher scales is not lost when reducing the scale.

3.2.6 Recurrent layers

When analysing temporal series, the aforementioned architectures are ill-suited, as they do not have any concept of time. Exploiting temporal redundancies and time-series information is typically achieved with recurrent layers that keep a memory state from the previous inputs, called the hidden state. The concept of hidden states is illustrated in Fig. 3.7. They usually contain a cell state $\boldsymbol{c}_{t-1}$, also called the memory state. This memory state is used alongside the input vector $\boldsymbol{x}_t$ to generate two outputs: the output $\boldsymbol{h}_t$ and the next cell state $\boldsymbol{c}_t$.

The two most popular recurrent layers are the long short-term memory (LSTM) cells and the gated recurrent unit (GRU) cells. The core ingredients of these cells are their different latent variables, called "gates". These gates determine how to update the output based on the previous memory state and the current input.

Long short-term memory cells

Long short-term memory cells were first introduced by (Hochreiter and Schmidhuber, 1997) as a solution to prevent vanishing gradient problems when dealing with time series. These cells have shown great performances in modeling time series for speech recognition (Graves, Mohamed, and Hinton, 2013), time series prediction (Schmidhuber, Wierstra, and Gomez, 2005), market prediction (Saiful Islam and Hossain, 2020), and handwriting recognition (Graves et al., 2009).

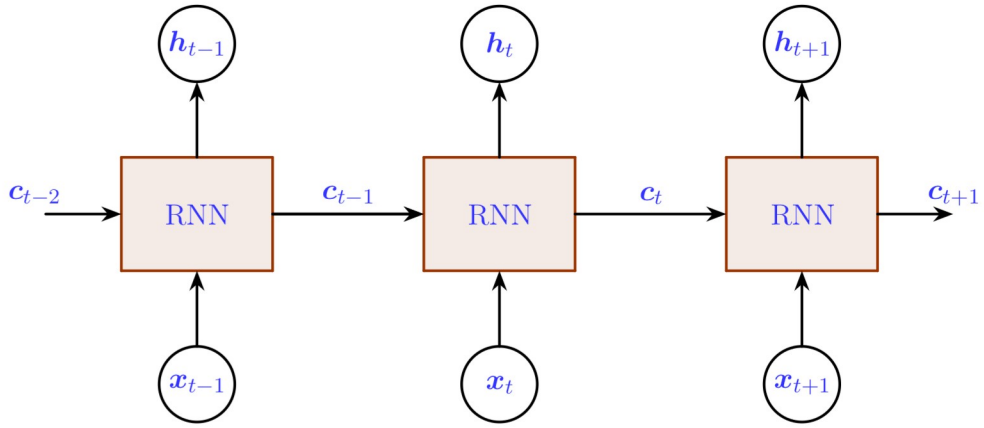


FIGURE 3.7 – Recurrent cell: The weights of the RNN are shared at all times. The recurrent cell takes as an input the cell state in the previous iteration $\boldsymbol{c}_{t-1}$, and the current inputs $\boldsymbol{x}_t$, and outputs a new cell state $\boldsymbol{c}_t$ and the output $\boldsymbol{h}_t$.

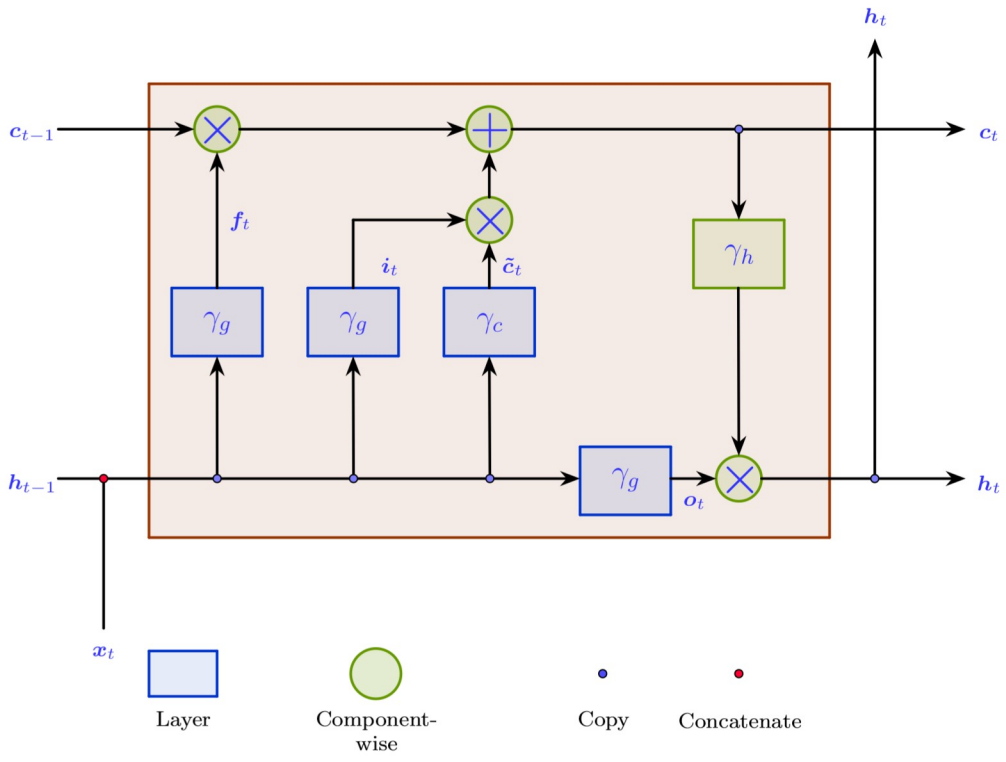


FIGURE 3.8 – Long short-term memory cell. The cell has three inputs: the previous cell state $\boldsymbol{c}_{t-1}$, the previous output $\boldsymbol{h}_{t-1}$, and the current input $\boldsymbol{x}_t$. It outputs the memory state $\boldsymbol{c}_t$ and the output $\boldsymbol{h}_t$.

As illustrated in Fig. 3.8, the LSTM cell has four gates: fully connected layers with inputs from the previous output $\boldsymbol{h}_{t-1}$, and the current input $\boldsymbol{x}_t$. It outputs $\boldsymbol{h}_t$ and a cell state (memory state) $\boldsymbol{c}_t$. The output will be the Hadamard product of the output gate $\boldsymbol{o}_t$ and the current cell state $\boldsymbol{c}_t$ applied to an activation function (γ_h). In other words, the cell state will balance out $\boldsymbol{o}_t$ with our previous data. The state vector is computed based on the previous cell state $\boldsymbol{c}_{t-1}$ and the new candidate cell state $\tilde{\boldsymbol{c}}_t$. The forget gate activation $\boldsymbol{f}_t$ and the input gate output $\boldsymbol{i}_t$ will weigh the importance of the new candidate cell state $\tilde{\boldsymbol{c}}_t$ with respect to the old cell state $\boldsymbol{c}_{t-1}$ to compute the new cell state $\boldsymbol{c}_t$. The different gates, memory state, and output of the cell are computed through the following formula :

$$\boldsymbol{f}_t = \gamma_g(\boldsymbol{W}_f\boldsymbol{x}_t + \boldsymbol{U}_f\boldsymbol{h}_{t-1} + \boldsymbol{b}_f), \tag{3.11}$$
$$\boldsymbol{i}_t = \gamma_g(\boldsymbol{W}_i\boldsymbol{x}_t + \boldsymbol{U}_i\boldsymbol{h}_{t-1} + \boldsymbol{b}_i), \tag{3.12}$$
$$\boldsymbol{o}_t = \gamma_g(\boldsymbol{W}_o\boldsymbol{x}_t + \boldsymbol{U}_o\boldsymbol{h}_{t-1} + \boldsymbol{b}_o), \tag{3.13}$$
$$\tilde{\boldsymbol{c}}_t = \gamma_c(\boldsymbol{W}_c\boldsymbol{x}_t + \boldsymbol{U}_c\boldsymbol{h}_{t-1} + \boldsymbol{b}_c), \tag{3.14}$$
$$\boldsymbol{c}_t = \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \tilde{\boldsymbol{c}}_t, \tag{3.15}$$
$$\boldsymbol{h}_t = \boldsymbol{o}_t \odot \gamma_h(\boldsymbol{c}_t), \tag{3.16}$$

where $\odot$ is the element-wise multiplication. The activation function γ_g is typically chosen as a sigmoid, whereas γ_c is chosen as the hyperbolic tangent function. The popular choices for γ_h are the hyperbolic tangent and the identity function. The natural interpretation to these choices is that we can then imagine the activation of the input and the forget as binary numbers. In doing so, if say $\boldsymbol{f}_t = \boldsymbol{0}$ and $\boldsymbol{i}_t = \boldsymbol{1}$, then according to Eq. 3.15, the new cell state completely forgets about the previous cell state, and is totally replaced by the new candidate cell state. Choosing the hyperbolic tangent function for γ_c, on the other hand, means that the cell state can typically be understood as a binary output with values -1 and 1. In turn, this means that it allows us to negatively weigh the output based on our previous observations.

Long short-term memory cells have been designed for vectors, but they can be adapted to images by replacing the matrix-vector multiplications by convolutions (Sainath et al., 2015). The convolutional variants of the LSTM cells have been successfully applied to video prediction (Xingjian et al., 2015; Burge et al., 2020).

Gated recurrent unit cells

Gated recurrent unit cells were initially introduced by (Cho et al., 2014), and they have shown similar performances to LSTM cells with less memory requirements (Chung et al., 2014). This has made them particularly popular in the later half of the last decade.

As illustrated in Fig. 3.9, GRU cells use only three gates, and use the same latent variable $\boldsymbol{h}_t$ as the memory state and the output. Like LSTM cells, we compute a candidate $\tilde{\boldsymbol{h}}_t$ cell state, and we weight that candidate output while comparing it to the previous cell state $\boldsymbol{h}_{t-1}$. The forget gate $\boldsymbol{z}_t$ is here to compare the candidate cell state with the previous state. The reset gate vector $\boldsymbol{r}_t$, on the other hand, decides if our next candidate cell state needs to completely disregard the previous outputs or not. The different gates and the output of the cells are computed through the

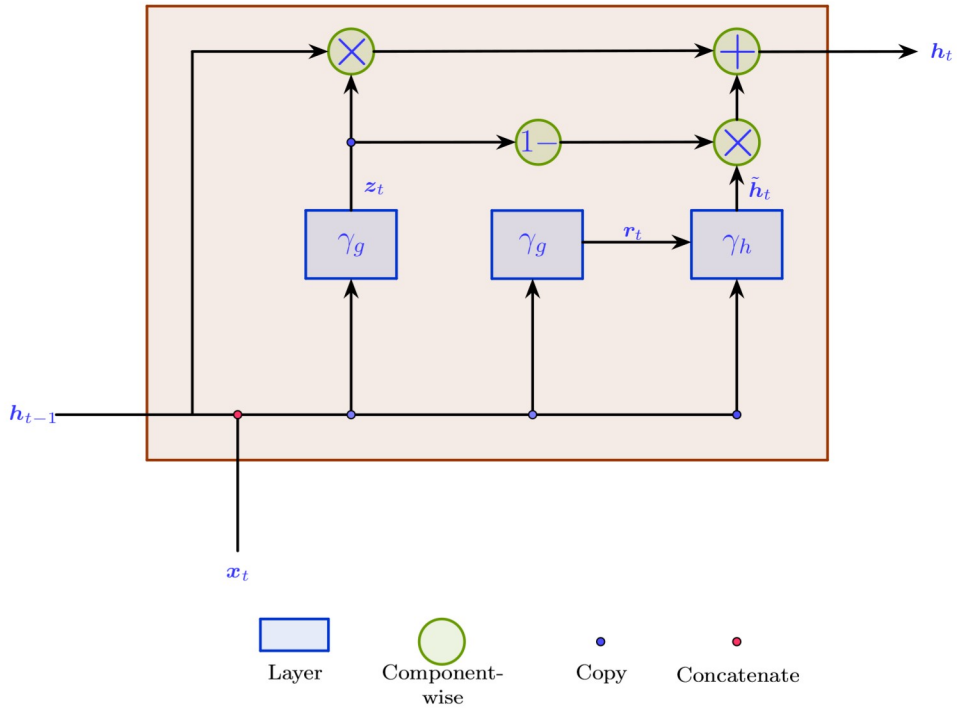


FIGURE 3.9 – Gated recurrent unit cell. The cell has two inputs: the previous output $\boldsymbol{h}_{t-1}$ and the current input $\boldsymbol{x}_t$. It outputs the memory state $\boldsymbol{c}_t$ and the output $\boldsymbol{h}_t$.

following formula

$$\boldsymbol{z}_t = \gamma_g(\boldsymbol{W}_z\boldsymbol{x}_t + \boldsymbol{U}_z\boldsymbol{h}_{t-1} + \boldsymbol{b}_z), \tag{3.17}$$

$$\boldsymbol{r}_t = \gamma_g(\boldsymbol{W}_r\boldsymbol{x}_t + \boldsymbol{U}_r\boldsymbol{h}_{t-1} + \boldsymbol{b}_r), \tag{3.18}$$

$$\tilde{\boldsymbol{h}}_t = \gamma_h(\boldsymbol{W}_h\boldsymbol{x}_t + \boldsymbol{U}_h(\boldsymbol{r}_t \odot \boldsymbol{h}_{t-1}) + \boldsymbol{b}_h), \tag{3.19}$$

$$\boldsymbol{h}_t = (\boldsymbol{1} - \boldsymbol{z}_t) \odot \boldsymbol{h}_{t-1} + \boldsymbol{z}_t \odot \tilde{\boldsymbol{h}}_t. \tag{3.20}$$

Once again, γ_g is typically the sigmoid function. To intuitively understand the GRU cell, we can once again imagine that $\boldsymbol{z}_t$ and $\boldsymbol{r}_t$ are binary. If $\boldsymbol{z}_t = \boldsymbol{0}$, then the new cell state $\boldsymbol{h}_t$ stays the same as it was previously, $\boldsymbol{h}_{t-1}$. On the other hand, if $\boldsymbol{z}_t = \boldsymbol{1}$, then the new cell state $\boldsymbol{h}_t$ is entirely replaced by the new candidate cell state $\tilde{\boldsymbol{h}}_t$. Likewise, if $\boldsymbol{r}_t = \boldsymbol{0}$, then the new candidate cell state $\tilde{\boldsymbol{h}}_t$ is completely independent of the previous cell state $\boldsymbol{h}_{t-1}$.

Similar to LSTM cells, GRU cells have been designed for vectors, although they can be adapted to images by replacing the matrix-vector multiplications by convolutions (Ballas et al., 2015). The convolutional variants of the GRU cells have been applied successfully to video segmentation problems (Siam et al., 2017), as well as to video prediction problems (Oliu, Selva, and Escalera, 2017) and single-pixel video restoration (Lorente Mur, Peyrin, and Ducros, 2020).

3.3 Training deep neural networks

3.3.1 Loss minimization

Deep neural networks are trained through gradient-based optimization methods. Our goal is to solve the following optimization problem:

$$\boldsymbol{\theta}^* \in \underset{\boldsymbol{\theta}\in\mathbb{R}^P}{\operatorname{argmin}}\, \mathcal{L}(\boldsymbol{\theta}), \quad \text{where,} \quad \mathcal{L}(\boldsymbol{\theta}) = \frac{1}{S}\sum_{i=0}^{S-1} \mathcal{C}(\boldsymbol{\theta}, \boldsymbol{m}_i, \boldsymbol{f}_i) + \mathcal{R}(\boldsymbol{\theta}), \tag{3.21}$$

where $\mathcal{R}$ is a regularization term on the network parameters, $\mathcal{C}$ is an error function, and $(\boldsymbol{f}_i, \boldsymbol{m}_i), i \in \{0, \dots, S-1\}$ are samples of $(p(\boldsymbol{f}), p(\boldsymbol{m}|\boldsymbol{f}))$ obtained from an image database. The choice of $\mathcal{C}$ depends on the type of Bayesian estimator we try to estimate. For instance, as shown in Section 3.1, to compute the conditional expectation of $\mathbf{f}$ given $\boldsymbol{m}$, then $\mathcal{C}(\boldsymbol{f}_i, \boldsymbol{m}_i, \boldsymbol{\theta}) = \|\boldsymbol{f}_i - \mathcal{G}_{\boldsymbol{\theta}}(\boldsymbol{m}_i)\|_2^2$. $\mathcal{R}$ is a regularization term for the weights. The idea of this sort of regularization term is similar to that of solving inverse problems; it helps the stabilize the solution, and penalizes solutions that are too large according to a certain metric. Classic choices of $\mathcal{R}$ include the *weight decay* loss $\mathcal{R}(.) = \|.\|_2^2$ that penalizes high values of $\boldsymbol{\theta}$ or *sparsifying losses*, such as $\mathcal{R}(.) = \|.\|_1$, that promote sparsity in network parameters.

3.3.2 Stochastic gradient descent

To find the solution to Eq. 3.21, a classic approach would be to search for the zeros of $\nabla\mathcal{L}$.

As $\mathcal{L}$ is highly nonlinear, this problem does not allow analytical solutions and is solved via iterative gradient-based algorithms. These algorithms compute a sequence $\{\boldsymbol{\theta}^k\}_{k\in\mathbb{N}}$ from an initial estimate $\boldsymbol{\theta}_0$

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - p_k \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}_k), \tag{3.22}$$

where $p_k > 0$ is called the learning rate. The learning rate determines the convergence rate of the iterative algorithm. Note that the convergence towards a global minimum is only guaranteed if the cost function in $\mathcal{L}$ is a convex function.

Eq. 3.22 can be further developed to understand the issues that arise with such an approach

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - p_k \nabla \mathcal{R}_{\boldsymbol{\theta}}(\boldsymbol{\theta}_k) - \frac{p_k}{S}\sum_{i=0}^{S-1} \nabla_{\boldsymbol{\theta}} \mathcal{C}(\boldsymbol{\theta}_k, \boldsymbol{m}_i, \boldsymbol{f}_i). \tag{3.23}$$

Computing the sum of gradients in Eq. 3.23 can be computationally challenging, as it requires evaluation of the gradient for each sample in the dataset. The stochastic gradient descent circumvents this problem by computing an estimate of the averaged gradients $\frac{p_k}{S}\sum_{i=0}^{S-1} \nabla_{\boldsymbol{\theta}} \mathcal{C}(\boldsymbol{\theta}_k, \boldsymbol{m}_i, \boldsymbol{f}_i)$ over a random smaller subset of indices in $\{1, \dots, S\}$. The subsets are of a size $B \leq S$. B is called the batch size, or mini-batch size. In pseudo-code, this can be summarized as follows:

Algorithm 3 Pseudo-code for the stochastic gradient descent

Data: $\{\boldsymbol{f}_i, \boldsymbol{m}_i\}_{0 \leq i \leq S}$
Initialization: $\boldsymbol{\theta}_0 \in \mathbb{R}^P$
Parameters : $B \in \{1, \dots, S-1\}$
while $\boldsymbol{\theta}_k$ has not converged **do**
  Choose a random permutation τ_k of $\{1, \dots, S\}$
  $\boldsymbol{\theta}_{k+1}^{(0)} = \boldsymbol{\theta}_k$
  for $j \in \{1, \dots, \lfloor S/B \rfloor\}$ **do**
    $\boldsymbol{\theta}_{k+1}^{(j)} = \boldsymbol{\theta}_{k+1}^{(j-1)} - p_{k+1} \nabla \mathcal{R}_{\boldsymbol{\theta}}(\boldsymbol{\theta}_{k+1}^{(j-1)}) - \frac{p_{k+1}}{B} \sum_{i=Bj}^{B(j+1)} \nabla_{\boldsymbol{\theta}} \mathcal{C}(\boldsymbol{\theta}_{k+1}^{(j-1)}, \boldsymbol{m}_{\tau_k(i)}, \boldsymbol{f}_{\tau_k(i)})$
  end for
  $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_{k+1}^{(\lfloor S/B \rfloor)}$
  $k = k + 1$
end while

The batch size B provides a compromise between the accuracy of the gradient descent and the computational complexity. For instance, choosing $B = S$ essentially brings us back to the regular gradient descent.

When the learning rates $(p_k)_k$ decrease with an appropriate rate, and under relatively mild assumptions, stochastic gradient descent converges almost certainly to a global minimum when the objective function is convex. Otherwise it converges almost certainly to a local minimum (Kiwiel, 2001).

3.3.3 Backpropagation

Optimisation of Eq. 3.22 is a high dimensional problem. It is possible to compute the partial derivative of $\mathcal{C}$ with respect to a parameter w_i by approximating the derivative with a small value of $\epsilon > 0$

$$\frac{\partial \mathcal{C}}{\partial w_i} \approx \frac{\mathcal{C}(\theta + \epsilon e_i) - \mathcal{C}(\theta)}{\epsilon} \tag{3.24}$$

where e_i is a unit vector with zeros on all of its components except for the j^{th} component. However, this approach would be very slow, as we need to evaluate $\mathcal{C}(\theta + \epsilon e_i)$ as many times as there are parameters in the neural network for every sample. This is not a viable strategy for training a deep neural network over many samples over many epochs. The alternative is the back-propagation algorithm.

Back-propagation computes the gradient of $\mathcal{C}$ by applying a forward and a backward pass of the data through the neural network. For simplicity, we will consider feed-forward networks with L fully connected layers. The idea of back-propagation can be adapted to other types of architectures by adapting the shapes of the matrices for the linear layers and the nonlinear layers. We note $\boldsymbol{W}^l$ as the matrix that contains the weights of the l^{th} layer. $\boldsymbol{b}^l$ is the vector that contains the biases of the l^{th} layer. We introduce the activation vectors $\boldsymbol{a}^l$ of the l^{th} layer as $\boldsymbol{a}^l = \gamma(\boldsymbol{W}^l \boldsymbol{a}^{l-1} + \boldsymbol{b}^l)$, with $\boldsymbol{a}^0 = \boldsymbol{m}$ as the input of the network, and $\boldsymbol{a}^L = \mathcal{G}_\theta$ as the output of the network. We also introduce the weighted input of every neuron $\boldsymbol{z}^l = \boldsymbol{W}^l \boldsymbol{a}^{l-1} + \boldsymbol{b}^l$.

Finally we introduce the error of neuron j in layer l as

$$\delta_j^l = \frac{\partial C}{\partial z_j^l}. \tag{3.25}$$

The back-propagation algorithm for deep neural networks was first introduced in the 1970s (Werbos, 1994), although knowledge of it only became widespread after

an article from (Rumelhart, Hinton, and Williams, 1986). The back-propagation algorithm is based on four fundamental equations (see Appendix D):

$$\boldsymbol{\delta}^L = \nabla_{a^L} C \odot \gamma'(\boldsymbol{z}^L), \tag{3.26}$$

$$\boldsymbol{\delta}^l = (\boldsymbol{w}^{l+1\top} \boldsymbol{\delta}^{l+1}) \odot \gamma'(\boldsymbol{z}^l), \tag{3.27}$$

$$\frac{\partial \mathcal{C}}{\partial b^l_j} = \delta^l_j, \tag{3.28}$$

$$\frac{\partial \mathcal{C}}{\partial w^l_{j,k}} = a^{l-1}_k \delta^l_j. \tag{3.29}$$

Eq. 3.29 and Eq. 3.28 imply that the gradient of $\mathcal{C}$ with respect to $\{\boldsymbol{w}^l\}_{0\leq l\leq L}$ and $\{\boldsymbol{b}^l\}_{0\leq l\leq L}$ can be acquired by computing $\boldsymbol{\delta}^l$ and storing $\boldsymbol{a}^l$. Computing Eq. 3.26 is straightforward, as the partial derivatives $\frac{\partial C}{\partial a^L_j}$ are all easy to obtain, assuming that $\mathcal{C}$ is easy to differentiate with respect to $\boldsymbol{a}^L$. For instance, if $\mathcal{C}(\boldsymbol{m}, \boldsymbol{f}, \boldsymbol{\theta}) = \|\mathcal{G}_{\boldsymbol{\theta}}(\boldsymbol{m}) - \boldsymbol{f}\|_2^2 = \|\boldsymbol{a}^L - \boldsymbol{f}\|_2^2 = \sum_{j=0}^N (f_j - a^L_j)^2$, then the partial derivatives $\frac{\partial C}{\partial a^L_j}$ are $\frac{\partial C}{\partial a^L_j} = 2(a^L_j - f_j)$.

Once Eq. 3.26 has been computed, the different values of $\boldsymbol{\delta}$ across the different layers can be computed iteratively through Eq. 3.27.

In terms of algorithmic complexity, the back-propagation takes about the same number of operations as two forward passes through the neural network. First we need to compute $\boldsymbol{a}^L$ and all of the values of $\{\boldsymbol{z}^l\}_{0\leq l\leq L}$, which means that we need to compute $\mathcal{G}_{\boldsymbol{\theta}}(\boldsymbol{m})$. Then, computing $\boldsymbol{\delta}^l$ from $\boldsymbol{\delta}^{l+1}$ is as computationally demanding as computing $\boldsymbol{a}^{l+1}$ from $\boldsymbol{a}^l$.

3.4 Practical aspects of training deep neural networks

3.4.1 Vanishing or exploding gradients

The vanishing or exploding gradient problems are fairly common problems when training a deep neural network.

If we combine Eq. 3.26 and Eq. 3.27, we get the expression of the error in the first layer

$$\boldsymbol{\delta}^1 = \nabla_{a^L} C \odot \gamma'(\boldsymbol{z}^L) \prod_{l=1}^{L-1} (\boldsymbol{w}^{l+1\top} \boldsymbol{\delta}^{l+1}) \odot \gamma'(\boldsymbol{z}^l). \tag{3.30}$$

With Eq. 3.30, we can see that the gradient of the early layers of the neural network are multiplied L times by a quantity smaller than the maximum value of γ' compared to the late layers. For activation functions, such as the sigmoid function, this means that to compute the gradient in the early layers we multiply our error on the early layers by a quantity lower than $(1/4)^L$ compared to the final layers. For a network with many layers, the early layers of the network might have very low gradients, which makes it hard to train very deep networks.

The exploding gradient can happen for high initial values of $\boldsymbol{w}^l \odot \gamma'(\boldsymbol{z}^l)$ for all of the layers. The product in Eq. 3.30 might give very high initial gradient values to the early layers compared to the final layers.

To avoid the vanishing or exploding gradients, we can normalize the input data and impose some conditions on the initialization of the weights of the neural network.

3.4.2 Data normalization and weight initialization

Data normalization and weight initialization are empirically known to accelerate and stabilize the training of deep neural networks (Bengio, 2012). They aim to ensure that the variance of the different activation vectors $\boldsymbol{a}^l$ remains the same throughout the network.

Data normalization The most popular normalization method us Z-normalization or standardization. This essentially produces input data that is zero-mean with a standard deviation of 1. The Z-normalized version $\hat{\boldsymbol{X}}^l$ of a feature map $\boldsymbol{X}^l$ with B samples and F features is computed as

$$\hat{\boldsymbol{X}}^l_{:,i} = \frac{\boldsymbol{X}^l_{:,i} - \boldsymbol{\mu}_i}{\boldsymbol{\sigma}_i}, \tag{3.31}$$

where $\boldsymbol{X}^l_{:,i}$ is the i^{th} feature map, and $\boldsymbol{X}^l_{j,:}$ is the j^{th} sample, and $\boldsymbol{\mu}_i = \frac{1}{N} \times \sum_{k=1}^{B} (\boldsymbol{X}_{k,i})$ and $\boldsymbol{\sigma}_i = \sqrt{\frac{1}{B-1} \times \sum_{k=1}^{B} (\boldsymbol{X}_{k,i} - \boldsymbol{\mu_i})^2}$. Data normalization is considered as a standard practice within the deep learning community (Bengio, 2012; Bishop, 1995) as it has been shown to improve the stability of the model, to speed up the training (Le Cun, Kanter, and Solla, 1991).

Weight initialization Weight initialization coupled with data normalization aims to maintain the variance of the activation vectors as stable through the network during the first iterations of the stochastic gradient descent. We assume that the input data is normalized with a mean of 0 and a variance of 1. Let us consider any layer l with n_l neurons. If we make the hypothesis that γ' is constant and worth Γ, the variance $\boldsymbol{a}^l$ of the l^{th} activation and the variance of the $(l+1)^{\text{th}}$ activation $\boldsymbol{a}^{l+1}$ are connected through the following formula

$$\mathbb{V}\text{ar}\left(\boldsymbol{a}^{l+1}\right) = \Gamma n_l \, \mathbb{V}\text{ar}\left(\boldsymbol{a}^l\right) \mathbb{V}\text{ar}\left(\boldsymbol{W}^l\right). \tag{3.32}$$

As we want the variance to be constant through the neural network to avoid the vanishing or exploding gradient problems, we want $\mathbb{V}\text{ar}\left(\boldsymbol{a}^{l+1}\right) = \mathbb{V}\text{ar}\left(\boldsymbol{a}^l\right)$. This implies

$$\mathbb{V}\text{ar}\left(\boldsymbol{W}^l\right) = \frac{1}{\Gamma n_l}. \tag{3.33}$$

We also want the variance of the gradient to be the same throughout the training; i.e., $\mathbb{V}\text{ar}\left(\boldsymbol{\delta}^{l+1}\right) = \mathbb{V}\text{ar}\left(\boldsymbol{\delta}^l\right)$. The back-propagation Eq. 3.27 implies that

$$\mathbb{V}\text{ar}\left(\boldsymbol{\delta}^l\right) = \Gamma n_{l+1} \, \mathbb{V}\text{ar}\left(\boldsymbol{\delta}^{l+1}\right) \mathbb{V}\text{ar}\left(\boldsymbol{W}^l\right). \tag{3.34}$$

This means that we would also want to chose weights $\boldsymbol{W}^l$ such that

$$\mathbb{V}\text{ar}\left(\boldsymbol{W}^l\right) = \frac{1}{\Gamma n_{l+1}}. \tag{3.35}$$

As Eq. 3.34 and Eq. 3.33 are mutually exclusive, LeCun et al., 1998 and Glorot and Bengio, 2010 recommended to simply take the harmonic mean as an initial value for $\boldsymbol{W}^l$

$$\mathbb{V}\text{ar}\left(\boldsymbol{W}^l\right) = \frac{2}{\Gamma(n_l + n_{l+1})}, \tag{3.36}$$

and distribute the weights either according to a centered normal distribution with a variance that follows Eq. 3.36, or a centered uniform distribution with a variance that follows Eq. 3.36.

Conclusion

This chapter reports some key aspects to the understanding of deep learning for solving linear inverse problems. In particular, we showed how to compute certain Bayesian estimators, such as the conditional expectation. Furthermore, we also showcased some of the key structures for state-of-the-art deep neural networks, and went through the practical aspects of training deep neural networks.

In recent years, deep learning has consistently outperformed more traditional methods. However, deep learning lacks certain theoretical guarantees. In particular, its performance when few samples are available and the potential lack of convergence when training neural networks have been the most debated topics in the community. For this reason, many researchers have considered deep neural networks as a "black box" that we do not always understand.

Chapter 4

Bayesian filtering and recursive state estimation

As indicated in Chapter 1, we aim to reconstruct single-pixel videos in real time. Video restoration algorithms are based on the idea that the different frames of a video share some similarities that can be exploited to improve the quality of the reconstructed videos. Video restoration problems can be modeled as Bayesian filtering procedures (Lu et al., 2020) that recursively use the previous restored frames as a prior for the current and future frames.

Bayesian filtering is the Bayesian approach to estimate the state of a time-varying system that is indirectly observed through noisy measurements. In this context, the state of the system can be interpreted as the ground truth frames of a video.

In this chapter, we introduce the concepts of Bayesian filtering, and the methods to solve them. In particular, we will see how the Kalman filter constitutes an exact solution of the Gaussian and linear cases. We will also explore different methods for dealing with nonlinearities within the system.

4.1 Probabilistic state space models and Bayesian filtering equations

4.1.1 Definition

We consider a probabilistic state space model as a sequence of conditional probability distributions:

$$\mathbf{f}_t \sim p(\boldsymbol{f}_t|\boldsymbol{f}_{t-1}), \tag{4.1a}$$
$$\mathbf{m}_t^\alpha \sim p(\boldsymbol{m}_t^\alpha|\boldsymbol{f}_t). \tag{4.1b}$$

In this context, $\boldsymbol{f}_t$ is the state of the system at a given time t (i.e., the ground truth of the t^{th} frame of the video), and $\boldsymbol{m}_t^\alpha$ are the different noisy measurements realized on the different frames $\boldsymbol{f}_t$. The probability density function $p(\boldsymbol{f}_t|\boldsymbol{f}_{t-1})$ describes the dynamic model of the system. This type of model has been used for video compressed sensing problems (Ding, Chen, and Wassell, 2014), video inpainting (Bugeau et al., 2010), and video compression artifact reduction (Lu et al., 2018, 2020). The stochastic nature of the dynamics of the system do not necessarily convey some sort of process noise; they can also model the uncertainties in the system dynamics. The probability density function $p(\boldsymbol{m}_t^\alpha|\boldsymbol{f}_t)$ is the measurement model; in the case of single-pixel imaging, this is a Gaussian distribution such that $\log p(\boldsymbol{m}_t^\alpha|\boldsymbol{f}_t) \propto \|\boldsymbol{A}\boldsymbol{f}_t - \boldsymbol{m}_t^\alpha\|^2_{(\boldsymbol{\Sigma}_t^\alpha)^{-1}}$ (see Eq. 1.28).

Given this definition, the states $\{\boldsymbol{f}_t\}_{t\in\mathbb{N}}$ form a Markov chain. This means the current state $\boldsymbol{f}_t$ is independent from happened before the time $t-1$, and that the

previous frame $\boldsymbol{f}_{t-1}$ is independent of the future frames given $\boldsymbol{f}_t$,

$$p(\boldsymbol{f}_t|\boldsymbol{f}_{1:t-1}, \boldsymbol{m}^{\alpha}_{1:t-1}) = p(\boldsymbol{f}_t|\boldsymbol{f}_{t-1}), \tag{4.2}$$
$$p(\boldsymbol{f}_{t-1}|\boldsymbol{f}_{t:T}, \boldsymbol{m}^{\alpha}_{t:T}) = p(\boldsymbol{f}_{t-1}|\boldsymbol{f}_t). \tag{4.3}$$

This also means that our measurements are conditionally independent of the previous states and measurements

$$p(\boldsymbol{m}^{\alpha}_t|\boldsymbol{f}_{1:t}, \boldsymbol{m}^{\alpha}_{1:t-1}) = p(\boldsymbol{m}^{\alpha}_t|\boldsymbol{f}_t). \tag{4.4}$$

4.1.2 Bayesian filtering equations

The Bayesian solution to Eq. 4.1 is the marginal posterior distribution of the state $\boldsymbol{f}_t$ at time t, given all of the history of the measurements that led to time t

$$p(\boldsymbol{f}_t|\boldsymbol{m}^{\alpha}_{1:t}). \tag{4.5}$$

Contrary to the Bayesian inversion methods in Chapter 2, we do not limit ourselves to the current measurements, but also add all the previous measurements.

The recursive equations of Bayesian filtering require two steps: a prediction step where the distribution $p(\boldsymbol{f}_t|\boldsymbol{m}^{\alpha}_{0:t-1})$ is predicted, and the update step where $p(\boldsymbol{f}_t|\boldsymbol{m}^{\alpha}_{0:t})$ is estimated. We also need a prior distribution for $p(\boldsymbol{f}_0)$ to start the algorithm (see Section 2.3 for choices of $p(\boldsymbol{f}_0)$).

The rest of the equations consist of the Chapman-Kolmogorov (Chapman, 1928) equation:

$$p(\boldsymbol{f}_t|\boldsymbol{m}^{\alpha}_{1:t-1}) = \int p(\boldsymbol{f}_t, \boldsymbol{f}_{t-1}|\boldsymbol{m}^{\alpha}_{1:t-1}) d\boldsymbol{f}_{t-1} \tag{4.6a}$$
$$= \int p(\boldsymbol{f}_t|\boldsymbol{f}_{t-1}, \boldsymbol{m}^{\alpha}_{0:t-1}) p(\boldsymbol{f}_{t-1}|\boldsymbol{m}^{\alpha}_{1:t-1}) d\boldsymbol{f}_{t-1} \tag{4.6b}$$
$$= \int p(\boldsymbol{f}_t|\boldsymbol{f}_{t-1}) p(\boldsymbol{f}_{t-1}|\boldsymbol{m}^{\alpha}_{1:t-1}) d\boldsymbol{f}_{t-1}, \tag{4.6c}$$

where Eq. 4.6b stems from the chain rule, and Eq. 4.6c is due to the Markov properties of $\boldsymbol{f}_t$.

Finally, the update equation can be computed using the Bayes rule:

$$p(\boldsymbol{f}_t|\boldsymbol{m}^{\alpha}_{1:t}) = \frac{1}{Z_k} p(\boldsymbol{m}^{\alpha}_{1:t}|\boldsymbol{f}_t) p(\boldsymbol{f}_t|\boldsymbol{m}^{\alpha}_{1:t-1}) \tag{4.7}$$
$$= \frac{1}{Z_k} p(\boldsymbol{m}^{\alpha}_t|\boldsymbol{f}_t) p(\boldsymbol{f}_t|\boldsymbol{m}^{\alpha}_{1:t-1}), \tag{4.8}$$

where $Z_k = \int p(\boldsymbol{m}^{\alpha}_t|\boldsymbol{f}_t) p(\boldsymbol{f}_t|\boldsymbol{m}^{\alpha}_{1:t-1}) d\boldsymbol{f}_t$ and $p(\boldsymbol{m}^{\alpha}_t|\boldsymbol{f}_t) = p(\boldsymbol{m}^{\alpha}_{1:t}|\boldsymbol{f}_t)$, due the conditional independence of $\mathbf{m}^{\alpha}_t$ from the measurement history. Assuming that $p(\boldsymbol{f}_0)$, $p(\boldsymbol{f}_t|\boldsymbol{f}_{t-1})$, and $p(\boldsymbol{m}^{\alpha}_t|\boldsymbol{f}_t)$ are known, we can compute $p(\boldsymbol{f}_t|\boldsymbol{m}^{\alpha}_{1:t})$ from $p(\boldsymbol{f}_{t-1}|\boldsymbol{m}^{\alpha}_{1:t-1})$ iteratively. For a general probability density function, for high dimensional problems, computing the full conditional density function might not be feasible. For these reasons, Bayesian filters rely either on exact formulae of low dimensional parameters that define the full distribution (such as the Kalman filter for the Gaussian linear case), or on sampling strategies to approximate the conditional probability density function.

4.2 Kalman filtering and linear state estimation

In this section, we consider the Gaussian linear case known as the Kalman filter (Kalman et al., 1960). In other words, we consider the specific case where $p(\boldsymbol{f}_t|\boldsymbol{f}_{t-1}) \propto \|\boldsymbol{f}_t - \boldsymbol{F}_{t-1}\boldsymbol{f}_{t-1}\|^2_{\boldsymbol{Q}_{t-1}^{-1}}$, and $p(\boldsymbol{m}_t^\alpha|\boldsymbol{f}_t) \propto \|\boldsymbol{A}_t\boldsymbol{f}_t - \boldsymbol{m}_t^\alpha\|^2_{(\boldsymbol{\Sigma}_t^\alpha)^{-1}}$. This is equivalent to having a system such that

$$\boldsymbol{f}_t = \boldsymbol{F}_{t-1}\boldsymbol{f}_{t-1} + \boldsymbol{q}_{t-1}, \tag{4.9}$$
$$\boldsymbol{m}_t^\alpha = \boldsymbol{A}_t\boldsymbol{f}_t + \boldsymbol{\epsilon}_t^\alpha, \tag{4.10}$$

where $\boldsymbol{q}_{t-1} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_{t-1})$ and $\boldsymbol{\epsilon}_t^\alpha \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma}_t^\alpha)$.

Assuming $\mathbf{f}_{t-1}|\boldsymbol{m}_{1:t-1}^\alpha$ to be Gaussian, $\mathbf{f}_t|\boldsymbol{m}_{1:t-1}^\alpha$ will be Gaussian as a sum of two Gaussian distributions. By the same arguments used in Section 2.3.2.2, the distribution of $\mathbf{f}_t|\boldsymbol{m}_{1:t}^\alpha$ is Gaussian, such that

$$\mathbf{f}_t|\boldsymbol{m}_{1:t-1}^\alpha \sim \mathcal{N}(\boldsymbol{f}_t^-, \boldsymbol{P}_t^-), \tag{4.11}$$
$$\mathbf{f}_t|\boldsymbol{m}_{1:t}^\alpha \sim \mathcal{N}(\boldsymbol{f}_t^+, \boldsymbol{P}_t^+). \tag{4.12}$$

The Gaussian distribution is fully defined by the mean and covariance matrices. So computing $\boldsymbol{P}_t^-$ and $\boldsymbol{f}_t^-$, as well as $\boldsymbol{f}_t^+$ and $\boldsymbol{P}_t^+$, fully characterizes the marginal posterior distribution. Similar to Section 2.3.2.2, these parameters can be fully computed through analytical formulae (see Theorem B.0.3 in Annex B for more details).

4.2.1 Prediction step

Assuming that we can compute $\mathbf{f}_{t-1}|\boldsymbol{m}_{1:t-1}^\alpha \sim \mathcal{N}(\boldsymbol{f}_{t-1}^+, \boldsymbol{P}_{t-1}^+)$ in the previous step, we can compute the mean and covariance as a linear combination of two Gaussian random variables (i.e., $\mathbf{f}_{t-1}$, $\mathbf{q}_{t-1}$) using Eq. 4.10. The prediction step becomes

$$\boldsymbol{f}_t^- = \mathbb{E}(\mathbf{f}_t|\boldsymbol{m}_{1:t-1}^\alpha) = \boldsymbol{F}_{t-1}\boldsymbol{f}_{t-1}^+, \tag{4.13}$$
$$\boldsymbol{P}_t^- = \mathbb{C}\text{ov}(\mathbf{f}_t|\boldsymbol{m}_{1:t-1}^\alpha) = \boldsymbol{F}_{t-1}\boldsymbol{P}_{t-1}^+\boldsymbol{F}_{t-1}^\top + \boldsymbol{Q}_{t-1}. \tag{4.14}$$

4.2.2 Update step

For the update step, given

$$\mathbf{f}_t|\boldsymbol{m}_{1:t-1}^\alpha \sim \mathcal{N}(\boldsymbol{f}_t^-, \boldsymbol{P}_t^-), \tag{4.15}$$
$$\mathbf{m}_t^\alpha|\boldsymbol{f}_t \sim \mathcal{N}(\boldsymbol{A}_t\boldsymbol{f}_t, \boldsymbol{\Sigma}_t^\alpha), \tag{4.16}$$

according to the result in Annex B, the distribution of $\mathbf{f}_t|\boldsymbol{m}_{1:t}^\alpha$ is normally distributed, and the average and covariance are computed via

$$\boldsymbol{f}_t^+ = \mathbb{E}(\mathbf{f}_t|\boldsymbol{m}_{1:t}^\alpha) = \boldsymbol{f}_t^- + \boldsymbol{P}_t^-\boldsymbol{A}_t^\top(\boldsymbol{A}_t\boldsymbol{P}_t^-\boldsymbol{A}_t^\top + \boldsymbol{\Sigma}_t^\alpha)^{-1}[\boldsymbol{m}_t^\alpha - \boldsymbol{A}_t\boldsymbol{f}_t^-], \tag{4.17}$$
$$\boldsymbol{P}_t^+ = \mathbb{C}\text{ov}(\mathbf{f}_t|\boldsymbol{m}_{1:t}^\alpha) = \boldsymbol{P}_t^- - \boldsymbol{P}_t^-\boldsymbol{A}_t^\top(\boldsymbol{A}_t\boldsymbol{P}_t^-\boldsymbol{A}_t^\top + \boldsymbol{\Sigma}_t^\alpha)^{-1}\boldsymbol{A}_t\boldsymbol{P}_t^-. \tag{4.18}$$

4.3 Nonlinear state estimation

When the dynamic and/or measurement models are not linear, and the noise models are not Gaussian, the use of Kalman filters will be ill-suited to describe the evolution of the system. In this section, we describe the alternatives to the Kalman

filter for nonlinear, nonGaussian systems. For the rest of this section, we consider the dynamics of the system to be described by

$$\boldsymbol{f}_t = \mathfrak{F}(\boldsymbol{f}_{t-1}, \boldsymbol{q}_{t-1}), \tag{4.19a}$$
$$\boldsymbol{m}_t^\alpha = \mathfrak{A}(\boldsymbol{f}_t, \boldsymbol{\epsilon}_t^\alpha), \tag{4.19b}$$

where $\boldsymbol{q}_{t-1} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{Q}_{t-1})$ and $\boldsymbol{\epsilon}_t^\alpha \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_t^\alpha)$.

4.3.1 Extended Kalman filter

The extended Kalman filter (Jazwinski, 1970) uses the Taylor first-order approximation to linearize the system of Eq. 4.19.

Prediction step

We consider the Taylor expansion of $\mathfrak{F}$ close to point $(\boldsymbol{f}_{t-1}, \mathbf{0})$. The fundamental assumption that we make is the approximation $\boldsymbol{f}_{t-1} = \boldsymbol{f}_{t-1}^+ + \delta\boldsymbol{f}_{t-1}$, with $\delta\boldsymbol{f}_{t-1} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{P}_{k-1}^+)$ and $\boldsymbol{q}_{t-1}$ small.

The Taylor expansion in the first-order yields

$$\boldsymbol{f}_t \approx \mathfrak{F}(\boldsymbol{f}_{t-1}^+, \mathbf{0}) + \frac{\partial \mathfrak{F}}{\partial \boldsymbol{f}}(\boldsymbol{f}_{t-1}^+, \mathbf{0})\delta\boldsymbol{f}_{t-1} + \frac{\partial \mathfrak{F}}{\partial \boldsymbol{q}}(\boldsymbol{f}_{t-1}^+, \mathbf{0})\boldsymbol{q}_{t-1}. \tag{4.20}$$

Noting $\boldsymbol{F}_{t-1} = \frac{\partial \mathfrak{F}}{\partial \boldsymbol{f}}(\boldsymbol{f}_{t-1}^+, \mathbf{0})$ and $\boldsymbol{L}_{t-1} = \frac{\partial \mathfrak{F}}{\partial \boldsymbol{q}}(\boldsymbol{f}_{t-1}^+, \mathbf{0})$, this can be re-written more simply as

$$\boldsymbol{f}_t \approx \mathfrak{F}(\boldsymbol{f}_{t-1}^+, \mathbf{0}) + \tilde{\boldsymbol{q}}_{t-1}, \tag{4.21}$$

where $\tilde{\boldsymbol{q}}_{t-1} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{F}_{t-1}\boldsymbol{P}_{t-1}^+\boldsymbol{F}_{t-1}^\top + \boldsymbol{L}_{t-1}\boldsymbol{Q}_{t-1}\boldsymbol{L}_{t-1}^\top)$. As Eq. 4.21 implies that we approximate $p(\boldsymbol{f}_t|\boldsymbol{f}_{t-1})$ as a Gaussian distribution, $\mathbf{f}_t|\boldsymbol{m}_{1:t-1}^\alpha$ is also normally distributed.

This leads to the following prediction step by applying the Kalman prediction equations

$$\boldsymbol{f}_t^- = \mathbb{E}(\mathbf{f}_t|\boldsymbol{m}_{1:t-1}^\alpha) = \mathfrak{F}(\boldsymbol{f}_{t-1}, \mathbf{0}), \tag{4.22}$$
$$\boldsymbol{P}_t^- = \mathbb{C}\mathrm{ov}(\mathbf{f}_t|\boldsymbol{m}_{1:t-1}^\alpha) = \boldsymbol{F}_{t-1}\boldsymbol{P}_{t-1}^+\boldsymbol{F}_{t-1}^\top + \boldsymbol{L}_{t-1}\boldsymbol{Q}_{t-1}\boldsymbol{L}_{t-1}^\top. \tag{4.23}$$

Update step

Similar to the prediction step, by applying the Taylor expansion of $\mathfrak{A}$ in point $(\boldsymbol{f}_t^-, \mathbf{0})$, we get

$$\boldsymbol{m}_t^\alpha \approx \mathfrak{A}(\boldsymbol{f}_t^-, \mathbf{0}) + \frac{\partial \mathfrak{A}}{\partial \boldsymbol{f}}(\boldsymbol{f}_t^-, \mathbf{0})\delta\boldsymbol{f}_t^- + \frac{\partial \mathfrak{F}}{\partial \boldsymbol{\epsilon}^\alpha}(\boldsymbol{f}_t^-, \mathbf{0})\boldsymbol{\epsilon}_t^\alpha, \tag{4.24}$$

where $\delta\boldsymbol{f}_t^- \sim \mathcal{N}(\mathbf{0}, \boldsymbol{P}_t^-)$. Noting $\frac{\partial \mathfrak{A}}{\partial \boldsymbol{f}}(\boldsymbol{f}_t^-, \mathbf{0}) = \boldsymbol{A}_t^f$, and $\frac{\partial \mathfrak{F}}{\partial \boldsymbol{\epsilon}^\alpha}(\boldsymbol{f}_t^-, \mathbf{0}) = \boldsymbol{A}_t^\epsilon$, we get the following approximation:

$$\boldsymbol{m}_t^\alpha \approx \mathfrak{A}(\boldsymbol{f}_t^-, \mathbf{0}) + \boldsymbol{A}_t^f\delta\boldsymbol{f}_t^- + \boldsymbol{A}_t^\epsilon\boldsymbol{\epsilon}_t^\alpha. \tag{4.25}$$

This means that we have the following approximation:

$$\begin{bmatrix} \boldsymbol{f}_t \\ \boldsymbol{m}_t^\alpha \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \boldsymbol{f}_t^- \\ \mathfrak{A}(\boldsymbol{f}_t^-, \mathbf{0}) \end{bmatrix}, \begin{bmatrix} \boldsymbol{P}_t^- & \boldsymbol{P}_t^-\boldsymbol{A}_t^{f\top} \\ \boldsymbol{A}_t^f\boldsymbol{P}_t^- & \boldsymbol{A}_t^f\boldsymbol{P}_t^-\boldsymbol{A}_t^{f\top} + \boldsymbol{A}_t^\epsilon\boldsymbol{\Sigma}_{\alpha t}\boldsymbol{A}_t^{\epsilon\top} \end{bmatrix} \right). \tag{4.26}$$

By the Lemma B.0.3, the conditional distribution average and covariance are

$$\boldsymbol{f}_t^+ = \mathbb{E}(\mathbf{f}_t|\boldsymbol{m}_{1:t-1}^\alpha) = \boldsymbol{f}_t^- + \boldsymbol{P}_t^- \boldsymbol{A}_t^{f\top}(\boldsymbol{A}_t^f \boldsymbol{P}_t^- \boldsymbol{A}_t^{f\top} + \boldsymbol{A}_t^\epsilon \boldsymbol{\Sigma}_t^\alpha \boldsymbol{A}_t^{\epsilon\top})^{-1}[\boldsymbol{m}_t^\alpha - \mathfrak{A}(\boldsymbol{f}_t^-, \mathbf{0})], \tag{4.27}$$

$$\boldsymbol{P}_t^+ = \mathbb{C}\mathrm{ov}(\mathbf{f}_t|\boldsymbol{m}_{1:t}^\alpha) = \boldsymbol{P}_t^- - \boldsymbol{P}_t^- \boldsymbol{A}_t^{f\top}(\boldsymbol{A}_t^f \boldsymbol{P}_t^- \boldsymbol{A}_t^{f\top} + \boldsymbol{A}_t^\epsilon \boldsymbol{\Sigma}_t^\alpha \boldsymbol{A}_t^{\epsilon\top})^{-1} \boldsymbol{A}_t^f \boldsymbol{P}_t^-. \tag{4.28}$$

One of the main advantages of this method is the relative ease of implementation of the equations, as they are essentially the same as the equations for the Kalman filter. This algorithm works better for nonlinear functions where the Hessian matrix is usually not very big compared to its first derivative. For highly nonlinear functions though, the approximation can be ill-suited, in which case it is better to use unscented Kalman filters, or particle filters. Furthermore, this method requires the computation of two Jacobian matrices, which can be challenging to compute. In particular, for high dimensional problems - i.e., problems with high values of N and M - the Jacobians of the problem will be of size $N \times N$ or $M \times N$, which can severely impact on the overall performance for real-time applications.

4.3.2 Unscented Kalman filter

The unscented Kalman filter essentially uses the unscented transform (Julier, Uhlmann, and Durrant-Whyte, 1995) on $\mathfrak{F}$, and $\mathfrak{A}$ to compute the prediction step (the mean and covariance of the predicted variable), as well as the update step. The idea behind the unscented transform is to deterministically determine a fixed number of points - called sigma points - and to use these points to capture the statistics of the transformed Gaussian distribution.

Similar to the extended Kalman filter, the unscented Kalman filter forms a Gaussian approximation of the predictive distribution $p(\boldsymbol{f}_k|\boldsymbol{m}_{1:t-1}^\alpha)$ followed by the posterior distribution $p(\boldsymbol{f}_k|\boldsymbol{m}_{1:t}^\alpha)$:

$$p(\boldsymbol{f}_k|\boldsymbol{m}_{1:t-1}^\alpha) \approx \mathcal{N}(\boldsymbol{f}_t^-, \boldsymbol{P}_t^-), \tag{4.29}$$

$$p(\boldsymbol{f}_k|\boldsymbol{m}_{1:t}^\alpha) \approx \mathcal{N}(\boldsymbol{f}_t^+, \boldsymbol{P}_t^+). \tag{4.30}$$

The algorithm therefore computes $\boldsymbol{f}_t^-$, $\boldsymbol{P}_t^-$, $\boldsymbol{f}_t^+$, and $\boldsymbol{P}_t^+$.

Prediction step

The unscented Kalman algorithm first uses the unscented transform on the function $\mathfrak{F}$ to the augmented random variable $\tilde{\boldsymbol{f}}_{k-1}^+ = (\boldsymbol{f}_{t-1}, \boldsymbol{q}_{t-1})$. The sigma points $\{\tilde{\mathcal{X}}_t^{(i)}\}_{0\leq i\leq 2(n+n_q)}$ of the augmented random variable $\tilde{\boldsymbol{f}}_{k-1}^+ = (\boldsymbol{f}_{t-1}, \boldsymbol{q}_{t-1})$ are first computed:

$$\tilde{\mathcal{X}}_t^{(0)} = \tilde{\boldsymbol{f}}_{t-1}^+, \tag{4.31}$$

$$\tilde{\mathcal{X}}_t^{(i)} = \tilde{\boldsymbol{f}}_{t-1}^+ + \sqrt{(n+n_q)+\lambda}[\sqrt{\tilde{\boldsymbol{P}}_{t-1}^+}]_i \quad \forall i \in \{1,\dots,n+n_q\}, \tag{4.32}$$

$$\tilde{\mathcal{X}}_t^{(i+n+n_q)} = \tilde{\boldsymbol{f}}_{t-1}^+ - \sqrt{(n+n_q)+\lambda}[\sqrt{\tilde{\boldsymbol{P}}_{t-1}^+}]_i \quad \forall i \in \{1,\dots,n+n_q\}, \tag{4.33}$$

where $\tilde{\boldsymbol{f}}_{t-1}^+$ and $\tilde{\boldsymbol{P}}_{t-1}^+$ are the mean and covariance of the augmented variable

$$\tilde{\boldsymbol{f}}_{t-1}^+ = \begin{bmatrix} \boldsymbol{f}_{t-1}^+ \\ \mathbf{0} \end{bmatrix}, \quad \tilde{\boldsymbol{P}}_{t-1}^+ = \begin{bmatrix} \boldsymbol{P}_{t-1}^+ & \mathbf{0} \\ \mathbf{0} & \boldsymbol{Q}_{t-1} \end{bmatrix}. \tag{4.34}$$

$[\sqrt{\tilde{\boldsymbol{P}}_{t-1}^{+}}]_i$ refers to the i^{th} column of a matrix such that $\sqrt{\tilde{\boldsymbol{P}}_{t-1}^{+}}\sqrt{\tilde{\boldsymbol{P}}_{t-1}^{+}} = \tilde{\boldsymbol{P}}_{t-1}^{+}$ (such a matrix exists because $\tilde{\boldsymbol{P}}_{t-1}^{+}$ is a defined positive matrix). λ is a scaling parameter, which is meant to define the spread of the sigma points around the mean (Wan and Merwe, 2001).

We further propagate the sigma points through $\mathfrak{F}$, to compute the transformed sigma points $\hat{\mathcal{X}}_t^{(i)}$:

$$\hat{\mathcal{X}}_t^{(i)} = \mathfrak{F}(\tilde{\mathcal{X}}_t^{(i)}) \qquad \forall i \in \{0, \dots, 2(n+n_q)\}. \tag{4.35}$$

Then the mean and covariance are estimated through the transformed sigma points:

$$\boldsymbol{f}_t^{-} \approx \sum_{i=0}^{2(n+n_q)} W_i^{(m)} \hat{\mathcal{X}}_t^{(i)}, \tag{4.36}$$

$$\boldsymbol{P}_t^{-} \approx \sum_{i=0}^{2(n+n_q)} W_i^{(c)} (\hat{\mathcal{X}}_t^{(i)} - \boldsymbol{f}_t^{-})(\hat{\mathcal{X}}_t^{(i)} - \boldsymbol{f}_t^{-})^{\top}, \tag{4.37}$$

Where the weights $W_i^{(m)}, W_i^{(c)}$ are computed as follows: (Wan and Merwe, 2001) :

$$W_0^{(m)} = \frac{\lambda}{\lambda + n + n_q}, \tag{4.38}$$

$$W_0^{(c)} = \frac{\lambda}{\lambda + n + n_q}(1 + \beta), \tag{4.39}$$

$$W_i^{(m)} = W_i^{(c)} = \frac{1}{2(\lambda + n + n_q)} \qquad \forall i \in \{0, \dots, 2(n+n_q)\}, \tag{4.40}$$

where β is an additional hand-tuned parameter.

Update step

The unscented transform is then used on the function $\mathfrak{A}$, to the augmented random variable $\tilde{\boldsymbol{f}}_t^{-} = (\boldsymbol{f}_t, \boldsymbol{\epsilon}_t^{\alpha})$. The sigma points $\{\tilde{\mathcal{Y}}_t^{(i)}\}_{0 \le i \le 2(n+n_q)}$ of the augmented random variable $\tilde{\boldsymbol{f}}_t^{-} = (\boldsymbol{f}_t, \boldsymbol{\epsilon}_t^{\alpha})$ are first computed :

$$\tilde{\mathcal{Y}}_t^{(0)} = \tilde{\boldsymbol{f}}_t^{-}, \tag{4.41}$$

$$\tilde{\mathcal{X}}_t^{(i)} = \tilde{\boldsymbol{f}}_t^{-} + \sqrt{(n+n_q) + \lambda}[\sqrt{\tilde{\boldsymbol{P}}_t^{-}}]_i \quad \forall i \in \{1, \dots, n+n_q\}, \tag{4.42}$$

$$\tilde{\mathcal{X}}_t^{(i+n+n_q)} = \tilde{\boldsymbol{f}}_t^{-} - \sqrt{(n+n_q) + \lambda}[\sqrt{\tilde{\boldsymbol{P}}_t^{-}}]_i \quad \forall i \in \{1, \dots, n+n_q\}, \tag{4.43}$$

where $\tilde{\boldsymbol{f}}_t^{-}$ and $\tilde{\boldsymbol{P}}_t^{-}$ are the mean and covariance of the augmented variable

$$\tilde{\boldsymbol{f}}_t^{-} = \begin{bmatrix} \boldsymbol{f}_{t-1}^{-} \\ \boldsymbol{0} \end{bmatrix}, \quad \tilde{\boldsymbol{P}}_t^{-} = \begin{bmatrix} \boldsymbol{P}_t^{-} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\Sigma}_{\alpha t} \end{bmatrix}. \tag{4.44}$$

We propagate the sigma points through $\mathfrak{A}$ to compute the transformed sigma points $\hat{\mathcal{Y}}_t^{(i)}$

$$\hat{\mathcal{Y}}_t^{(i)} = \mathfrak{A}(\tilde{\mathcal{Y}}_t^{(i)}) \qquad \forall i \in \{0, \dots, 2(n+n_q)\}. \tag{4.45}$$

Then the mean and covariance are estimated through the transformed sigma points :

$$\boldsymbol{f}_t^+ \approx \sum_{i=0}^{2(n+n_q)} W_i^{(m)} \hat{\mathcal{Y}}_t^{(i)}, \tag{4.46}$$

$$\boldsymbol{P}_t^+ \approx \sum_{i=0}^{2(n+n_q)} W_i^{(c)} (\hat{\mathcal{Y}}_t^{(i)} - \boldsymbol{f}_t^+)(\hat{\mathcal{Y}}_t^{(i)} - \boldsymbol{f}_t^+)^\top, \tag{4.47}$$

Where the weights $W_i^{(m)}, W_i^{(c)}$ are the same as in Section 4.3.2.

4.3.3 Particle filtering

Particle filters are Monte-Carlo-based estimations of the posterior probability $p(\boldsymbol{f}_t|\boldsymbol{m}_{1:t}^\alpha)$ density function. They use importance sampling (see Section 2.2.3) to generate samples of the posterior distribution. These samples then allow computing of the expectation estimates of the posterior distribution $p(\boldsymbol{f}_t|\boldsymbol{m}_{1:t}^\alpha)$.

We will present the sequential importance sampling method (Godsill, Doucet, and West, 2004) upon which many other particle filters are based. The sequential importance sampling algorithm is based on the importance sampling algorithm combined with the Bayesian filtering equations. In other words, we aim to obtain estimators of conditional expectation through some samples drawn from a distribution $\pi(\boldsymbol{f}_t|\boldsymbol{m}_{1:t}^\alpha, \boldsymbol{f}_{1:t-1})$ (from which it is easy to draw samples), such that:

$$\mathbb{E}(\mathcal{F}(\boldsymbol{f}_t)|\boldsymbol{m}_{1:t}^\alpha) \approx \sum_{i=1}^{K} w_t^{(i)} \mathcal{F}(\boldsymbol{f}_t^i), \tag{4.48}$$

which is equivalent to approximating the posterior $p(\boldsymbol{f}_t|\boldsymbol{m}_{1:t}^\alpha)$ by

$$p(\boldsymbol{f}_t|\boldsymbol{m}_{1:t}^\alpha) \approx \sum_{i=1}^{K} w_t^{(i)} \delta(\boldsymbol{f}_t - \boldsymbol{f}_t^{(i)}). \tag{4.49}$$

Let us consider the full posterior distribution of states $\boldsymbol{f}_{0:t}$ given the measurements $\boldsymbol{m}_{1:t}^\alpha$:

$$\begin{aligned} p(\boldsymbol{f}_{1:t}|\boldsymbol{m}_{1:t}^\alpha) &\propto p(\boldsymbol{m}_t^\alpha|\boldsymbol{f}_{0:t}, \boldsymbol{m}_{1:t}^\alpha) p(\boldsymbol{f}_{0:t}|\boldsymbol{m}_{1:t-1}^\alpha), && \text{(4.50a)} \\ &= p(\boldsymbol{m}_t^\alpha|\boldsymbol{f}_{0:t}, \boldsymbol{m}_{1:t}^\alpha) p(\boldsymbol{f}_t|\boldsymbol{f}_{0:t-1}\boldsymbol{m}_{1:t-1}^\alpha) p(\boldsymbol{f}_{0:t-1}|\boldsymbol{m}_{1:t-1}^\alpha), && \text{(4.50b)} \\ &= p(\boldsymbol{m}_t^\alpha|\boldsymbol{f}_t) p(\boldsymbol{f}_t|\boldsymbol{f}_{t-1}) p(\boldsymbol{f}_{0:t-1}|\boldsymbol{m}_{1:t-1}^\alpha). && \text{(4.50c)} \end{aligned}$$

Eq. 4.50a is due to the Bayes rule, Eq. 4.50b is due to the chain rule, and Eq. 4.50c is due to the Markov properties of our model.

By the same logic as in Section 2.2.3, we know that the weights $w_t^{(i)}$ used for the importance sampling approximation need to be computed as

$$w_t^{(i)} = \frac{p(\boldsymbol{f}_{1:t}^{(i)}|\boldsymbol{m}_{1:t}^\alpha)}{\pi(\boldsymbol{f}_{1:t}^{(i)}|\boldsymbol{m}_{1:t}^\alpha)} \tag{4.51}$$

$$w_t^{(i)} \propto \frac{p(\boldsymbol{m}_t^\alpha|\boldsymbol{f}_t^{(i)}) p(\boldsymbol{f}_t^{(i)}|\boldsymbol{f}_{t-1}^{(i)}) p(\boldsymbol{f}_{0:t-1}^{(i)}|\boldsymbol{m}_{1:t-1}^\alpha)}{\pi(\boldsymbol{f}_{1:t}^{(i)}|\boldsymbol{m}_{1:t}^\alpha)}. \tag{4.52}$$

As we choose the shape of $\pi(\boldsymbol{f}_{1:t}|\boldsymbol{m}_{1:t}^\alpha)$, ideally we would choose them with Markovian properties. That way, through the chain rule,

$$\pi(\boldsymbol{f}_{1:t}|\boldsymbol{m}_{1:t}^\alpha) = \pi(\boldsymbol{f}_t|\boldsymbol{f}_{1:t-1}, \boldsymbol{m}_{1:t}^\alpha) \pi(\boldsymbol{f}_{1:t-1}|\boldsymbol{m}_{1:t-1}^\alpha), \tag{4.53}$$

$$\pi(\boldsymbol{f}_{1:t}|\boldsymbol{m}_{1:t}^\alpha) = \pi(\boldsymbol{f}_t|\boldsymbol{f}_{t-1}, \boldsymbol{m}_{1:t}^\alpha) \pi(\boldsymbol{f}_{1:t-1}|\boldsymbol{m}_{1:t-1}^\alpha). \tag{4.54}$$

This gives us a recursive expression of the weights $w_t^{(i)}$, which only depend on the previously computed weights $w_{t-1}^{(i)}$

$$w_t^{(i)} \propto \frac{p(\boldsymbol{m}_t^{\alpha}|\boldsymbol{f}_t^{(i)})p(\boldsymbol{f}_t^{(i)}|\boldsymbol{f}_{t-1}^{(i)})p(\boldsymbol{f}_{0:t-1}^{(i)}|\boldsymbol{m}_{1:t-1}^{\alpha})}{\pi(\boldsymbol{f}_t^{(i)}|\boldsymbol{f}_{t-1}^{(i)},\boldsymbol{m}_{1:t}^{\alpha})\pi(\boldsymbol{f}_{1:t-1}^{(i)}|\boldsymbol{m}_{1:t-1}^{\alpha})} = \frac{p(\boldsymbol{m}_t^{\alpha}|\boldsymbol{f}_t^{(i)})p(\boldsymbol{f}_t^{(i)}|\boldsymbol{f}_{t-1}^{(i)})}{\pi(\boldsymbol{f}_t^{(i)}|\boldsymbol{f}_{t-1}^{(i)},\boldsymbol{m}_{1:t}^{\alpha})} w_{t-1}^{(i)}. \tag{4.55}$$

Algorithm 4 Sequential importance sampling

Initialization: Draw $\{\boldsymbol{f}_0^{(i)}\}_{1\leq i\leq S} \sim p(\boldsymbol{f}_0)$, S samples drawn from the prior distribution. Set all the weights $w_0^{(i)} = 1/S$.
for $t \in \{1,\dots,T\}$ **do**
1 - Draw S samples $\{\boldsymbol{f}_t^{(i)}\}_{1\leq i\leq S}$ from the importance distribution:
$\boldsymbol{f}_t^{(i)} \sim \pi(\boldsymbol{f}_t|\boldsymbol{f}_{1:t-1},\boldsymbol{m}_{1:t}^{\alpha})$.
2 - Compute the nonnormalized weights $\tilde{w}_t^{(i)}$
$\tilde{w}_t^{(i)} = \frac{p(\boldsymbol{m}_t^{\alpha}|\boldsymbol{f}_t^{(i)})p(\boldsymbol{f}_t^{(i)}|\boldsymbol{f}_{t-1}^{(i)})}{\pi(\boldsymbol{f}_t^{(i)}|\boldsymbol{f}_{t-1}^{(i)},\boldsymbol{m}_{1:t}^{\alpha})} w_{t-1}^{(i)}$
3 - Normalize the weights $\tilde{w}_t^{(i)}$ to obtain the normalized weights $w_t^{(i)}$ that add up to unity.
end for

Sometimes this algorithm leads to the creation of many samples with near-zero weights. To solve this "degeneracy" problem, re-sampling (Kitagawa, 1996) can be used as a 4^{th} step (see Annex E for some details on potential implementation of the re-sampling step).

When the evaluation of certain quantities such as $p(\boldsymbol{f}_t^{(i)}|\boldsymbol{f}_{t-1}^{(i)})$ is challenging, an alternative to simplify the algorithm is to choose the importance distribution $\pi(\boldsymbol{f}_t|\boldsymbol{f}_{1:t-1},\boldsymbol{m}_{1:t}^{\alpha}) = p(\boldsymbol{f}_t|\boldsymbol{f}_{t-1}^{(i)})$. This variant of the sequential importance sampling - known as a bootstrap filter (Gordon, Salmond, and Smith, 1993) - simplifies the implementation of the algorithm, as we only need to sample from $p(\boldsymbol{f}_t^{(i)}|\boldsymbol{f}_{t-1}^{(i)})$ instead of evaluating it. Despite simplifying the algorithm, more samples are typically required through this choice of importance distribution.

4.4 Conclusion

In this chapter, we explored Bayesian filtering, and the different methods associated with Bayesian filtering. Bayesian filtering aims to compute the posterior distribution of a time-varying system observed through noisy and potentially incomplete measurements. Bayesian filters have exact solutions in the linear Gaussian case through Kalman filters. However for nonlinear systems, there is no general solution. Therefore, nonlinear systems are generally approximated as Gaussian, or are dealt with using sampling Monte-Carlo-based simulations.

Video restoration problems can often be modeled by a Bayesian filtering problem. However, one of the limitations with these methods is the need for a predictive model of the dynamics of the system.

Part II

Contributions

Chapter 5

Single-Pixel Image Reconstruction from Experimental Data Using Neural Networks

Deep learning image reconstruction has shown results comparable to traditional image reconstruction methods. Recent deep learning libraries have also optimised the use of neural networks on graphics processing units making deep learned reconstructors suitable for real-time applications. One of the limiting factors is the "black box" approach to deep learning that makes it hard to understand, and to apply to experimental data.

In this chapter we report on a deep-learning based reconstruction method that combines Tikhonov regularisation with an image-to-image deep learning reconstructor. Applying a Tikhonov regularisation to the raw data before applying deep-learned reconstructor helps generalising to different values of photon counts making its application to data from an experimental setup easier. Some of the material from this chapter comes from two conference proceedings : IEEE ISBI 2020 (Ducros, Lorente Mur, and Peyrin, 2020) where the noiseless case is considered, SPIE photonics Europe 2020 (Lorente Mur et al., 2020), where we present the experimental setup used to validate our algorithms. This chapter also mostly contains an article published in Optics Express in 2021 (Lorente Mur et al., 2021).

abstract

Single-pixel cameras that measure image coefficients have various promising applications, in particular for hyper-spectral imaging. Here, we investigate deep neural networks that when fed with experimental data, can output high-quality images in real time. Assuming that the measurements are corrupted by mixed Poisson-Gaussian noise, we propose to map the raw data from the measurement domain to the image domain based on a Tikhonov regularization. This step can be implemented as the first layer of a deep neural network, followed by any architecture of layers that acts in the image domain. We also describe a framework for training the network in the presence of noise. In particular, our approach includes an estimation of the image intensity and experimental parameters, together with a normalization scheme that allows varying noise levels to be handled during training and testing. Finally, we present results from simulations and experimental acquisitions with varying noise levels. Our approach yields images with improved peak signal-to-noise ratios, even for noise levels that were foreseen during the training of the networks, which makes the approach particularly suitable to deal with experimental data. Furthermore, while this

approach focuses on single-pixel imaging, it can be adapted for other computational optics problems.

5.1 Introduction

Single-pixel imaging is an extreme configuration of computational optics, where a single point detector is used to recover an image (Edgar, Gibson, and Padgett, 2018). Since the seminal work by Duarte and coworkers (Duarte et al., 2008), single-pixel imaging has been successfully applied to fluorescence microscopy (Studer et al., 2012), hyperspectral imaging (Rousset et al., 2018; Arce et al., 2014), diffuse optical tomography (Pian et al., 2017), image-guided surgery (Aguénounon et al., 2019), short-wave infrared imaging (Zhang et al., 2020), and imaging through scattering media (Li et al., 2020a). Single-pixel measurements can be modeled as dot products between an image and some two-dimensional functions that are implemented through a spatial light modulator (Edgar, Gibson, and Padgett, 2018). To limit acquisition times, it is highly desirable to reduce the number of light patterns, which leads to an undetermined inverse problem to recover the image.

In the field of single-pixel imaging, deep learning has been used to unmix the fluorescence intensity and lifetime from time-resolved measurements (Yao et al., 2019; Smith, Ochoa, and Intes, 2020). Higham and coworkers (Higham et al., 2018) proposed a convolutional auto-encoder for single-pixel image reconstruction imaging that outperformed compressed sensing approaches. This network directly maps the measurement vector to the desired image, using a fully connected layer followed by convolutional layers. Several Deep Learning architectures have been proposed since then, to solve single-pixel reconstruction problems. For instance, Li et al. in (Li et al., 2020a) used a similar network to that introduced by (Higham et al., 2018). For measurements with very low signal-to-noise ratios, in (Rizvi et al., 2020) the authors proposed a U-net for highly compressed single-pixel Fourier imaging, while in (Hoshi et al., 2020), a recurrent neural network was used. Further, Li and coworkers (Li et al., 2020b) investigated a conditional generative adversarial network to reconstruct highly compressed data from measurements with random binary patterns.

In the present work, we propose a new deep learning methodology for image reconstruction from single-pixel measurements corrupted by a mixed Poisson-Gaussian noise model. Traditionally, inverse problems with data corrupted by Poisson noise are tackled using variance stabilizing transforms, such as the Anscombe transform (Anscombe, 1948), followed by a Wiener filter (Wiener, 1949). However, the resulting image can be blurred, and in particular for undersampled data. More recent alternatives have been used to solve this issue by exploiting statistical or handmade image priors (Lefkimmiatis and Unser, 2013; Ding et al., 2018; Li, Luisier, and Blu, 2016). Solving the resulting problems is usually prohibitive for real-time applications, as a single reconstructed image can take several seconds. In this regard, deep networks are ideal candidates due to their rapid evaluation. As such, deep learning has been used for single-pixel imaging in the presence of Poisson noise (Liu et al., 2020), although no special modifications were made to the neural network to allow for the Poisson noise.

Many studies have explored many different neural network architectures; however, they usually consider convolutional layers that act in the image domain, while the raw data are in the measurement domain. Moreover, the interpretation of the output of a neural network is still an open question, specifically in the presence of noise where robustness issues can occur, as indicated by (Kim et al., 2018). In particular, in the

domain of single-pixel imaging, a neural network that is not finely tuned to the system can be outperformed by linear reconstructors (Jiao et al., 2020).

5.1.1 Contribution

First, we explore the best way to map the raw data to the image domain, before applying a cascade of convolutional layers. This mapping is traditionally learned or computed through the Moore-Penrose pseudo-inverse. We describe a linear mapping that can be interpreted in terms of traditional image reconstruction. This linear mapping is implemented as the first layer of a (nonlinear) deep neural network. We introduced this idea in (Ducros, Lorente Mur, and Peyrin, 2020) for noiseless data, and here we extend it to noisy data. In particular, we propose to estimate the image intensity, which is unknown in practice, and to use this estimation to approximate the covariance matrix of the measurements.

Next, we describe a machinery that allows a network to be trained in the presence of Poisson noise. We provide a normalization scheme that allows raw data with different orders of magnitudes to be considered, which is mandatory for realistic scenarios where the image intensity is not known.

Finally, we validate our approach by reconstruction of an experimental dataset that we acquired with varying integration times and light fluxes. Upon acceptance, the datasets will be made available alongside implementations of our reconstruction methods in the Python toolbox (SPyRiT (Lorente Mur and Ducros, 2020)).

5.2 Problem, assumptions and limitations

5.2.1 Single-pixel imaging

Let $\boldsymbol{f} \in [0,1]^N$ be the image to acquire. The main idea of compressive optics is to measure $\boldsymbol{m} = \boldsymbol{H}_1 \boldsymbol{f}$ using hardware, and to recover $\boldsymbol{f}$ using software. The system matrix $\boldsymbol{H}_1 \in \mathbb{R}^{M \times N}$, with $M < N$, collects the patterns that are sequentially uploaded on a spatial light modulator, to obtain the measurement vector. The patterns are traditionally chosen from within a basis $\boldsymbol{H} \in \mathbb{R}^{N \times N}$; i.e., $\boldsymbol{H}_1 = \boldsymbol{S}\boldsymbol{H}$ where $\boldsymbol{S} = [\boldsymbol{I}_M, \boldsymbol{0}]$ describes the subsampling strategy. Classical choices for the basis $\boldsymbol{H}$ include Fourier, discrete cosines, wavelets, and Hadamard basis, as discussed in (Ochoa et al., 2018). In the present study, we consider the Hadamard basis. The subsampling strategies can be divided into adaptive and nonadaptive. Adaptive strategies progressively determine $\boldsymbol{S}$ during the acquisition (Rousset et al., 2017), while nonadaptive chose $\boldsymbol{S}$ beforehand. Here, for simplicity, we consider the nonadaptive strategy introduced in (Baldassarre et al., 2016; Higham et al., 2018).

5.2.2 Acquisition model

As Hadamard patterns contain negative values, we adopt a differential strategy (Lorente Mur et al., 2019). We denote by $\hat{\mathbf{m}}_+^\alpha$ the measurements of the positive parts of the patterns, and by $\hat{\mathbf{m}}_-^\alpha$ the measurements of the negative parts. We model noise as a mixture of Poisson and Gaussian distributions (Foi et al., 2008; Rosenberger et al., 2016). The Poisson noise is signal dependant and originates from the discrete nature of the electronic charge, while the signal-independent Gaussian noise accounts for readout and circuit fluctuations. For both the positive and negative measurements, we have

$$\hat{\mathbf{m}}_{+,-}^\alpha \sim K\,\mathcal{P}(\alpha \boldsymbol{H}_1^{+,-} \boldsymbol{f}) + \mathcal{N}(\mu_{\text{dark}}, \sigma_{\text{dark}}^2) \tag{5.1}$$

where $\mathcal{P}$ and $\mathcal{N}$ are the Poisson and Gaussian distributions, K is a constant that represents the overall system gain (in counts/electron), α is the intensity (in photons) of the image (which is proportional to the integration time), μ_{dark} is the dark current (in counts), and σ_{dark} is the dark noise (in counts). We further hypothesize that μ_{dark} and σ_{dark} are independent of the image intensity α.

The normalized measurements $\mathbf{m}^\alpha$ are finally defined as

$$\mathbf{m}^\alpha = (\hat{\mathbf{m}}_+^\alpha - \hat{\mathbf{m}}_-^\alpha)/(\alpha K), \tag{5.2}$$

5.2.3 Deep learning for image reconstruction.

Deep-learning-based methods reconstruct an image $\tilde{\boldsymbol{f}}$ using nonlinear mapping $\tilde{\boldsymbol{f}} = \mathcal{G}_{\boldsymbol{\omega}}(\boldsymbol{m}^\alpha)$ where the weights of the network $\boldsymbol{\omega}$ are optimized with respect to a loss functions; e.g., to minimize the quadratic error over an image database

$$\mathcal{G}_{\boldsymbol{\omega}} = \underset{\boldsymbol{\Omega}}{\operatorname{argmin}} \frac{1}{s} \sum_{i=0}^{s-1} \|\mathcal{G}_{\boldsymbol{\Omega}}(\boldsymbol{m}_i^\alpha) - \boldsymbol{f}_i\|^2. \tag{5.3}$$

where $(\boldsymbol{f}_i)_{0\le i\le s-1}$ are the image samples of the image database, and $(\boldsymbol{m}_i^\alpha)_{0\le i\le s-1}$ are the corresponding normalized noisy measurements given by Equation (5.2). Traditionally, neural networks are made of cascading layers, and can be written as

$$\mathcal{G}_{\boldsymbol{\omega}} = \mathcal{G}_{\boldsymbol{\omega}}^L \circ \ldots \circ \mathcal{G}_{\boldsymbol{\omega}}^1 \tag{5.4}$$

where $\mathcal{G}^\ell$, $1 \le \ell \le L$ is the ℓ-th (nonlinear) layer of the network, and $\circ$ is the function composition. The first layer is generally a fully connected layer that maps the normalized measurements $\boldsymbol{m}^\alpha \in \mathbb{R}^M$ to a raw solution in $\boldsymbol{f}^* \in \mathbb{R}^N$. In Section 5.3.2, we explore different strategies for the design of this layer.

5.3 Theory

5.3.1 Experimental set-up

Our measurements are obtained from the single-pixel camera experimental set-up depicted in Fig. 5.1, as first described by (Lorente Mur et al., 2020). The telecentric lens (Edmund Optics 62901) is positioned such that its image side projects the image of the scene onto the digital micro-mirror device (DMD; vialux V-7001), which is positioned at the object side of the lens. The object is transparent and is illuminated by a LED lamp (Thorlabs LIUCWHA/M00441662). The DMD can implement different light patterns (denoted as $\boldsymbol{H}_1$ in Section 5.2) by reflection of the incident light onto a relay lens, which projects the light into an optical fiber (Thorlabs FT1500UMT 0.39NA). This optical fiber is connected to a compact spectrometer (BWTek exemplar BRC115P-V-ST1). For every object, we sequentially upload onto the DMD all of the $M = 4096$ Hadamard patterns of dimension $N = 64 \times 64$ pixels. We can down-sample the full measurement vector a posteriori to achieve any sampling ratio. To consider different noise levels, we acquire the same object with varying integration times and neutral optical densities.

5.3.2 Mapping of the raw data to the image domain

We propose to use a Tikhonov-regularized interpretable solution (Tarantola, 2005) to map the raw data into the image domain. In particular, we choose

$$\tilde{\boldsymbol{f}} = \mathcal{G}^1(\hat{\boldsymbol{m}}^\alpha) = \boldsymbol{H}^\top \boldsymbol{y}^*, \tag{5.5}$$

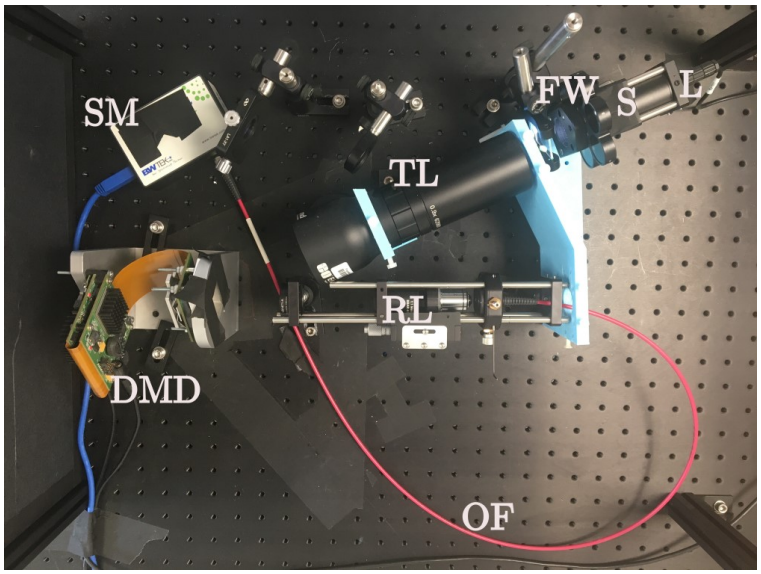


FIGURE 5.1 – Optical set-up of the single-pixel camera (Duarte et al., 2008). This set-up is composed of a sample (S) illuminated by a lamp (L) in front of a filter wheel (FW), a telecentric lens (TL), a digital micro-mirror device (DMD), some relay lenses (RL), an optical fiber (OF), and a spectrometer (SM).

where the regularized data $\boldsymbol{y}^*$ is such that $\boldsymbol{y}^* = [\boldsymbol{y}_1^*, \boldsymbol{y}_2^*]^\top$, where $\boldsymbol{y}_1^* \in \mathbb{R}^M$ and $\boldsymbol{y}_2^* \in \mathbb{R}^{(N-M)}$ are regularized versions of the acquired and missing coefficients, respectively.

Let $\boldsymbol{\Sigma_\alpha}$ be the variance of our noisy measurements, $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are respectively the mean and variance of our prior model in the Hadamard domain. The computation of $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ is described in Section 5.3.3, and that of $\boldsymbol{\Sigma_\alpha}$ in Section 5.3.4. We derived the analytical Tikhonov regularized solution given by

$$\boldsymbol{y}_1^*(\boldsymbol{m}^\alpha) = \boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_1[\boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_\alpha]^{-1}(\boldsymbol{m}^\alpha - \boldsymbol{\mu}_1), \tag{5.6a}$$

$$\boldsymbol{y}_2^*(\boldsymbol{m}^\alpha) = \boldsymbol{\mu}_2 + \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_1^{-1}[\boldsymbol{y}_1^*(\boldsymbol{m}^\alpha) - \boldsymbol{\mu}_1]. \tag{5.6b}$$

where $\boldsymbol{\Sigma}_1 \in \mathbb{R}^{M\times M}$, $\boldsymbol{\Sigma}_{21} \in \mathbb{R}^{(N-M)\times M}$ and $\boldsymbol{\Sigma}_2 \in \mathbb{R}^{(N-M)\times(N-M)}$ are the blocks of the covariance $\boldsymbol{\Sigma}$ in the measurement domain, $\boldsymbol{\mu}_1 \in \mathbb{R}^M$ and $\boldsymbol{\mu}_2 \in \mathbb{R}^{(N-M)}$ are the blocks of $\boldsymbol{\mu}$ (see Section 5.3.3). Interestingly, Equation (5.6a) can be interpreted as the denoising of the raw data in the measurement domain, while Equation (5.6b) is the estimation of the missing coefficients from the denoised acquired coefficients.

To circumvent the difficulty of inverting the signal-dependant matrix in Equation (5.6a), we choose to neglect the nondiagonal terms of $\boldsymbol{\Sigma}_1$;Denoting $\boldsymbol{\sigma}_1^2 = \mathsf{diag}\,(\boldsymbol{\Sigma}_1)$, we get

$$\boldsymbol{y}_1^*(\boldsymbol{m}^\alpha) = \boldsymbol{\mu}_1 + \boldsymbol{\sigma}_1^2/(\boldsymbol{\sigma}_1^2 + \boldsymbol{\sigma}_\alpha^2)(\boldsymbol{m}^\alpha - \boldsymbol{\mu}_1), \tag{5.7}$$

where division and multiplication apply element-wise.

The final proposed method is summarized in Fig. 5.2, where the output of the denoising and completion steps are then corrected by a deep neural network in the image domain.

5.3.3 Prior mean and covariance

We compute the mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$ as introduced in Equation (5.6) from the the samples of an image database. We consider the classical estimators of the

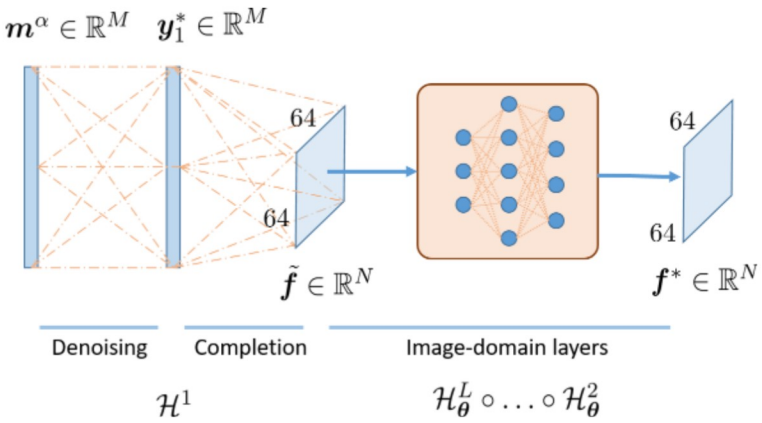


FIGURE 5.2 – Proposed network. The first two layers map the measurements into the image domain according to the analytical solution given by Equation (5.6). These two layers can be interpreted as a layer that denoises Equation (5.6a) of the raw measurements, followed by a layer that estimates the missing coefficients from the denoised measurements of Equation (5.6b). The raw image $\tilde{\boldsymbol{f}}$ is corrected by a cascade of Image-domain convolution layers (CL) and non-linear layers, such as the ReLU layers.

mean and covariance

$$\boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{bmatrix} = \frac{1}{S} \sum_{i=1}^{S} \boldsymbol{H} \boldsymbol{f}_i, \tag{5.8}$$

$$\boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_1 & \boldsymbol{\Sigma}_{21}^\top \\ \boldsymbol{\Sigma}_{21} & \boldsymbol{\Sigma}_2 \end{bmatrix} = \frac{1}{S-1} \sum_{i=1}^{S} (\boldsymbol{H} \boldsymbol{f}_i - \boldsymbol{\mu})(\boldsymbol{H} \boldsymbol{f}_i - \boldsymbol{\mu})^\top. \tag{5.9}$$

Note that both quantities are computed once and for all. We load them into the graphical processor unit when the network is initialized, with no need to re-compute them later (e.g., during training or evaluation).

5.3.4 Noise covariance estimation

To use the mapping proposed in Section 5.3.2, we need an estimation of the covariance matrix of our measurements. Assuming that the raw measurements are independent, from Equation (5.2) we can determine the values of the covariance as

$$\boldsymbol{\Sigma}_\alpha = \mathsf{Diag}\left(\boldsymbol{\sigma}_\alpha^2\right) = \mathsf{Diag}\left(\frac{1}{\alpha} \boldsymbol{H}_1^+ \boldsymbol{f} + \frac{1}{\alpha} \boldsymbol{H}_1^- \boldsymbol{f}\right) + \frac{2\sigma_{\text{dark}}^2}{K^2 \alpha^2} \boldsymbol{I}, \tag{5.10}$$

where $\mathsf{Diag}\,(\boldsymbol{x})$ refers to the diagonal matrix where the diagonal coefficients are the elements of $\boldsymbol{x}$, and $\boldsymbol{I}$ refers to the identity matrix.

As $\boldsymbol{\sigma}_\alpha^2$ depends on the unknown image $\boldsymbol{f}$ as well as on the intensity α, we exploit the raw data that also depends on $\boldsymbol{f}$ and α. Recalling that the variance of a Poisson variable is the same as its expected value, we approximate the expected value by the noisy sample; i.e.,

$$\boldsymbol{\sigma}_\alpha^2 \approx \frac{1}{\alpha^2 K} (\hat{\boldsymbol{m}}_+^\alpha + \hat{\boldsymbol{m}}_-^\alpha - 2\mu_{\text{dark}}) + \frac{2\sigma_{\text{dark}}^2}{K^2 \alpha^2}. \tag{5.11}$$

The experimental parameters μ_{dark}, K and σ_{dark} can be estimated as described in (Rosenberger et al., 2016). We describe how to estimate α in Section 5.3.5.

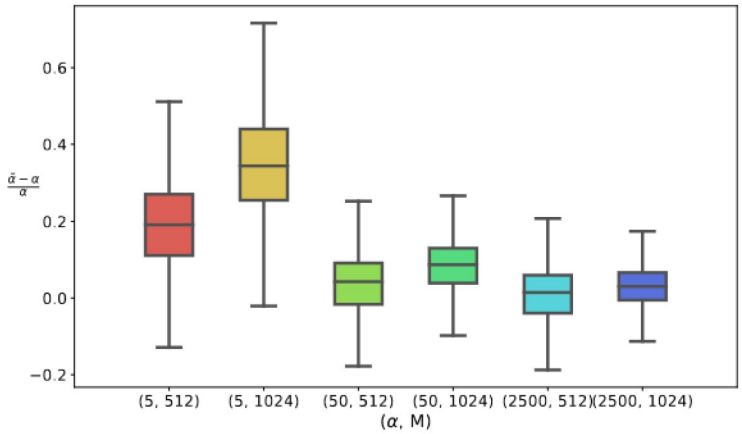


FIGURE 5.3 – Box plot of the distribution of the relative error on the estimation of α using Equation (5.12) with six combinations of values of (α, M).

5.3.5 Estimation of the image intensity

For image denoising, estimation of α can be achieved through fitting methods (e.g., see (Foi et al., 2008)). These methods usually exploit homogeneous regions of the image, and are therefore not suited to raw measurements. Instead, we propose a simple empirical method, which consists of estimating α from a rough estimate of the nonnormalized image $\alpha \boldsymbol{f}$. Considering the pseudo inverse, which is a linear operator that scales with α, we consider

$$\tilde{\alpha} = \max_{i \in \{1,\dots,N\}} (\boldsymbol{H}^\top \boldsymbol{y}^*)_i, \quad \text{with } \boldsymbol{y}^* = \begin{bmatrix} \hat{\boldsymbol{m}}^\alpha \\ \boldsymbol{0} \end{bmatrix} \tag{5.12}$$

In practice, this simple estimator can provide the order of magnitude of α with good approximation, but it obviously leads to some inaccuracies. This is illustrated in Fig. 5.3, where we show the distribution of the error of our estimation over the STL-10 dataset (Coates, Ng, and Lee, 2011) (i.e., 113,000 images obtained by merging the unlabeled, training and testing sets) for different values of M (512, 1024) and α (5, 50, 2500 photons). Overall, the relative error is usually below 50%.

5.4 Experiments

5.4.1 Training neural networks using simulated data

We train our network using the STL10 database (Coates, Ng, and Lee, 2011), with $S = 105,000$ images that correspond to the 'unlabeled' and 'train' subsets. We consider 8,000 images for the testing (i.e., the 'test' subset). The original 96×96 images are resized to 64×64 using bicubic transform, and are normalized between -1 and 1.

We implement the full networks using Pytorch (Paszke et al., 2019) (version 1.5.1; cuda V10.2.89). We train the network by solving Equation (5.3) using the ADAM optimizer (Kingma and Ba, 2014), with an initial learning rate of 10^{-3}, which is halved every 10 epochs, for a maximum of 100 epochs. In our experiments, the validation loss traditionally stops decreasing after 70 epochs. The training phase takes 3 h and 45 min on a NVIDIA GP107GLM [Quadro P1000 Mobile]. As seen in Section 5.3.5, it is not realistic to consider the intensity α as a known constant. Therefore, we make

α vary during the training, with mean μ_α and standard deviation σ_α. Therefore, the network has to learn how to be robust to vary α about μ_α. To fit with the observations made in Fig. 5.3, we choose $\sigma_\alpha = 0.5\mu_\alpha$ to account for the worst-case scenarios. As shown in Section 5.5, varying alpha during the training phase, i.e., considering different noise levels, is crucial to get a robust network.

However, such an approach does not guarantee that a network trained around a given intensity μ_α can perform well for another intensity. A neural network trained for a certain probability distribution cannot generalize 'for free' to another probability distribution. We will further develop this point in Section 5.5.

Our experiments were done with the same Image domain layer structure as in (Higham et al., 2018). This means that our network has three convolutional layers, with each layer separated by a ReLU layer. The first has a kernel size of 9 and a depth of 64, the second has a kernel size of 1 and a depth of 32, and the final one has a kernel size of 5 and a depth of 1. This structure can, however, be changed.

5.4.2 Experimental data

As shown in the first column of Fig. 5.4, we acquire three different objects: the LED lamp directly (first row); a cat from the STL-10 test set printed on a transparent sheet (middle row); and the Siemens Star resolution target (bottom row) (*Photography — Electronic still picture imaging — Resolution and spatial frequency responses* 2019). The ground-truth image is computed (see Section 5.4.3) from a fully sampled measurement vector that is acquired with the highest signal-to-noise ratio (i.e., high flux illumination, long acquisition time). Specifically, we consider no neutral density, and the integration time to 1 ms per pattern for the lamp, to 4 ms per pattern for the STL-10 cat, and to 8 ms per pattern for the Siemens target,

For each object, we also acquire a high-noise dataset by placing neutral optical density behind the lamp, to reduce the light flux. We consider optical densities of 1.3 for the LED lamp, 0.6 for the STL-10 cat, and 0.3 for the Siemens target. We set the integration time to 4 ms for both the LED lamp and the STL-10 cat, and to 8 ms for the Siemens target, we retain only $M = 512$ measurements for both the LED lamp and the STL-10 cat, which accelerates the acquisition by a factor of 8. For the Siemens target, which has a richer spatial frequency content, we keep more measurements ($M = 2048$; acceleration factor of 2).

Our estimated instrumental parameters were : $\mu_{\text{dark}} = 1070$ counts, $K = 1.54$ count/electron, and $\sigma_{\text{dark}} = 53$ counts.

5.4.3 Evaluation metrics

Given the ground-truth image $\boldsymbol{f}$, we compute the peak signal-to-noise ratio (PSNR) of a reconstructed image $\boldsymbol{f}^*$ as

$$\mathsf{PSNR}(\boldsymbol{f}^*, \boldsymbol{f}) = 10 \log_{10} \frac{2^2}{\|\boldsymbol{f}^* - \boldsymbol{f}\|^2} \tag{5.13}$$

For the experimental data, we have no direct access to the ground truth image. This image is computed as $\boldsymbol{f} = \boldsymbol{H}^\top \boldsymbol{y}_{\text{gt}}$, where $\boldsymbol{y}_{\text{gt}}$ is a fully sampled measurement with high signal-to-noise ratio. Before computing the PSNR according to Equation (5.13), we normalize the ground-truth image in the range $[-1, 1]$.

Testing α (in ph.)	Training 50	Training 50 ± 25
50	23.19 ± 1.98	22.13 ± 1.88
50 ± 25	19.33 ± 1.67	21.07 ± 1.63

TABLE 5.1 – Training and testing under varying noise levels. Top row: Data simulated for a given source intensity ($\alpha = 50$ photons), which is the same for all of the test images. Bottom row: Data simulated for test images with varying intensities (mean $\mu_\alpha = 50$ photons, standard deviation $\sigma_\alpha = 25$ photons). Reconstruction PSNRs are reported for a network trained using constant intensity (middle column) and varying intensity (right column).

5.5 Results

5.5.1 Simulated results

5.5.1.0.1 Training for different levels of noise In Table 5.1, we evaluate the effects of training a network under varying noise levels. We also simulate the acquisition of the test images with the same source intensity ($\alpha = 50$ photons) and for varying intensities (mean values $\mu_\alpha = 50$ photons and standard deviation $\sigma_\alpha = 25$ photons). Training a network under different noise levels is detrimental when the image intensity is known; on average, it results in a PSNR drop of $23.19 - 22.13 = 1.06$ dB. On the contrary, noise-varying training improves the reconstruction when the image intensity is unknown; on average, we obtain an enhancement of $21.07 - 19.33 = 1.74$ dB (second row of Table 5.1).

This observation is crucial, as the exact image intensity is not available beforehand in real-life experiments. Although we can estimate the image intensity from the raw data (e.g., by the method introduced in Section 5.3.5), this inevitably leads to inaccuracies. In the following, we only consider realistic scenarios where the image intensity is not known, which requires training with varying noise levels.

In Table 5.2, we report the reconstruction PSNRs obtained using the proposed network, and for a network with the same architecture where the mapping is learned ('Free Layer') for different levels of noise dictated by different image intensities α ($\alpha = \infty$ corresponds to the noiseless version of the proposed network). The standard deviation of the image intensity is set to 50%, in agreement with the findings of Section 5.3.5.

As expected, the network trained with no noise (i.e., $\alpha = \infty$) performs very poorly in the presence of noise. Therefore, we focus our analysis on the cases where our network is trained with noise. We divide our analysis into three cases, which depend on the relative noise levels used during training and testing.

5.5.1.0.2 Training noise and testing noise have the same levels In Table 5.2, the PSNRs of these experiments are shown in blue (i.e., diagonals). In all of our simulations, the proposed method outperforms the 'Free layer' network proposed in (Higham et al., 2018).

5.5.1.0.3 Training noise is higher than testing noise In Table 5.2, the PSNRs of these experiments are shown in red (below lower diagonal). We can observe that

Testing	Training					
	α (in ph.)	∞	2 ± 1	10 ± 5	50 ± 25	2500 ± 1250
Proposed	2 ± 1	9.48 ± 1.78	18.79 ± 1.47	**18.60 ± 1.55**	**18.31 ± 1.55**	**18.14 ± 1.55**
	10 ± 5	13.58 ± 1.94	19.88 ± 1.33	**20.82 ± 1.51**	**20.55 ± 1.61**	**20.21 ± 1.62**
	50 ± 25	15.32 ± 2.06	18.68 ± 1.43	**21.04 ± 1.39**	**21.86 ± 1.54**	**21.69 ± 1.61**
	2500 ± 1250	15.88 ± 2.11	17.40 ± 1.71	19.94 ± 1.59	**21.69 ± 1.51**	**22.17 ± 1.56**
Free Layer	2 ± 1		18.72 ± 1.44	17.93 ± 1.35	17.35 ± 1.37	16.99 ± 1.37
	10 ± 5		19.80 ± 1.64	20.10 ± 1.48	20.04 ± 1.45	19.91 ± 1.42
	50 ± 25		**20.03 ± 1.71**	20.63 ± 1.58	20.81 ± 1.56	20.75 ± 1.53
	2500 ± 1250		**20.09 ± 1.72**	**20.76 ± 1.61**	21.02 ± 1.60	20.97 ± 1.58

TABLE 5.2 – Reconstruction of peak signal-to-noise ratios (PSNRs) for different training strategies. 'Proposed' refers to the neural network method with the proposed mapping in Section 5.3. 'Free Layer' refers to the neural network method where the mapping of the raw measurements is learnt jointly with the postprocessing layers, as in (Higham et al., 2018). Note that the network trained with no noise ($\alpha = \infty$) corresponds to the case where we choose $\boldsymbol{\Sigma}_\alpha = \mathbf{0}$. From top to bottom, image acquisition is simulated assuming increasing light intensity α (i.e., decreasing noise levels). From left to right, images are reconstructed by networks that are trained using decreasing noise levels. Blue font, the testing and training noise levels are the same; green font, the testing noise is lower than the training noise; and red font, the testing noise is higher than the training noise. To facilitate the comparison between the two networks, we underline similar PSNRs (i.e., difference <0.1 dB) and use bold font to indicate the best performing networks (i.e., difference >0.1 dB).

networks trained on high levels of noise (e.g., $\alpha = 2$ or 5 photons) with our proposed method perform poorly when they are tested on data with low levels of noise (e.g., $\alpha = 50$ or 2500 photons). For instance, a neural network trained with $\alpha = 2$ photons yields an average PSNR of 17.40 dB when it is tested for $\alpha = 2500$ photons. This is a PSNR drop of 4.77 dB compared to the optimal training conditions ($\alpha_{\text{train}} = \alpha_{\text{test}} =$ 2500 photons). We can also observe that in these cases the 'Free Layer' (Higham et al., 2018) outperforms the proposed method. We can see that in the most extreme cases where the training was with very high levels of noise, and the testing was with very low levels of noise, the 'Free Layer' network can increase the PSNR by 20.09-17.40 = 2.69 dB on average.

5.5.1.0.4 Training noise is lower than testing noise In Table 5.2, the PSNRs of these experiments are shown in green (above diagonals). We can see that the proposed network behaves similarly to the optimal training conditions. The worst drop in PSNR is relatively low, as 0.71 dB. This shows that the proposed network has high reconstruction quality provided that it is trained with relatively low levels of noise (e.g., $\alpha = 50$ or 2500 photons). Contrary to the previous scenario, the proposed method outperforms the 'Free Layer' network in all of these experiments. Here, the presence of the denoising layer has a great advantage; the proposed method generalizes better to high noise experiments.

5.5.2 Experimental Data

Fig. 5.4 illustrates the performance of our networks on experimental data acquired with the experimental set-up of a single-pixel camera presented in Section 5.3.1.

We compare the performance of our proposed method with the result of the total variation regularized solution (Li, 2010), the Tikhonov-regularized solution of Equation (5.6), the proposed method trained without noise ('Noiseless Net'), and with noise ('Proposed'), as well as the Tikhonov-regularized solution of Equation (5.6) to which we applied a state-of-the-art denoising method BM3D (Dabov et al., 2007) ('Tikhonov+bm3d'), and a neural network reconstructor with the same architecture as the proposed method but where the mapping is learned jointly with the postprocessing layer ('Free Layer'), as in (Higham et al., 2018).

First, we observe that the network trained with no noise performs very poorly on noisy data, and it is outperformed by the pseudo-inverse in the case of the LED lamp and the STL-10 cat. Visually, the reconstructed image retains many of the artifacts introduced by Poisson noise, which emphasizes the need for accounting for noise when training a neural network. Furthermore, we observe that learning the fully connected layer leads to many artifacts and a loss of detail (see the star sector target with the 'Free Layer'). This can be attributed to the learning of the mapping for specific noise values, instead of adapting to the noise, as in our proposed method. We also observe that the Tikhonov-regularized reconstruction is itself quite powerful. Even if it keeps some of the reconstruction artifacts, it compares very well to the results obtained using total variation regularization. In terms of PSNRs, Tikhonov-regularized reconstruction performs very similarly to the 'Free Layer' neural network, which is strong motivation to use it for mapping to the image domain.

Finally, we observe that the proposed network and 'Tikhonov+bm3d' offer the smoothest reconstructed images that are almost free of the artifacts introduced by down-sampling or Poisson noise. In terms of PSNRs, we observe that our method outperforms the others on experimental data, while being similar to 'Tikhonov+bm3d'. However, the proposed method is 1,000 times faster than the latter (0.2 seconds for BM3D versus 0.2 milliseconds for the proposed network). Moreover, 'Tikhonov+bm3d' is a two-step method where denoising is agnostic to reconstruction, while our method solves both problems at once, adapting denoising to the noise level. Although the PSNR gain is relatively low compared to the Tikhonov-regularized reconstruction, the proposed network leads to images that are well enhanced visually.

5.6 Discussion

The proposed mapping generalises much better to higher levels of noise than the learned mapping proposed in (Higham et al., 2018). This is an advantage when dealing with experimental data in real time, as it can be time consuming to dynamically load different neural networks for different lighting conditions. In experimental scenarios, we do not know α, and therefore our proposed mapping with a training value of $\alpha = 2500 \pm 1250$ is the most suited for processing experimental data. Indeed, for all of the testing conditions, that neural network offers a very similar quality of reconstruction to the optimally trained neural network for that noise value.

Our method allows visually convincing reconstructions and good performance with respect to PSNR against several levels of noise, when trained for low levels of noise (high values of α). It therefore shows good proprieties for dealing with experimental data. Unlike previous single-pixel deep-learning studies, such as (Higham et al., 2018; Li, 2010; Rizvi et al., 2020; Hoshi et al., 2020), we have showcased the robustness of our method to different lighting conditions, and given an interpretable meaning to the first layer of our neural network.

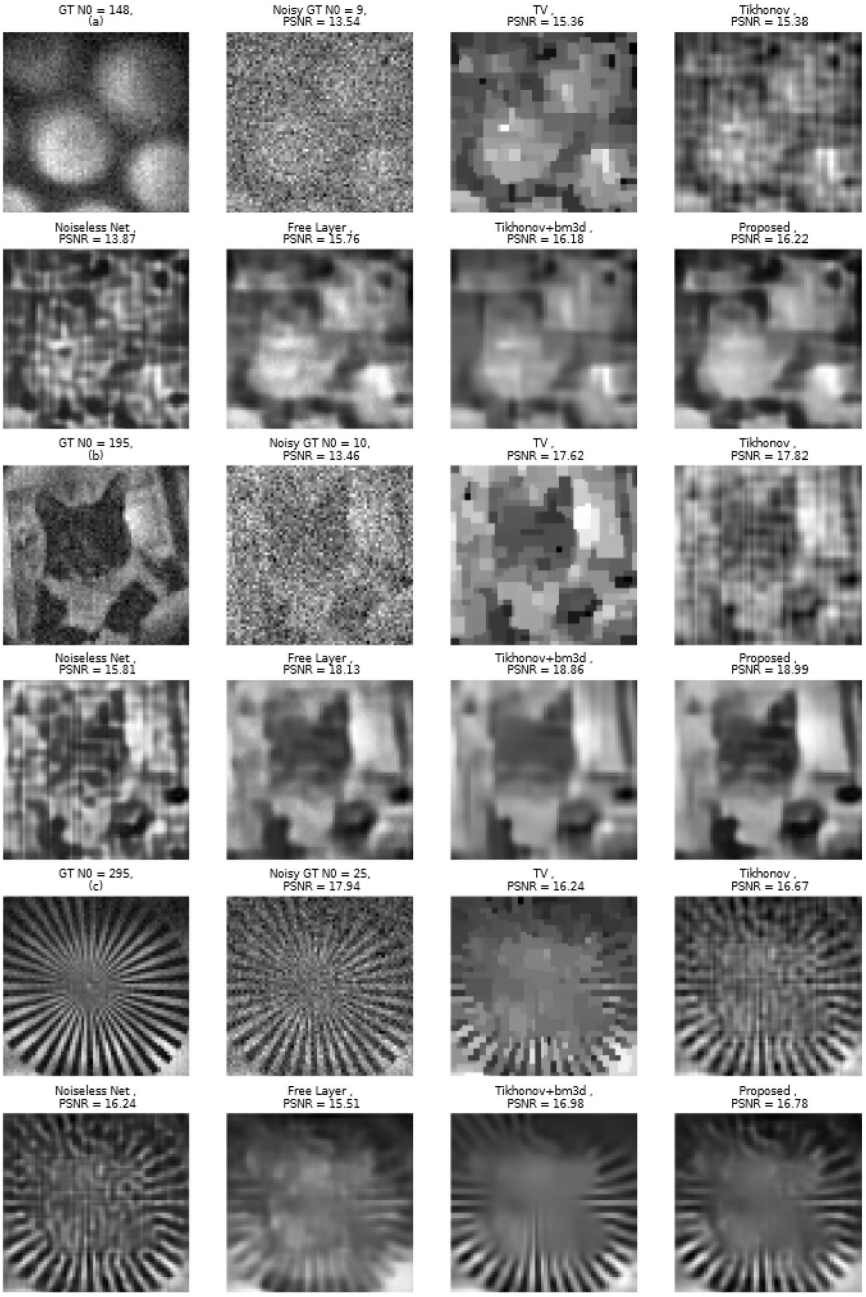


FIGURE 5.4 – Reconstructions of three experimental datasets by the different methods (top row: LED lamp with $M = 512$; middle row: STL-10 cat with $M = 512$; bottom row: Siemens Star resolution target with $M = 1024$). We display the images reconstructed from a fully sampled dataset (ground-truth; GT) acquired with high image intensity (first column, $\alpha = 148$ photons, $\alpha = 195$ photons and $\alpha = 295$ photons for the LED lamp, STL-10 cat and Siemens Star resolution target, respectively) and lower image intensity ('Noisy GT' second column, $\alpha = 9$ photons, $\alpha = 10$ photons and $\alpha = 25$ photons for the LED lamp, STL-10 cat and Siemens Star resolution target, respectively). The following columns show reconstructions using the total variation regularized solution (Li, 2010), the Tikhonov-regularized solution of Equation (5.7), the noiseless proposed network (Noiseless Net) Section 5.3.2 (network trained with no noise and where we assume $\boldsymbol{\Sigma}_\alpha = \mathbf{0}$), a Deep neural network with the same architecture as our proposed method, where the mapping is learned as in (Higham et al., 2018) (Free Layer), the Tikhonov-regularized method combined with BM3D (Dabov et al., 2007) denoising and the proposed network Section 5.3.2 (trained with $\mu_\alpha = 50$ photons and $\sigma_\alpha = 0.5\mu_\alpha$). All of the PSNRs are computed as described in Section 5.4.3, with the first column as the ground-truth.

The noise level of the image under acquisition is a key feature in real-life experiments. This parameter can be estimated first, and then the network that fits the actual noise level can be evaluated. However, as the noise level is unknown, this requires the loading of several networks, which might be a severe limitation for real-time applications, and in particular if the models are too large to be all stored on the graphics processor unit. Our proposed method is robust to different levels of noise as long as it was trained with low levels of noise (high values of α). Therefore, the same network can be used almost optimally in a wide variety of situations. The Tikhonov-regularized mapping that we propose helps to denoise the raw data, to provide an approximate reconstruction to the convolutional layers that can focus on learning spatial features.

One limitation of our deep network compared to more classic approaches such as (Lefkimmiatis and Unser, 2013; Luisier, Blu, and Unser, 2011; Li, Luisier, and Blu, 2016) is that there is no theoretical guarantee that it will work for any image; in particular, if the image under acquisition significantly differs from those of our training set. This is a common concern for deep-learning approaches that are, however, seen to work well in practice. While this tends to be confirmed by our experimental results where our approach worked on the Siemens Star sector and on a LED light (both of which are very different from the images of the stl-10 database), there are no theoretical guarantees that this will always be the case.

Another limiting aspect of our study might be the choice of our architecture, which is shallower than popular architectures, such as the U-Net used in (Jin et al., 2017), and more recent variants. However, we are keen to keep the number of network parameters as low as possible, to keep both the training and evaluation times as short as possible. Another limitation of this study concerns the analysis of the PSNRs of the images reconstructed from the experimental data, where the ground truth is not known. We limit this common issue by acquiring fully sampled low-noise images. Finally, we only test our algorithms on $N \times N$ pixel images, with $N = 64$. By considering a database with high-resolution images (e.g., ImageNet), our network can be generalized to handle the case where $N > 64$, either directly or by implementing a patch-based strategy to limit the memory requirements.

Compared to most studies dedicated to deep learning for inverse problems, we mainly focus on the (fully connected) layers of the network, which acts in the measurement domain. The post-processing layers that act in the image domain can be replaced by any variants (e.g., U-Net, resNet, others). Moreover, our approach is compatible with architectures inspired by conventional variational methods (e.g., (Aggarwal, Mani, and Jacob, 2019)). This will be the object of future work. Although this work focuses on single-pixel imaging, it can be used for any linear problem where the measured data is scarcely sampled in an orthogonal basis, just by setting $\boldsymbol{H}$ to the corresponding basis. Finally, this network can easily be adapted to other noise models, as we only need an estimation of the mean and covariance of the noisy measurements.

5.7 Conclusion

We present a deep learning method to recover an image in single-pixel imaging by introducing mapping of the raw data to the image domain. This mapping is implemented as the first layer of a neural network, which provides an approximate reconstruction to a traditional cascade of convolutional layers. We also describe a framework for training a network using simulated data only. We describe how to exploit all of the experimental parameters to deal with experimental data, and also how

to normalize the measurement, which is fundamental when a nonlinear reconstructor (e.g., neural network) is considered.

Although our network is trained using simulations only, it performs very well on experimental data, even when the noise level is unknown. The estimation of the noise covariance coupled with an appropriate training process appears to be efficient in a wide variety of scenarios. This is particularly interesting in real-time configurations where it is not possible to evaluate many different reconstructors. In future work, we will explore more complex interpretable network architectures.

Chapter 6

A 3D Denoised Completion Network for Deep Single-pixel Reconstruction of Hyperspectral Images

Hyperspectral imaging is of interest in biomedical applications, and the single-pixel camera can be used as a hyperspectral imager by focusing the light onto an optical fiber connected with a spectrometer. This hyperspectral single-pixel camera inherits the spectral resolution of a spectrometer. By adding an extra dimension to the input data and exploiting the correlations across that dimension, we can regularise the hyperspectral images to improve the reconstruction quality.

This chapter reports on our experiments with neural networks that use higher dimension convolutions to reconstruct hyperspectral images from data from a hyperspectral single-pixel camera. The materials of this chapter were realised in collaboration with V. Pronina from Skolkovo Institute of Science and Technology and was submitted to *Optics Express*.

abstract

Single-pixel imaging acquires an image by measuring its coefficients in a transform domain, thanks to a spatial light modulator. However, as measurements are sequential, only a few coefficients can be measured in the real-time applications. Therefore, single-pixel reconstruction is usually an underdetermined inverse problem that requires regularization to obtain an appropriate solution.

Combined with a spectral detector, the concept of single-pixel imaging allows for hyperspectral imaging. While each channel can be reconstructed independently, we propose to exploit the spectral redundancy between channels to regularize the reconstruction problem. In particular, we introduce a denoised completion network that includes 3D convolution filters. Contrary to black-box approaches, our network combines the classical Tikhonov theory with the deep learning methodology, leading to an explainable network.

Considering both simulated and experimental data, we demonstrate that the proposed approach yields hyperspectral images with higher peak signal-to-noise ratios and structural similarity than the approaches developed for grayscale images.

6.1 Introduction

Single-pixel imaging is a computational technique that can reconstruct an image from a single point detector (Gibson, Johnson, and Padgett, 2020). It has found applications in fluorescence microscopy (Studer et al., 2012), image-guided surgery (Aguénounon et al., 2019), diffuse optical tomography (Pian et al., 2017), short-wave infrared imaging (Zhang et al., 2020), imaging through scattering media (Li et al., 2020a). Considering a spectral detector, the concept of single-pixel imaging extends to generic imaging. Multispectral imaging was demonstrated with a dispersive unit followed by photomultiplier tubes (Rousset et al., 2018; Tao et al., 2021). Hyperspectral single-pixel imaging has been demonstrated with a compact fiber spectrometer (Peller, Farahi, and Trammell, 2018; Lorente Mur et al., 2020) or exploiting a Michelson interferometer (Jin et al., 2017).

Single-pixel measurements can be modelled as dot products between an underlying image and some two-dimensional patterns that are implemented through a spatial light modulator (SLM) (Edgar, Gibson, and Padgett, 2019). To limit the acquisition times, it is highly desirable to reduce the number of the light patterns, which leads to an under-determined ill-posed inverse problem. Classical approaches to solve such problems are based on compressive sensing (Duarte et al., 2008); however, deep learning (DL) has also proven efficient in many optical problems, including image deblurring (Pronina et al., 2020), image reconstruction (Wang et al., 2019), image denoising (Jiang et al., 2016), and spectral and lifetime unmixing (Yao et al., 2019; Smith, Ochoa, and Intes, 2020). Not surprisingly, single-pixel imaging has also benefited from DL. In (Higham et al., 2018), a deep convolutional auto-encoder network was shown to outperform compressed sensing in image reconstruction task. Contrary to the empirical design of the network in (Higham et al., 2018), (Ducros, Lorente Mur, and Peyrin, 2020) introduced an architecture where the first layer was shown to estimate the conditional expectation of the image given noiseless measurements. In (Lorente Mur et al., 2021), the network was generalized to the case of noisy data and to various noise levels. However, while the other computational hyperspectral approaches (*e.g.*, CASSI (Wang et al., 2019) or CS MUSI (Gedalin, Oiknine, and Stern, 2019)) exploit full 3D convolutional networks to engage the hyperspectral dimension, the existing *single-pixel* reconstruction networks are still limited to 2D convolutions, with the problem of hyperspectral reconstruction still awaiting to be addressed.

6.1.1 Contribution

In this paper, we first propose to generalize the reconstruction network proposed in (Lorente Mur et al., 2021) for single-pixel grayscale imaging to the problem of hyperspectral single-pixel imaging. In particular, we consider higher-order convolutions to learn the regularization across the hyperspectral dimension. To the best of our knowledge, this is the first effort to develop the 3D reconstructor for the single-pixel HSI. Our networks combine the classical optimization scheme for denoising in the measurement domain with the deep learning approach for projection from the measurement domain to the image domain and a final regularization of the solution in the image domain. To train our network, we create a HSI dataset from the set of RGB images as described in Section 6.4.1. We validate our approach on both simulated and experimental hyperspectral data. Given acceptance of this manuscript, our hyperspectral reconstruction network will be integrated into the open source `spyrit` package (Lorente Mur and Ducros, 2020).

6.2 Single-pixel imaging

6.2.1 Image acquisition

Single-pixel imaging can be modeled as $\boldsymbol{m} = \alpha \boldsymbol{H}_1 \boldsymbol{f}$, where $\boldsymbol{m} \in \mathbb{R}^M$ represents the raw measurements, $\boldsymbol{H}_1 \in \mathbb{R}^{M \times N}$, with $M < N$, are the patterns uploaded onto a SLM, $\boldsymbol{f} \in [0; 1]^N$ is the unknown image, and α is the unknown image intensity (in photons). The patterns $\boldsymbol{H}_1$ are traditionally chosen in an orthogonal basis $\boldsymbol{H} \in \mathbb{R}^{N \times N}$, with the classical choices including Fourier, discrete cosine transform, wavelets, and Hadamard bases. They can be either chosen before the acquisition (Baldassarre et al., 2016) or sampled adaptively during the acquisition (Rousset et al., 2017). In practice, the raw measurements are corrupted by the mixture of Poisson, and Gaussian noise (Foi et al., 2008; Rosenberger et al., 2016), where the Poisson noise is signal-dependent and arises due to the discrete nature of the electronic charge and the Gaussian noise occurs due to the fluctuations in the circuit and is signal-independent. Because the SLMs, such as digital micromirror devices (DMDs), cannot implement the negative values, it is necessary to decompose the patterns $\boldsymbol{H}_1$ into a set of patterns with only the positive values (Lorente Mur et al., 2019), leading to the measurement of

$$\boldsymbol{m}^{\alpha}_{+,-} \sim K\, \mathcal{P}(\alpha \boldsymbol{H}_1^{+,-} \boldsymbol{f}) + \mathcal{N}(\mu_{\text{dark}}, \sigma^2_{\text{dark}}). \tag{6.1}$$

Here, $\mathcal{P}$ denotes the Poisson distribution, while $\boldsymbol{H}_1^+$ and $\boldsymbol{H}_1^-$ are the positive and the negative parts of $\boldsymbol{H}_1$, respectively, K is the calibration coefficient that represents the overall system gain, μ_{dark} and σ^2_{dark} are the mean and the variance of the dark current. One simple, yet efficient, differential approach is to measure the fraction

$$\boldsymbol{m}^{\alpha} = \frac{\boldsymbol{m}^{\alpha}_{+} - \boldsymbol{m}^{\alpha}_{-}}{K\alpha}, \tag{6.2}$$

where the normalization factor $K\alpha$ is introduced to get the images with intensities in $[0, 1]$.

6.2.2 Deep image reconstruction

Following the recent widespread success of deep neural networks, the reconstruction of $\boldsymbol{f}$ was proposed to be performed with the non-linear models (Ducros, Lorente Mur, and Peyrin, 2020)

$$\hat{\boldsymbol{f}} = \mathcal{G}_{\boldsymbol{\theta}}(\boldsymbol{m}^{\alpha}), \tag{6.3}$$

where $\mathcal{G}$ is a neural network, the parameters of which are denoted as $\boldsymbol{\theta}$. Given an image database consisting of S measurement-image pairs $\{\boldsymbol{m}^{\alpha,(s)}, \boldsymbol{f}^{(s)}\}$, $1 \leq s \leq S$, the network parameters $\boldsymbol{\theta}$ are optimized during a training phase by minimizing the so-called loss function

$$\min_{\boldsymbol{\theta}} \sum_{s=1}^{S} \|\boldsymbol{f}^{(s)} - \mathcal{G}_{\boldsymbol{\theta}}(\boldsymbol{m}^{\alpha,(s)})\|_2^2 + \mathcal{J}(\boldsymbol{\theta}), \tag{6.4}$$

where $\mathcal{J}$ is typically chosen to penalize large network parameters and reduce the freedom in the neural network. Traditionally, neural networks are a composition of cascaded layers

$$\mathcal{G}_{\boldsymbol{\theta}} = \mathcal{G}^L_{\boldsymbol{\theta}} \circ \ldots \circ \mathcal{G}^1_{\boldsymbol{\theta}}, \tag{6.5}$$

where $\mathcal{G}^{\ell}_{\boldsymbol{\theta}}$, $1 \leq \ell \leq L$ is the ℓ-th (nonlinear) layer of the network and $\circ$ is the function composition.

6.2.3 Denoised completion (Tikhonov) network

The denoised completion network (DC-Net) establishes a link between traditional and deep image reconstruction methods by freezing the first layer of the network such that it provides the best linear estimator of the minimum mean squared error solution, also known as the Tikhonov solution. An overview of the DC-Net architecture is given in Fig.6.1a.

As described in (Lorente Mur et al., 2021), the Tikhonov solution is given by $\tilde{\boldsymbol{f}} = \boldsymbol{H}^\top \tilde{\boldsymbol{y}}$, where $\tilde{\boldsymbol{y}}(\boldsymbol{m}^\alpha) = [\boldsymbol{y}_1^\top, \boldsymbol{y}_2^\top]^\top$ can be computed in two steps as

$$\boldsymbol{y}_1 = \boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_1[\boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_\alpha]^{-1}(\boldsymbol{m}^\alpha - \boldsymbol{\mu}_1), \tag{6.6a}$$

$$\boldsymbol{y}_2 = \boldsymbol{\mu}_2 + \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_1^{-1}[\boldsymbol{y}_1 - \boldsymbol{\mu}_1]. \tag{6.6b}$$

Here, $\boldsymbol{\mu}_1 \in \mathbb{R}^M$ and $\boldsymbol{\Sigma}_1 \in \mathbb{R}^{M\times M}$ are the expected value and covariance of the measured coefficients, respectively, $\boldsymbol{\Sigma}_\alpha \in \mathbb{R}^{M\times M}$ is the noise covariance, $\boldsymbol{\mu}_2 \in \mathbb{R}^{N-M}$ is the expected value of the missing coefficients, and $\boldsymbol{\Sigma}_{21} \in \mathbb{R}^{(N-M)\times M}$ is the covariance between the missing and measured coefficients. The computation of these quantities is described in 6.3.3.

The first step given by (6.6a) can be interpreted as the denoising of the raw measurements while the second step given by (6.6b) can be interpreted as the completion of the missing coefficients with relevant values. This choice for the first layer allows also for faster training compared to alternatives based on pseudo-inverse or back projection (Ducros, Lorente Mur, and Peyrin, 2020).

6.3 Proposed deep hyperspectral reconstruction

In the specific case of hyperspectral imaging, the (noiseless) image formation process can be written as

$$\boldsymbol{m}_\lambda^\alpha = \alpha \boldsymbol{H}_1 \boldsymbol{f}_\lambda, \quad 1 \le \lambda \le \Lambda, \tag{6.7}$$

where $\boldsymbol{m}_\lambda$ and $\boldsymbol{f}_\lambda$ are the λ-th channel of a measurement and its corresponding image, respectively. We also denote the full measurements vector by $\boldsymbol{m} = [\boldsymbol{m}_1^\top \dots \boldsymbol{m}_\Lambda^\top]^\top$ and the full hyperspectral image (HSI) by $\boldsymbol{f} = [\boldsymbol{f}_1^\top \dots \boldsymbol{f}_\Lambda^\top]^\top$.

In the next sections, we explore the approaches for reconstruction of the hyperspectral $\boldsymbol{f}$ from the measurement vector $\boldsymbol{m}$. Our final architecture of the proposed 3D denoised completion network for the reconstruction of a single-pixel hyperspectral images is presented in Fig. 6.1a.

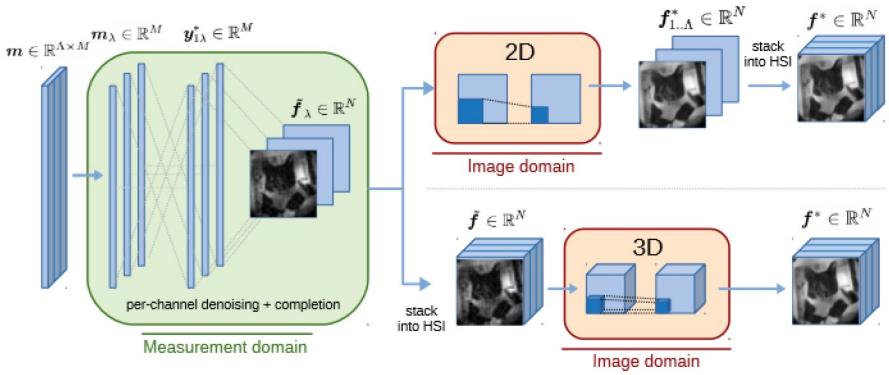
6.3.1 2D denoised completion network

A naive approach consists in processing each spectral channel independently, which can be summarized by

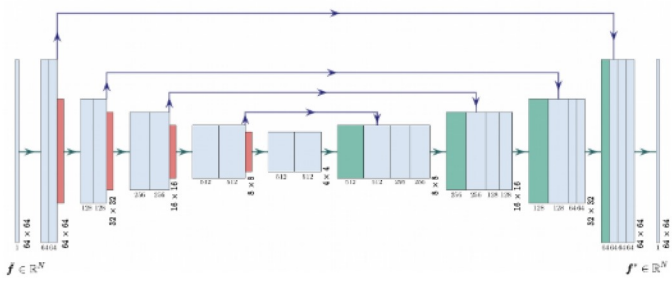
$$\tilde{\boldsymbol{f}}_\lambda = \mathcal{G}^1_{\boldsymbol{\theta}}(\boldsymbol{m}^\alpha_\lambda), \qquad 1 \le \lambda \le \Lambda, \tag{6.8a}$$

$$\hat{\boldsymbol{f}}_\lambda = (\mathcal{G}^L_{\boldsymbol{\theta}} \circ \dots \circ \mathcal{G}^2_{\boldsymbol{\theta}})(\tilde{\boldsymbol{f}}_\lambda), \qquad 1 \le \lambda \le \Lambda, \tag{6.8b}$$

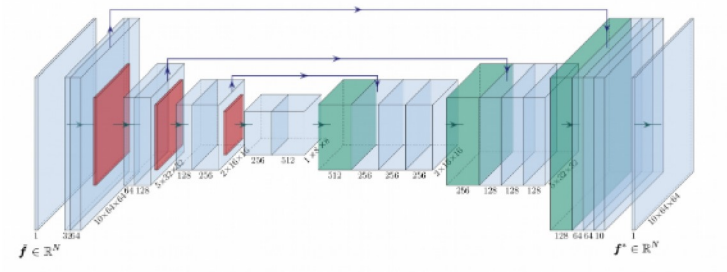
where $\mathcal{G}^1$ implements the denoised completion solution given by (6.6). Note that each spectral channel is processed independently. In particular, the layers of $(\mathcal{G}^L_{\boldsymbol{\theta}} \circ \dots \circ \mathcal{G}^2_{\boldsymbol{\theta}})$ rely on 2D convolution kernels that act in the spatial domain only and that are trained using grayscale images.



(A)



(B)



(C)

FIGURE 6.1 – Overview of the deep reconstruction networks. (a) Architecture of the 2D and 3D denoised completion networks (DC-Net). Denoising and completion occur in the measurement domain, for all channels independently in both the 2D and 3D cases. In the 2D case (top part of the diagram), the learnable layers ($\mathcal{G}^L_{\boldsymbol{\theta}} \circ \ldots \circ \mathcal{G}^2_{\boldsymbol{\theta}}$) correspond to 2D UNet; in the 3D case (bottom part of the diagram), to a 3D UNet. (b) Architecture of the 2D UNet. (c) Architecture of the 3D UNet.

6.3.2 3D denoised completion network

To exploit the correlation between the spectral channels of a hyperspectral image, we introduce a 3D neural network. After the reconstruction in a per-channel manner using Eq.6.6, we introduce higher-order convolutions into the image domain

$$\tilde{\boldsymbol{f}}_\lambda = \mathcal{G}^1_{\boldsymbol{\theta}}(\boldsymbol{m}^\alpha_\lambda), \quad 1 \leq \lambda \leq \Lambda, \tag{6.9a}$$

$$\hat{\boldsymbol{f}} = \mathcal{G}_{\boldsymbol{\theta}}(\tilde{\boldsymbol{f}}), \tag{6.9b}$$

where the learnable layers of $\mathcal{G}_{\boldsymbol{\theta}}$ act across all of the three dimensions of a hypercube, regularizing both the spatial and spectral dimensions contrary to the 2D network.

	Tikhonov		DC-UNet-2		DC-UNet-3	
α	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
150	21.91 ± 2.06	$.6699 \pm .0724$	22.90 ± 2.47	$.7311 \pm .0626$	$\mathbf{27.70 \pm 1.98}$	$\mathbf{.8908 \pm .0366}$
200	22.37 ± 2.17	$.6947 \pm .0711$	23.33 ± 2.58	$.7504 \pm .0612$	$\mathbf{28.68 \pm 2.20}$	$\mathbf{.9111 \pm .0357}$
500	23.66 ± 2.53	$.7639 \pm .0631$	24.58 ± 2.95	$.8065 \pm .0559$	$\mathbf{29.31 \pm 2.26}$	$\mathbf{.9239 \pm .0313}$

Table 6.1 – PSNR and SSIM of simulated reconstructed images under various noise levels (the entire HSI $\alpha = 150, 200, 500$) on the STL-10 dataset. 'Tikhonov' refers to the generalised Tikhonov regularisation (Eq. 6.6), 'DC-UNet-2D' refers to the denoised completion network with a UNet using 2D convolutions (Sec. 6.2.3), 'DC-UNet-3D' refers to the denoised completion network with a UNet using 3D convolutions (Sec. 6.3.2). The highest metrics are highlighted in bold.

6.3.3 Estimation of the mean and covariance matrices

The first steps of the denoised completion network, given by Eq. 6.6, require the mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$ of the data, as well as the noise covariance $\boldsymbol{\Sigma}_\alpha$. As described in (Lorente Mur et al., 2021), we compute the mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$ from the STL-10 database prior to the training phase and estimate the noise covariance from the raw measurements. Here, we have

$$\boldsymbol{\Sigma}_{\alpha_\lambda} = \mathrm{Diag}(\boldsymbol{\sigma}^2_{\alpha_\lambda}) = \frac{1}{K\alpha_\lambda^2}\mathrm{Diag}\left(\boldsymbol{m}^\alpha_{+\lambda} + \boldsymbol{m}^\alpha_{-\lambda}\right) - \rho\frac{2\mu_{\mathrm{dark}}}{\alpha_\lambda^2 K} + \rho\frac{2\sigma^2_{\mathrm{dark}}}{\alpha_\lambda^2 K^2}, \tag{6.10}$$

where $\mathrm{Diag}(\boldsymbol{\sigma}^2_{\alpha_\lambda})$ refers to the diagonal matrix with the diagonal coefficients being the elements of the noise variance $\boldsymbol{\sigma}^2_{\alpha_\lambda}$, which depends on the spectral channel, as different channels of the same measurement contain different numbers of photons. The experimental parameters μ_{dark}, σ_{dark} and K can be estimated as described in (Rosenberger et al., 2016). The parameter ρ corresponds to the number of raw channels that are binned to produce each of the Λ spectral channels.

6.4 Experiments

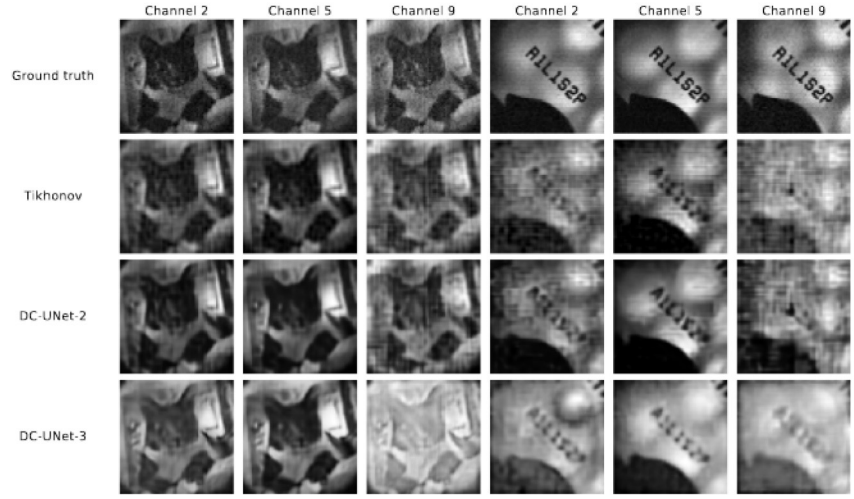
We compare the proposed 3D denoised completion network (DC-UNet-3) against the 2D denoised completion network (DC-UNet-2) and the Tikhonov solution given by (6.6). For DC-UNet-2, we use a customized 2D UNet architecture (with 13,394,177 learned parameters) based on the one described in (Ronneberger et al., 2015) and presented in Fig. 6.1b. For DC-UNet-3, we employ a 3D UNet architecture (with 16,320,257 learned parameters) described in (Çiçek et al., 2016) and presented in Fig. 6.1c.

6.4.1 Training of the networks

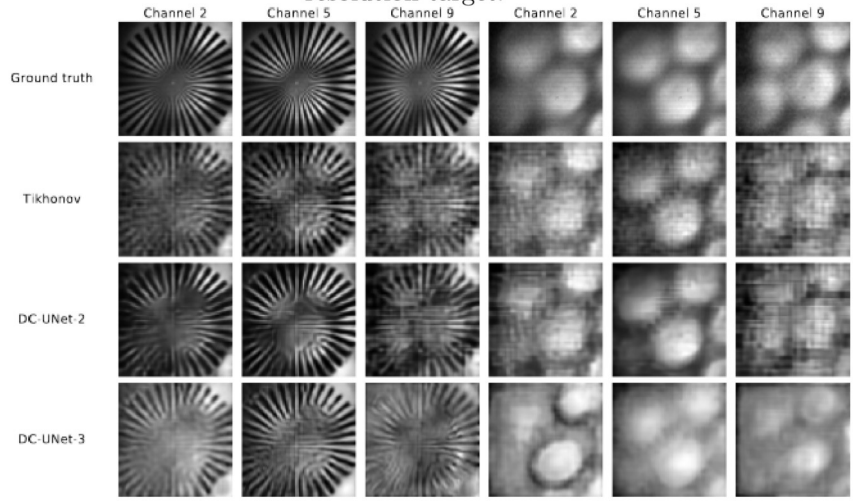
To train our networks, we use the STL-10 database (Coates, Ng, and Lee, 2011) which consists of 105,000 color images corresponding to the 'train' and 'unlabeled' subsets. The original 96×96 images are resized to 64×64 using bicubic transform. We create a dataset of hyperspectral images from the RGB images $[\boldsymbol{f}^{\mathrm{r}}, \boldsymbol{f}^{\mathrm{g}}, \boldsymbol{f}^{\mathrm{b}}] \in \mathbb{R}^{N\times 3}$ by combining the three channels for every wavelength λ as

$$\boldsymbol{f}_\lambda = \boldsymbol{f}^{\mathrm{r}}\eta^{\mathrm{r}}_\lambda + \boldsymbol{f}^{\mathrm{g}}\eta^{\mathrm{g}}_\lambda + \boldsymbol{f}^{\mathrm{b}}\eta^{\mathrm{b}}_\lambda \tag{6.11}$$

where $\eta_\lambda^{\mathrm{r}} \in \mathbb{R}$, $\eta_\lambda^{\mathrm{g}} \in \mathbb{R}$, and $\eta_\lambda^{\mathrm{b}} \in \mathbb{R}$ are spectral weights, chosen here to represent the spectral sensitivity of the long-, medium- and short-wavelength human photoreceptors, respectively (Stockman and Sharpe, 1999; Foster, 2010). As such, we create HSIs with $\Lambda = 10$ spectral channels.



(a) Reconstruction of two experimental images - STL-10 cat and off-centered Siemens star resolution target.



(b) Reconstruction of two experimental images - centered Siemens star target and LED lamp.

FIGURE 6.2 – Reconstruction results of four experimental hyperspectral images. In both subfigures, top row displays the ground truth images for the 2nd, 5th and 9th channels; second row displays results of denoising and completion with generalized Tikhonov regularization solution; third row displays reconstruction with the denoised completion network with 2D UNet regularization; bottom row displays reconstruction with the denoised completion network with 3D UNet regularization.

During training, we simulate the raw data from the hyperspectral dataset according to (6.1), where the system gain is set to one and the dark noise is set to to zero for simplicity. However, the actual system gain and dark noise of our system are taken into account through (6.10) for the reconstruction of experimental data.

We implement the full network training using PyTorch (Paszke et al., 2019) and optimize it using Adam (Kingma and Ba, 2014), with an initial learning rate of 10^{-3} that is divided by 5 every 20 epochs. We use a maximum of 100 epochs. We trained

our proposed 3D reconstructor with an NVIDIA Quadro RTX 6000, and training process took 28 h 51 min on 4 GPUs. The two dimensional neural networks were trained with a NVIDIA GeForce GTX 1080Ti, and training took 10 h 35 min on 2 GPUs. To accelerate and stabilize the training process (Le Cun, Kanter, and Solla, 1991), we normalize our images, both grayscale and hyperspectral, in the range [-1;1].

6.4.2 Experimental data

We consider the single-pixel camera described in (Lorente Mur et al., 2020) and the open single-pixel hyperspectral imaging (SPIHIM) dataset (Ducros and Lorente Mur, 2020). The SPIHIM dataset contains different objects acquired using 4096 Hadamard patterns of dimension $N = 64 \times 64$ for 2048 spectral channels in the range $[317, 1064]$. The objects are available for different light intensities corresponding to different neutral densities placed between the illumination lamp and the object.

In this work, we select four grayscale objects: the STL-10 cat, the Siemens star resolution target (*Photography — Electronic still picture imaging — Resolution and spatial frequency responses* 2019), the off-centered Siemens star target, and the LED lamp. For each object, we obtain a high noise dataset by placing a neutral optical density behind the lamp to reduce the light flux. We consider an optical density of 0.6 for the STL-10 cat and of 1.3 for the other objects. We choose an integration time of 8 ms for the centered star target and to 4 ms for all other objects. We also consider a color Siemens star resolution target where the colors are taken from the hue color wheel.

Finally, we down-sample the full measurement vector by retaining only $M = 1024$ measurements for all objects. We only retain the spectral channels in the visible range [390, 700] nm that we merge into $\Lambda = 10$ new spectral channels by summing $\rho = 77$ original channels, *i.e.*, each new spectral channel contains 77 raw spectral channels. The ground truth images are obtained from the full measurement vectors with the lowest neutral optical density, either 0 or 0.3.

6.4.3 Evaluation metrics

To assess the performance of the reconstruction methods quantitatively, we use the standard peak signal-to-noise ratio (PSNR) and structural similarity (SSIM) metrics, implemented in scikit-image library (Walt et al., 2014). For experimental data, we estimate the ground truth as $\boldsymbol{f} = \boldsymbol{H}^\top \boldsymbol{y}_{\text{gt}}$, where $\boldsymbol{y}_{\text{gt}}$ is a fully sampled (*i.e.*, $M = N$) measurement vector with high signal-to-noise ratio. Before computing metrics, we normalize the ground-truth images in the range [-1;1].

We compute the PSNR of the hyperspectral image $\hat{\boldsymbol{f}}$ as its mean PSNR across all channels

$$\mathsf{PSNR}(\hat{\boldsymbol{f}}, \boldsymbol{f}) = \frac{1}{\Lambda} \sum_{\lambda=1}^{\Lambda} \mathsf{PSNR}_2(\hat{\boldsymbol{f}}_\lambda, \boldsymbol{f}_\lambda), \tag{6.12}$$

where PSNR_2 is classically defined by

$$\mathsf{PSNR}_2(\hat{\boldsymbol{f}}_\lambda, \boldsymbol{f}_\lambda) = 10 \log_{10} \frac{MAX_\lambda^2}{\|\hat{\boldsymbol{f}}_\lambda - \boldsymbol{f}_\lambda\|^2}, \tag{6.13}$$

with MAX_λ corresponding to the dynamic range of the λ-th spectral channel. The same way, we compute the SSIM of a hyperspectral image $\hat{\boldsymbol{f}}$ as

$$\mathsf{SSIM}(\hat{\boldsymbol{f}}, \boldsymbol{f}) = \frac{1}{\Lambda} \sum_{\lambda=1}^{\Lambda} \mathsf{SSIM}_2(\hat{\boldsymbol{f}}_\lambda, \boldsymbol{f}_\lambda). \tag{6.14}$$

where SSIM_2 represents the SSIM between two images from one channel.

6.5 Results and discussion

6.5.1 Reconstruction from simulated data

We evaluate the Tikhonov, DC-UNet-2 and DC-UNet-3 methods on the (unseen) images of the test set of our hyperspectral database, for three different image intensities α (150, 200 and 500 photons) corresponding to different noise levels. In Table 6.1, we report the PSNR and the SSIM obtained from the networks that are trained for the corresponding intensities α.

We observe that DC-UNet-3 provides the best reconstruction metrics, outperforming the Tikhonov PSNR by 6.3 dB and the DC-UNet-2 PSNR by 5.35 dBs. As expected, the 3D-UNet acts as a regularization step that accounts for spectral correlation between the channels, which allows to recover missing information from the adjacent spectral channels.

6.5.2 Reconstruction from experimental data

Fig. 6.2 illustrates the performance of the three methods for the reconstruction of the four experimental datasets described in 6.4.2. We observe that both DC-UNet-2 and DC-UNet-3 provide smoother and less noisy output images than the Tikhonov method, thanks to the artifact correction network acting in the image domain. Despite some remaining artifacts, DC-UNet-3 allows to restore sharper image details in all spectral channels. This is especially noticeable on the boundary channels – cat face in STL-10 cat and letters in the off-centered Siemens star resolution target are closer to those of the ground truth images. This is achieved thanks to the spatial-spectral regularization nature of the postprocessing 3D network.

DC-UNet-3 achieves superior results in terms of noise reduction compared to Tikhonov and better preservation of details compared to DC-UNet-2. We observe that taking into account spectral correlation between adjacent channels helps to recover channels that initially contain less information. This can be seen clearly from the off-centered Siemens star resolution target, where the inscription is preserved in all spectral channels in contradistinction to the results obtained with Tikhonov and DC-UNet-2.

Fig. 6.3 presents the reconstruction of the color Siemens star target recovery using Tikhonov, DC-UNet-2 and DC-UNet-3 methods. The Tikhonov solution presents some artifacts such as the square-like structures on the frontier of the branches that stems from downsampling in the Hadamard basis, or the grain-like appearance from the photon-based noise, whereas reconstruction with DC-UNet-2 and DC-UNet-3 results in smoother images. Compared to DC-UNet-2, DC-UNet-3 is able to retain certain details that are lost when removing the artifacts in some cases. For instance, this can be observed in 'Channel 4' and 'Channel 6', where the upper branches of the star were mostly erased and lost their alignment when reconstructed using DC-UNet-2, but are mostly intact after recovery with DC-UNet-3. Furthermore, in 'Channel 9', DC-UNet-2 almost erased the bottom left branches, as opposed to the results provided by DC-UNet3. This appears to indicate that DC-UNet-3 exploits spectral channel redundancy. To get a visual comparison of the hyperspectral images, we also compute a pseudo color image as described in (Magnusson et al., 2020). Comparing the pseudo color images in the right column of Fig.6.3, one can see that all three methods succeed in color rendering to an extent. However, all methods seem to give a slight shift in the

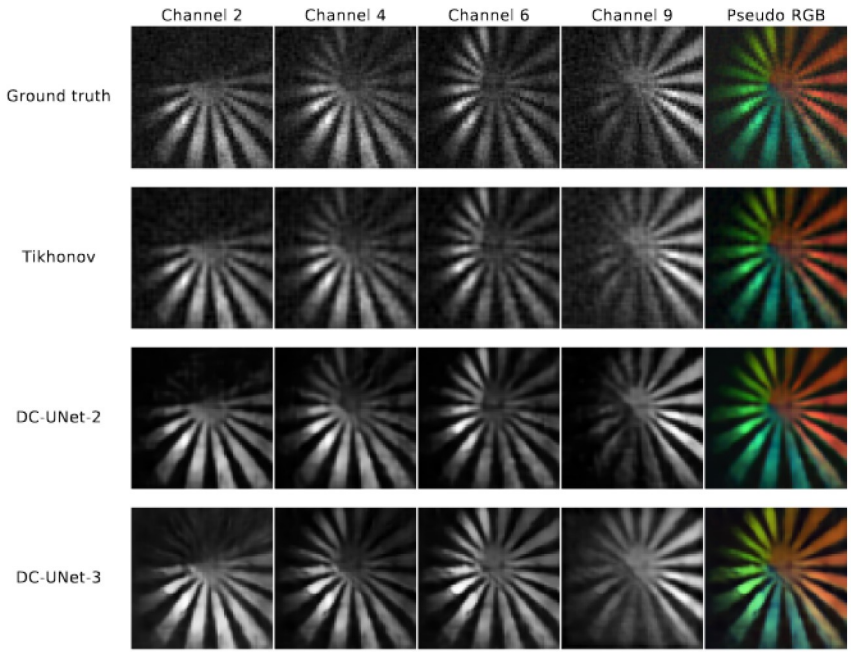


FIGURE 6.3 – Reconstruction of the experimental color Siemens target. Top row: ground truth images in the 2nd, 4th, 6th and 9th channels. Second row: Tikhonov solution. Third row: DC-Net solution using a 2D UNet. Bottom row: DC-Net solution using a 3D UNet. The right column represents the pseudo color image computed with the corresponding method recovery result according to (Magnusson et al., 2020).

color spectrum. The red branches reconstructed via Tikhonov and DC-UNet-2 appear to be biased towards red wavelengths, while the recovery with DC-UNet-3 causes the red branches to shift towards orange wavelengths. A similar situation is observed in the blue branches of the star, where Tikhonov and DC-UNet-2 applications result into the shift towards the lower blue wavelengths, while reconstruction via DC-UNet-3 shifts the branches into the upper-blue or lower-green wavelengths. However, in terms of spacial resolution, the branches produced by DC-UNet-3D appear to be of higher resolution, in particular in the upper left branches.

6.5.3 Limitations

The proposed three-dimensional network regularizes the reconstruction in both the spatial and the spectral dimensions, being capable of filling in the missing information and of improving the reconstruction quality. Still, there are some limitations to the proposed reconstruction pipeline.

The first limitation stems from a small number of spectral channels used in our study. Given that the conventional hyperspectral applications may require additional channels, one should carefully explore whether the spectral correlations between the channels can still be exploited when there is a number of them. Another limitation is associated with the choice of the architecture of the neural network. Here, we focused on the U-Net because it has demonstrated its effectiveness in many problems across many disciplines. However, this architecture could prove too heavy should larger convolutional filters be required (to exploit additional channels or dimensions, *e.g.*, third spatial coordinate or time). In this case, the choice of other architectures may be more sensible. Lastly, the proposed DC-UNet-3 tends to correlate the adjacent

spectral channels during the reconstruction. While this feature can be considered as an advantage for an object with a smooth spectra (*e.g.*, the cat's face or the off-centered Siemens target in Fig. 6.2), it could be a disadvantage if there are spectral discontinuities (*e.g.*, the color Siemens target in Fig. 6.3). Training a 3D DC-Net using a hyperspectral image database with distinct spectral features should alleviate this issue.

6.6 Conclusion

We proposed the first deep neural network for reconstructing hyperspectral images from a small number of noisy single-pixel measurements. The layers of the proposed network combine the classical Tikhonov solution with learnable convolution filters. To exploit the correlation between the spectral channels, it considers the filters that act in both the spatial and the spectral dimensions. Our experiments show that a 3D UNet improves the reconstruction quality compared to a 2D UNet that acts in the image domain only. The proposed 3D network demonstrates superior performance in terms of quantitative metrics as well as visually. This is particularly evident in the channels with low photon counts, *i.e.*, with the overwhelming noise. In the future work, we plan to consider spectral discontinuities, additional dimensions, and other architectures of the base network.

Chapter 7

Deep Expectation-Maximization for Single-Pixel image reconstruction with signal-dependent noise

Model-based deep learning architectures for image reconstruction are family of deep learning methods whose structure can be seen as a maximum a posteriori optimisation scheme. These architectures use the knowledge of the forward operator and a noise model to alternate between a data consistency layer and an image-to-image denoising layer. The data consistency layer ensures that the consistency of the reconstructed image with respect to the forward operator and noise levels, while the image-to-image denoising

In this chapter we proposed a model based deep learning architecture based on the expectation-maximisation algorithm. This architecture has been designed to tackle single-pixel imaging problems where the data is corrupted by mixed Skellam-Gaussian noise. The contents of this chapter are based on a conference proceeding from IEEE ISBI 2020 (Lorente Mur et al., 2021). Generalising upon that conference proceeding, the contents of this chapter were submitted to *IEEE Transactions on computational imaging.*

abstract

Image reconstruction from a sequence of a few linear measurements that are corrupted by signal-dependent normally distributed noise is an inverse problem with many biomedical imaging applications, such as computerized tomography, positron emission tomography, and optical microscopy. In this study, we focus on single-pixel imaging, where the set-up acquires a down-sampled Hadamard transform of an image of the scene. Deep learning is a very efficient framework to solve inverse problems in imaging. Several neural-network architectures provide a link between deep and optimization-based image reconstruction methods. These deep-learning methods rely on a forward operator and lead to more interpretable networks. Here, we propose a novel network architecture obtained by unrolling the expectation-maximization algorithm. In particular, we compute the maximum *a-posteriori* estimate of the unknown image given measurements corrupted by normally distributed signal-dependent noise. We show that the so-called expectation maximization reconstruction network (EM-Net) applies to mixed Skellam-Gaussian noise models that are common in single-pixel imaging. We present reconstruction results from simulated and experimental single-pixel acquisitions. We show that EM-Net generalizes very well to the unseen noise

levels during training, despite having fewer learned parameters than alternative methods. The proposed EM-Net generally reconstructs images with fewer artifacts and higher signal-to-noise ratios, in particular in high-noise situations.

7.1 Introduction

Single-pixel imaging is a computational imaging configuration where a single point detector is used to recover an image (Duarte et al., 2008). It has been successfully applied to fluorescence microscopy (Studer et al., 2012), hyperspectral imaging (Rousset et al., 2018; Arce et al., 2014), diffuse optical tomography (Pian et al., 2017), image-guided surgery (Aguénounon et al., 2019), and short-wave infrared imaging (Zhang et al., 2020). A single-pixel camera acquires the dot product between the image of a scene and some two-dimensional light patterns that are displayed sequentially using a spatial light modulator (Edgar, Gibson, and Padgett, 2018). The image of the scene is then reconstructed from the raw measurements. To limit acquisition times, it is highly desirable to reduce the number of light patterns, which leads to an under-determined inverse problem.

Recent advances in deep learning have revolutionized image reconstruction (Wang et al., 2018; Arridge et al., 2019). In particular, convolutional neural networks (CNNs) are very efficient for solving the computed tomography problem, either by learning direct inverse mapping (McCann, Jin, and Unser, 2017), or through the use of adversarial neural networks (Kang et al., 2019). Much effort is being devoted to bridging the gap between more traditional model-based approaches for image reconstruction and deep-learning-based approaches for inverse problems. CNNs can provide sparsifying transforms (Chun and Fessler, 2020) and model the manifold of natural images (Aggarwal, Mani, and Jacob, 2019), or of a projector onto this space (Gupta et al., 2018). The CNN priors are then plugged into an unrolled optimization algorithm (Adler and Öktem, 2018; Reader et al., 2021) or used as in expansions, like the Neumann series (Gilton, Ongie, and Willett, 2020).

This trend also greatly benefits computational optics (Barbastathis, Ozcan, and Situ, 2019; Kellman et al., 2019). In single-pixel imaging, in particular, fully learned mapping from the measurement space to the image space can outperform compressed sensing reconstructions (Higham et al., 2018). (Dave et al., 2019) used a deep auto-regressive model that is plugged into an alternating direction method of multiplier algorithms for single-pixel images. In (Ducros, Lorente Mur, and Peyrin, 2020), we proposed a simple reconstruction CNN where the first layer computes the conditional expectation of the unknown image, given noiseless measurements. We generalized this idea to the case of noisy data with varying noise levels in (Lorente Mur et al., 2021), and we introduced an iterative architecture in (Lorente Mur et al., 2021).

Single-pixel measurements result from the difference between two Poisson variables. Therefore single-pixel data are corrupted by mixed Skellam-Gaussian noise, which has no closed-form expression. For similar problems, such as mixed Poissonian-Gaussian denoising, this difficulty is circumvented by using variance stabilizing transforms, such as the generalized Anscombe transform (Anscombe, 1948), before processing the Gaussian noise (Murtagh, Starck, and Bijaoui, 1995). However, this introduces a bias (Makitalo and Foi, 2011). Alternative approaches rely on approximations. In (Benvenuto et al., 2008; Chouzenoux et al., 2015) and (Ghulyani and Arigovindan, 2021), the likelihood of mixed Poisson-Gaussian is approximated as a finite sum. The Poisson-Gaussian unbiased risk-estimator linear expansion of thresholds (Li, Luisier, and Blu, 2018) seeks to approximate an unbiased estimate of the mean squared error

with linear combinations of Wiener filters followed by a thresholding of Haar wavelet coefficients. Other approaches simplify the problem by either approximating the Poisson contribution by normally distributed signal-dependent noise (Li et al., 2015) or by neglecting the normal distribution. The maximum *a posteriori* (MAP) can then be computed using the expectation-maximization algorithm (Dempster, Laird, and Rubin, 1977). However, the resulting optimization problem is usually solved using hand-crafted priors (*e.g.*, Hessian-based or total variation [TV]-based penalization).

7.1.1 Contribution

In this paper, we derive and analyze a deep neural-network architecture based on the expectation-maximization algorithm. The so-called expectation maximization reconstruction network (EM-Net) can tackle inverse problems with signal-dependent noise, such as those involved in computational optics. We significantly extend from our preliminary results (Lorente Mur et al., 2021) in three directions. First, we model single-pixel reconstruction as a mixed Skellam-Gaussian reconstruction problem, for which we derive the corresponding likelihood. As this is intractable numerically, we propose a normal approximation that leads to signal-dependent covariance. Secondly, we propose a four-step iterative algorithm where the image prior is learnt from training examples, and we show how to reduce the number of learnable parameters, in agreement with EM theory. We also propose a training strategy where most of the parameters of our EM-Net can be precomputed. Thirdly, we conduct extensive numerical comparisons with data-driven reconstruction methods designed for single-pixel imaging, and also other modalities. As the noise level is usually unknown, in practical experiments we focus on the robustness of the methods in the presence of noise, where the level differs from that used during training. We consider both simulated measurements corrupted by mixed Skellam-Gaussian noise and experimental data.

Upon acceptance, our reconstruction method will be made available in the Python toolbox SPyRiT(Lorente Mur and Ducros, 2020).

7.1.2 Organization of the paper

In Section 7.2, we model single-pixel acquisition and describe the associated reconstruction problem. In Section 7.3, we introduce a deep network based on the EM algorithm. In Section 7.4, we describe the numerical experiments and experimental data that we considered to validate our approach. Finally, we analyze our reconstruction results in Section 7.5.

7.2 Compressive single-pixel acquisition

7.2.1 Compressed single-pixel acquisition

Let $\boldsymbol{f} \in [0,1]^N$ be the image to be acquired. The main idea of compressive single-pixel imaging is to measure $\boldsymbol{y} = \boldsymbol{H}\boldsymbol{f}$ using hardware, and to recover $\boldsymbol{f}$ using software. The matrix $\boldsymbol{H} = [\boldsymbol{h}_1, \dots, \boldsymbol{h}_N] \in \mathbb{R}^{N \times N}$ collects the patterns that are sequentially uploaded in a spatial light modulator. These patterns can be chosen as a basis matrix, such as Fourier, discrete cosines, wavelets, and Hadamard bases (Ochoa et al., 2018). In practice, to accelerate acquisition times, only a few patterns in a basis are acquired. Moreover, as the spatial light modulator cannot implement negative values, the patterns $\boldsymbol{H}$ are split into positive and negative patterns $\boldsymbol{H}^+$ and $\boldsymbol{H}^-$, such that $(\boldsymbol{H}^+)_{i,n} = \max((\boldsymbol{H})_{i,n}, 0)$ and $(\boldsymbol{H}^+)_{i,n} = \max(-(\boldsymbol{H})_{i,n}, 0)$ (see (Lorente Mur et al., 2019) for details).

We model the raw measurement as mixed Poisson-Gaussian noise (Foi et al., 2008)

$$\hat{\mathbf{m}}^{\alpha}_{+,-} \sim K\,\mathcal{P}(\alpha \boldsymbol{S}\boldsymbol{H}^{+,-}\boldsymbol{f}) + \mathcal{N}(\mu_{\text{dark}}, \sigma^2_{\text{dark}}) \tag{7.1}$$

where $\mathcal{P}$ and $\mathcal{N}$ are the Poisson and Gaussian distributions, $\boldsymbol{S} = [\boldsymbol{I}_M, \boldsymbol{0}] \in \mathbb{R}^{M\times N}$ is a down-sampling matrix (where $M \leq N$), K is a constant that represents the overall system gain (in counts/electron), α is the intensity (in photons) of the image, μ_{dark} is the dark current (in counts), and σ_{dark} is the dark noise (in counts). We further hypothesize that μ_{dark} and σ_{dark} are independent of the image intensity α.

The raw measurements are finally combined back and normalized. This standard prepossessing step is

$$\mathbf{m}^{\alpha} = (\hat{\mathbf{m}}^{\alpha}_{+} - \hat{\mathbf{m}}^{\alpha}_{-})/(\alpha K). \tag{7.2}$$

Plugging the noise model of the raw measurements given by Equation (7.3) into the previous equation leads to the mixed Skellam-Gaussian noise model

$$\mathbf{m}^{\alpha} \sim \frac{1}{\alpha}\,\mathcal{S}_{\text{k}}(\alpha \boldsymbol{S}\boldsymbol{H}^{+}\boldsymbol{f}, \alpha \boldsymbol{S}\boldsymbol{H}^{-}\boldsymbol{f}) + \frac{2}{\alpha K}\,\mathcal{N}(0, \sigma^2_{\text{dark}}), \tag{7.3}$$

where the Skellam distribution (Skellam, 1946) $\mathcal{S}_{\text{k}}(\boldsymbol{\mu}_1, \boldsymbol{\mu}_2)$ is the distribution of the difference between two Poisson-distributed variables with means of $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$. The expectation of the measurements is $\mathbf{m}^{\alpha} = \boldsymbol{S}\boldsymbol{H}\boldsymbol{f}$.

7.2.2 Normal approximation of the mixed Skellam-Gaussian noise model

The likelihood of the mixed Skellam-Gaussian noise model involves an infinite sum of exponentials multiplied by modified Bessel functions (see appendix A for the derivation). Exploitation of this analytical expression is numerically intractable, as it involves an infinite series and the evaluation of Bessel functions.

$$p(\boldsymbol{m}|\boldsymbol{f}) \propto \prod_{i=1}^{M} \sum_{n\in\mathbb{N}} e^{-\alpha(\boldsymbol{h}_{i,+}^{\top}\boldsymbol{f} + \boldsymbol{h}_{i,-}^{\top}\boldsymbol{f})} \left(\frac{\boldsymbol{h}_{i,+}^{\top}\boldsymbol{f}}{\boldsymbol{h}_{i,-}^{\top}\boldsymbol{f}}\right)^{\frac{n-\alpha\boldsymbol{h}_i^{\top}\boldsymbol{f}}{2\alpha}} \mathcal{I}_{|\frac{n}{\alpha}-\boldsymbol{h}^{\top}\boldsymbol{f}|}\left(2\alpha\sqrt{(\boldsymbol{h}_{i,+}^{\top}\boldsymbol{f})(\boldsymbol{h}_{i,-}^{\top}\boldsymbol{f})}\right) e^{-\frac{(\boldsymbol{m}_i - \frac{n}{\alpha})^2}{\beta^2}}. \tag{7.4}$$

Similar to work that has addressed mixed Poisson-Gaussian noise, we propose to introduce an approximation. As the Gaussian distributions are good approximations of Poisson distributions with means greater than 5 (see (Huang, 2001)) – single-pixel measurements are typically several orders of magnitude above this – we use the normal approximation to the Poisson distributions in Equation (7.3). We obtain

$$\mathbf{m}^{\alpha} = \boldsymbol{S}\boldsymbol{H}\boldsymbol{f} + \boldsymbol{\epsilon}^{\alpha}, \text{with} \quad \boldsymbol{\epsilon}^{\alpha} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma}_{\alpha}) \tag{7.5}$$

where the noise covariance $\boldsymbol{\Sigma}_{\alpha}$ is given by

$$\boldsymbol{\Sigma}_{\alpha} = \mathsf{Diag}\left(\boldsymbol{\sigma}^2_{\alpha}\right) = \mathsf{Diag}\left(\frac{1}{\alpha}\boldsymbol{H}_1^{+}\boldsymbol{f} + \frac{1}{\alpha}\boldsymbol{H}_1^{-}\boldsymbol{f}\right) + \frac{2\sigma^2_{\text{dark}}}{K^2\alpha^2}\boldsymbol{I}_M. \tag{7.6}$$

Note that the noise variance $\boldsymbol{\sigma}^2_{\alpha}$ is unknown, as it depends on both the unknown image $\boldsymbol{f}$ and the unknown intensity α. Therefore, as in (Lorente Mur et al., 2021), we

exploit the raw measurement to estimate $\boldsymbol{\sigma}_\alpha^2$. We first note that from Equation (7.3) the variance and expectation of the raw measurements are related by $\mathbb{V}\mathrm{ar}\left(\hat{\mathbf{m}}_{+,-}^\alpha\right) = K\,\mathbb{E}\left(\hat{\mathbf{m}}_{+,-}^\alpha\right) - K\mu_{\mathrm{dark}} + \sigma_{\mathrm{dark}}^2$. Then, we approximate the unknown expectation by the noisy sample; *i.e.*, $\mathbb{E}\left(\hat{\mathbf{m}}_{+,-}^\alpha\right) \approx \hat{\boldsymbol{m}}_{+,-}^\alpha$, which leads to the following variance for the preprocessed measurements

$$\boldsymbol{\sigma}_\alpha^2 \approx \tilde{\boldsymbol{\sigma}}_\alpha^2 = \frac{1}{\alpha^2 K}(\hat{\boldsymbol{m}}_+^\alpha + \hat{\boldsymbol{m}}_-^\alpha - 2\mu_{\mathrm{dark}}\mathbf{1}) + \frac{2\sigma_{\mathrm{dark}}^2}{K^2\alpha^2}\mathbf{1}. \tag{7.7}$$

7.3 Proposed deep EM network

7.3.1 Objective

We adopt a Bayesian framework and reconstruct $\boldsymbol{f}$ from $\boldsymbol{m}^\alpha$ by computing a point-wise estimator of $p(\boldsymbol{f}|\boldsymbol{m}^\alpha)$. In particular, we aim to compute the MAP solution that solves

$$\underset{\boldsymbol{f}}{\operatorname{argmax}} \quad \log p(\boldsymbol{m}^\alpha|\boldsymbol{f}) + \log p(\boldsymbol{f}), \tag{7.8}$$

where we assume the probability density function $p(\boldsymbol{m}^\alpha|\boldsymbol{f}) \propto \exp\left(-\frac{1}{2}\|\boldsymbol{S}\boldsymbol{H}\boldsymbol{f} - \boldsymbol{m}^\alpha\|^2_{\boldsymbol{\Sigma}_\alpha^{-1}}\right)$ according to Equation (7.5), while $p(\boldsymbol{f})$ is an unknown probability density function.

Note that $\boldsymbol{\Sigma}_\alpha$ introduces signal-dependant noise, which prevents signal-independent strategies from being used for this problem.

7.3.2 The EM algorithm

The EM algorithm (Dempster, Laird, and Rubin, 1977) has commonly been used to estimate the MAP for image reconstruction tasks (Zhou et al., 2007). This is an iterative algorithm that produces a sequence of estimations $\{\boldsymbol{f}^{(k)}\}$ such that the sequence $\{\log p(\boldsymbol{f}^{(k)}|\boldsymbol{m}^\alpha)\}_k$ is monotonically nondecreasing. Every iteration of the EM algorithm is based on two steps: the expectation step, and the maximization step.

The expectation step computes the conditional expectation of the log-likelihood of $\mathbf{f}$ with respect to an auxiliary random variable $\mathbf{x}$, given the current estimate $\boldsymbol{f}^{(k)} \in \mathbb{R}^N$ and the measurements $\boldsymbol{m}^\alpha \in \mathbb{R}^M$

$$\mathcal{Q}(\boldsymbol{f}|\boldsymbol{f}^{(k)}) = \mathbb{E}_{\mathbf{x}}[\log p(\boldsymbol{x}|\boldsymbol{f})|\boldsymbol{m}^\alpha, \boldsymbol{f}^{(k)}] + p(\boldsymbol{f}). \tag{7.9}$$

In the EM literature, the variable $\boldsymbol{x} \in \mathbb{R}^N$ is referred to as the complete data, as it contains all of the variables in a problem, including those that are unobserved.

This conditional expectation is then maximized with respect to $\boldsymbol{f}$ during the maximization step, to produce the next iteration

$$\boldsymbol{f}^{(k+1)} = \underset{\boldsymbol{f}}{\operatorname{argmax}}\, \mathcal{Q}(\boldsymbol{f}|\boldsymbol{f}^{(k)}). \tag{7.10}$$

The EM algorithm requires the complete data $\mathbf{x}$ to satisfy the following admissibility proprieties. When $p(\boldsymbol{m}^\alpha|\boldsymbol{f})$ is Gaussian, $\mathbf{x}$ and $\mathbf{m}^\alpha|\boldsymbol{x}$ can be chosen as Gaussian vectors (Fessler and Hero, 1994). In particular, we can choose to interpret $\mathbf{x}$ as the full measurement vector and $\mathbf{m}^\alpha|\boldsymbol{x}$ as the subsampled measurements by setting

$$\mathbf{x} \sim \mathcal{N}(\boldsymbol{H}\boldsymbol{f}, \boldsymbol{\Sigma}), \tag{7.11}$$

$$\mathbf{m}^\alpha|\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{S}\boldsymbol{x}, \tilde{\boldsymbol{\Sigma}}). \tag{7.12}$$

where the covariance $\boldsymbol{\Sigma}$ and $\tilde{\boldsymbol{\Sigma}}$ must be chosen such that the variance of $\mathbf{m}^\alpha|\boldsymbol{f}$ derived from Equation (7.12)–Equation (7.11) must equal that of Equation (7.6), to lead to the following consistency condition

$$\boldsymbol{\Sigma}_\alpha = \tilde{\boldsymbol{\Sigma}} + \boldsymbol{S}\boldsymbol{\Sigma}\boldsymbol{S}^\top. \tag{7.13}$$

Here, interpreting Equation (7.12) as a measurement likelihood, we choose $\tilde{\boldsymbol{\Sigma}}$ as the approximate covariance $\tilde{\boldsymbol{\Sigma}}_\alpha$ introduced in Equation (7.7). As $\boldsymbol{\Sigma}_\alpha$ is diagonal, choosing $\tilde{\boldsymbol{\Sigma}}$ as diagonal implies that $\boldsymbol{S}\boldsymbol{\Sigma}\boldsymbol{S}^\top$, and therefore $\boldsymbol{\Sigma}$, are also diagonal.

7.3.3 Unrolling the EM algorithm

Under the Gaussian assumption Equation (7.11)–Equation (7.12), the expectation step of Equation (7.9) simplifies to (Fessler and Hero, 1994)

$$\bar{\boldsymbol{x}}^{(k)} = \mathbb{E}(\mathbf{x}|\boldsymbol{m}^{\boldsymbol{\alpha}}, \boldsymbol{f}^{(k)}), \tag{7.14}$$

$$\mathcal{Q}(\boldsymbol{f}|\boldsymbol{f}^{(k)}) = \log p(\bar{\boldsymbol{x}}^{(k)}|\boldsymbol{f}) + \log p(\boldsymbol{f}). \tag{7.15}$$

Using classical properties of Gaussian vectors (see Chapter 5 of (Tarantola, 2005)), we can rewrite these two steps

$$\bar{\boldsymbol{x}}^{(k)} = \underset{\boldsymbol{x}}{\operatorname{argmin}} \|\boldsymbol{S}\boldsymbol{x} - \boldsymbol{m}^\alpha\|^2_{\tilde{\boldsymbol{\Sigma}}_\alpha^{-1}} + \|\boldsymbol{x} - \boldsymbol{H}\boldsymbol{f}^{(k)}\|^2_{\boldsymbol{\Sigma}^{-1}} \tag{7.16a}$$

$$\boldsymbol{f}^{(k+1)} = \underset{\boldsymbol{f}}{\operatorname{argmin}} \|\bar{\boldsymbol{x}}^{(k)} - \boldsymbol{H}\boldsymbol{f}\|^2_{\boldsymbol{\Sigma}^{-1}} - \log p(\boldsymbol{f}). \tag{7.16b}$$

As $p(\boldsymbol{f})$ is an unknown, we propose to replace Equation (7.16b) by a nonlinear model $\mathcal{D}_{\boldsymbol{\omega}}$

$$\bar{\boldsymbol{x}}^{(k)} = \underset{\boldsymbol{x}}{\operatorname{argmin}} \|\boldsymbol{S}\boldsymbol{x} - \boldsymbol{m}^\alpha\|^2_{\tilde{\boldsymbol{\Sigma}}_\alpha^{-1}} + \|\boldsymbol{x} - \boldsymbol{H}\boldsymbol{f}^{(k)}\|^2_{\boldsymbol{\Sigma}^{-1}} \tag{7.17a}$$

$$\boldsymbol{f}^{(k+1)} = \mathcal{D}_{\boldsymbol{\omega}}(\boldsymbol{H}^\top \bar{\boldsymbol{x}}^{(k)}) \tag{7.17b}$$

where $\boldsymbol{\omega}$ represents the parameters of the model $\mathcal{D}_{\boldsymbol{\omega}}$, which will be optimized during the training phase, such that Equation (7.17b) solves Equation (7.16b). (Aggarwal, Mani, and Jacob, 2019).

In the literature on deep unrolled methods, Equation (7.17a) is commonly called the data-consistency layer. Introducing the variable $\boldsymbol{y}^{(k)} = \bar{\boldsymbol{x}}^{(k)} - \boldsymbol{H}\boldsymbol{f}^{(k)}$, the analytical solution of Equation (7.17a) is given by (see appendix B)

$$\boldsymbol{y}^{(k)} = \begin{bmatrix} \boldsymbol{\Sigma}_1 \\ \boldsymbol{\Sigma}_{21} \end{bmatrix} (\tilde{\boldsymbol{\Sigma}}_\alpha + \boldsymbol{\Sigma}_1)^{-1} (\boldsymbol{m}^\alpha - \boldsymbol{S}\boldsymbol{H}\boldsymbol{f}^{(k)}), \tag{7.18}$$

where $\boldsymbol{\Sigma}_1 \in \mathbb{R}^{M\times M}$, $\boldsymbol{\Sigma}_{21} \in \mathbb{R}^{(N-M)\times M}$, and $\boldsymbol{\Sigma}_2 \in \mathbb{R}^{(N-M)\times(N-M)}$ are the blocks of the covariance $\boldsymbol{\Sigma}$

$$\boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_1 & \boldsymbol{\Sigma}_{21}^\top \\ \boldsymbol{\Sigma}_{21} & \boldsymbol{\Sigma}_2 \end{bmatrix}. \tag{7.19}$$

The computational burden of inverting the signal-dependant matrix $\tilde{\boldsymbol{\Sigma}}_\alpha + \boldsymbol{\Sigma}_1$ in Equation (7.18) is alleviated by the consistency condition of Equation (7.13), which implies that $\boldsymbol{S}\boldsymbol{\Sigma}\boldsymbol{S}^\top = \boldsymbol{\Sigma}_1$ is diagonal. Denoting $\boldsymbol{\sigma}_1^2 = \mathsf{diag}\,(\boldsymbol{\Sigma}_1)$, we get

$$\boldsymbol{y}_1^{(k)}(\boldsymbol{m}^\alpha) = \boldsymbol{\sigma}_1^2/(\boldsymbol{\sigma}_1^2 + \tilde{\boldsymbol{\sigma}}_\alpha^2)(\boldsymbol{m}^\alpha - \boldsymbol{S}\boldsymbol{H}\boldsymbol{f}^{(k)}). \tag{7.20}$$

where the division and multiplication apply element-wise.

Finally, we can summarize our algorithm by the following four steps

$$\boldsymbol{y}_1^{(k)} = \boldsymbol{\sigma}_1^2/(\boldsymbol{\sigma}_1^2 + \tilde{\boldsymbol{\sigma}}_\alpha^2)(\boldsymbol{m}^\alpha - \boldsymbol{S}\boldsymbol{H}\boldsymbol{f}^{(k)}) \tag{7.21a}$$

$$\boldsymbol{y}_2^{(k)} = \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_1^{-1}\boldsymbol{y}_1^{(k)} \tag{7.21b}$$

$$\bar{\boldsymbol{f}}^{(k)} = \boldsymbol{f}^{(k)} + \boldsymbol{H}^\top \boldsymbol{y}^{(k)} \tag{7.21c}$$

$$\boldsymbol{f}^{(k+1)} = \mathcal{D}_{\boldsymbol{\omega}}(\bar{\boldsymbol{f}}^{(k)}), \tag{7.21d}$$

which can be seen as the unrolled network with shared weights depicted in Fig. 7.1. At each iteration, the first layer denoises the measurements, the second completes the missing measurements, the third maps the completed measurements to the image space, and the last denoises the solution in the image space.

We refer to this network as EM-Net, and denote it by

$$\boldsymbol{f}^{(K)} = \mathcal{G}_{\boldsymbol{\theta}}^K(\boldsymbol{m}^\alpha) \tag{7.22}$$

where K is the number of EM iterations. For $\boldsymbol{f}^{(0)} = \boldsymbol{0}$, the case $K = 1$ corresponds to the denoised completion network (*i.e.*, the DC-Net) proposed in (Lorente Mur et al., 2021).

7.3.4 EM-Net training strategy

Given an image database $\{\boldsymbol{f}_s\}$, $1 \leq s \leq S$ and the corresponding measurements $\{\boldsymbol{m}_s^\alpha\}$ and $1 \leq s \leq S$, simulated via Equation (7.2)–Equation (7.3), we train the neural network in an end-to-end fashion

$$\min_{\boldsymbol{\theta}} \frac{1}{S}\sum_{s=1}^{S} \|\mathcal{G}_{\boldsymbol{\theta}}^{(K)}(\boldsymbol{m}_s^\alpha) - \boldsymbol{f}_s\|_2^2 + \lambda\|\boldsymbol{\theta}\|_2^2, \tag{7.23}$$

where $\boldsymbol{\theta}$ represent the learnable parameters of the network, K is the number of iterations, and λ is the weight decay parameter. We set $\boldsymbol{f}^{(0)} = \boldsymbol{0}$ and $K = 5$.

Here, we chose to learn $\boldsymbol{\sigma}_1$, the error in our estimation of $\boldsymbol{\Sigma}_\alpha$ and the parameters of the image-domain denoiser; *i.e.*, $\boldsymbol{\theta} = \{\boldsymbol{\omega}, \boldsymbol{\sigma}_1\}$. Learning $\boldsymbol{\sigma}_1$ allows us to compensate for the inaccuracy in the estimation of $\boldsymbol{\Sigma}_\alpha$ in Equation (7.7). We precomputed $\boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_1^{-1}$, which drastically reduced the number of parameters to be optimized. While $\boldsymbol{\Sigma}_1$ is diagonal, when combined with $\boldsymbol{\Sigma}_{21}$, this no longer holds. To precompute $\boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_1^{-1}$, we exploit the Bayesian interpretation of Equation (7.16), whereby $\boldsymbol{H}\boldsymbol{f}^{(k)}$ and $\boldsymbol{\Sigma}$ are the mean and covariance of the prior $\mathbf{x}$. Therefore, we first train a one-iteration EM-Net $\mathcal{G}_{\boldsymbol{\theta}}^{(1)}$, and estimate $\boldsymbol{\Sigma}$ as

$$\boldsymbol{\Sigma} = \frac{1}{S-1}\sum_{s=1}^{S} \boldsymbol{H}(\boldsymbol{f}_s - \boldsymbol{f}_s^{(1)})(\boldsymbol{f}_s - \boldsymbol{f}_s^{(1)})^\top \boldsymbol{H}^\top. \tag{7.24}$$

where $\boldsymbol{f}_i^{(1)} = \mathcal{G}_{\boldsymbol{\theta}}^{(1)}(\boldsymbol{m}^\alpha)$.

7.4 Experiments

7.4.1 Comparison methods

We compare the proposed EM-Net with four methods. We first consider a hand-crafted MAP solution where the prior $\log p(\boldsymbol{f})$ is the TV, which we compute using the split-Bregman algorithm (Goldstein and Osher, 2009). The other three methods

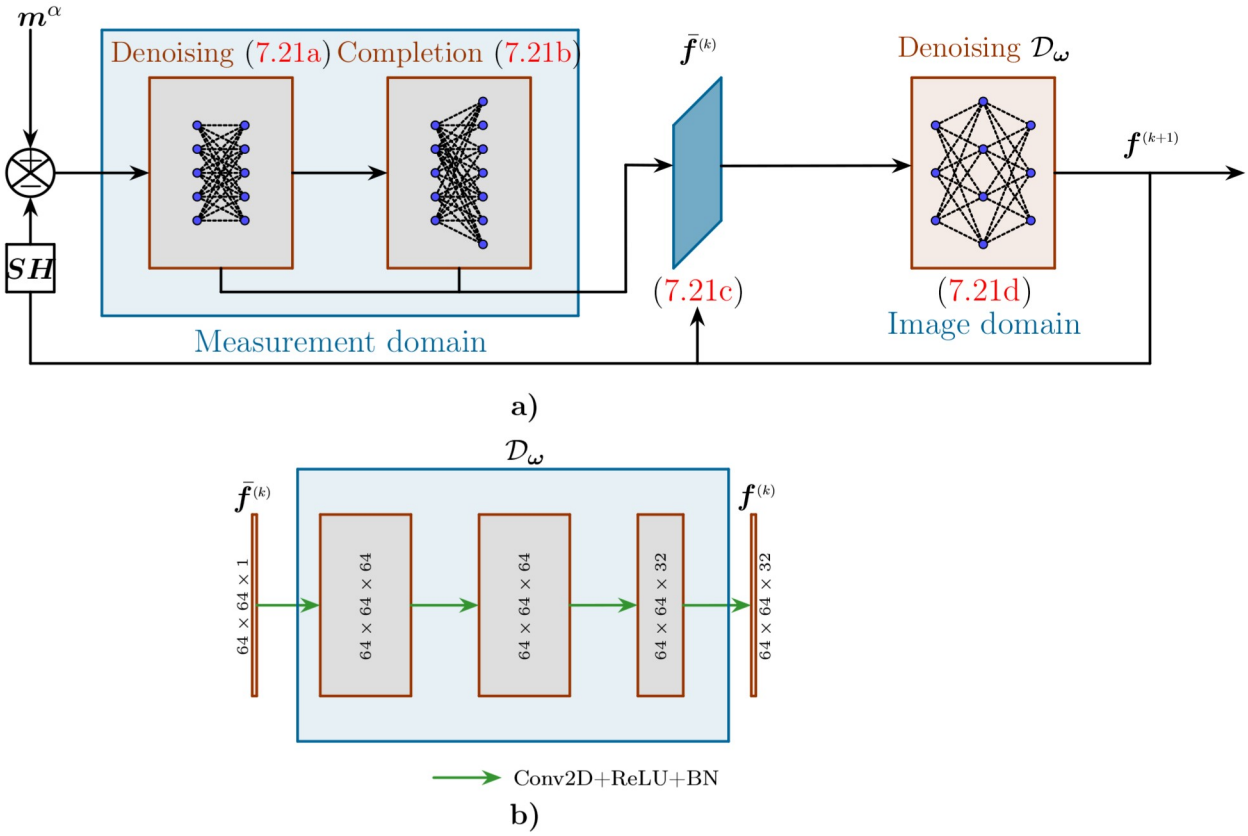


FIGURE 7.1 – Proposed EM-Net framework for single-pixel image reconstruction. **(a)** The recursive algorithm that alternates between completion and denoising in the measurement domain for Equations (7.21a),(7.21b), and denoising in the image domain for Equation (7.21d). **(b)** Architecture of the neural network in charge of the image domain denoising step.

are data-driven approaches based on deep neural networks. First, we postprocess the Moore-Penrose pseudo-inverse using a neural network chosen as either the CNN presented in Fig. 7.1b or a U-Net (Ronneberger et al., 2015). We refer to the two variants as CNN and U-Net, respectively. Secondly, we consider the model-based reconstruction with deep learned priors (MoDL) (Aggarwal, Mani, and Jacob, 2019). MoDL is an unrolled algorithm with a convolutional neural-network-based regularization prior. Finally, we consider a Neumann network (NN) (Gilton, Ongie, and Willett, 2020), as a neural-network architecture based on the Neumann series decomposition.

Model-based reconstruction with deep learned priors is a particular case of the proposed deep-EM algorithm, which assumes additive signal-independent Gaussian noise; *i.e.*, $\boldsymbol{\Sigma}_{21} = \mathbf{0}$ and $\boldsymbol{\Sigma}_1 = \lambda \boldsymbol{I}$ in Equation (7.21), where λ is a regularization term. CNN and U-Net are one-iteration EM-Nets that further assume no noise; *i.e.*, $K = 1$ and $\boldsymbol{\Sigma}^{\alpha} = \mathbf{0}$.

7.4.2 Datasets

We train the networks using two different datasets, and also consider an experimental dataset to test them.

7.4.2.1 STL-10

The STL-10 (Coates, Ng, and Lee, 2011) dataset is an image dataset that contains 120,000 images, as 100,000 unlabeled images, 5,000 labeled images called the 'training' set, and 8,000 images called the 'test' set. Our methods are trained using the 105,000 images from the training and unlabeled sets, and are tested on the test set. We randomly crop 64×64 patches from the original 96×96 images, and transform these into gray-scale images.

7.4.2.2 ImageNet

The Imagenet dataset (Deng et al., 2009) is a large-scale dataset that includes a training set of 1,281,167 images, and 50,000 validation images. Here, we consider the 'down-sampled Imagenet' (Chrabaszcz, Loshchilov, and Hutter, 2017) dataset, where all of the images have been resized to 64×64 images.

7.4.2.3 SPIHIM

The SPIHIM dataset (Ducros and Lorente Mur, 2020) contains 20 hyperspectral cubes acquired with a single-pixel camera. Each hyperspectral image has 2,048 channels in the visible range; however, we limit ourselves to the channel $\lambda = 610$ nm. As the dataset has a small number of hyperspectral images, we did not train any neural network based on this dataset; instead, we used this dataset to validate the generalization of our algorithms.

7.4.3 Metrics

We evaluate our reconstructed images with the peak signal-to-noise ratio (PSNR) and the structural similarity (SSIM). Given the ground-truth image $\boldsymbol{f}$, we compute the PSNR of a reconstructed image $\boldsymbol{f}^*$ as

$$\mathsf{PSNR}(\boldsymbol{f}^*, \boldsymbol{f}) = 10 \log_{10} \frac{2^2}{\|\boldsymbol{f}^* - \boldsymbol{f}\|_2^2}. \tag{7.25}$$

Similarly, we compute the SSIM as

$$\mathsf{SSIM}(\boldsymbol{f}, \boldsymbol{f}^*) = \frac{(2\mu\mu_* + c_1)(2\tilde{\sigma} + c_2)}{(\mu^2 + \mu_*^2 + c_1)(\sigma^2 + \sigma_*^2 + c_2)} \tag{7.26}$$

where μ is the average of $\boldsymbol{f}$, μ_* is the average of $\boldsymbol{f}^*$, σ is the variance of $\boldsymbol{f}$, σ_* is the variance of $\boldsymbol{f}^*$, $\tilde{\sigma}$ is the covariance of $\boldsymbol{f}$, and $\boldsymbol{f}^*$, $c_1 = (k_1 L)^2$ and $c_2 = (k_2 L)^2$ are two variables that are meant to stabilize the division, where L is the dynamic range of pixel values, and $k_1 = 0.01$ and $k_2 = 0.03$ by default.

7.4.4 Implementation and training details

In our experiments, we consider $M = 512$ and $M = 1,024$ Hadamard patterns of size $N = 64 \times 64$ pixels from both the STL-10 and Imagenet databases. We set the number of iterations to $K = 5$ for all of the iterative networks (*i.e.*, MoDL, NN, EM-Net). MoDL, NN, and the proposed EM-Net rely on a learned image-domain denoiser. For MoDL and NN, we chose this as a U-net, which is the state-of-the-art architecture. For EM-Net, we considered a tiny CNN with fewer parameters. The numbers of learned parameters were 28,993 for CNN, 499,985 for U-net, 499,986 for MoDL, 499,986 for NN, and 28,993+M for EM-Net. The weights of the image-domain denoiser in MoDL, NN, and EM-Net were initialised as the weights of a Gaussian denoiser. All of the networks were implemented using Pytorch (Paszke et al., 2019), and trained in a end-to-end fashion for the same noise level corresponding to $\alpha = 10$ photons. We considered the ADAM optimiser for 100 epochs, with an initial step size of 10^{-3} that we divided by a factor of 5 every 10 epochs. We set the weight decay regularization parameter to 10^{-6}. We stopped the training early after 60 epochs if the networks had reached convergence (*i.e.*, the relative variations of the cost function were below 1%).

7.5 Results and discussion

7.5.1 Reconstruction metric for the STL-10 and ImageNet datasets

Table 7.1 gives the comparisons of the reconstruction qualities of the four methods on 200 images from the STL-10 test set. All of the networks were trained using the STL-10 dataset for $\alpha = 10$ ph. Similarly, the four methods are compared for 500 images of the ImageNet test in Table 7.2. All of the networks were trained with the ImageNet dataset for $\alpha = 10$ ph. To explore the robustness of the methods in the presence of noise, the levels of which are unknown in practice, all of the networks were tested at a higher noise level ($\alpha = 3$ photons) and a lower noise level ($\alpha = 50$ photons).

According to our simulations at $M = 1,024$, the best performing methods are EM-Net at higher noise ($\alpha = 3$ photons) and MoDL at lower noise ($\alpha = 50$ photons). For $M = 1,024$, TV, CNN, and MoDL perform relatively well at low levels of noise, with PSNR values typically less than 1 dB below the best PSNR (-1.67 dB for CNN, -0.56 dB for U-Net, with MoDL as the best at 20.02 dB). At the high level of noise, their reconstruction PSNRs are significantly lower than the best one (-6.12 dB for CNN, -1.03 dB for U-Net, -3.71 dB for MoDL). The opposite was seen for NN, which yields good PSNR for higher noise (only -0.52 dB compared to EM-Net), but performs poorly for lower noise (-2.85 dB compared to MoDL). The same trend is observed for $M = 512$, except that NN also strongly degrades at higher noise (-1.41 dB compared to EM-Net). For high levels of noise, the PSNR might be worse for $M = 1,024$ than

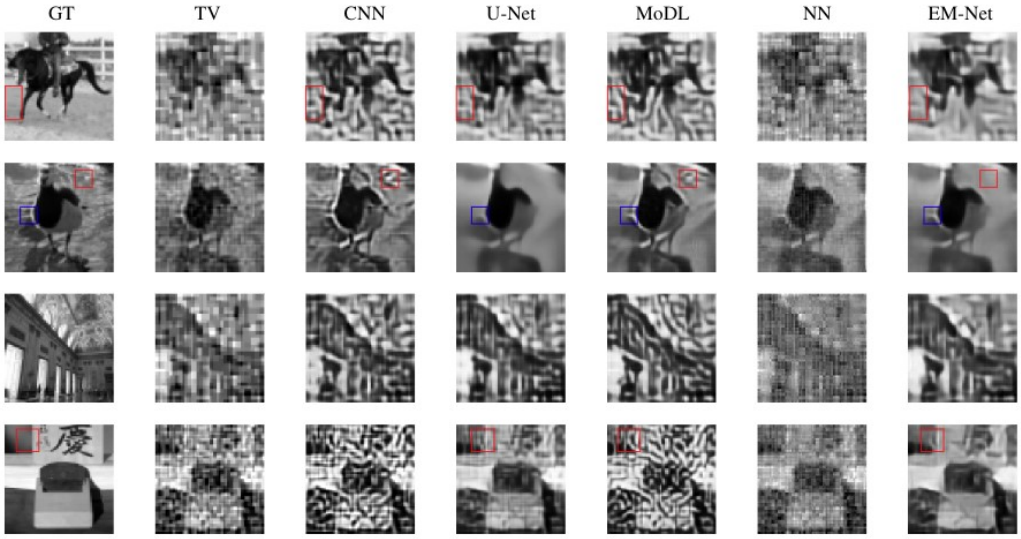


FIGURE 7.2 – Reconstructions of the simulated images using the different methods. Top row: STL-10 horse image with $M = 512$ and $\alpha = 3$ ph. Second row: STL-10 duck with $M = 1,024$ and $\alpha = 30$ ph. Third row: ImageNet hall with $M = 512$ and $\alpha = 2$ ph. Fourth row: ImageNet couch with $M = 1,024$ and $\alpha = 5$ ph. The images were reconstructed from simulated measurements from the ground-truth (GT) image. The following columns show reconstructions using total variation (TV), the ConvNet reconstructor presented in Fig. 7.1b (CNN), a U-Net reconstructor (Ronneberger et al., 2015) (U-net),a model-based reconstruction with deep learned priors (Aggarwal, Mani, and Jacob, 2019) (MoDL), and the proposed method (EM-Net).

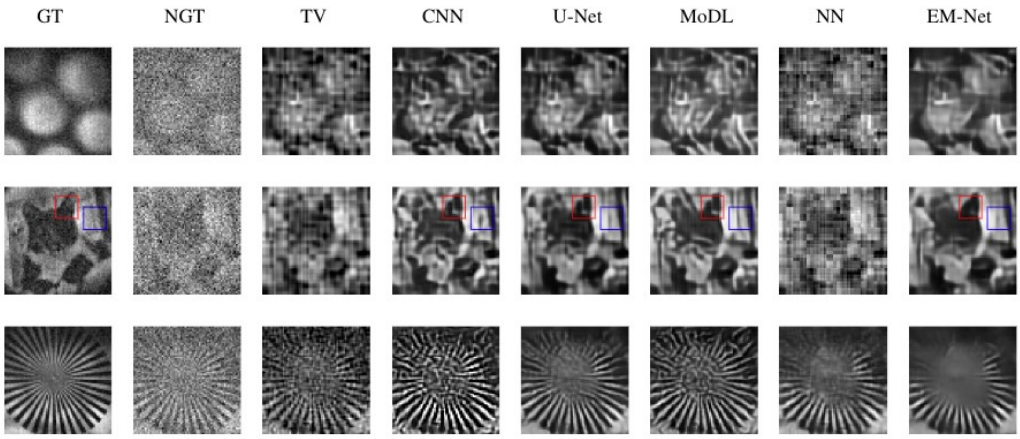


FIGURE 7.3 – Reconstructions of the three experimental datasets using the six different methods. Top row: LED lamp with $M = 512$ measurements. Second row: STL-10 cat with $M = 512$ measurements. Third row: Siemens star with $M = 2,048$ measurements. The images were reconstructed from a fully sampled dataset (ground-truth; GT) acquired with high image intensity (first column: $\alpha = 148$ ph, $\alpha = 195$ ph, $\alpha = 295$ ph) and a noisy fully sampled dataset (noisy ground-truth; NGT) acquired at lower image intensity (second column: $\alpha = 9$ ph, $\alpha = 10$ ph, $\alpha = 14$ ph). The following columns show reconstructions from the down-sampled NGT image using total variation (TV), the ConvNet reconstructor presented in Fig. 7.1b (CNN), a U-Net reconstructor (Ronneberger et al., 2015) (U-net), a model-based reconstruction with deep learned priors (Aggarwal, Mani, and Jacob, 2019) (MoDL), and the proposed method (EM-Net).

		$M = 512$		$M = 1024$	
		$\alpha = 3$	$\alpha = 50$	$\alpha = 3$	$\alpha = 50$
TV	PSNR	14.67±1.46	19.23±3.71	12.78±0.93	19.36±3.7
	SSIM	0.73±0.14	**0.94±0.04**	0.63±0.17	0.94±0.04
CNN	PSNR	13.2±1.05	19.19±3.75	9.67±0.82	18.35±3.23
	SSIM	0.66±0.16	**0.94±0.05**	0.43±0.19	0.92±0.05
Unet	PSNR	16.17±2.04	19.29±3.75	14.76±1.43	19.46±3.83
	SSIM	0.79±0.13	0.92±0.07	0.72±0.16	0.93±0.07
Modl	PSNR	14.8±1.55	**19.47±3.87**	12.08±0.78	**20.02±4.19**
	SSIM	0.73±0.15	0.93±0.07	0.58±0.18	**0.95±0.05**
NN	PSNR	14.87±1.58	18.3±3.26	15.27±2.1	17.07±3.21
	SSIM	0.6±0.27	0.71±0.33	0.39±0.43	0.48±0.47
EM-Net	PSNR	**16.28±2.06**	19.44±3.89	**15.79±1.79**	19.81±4.09
	SSIM	**0.8±0.13**	0.93±0.06	**0.78±0.16**	**0.95±0.05**

TABLE 7.1 – Peak signal-to-noise ratios (PSNR) and structural similarities (SSIM) for the different reconstruction methods. The metrics are computed from 200 simulated images from the STL-10 dataset. All of the neural networks were trained using the STL-10 dataset with $\alpha = 10$ photons, and were tested for $\alpha = 3$ and $\alpha = 50$ photons for different compression rates ($M = 512$, $M = 1,024$). Results are shown for total variation (TV), direct Unet reconstructor (Unet), model-based reconstruction with deep learned priors (MoDL), Neumann networks (NN), and the proposed EM-Net. **Blue** indicates the highest PSNRs; **green** indicates the highest SSIMs.

for $M = 512$. This is explained on the basis that mostly noise is acquired for the high frequency coefficients. This addition of noisy information can highly degrade the image quality despite the higher number of acquisitions.

The proposed EM-Net method is very effective in a wide varieties of situations. At high noise (*i.e.*, $\alpha = 3$ ph), EM-Net is the best performing method in terms of PSNR and SSIM for both the STL-10 and ImageNet images, whatever the compression ratio. At lower noise (*i.e.*, $\alpha = 50$ ph), EM-Net is the second-best performing method, and is only slightly behind the best performing method (-0.03 dB and -0.21 dB compared to MoDL for $M = 512$ and $M = 1,024$, respectively).

In summary, the metrics indicate that EM-Net is among the best methods that we assessed here, with excellent reconstruction under noise levels that differ from that used during training, which is often the case in real-life applications, and also under different compression ratios.

7.5.2 Visual assessment of the reconstructed images

7.5.2.1 Simulated data

Fig. 7.2 shows the assessment of our deep EM-Net with data simulated from the STL-10 and ImageNet databases. In particular, we consider two images from the STL-10 dataset, a horse and a duck. We simulate the acquisition of the horse image with $M = 512$ measurements and $\alpha = 3$ ph, and that of the duck with $M = 1,024$ measurements and $\alpha = 30$ ph. We also consider two images from the ImageNet test set, a couch and a hall. We simulate the acquisition of the couch with $M = 1,024$ measurements and $\alpha = 5$ ph, and that of the hall with $M = 512$ measurements and $\alpha = 2$ ph.

		$M = 512$		$M = 1024$	
		$\alpha = 3$	$\alpha = 50$	$\alpha = 3$	$\alpha = 50$
TV	PSNR	14.07±1.83	17.89±4.01	12.42±1.29	18.11±4.02
	SSIM	0.71±0.14	**0.92±0.06**	0.62±0.15	0.93±0.05
CNN	PSNR	12.94±1.35	17.89±4.12	10.49±0.82	17.3±3.66
	SSIM	0.65±0.17	**0.92±0.06**	0.49±0.17	0.91±0.06
Unet	PSNR	15.27±2.35	17.97±4.09	14.16±1.79	18.2±4.21
	SSIM	0.77±0.14	0.91±0.08	0.71±0.15	0.91±0.08
Modl	PSNR	14.03±1.83	**18.22±4.28**	10.88±0.82	**18.65±4.53**
	SSIM	0.7±0.17	0.91±0.08	0.51±0.17	**0.94±0.06**
NN	PSNR	14.82±2.35	16.2±3.18	14.34±2.01	17.42±3.79
	SSIM	0.51±0.35	0.6±0.34	0.6±0.26	0.72±0.32
EM-Net	PSNR	**15.52±2.46**	18.08±4.21	**15.17±2.23**	18.41±4.4
	SSIM	**0.79±0.14**	0.91±0.08	**0.77±0.15**	0.92±0.07

TABLE 7.2 – Peak signal-to-noise ratios (PSNR) and structural similarities (SSIM) for the different reconstruction methods. The metrics were computed from 200 simulated images from the ImageNet dataset. All of the neural networks were trained using the ImageNet dataset with $\alpha = 10$ photons, and were tested for $\alpha = 3$ and $\alpha = 50$ photons for different compression rates ($M = 512$, $M = 1,024$). Results are shown for total variation (TV), direct Unet reconstructor (Unet), model-based reconstruction with deep learned priors (MoDL), Neumann networks (NN), and the proposed EM-Net. **Blue** indicates the highest PSNRs; **green** indicates the highest SSIMs.

In all of the simulations, the TV reconstruction suffers from grid-like or staircase artefacts. NN suffers from similar artifacts. While it preserves more image details at the lower compression ratio (*i.e.*, $M = 1,024$), the NN reconstruction presents strong artifacts at the high compression ratio (*i.e.*, $M = 512$). CNN, U-Net, MoDL, and EM-Net perform visually better than TV and NN. However, these reconstructions have different behaviors, as indicated in Fig. 7.2 using red and blue rectangles. The red rectangles highlight areas where the ground-truth images are mostly flat, while the blue rectangle highlights an area where a detail is present. In the red areas, the CNN, U-Net, and MoDL reconstructions tend to include worm-like artifacts that are not seen for EM-Net (*e.g.*, compare the image homogeneity within the red rectangles of the horse image in Fig. 7.2). On the other hand, despite low noise conditions, UNet blurs fine details, such as the white detail in the blue rectangle, while MoDL and EM-Net mostly keep these fine details intact.

In summary, U-net, MoDL, and the proposed EM-Net are visually best performing. Among these methods, EM-Net does not show most of the grid-like and worm-like artifacts at low photon counts, while it preserves the fine details at higher photon counts, which can be blurred with U-net and MoDL. This visual assessment confirms the excellent robustness of EM-Net to a wide variety of simulation parameters.

7.5.2.2 Experimental Data

Fig. 7.3 shows the assessment of the different methods using experimental data from the Spihim experimental dataset. In particular, this includes the images of a LED lamp, a cat from the STL-10 dataset printed on a transparent sheet, and the 32 branch Siemens star (*Photography — Electronic still picture imaging — Resolution and spatial frequency responses* 2019). We display the images that we reconstruct

with fully sampled measurements (*i.e.*, $M = 4,096$) in the first and second columns of Fig. 7.3. The first column corresponds to very low noise acquisitions, and the second to the higher noise acquisitions that we assessed for these methods. We estimated the image intensities for each of the acquisitions, and obtained $\tilde{\alpha} = 9$ ph for the LED, $\tilde{\alpha} = 10$ ph for the cat, and $\tilde{\alpha} = 14$ ph for the star sector. Then, we down-sampled the acquisitions *a posteriori*, keeping $M = 512$ measurements for the LED and for the cat, and $M = 1,024$ for the star sector.

For the LED lamp, as for the simulated measurements, there are grid-like artefacts in the TV and NN reconstructions, which are most visible at the borders of the LED disks and in the background. On the other hand, the CNN, MoDL, Unet, and EM-Net reconstructors are more smooth. Here again, CNN, Unet, and MoDL present worm-like artefacts, while EM-Net leads to a more homogeneous background and LED disks.

For the STL-10 cat, as for the LED lamp, high-frequency grid artefacts are seen for the TV and NN reconstructions, while the other methods are visually more similar. However, EM-Net removes some of the reconstruction artifacts seen in CNN, UNet, and TV (*e.g.*, compare the red rectangles in the second row of Fig. 7.3; the ear of the cat is mostly artifact free using EM-Net). We further note that EM-Net preserves image details well (*e.g.*, compare the blue rectangles in the second row of Fig. 7.3; EM-Net preserves the edge of the door while keeping the image intensity closer to the ground-truth grayscale value).

For the siemens star sector, CNN and MoDL best reconstruct the branches of the star sector. However, both of these methods also generate strong artifacts at the center of the target. The other methods have comparable performances with regards to the reconstruction of the high frequency branches, while EM-Net smooths the center of the target out the most. This object has a high-frequency content, and structures that are not like any of the images in the training set, which partially explains these results.

7.6 Conclusions and perspectives

We propose an iterative network based on the EM algorithm. This deep EM network can solve linear under-determined inverse problems where the measurements are corrupted by normally distributed signal-dependent noise (*e.g.*, Poisson noise, mixed Poisson-Gaussian noise, mixed Skellam-Gaussian noise). The EM-Net alternates over four steps, which have straightforward interpretation, as measurement denoising, measurement completion, measurement to image domain mapping, and image denoising.

Most unrolled networks assume additive Gaussian noise with constant variance. Therefore, they tend to generalize poorly to different noise levels, and might be ill-suited to applications where the noise levels cannot be predicted beforehand. On the contrary, the proposed EM-Net method explicitly estimates the noise covariance to denoise the measurement domain. As a consequence, EM-Net generalizes very well to noise levels that differ to that used during training. In particular, it outperforms all of the other methods assessed here when in the presence of noise with variance higher than that used during training, and it is comparable to the best method in the presence of noise, with lower variance.

Chapter 8

Deep Kalman adaptive sampling for real-time video reconstruction

So far during this thesis we have considered static cases, where we recover the current image of the scene through the sole use of the current acquired measurements. While this type of algorithm can be used sequentially to reconstruct images, the acquired images do not use the previously acquired images as prior information. Furthermore, this sequential approach would not allow to reduce the number of acquired coefficients while keeping roughly the same image quality. This chapter aims to develop methods that recover the current image using the previous acquired images. To do so, we give ourselves a predictive model for natural videos. This predictive model will allow to determine the prior image, to determine which coefficients we need to sample, and to reconstruct our current image.

In sec. 8.1, we study the implications of a linear predictive model. In sec. 8.2, we offer a downsampling scheme based on deep learning variance estimation. Finally in sec. 8.3, we show the different perspectives offered by the introduction of predictive generative neural networks.

8.1 Linear dynamic models, and Kalman filtering for single-pixel reconstruction

8.1.1 Linear model

To begin with, we consider the case where our predictive model is a linear one. This means that our model is in the form

$$\boldsymbol{f}_t = \boldsymbol{F}_{t-1}\boldsymbol{f}_{t-1} + \boldsymbol{q}_{t-1}, \tag{8.1a}$$
$$\boldsymbol{m}_t^\alpha = \boldsymbol{H}_t\boldsymbol{f}_t + \boldsymbol{\epsilon}_t^\alpha, \tag{8.1b}$$
$$\boldsymbol{q}_{t-1} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_{t-1}), \tag{8.1c}$$
$$\boldsymbol{\epsilon}_t^\alpha \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma}_\alpha), \tag{8.1d}$$

where $\boldsymbol{f}_t \in \mathbb{R}^N$ is the t^{th} frame of the video, $\boldsymbol{F}_t \in \mathbb{R}^{N\times N}$ is the linear predictive matrix, $\boldsymbol{q}_t \in \mathbb{R}^N$ is the t^{th} predictive noise vector, $\boldsymbol{\epsilon}_t^\alpha \in \mathbb{R}_t^M$ is the t^{th} measurement noise vector and $\boldsymbol{m}_t^\alpha \in \mathbb{R}_t^M$ is the t^{th} measurement vector. Eq. 8.1b is consistent with the measurement vector presented in Ch. 1. As a choice for $\boldsymbol{F}_t$, the most straightforward intuitive model for it would be the identity matrix i.e. $\boldsymbol{F}_t = \boldsymbol{I} \quad \forall t > 0$ (Ding, Chen, and Wassell, 2014; Bugeau et al., 2010).

This model assumes that every frame is fairly similar to the previous one though Eq. 8.1a, and that the slight differences between each frame can be interpreted as a noise $\boldsymbol{q}_t$ that represents the drift to the model. To further simplify this model, we assume that the covariance of the predictive noise is constant through all frames (i.e.

$\boldsymbol{Q}_t = \boldsymbol{Q} \quad \forall \quad t > 0$) and compute the covariance model of our predictive noise through a dataset of videos :

$$\mathbf{Q} = \frac{1}{T * S - 1} \sum_{i=1}^{N} \sum_{t=1}^{T} (\mathbf{f}_t^i - \mathbf{f}_{t-1}^i)(\mathbf{f}_t^i - \mathbf{f}_{t-1}^i)^\top, \tag{8.2}$$

where $\{\boldsymbol{f}_{1:T}\}_{1 \leq i \leq S}$ is the video dataset made up of S videos of T frames.

Finally, the linear model that we consider can be summarized as

$$\boldsymbol{f}_t = \boldsymbol{f}_{t-1} + \boldsymbol{q}_{t-1}, \tag{8.3a}$$
$$\boldsymbol{m}_t^\alpha = \boldsymbol{H}_t \boldsymbol{f}_t + \boldsymbol{\epsilon}_t^\alpha, \tag{8.3b}$$
$$\boldsymbol{q}_{t-1} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{Q}), \tag{8.3c}$$
$$\boldsymbol{\epsilon}_t^\alpha \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_\alpha). \tag{8.3d}$$

Following the results of section 4.2, the Bayesian solution to 8.3 would be normally distributed

$$\mathbf{f}_t | \boldsymbol{m}_{1:t-1}^\alpha \sim \mathcal{N}(\boldsymbol{f}_t^-, \boldsymbol{P}_t^-), \tag{8.4}$$
$$\mathbf{f}_t | \boldsymbol{m}_{1:t}^\alpha \sim \mathcal{N}(\boldsymbol{f}_t^+, \boldsymbol{P}_t^+). \tag{8.5}$$

The different parameters of the Bayesian solution can further be analytically computed through the prediction step, and the update step.

The prediction step gives

$$\boldsymbol{f}_t^- = \boldsymbol{f}_{t-1}^+, \tag{8.6a}$$
$$\boldsymbol{P}_t^- = \boldsymbol{P}_{t-1}^+ + \boldsymbol{Q}_{t-1}. \tag{8.6b}$$

Meanwhile, the update step gives

$$\boldsymbol{f}_t^+ = \boldsymbol{f}_t^- + \boldsymbol{P}_t^- \boldsymbol{H}_t^\top (\boldsymbol{H}_t \boldsymbol{P}_t^- \boldsymbol{H}_t^\top + \boldsymbol{\Sigma}_{\alpha t})^{-1} [\boldsymbol{m}_t^\alpha - \boldsymbol{H}_t \boldsymbol{f}_t^-], \tag{8.7a}$$
$$\boldsymbol{P}_t^+ = \boldsymbol{P}_t^- - \boldsymbol{P}_t^- \boldsymbol{H}_t^\top (\boldsymbol{H}_t \boldsymbol{P}_t^- \boldsymbol{H}_t^\top + \boldsymbol{\Sigma}_{\alpha t})^{-1} \boldsymbol{H}_t \boldsymbol{P}_t^-. \tag{8.7b}$$

8.1.2 Flaws of the linear model

We consider the case of a video of a white point in a black background moving towards one side at the speed of one pixel per frame. Every frame is fairly similar to the previous frame, so the model above should be able to be applied. This example will help us identify the issues of the Kalman filter faces when applying the linear model.

Numerical simulation

In order to assess the effectiveness, we compare in fig. 8.1 some methods based on the Kalman filter on a video of a white spot on a black background. First of all, our wish is to reduce the number of acquired measurements on every frame in order to improve measurement and reconstruction times. To evaluate this, we reconstruct the first frame using the generalised Tikhovov regularisation with $M = 512$ measurements, while for the following frames we only acquire 200 measurements. First of all, we wish to evaluate the performance of the Kalman filter assuming that for every iteration, we use the same measurement patterns i.e. $\forall t > 0 \quad \boldsymbol{H}_t = \boldsymbol{H}_1$. In this specific case, the predictive model $\boldsymbol{f}_t = \boldsymbol{F}_{t-1} \boldsymbol{f}_t$ is known (it is a permutation matrix), and we can apply the Kalman equations on with $\boldsymbol{F}_{t-1}$ instead of the identity. We assume

$\forall t > 0 \quad \boldsymbol{H}_t = \boldsymbol{H}_1$ and call this case 'Exact Kalman' in fig. 8.1. In order to assess the viability of the identity as predictive model, we also use the Kalman equations with the model presented in eq. 8.3, where $\forall t > 0 \quad \boldsymbol{H}_t = \boldsymbol{H}_1$. We refer to this case as 'Kalman' in fig. 8.1. We wish to compare the Kalman filtering methods with a purely static method such as the generalised Tikhonov solution with 200 measurements assuming $\forall t > 0 \quad \boldsymbol{H}_t = \boldsymbol{H}_1$. We refer to this case as 'Tikhonov' in fig. 8.1. Finally, we wish to study the hypothesis that adaptively sampling over time helps improve the quality of the Kalman filtering methods. The idea is that we should not measure the low-frequency Hadamard coefficients on every frame, but rather measure the coefficients that change the most from one frame to the other. In the row 'Adap Kalman', we choose $\boldsymbol{H}_t$ as the Hadamard downsampling matrix that selects the coefficients of highest variance. In this situation, we use an exact estimate of the variance using the ground truth.

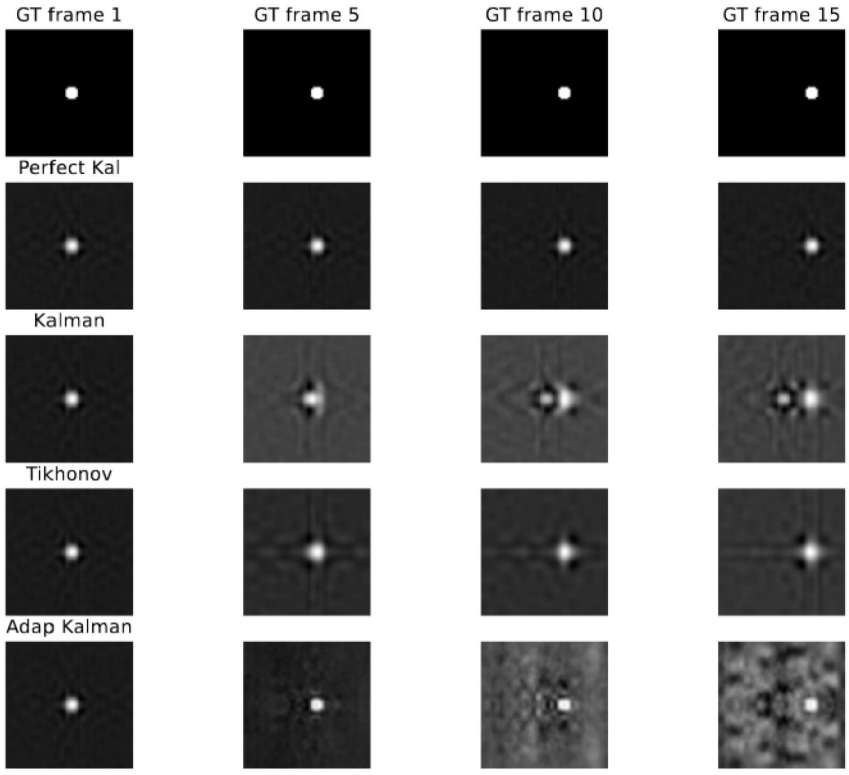


FIGURE 8.1 – Reconstruction methods The first frame is estimated via the Tikhonov regularisation (obtained by computing 2.21) with 512 measurements. For the rest of the frames, we estimate the reconstructed image considering 200 measurements with 'Exact Kalman' : the Kalman filtering equations when the predictive model is perfect (as a translation); 'Kalman' : the Kalman filtering equations are used with the model presented in Eq. 8.3 and the acquired coefficients are the same for every frame; 'Tikhonov' : the generalised Tikhonov regularised solution (obtained by computing 2.21) 'Adap Kalman' : the Kalman filtering equations are used with the model presented in Eq. 8.3 but where the sampled coefficients are acquired based on the exact estimation of the variance.

Results

We observe that 'Exact Kalman' is able to generate a sequence of frames with the same resolution as the first frame. In other words, despite having only 200 measurements, we are able to retain the reconstructed quality of 512 measurements. In this case, as the predictive model is perfectly known, the predictive model makes up for the loss in information over time. We also observe that 'Kalman' produces an artefact that retains the mark from the first frame. As a matter of fact, it would seem like 'Kalman' is the average between 'Tikhonov' and the first frame. In order to explain this, we simplify the equations of the Kalman filter. As seen previously, the output of the update step in the Kalman filter solves

$$\mathbf{f}_t^+ = \underset{\mathbf{f}}{\operatorname{argmin}} \|\mathbf{H}_t\mathbf{f} - \mathbf{m}_t^\alpha\|^2_{(\mathbf{\Sigma}_t^\alpha)^{-1}} + \|\mathbf{f} - \mathbf{f}_{t-1}^+\|^2_{(\mathbf{P}_t^-)^{-1}} \tag{8.8}$$

We consider the simpler case where we assume $\mathbf{\Sigma}_t^\alpha = \lambda^2\mathbf{I}$, and $\mathbf{P}_t^- = \mu^2\mathbf{I}$. This leads to considering the solution to the inverse problem

$$\mathbf{f}_t^+ = \underset{\mathbf{f}}{\operatorname{argmin}} \|\mathbf{H}_t\mathbf{f} - \mathbf{m}_t^\alpha\|^2 + \rho^2\|\mathbf{f} - \mathbf{f}_{t-1}\|^2, \tag{8.9}$$

where $\rho = \lambda/\mu$. The analytical $\boldsymbol{z}_t = \boldsymbol{H}\boldsymbol{f}_t^+$ solution in the Hadamard domain would be

$$\mathbf{z}_t[m] = \begin{cases} \frac{\mathbf{m}_t^\alpha[m] + \rho\mathbf{z}_{t-1}[m]}{1+\rho}, & \text{if the coefficient is acquired,} \\ \mathbf{z}_{t-1}[m], & \text{otherwise.} \end{cases} \tag{8.10}$$

We understand that the coefficients that are not acquired will remain the same from one frame to the other. This explains why we retain the information from the first frame into the later frames leading to the aforementioned artefacts. When the predictive model is well known this is not an issue since the non-acquired coefficients will evolve according to the exact predictive model. But when the predictive model is not exactly known, we need to acquire the coefficients that we assume will be most important through time leading to time-adaptive down-sampling schemes. 'Adap Kalman' shows that if we are capable of getting a perfect estimate of the variance for sampling purposes alone, then we are capable of removing the artefacts from the first frame and assess the nature of the movements of the scene. Over time, the model drift still causes the background to diverge from the ground truth, but for the first 10 frames, the adaptive sampling scheme alone is capable of reducing the artefacts caused by the model drift.

As we saw in sec. 1.2.3.2, optimal downsampling schemes for the ℓ_2 norm are based on the variance of the reconstructed image, alongside with the mean of the reconstructed image. This is indeed confirmed by the row 'Adap Kalman' in fig. 8.1. However, we still need to see how we can compute the variance of the image. An estimation of the variance can be obtained through the diagonal of $\boldsymbol{P}_t^+$. We can visualise the aforementioned estimation in fig 8.2. As we can see, the Gaussian approximation leads to highly homogeneous variance estimates making it unsuitable for object-based sampling.

In sec. 8.2, we will explore solutions based on this very concept that explore the use of Deep learning to compute the aforementioned estimates of the variance to choose the downsampling schemes. And in sec. 8.3, we explore the perspectives opened by the field of video prediction in order to prevent the model drift from happening as much as possible.

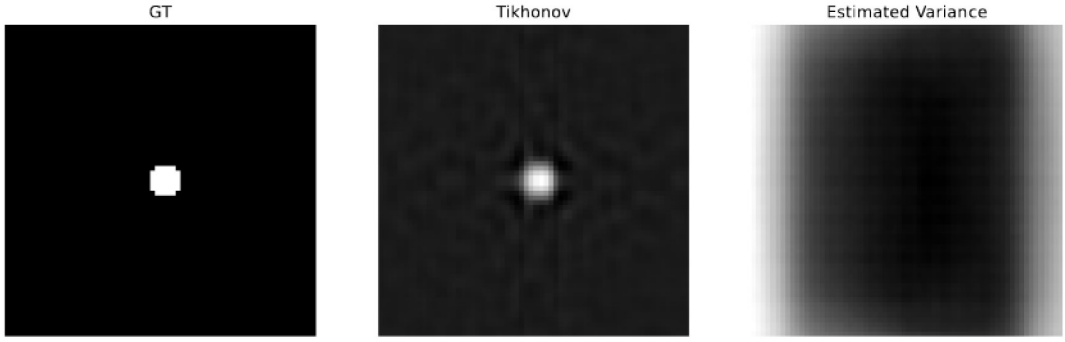


FIGURE 8.2 – Mean and variance of the linear estimates using the Bayesian inversion formulas with Gaussian prior (see sec. 2.3.2). 'GT' is the ground truth image; 'Tikhonov' is the Tikhonov regularised solution to the inverse problem (see eq. 2.21) and represents the mean of the Gaussian estimator; 'Estimated variance' is the diagonal of the covariance of the posterior Gaussian density function obtained by computing Eq. 2.22.

8.2 Deep-learning variance estimation for dynamic down-sampling

8.2.1 Training a neural-network to compute the variance

Based on the results shown in sec. 8.1.2, we aim to obtain an algorithm to express the variance generated by linear estimators such as the ones delivered by the Kalman filter. We can train a neural network to compute the variance of a linear estimator,

$$\mathbf{f}_t^+ = \mathcal{M}_t(\mathbf{m}_t^\alpha) = \boldsymbol{\mu}_t + \boldsymbol{\Sigma}\mathbf{H}_t^\top(\mathbf{H}_t\boldsymbol{\Sigma}_t\mathbf{H}_t^\top + \boldsymbol{\Sigma}_{\alpha,t})^{-1}(\mathbf{m}_t^\alpha - \mathbf{H}_t\boldsymbol{\mu}_t), \tag{8.11}$$

for all values of $\boldsymbol{\mu}_t$, $\boldsymbol{H}_t$, $\boldsymbol{\Sigma}_t$, and $\boldsymbol{\Sigma}_{\alpha,t}$. To do so, we need to find a function $\mathcal{C}$ that solves

$$\mathcal{C}_\theta = \underset{\theta}{\operatorname{argmin}} \frac{1}{s}\sum_{i=0}^{s} \|\mathcal{C}_\theta(\mathbf{m}_t^{\alpha(i)}) - (\mathbf{f}_t^{(i)} - \mathcal{M}_t(\mathbf{m}_t^{\alpha(i)}))\|_2^2 \tag{8.12}$$

As said previously though, our estimates linear estimates need to be acquires adaptively (i.e. with different values of $\boldsymbol{H}_t$ for each frame). This is an issue as it does mean that we need to consider all potential downsampling schemes when training.

In order to avoid the aforementioned problem, we will simply assume that the other frames have similar image quality as the first frame. This is consistent with our observations in Fig 8.1, where the frames have similar image quality to the first one despite having fewer measurements. This means that we will train our neural network under the conditions of the first reconstructed frame, and apply it on the rest of the reconstructed frames.

In other words, we aim to find an estimator of the variance for the linear estimator

$$\mathbf{f}^+ = \mathcal{M}_0(\mathbf{m}^\alpha) = \boldsymbol{\mu} + \boldsymbol{\Sigma}\mathbf{H}_0^\top(\mathbf{H}_0\boldsymbol{\Sigma}\mathbf{H}_0^\top + \boldsymbol{\Sigma}_\alpha)^{-1}(\mathbf{m}^\alpha - \mathbf{H}_0\boldsymbol{\mu}). \tag{8.13}$$

Since we intend to apply it to different frames with different linear reconstructors but with similar properties, we will give $\mathcal{C}_\theta$ the shape

$$\mathcal{C}_\theta = \mathcal{G}_\theta \circ \mathcal{M}_0, \tag{8.14}$$

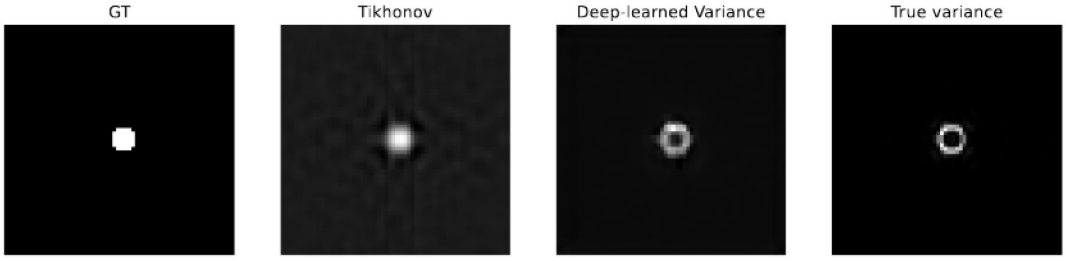


FIGURE 8.3 – Estimated variance of Tikhonov reconstructors via deep-learning. 'GT' is the ground truth image; 'Tikhonov' is the Tikhonov regularised solution to the inverse problem (see Eq. 2.21); 'Deep learned variance' is the estimated variance of the Tikhonov reconstructor via Deep-learning; 'True Variance' is the variance estimated from the ground truth and the image obtained via Tikhonov regularisation.

and we train $\mathcal{G}_\theta$ to solve

$$\mathcal{G}_\theta = \underset{\theta}{\operatorname{argmin}} \frac{1}{s} \sum_{i=0}^{s} \|\mathcal{G}_\theta(\mathcal{M}_0(\mathbf{m}^{\alpha(i)})) - (\mathbf{f}^{(i)} - \mathcal{M}_0(\mathbf{m}^{\alpha(i)}))\|_2^2. \tag{8.15}$$

We trained a neural network to solve that cost function using the stl-10 (Coates, Ng, and Lee, 2011) database. The chosen architecture is the one presented in Figure 7.1. The training conditions are the same as the ones in Ch. 5, with the only difference being that of the cost function.

In fig. 8.3, we compare the results yielded by such a neural network on the first reconstructed frame compared to the ground truth estimation of the variance computed from the ground truth image. As we can see while there is a small error in terms of range, it still retains the important information with respect to which regions of the image are the most prone to error with respect to the linear estimator.

8.2.2 Using a neural network to choose the down-sampling masks

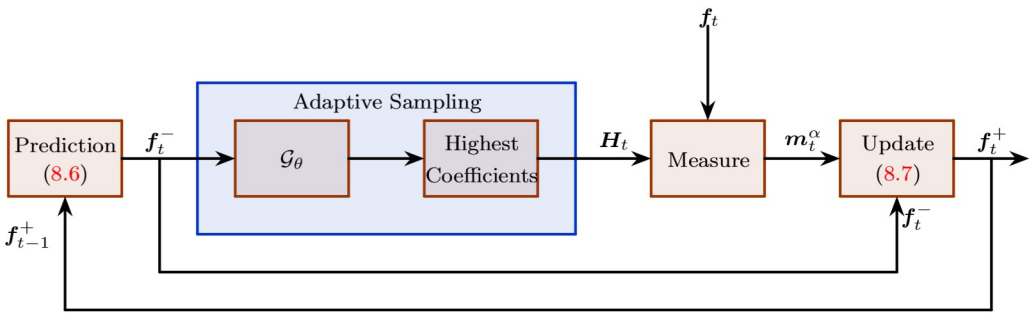


FIGURE 8.4 – Variance-based time adaptive downsampling algorithm for single-pixel imaging.

Based on our observations in sec. 8.1.2 and the results with respect to variance estimation in sec. 8.2.1, we propose the variance-based time adaptive subsampling algorithm presented in fig. 8.4. As seen in fig. 8.4, after the prediction step is finished, we compute the variance

$$\boldsymbol{v}_t^- = \mathcal{G}_\theta(\boldsymbol{f}_t^-), \tag{8.16}$$

and then we select $\boldsymbol{H}_t$ as the subsampled Hadamard matrix that where the coefficients of $\boldsymbol{v}_t^-$ are maximal. The measures are then acquired by uploading $\boldsymbol{H}_t$ into the DMD to acquire the measurement vector $\boldsymbol{m}_t^\alpha$. Finally, our final estimate is obtained by applying the update equations 8.7.

Algorithm 5 Variance-based time adaptive downsampling algorithm for single-pixel imaging.

Initialisation : Compute the first estimate $\boldsymbol{f}_0^+ = \mathcal{M}_0(\boldsymbol{m}_0^\alpha)$.
for $t \in \{1, \dots, T\}$ **do**
1 - Prediction Step : Compute
$\boldsymbol{f}_t^- = \boldsymbol{f}_{t-1}^+$,
$\boldsymbol{P}_t^- = \boldsymbol{P}_{t-1}^+ + \boldsymbol{Q}_{t-1}$.
2 - Select the coefficients :
Select the coefficients that maximise $\mathcal{G}_\theta(\boldsymbol{f}_t^-) + (\boldsymbol{f}_t^-)^2$,
Acquire the corresponding measurements $\boldsymbol{m}_t^\alpha$.
3 - Update step : Compute
$\boldsymbol{f}_t^+ = \boldsymbol{f}_t^- + \boldsymbol{P}_t^- \boldsymbol{H}_t^\top (\boldsymbol{H}_t \boldsymbol{P}_t^- \boldsymbol{H}_t^\top + \boldsymbol{\Sigma}_{\alpha t})^{-1} [\boldsymbol{m}_t^\alpha - \boldsymbol{H}_t \boldsymbol{f}_t^-]$,
$\boldsymbol{P}_t^+ = \boldsymbol{P}_t^- - \boldsymbol{P}_t^- \boldsymbol{H}_t^\top (\boldsymbol{H}_t \boldsymbol{P}_t^- \boldsymbol{H}_t^\top + \boldsymbol{\Sigma}_{\alpha t})^{-1} \boldsymbol{H}_t \boldsymbol{P}_t^-$.
end for

In Fig. 8.5, we compare the results yielded by this algorithm 'DL Adaptive Kalman' (Deep learned based adaptive Kalman) with the methods discussed in sec. 8.1.2. As we can see, the reconstructed image is similar to the one obtained via adaptive sampling with exact estimations of the variance. This method is capable of capturing the movement of objects for the first 10 frames. After 15 frames, the background starts to get corrupted with artefacts, and we would probably need to refresh the image with new longer acquisitions such as the one on the first frame.

These results show that adaptive sampling allows us to partially overcome the model drift from the linear approximation for some frames, but that eventually, we would still suffer from it. In sec. 8.3, we explore some perspectives to tackle the issues posed by predictive model drift.

8.3 Perspectives : predictive generative neural network for predictive particle filtering

8.3.1 Predictive neural networks

Video prediction (Oprea et al., 2020) is a recent paradigm that has highly benefited from the advent of Deep learning. The idea of video prediction is that one may predict future frames of a video given the past frames. While initially the models that were considered for video prediction were deterministic ones (see e.g (Srivastava, Mansimov, and Salakhutdinov, 2015)), these models would normally compute the expected frame that would be somewhat blurry. Since predicting the future is a highly uncertain task, more recent video prediction methods have aimed to draw samples of the future frames given previous frames. Most of the sampling-based video prediction schemes use Generative adversarial networks (Liang et al., 2017; Lee et al., 2018; Aigner and Körner, 2018) which leads to predictive schemes in the form

$$\boldsymbol{f}_{t+1} = \tilde{\mathfrak{F}}_\theta(\boldsymbol{f}_t, \boldsymbol{q}_t), \tag{8.17}$$
$$\boldsymbol{q}_t \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}), \tag{8.18}$$

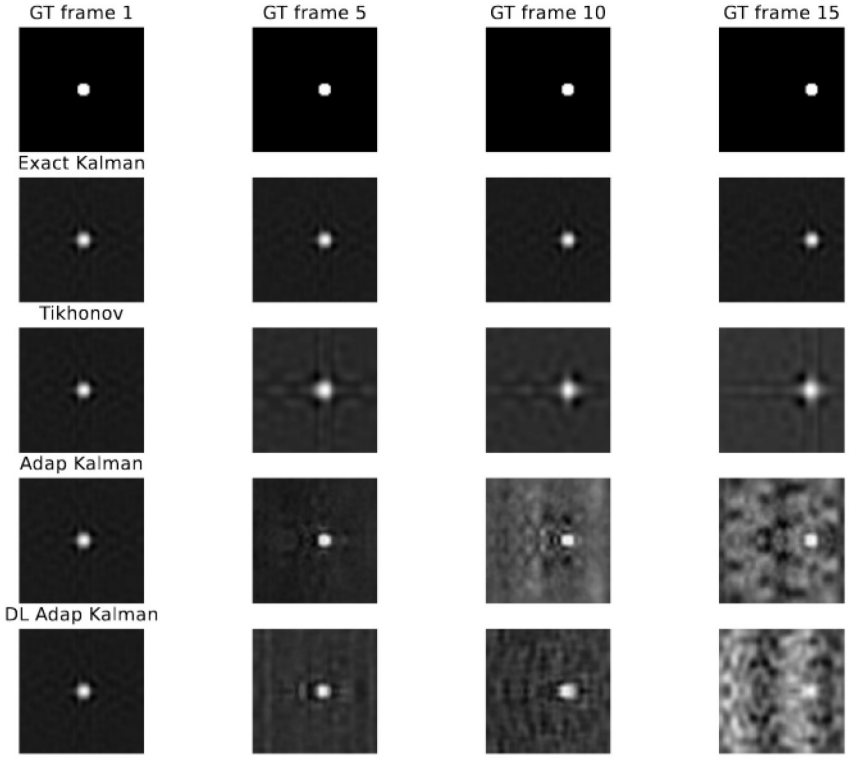


FIGURE 8.5 – Reconstruction methods The first frame is estimated via the Tikhonov regularisation (obtained by computing 2.21) with 512 measurements. For the rest of the frames, we estimate the reconstructed image considering 200 measurements with 'Exact Kalman' : the Kalman filtering equations when the predictive model is perfect (as a translation); 'Tikhonov' : the generalised Tikhonov regularised solution (obtained by computing 2.21); 'Adap Kalman' : the Kalman filtering equations are used with the model presented in Eq. 8.3 but where the sampled coefficients are acquired based on the exact estimation of the variance; 'DL Adap Kalman' : the Kalman filtering equations are used with the model presented in Eq. 8.3 but where the sampled coefficients are acquired based on the estimation of the variance given by a deep neural network.

where $\mathfrak{F}_\theta$ is a deep-neural network. This sort of predictive model is consistent with the theory of non-linear Bayesian state estimation presented in sec. 8.1.2. We will explore here two potential algorithms to explore for real-time video reconstruction.

8.3.2 Extended Kalman filtering through predictive neural networks

As explored in sec. 4.3.1, the most straightforward approach to non-linear state estimation is the linearisation of the non-linear predictive model via a Taylor expansion of $\mathfrak{F}_\theta$. The resulting posterior distribution would then be approximated as a Gaussian distribution, and we would compute the prediction step as

$$\boldsymbol{f}_t^- = \mathfrak{F}_\theta(\boldsymbol{f}_{t-1}, \mathbf{0}), \tag{8.19}$$

$$\boldsymbol{P}_t^- = \mathbb{C}\text{ov}(\mathbf{f}_t|\boldsymbol{m}_{1:t-1}^\alpha) = \boldsymbol{F}_{t-1}\boldsymbol{P}_{t-1}^+\boldsymbol{F}_{t-1}^\top + \boldsymbol{L}_{t-1}\boldsymbol{L}_{t-1}^\top, \tag{8.20}$$

where $\boldsymbol{F}_{t-1} = \frac{\partial \mathfrak{F}_\theta}{\partial \boldsymbol{f}}(\boldsymbol{f}_{t-1}, \mathbf{0})$ and $\boldsymbol{L}_{t-1} = \frac{\partial \mathfrak{F}_\theta}{\partial \boldsymbol{q}}(\boldsymbol{f}_{t-1}, \mathbf{0})$. The update step on the other hand would remain exactly the same as the one in sec. 8.1.

This method has two potential downsides. Firstly, the computation of $\boldsymbol{F}_{t-1}$ and $\boldsymbol{L}_{t-1}$ could be too demanding computationally. This could make it unsuitable for real-time applications unless we manage to find an alternative algorithm to compute those matrices in an efficient way. The second issue could be that due to the numerous non-linear layers of $\mathfrak{F}_\theta$, the linear approximation could deviate the expectancy and the covariance.

8.3.3 Predictive networks for particle filtering estimation

If the extended Kalman filter fails to capture the evolution over time of the neural network-based predictive model, we can also turn to solutions based on sequential importance sampling (see sec. 4.3.3) such as the particle filter. The idea being to sample K samples $\{\boldsymbol{f}_t^{(i)}\}_{1\leq K}$ from an auxiliary distribution $\pi(\boldsymbol{f}_t|\boldsymbol{f}_{1:t-1}, \boldsymbol{m}_{1:t}^\alpha)$, and then compute statistical estimators in the shape of

$$\mathbb{E}(\boldsymbol{f}_t|\boldsymbol{m}_{1:t}^\alpha) \approx \sum_{i=1}^{K} w_t^{(i)} \boldsymbol{f}_t^i, \tag{8.21}$$

where the weights $\{w^{(i)}\}_{1\leq i\leq K}$ need to be computed. A potential choice of $\pi(\boldsymbol{f}_t|\boldsymbol{f}_{1:t-1}, \boldsymbol{m}_{1:t}^\alpha)$ in the case of a predictive model based on a generative neural network could simply be to choose $\pi(\boldsymbol{f}_t|\boldsymbol{f}_{1:t-1}, \boldsymbol{m}_{1:t}^\alpha) = p(\boldsymbol{f}_t|\boldsymbol{f}_{t-1})$. This choice is straightforward as Eq. 8.18 makes sampling $p(\boldsymbol{f}_t|\boldsymbol{f}_{t-1})$ very easy and computationally efficient, but evaluating $p(\boldsymbol{f}_t|\boldsymbol{f}_{t-1})$ for a given $(\boldsymbol{f}_t, \boldsymbol{f}_{t-1})$ would still be challenging. This would lead to Algorithm 6.

Algorithm 6 Particle filtering algorithm for single-pixel video reconstruction with Generative neural network prediction.

Initialisation : Draw $\{\boldsymbol{f}_0^{(i)}\}_{1\leq i\leq S} \sim p(\boldsymbol{f}_0)$, S samples drawn from the prior distribution. Set all the weights $w_0^{(i)} = 1/S$.
for $t \in \{1, \ldots, T\}$ **do**
1 - Draw S $\{\boldsymbol{f}_t^{(i)}\}_{1\leq i\leq S}$ samples from the importance distribution :
sample $\boldsymbol{q}_{t-1}^{(i)} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$,
$\boldsymbol{f}_t^{(i)} = \mathfrak{F}_\theta(\boldsymbol{f}_{t-1}^{(i)}, \boldsymbol{q}_{t-1}^{(i)})$.
2 - Compute the non-normalized weights $\tilde{w}_t^{(i)}$
$\tilde{w}_t^{(i)} = p(\boldsymbol{m}_t^\alpha | \boldsymbol{f}_t^{(i)})$
3 - Normalize the weights $\tilde{w}_t^{(i)}$ to obtain the normalized weights $w_t^{(i)}$ that sum to unity.
4 - Perform re-sampling on $\{\boldsymbol{f}_t^{(i)}, w_t^{(i)}\}_{1\leq i\leq S}$ (see Annex E)
end for

One advantage of this algorithm is that it's easy to implement. Furthermore, unlike unscented transform-based methods, we also get to choose the number of samples we wish to draw. While the number of samples impacts on the error of the estimator, it is worth considering a compromise between the desired precision, and the reconstruction speed we desire.

Conclusion

In this chapter, we explore reconstruction algorithms based on Bayesian state estimation methods. This allowed us to develop reconstruction schemes that aim to reduce the number of acquired measurements by using the information from the previously acquired frames. In particular, we developed a time adaptive downsampling scheme based on variance estimation using deep learning. Our experiments indicate that this method would allow to reduce for about 10 frames from 512 measurements to 200 measurements and still retain similar image quality. We further presented some perspectives that remain to be explored with respect to the use of generative adversarial networks as a predictive model. Said generative network could be integrated in a particle filter, or be linearized through an extended Kalman filter.

Conclusion and perspectives

Conclusion Single-pixel imaging is an imaging modality that captures bi-dimensional images with a single point detector. By selecting the single point detector as a spectral analyser, the single-pixel camera becomes a hyperspectral camera with the spectral resolution of the chosen spectral analyser. The goal of this thesis was to investigate reconstruction and acquisition algorithms for the single-pixel camera that would enable real-time hyperspectral single-pixel imaging.

Single-pixel reconstruction leads to a linear inverse problem that we chose to solve using deep learning mainly due to its reconstruction speed. One of the main challenges with deep learning based reconstruction is the lack of interpretability of deep learned reconstructors. This makes them challenging to use on experimental data since the 'black box' approach to deep learning makes the learned models susceptible to changes in the levels of noise. Our contributions on that topic include a Tikhonov based mapping from the measurement domain to the image domain that we integrate in a deep learning framework. We further demonstrated the applicability of said method to experimental data from a single-pixel camera in chapter 5, and from a hyperspectral single-pixel camera in chapter 6. furthermore, in chapter 7, we generalised the aforementioned method by proposing a deep learning architecture based on the expectation-maximisation algorithm. The proposed algorithms are designed with Skellam Gaussian noise models, and have shown sufficient generalisation to noise levels that differ from the noise levels on the training phase.

While these models offer quick image reconstruction (of a few milliseconds per image), they do not take into account the potential spatiotemporal redundancies in natural videos. Doing so would open the path to a potential reduction to the number of acquired coefficients. Therefore, we have opted for a Kalman filtering approach to exploit the information from previous frames. Using this framework, we came to the conclusion that unless the update model in the system is perfectly known, reducing the number of acquired measurements would result in some artefacts with residual inaccurate information from previous frames. To counter this, in chapter 8, we proposed a time adaptive sampling scheme, where we acquire for every frame the coefficients that we assume have the highest variation with respect to the previous frame.

Overall, the proposed static reconstruction methods have been tested on 64×64 images, and have shown state-of-the-art reconstruction quality for compression rates of 88%. Our algorithms rely on a three step approach : first we normalise and denoise the acquired measurements, then we estimate the missing measurements in the Hadamard domain, finally, we denoise the image in the image domain. We showed that the compression rate can further be decreased to 5% using time adaptive down-sampling methods coupled with Kalman filtering equations.

Perspectives A first perspective to this thesis is the use of improved predictive models coupled with state estimation methods. These are presented in section 8.3. They aim to reduce the effects caused by by the drift between the real update equations, and the linear update equations.

Another perspective would be using hybrid single-pixel setups that exploit both the single-pixel measurements for hyperspectral imaging along with a grayscale image from a conventional camera to serve as a prior for sampling and for reconstruction purposes.

Part III

Appendix

Appendix A

Likelihood of mixed Skellam Gaussian variables

Lemma A.0.1. *Let X and Y be two independent real-valued, one-dimensional random variables whose respective probability density functions are p_X and p_Y. The random variable $Z = X + Y$ has a probability density function equal to the convolution of p_X and p_Y, i.e. $p_Z = p_X * p_Y$.*

Proof. We note $p_{X,Y}$, as the joint probability density function of (X, Y). Using the definition of P_Z, the cumulative density function, we derive

$$P_Z(z) = \mathbb{P}(Z \leq z) = \mathbb{P}(Y \leq z - x), \tag{A.1}$$

$$= \int_{-\infty}^{+\infty} \int_{-\infty}^{z-x} p_{X,Y}(x, y)dxdy. \tag{A.2}$$

By substituting $y = v - x$, we get,

$$P_Z(z) = \int_{-\infty}^{+\infty} \int_{-\infty}^{z} p_{X,Y}(x, v - x)dxdv, \tag{A.3}$$

$$P_Z(z) = \int_{-\infty}^{z} dv \int_{-\infty}^{+\infty} p_{X,Y}(x, v - x)dx. \tag{A.4}$$

By the definition of probability density functions, we get

$$p_Z(z) = \int_{-\infty}^{+\infty} p_{X,Y}(x, z - x)dx. \tag{A.5}$$

Since X and Y are independent, finally,

$$p_Z(z) = \int_{-\infty}^{+\infty} p_X(x)p_Y(z - x)dx = p_X * p_Y(z). \tag{A.6}$$

□

Lemma A.0.2. *Let X and Y be two Poisson random variables of parameters μ_1, μ_2. The random variable $Z = X - Y$ is a Skellam random variable (Skellam, 1946) (noted $\mathcal{S}_k(\mu_1, \mu_2)$) of parameters μ_1 and μ_2, with probability mass function*

$$\mathbb{P}(Z = k) = e^{-(\mu_1+\mu_2)} \left(\frac{\mu_1}{\mu_2}\right)^{\frac{k}{2}} \mathcal{I}_{|k|}(2\sqrt{\mu_1\mu_2}), \tag{A.7}$$

where $\mathcal{I}_.$ are the modified Bessel functions of the first kind.

Proof. This proof simply uses the fact that in order for $Z = k$ to be true, it simply means that there is an integer n, such that $X = k + n$, and $Y = n$. This leads to the following decomposition.

$$\mathbb{P}(Z = k) = \sum_{n=-\infty}^{+\infty} \mathbb{P}(X = n + k, Y = n) \tag{A.8}$$

$$= \sum_{n=-\infty}^{+\infty} \mathbb{P}(X = n + k)\mathbb{P}(Y = n) \tag{A.9}$$

$$= \sum_{n=\max(0,-k)}^{+\infty} e^{-\mu_1} \frac{\mu_1^{k+n}}{(k+n)!} e^{-\mu_2} \frac{\mu_2^n}{(n)!} \tag{A.10}$$

By rearranging the terms, we get the desired result. □

Theorem A.0.3. *We consider* $\mathbf{m}^\alpha$ *computed as*

$$\mathbf{m} = \frac{\hat{\mathbf{m}}_+ - \hat{\mathbf{m}}_-}{\alpha K} \tag{A.11}$$

where,

$$\hat{\mathbf{m}}_{+,-i} \sim K\,\mathcal{P}(\alpha \boldsymbol{h}_i^{+,-\top} \boldsymbol{f}) + \mathcal{N}(\mu_{\text{dark}}, \sigma^2_{\text{dark}}). \tag{A.12}$$

The likelihood $p(\boldsymbol{m}|\boldsymbol{f})$ *of the measurement vector* $\boldsymbol{m}$ *given* $\boldsymbol{f}$ *is*

$$p(\boldsymbol{m}|\boldsymbol{f}) = \prod_{i=1}^{M} \sum_{n\in\mathbb{N}} e^{-\alpha(\boldsymbol{h}_{i,+}^\top \boldsymbol{f} + \boldsymbol{h}_{i,-}^\top \boldsymbol{f})} \left(\frac{\boldsymbol{h}_{i,+}^\top \boldsymbol{f}}{\boldsymbol{h}_{i,-}^\top \boldsymbol{f}}\right)^{\frac{n-\alpha \boldsymbol{h}_i^\top \boldsymbol{f}}{2\alpha}} \mathcal{I}_{|\frac{n}{\alpha} - \boldsymbol{h}^\top \boldsymbol{f}|}\left(2\alpha\sqrt{(\boldsymbol{h}_{i,+}^\top \boldsymbol{f})(\boldsymbol{h}_{i,-}^\top \boldsymbol{f})}\right) \frac{1}{\sqrt{2\pi\beta^2}} e^{-\frac{(\boldsymbol{m}_i - \frac{n}{\alpha})^2}{\beta^2}}. \tag{A.13}$$

Proof. We first rewrite the i-th component of $\boldsymbol{m}$ in eq. A.11, denoted m for simplicity, as

$$m = \boldsymbol{h}^\top \boldsymbol{f} + \epsilon_f + \epsilon, \tag{A.14}$$

where

$$\epsilon_f \sim \frac{1}{\alpha} \mathcal{S}_k(\alpha \boldsymbol{h}_+^\top \boldsymbol{f}, \alpha \boldsymbol{h}_-^\top \boldsymbol{f}) - \boldsymbol{h}^\top \boldsymbol{f}, \tag{A.15}$$

$$\epsilon \sim \mathcal{N}(0, \beta^2), \quad \beta^2 = \frac{2\sigma^2_{\text{dark}}}{K^2\alpha^2}. \tag{A.16}$$

where $\boldsymbol{h} = \boldsymbol{h}_+ - \boldsymbol{h}_-$ represents a row of the Hadamard matrix.

To derive the the probability mass function (PMF) of ϵ_f, we recall the PMF of a Skellam-distributed random variable $Z \sim \mathcal{S}_{\text{k}}(\mu_1, \mu_2)$ Skellam, 1946

$$\mathbb{P}(Z = k; \mu_1, \mu_2) = e^{-(\mu_1+\mu_2)} \left(\frac{\mu_1}{\mu_2}\right)^{\frac{k}{2}} \mathcal{I}_{|k|}(2\sqrt{\mu_1\mu_2}), \tag{A.17}$$

where $\mathcal{I}_\cdot$ are the modified Bessel functions of the first kind. By linear tranformation of (A.17) according to (A.15), we obtain $p_n = \mathbb{P}(\epsilon_f = \frac{n}{\alpha} - \boldsymbol{h}^\top \boldsymbol{f})$, $\forall n \in \mathbb{N}$,

$$p_n \quad = \quad e^{-\alpha(\boldsymbol{h}_+^\top \boldsymbol{f} + \boldsymbol{h}_-^\top \boldsymbol{f})} \left(\frac{\boldsymbol{h}_+^\top \boldsymbol{f}}{\boldsymbol{h}_-^\top \boldsymbol{f}}\right)^{\frac{n-\alpha \boldsymbol{h}^\top \boldsymbol{f}}{2\alpha}} \mathcal{I}_{|\frac{n}{\alpha} - \boldsymbol{h}^\top \boldsymbol{f}|}\left(2\alpha\sqrt{(\boldsymbol{h}_+^\top \boldsymbol{f})(\boldsymbol{h}_-^\top \boldsymbol{f})}\right). \tag{A.18}$$

Therefore, the probability density function (PDF) of ϵ_f is

$$p(\epsilon_f) = \sum_{n\in\mathbb{N}} p_n \delta(\epsilon_f - \frac{n}{\alpha} + \boldsymbol{h}^\top \boldsymbol{f}), \tag{A.19}$$

where δ denotes the Dirac delta function. Since the PDF of $E = \epsilon + \epsilon_f$ is the convolution of (A.19) with the PDF of ϵ (see lemma A.0.1), we have

$$p_E(E) = \sum_{n\in\mathbb{N}} p_n \frac{1}{\sqrt{2\pi\beta^2}} \exp\left(-\frac{(E - \frac{n}{\alpha} - \boldsymbol{h}^\top \boldsymbol{f})^2}{\beta^2}\right). \tag{A.20}$$

Next, we write the likelihood of the measurement as

$$p(m|\boldsymbol{f}) = p_E(m - \boldsymbol{h}^\top \boldsymbol{f}) \tag{A.21}$$

$$= \sum_{n\in\mathbb{N}} p_n \frac{1}{\sqrt{2\pi\beta^2}} \exp\left(-\frac{(m - \frac{n}{\alpha})^2}{\beta^2}\right). \tag{A.22}$$

Assuming independent measurements, the chain rule leads to

$$p(\boldsymbol{m}|\boldsymbol{f}) = \prod_{i=1}^{M} p(\boldsymbol{m}_i|\boldsymbol{f}). \tag{A.23}$$

Injecting (A.22) and the expression of p_n in (A.18) into (A.23), we finally obtain

$$p(\boldsymbol{m}|\boldsymbol{f}) = \prod_{i=1}^{M} \sum_{n\in\mathbb{N}} e^{-\alpha(\boldsymbol{h}_{i,+}^\top \boldsymbol{f} + \boldsymbol{h}_{i,-}^\top \boldsymbol{f})} \left(\frac{\boldsymbol{h}_{i,+}^\top \boldsymbol{f}}{\boldsymbol{h}_{i,-}^\top \boldsymbol{f}}\right)^{\frac{n-\alpha \boldsymbol{h}_i^\top \boldsymbol{f}}{2\alpha}}$$

$$\mathcal{I}_{|\frac{n}{\alpha} - \boldsymbol{h}^\top \boldsymbol{f}|}\left(2\alpha\sqrt{(\boldsymbol{h}_{i,+}^\top \boldsymbol{f})(\boldsymbol{h}_{i,-}^\top \boldsymbol{f})}\right) \frac{1}{\sqrt{2\pi\beta^2}} e^{-\frac{(\boldsymbol{m}_i - \frac{n}{\alpha})^2}{\beta^2}}. \tag{A.24}$$

□

Appendix B

Properties of the Gaussian distributions

Lemma B.0.1. *Let* $\mathbf{x} \in \mathbb{R}^N$ *and* $\mathbf{y} \in \mathbb{R}^M$ *be two Gaussian random variables such that*

$$\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}), \tag{B.1}$$

$$\mathbf{y}|\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{H}\boldsymbol{x}, \boldsymbol{\Sigma}_\alpha). \tag{B.2}$$

Where $\boldsymbol{M} \in \mathbb{R}^{M \times N}$. *The joint probability distribution of* $(\mathbf{x}, \mathbf{y})$ *is given as*

$$\begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\mu} \\ \boldsymbol{H}\boldsymbol{\mu} \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma} & \boldsymbol{\Sigma}\boldsymbol{H} \\ \boldsymbol{H}^\top\boldsymbol{\Sigma} & \boldsymbol{H}\boldsymbol{\Sigma}\boldsymbol{H}^\top + \boldsymbol{\Sigma}_\alpha \end{bmatrix} \right). \tag{B.3}$$

Proof. Due to the chain rule applied to the random variables $\mathbf{x}$ and $\mathbf{y}$, we get

$$p(\boldsymbol{x}, \boldsymbol{y}) = p(\boldsymbol{y}|\boldsymbol{x})p(\boldsymbol{x}). \tag{B.4}$$

This means that if $p(\boldsymbol{y}|\boldsymbol{x})$ and $p(\boldsymbol{x})$ are both Gaussian, then the resulting joint distribution is also Gaussian :

$$p(\boldsymbol{x}) \propto \exp(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}(\boldsymbol{x} - \boldsymbol{\mu})) \tag{B.5}$$

$$p(\boldsymbol{y}|\boldsymbol{x}) \propto \exp(-\frac{1}{2}(\boldsymbol{y} - \boldsymbol{H}\boldsymbol{x})^\top \boldsymbol{\Sigma}_\alpha(\boldsymbol{y} - \boldsymbol{H}\boldsymbol{x})) \tag{B.6}$$

Which means that

$$p(\boldsymbol{x}, \boldsymbol{y}) \propto \exp(-\frac{1}{2}[(\boldsymbol{x} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{x} - \boldsymbol{\mu}) + (\boldsymbol{y} - \boldsymbol{H}\boldsymbol{x})^\top \boldsymbol{\Sigma}_\alpha^{-1}(\boldsymbol{y} - \boldsymbol{H}\boldsymbol{x})]). \tag{B.7}$$

Let us consider

$$(\boldsymbol{x} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{x} - \boldsymbol{\mu}) + (\boldsymbol{y} - \boldsymbol{H}\boldsymbol{x})^\top \boldsymbol{\Sigma}_\alpha^{-1}(\boldsymbol{y} - \boldsymbol{H}\boldsymbol{x}) \tag{B.8}$$

$$= (\boldsymbol{x} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{x} - \boldsymbol{\mu}) + (\boldsymbol{y} - \boldsymbol{H}\boldsymbol{\mu} + \boldsymbol{H}\boldsymbol{\mu} - \boldsymbol{H}\boldsymbol{x})^\top \boldsymbol{\Sigma}_\alpha^{-1}(\boldsymbol{y} - \boldsymbol{H}\boldsymbol{\mu} + \boldsymbol{H}\boldsymbol{\mu} - \boldsymbol{H}\boldsymbol{x}) \tag{B.9}$$

$$= (\boldsymbol{x} - \boldsymbol{\mu})^\top (\boldsymbol{\Sigma}^{-1} + \boldsymbol{H}^\top \boldsymbol{\Sigma}^{-1}\boldsymbol{H})(\boldsymbol{x} - \boldsymbol{\mu}) - (\boldsymbol{x} - \boldsymbol{\mu})^\top \boldsymbol{H}^\top \boldsymbol{\Sigma}_\alpha^{-1}(\boldsymbol{y} - \boldsymbol{H}\boldsymbol{\mu}) - (\boldsymbol{y} - \boldsymbol{H}\boldsymbol{\mu})^\top \boldsymbol{\Sigma}_\alpha^{-1}\boldsymbol{H}(\boldsymbol{x} - \boldsymbol{\mu}) + (\boldsymbol{y} - \boldsymbol{H}\boldsymbol{\mu})^\top \boldsymbol{\Sigma}_\alpha^{-1}(\boldsymbol{y} - \boldsymbol{H}\boldsymbol{\mu}) \tag{B.10}$$

Which is equivalent to

$$p(\boldsymbol{x}, \boldsymbol{y}) \propto \exp\left(-\frac{1}{2} \left[\begin{bmatrix} \boldsymbol{x} - \boldsymbol{\mu} \\ \boldsymbol{y} - \boldsymbol{H}\boldsymbol{\mu} \end{bmatrix}^\top \begin{bmatrix} \boldsymbol{\Sigma}^{-1} + \boldsymbol{H}^\top \boldsymbol{\Sigma}^{-1}\boldsymbol{H} & \boldsymbol{H}^\top \boldsymbol{\Sigma}_\alpha^{-1} \\ \boldsymbol{\Sigma}_\alpha^{-1}\boldsymbol{H} & \boldsymbol{\Sigma}_\alpha^{-1} \end{bmatrix} \begin{bmatrix} \boldsymbol{x} - \boldsymbol{\mu} \\ \boldsymbol{y} - \boldsymbol{H}\boldsymbol{\mu} \end{bmatrix} \right] \right). \tag{B.11}$$

□

To conclude the proof, we just use Schur's complement formula for matrix inversion, which gives

$$\begin{bmatrix} \boldsymbol{\Sigma}^{-1} + \boldsymbol{H}^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{H} & \boldsymbol{H}^\top \boldsymbol{\Sigma}_\alpha^{-1} \\ \boldsymbol{\Sigma}_\alpha^{-1} \boldsymbol{H} & \boldsymbol{\Sigma}_\alpha^{-1} \end{bmatrix}^{-1} = \begin{bmatrix} \boldsymbol{\Sigma} & \boldsymbol{\Sigma} \boldsymbol{H} \\ \boldsymbol{H}^\top \boldsymbol{\Sigma} & \boldsymbol{H} \boldsymbol{\Sigma} \boldsymbol{H}^\top + \boldsymbol{\Sigma}_\alpha \end{bmatrix}. \tag{B.12}$$

Lemma B.0.2. *Let* $\mathbf{x} \in \mathbb{R}^N$ *and* $\mathbf{y} \in \mathbb{R}^N$ *have a joint Gaussian probability distribution*

$$\begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \boldsymbol{a} \\ \boldsymbol{b} \end{bmatrix}, \begin{bmatrix} \boldsymbol{A} & \boldsymbol{C} \\ \boldsymbol{C}^\top & \boldsymbol{B} \end{bmatrix} \right). \tag{B.13}$$

The marginal distributions of $\boldsymbol{x}$ *and* $\boldsymbol{y}$ *are given by*

$$\mathbf{x}|\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{a} + \boldsymbol{C}\boldsymbol{B}^{-1}(\boldsymbol{y} - \boldsymbol{b}), \boldsymbol{A} - \boldsymbol{C}\boldsymbol{B}^{-1}\boldsymbol{C}^\top) \tag{B.14}$$

$$\mathbf{y}|\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{b} + \boldsymbol{C}^\top \boldsymbol{A}^{-1}(\boldsymbol{x} - \boldsymbol{a}), \boldsymbol{B} - \boldsymbol{C}\boldsymbol{A}^{-1}\boldsymbol{C}) \tag{B.15}$$

Proof. We define

$$\mathbf{z} = \mathbf{x} - \boldsymbol{C}\boldsymbol{B}^{-1}\mathbf{y} \tag{B.16}$$

We can see that $\mathbf{z}$ has a multivariate Gaussian distribution as a linear combination of multivariate Gaussian variables. Furthermore, $\mathbf{y}$ and $\mathbf{z}$ are decorrelated :

$$\begin{aligned} \mathbb{C}\mathrm{ov}(\mathbf{z}, \mathbf{y}) &= \mathbb{C}\mathrm{ov}(\mathbf{x}, \mathbf{y}) + \mathbb{C}\mathrm{ov}(-\boldsymbol{C}\boldsymbol{B}^{-1}\mathbf{y}, \mathbf{y}), && \text{(B.17)} \\ &= \mathbb{C}\mathrm{ov}(\mathbf{x}, \mathbf{y}) - \boldsymbol{C}\boldsymbol{B}^{-1}\mathbb{C}\mathrm{ov}(\mathbf{y}, \mathbf{y}), && \text{(B.18)} \\ &= \boldsymbol{B} - \boldsymbol{C}\boldsymbol{B}^{-1}\boldsymbol{B} = \mathbf{0}. && \text{(B.19)} \end{aligned}$$

Since $\boldsymbol{z}$, and $\boldsymbol{y}$ are Gaussian variables, their decorrelation also means that they are independent. We can now compute the conditional expectation

$$\begin{aligned} \mathbb{E}(\mathbf{x}|\mathbf{y}) &= \mathbb{E}(\mathbf{z} + \boldsymbol{C}\boldsymbol{B}^{-1}\mathbf{y}|\mathbf{y}) && \text{(B.20)} \\ &= \mathbb{E}(\mathbf{z}|\mathbf{y}) + \boldsymbol{C}\boldsymbol{B}^{-1}\mathbb{E}(\mathbf{y}|\mathbf{y}) && \text{(B.21)} \\ &= \mathbb{E}(\mathbf{z}) + \boldsymbol{C}\boldsymbol{B}^{-1}\mathbf{y} && \text{(B.22)} \\ &= \boldsymbol{a} - \boldsymbol{C}\boldsymbol{B}^{-1}\boldsymbol{b} + \boldsymbol{C}\boldsymbol{B}^{-1}\mathbf{y} && \text{(B.23)} \\ &= \boldsymbol{a} + \boldsymbol{C}\boldsymbol{B}^{-1}(\mathbf{y} - \boldsymbol{b}). && \text{(B.24)} \end{aligned}$$

We can further compute the conditional covariance matrix :

$$\begin{aligned} \mathbb{V}\mathrm{ar}(\mathbf{x}|\mathbf{y}) &= \mathbb{V}\mathrm{ar}(\mathbf{z} + \boldsymbol{C}\boldsymbol{B}^{-1}\mathbf{y}|\mathbf{y}) && \text{(B.25)} \\ &= \mathbb{V}\mathrm{ar}(\mathbf{z} + |\mathbf{y}) + \mathbb{V}\mathrm{ar}(\boldsymbol{C}\boldsymbol{B}^{-1}\mathbf{y}|\mathbf{y}) + \boldsymbol{C}\boldsymbol{B}^{-1}\mathbb{C}\mathrm{ov}(\mathbf{z}, \mathbf{y}) + \mathbb{C}\mathrm{ov}(\mathbf{z}, \mathbf{y})\boldsymbol{C}\boldsymbol{B}^{-1\top} && \text{(B.26)} \\ &= \mathbb{V}\mathrm{ar}(\mathbf{z}|\mathbf{y}) && \text{(B.27)} \\ &= \mathbb{V}\mathrm{ar}(\mathbf{z}) && \text{(B.28)} \\ &= \mathbb{V}\mathrm{ar}(\mathbf{x} - \boldsymbol{C}\boldsymbol{B}^{-1}\mathbf{y}) && \text{(B.29)} \\ &= \mathbb{V}\mathrm{ar}(\mathbf{x}) + \boldsymbol{C}\boldsymbol{B}^{-1}\mathbb{V}\mathrm{ar}(\mathbf{y})\boldsymbol{B}^{-1}\boldsymbol{C}^\top + \boldsymbol{C}\boldsymbol{B}^{-1}\mathbb{C}\mathrm{ov}(\mathbf{x}, \mathbf{y}) + \mathbb{C}\mathrm{ov}(\mathbf{y}, \mathbf{x})\boldsymbol{B}^{-1}\boldsymbol{C}^\top && \text{(B.30)} \\ &= \boldsymbol{A} - \boldsymbol{C}\boldsymbol{B}^{-1\top}\boldsymbol{C}^\top && \text{(B.31)} \end{aligned}$$

This concludes the proof for $\mathbf{x}|\mathbf{y}$. A similar calculation gives the results for $\mathbf{y}|\mathbf{x}$. □

Theorem B.0.3. *Let* $\mathbf{f} \in \mathbb{R}^N$ *and* $\mathbf{m}^\alpha \in \mathbb{R}^M$ *be two Gaussian random variables such that*

$$\mathbf{f} \sim \mathcal{N}(\boldsymbol{\mu}, \tilde{\boldsymbol{\Sigma}}), \tag{B.32}$$
$$\mathbf{m}^\alpha | \boldsymbol{f} \sim \mathcal{N}(\boldsymbol{A}\boldsymbol{f}, \boldsymbol{\Sigma}_\alpha). \tag{B.33}$$

Where $\boldsymbol{A} \in \mathbb{R}^{M \times N}$. *The posterior probability distribution of* $\mathbf{x}|\mathbf{y}$ *is given as*

$$\mathbf{f} | \boldsymbol{m}^\alpha \sim \mathcal{N}(\boldsymbol{f}^*, \boldsymbol{P}^*), \tag{B.34}$$

where,

$$\boldsymbol{f}^* = \boldsymbol{\mu} + \tilde{\boldsymbol{\Sigma}}\boldsymbol{A}^\top (\boldsymbol{A}^\top \tilde{\boldsymbol{\Sigma}}\boldsymbol{A} + \boldsymbol{\Sigma}_\alpha)^{-1}(\boldsymbol{m}^\alpha - \boldsymbol{A}\boldsymbol{\mu}), \tag{B.35}$$

and

$$\boldsymbol{P}^* = \tilde{\boldsymbol{\Sigma}} - \tilde{\boldsymbol{\Sigma}}\boldsymbol{A}^\top (\boldsymbol{A}\tilde{\boldsymbol{\Sigma}}\boldsymbol{A} + \boldsymbol{\Sigma}_\alpha)^{-1}\boldsymbol{A}\tilde{\boldsymbol{\Sigma}}. \tag{B.36}$$

Proof. The proof is a direct application of the Lemmas B.0.1 and B.0.2. □

Theorem B.0.4. *Let* $\boldsymbol{f}^* \in \mathbb{R}^N$ *be the solution of*

$$\boldsymbol{f}^* = \underset{\boldsymbol{f}}{\operatorname{argmin}} \|\boldsymbol{A}\boldsymbol{f} - \boldsymbol{m}^\alpha\|^2_{\boldsymbol{\Sigma}_\alpha^{-1}} + \|\boldsymbol{f} - \boldsymbol{\mu}\|^2_{\tilde{\boldsymbol{\Sigma}}^{-1}}. \tag{B.37}$$

Then, $\boldsymbol{f}^*$ *has an analytical expression :*

$$\boldsymbol{f}^* = \boldsymbol{\mu} + \tilde{\boldsymbol{\Sigma}}\boldsymbol{A}^\top (\boldsymbol{A}^\top \tilde{\boldsymbol{\Sigma}}\boldsymbol{A} + \boldsymbol{\Sigma}_\alpha)^{-1}(\boldsymbol{m}^\alpha - \boldsymbol{A}\boldsymbol{\mu}). \tag{B.38}$$

Proof. Let us define $J(\boldsymbol{f})$ for any $\boldsymbol{f} \in \mathbb{R}^N$ by :

$$J(\boldsymbol{f}) = \|\boldsymbol{A}\boldsymbol{f} - \boldsymbol{m}^\alpha\|^2_{\boldsymbol{\Sigma}_\alpha^{-1}} + \|\boldsymbol{f} - \boldsymbol{\mu}\|^2_{\tilde{\boldsymbol{\Sigma}}^{-1}} \tag{B.39}$$
$$= (\boldsymbol{A}\boldsymbol{f} - \boldsymbol{m}^\alpha)^\top \boldsymbol{\Sigma}_\alpha^{-1}(\boldsymbol{A}\boldsymbol{m} - \boldsymbol{m}^\alpha) + (\boldsymbol{f} - \boldsymbol{\mu})^\top \tilde{\boldsymbol{\Sigma}}^{-1}(\boldsymbol{f} - \boldsymbol{\mu}) \tag{B.40}$$

By developing that expression, using the fact that $\boldsymbol{\Sigma}_\alpha^{-1}$ and $\tilde{\boldsymbol{\Sigma}}^{-1}$ are symmetric, and the symmetry of the inner product, we get :

$$J(\boldsymbol{f}) = \boldsymbol{f}^\top (\boldsymbol{A}^\top \boldsymbol{\Sigma}_\alpha^{-1}\boldsymbol{A} + \tilde{\boldsymbol{\Sigma}}^{-1})\boldsymbol{f} + \boldsymbol{m}^{\alpha\top} \boldsymbol{\Sigma}_\alpha^{-1}\boldsymbol{m}^\alpha + \boldsymbol{\mu}^\top \tilde{\boldsymbol{\Sigma}}^{-1}\boldsymbol{\mu}$$
$$-2(\boldsymbol{m}^{\alpha\top} \boldsymbol{\Sigma}_\alpha^{-1}\boldsymbol{A} + \boldsymbol{\mu}^\top \tilde{\boldsymbol{\Sigma}}^{-1})\boldsymbol{f} \tag{B.41}$$

We use this expression to compute $J(\boldsymbol{f} + h\boldsymbol{\epsilon}) - J(\boldsymbol{f})$, for any $h > 0$, and any $\boldsymbol{\epsilon} \in \mathbb{R}^N$:

$$J(\boldsymbol{f} + h\boldsymbol{\epsilon}) - J(\boldsymbol{f}) = 2h\boldsymbol{f}^\top (\boldsymbol{A}^\top \boldsymbol{\Sigma}_\alpha^{-1}\boldsymbol{A} + \tilde{\boldsymbol{\Sigma}}^{-1})\boldsymbol{\epsilon}$$
$$- 2h(\boldsymbol{m}^{\alpha\top} \boldsymbol{\Sigma}_\alpha^{-1}\boldsymbol{A} + \boldsymbol{\mu}^\top \tilde{\boldsymbol{\Sigma}}^{-1})\boldsymbol{\epsilon} + h^2 \boldsymbol{\epsilon}^\top (\boldsymbol{A}^\top \boldsymbol{\Sigma}_\alpha^{-1}\boldsymbol{A} + \tilde{\boldsymbol{\Sigma}}^{-1})\boldsymbol{\epsilon} \tag{B.42}$$

We can compute the gradient by taking the limit $h \to 0$:

$$\lim_{h \to 0} \frac{J(\boldsymbol{f} + h\boldsymbol{\epsilon}) - J(\boldsymbol{f})}{h} = 2\langle (\boldsymbol{A}^\top \boldsymbol{\Sigma}_\alpha^{-1}\boldsymbol{A} + \tilde{\boldsymbol{\Sigma}}^{-1})\boldsymbol{f} - \boldsymbol{A}^\top \boldsymbol{\Sigma}_\alpha^{-1}\boldsymbol{m}^\alpha - \tilde{\boldsymbol{\Sigma}}^{-1}\boldsymbol{\mu}, \boldsymbol{\epsilon} \rangle \tag{B.43}$$
$$= \langle \boldsymbol{\nabla} J(\boldsymbol{f}), \boldsymbol{\epsilon} \rangle \tag{B.44}$$

We can therefore conclude that $\boldsymbol{\nabla} J(\boldsymbol{f}) = (\boldsymbol{A}^\top \boldsymbol{\Sigma}_\alpha^{-1}\boldsymbol{A} + \tilde{\boldsymbol{\Sigma}}^{-1})\boldsymbol{f} - \boldsymbol{A}^\top \boldsymbol{\Sigma}_\alpha^{-1}\boldsymbol{m}^\alpha - \tilde{\boldsymbol{\Sigma}}^{-1}\boldsymbol{\mu}$. By searching for $\boldsymbol{f}^* \in \mathbb{R}^N$ such that $\boldsymbol{\nabla} J(\boldsymbol{f}) = \mathbf{0}$, we get :

$$\boldsymbol{f}^* = (\boldsymbol{A}^\top \boldsymbol{\Sigma}_\alpha^{-1}\boldsymbol{A} + \tilde{\boldsymbol{\Sigma}}^{-1})^{-1}(\boldsymbol{A}^\top \boldsymbol{\Sigma}_\alpha^{-1}\boldsymbol{m}^\alpha + \tilde{\boldsymbol{\Sigma}}^{-1}\boldsymbol{\mu}) \tag{B.45}$$

By developing this expression, we get :

$$\boldsymbol{f}^* = \boldsymbol{\mu} + (\boldsymbol{A}^\top \boldsymbol{\Sigma}_\alpha^{-1} \boldsymbol{A} + \tilde{\boldsymbol{\Sigma}}^{-1})^{-1} (\boldsymbol{A}^\top \boldsymbol{\Sigma}_\alpha^{-1} (\boldsymbol{m}^\alpha - \boldsymbol{A}\boldsymbol{\mu})) \tag{B.46}$$

By using the matrix inversion Lemma B.0.5, we finally get

$$\boldsymbol{f}^* = \boldsymbol{\mu} + \tilde{\boldsymbol{\Sigma}} \boldsymbol{A}^\top (\boldsymbol{A} \tilde{\boldsymbol{\Sigma}} \boldsymbol{A}^\top + \boldsymbol{\Sigma}_\alpha)^{-1} (\boldsymbol{m}^\alpha - \boldsymbol{A}\boldsymbol{\mu}). \tag{B.47}$$

□

Lemma B.0.5. *Let $\boldsymbol{A}, \boldsymbol{\Sigma}_\alpha, \tilde{\boldsymbol{\Sigma}}$ be defined as in the above theorem. Then,*

$$\tilde{\boldsymbol{\Sigma}} \boldsymbol{A}^\top (\boldsymbol{\Sigma}_\alpha + \boldsymbol{A} \tilde{\boldsymbol{\Sigma}} \boldsymbol{A}^\top)^{-1} = (\boldsymbol{A}^\top \boldsymbol{\Sigma}_\alpha^{-1} \boldsymbol{A} + \tilde{\boldsymbol{\Sigma}}^{-1})^{-1} \boldsymbol{A}^\top \boldsymbol{\Sigma}_\alpha^{-1} \tag{B.48}$$

Proof. Let us consider :

$$\begin{aligned}
\boldsymbol{K} &= \tilde{\boldsymbol{\Sigma}} \boldsymbol{A}^\top (\boldsymbol{\Sigma}_\alpha + \boldsymbol{A} \tilde{\boldsymbol{\Sigma}} \boldsymbol{A}^\top)^{-1} && \text{(B.49)} \\
\boldsymbol{T} &= (\boldsymbol{\Sigma}_\alpha + \boldsymbol{A} \tilde{\boldsymbol{\Sigma}} \boldsymbol{A}^\top) && \text{(B.50)} \\
\boldsymbol{P} &= (\boldsymbol{I} - \boldsymbol{K}\boldsymbol{A}) \tilde{\boldsymbol{\Sigma}} (\boldsymbol{I} - \boldsymbol{K}\boldsymbol{A})^\top + \boldsymbol{K} \boldsymbol{\Sigma}_\alpha \boldsymbol{K}^\top && \text{(B.51)}
\end{aligned}$$

Let us start by developing $\boldsymbol{P}$:

$$\begin{aligned}
\boldsymbol{P} &= (\boldsymbol{I} - \tilde{\boldsymbol{\Sigma}} \boldsymbol{A}^\top \boldsymbol{T}^{-1} \boldsymbol{A}) \tilde{\boldsymbol{\Sigma}} (\boldsymbol{I} - \tilde{\boldsymbol{\Sigma}} \boldsymbol{A}^\top \boldsymbol{T}^{-1} \boldsymbol{A})^\top + \boldsymbol{K} \boldsymbol{\Sigma}_\alpha \boldsymbol{K}^\top && \text{(B.52)} \\
&= \tilde{\boldsymbol{\Sigma}} - \tilde{\boldsymbol{\Sigma}} \boldsymbol{A}^\top \boldsymbol{T}^{-1} \boldsymbol{A} \tilde{\boldsymbol{\Sigma}} - \tilde{\boldsymbol{\Sigma}} \boldsymbol{A}^\top \boldsymbol{T}^{-1} \boldsymbol{A} \tilde{\boldsymbol{\Sigma}} \\
&\quad + \tilde{\boldsymbol{\Sigma}} \boldsymbol{A}^\top \boldsymbol{T}^{-1} \boldsymbol{A} \tilde{\boldsymbol{\Sigma}} \boldsymbol{A}^\top \boldsymbol{T}^{-1} \boldsymbol{A} \tilde{\boldsymbol{\Sigma}} + \tilde{\boldsymbol{\Sigma}} \boldsymbol{A}^\top \boldsymbol{T}^{-1} \boldsymbol{\Sigma}_\alpha \boldsymbol{T}^{-1} \boldsymbol{A} \tilde{\boldsymbol{\Sigma}} && \text{(B.53)} \\
&= \tilde{\boldsymbol{\Sigma}} - 2 \tilde{\boldsymbol{\Sigma}} \boldsymbol{A}^\top \boldsymbol{T}^{-1} \boldsymbol{A} \tilde{\boldsymbol{\Sigma}} + \tilde{\boldsymbol{\Sigma}} \boldsymbol{A}^\top \boldsymbol{T}^{-1} \boldsymbol{T} \boldsymbol{T}^{-1} \boldsymbol{A} \tilde{\boldsymbol{\Sigma}} && \text{(B.54)} \\
&= \tilde{\boldsymbol{\Sigma}} - \tilde{\boldsymbol{\Sigma}} \boldsymbol{A}^\top \boldsymbol{T}^{-1} \boldsymbol{A} \tilde{\boldsymbol{\Sigma}} && \text{(B.55)} \\
\boldsymbol{P} &= \tilde{\boldsymbol{\Sigma}} - \boldsymbol{K} \boldsymbol{A} \tilde{\boldsymbol{\Sigma}} && \text{(B.56)}
\end{aligned}$$

Let's invert $\boldsymbol{P}$ using the Woodburry Matrix identity.

$$\begin{aligned}
\boldsymbol{P}^{-1} &= (\tilde{\boldsymbol{\Sigma}} - \tilde{\boldsymbol{\Sigma}} \boldsymbol{A}^\top (\boldsymbol{\Sigma}_\alpha + \boldsymbol{A} \tilde{\boldsymbol{\Sigma}} \boldsymbol{A}^\top)^{-1} \boldsymbol{A} \tilde{\boldsymbol{\Sigma}})^{-1} && \text{(B.57)} \\
&= \tilde{\boldsymbol{\Sigma}}^{-1} + \tilde{\boldsymbol{\Sigma}}^{-1} \tilde{\boldsymbol{\Sigma}} \boldsymbol{A}^\top [(\boldsymbol{A} \tilde{\boldsymbol{\Sigma}} \boldsymbol{A}^\top + \boldsymbol{\Sigma}_\alpha) - \boldsymbol{A} \tilde{\boldsymbol{\Sigma}} \tilde{\boldsymbol{\Sigma}}^{-1} (\tilde{\boldsymbol{\Sigma}} \boldsymbol{A}^\top)] \boldsymbol{A} \tilde{\boldsymbol{\Sigma}} \tilde{\boldsymbol{\Sigma}}^{-1} && \text{(B.58)} \\
\boldsymbol{P}^{-1} &= \tilde{\boldsymbol{\Sigma}}^{-1} + \boldsymbol{A}^\top \boldsymbol{\Sigma}_\alpha^{-1} \boldsymbol{A} && \text{(B.59)}
\end{aligned}$$

By further inverting this expression, we get

$$\boldsymbol{P} = [\tilde{\boldsymbol{\Sigma}}^{-1} + \boldsymbol{A}^\top \boldsymbol{\Sigma}_\alpha^{-1} \boldsymbol{A}]^{-1} \tag{B.60}$$

At this point, we only need to prove that $\boldsymbol{K} = \boldsymbol{P} \boldsymbol{A}^\top \boldsymbol{\Sigma}_\alpha$ to prove the lemma. Let us premultiply $\boldsymbol{K}$ by $\boldsymbol{P}\boldsymbol{P}^{-1}$:

$$\begin{aligned}
\boldsymbol{K} &= \boldsymbol{P}\boldsymbol{P}^{-1} \tilde{\boldsymbol{\Sigma}} \boldsymbol{A}^\top (\boldsymbol{\Sigma}_\alpha + \boldsymbol{A} \tilde{\boldsymbol{\Sigma}} \boldsymbol{A}^\top)^{-1}, && \text{(B.61)} \\
&= \boldsymbol{P} [\tilde{\boldsymbol{\Sigma}}^{-1} + \boldsymbol{A}^\top \boldsymbol{\Sigma}_\alpha^{-1} \boldsymbol{A}] \tilde{\boldsymbol{\Sigma}} \boldsymbol{A}^\top (\boldsymbol{\Sigma}_\alpha + \boldsymbol{A} \tilde{\boldsymbol{\Sigma}} \boldsymbol{A}^\top)^{-1}. && \text{(B.62)}
\end{aligned}$$

By pushing the term $\tilde{\boldsymbol{\Sigma}} \boldsymbol{A}^\top$ in the first parenthesis, we get

$$\boldsymbol{K} = \boldsymbol{P} [\boldsymbol{A}^\top + \boldsymbol{A}^\top \boldsymbol{\Sigma}_\alpha^{-1} \boldsymbol{A} \tilde{\boldsymbol{\Sigma}} \boldsymbol{A}^\top] \tilde{\boldsymbol{\Sigma}} \boldsymbol{A}^\top (\boldsymbol{\Sigma}_\alpha + \boldsymbol{A} \tilde{\boldsymbol{\Sigma}} \boldsymbol{A}^\top)^{-1}. \tag{B.63}$$

By further factoring the first parenthesis by $\boldsymbol{A}^\top$, and premultiplying by $\boldsymbol{\Sigma}_\alpha^{-1} \boldsymbol{\Sigma}_\alpha$ at the left of the first parenthesis, we get :

$$\begin{aligned}
\boldsymbol{K} &= \boldsymbol{P} \boldsymbol{A}^\top \boldsymbol{\Sigma}_\alpha^{-1} [\boldsymbol{\Sigma}_\alpha + \boldsymbol{A} \tilde{\boldsymbol{\Sigma}} \boldsymbol{A}^\top] \tilde{\boldsymbol{\Sigma}} \boldsymbol{A}^\top (\boldsymbol{\Sigma}_\alpha + \boldsymbol{A} \tilde{\boldsymbol{\Sigma}} \boldsymbol{A}^\top)^{-1}, && \text{(B.64)} \\
&= \boldsymbol{P} \boldsymbol{A}^\top \boldsymbol{\Sigma}_\alpha^{-1}. && \text{(B.65)}
\end{aligned}$$

Which concludes the proof. □

Appendix C

Conditional Mean theorem

Theorem C.0.1. *Let* $\mathbf{f}$ *be a random variable with values in* $\mathbb{R}^N$, $\mathbf{m}^\alpha$ *a random variable with values in* $\mathbb{R}^M$, *and* $\mathcal{F}(\mathbf{f})$ *a random variable with values in* $\mathbb{R}^N$, *if*

$$\mathcal{G}^* = \underset{\mathcal{G}:\mathbb{R}^M \mapsto \mathbb{R}^N}{\operatorname{argmin}} \mathbb{E}\left[\|\mathcal{G}(\boldsymbol{m}^\alpha) - \mathcal{F}(\boldsymbol{f})\|_2^2\right], \tag{C.1}$$

then $\mathcal{G}^*(\boldsymbol{m}^\alpha) = \mathbb{E}(\mathcal{F}(\mathbf{f})|\mathbf{m}^\alpha = \boldsymbol{m}^\alpha)$.

Proof. For any measurable function $\mathcal{G} : \mathbb{R}^M \mapsto \mathbb{R}^{\mathbb{N}}$, due to properties from the conditional expectation,

$$\mathbb{E}\left[\|\mathcal{G}(\boldsymbol{m}^\alpha) - \mathcal{F}(\boldsymbol{f})\|_2^2\right] = \mathbb{E}\left[\mathbb{E}\left[\|\mathcal{G}(\boldsymbol{m}^\alpha) - \mathcal{F}(\boldsymbol{f})\|^2\right]|\mathbf{m}^\alpha\right]. \tag{C.2}$$

Since $\mathbb{R}^N$ is a Hilbert space, and the ℓ_2 norm is associated with an inner product, we can expand the squared norm with the parallelogram law :

$$\begin{aligned}\|\mathcal{G}(\boldsymbol{m}^\alpha) - \mathcal{F}(\boldsymbol{f})\|_2^2 &= \|\mathcal{G}(\boldsymbol{m}^\alpha) + \mathbb{E}(\mathcal{F}(\boldsymbol{f})|\mathbf{m}^\alpha) - \mathbb{E}(\mathcal{F}(\boldsymbol{f})|\mathbf{m}^\alpha) - \mathcal{F}(\boldsymbol{f})\|_2^2 \\ &= \|\mathcal{G}(\boldsymbol{m}^\alpha) - \mathbb{E}(\mathcal{F}(\boldsymbol{f})|\mathbf{m}^\alpha)\|_2^2 + \|\,\mathbb{E}(\mathcal{F}(\boldsymbol{f})|\mathbf{m}^\alpha) - \mathcal{F}(\boldsymbol{f})\|_2^2 \\ &\quad + 2\langle \mathcal{G}(\boldsymbol{m}^\alpha) - \mathbb{E}(\mathcal{F}(\boldsymbol{f})|\mathbf{m}^\alpha), \mathbb{E}(\mathcal{F}(\boldsymbol{f})|\mathbf{m}^\alpha) - \mathcal{F}(\boldsymbol{f})\rangle\end{aligned}$$

Also, due to the linear properties of the inner product,

$$\begin{aligned}&\mathbb{E}(\langle \mathcal{G}(\boldsymbol{m}^\alpha) - \mathbb{E}(\mathcal{F}(\boldsymbol{f})|\mathbf{m}^\alpha), \mathbb{E}(\mathcal{F}(\boldsymbol{f})|\mathbf{m}^\alpha) - \mathcal{F}(\boldsymbol{f})\rangle|\mathbf{m}^\alpha) \\ &= \langle \mathbb{E}(\mathcal{G}(\boldsymbol{m}^\alpha)|\mathbf{m}^\alpha) - \mathbb{E}(\mathbb{E}(\mathcal{F}(\boldsymbol{f})|\mathbf{m}^\alpha)|\mathbf{m}^\alpha), \mathbb{E}(\mathbb{E}(\mathcal{F}(\boldsymbol{f})|\mathbf{m}^\alpha)|\mathbf{m}^\alpha) - \mathbb{E}(\mathcal{F}(\boldsymbol{f})|\mathbf{m}^\alpha)\rangle\end{aligned}$$

Since $\mathcal{G}(\boldsymbol{m}^\alpha)$ is $\mathbf{m}^\alpha$-measurable, then $\mathbb{E}(\mathcal{G}(\boldsymbol{m}^\alpha)|\mathbf{m}^\alpha) = \mathcal{G}(\boldsymbol{m}^\alpha)$. Also due to the standard properties of the conditional expectation, we have $\mathbb{E}(\mathbb{E}(\mathcal{F}(\boldsymbol{f})|\mathbf{m}^\alpha)|\mathbf{m}^\alpha) = \mathbb{E}(\mathcal{F}(\boldsymbol{f})|\mathbf{m}^\alpha)$.

This leads to the following result

$$\begin{aligned}&\mathbb{E}(\langle \mathcal{G}(\boldsymbol{m}^\alpha) - \mathbb{E}(\mathcal{F}(\boldsymbol{f})|\mathbf{m}^\alpha), \mathbb{E}(\mathcal{F}(\boldsymbol{f})|\mathbf{m}^\alpha) - \mathbb{E}(\mathcal{F}(\boldsymbol{f})\rangle|\mathbf{m}^\alpha) \\ &= \langle \mathcal{G}(\boldsymbol{m}^\alpha) - \mathbb{E}(\mathcal{F}(\boldsymbol{f})|\mathbf{m}^\alpha), \mathbb{E}(\mathcal{F}(\boldsymbol{f})|\mathbf{m}^\alpha) - \mathbb{E}(\mathcal{F}(\boldsymbol{f})|\mathbf{m}^\alpha)\rangle \\ &= \langle \mathcal{G}(\boldsymbol{m}^\alpha) - \mathbb{E}(\mathcal{F}(\boldsymbol{f})|\mathbf{m}^\alpha), \mathbf{0}\rangle = 0\end{aligned}$$

Therefore,

$$\begin{aligned}&\underset{\mathcal{G}:\mathbb{R}^M \mapsto \mathbb{R}^N}{\operatorname{argmin}} \mathbb{E}\left[\|\mathcal{G}(\boldsymbol{m}^\alpha) - \mathcal{F}(\boldsymbol{f})\|_2^2\right] \\ &= \underset{\mathcal{G}:\mathbb{R}^M \mapsto \mathbb{R}^N}{\operatorname{argmin}} \mathbb{E}\left[\|\mathcal{G}(\boldsymbol{m}^\alpha) - \mathbb{E}(\mathcal{F}(\boldsymbol{f})|\mathbf{m}^\alpha)\|_2^2 + \|\,\mathbb{E}(\mathcal{F}(\boldsymbol{f})|\mathbf{m}^\alpha) - \mathcal{F}(\boldsymbol{f})\|_2^2\right].\end{aligned}$$

But since $\|\,\mathbb{E}(\mathcal{F}(\boldsymbol{f})|\mathbf{m}^\alpha) - \mathcal{F}(\boldsymbol{f})\|_2^2$ is independent of $\mathcal{G}$, we get

$$\underset{\mathcal{G}:\mathbb{R}^M \mapsto \mathbb{R}^N}{\operatorname{argmin}} \mathbb{E}\left[\|\mathcal{G}(\boldsymbol{m}^\alpha) - \mathcal{F}(\boldsymbol{f})\|_2^2\right] = \underset{\mathcal{G}:\mathbb{R}^M \mapsto \mathbb{R}^N}{\operatorname{argmin}} \mathbb{E}\left[\|\mathcal{G}(\boldsymbol{m}^\alpha) - \mathbb{E}(\mathcal{F}(\boldsymbol{f})|\mathbf{m}^\alpha)\|_2^2\right]. \tag{C.3}$$

Also the minimum of $\|\mathcal{G}(\boldsymbol{m}^\alpha) - \mathbb{E}(\mathcal{F}(\boldsymbol{f})|\mathbf{m}^\alpha)\|_2^2 = 0$. Said minimum is obtained if and only if $\mathcal{G}(\boldsymbol{m}^\alpha) = \mathbb{E}(\mathcal{F}(\boldsymbol{f})|\mathbf{m}^\alpha)$. □

Appendix D

Backpropagation Equations.

We aim to derive the 4 equations of the backpropagation :

$$\boldsymbol{\delta}^L = \nabla_{a^L} C \odot \gamma'(\boldsymbol{z}^L) \tag{D.1}$$

$$\boldsymbol{\delta}^l = (\boldsymbol{w}^{l+1\top} \boldsymbol{\delta}^{l+1}) \odot \gamma'(\boldsymbol{z}^l) \tag{D.2}$$

$$\frac{\partial \mathcal{C}}{\partial b^l_j} = \delta^l_j \tag{D.3}$$

$$\frac{\partial \mathcal{C}}{\partial w^l_{j,k}} = a^{l-1}_k \delta^l_j. \tag{D.4}$$

Let's start with Eq. D.1.

By definition,

$$\delta^L_j = \frac{\partial C}{\partial z^L_j}. \tag{D.5}$$

We further apply the chain rule to express δ^L_j with respect to the output activations :

$$\delta^L_j = \sum_k \frac{\partial C}{\partial a^L_k} \frac{\partial a^L_k}{\partial z^L_j}, \tag{D.6}$$

where the sum is effectuated over all k neurons in the output layer. Thanks to the definition of $\boldsymbol{z}^L$ as the vector such that $\boldsymbol{a}^L = \gamma(\boldsymbol{z}^L)$, a^L_k only depends on ∂z^L_j if $j = k$. This means that for $j \neq k$, $\partial z^L_j = 0$. This simplifies the above equation :

$$\delta^L_j = \frac{\partial C}{\partial a^L_j} \frac{\partial a^L_j}{\partial z^L_j}. \tag{D.7}$$

And since $\boldsymbol{a}^L_j = \gamma(\boldsymbol{z}^L_j)$, we get,

$$\delta^L_j = \frac{\partial C}{\partial a^L_j} \gamma'(z^L_j), \tag{D.8}$$

which is Eq. D.1 in component form.

Then we'll prove Eq. D.2. We will use the chain rule to express for any layer l, $\forall j, \delta^l_j$ with respect to $\forall k, \delta^l_k$:

$$\delta^l_j = \frac{\partial C}{\partial z^l_j}, \tag{D.9}$$

$$= \sum_k \frac{\partial C}{\partial z^{l+1}_k} \frac{\partial z^{l+1}_k}{\partial z^L_j}, \tag{D.10}$$

$$= \sum_k \delta^{l+1}_k \frac{\partial z^{l+1}_k}{\partial z^L_j}. \tag{D.11}$$

Since by definition,

$$z_k^{l+1} = \sum_j w_{kj}^{l+1} \gamma(z_j^l) + b_k^{l+1}, \tag{D.12}$$

by substituting in D.11, we finally get

$$\delta_j^l = \sum_k \delta_k^{l+1} w_{kj}^{l+1} \gamma'(z_j^l), \tag{D.13}$$

Which is Eq. D.2 in component form.

For D.3, we further use the chain rule :

$$\frac{\partial \mathcal{C}}{\partial b_j^l} = \frac{\partial \mathcal{C}}{\partial z_j^l} \frac{\partial z_j^l}{\partial b_j^l} \tag{D.14}$$

$$= \delta_j^l \frac{\partial z_j^l}{\partial b_j^l}. \tag{D.15}$$

Since $z_j^l = \sum_k w_{jk}^l a_k^{l-1} + b_j^l$, finally

$$\frac{\partial \mathcal{C}}{\partial b_j^l} = \delta_j^l. \tag{D.16}$$

We prove D.4 by using the chain rule once again :

$$\frac{\partial \mathcal{C}}{\partial w_{j,k}^l} = \frac{\partial \mathcal{C}}{\partial z_j^l} \frac{\partial z_j^l}{\partial w_{j,k}^l} \tag{D.17}$$

$$= \delta_j^l \frac{\partial z_j^l}{\partial w_{j,k}^l}. \tag{D.18}$$

Since $z_j^l = \sum_k w_{jk}^l a_k^{l-1} + b_j^l$, finally

$$\frac{\partial \mathcal{C}}{\partial w_{j,k}^l} = a_k^{l-1} \delta_j^l. \tag{D.19}$$

Appendix E

Resampling in particle filtering.

When applying the sequential importance filter to a set number of particles, over time some weights $w_t^{(i)}$ tend to get closer to 0. This is an issue, as we would be carrying samples that are very unlikely. Ideally we would like to have all samples of equal probability.

Usually resampling is done when the estimate for the effective number of particles,

$$\eta_{\text{eff}} = \frac{1}{\sum_{i=1}^{S}(w_t^i)^2} \tag{E.1}$$

is small. A rule of thumb is usually to do the resampling when $\eta_{\text{eff}} < S/10$.

The resampling algorithm can be described as follows Särkkä, 2013 The idea of

Algorithm 7 Resampling procedure

Init - $\{\boldsymbol{f}_t^{(i)}\}_{0\leq i\leq S}$, S samples of $\pi(\boldsymbol{f}_t|\boldsymbol{f}_{1:t-1}, \boldsymbol{m}_{1:t}^{\alpha})$, and the corresponding weights $\{w_t^{(i)}\}_{0\leq i\leq S}$.
1 - Each weight $w_t^{(i)}$ is interpreted as the probability of drawing $\boldsymbol{f}_t^{(i)}$ from the set $\{\boldsymbol{f}_t^{(i)}\}_{0\leq i\leq S}$.
2 - Draw N samples from the discrete distribution $\{\boldsymbol{f}_t^{(i)}\}_{0\leq i\leq S}$.
for $i \in \{1, \ldots, S\}$ **do**
  Draw $r(i)_t \sim \mathcal{U}([0,1])$
  Find $j_t^i \in \mathbb{N}$ such that :
  $\sum_{m=1}^{j_t^i-1} w_t^{(m)} < r(i)_t$ and $\sum_{m=1}^{j_t^i} w_t^{(m)} \geq r(i)_t$
  The new chosen sample is $\tilde{\boldsymbol{f}}_t^{(i)} = \boldsymbol{f}_t^{(j^i)}$.
end for
3 - Set all new weights $\hat{w}_t^{(i)} = 1/S$.

this algorithm is to remove the particles with low weights, and retain and duplicate the particles with high weights.

Appendix F

Résumé étendu en français

F.1 Introduction

L'imagerie mono-pixel permet de reconstruire des images bidimensionnelles à partir d'un détecteur ponctuel. Cette technologie a historiquement été utilisée dans les modalités d'imagerie où chaque détecteur est soit trop cher, soit trop encombrant pour permettre l'utilisation de grilles de capteurs. Cette caméra acquiert, de façon matérielle une version compressée de l'image de la scène : un modulateur spatial de lumière module l'image de la scène qui est focalisée ensuite dans un détecteur ponctuel un système de lentilles. En chargeant séquentiellement une série de motifs dans le modulateur spatial de lumière, nous acquérons donc une séquence de mesures à partir desquelles nous souhaitons récupérer l'image originale de la scène.

L'idée de la caméra à pixel unique a été attribuée au groupe (Duarte et al., 2008) de l'université de Rice. Leur configuration initiale était utilisée comme une mise en œuvre matérielle des principes de la acquisition comprimée (Donoho, 2006) qui récupérait l'image par le biais d'un algorithme de minimisation ℓ_1. Ce sous-échantillonnage à une fréquence inférieure à la fréquence de Nyquist conduit à des problèmes inverses sous-déterminés, mais a ouvert la porte à l'acquisition en temps réel d'images à un seul pixel.

Si l'idée de l'imagerie à pixel unique est peu attrayante pour les techniques d'imagerie conventionnelles qui reposent sur des grilles de capteurs (comme les caméras CCD ou CMOS), elle est particulièrement intéressante pour les modalités d'imagerie où un capteur unique est soit trop cher, soit trop encombrant pour être disposé sous forme de réseau. À ce titre, ils ont été utilisés pour l'imagerie infrarouge (Radwell et al., 2014), l'imagerie hyperspectrale (Magalhaes et al., 2012) et la microscopie (Radwell et al., 2014; Rodríguez et al., 2016), entre autres.

L'imagerie mono-pixel initialement basée sur l'acquisition comprimée, a permis l'acquisition matérielle en temps réel des mesures. Cependant, l'algorithme de minimisation de ℓ_1, qui prend longtemps, rendait difficile la reconstruction d'images en temps réel. L'avènement des algorithmes d'apprentissage profond pour les problèmes inverses de reconstruction d'image (Arridge et al., 2019) a notamment ouvert la voie à la reconstruction d'image en temps réel (Higham et al., 2018).

L'objectif de cette thèse est d'étudier les algorithmes de reconstruction d'images par apprentissage profond pour l'imagerie mono-pixel. L'une des principales limites de ces algorithmes est leur manque d'interprétabilité qui les rend plus difficiles à appliquer aux données expérimentales. De plus, il manque un cadre d'imagerie mono-pixel où la redondance spatio-temporelle des images est exploitée afin d'éviter d'acquérir les coefficients qui ne varient pas dans le temps. Un tel cadre permettrait de réduire davantage la quantité de mesures nécessaires à l'acquisition d'images pour l'acquisition vidéo en temps réel.

Pour répondre aux objectifs de cette thèse, nous avons développé des algorithmes basés sur l'apprentissage profond pour reconstruire des images à partir de données expérimentales issues d'une caméra mono-pixel. Plus précisément, nous avons proposé l'usage d'une fonction analytique du domaine de la mesure au domaine de l'image basée sur la régularisation de Tikhonov. Le résultat de cette fonction est ensuite corrigée par un réseau neuronal profond. L'applicabilité de cette méthode est démontrée pour des mesures en niveaux de gris et hyperspectrales. De plus, nous avons proposé une architecture d'apprentissage profond facile à interpréter basée sur l'algorithme de maximisation des espérances (Dempster, Laird, and Rubin, 1977). Enfin, nous explorons le cadre du filtrage de Kalman pour la reconstruction de vidéos à un seul pixel afin d'exploiter la redondance temporelle des vidéos naturelles. Nous avons également proposé un schéma de sous-échantillonnage adaptatif en fonction du temps, ainsi qu'une méthode de reconstruction de la vidéo par Kalman.

Cette thèse a été réalisée dans le cadre du projet ARMONI ANR (Ducros, 2017) qui vise à développer des outils d'imagerie hyperspectrale pour la chirurgie guidée par l'image. A ce titre, l'application visée de la caméra mono-pixel est l'imagerie hyperspectrale. Les algorithmes et les méthodologies ont été réalisés dans le laboratoire CREATIS. La plupart de nos méthodes ont été testées sur des données expérimentales issues du prototype de caméra hyperspectrale mono-pixel développé à CREATIS.

Les 4 premiers sections explorent l'état de l'art, et présentent les concepts clés utilisés. Les 4 dernieres sections expliquent nos contributions.

Dans la section F.2, le concept d'imagerie mono-pixel est expliqué avec tous ses défis. Nous expliquons également les choix fondamentaux faits au cours de cette thèse pour relever les défis susmentionnés.

Dans la section ??, nous expliquons les concepts fondamentaux des approches bayésiennes pour résoudre les problèmes inverses. Ces concepts sont au cœur de la reconstruction d'images mono-pixel.

Dans la section F.3, nous présentons l'apprentissage profond, la théorie qui le sous-tend, ainsi que ses aspects plus pratiques.

Dans la section F.5, nous présentons les concepts de base de l'estimation d'état bayésienne récursive qui peut être utilisée pour récupérer des signaux dans le temps à partir d'une série de mesures bruitées.

Dans la section F.6, nous nous concentrons sur l'utilisation d'une cartographie basée sur Tikhonov pour la reconstruction par apprentissage profond. Ce travail est basé sur un article publié dans *Optics Express*, et vise en particulier à définir un cadre pour appliquer les réseaux neuronaux profonds aux données provenant d'une configuration expérimentale de caméra à un seul pixel.

Dans la section F.7, nous développons davantage les concepts de la section F.6 en appliquant la cartographie basée sur Tikhonov pour les données hyperspectrales combinée à des convolutions de dimension supérieure. Ce travail est basé sur un article soumis à *Optics Express*, et est le résultat d'une collaboration conjointe avec V. Pronina de l'Institut des Sciences et Technologies de Skolkovo.

Dans la section F.8, nous proposons une architecture d'apprentissage profond basée sur la maximisation des attentes pour la reconstruction d'images à un seul pixel. Ce réseau neuronal offre un cadre hautement interprétable pour la reconstruction basée sur l'apprentissage profond. Ce réseau de neurones est spécifiquement conçu pour s'attaquer aux mesures corrompues par un bruit dépendant du signal approximativement normalement distribué. Ce travail est basé sur un article soumis à *IEEE Transactions on Computational Imaging*.

Dans la section F.9, nous explorons l'utilisation de l'estimation d'état récursive pour la reconstruction vidéo à un seul pixel. Sur cette base, nous proposons un schéma

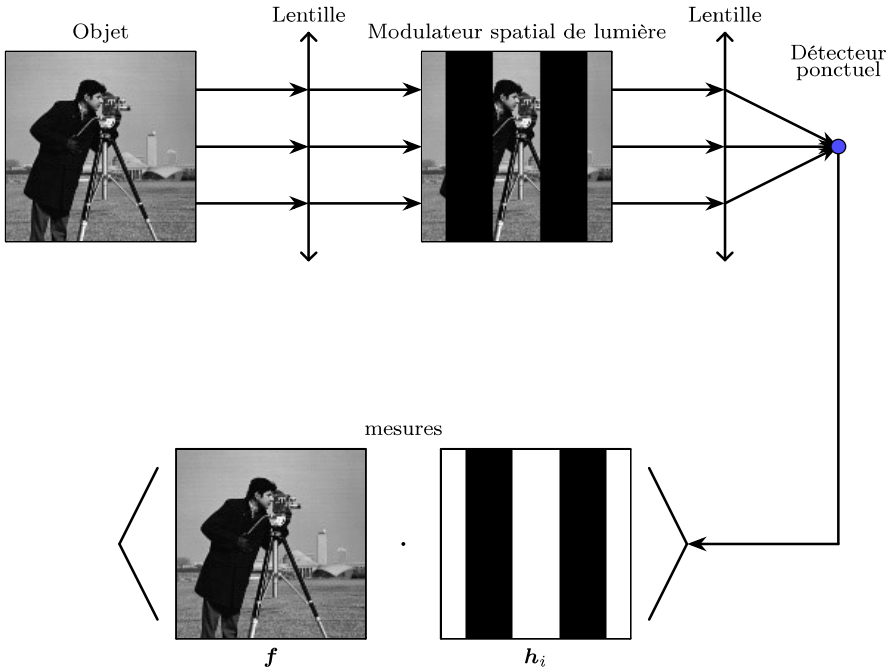


FIGURE F.1 – Principe de la caméra mono-pixel. Les mesures acquises sont le produit scalaire entre l'objet à acquérir et les motifs définis par l'utilisateur sur le modulateur spatial de lumière.

de sous-échantillonnage adaptatif au temps basé sur la variance. Sur la base de ce schéma de sous-échantillonnage, nous montrons que nous pouvons réduire le nombre de mesures d'acquisition pour la reconstruction vidéo. Nous présentons également certaines des perspectives présentées par le problème de la reconstruction vidéo à un seul pixel.

F.2 Imagerie mono-pixel, concepts et applications

La plupart des caméras traditionnelles acquièrent des images grâce à des réseaux de photo-détecteurs. En revanche, les caméras mono-pixel visent à récupérer des images à l'aide d'un détecteur à point unique. Dans ce chapitre, nous souhaitons couvrir les concepts de base de l'imagerie mono-pixel. Cela englobe les implémentations matérielles, la conception des masques, le modèle mathématique de la caméra mono-pixel, ainsi qu'une explication des choix effectués qui auront un impact sur le reste du manuscrit.

F.2.1 Concepts de l'imagerie mono-pixel

F.2.1.1 Description du système

Bien que de nombreuses implémentations matérielles de systèmes d'imagerie à un seul pixel aient été proposées au fil des ans (Gibson, Johnson, and Padgett, 2020; Edgar, Gibson, and Padgett, 2019), les principes restent les mêmes dans la plupart des cas. En fonction des applications, certaines technologies peuvent être mieux adaptées que d'autres. Cette section couvre la configuration conventionnelle d'une caméra mono-pixel(SPC de l'alglais single-pixel camera), décrite dans la Fig. F.1.

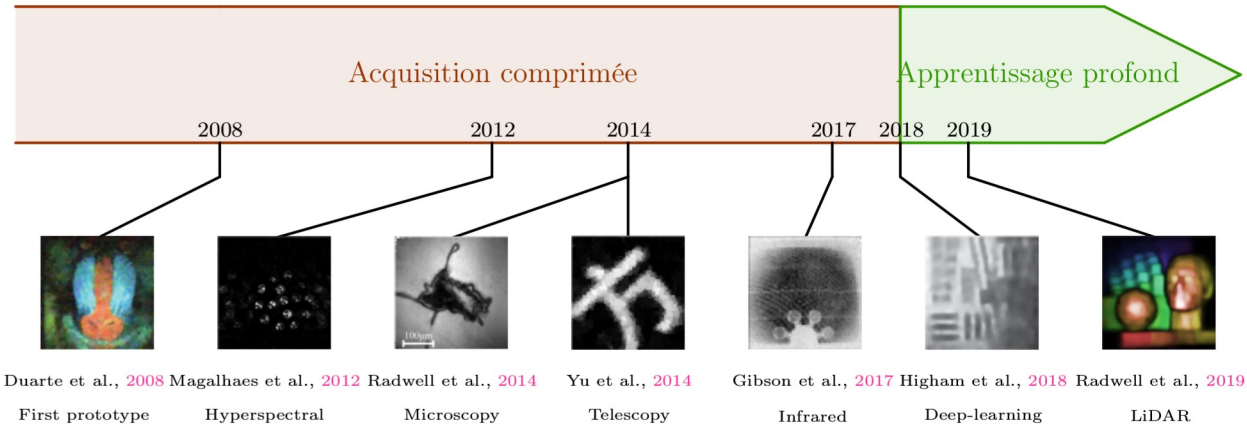


FIGURE F.2 – Chronologie des développements matériels et logiciels de l'imagerie mono-pixel. Les publications sont présentées par année et indiquent la méthode de reconstruction utilisée ainsi que l'application d'imagerie.

Les composants communs à toutes les configurations mono-pixeliques sont un ensemble de lentilles, un modulateur de lumière spatial (SLM de l'anglais spatial light modulator) et un photo-détecteur à point unique.

The image of the object $\boldsymbol{f} \in \mathbb{R}^N$ is formed in the spatial light SLM plane. The SLM modulates the image of the object by a user-defined patterns $\boldsymbol{h}_i \in \mathbb{R}^N$ (in Fig. F.1 for instance, the image is modulated by a Hadamard basis function). A lens then focuses the output of the SLM into the single point detector. The numerical data is then digitized via a numerical converter to acquire the inner product $\langle \boldsymbol{f}, \boldsymbol{h}_i \rangle$ of the object to be acquired $\boldsymbol{f}$, and the patter on the spatial light modulator $\boldsymbol{h}_i$. When acquiring M sequential measurements, the measurement vector $\boldsymbol{m} \in \mathbb{R}^M$ is said to acquire the product

L'image de l'objet $\boldsymbol{f} \in \mathbb{R}^N$ est formée dans le plan du SLM. Le SLM module l'image de l'objet par un motif défini par l'utilisateur $\boldsymbol{h}_i \in \mathbb{R}^N$ (dans la Fig. F.1 par exemple, l'image est modulée par une fonction de base Hadamard). Une lentille focalise ensuite la sortie du SLM sur le détecteur ponctuel. Les données numériques sont ensuite numérisées via un convertisseur numérique afin d'acquérir le produit scalaire $\langle \boldsymbol{f}, \boldsymbol{h}_i \rangle$ de l'objet à acquérir $\boldsymbol{f}$, et le motif sur le modulateur spatial de lumière $\boldsymbol{h}_i$. Lors de l'acquisition de M mesures séquentielles, le vecteur de mesure $\boldsymbol{m} \in \mathbb{R}^M$ acquiert

$$\boldsymbol{m} = \boldsymbol{H}\boldsymbol{f}, \tag{F.1}$$

où $\boldsymbol{H} = [\boldsymbol{h}_1, \ldots, \boldsymbol{h}_m]^\top \in \mathbb{R}^{M \times N}$.

La nature du photo-détecteur ponctuel dépend de l'application visée pour la caméra mono-pixel. La conception originale de Duarte utilisait un photo-détecteur à base de silicium. Depuis lors, de nombreuses variantes ont été appliquées pour répondre aux besoins de l'application. Les détecteurs infrarouges, les spectromètres et les détecteurs TeraHertz en sont quelques exemples.

F.2.1.2 Applications

Les avantages et la polyvalence de la SPC ont conduit à son application à de nombreux problèmes d'imagerie. Nous montrons quelques exemples d'applications à un seul pixel dans la Fig. F.2.

Imagerie infrarouge Dans (Radwell et al., 2014), les auteurs ont conçu un microscope basé sur le SPC capable d'imager dans l'infrarouge et le spectre visible. Pour réaliser une telle imagerie, un détecteur infrarouge et un détecteur visible ont été utilisés pour recueillir la lumière provenant de l'état ON et OFF du DMD. Ce système a ensuite été utilisé pour imager les fuites de gaz méthane (Gibson et al., 2017), prouvant ainsi le potentiel de l'imagerie à un seul pixel en imagerie infrarouge.

Imagerie télescopique Dans le (Yu et al., 2014), un télescope basé sur le SPC a été utilisé pour observer un objet à 2 km de distance. Cette configuration offre un grand champ de vision sur de grandes distances et donne des résultats prometteurs pour l'imagerie de cibles distantes.

Imagerie LiDAR Les auteurs de (Radwell et al., 2019) ont combiné une caméra à un seul pixel avec un système de détection et de télémétrie par la lumière, ce qui leur a permis de récupérer des cartes 3D de la profondeur sur de longues distances, démontrant l'applicabilité potentielle des caméras à un seul pixel pour les voitures à conduite autonome et plusieurs autres industries robotiques.

Microscopie La caméra mono-pixel a été utilisée pour de nombreux types d'imagerie microscopique. Leur capacité à prendre des images dans des longueurs d'onde où les détecteurs à base de silicium sont aveugles les a rendues très populaires dans la microscopie non visible (Radwell et al., 2014). Dans la (Wu et al., 2021), un séparateur de faisceau a été utilisé pour l'imagerie holographique microscopique de tissus biologiques. Dans (Rodríguez et al., 2016), un microscope inversé bimode permettant d'exploiter les informations de réflexion et de transmission a été proposé. Il a ensuite été utilisé pour la microscopie de Fourier (Peng et al., 2021), où le spectre de Fourier était capturé dans le plan focal arrière de l'objectif.

Imagerie hyper-spectrale Dans (Magalhaes et al., 2012), un système d'imagerie hyperspectrale a été obtenu en remplaçant le détecteur ponctuel par un analyseur spectral avec une résolution spectrale de 10 pm. Cela a donné naissance à l'imagerie hyperspectrale mono-pixel qui est un domaine de recherche florissant en raison de son faible prix et de sa capacité à produire des images bidimensionnelles avec des résolutions spectrales que les autres imageurs hyperspectraux ne peuvent atteindre. Ceci est particulièrement important pour les applications biomédicales, où les imageurs à haute résolution spectrale peuvent être utilisés pour analyser les signatures spectrales de certaines molécules.

F.2.2 Acquisition comprimée

The iLes idées à l'origine de l'imagerie à un seul pixel ont été inspirées par la théorie de la acquisition comprimée (Donoho, 2006). Cela signifie que la caméra monopixel a été conçue pour acquérir une version sous-échantillonnée de l'image soumise à un certain bruit

$$\boldsymbol{m} = \boldsymbol{H}\boldsymbol{f} + \boldsymbol{\epsilon}, \tag{F.2}$$

où $\boldsymbol{H} \in \mathbb{R}^{M \times N}$ est la matrice qui contient les M motifs choisis et $\boldsymbol{\epsilon} \in \mathbb{R}^{M}$ est un bruit additif aléatoire. Cette fois, cependant, le but est de reconstruire l'image avec $M \ll N$ mesures. Cela permet d'obtenir des taux d'acquisition plus rapides et de réduire les besoins en mémoire Le concept de pixel unique a été introduit pour la première fois

par (Takhar et al., 2006), et a ensuite été perfectionné par (Duarte et al., 2008) qui a fourni une implémentation matérielle de la acquisition comprimée.

La acquisition comprimée s'appuie sur un opérateur de transformation $\mathbf{\Lambda}\mathbb{R}^{N\times N}$ qui sparcifie $\boldsymbol{f}$.

$$\boldsymbol{f} = \mathbf{\Lambda}\boldsymbol{s}, \tag{F.3}$$

où $\boldsymbol{s} \in \mathbb{R}^N$ est L-sparse (a L composantes non nulles).

Selon la théorie de la acquisition comprimée, si la matrice $\boldsymbol{H}$ satisfait à la propriété dite d'isométrie restreinte (Donoho, 2006), alors l'image originale peut être parfaitement récupérée. L'image est alors récupérée en résolvant le problème inverse suivant :

$$\hat{\boldsymbol{s}} = \underset{\boldsymbol{s}\in\mathbb{R}^N}{\operatorname{argmin}} \|\boldsymbol{s}\|_1 \quad \text{de sorte que } \boldsymbol{H}\mathbf{\Lambda}\boldsymbol{s} = \boldsymbol{m}. \tag{F.4}$$

Les matrices qui vérifient la condition d'isométrie restreinte avec une probabilité élevée peuvent être tirées d'une distribution de Bernoulli,

$$(\boldsymbol{H})_{i,j} \sim \mathcal{B}(0, 1/2), \tag{F.5}$$

à condition que

$$M \geq cL\log(\frac{D}{L}), \tag{F.6}$$

pour certaines valeurs de c (typiquement dans l'intervalle $[1; 100]$ (Donoho, 2006; J. Candès, K. Romberg, and Tao, 2006; Takhar et al., 2006)). Ce choix est donc binaire et bien adapté pour être implémenté sur un SLM.

Cette approche permet une reconstruction parfaite du signal clairsemé en supposant des acquisitions sans bruit. Dans des conditions de bruit, le problème à résoudre est le suivant

$$\hat{\boldsymbol{s}} = \underset{\boldsymbol{s}\in\mathbb{R}^N}{\operatorname{argmin}} \|\boldsymbol{s}\|_1 \quad \text{subject to } \|\boldsymbol{H}\mathbf{\Lambda}\boldsymbol{s} - \boldsymbol{m}\|_2 \leq \|\boldsymbol{\epsilon}\|_2. \tag{F.7}$$

L'erreur de récupération peut alors être bornée ci-dessus par $C_0\|\boldsymbol{\epsilon}\|_2$, où C_0 est une constante.

Une difficulté de la acquisition comprimée est de trouver la transformée sparcifiante $\mathbf{\Lambda}$. En pratique, la plupart des approches utilisent des bases telles que la base d'ondelettes (Mallat, 1999), ou des dictionnaires appris. Aucune de ces approches ne parvient à obtenir une représentation clairsemée parfaite de toutes les images naturelles, ce qui conduirait souvent à des reconstructions avec des artefacts pour des taux de compression élevés. Une alternative populaire à l'algorithme de minimisation de ℓ_1 était la solution de variation totale (Li, 2010), qui est obtenue en résolvant

$$\hat{\boldsymbol{f}} = \underset{\boldsymbol{f}\in\mathbb{R}^N}{\operatorname{argmin}} \|\boldsymbol{f}\|_{\mathrm{TV}} \quad \text{such that } \boldsymbol{H}\boldsymbol{f} = \boldsymbol{m}, \tag{F.8}$$

ce qui correspond à choisir $\mathbf{\Lambda}$ comme opérateur de gradient discret.

F.2.3 Apprentissage profond

Dans (Higham et al., 2018), un cadre d'apprentissage profond pour l'imagerie à un seul pixel a été proposé. Ce cadre a ouvert la voie à la reconstruction vidéo monopixel

en temps réel. Les idées principales consistaient à considérer une structure d'auto-encodeur où la partie encodage était modélisée par une matrice de motifs binaires $\boldsymbol{H}$, ainsi qu'un reconstructeur neuronal profond $\mathcal{G}_\theta$ tel que

$$\hat{\boldsymbol{f}} \approx \mathcal{G}_\theta(\boldsymbol{H}\boldsymbol{f}). \tag{F.9}$$

Le réseau neuronal est entraîné à minimiser

$$\boldsymbol{\theta}, \boldsymbol{H} = \underset{\boldsymbol{\theta}^*, \boldsymbol{H}^*}{\operatorname{argmin}} \frac{1}{S} \sum_{j=1}^{S} \|\mathcal{G}_{\theta^*}(\boldsymbol{H}^* \boldsymbol{f}^{(i)}) - \boldsymbol{f}^{(i)}\|_2^2 + \beta(\|\boldsymbol{H}^* - \mathbf{1}\|_2^2 + \|\boldsymbol{H}^* + \mathbf{1}\|_2^2), \tag{F.10}$$

où $\{\boldsymbol{f}^{(i)}\}_{1 \leq i \leq S}$ sont des échantillons d'images provenant de l'ensemble de données d'images STL-10 (Coates, Ng, and Lee, 2011), et β est un paramètre de régularisation qui équilibre la perte pour garantir que les motifs sont binaires, tout en essayant de minimiser l'erreur quadratique moyenne. Ce cadre n'a pas seulement appris la matrice de mesure $\boldsymbol{H}$ comme une couche linéaire de perceptron multicouche, mais il a également offert un reconstructeur rapide pour les mesures à un seul pixel.

F.2.4 Objectifs de la thèse

Cette thèse a été financée par le projet ANR ARMONI (Ducros, 2017) qui vise à réduire la marge tumorale en exploitant l'image hyperspectrale acquise par une caméra mono-pixel. En analysant la signature spectrale, il a été montré que nous pouvons différencier les marges saines et tumorales (Alston, 2017; Valdés et al., 2015).
En particulier, cette thèse vise à développer des algorithmes d'imagerie mono-pixel en temps réel. L'acquisition rapide implique des mesures compressives avec des temps d'intégration courts. Cela conduit à un problème inverse sous-déterminé avec des niveaux de bruit élevés.
De plus, en considérant l'acquisition de vidéos, nous voulons exploiter la redondance temporelle des vidéos naturelles pour éviter de ré-acquérir des coefficients redondants au cours du temps.

F.2.5 Approche considérée

L'imagerie mono-pixel implique de choisir un ensemble de motifs, et de choisir comment sous-échantillonner la base de fonctions choisie. L'approche adoptée au cours de cette thèse a été d'utiliser une base Hadamard sous-échantillonnée par échantillonnage statistique. L'utilisation de la base Hadamard semble naturelle compte tenu de ses propriétés de réduction du bruit gaussien additif. Elles permettent d'acquérir plus de lumière en augmentant le rapport signal/bruit de nos mesures par rapport au bruit de comptage de photons de Poisson. Enfin, la nature binaire des motifs Hadamard permet une mise en œuvre matérielle rapide pour les applications d'imagerie à un seul pixel.

Leur plus grand inconvénient est leur capacité de compression, qui est inférieure à celle de la plupart des concurrents. Bien que ce point puisse être problématique, leur capacité à réduire le bruit signifie que nous pouvons réduire le temps d'intégration pour chaque motif. Cette propriété de réduction du bruit, associée à la fréquence de retournement rapide des micro-miroirs - en raison de la nature binaire de nos motifs -, nous permet d'acquérir davantage de mesures. Cela contrebalance quelque peu leurs capacités de compression comparativement plus faibles.

Bien que la reconstruction rapide ait été abordée principalement par l'utilisation d'algorithmes d'apprentissage profond, leur manque d'interprétabilité pose toujours problème. Nous avons donc cherché à combiner l'apprentissage profond avec la théorie classique des problèmes inverses. En particulier, nous cherchons à développer des méthodes qui sont interprétables et qui peuvent facilement être adaptées pour traiter spécifiquement un modèle de bruit propre à l'imagerie mono-pixel.

Conclusion

Cette section présente les principes de base de l'imagerie à un seul pixel. Il introduit les aspects fondamentaux derrière l'implémentation matérielle de la SPC, le choix des motifs, les stratégies de sous-échantillonnage. Dans cette thèse, nous avons choisi de sous-échantillonner l'image dans la base Hadamard avec un sous-échantillonnage statistique pour réduire les temps d'acquisition. Ceci conduit à un problème inverse sous-déterminé, corrompu par un bruit mixte Skellam-Gaussien. Nous explorerons des méthodes pour résoudre ce problème inverse dans les chapitres suivants. Les cas statiques et dynamiques seront explorés.

F.3 Reconstruction par apprentissage profond pour les problèmes inverses linéaires

Ce chapitre fournit une brève introduction à la philosophie et aux fondements mathématiques de l'inférence bayésienne pour les problèmes inverses en imagerie computationnelle. En particulier, nous montrons leur utilisation pour la reconstruction d'images à partir de problèmes inverses linéaires sous-déterminés.

F.3.1 Inversion bayesienne pour les problèmes inverses linéaires

F.3.1.1 Problème

Nous considérons des problèmes inverses linéaires

$$\boldsymbol{m} = \boldsymbol{A}\boldsymbol{f} + \boldsymbol{\epsilon}, \tag{F.11}$$

$$\boldsymbol{\epsilon} \sim p(\boldsymbol{\epsilon}), \tag{F.12}$$

où $\boldsymbol{m} \in \mathbb{R}^M$ sont les mesures, $\boldsymbol{f} \in \mathbb{R}^N$ sont les données originales que nous cherchons à récupérer à partir des mesures. $\boldsymbol{\epsilon} \in \mathbb{R}^M$ est le bruit suivant une fonction de densité de probabilité. Dans le cas de mesures indépendantes (comme c'est souvent le cas), par la règle de la chaîne, $p(\boldsymbol{\epsilon}) = \prod_{i=1}^{M} p(\epsilon_i)$. Nous considérons le cas particulier où $M < N$. Cette famille de problèmes est communément connue sous le nom de problèmes inverses linéaires sous-déterminés, et peut être utilisée pour modéliser une grande variété de problèmes d'imagerie tels que la super-résolution, et bien sûr, le problème de la caméra à un seul pixel considéré (en considérant simplement $\boldsymbol{A} = \boldsymbol{S}\boldsymbol{H}$ dans l'Eq. F.12, et $\boldsymbol{\epsilon}$ dans l'Eq. F.12 $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma}_\alpha)$).

F.3.1.2 L'approche bayésienne

L'inférence bayésienne vise à utiliser les distributions de probabilité et la théorie du calcul des probabilités comme outil pour résoudre les problèmes inverses. L'incertitude produite par la nature aléatoire du bruit et la perte d'information de l'opérateur d'avance $\boldsymbol{A}$ est modélisée par l'utilisation de fonctions de densité de probabilité.

L'approche bayésienne suppose que les mesures et l'image inconnue sont des échantillons d'une distribution de probabilité. Mathématiquement, $\boldsymbol{m}$ découle d'une variable aléatoire $\mathbf{m}$ avec une fonction de densité de probabilité $p(\boldsymbol{m})$. De même, elle suppose que $\boldsymbol{f}$ provient d'une variable aléatoire $\mathbf{f}$ avec une fonction de densité de probabilité - appelée distribution antérieure - $p(\boldsymbol{f})$.

En outre, les approches bayésiennes supposent également que nous avons accès à la distribution de l'observation $\mathbf{m}$ étant donné $\mathbf{f}$ à travers le modèle de mesure F.12. Ceci est également connu sous le nom de vraisemblance du modèle $p(\boldsymbol{m}|\boldsymbol{f}, \boldsymbol{A})$, qui est directement liée à la fonction de densité de probabilité de $\boldsymbol{\epsilon}$. Pour une $\boldsymbol{A}$ et une $\boldsymbol{f}$ données, $p_{\boldsymbol{\epsilon}}(\boldsymbol{\epsilon}) = p_{\boldsymbol{\epsilon}}(\boldsymbol{m} - \boldsymbol{A}\boldsymbol{f}|\boldsymbol{A}, \boldsymbol{f})$. Si $\boldsymbol{A}$ et $\boldsymbol{f}$ sont maintenus constants, alors $p_{\boldsymbol{\epsilon}}(\boldsymbol{m} - \boldsymbol{A}\boldsymbol{f}|\boldsymbol{A}, \boldsymbol{f})$ ne dépend que de $\boldsymbol{m}$. Cela nous permet de définir la fonction de densité de probabilité de $p(\boldsymbol{m}|\boldsymbol{A}, \boldsymbol{f}) = p_{\boldsymbol{\epsilon}}(\boldsymbol{m} - \boldsymbol{A}\boldsymbol{f}|\boldsymbol{A}, \boldsymbol{f})$. Notez que la dépendance vis-à-vis de $\boldsymbol{A}$ existe toujours, mais par souci de simplicité, elle est le plus souvent omise dans les notations, et nous ne notons que $p(\boldsymbol{m}|\boldsymbol{f})$ pour la vraisemblance.

La solution bayésienne du problème inverse est définie comme la fonction de densité de probabilité de $\mathbf{f}$ étant donné $\mathbf{m}$, elle est également appelée distribution postérieure $p(\boldsymbol{f}|\boldsymbol{m})$. Cette solution fournit non seulement des estimateurs de $\boldsymbol{f}$, mais nous permet également de déterminer la probabilité que ces estimateurs soient vrais pour une mesure donnée.

En supposant que $p(\boldsymbol{f})$ est connu, la distribution cible peut alors être calculée pour un échantillon donné $\boldsymbol{m}$ en utilisant la règle de Bayes

$$p(\boldsymbol{f}|\boldsymbol{m}) = \frac{p(\boldsymbol{f})p(\boldsymbol{m}|\boldsymbol{f})}{p(\boldsymbol{m})}. \tag{F.13}$$

Si, à première vue, il peut sembler que $p(\boldsymbol{m})$ soit également nécessaire, il convient de noter que pour une valeur donnée de $\boldsymbol{m}$, il s'agit d'une constante, et que l'on peut en déduire qu'il s'agit du paramètre de normalisation permettant de normaliser la fonction de densité de probabilité générée par $p(\boldsymbol{f})p(\boldsymbol{m}|\boldsymbol{f})$.

Malheureusement, cette solution est peu pratique pour les problèmes de grande dimension. En effet, $p(\boldsymbol{f}|\boldsymbol{m})$ est une application de $\mathbb{R}^N$ à $\mathbb{R}$, de sorte que même les problèmes discrets de petite taille nécessitent une grande quantité de mémoire et sont insolubles. Par exemple, le stockage de la fonction de densité de probabilité d'images de 64×64 en utilisant des variables à virgule flottante en simple précision nécessiterait $32^{64\times64} = 1,2 \cdot 10^{6165}$ octets de mémoire. Cela explique pourquoi, pour les problèmes inverses d'imagerie, les approches bayésiennes ont tendance à se limiter à des estimateurs ponctuels de la postérieure, que nous aborderons dans la section suivante.

F.3.2 Estimateurs ponctuels de la distribution postérieure

Comme nous l'avons vu dans la section précédente, l'obtention de la distribution postérieure complète étant impossible pour l'image de la plupart des problèmes inverses, nous avons tendance à nous limiter à des estimateurs ponctuels de $p(\boldsymbol{f}|\boldsymbol{m})$. Dans cette section, nous couvrons les estimateurs les plus populaires.

F.3.2.1 Estimation du maximum a posteriori

Le maximum a posteriori (MAP) vise à calculer la valeur qui maximise $p(\boldsymbol{f}|\boldsymbol{m})$. Cet estimateur donne le $\boldsymbol{f}$ le plus probable pour un $\boldsymbol{m}$ donné en minimisant

$$\boldsymbol{f}^* = \underset{\boldsymbol{f}}{\operatorname{argmin}}\{-\log(p(\boldsymbol{f}|\boldsymbol{m}))\}, \tag{F.14a}$$

$$\boldsymbol{f}^* = \underset{\boldsymbol{f}}{\operatorname{argmin}}\{-\log p(\boldsymbol{m}|\boldsymbol{f}) - \log p(\boldsymbol{f}) + \log p(\boldsymbol{m})\}, \tag{F.14b}$$

$$\boldsymbol{f}^* = \underset{\boldsymbol{f}}{\operatorname{argmin}}\{-\log p(\boldsymbol{m}|\boldsymbol{f}) - \log p(\boldsymbol{f})\}, \tag{F.14c}$$

où eq. F.14b suit par la règle de Bayes, et eq. F.14c suit puisque $p(\boldsymbol{m})$ ne dépend pas de $\boldsymbol{f}$. Cette minimisation garantit effectivement que nous maximisons $p(\boldsymbol{f}|\boldsymbol{m})$, mais elle simplifie également l'expression des distributions préalables exponentielles (Duda and Hart, 1973), des distributions de bruit gaussiennes et de Laplace, et elle convertit les multiplications en additions, ce qui tend à simplifier le problème de minimisation.

F.3.3 Espérance conditionnelle

L'espérance conditionnelle est un autre estimateur ponctuel populaire de la distribution postérieure. Elle donne la valeur attendue de $\boldsymbol{f}$ étant donné $\boldsymbol{m}$:

$$\boldsymbol{f}^* = \mathbb{E}(\mathbf{f}|\boldsymbol{m}) = \int_{\mathbb{R}^N} p(\boldsymbol{f}|\boldsymbol{m})\boldsymbol{f}\ d\boldsymbol{f}. \tag{F.15}$$

Le calcul d'un tel estimateur est difficile car il nécessite d'intégrer sur $\mathbb{R}^N$. Cependant, comme nous le montrerons dans la Sec. F.3.3.1, cet estimateur peut être calculé en générant des échantillons de la postérieure. Comme le montre la Sec. F.4.1, il peut également être approximé en utilisant des reconstructeurs à apprentissage profond.

F.3.3.1 Estimation par échantillonnage

L'estimation par échantillonnage vise à générer K échantillons $\{\boldsymbol{f}^i\}_{1\leq i\leq K}$ à partir d'une fonction de densité de probabilité $p(\boldsymbol{f}|\boldsymbol{m})$. Ensuite, $p(\boldsymbol{f}|\boldsymbol{m})$ est approximé analytiquement comme suit

$$p(\boldsymbol{f}|\boldsymbol{m}) \approx \sum_{i=1}^{K} w^{(i)}\delta(\boldsymbol{f} - \boldsymbol{f}^i), \tag{F.16}$$

où les poids $w^{(i)}$ doivent être calculés en fonction de la vraisemblance de $\boldsymbol{f}^i$.

Cette approximation permet en outre de calculer des estimateurs de la forme $\mathbb{E}(\boldsymbol{f}|\boldsymbol{m})$, par une approximation de Monte-Carlo

$$\mathbb{E}(\boldsymbol{f}|\boldsymbol{m}) \approx \frac{1}{K}\sum_{i=1}^{K} w^{(i)}\boldsymbol{f}^i. \tag{F.17}$$

Cette approximation convergera avec une vitesse $\mathcal{O}(K^{-1/2})$ grâce au théorème central limite, et est indépendante de la dimension du problème traité.

F.3.3.2 Distributions à priori exponentiels

Les distributions à priori exponentiels sont très répandus dans la communauté de la reconstruction d'images et sont donnés par la formule suivante

$$p(\boldsymbol{f}) = \frac{e^{-g(\boldsymbol{f})}}{\int e^{-g(\boldsymbol{h})} d\boldsymbol{h}} \propto e^{-g(\boldsymbol{f})}. \tag{F.18}$$

La fonction g est généralement choisie comme une fonction convexe, ce qui donne lieu à des estimations MAP qui bénéficient de la théorie de l'optimisation convexe. Parmi les choix les plus populaires de g figurent la norme ℓ_2 (Tikhonov, 1995), la norme ℓ_1 (J. Candès, K. Romberg, and Tao, 2006), et la variation totale (Rudin, Osher, and Fatemi, 1992). La norme ℓ_1 en particulier a été largement identifiée comme une norme qui recherche des solutions éparses. Elle a été principalement associée à la théorie de la détection comprimée. La variation totale a surtout été identifiée comme un antériorité qui favorise les fonctions linéaires par morceaux. Les antériorités exponentielles simplifient l'expression MAP

$$\boldsymbol{f}^* = \underset{\boldsymbol{f}}{\operatorname{argmin}} \|\boldsymbol{A}\boldsymbol{f} - \boldsymbol{m}\|^2_{\boldsymbol{\Sigma}_{\boldsymbol{\alpha}}^{-1}} + g(\boldsymbol{f}), \tag{F.19}$$

Conclusion

Ce chapitre présente l'approche et la philosophie bayésiennes pour la résolution de problèmes inverses linéaires pour des tâches de traitement d'images. Nous avons montré la nécessité de calculer des estimateurs ponctuels de la distribution postérieure, tels que le maximum a posteriori et les esperances conditionnelles de la postérieure. Les problèmes inverses linéaires présentent deux défis principaux. Premièrement, alors que les distributions à priori gaussiens admettent une solution analytique pour l'inversion bayésienne, les distributions à priori non gaussiens ne le font pas. Par conséquent, les problèmes inverses avec des distributions à priori non gaussiens sont généralement limités à des estimations ponctuelles de la distributions à priori. Deuxièmement, la fonction de densité de probabilité de l'ensemble des images naturelles n'est pas connue analytiquement, et nous devons généralement l'approximer. Dans le chapitre F.3, nous verrons comment l'apprentissage profond peut être utilisé pour calculer certains estimateurs de l'espérance conditionnelle de la distribution postérieure étant donné certains échantillons d'entraînement de la postérieure. Et au chapitre F.5, nous explorerons certaines extensions de ces concepts aux séries temporelles.

F.4 Reconstruction par apprentissage profond pour les problèmes inverses linéaires

L'apprentissage profond est de plus en plus utilisé pour résoudre des tâches de traitement d'images. En particulier, les méthodes d'apprentissage profond ont montré des résultats qui ont dépassé la plupart des méthodes traditionnelles pour calculer des estimateurs ponctuels de variables aléatoires. Dans ce chapitre, nous examinons l'apprentissage profond dans le but de résoudre des problèmes inverses linéaires. En particulier, dans la Sec. F.4.1, nous montrons comment le choix de la fonction de coût d'un réseau de neurones profonds nous permet de calculer une famille d'estimateurs bayésiens ponctuels. Dans la Sec. F.4.2, nous passons en revue les différentes structures de base communes à la plupart des réseaux de neurones profonds.

F.4.1 Reconstruction basée sur l'apprentissage profond pour les problèmes inverses

Nous considérons le problème inverse présenté dans le chapitre **??**

$$\boldsymbol{m} = \boldsymbol{A}\boldsymbol{f} + \boldsymbol{\epsilon}, \tag{F.20}$$
$$\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_\alpha). \tag{F.21}$$

Ce problème inverse peut être utilisé pour modéliser l'acquisition mono-pixel présentée dans Eq. F.21. Comme indiqué dans la section F.3.3, lorsque l'inversion bayésienne complète est irréalisable, un estimateur de $\boldsymbol{f}$ peut être obtenu via l'espérance conditionnelle $\mathbb{E}(\mathbf{f}|\boldsymbol{m})$.

Les méthodes de reconstruction directe par apprentissage profond visent à fournir une correspondance $\mathcal{G}$ qui estime l'espérance conditionnelle.

$$\mathcal{G}(\boldsymbol{m}) = \mathbb{E}(\mathbf{f}|\boldsymbol{m}), \tag{F.22}$$

Pour obtenir un tel estimateur, on résout (voir Annexe C)

$$\underset{\mathcal{G}:\mathbb{R}^M \mapsto \mathbb{R}^N}{\operatorname{argmin}} \; \mathbb{E}\left[\|\mathcal{G}(\boldsymbol{m}) - \boldsymbol{f}\|^2\right]. \tag{F.23}$$

Pour les distributions générales de $\mathbf{f}$, la correspondance $\mathcal{G}^*$ est non linéaire et il n'existe pas de solutions à forme fermée. Au lieu de résoudre l'équation (F.23), qui est généralement difficile à calculer, les méthodes basées sur l'apprentissage profond remplacent l'espérance par la moyenne empirique sur une base de données, et optimisent une correspondance $\mathcal{G}_{\boldsymbol{\theta}}$ dans une famille de correspondances paramétrées par certains poids $\boldsymbol{\theta} \in \mathbb{R}^P$.

Les réseaux de neurones profonds à action directe sont un choix naturel puisque le théorème d'approximation universelle (Hornik, Stinchcombe, and White, 1989) établit qu'ils peuvent approximer toute fonction Borel-mesurable d'un espace de dimension finie à un autre avec n'importe quel degré de précision. Le remplacement de l'espérance par sa contrepartie empirique est justifié par la loi des grands nombres selon laquelle l'espérance empirique converge vers l'espérance réelle lorsque le nombre d'échantillons tend vers l'infini.

Par conséquent, les estimateurs d'apprentissage profond de l'espérance conditionnelle sont approximés en résolvant le problème suivant

$$\mathcal{G}_\theta^* \in \underset{\theta \in \mathbb{R}^P}{\operatorname{argmin}} \frac{1}{S} \sum_{i=0}^{S-1} \left[\|\mathcal{G}_\theta(\boldsymbol{m}_i) - \boldsymbol{f}_i\|^2\right] \tag{F.24}$$

où $(\boldsymbol{f}_i, \boldsymbol{m}_i)_{i \in \{0,\dots,S-1\}}$ sont des échantillons de $(p(\mathbf{f}), p(\mathbf{m}|\mathbf{f}))$ issues d'une base de données d'images.

F.4.2 Structures fondamentales de l'apprentissage profond

F.4.2.1 Le perceptron

Le perceptron a été présenté pour la première fois par Frank Rosenblatt (Rosenblatt, 1961) comme un modèle simple de prise de décision. Il prend plusieurs entrées binaires $x_1, \dots, x_v$ et produit une seule sortie binaire y. Rosenblatt a conçu une règle simple pour calculer la sortie. En introduisant des poids $w_1, \dots, w_v$ qui pondèrent l'importance respective de chaque entrée, et un biais unique b, la sortie est obtenue en comparant la somme pondérée au biais :

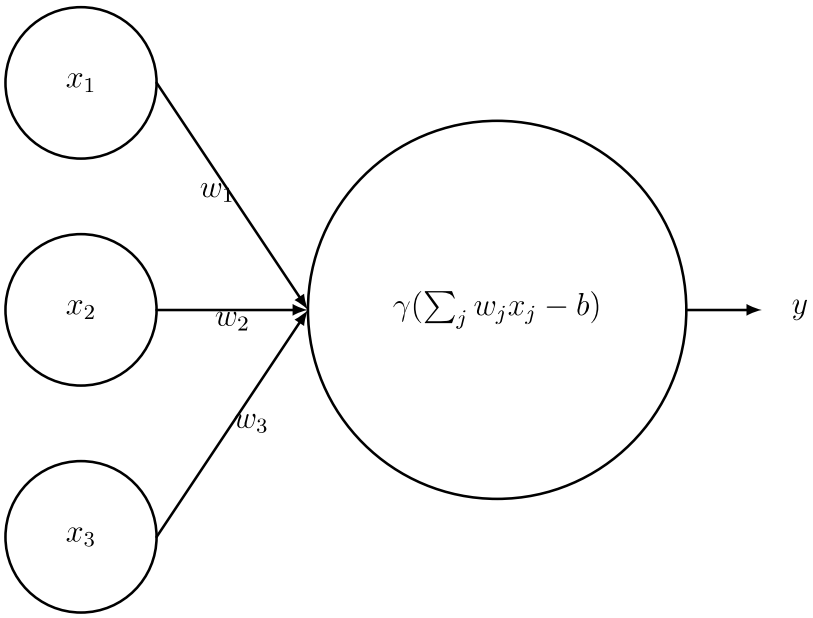


FIGURE F.3 – Perceptron de Rosenblatt à 3 entrées x_1, x_2, x_3. La sortie est générée en appliquant la fonction d'activation non linéaire γ à la somme pondérée $\sum_j w_j x_j - b$.

$$y = \begin{cases} 0 & \text{si } \sum_{j=1}^{v} w_j x_j - b \leq 0, \\ 1 & \text{sinon.} \end{cases} \tag{F.25}$$

Malgré sa simplicité, l'empilement de plusieurs couches de perceptrons permet de réaliser des opérations complexes. Par exemple, l'opération NON-ET, qui est une structure fondamentale de nombreux systèmes électroniques complexes, peut être explicitement implémentée par des perceptrons avec des poids fixes (chaque poids aurait une valeur de -2, et le biais serait de 3 pour produire ladite porte NON-ET).

En raison de la discontinuité de l'opération de seuillage qui est appliquée à la somme pondérée, ce perceptron ne conviendrait pas aux algorithmes qui résolvent l'eq F.24. L'opération de seuillage est remplacée par une fonction continue non linéaire γ appelée "fonction d'activation". Les exemples classiques de cette fonction sont la fonction sigmoïde, la tangente hyperbolique et l'unité linéaire rectifiée (ReLU). Comme le montre la fig. F.3, la sortie y d'un perceptron sera donc calculée comme suit :

$$y = \gamma(\sum_{j=1}^{v} w_j x_j - b). \tag{F.26}$$

F.4.2.2 Réseaux de neurones artificiels

Comme le montre la figure F.4, la manière la plus générale de combiner des perceptrons est la couche de perceptron multicouche. Les couches de perceptron multicouches sont appliquées à la sortie d'autres couches de perceptron multicouches, ce qui conduit à des réseaux neuronaux artificiels à plusieurs couches.

Une couche de perceptron multicouche peut prendre en entrée un vecteur $\boldsymbol{a}^l \in \mathbb{R}^{F_l}$, et sortir un vecteur dans $\boldsymbol{a}^{l+1} \in \mathbb{R}^{F_{l+1}}$. Cette opération $\boldsymbol{a}^l$ est généralement

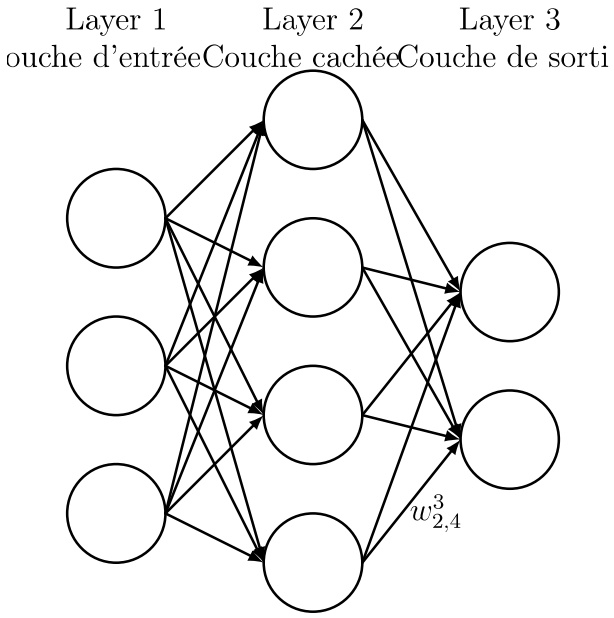


FIGURE F.4 – Perceptron multicouche à deux couches. Les entrées sont de dimension 3, et les sorties de dimension 2. $w^l_{j,k}$ est le poids du $k^{\text{ème}}$ neurone de la $(l-1)^{\text{ème}}$ couche au $j^{\text{ème}}$ neurone de la $l^{\text{ème}}$ couches.

appelée l'état caché de la $l^{\text{ème}}$ couche. Une couche de perceptron multicouche est une combinaison d'une couche linéaire et d'une fonction non linéaire. Chaque composante du vecteur de sortie est le résultat d'un perceptron appliqué au vecteur d'entrée.

$$\boldsymbol{a}^{l+1} = \gamma(\boldsymbol{W}^{l+1}\boldsymbol{a}^l + \boldsymbol{b}^{l+1}), \tag{F.27}$$

où $\boldsymbol{W}^{l+1}$ est la matrice de poids qui contient les différents poids $(w_{j,k})_{1\leq j\leq F_{l+1},1\leq k\leq F_l}$ de tous les différents neurones, $\boldsymbol{b}^{l+1}$ est le vecteur de biais qui contient tous les biais $(b_j^{l+1})_{1\leq j\leq\leq F_{l+1}}$. Cela signifie qu'une couche de perceptron multicouche avec F_{l+1} de sorties et F_l d'entrées a $F_{l+1}\times F_l$ poids, et F_{l+1} biais.

Pour les réseaux neuronaux à propagation avant, nous noterons pour le reste du chapitre w^l_{jk} la valeur du poids k^{th} du neurone j^{th} de la couche l^{th} et b^l_j le biais du neurone j^{th} de la couche l^{th}.

F.4.2.3 Couches de convolution

Les réseaux de neurones convolutifs (LeCun, 1989) ont fait preuve d'une performance exceptionnelle lors de la résolution de problèmes inverses d'imagerie tels que le débruitage (Zhang et al., 2017a), la super-résolution (Dong et al., 2015), et la déconvolution (Xu et al., 2014) ainsi que de nombreux problèmes inverses d'imagerie biomédicale tels que la tomographie computationelle (Kang, Min, and Ye, 2017) et l'IRM accélérée (Jin et al., 2017). Elles constituent donc l'épine dorsale de nombreux réseaux neuronaux. La théorie des couches convolutionnelles est liée à la théorie des framelets (Yin et al., 2017; Yoo, Wahab, and Ye, 2018), selon laquelle les couches convolutionnelles servent de transformation linéaire vers un espace linéaire où le problème inverse est plus facile à résoudre.

Les deux idées principales pour la mise en œuvre des couches convolutionnelles sont l'utilisation de champs réceptifs locaux et l'utilisation de poids et de biais partagés entre les différents champs réceptifs.

— **Champs réceptifs locaux** : Pour les couches entièrement connectées, les entrées et les états cachés sont représentés comme un vecteur de neurones, tous affichés sur une ligne verticale. Dans les réseaux convolutifs, il est plus utile de représenter les entrées sous forme de matrices de neurones élevées au carré. Comme l'illustre la Fig. F.5, les entrées sont ensuite connectées à une couche de neurones cachés, mais contrairement aux couches entièrement connectées, chaque neurone d'une n'est connecté qu'à quelques neurones dans une région carrée (appelée champ réceptif) de la couche. Le champ perceptif local est déplacé dans l'ensemble de l'image pour produire la couche cachée

— **Partage des poids et des biais** : Chaque neurone d'une couche est connecté à un nombre limité de pixels de la couche précédente par l'intermédiaire de son champ réceptif local, et ses poids et biais sont partagés par tous les neurones de la couche cachée. La sortie du $(j,k)^{\text{ème}}$ neurone de la $(l+1)^{\text{ème}}$ couche est la suivante

$$\boldsymbol{a}^{l+1} = \gamma(b^l + \boldsymbol{w}^l * \boldsymbol{a}^l), \tag{F.28}$$

où $*$ désigne l'opération de convolution bidimensionnelle. Cette propriété de partage du poids de la couche convolutive signifie également que tous les neurones détectent la même caractéristique. C'est pourquoi les couches cachées des réseaux neuronaux convolutifs sont appelées *feature maps*.

Comme le montre la Fig. F.6, les couches convolutionnelles peuvent produire plusieurs cartes de caractéristiques en utilisant plusieurs ensembles de noyaux convolutionnels. Pour simplifier leur représentation, les états cachés des couches convolutionnelles sont souvent représentés comme des piles d'images bidimensionnelles où chaque couche est une carte de caractéristiques différente, comme illustré. Nous notons l'état caché de la couche $l^{\text{ème}}$ $\boldsymbol{a}^l \in \mathbb{R}^{M^l \times h \times w}$, où M^l représente le nombre de cartes de caractéristiques dans l'état caché l^{th}, et h et w représentent la hauteur et la largeur de chaque carte de caractéristiques. Enfin, la sortie d'une couche convolutive qui prend en entrée un état caché avec M^l cartes de caractéristiques et en sortie M^{l+1} cartes de caractéristiques est la suivante

$$\boldsymbol{a}^{l+1,m} = \gamma(b^{l,m} + \sum_{\tilde{m}=0}^{M^l} \boldsymbol{w}^{l,m,\tilde{m}} * \boldsymbol{a}^{l,\tilde{m}}), \quad \forall 0 \leq m \leq M^{l+1}. \tag{F.29}$$

F.4.2.4 couches de mise en commun

Outre les couches convolutionnelles, les réseaux neuronaux convolutionnels peuvent également comporter des couches de mise en commun. Leur but est de simplifier les informations contenues dans les cartes de caractéristiques en réduisant la dimension de ces dernières.

La couche de mise en commun produit donc une carte de caractéristiques de dimension réduite où chaque élément est calculé à partir d'une région de la carte de caractéristiques d'entrée. La couche de mise en commun la plus populaire est la couche de mise en commun maximale. Comme illustré dans la Fig. F.7. Une couche max-pool est définie par 2 paramètres : la taille du noyau et le pas. La taille du noyau détermine la taille de la région considérée dans la carte des caractéristiques. Le stride

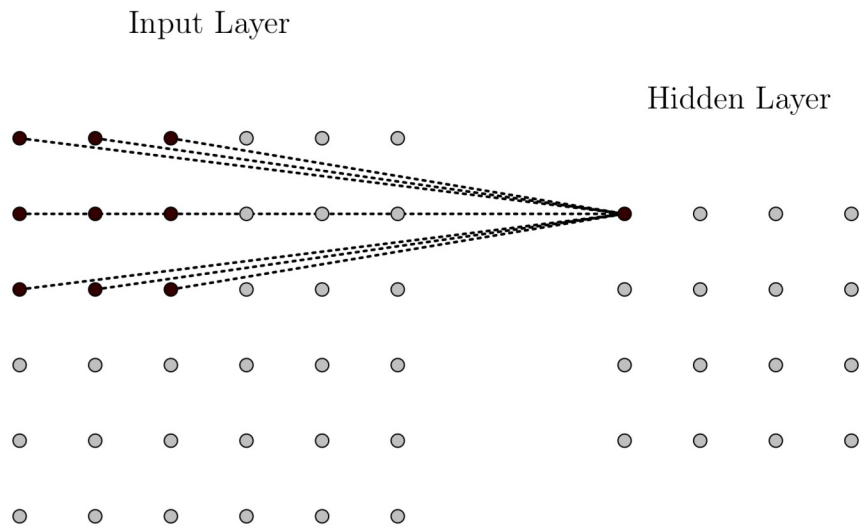


FIGURE F.5 – Illustration des champs réceptifs locaux pour les couches convolutives. On montre une image de taille 8×8 et un champ réceptifs local de taille 3×3.

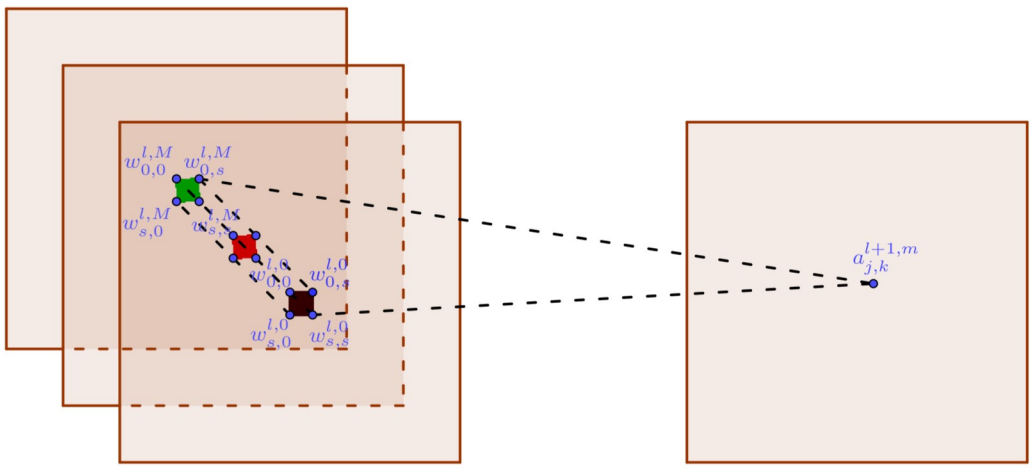


FIGURE F.6 – Représentation d'une couche convolutive avec un noyau de taille $s \times s$. $a^{l+1,m}_{j,k}$ est la sortie du $j,k^{\text{ème}}$ neurone de la $l^{\text{ème}}$ couche, au $m^{\text{ème}}$ carte de fonction.

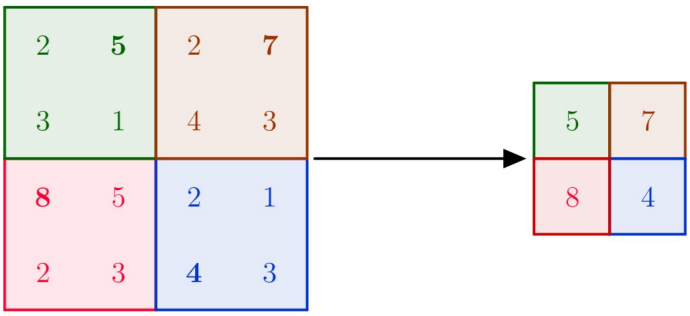


FIGURE F.7 – Exemple de max-pool layer avec noyau de taille 2, une stride de 2. Il conserve le maximum sur les régions et renvoie une image sous-échantillonnée.

détermine le nombre de pixels que nous sautons avant de réappliquer la couche de max-pooling. Ces couches nous permettent d'émuler l'analyse multi-échelle : l'image est analysée à différentes échelles par des réseaux neuronaux profonds, de manière similaire à la théorie de l'analyse en ondelettes.

F.4.2.5 U-net

Parmi les nombreuses combinaisons possibles de couches entièrement connectées, de couches convolutionnelles et de couches de mise en commun, l'architecture U-net est sans doute la plus populaire. Introduite initialement par (Ronneberger et al., 2015) pour la segmentation en IRM, elle a été utilisée avec succès pour résoudre de nombreux problèmes inverses tels que le CT (Jin et al., 2017), la reconstruction en IRM compressée (Sun et al., 2019), le débruitage (Gurrola-Ramos, Dalmau, and Alarcón, 2021).

Le succès de cette architecture réside dans ses deux concepts fondamentaux : le saut de connexions et l'analyse multi-échelle (voir Fig F.8). Les connexions de saut dans le réseau neuronal permettent d'éviter le problème du gradient de fuite car elles réduisent la distance entre une couche donnée et l'extrémité du réseau. En outre, les couches de mise en commun qui sous-échantillonnent l'image permettent d'analyser l'image à plusieurs niveaux de détail. L'intuition du sous-échantillonnage est qu'il permet d'analyser les détails à plusieurs échelles, et la connexion de saut permet également de s'assurer que les informations sur les échelles supérieures ne sont pas perdues lors de la réduction de l'échelle.

F.4.2.6 Conclusion

Ce chapitre présente certains aspects clés de la compréhension de l'apprentissage profond pour la résolution de problèmes inverses linéaires. En particulier, nous avons montré comment calculer certains estimateurs bayésiens tels que l'espérance conditionnelle. En outre, nous avons également présenté certaines des structures clés qui composent les réseaux neuronaux profonds de pointe et avons abordé les aspects pratiques de la formation des réseaux neuronaux profonds.

Ces dernières années, l'apprentissage profond a régulièrement dépassé les méthodes plus traditionnelles. Cependant, l'apprentissage profond manque de certaines garanties théoriques. En particulier, ses performances lorsque peu d'échantillons sont disponibles, ou le manque potentiel de convergence lors de la formation de réseaux

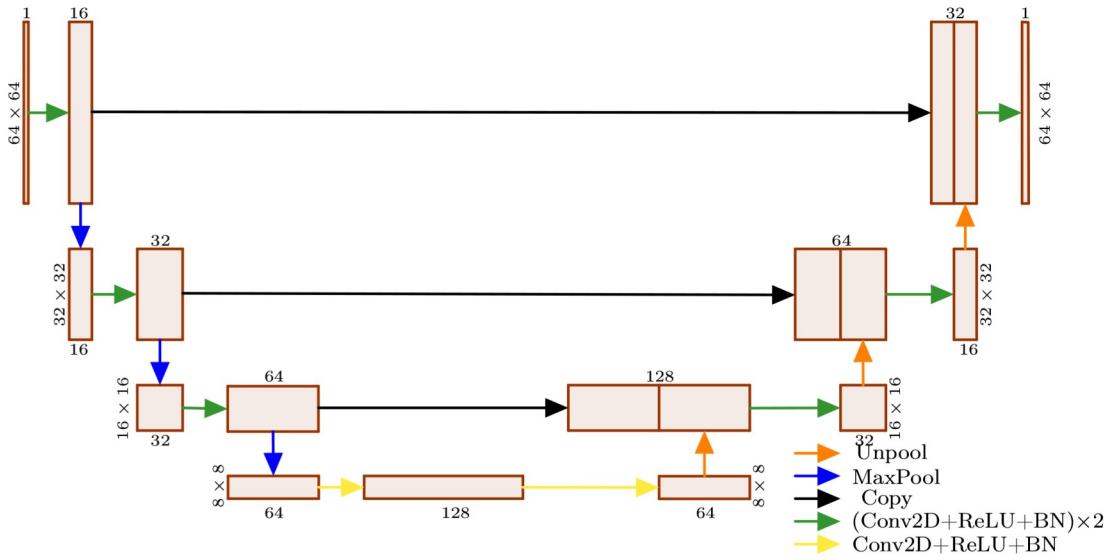


FIGURE F.8 – Exemple de U-net pour des images de traille 64 × 64, avec un canal d'entrée et un canal de sortie. Chaque rectangle représente une carte de fonction dont la profondeur est donnée au dessus de la carte de fonction.

neuronaux, ont été l'un des sujets les plus débattus dans la communauté. Pour cette raison, de nombreux chercheurs ont considéré les réseaux neuronaux profonds comme une "boîte noire" que nous ne comprenons pas toujours.

F.5 Estimation récursive et filtrage séquentiel Bayesien

Comme indiqué dans le chapitre F.2, notre objectif est de reconstruire des vidéos à un seul pixel en temps réel. Les algorithmes de restauration vidéo utilisent l'idée que les différentes images d'une vidéo partagent certaines similarités qui peuvent être exploitées pour améliorer la qualité des vidéos reconstruites. Les problèmes de restauration vidéo peuvent être modélisés comme une procédure de filtrage bayésien (Lu et al., 2020) qui utilise de manière récursive les images restaurées précédentes comme antériorité pour les images actuelles et futures.

Le filtrage bayésien est l'approche bayésienne permettant d'estimer l'état d'un système variant dans le temps qui est indirectement observé par des mesures bruitées. Dans ce contexte, l'état du système peut être interprété comme les images de la vérité du sol d'une vidéo.

Dans ce chapitre, nous introduisons le concept de filtrage bayésien, et les méthodes pour les résoudre. En particulier, nous verrons comment le filtre de Kalman constitue une solution exacte du cas gaussien et linéaire. Nous explorerons également différentes méthodes pour traiter les non-linéarités du système.

F.5.1 Modèles probabilistes d'espace d'état et équations de filtrage bayésien

Nous considérons un modèle d'espace d'état probabiliste comme une séquence de distributions de probabilité conditionnelles

$$\mathbf{f}_t \sim p(\boldsymbol{f}_t|\boldsymbol{f}_{t-1}), \tag{F.30a}$$
$$\mathbf{m}_t^\alpha \sim p(\boldsymbol{m}_t^\alpha|\boldsymbol{f}_t). \tag{F.30b}$$

Dans notre contexte, $\boldsymbol{f}_t$ est l'état du système à un instant t donné (c'est-à-dire la vérité de base de la $^{\text{th}}$ image de la vidéo), $\boldsymbol{m}_t^\alpha$ sont les différentes mesures bruitées réalisées sur les différentes images $\boldsymbol{f}_t$. La fonction de densité de probabilité $p(\boldsymbol{f}_t||\boldsymbol{f}_{t-1})$ décrit le modèle dynamique du système. Ce type de modèle a été utilisé pour les problèmes de détection vidéo comprimée (Ding, Chen, and Wassell, 2014), d'inpainting vidéo (Bugeau et al., 2010), et de réduction des artefacts de compression vidéo (Lu et al., 2018, 2020). La nature stochastique de la dynamique du système ne traduit pas nécessairement une sorte de bruit de processus, elle peut également modéliser les incertitudes de la dynamique du système. La fonction de densité de probabilité $p(\boldsymbol{m}_t^\alpha|\boldsymbol{f}_t)$ est le modèle de mesure ; dans le cas de l'imagerie d'un seul pixel, il s'agit d'une distribution gaussienne telle que $\log p(\boldsymbol{m}_t^\alpha|\boldsymbol{f}_t) \propto \|\boldsymbol{A}\boldsymbol{f}_t - \boldsymbol{m}_t^\alpha\|^2_{(\boldsymbol{\Sigma}_t^\alpha)^{-1}}$.

F.5.2 Filtre de Kalman

Dans cette section, nous considérons le cas linéaire gaussien connu sous le nom de filtre de Kalman (Kalman et al., 1960). En d'autres termes, nous considérons le cas particulier où $p(\boldsymbol{f}_t|\boldsymbol{f}_{t-1}) \propto \|\boldsymbol{f}_t - \boldsymbol{F}_{t-1}\boldsymbol{f}_{t-1}\|^2_{\boldsymbol{Q}_{t-1}^{-1}}$, et $p(\boldsymbol{m}_t^\alpha|\boldsymbol{f}_t) \propto \|\boldsymbol{A}_t\boldsymbol{f}_t - \boldsymbol{m}_t^\alpha\|^2_{(\boldsymbol{\Sigma}_t^\alpha)^{-1}}$. Ceci est équivalent à

$$\boldsymbol{f}_t = \boldsymbol{F}_{t-1}\boldsymbol{f}_{t-1} + \boldsymbol{q}_{t-1}, \tag{F.31}$$
$$\boldsymbol{m}_t^\alpha = \boldsymbol{A}_t\boldsymbol{f}_t + \boldsymbol{\epsilon}_t^\alpha, \tag{F.32}$$

où $\boldsymbol{q}_{t-1} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_{t-1})$, et $\boldsymbol{\epsilon}_t^\alpha \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma}_t^\alpha)$.

En supposant $\mathbf{f}_{t-1}|\boldsymbol{m}_{1:t-1}^\alpha$ Gaussien, $\mathbf{f}_t|\boldsymbol{m}_{1:t-1}^\alpha$ est Gaussien en tant que somme de deux gaussinennes. la distribution de $\mathbf{f}_t|\boldsymbol{m}_{1:t}^\alpha$ est alors

$$\mathbf{f}_t|\boldsymbol{m}_{1:t-1}^\alpha \sim \mathcal{N}(\boldsymbol{f}_t^-, \boldsymbol{P}_t^-), \tag{F.33}$$
$$\mathbf{f}_t|\boldsymbol{m}_{1:t}^\alpha \sim \mathcal{N}(\boldsymbol{f}_t^+, \boldsymbol{P}_t^+). \tag{F.34}$$

La distribution gaussienne est entièrement définie par les matrices de moyenne et de covariance. Ainsi, le calcul de $\boldsymbol{P}_t^-$, $\boldsymbol{f}_t^-$ ainsi que de $\boldsymbol{f}_t^+$ et $\boldsymbol{P}_t^+$ caractérise entièrement la distribution marginale postérieure. Ces paramètres peuvent être entièrement calculés par des formules analytiques

Prédiction

En supposant que nous ayons pu calculer $\mathbf{f}_{t-1}|\boldsymbol{m}_{1:t-1}^\alpha \sim \mathcal{N}(\boldsymbol{f}_{t-1}^+, \boldsymbol{P}_{t-1}^+)$ à l'étape précédente, nous pouvons calculer la moyenne et la covariance comme une combinaison linéaire de deux variables aléatoires gaussiennes (c'est-à-dire $\mathbf{f}_{t-1}$ et $\mathbf{q}_{t-1}$) en utilisant F.32. c'est-à-dire $\mathbf{f}_{t-1}$, et $\mathbf{q}_{t-1}$) en utilisant F.32 L'étape de prédiction est

$$\boldsymbol{f}_t^- = \mathbb{E}(\mathbf{f}_t|\boldsymbol{m}_{1:t-1}^\alpha) = \boldsymbol{F}_{t-1}\boldsymbol{f}_{t-1}^+, \tag{F.35}$$
$$\boldsymbol{P}_t^- = \mathbb{C}\text{ov}(\mathbf{f}_t|\boldsymbol{m}_{1:t-1}^\alpha) = \boldsymbol{F}_{t-1}\boldsymbol{P}_{t-1}^+\boldsymbol{F}_{t-1}^\top + \boldsymbol{Q}_{t-1}. \tag{F.36}$$

Mise à jour

La mise à jour donne

$$\mathbf{f}_t|\boldsymbol{m}_{1:t-1}^{\alpha} \sim \mathcal{N}(\boldsymbol{f}_t^-, \boldsymbol{P}_t^-), \tag{F.37}$$
$$\mathbf{m}_t^{\alpha}|\boldsymbol{f}_t \sim \mathcal{N}(\boldsymbol{A}_t\boldsymbol{f}_t, \boldsymbol{\Sigma}_t^{\alpha}), \tag{F.38}$$

la distribution de $\mathbf{f}_t|\boldsymbol{m}_{1:t}^{\alpha}$ est normalement distribuée, et la moyenne et la covariance sont calculées via la formule suivante

$$\boldsymbol{f}_t^+ = \mathbb{E}(\mathbf{f}_t|\boldsymbol{m}_{1:t}^{\alpha}) = \boldsymbol{f}_t^- + \boldsymbol{P}_t^-\boldsymbol{A}_t^\top(\boldsymbol{A}_t\boldsymbol{P}_t^-\boldsymbol{A}_t^\top + \boldsymbol{\Sigma}_t^{\alpha})^{-1}[\boldsymbol{m}_t^{\alpha} - \boldsymbol{A}_t\boldsymbol{f}_t^-], \tag{F.39}$$
$$\boldsymbol{P}_t^+ = \mathbb{C}\mathrm{ov}(\mathbf{f}_t|\boldsymbol{m}_{1:t}^{\alpha}) = \boldsymbol{P}_t^- - \boldsymbol{P}_t^-\boldsymbol{A}_t^\top(\boldsymbol{A}_t\boldsymbol{P}_t^-\boldsymbol{A}_t^\top + \boldsymbol{\Sigma}_t^{\alpha})^{-1}\boldsymbol{A}_t\boldsymbol{P}_t^-. \tag{F.40}$$

F.5.3 Estimation d'état non-lineaire

Lorsque les modèles dynamiques et/ou de mesure ne sont pas linéaires, et que les modèles de bruit ne sont pas gaussiens, l'utilisation des filtres de Kalman sera mal adaptée pour décrire l'évolution du système. Dans cette section, nous décrivons les alternatives au filtre de Kalman pour les systèmes non linéaires et non gaussiens. Pour le reste de cette section, nous considérons que la dynamique du système est décrite par

$$\boldsymbol{f}_t = \mathfrak{F}(\boldsymbol{f}_{t-1}, \boldsymbol{q}_{t-1}), \tag{F.41a}$$
$$\boldsymbol{m}_t^{\alpha} = \mathfrak{A}(\boldsymbol{f}_t, \boldsymbol{\epsilon}_t^{\alpha}), \tag{F.41b}$$

où $\boldsymbol{q}_{t-1} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{Q}_{t-1})$, and $\boldsymbol{\epsilon}_t^{\alpha} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_t^{\alpha})$.

F.5.3.1 Filtre de Kalman étendu

Le filtre de Kalman étendu (Jazwinski, 1970) utilise l'approximation de Taylor au premier ordre pour linéariser le système d'équations F.41.

Prédiction

Nous considérons le développement de Taylor de $\mathfrak{F}$ près du point $(\boldsymbol{f}_{t-1}, \mathbf{0})$. Cela conduit à l'étape de prédiction suivante en appliquant les équations de prédiction de Kalman

$$\boldsymbol{f}_t^- = \mathbb{E}(\mathbf{f}_t|\boldsymbol{m}_{1:t-1}^{\alpha}) = \mathfrak{F}(\boldsymbol{f}_{t-1}, \mathbf{0}), \tag{F.42}$$
$$\boldsymbol{P}_t^- = \mathbb{C}\mathrm{ov}(\mathbf{f}_t|\boldsymbol{m}_{1:t-1}^{\alpha}) = \boldsymbol{F}_{t-1}\boldsymbol{P}_{t-1}^+\boldsymbol{F}_{t-1}^\top + \boldsymbol{L}_{t-1}\boldsymbol{Q}_{t-1}\boldsymbol{L}_{t-1}^\top, \tag{F.43}$$

où $\boldsymbol{F}_{t-1} = \frac{\partial\mathfrak{F}}{\partial\boldsymbol{f}}(\boldsymbol{f}_{t-1}^+, \mathbf{0})$ and $\boldsymbol{L}_{t-1} = \frac{\partial\mathfrak{F}}{\partial\boldsymbol{q}}(\boldsymbol{f}_{t-1}^+, \mathbf{0})$.

Mise à jour

De manière similaire à l'étape de prédiction, en appliquant l'expansion de Taylor de $\mathfrak{A}$ au point $(\boldsymbol{f}_t^-, \mathbf{0})$, on obtient moyenne and covariance de la distribution conditionnelle

$$\boldsymbol{f}_t^+ = \mathbb{E}(\mathbf{f}_t|\boldsymbol{m}_{1:t-1}^{\alpha}) = \boldsymbol{f}_t^- + \boldsymbol{P}_t^-\boldsymbol{A}_t^{f\top}(\boldsymbol{A}_t^f\boldsymbol{P}_t^-\boldsymbol{A}_t^{f\top} + \boldsymbol{A}_t^{\epsilon}\boldsymbol{\Sigma}_t^{\alpha}\boldsymbol{A}_t^{\epsilon\top})^{-1}[\boldsymbol{m}_t^{\alpha} - \mathfrak{A}(\boldsymbol{f}_t^-, \mathbf{0})], \tag{F.44}$$
$$\boldsymbol{P}_t^+ = \mathbb{C}\mathrm{ov}(\mathbf{f}_t|\boldsymbol{m}_{1:t}^{\alpha}) = \boldsymbol{P}_t^- - \boldsymbol{P}_t^-\boldsymbol{A}_t^{f\top}(\boldsymbol{A}_t^f\boldsymbol{P}_t^-\boldsymbol{A}_t^{f\top} + \boldsymbol{A}_t^{\epsilon}\boldsymbol{\Sigma}_t^{\alpha}\boldsymbol{A}_t^{\epsilon\top})^{-1}\boldsymbol{A}_t^f\boldsymbol{P}_t^-. \tag{F.45}$$

F.5.3.2 Filtre particulaire

Les filtres particulaires sont des estimations basées sur la méthode de Monte-Carlo de la fonction de densité de probabilité postérieure $p(\boldsymbol{f}_t|$ $vecm^{\alpha}_{1:t})$. Ils utilisent l'échantillonnage préférentiel pour générer des échantillons de la distribution postérieure. Ces échantillons permettent ensuite de calculer des estimations de l'espérance de la distribution postérieure $p(\boldsymbol{f}_t|$ $vecm^{\alpha}_{1:t})$.

Nous présenterons la méthode d'échantillonnage préférentiel séquentielle (Godsill, Doucet, and West, 2004) sur laquelle de nombreux autres filtres particulaires sont basés.

Algorithm 8 Échantillonnage préférentiel séquentiel

Initialisation : Échantillonner S échantillons $\{\boldsymbol{f}_0^{(i)}\}_{1\leq i\leq S} \sim p(\boldsymbol{f}_0)$. Choisir les poids $w_0^{(i)} = 1/S$.
for $t \in \{1, \ldots, T\}$ **do**
1 - Échantillonner S échantillons $\{\boldsymbol{f}_t^{(i)}\}_{1\leq i\leq S}$ de la densité biaisée :
$\boldsymbol{f}_t^{(i)} \sim \pi(\boldsymbol{f}_t|\boldsymbol{f}_{1:t-1}, \boldsymbol{m}^{\alpha}_{1:t})$.
2 - Calculer les poinds non-normalisés $\tilde{w}_t^{(i)}$
$\tilde{w}_t^{(i)} = \frac{p(\boldsymbol{m}_t^{\alpha}|\boldsymbol{f}_t^{(i)})p(\boldsymbol{f}_t^{(i)}|\boldsymbol{f}_{t-1}^{(i)})}{\pi(\boldsymbol{f}_t^{(i)}|\boldsymbol{f}_{t-1}^{(i)}, \boldsymbol{m}^{\alpha}_{1:t})} w_{t-1}^{(i)}$
3 - Normaliser les poids $\tilde{w}_t^{(i)}$ pour avoir des poids normalisés $w_t^{(i)}$ dont la somme est 1.
end for

Parfois, cet algorithme conduit à la création de nombreux échantillons avec des poids proches de zéro. Pour résoudre ce problème de "dégénérescence", le ré-échantillonnage (Kitagawa, 1996) peut être utilisé comme une étape de $4^t extth$ (voir l'annexe E pour plus de détails sur une implémentation potentielle de l'étape de ré-échantillonnage).

Conclusion

Dans ce chapitre, nous explorons le filtrage bayésien, ainsi que les différentes méthodes associées au filtrage bayésien. Le filtrage bayésien a pour but de calculer la distribution a posteriori d'un système variant dans le temps, observé par des mesures bruyantes et potentiellement incomplètes. Les filtres bayésiens ont des solutions exactes dans le cas linéaire gaussien grâce aux filtres de Kalman. Cependant, pour les systèmes non linéaires, il n'existe pas de solution générale. Par conséquent, les systèmes non linéaires sont généralement approximés comme gaussiens, ou sont traités en utilisant des simulations basées sur l'échantillonnage de Monte-Carlo.

Les problèmes de restauration vidéo peuvent souvent être modélisés par un problème de filtrage bayésien. Cependant, l'une des limites de ces méthodes est la nécessité d'un modèle prédictif de la dynamique du système.

F.6 Reconstruction d'images mono-pixel à partir de données expérimentales à l'aide de réseaux neuronaux

La reconstruction d'images par apprentissage profond a donné des résultats comparables aux méthodes traditionnelles de reconstruction d'images. Les bibliothèques

d'apprentissage profond récentes ont également optimisé l'utilisation des réseaux neuronaux sur les unités de traitement graphique, ce qui rend les reconstructeurs à apprentissage profond adaptés aux applications en temps réel. L'un des facteurs limitants est l'approche "boîte noire" de l'apprentissage profond, qui le rend difficile à comprendre et à appliquer aux données expérimentales.

Dans ce chapitre, nous présentons une méthode de reconstruction basée sur l'apprentissage profond qui combine la régularisation de Tikhonov avec un reconstructeur d'apprentissage profond image par image. L'application d'une régularisation de Tikhonov aux données brutes avant l'application du reconstructeur à apprentissage profond permet de généraliser à différentes valeurs du nombre de photons, ce qui facilite l'application aux données d'un dispositif expérimental. Une partie du matériel de ce chapitre provient de deux actes de conférence : IEEE ISBI 2020 (Ducros, Lorente Mur, and Peyrin, 2020) où le cas sans bruit est considéré, SPIE photonics Europe 2020 (Lorente Mur et al., 2020), où nous présentons le dispositif expérimental utilisé pour valider nos algorithmes. Ce chapitre contient aussi en grande partie un article publié dans Optics Express en 2021 (Lorente Mur et al., 2021).

résumé

Les caméras monopixel qui mesurent les coefficients d'image ont diverses applications prometteuses, en particulier pour l'imagerie hyper-spectrale. Nous étudions ici les réseaux neuronaux profonds qui, lorsqu'ils sont alimentés par des données expérimentales, peuvent produire des images de haute qualité en temps réel. En supposant que les mesures sont corrompues par un bruit mixte Poisson-Gaussien, nous proposons de faire correspondre les données brutes du domaine de la mesure au domaine de l'image sur la base d'une régularisation Tikhonov. Cette étape peut être implémentée comme la première couche d'un réseau neuronal profond, suivie de toute architecture de couches qui agit dans le domaine de l'image. Nous décrivons également un cadre pour l'entraînement du réseau en présence de bruit. En particulier, notre approche comprend une estimation de l'intensité de l'image et des paramètres expérimentaux, ainsi qu'un schéma de normalisation qui permet de gérer des niveaux de bruit variables pendant la formation et les tests. Enfin, nous présentons les résultats de simulations et d'acquisitions expérimentales avec des niveaux de bruit variables. Notre approche produit des images avec des rapports signal/bruit de pointe améliorés, même pour des niveaux de bruit prévus lors de l'apprentissage des réseaux, ce qui rend l'approche particulièrement adaptée au traitement des données expérimentales. En outre, bien que cette approche se concentre sur l'imagerie à un seul pixel, elle peut être adaptée à d'autres problèmes d'optique computationnelle.

Introduction

L'imagerie monopixel est une configuration extrême de l'optique computationnelle, dans laquelle un détecteur à point unique est utilisé pour récupérer une image (Edgar, Gibson, and Padgett, 2018). Depuis les travaux précurseurs de Duarte et de ses collègues (Duarte et al., 2008), l'imagerie monopixel a été appliquée avec succès à la microscopie à fluorescence (Studer et al., 2012), à l'imagerie hyperspectrale (Rousset et al., 2018; Arce et al., 2014), la tomographie optique diffuse (Pian et al., 2017), la chirurgie guidée par l'image (Aguénounon et al., 2019), l'imagerie infrarouge à ondes courtes (Zhang et al., 2020), et l'imagerie à travers des milieux diffusants (Li et al., 2020a). Les mesures sur un seul pixel peuvent être modélisées comme des produits scalaires entre une image et certaines fonctions bidimensionnelles qui sont mises en

œuvre par un modulateur de lumière spatial (Edgar, Gibson, and Padgett, 2018). Pour limiter les temps d'acquisition, il est fortement souhaitable de réduire le nombre de motifs lumineux, ce qui conduit à un problème inverse indéterminé pour récupérer l'image.

Dans le domaine de l'imagerie monopixel, l'apprentissage profond a été utilisé pour démixer l'intensité et la durée de vie de la fluorescence à partir de mesures résolues dans le temps (Yao et al., 2019; Smith, Ochoa, and Intes, 2020). Higham et collaborateurs (Higham et al., 2018) ont proposé un auto-encodeur convolutif pour l'imagerie de reconstruction d'images à un seul pixel qui surpasse les approches de détection comprimée. Ce réseau mappe directement le vecteur de mesure vers l'image souhaitée, en utilisant une couche entièrement connectée suivie de couches convolutives. Plusieurs architectures de Deep Learning ont été proposées depuis, pour résoudre les problèmes de reconstruction à un seul pixel. Par exemple, Li et al. in (Li et al., 2020a) ont utilisé un réseau similaire à celui introduit par (Higham et al., 2018). Pour les mesures avec des rapports signal/bruit très faibles, dans (Rizvi et al., 2020) les auteurs ont proposé un U-net pour l'imagerie de Fourier à pixel unique hautement compressée, alors que dans (Hoshi et al., 2020), un réseau neuronal récurrent a été utilisé. En outre, Li et ses collègues (Li et al., 2020b) investigué un réseau adversarial génératif conditionnel pour reconstruire des données hautement compressées à partir de mesures avec des modèles binaires aléatoires.

Dans le présent travail, nous proposons une nouvelle méthodologie d'apprentissage profond pour la reconstruction d'images à partir de mesures à un seul pixel corrompues par un modèle de bruit mixte Poisson-Gaussien. Traditionnellement, les problèmes inverses avec des données corrompues par un bruit de Poisson sont abordés en utilisant des transformées stabilisant la variance, comme la transformée d'Anscombe (Anscombe, 1948), suivie d'un filtre de Wiener (Wiener, 1949). Cependant, l'image résultante peut être floue, en particulier pour les données sous-échantillonnées. Des alternatives plus récentes ont été utilisées pour résoudre ce problème en exploitant des prieurs d'image statistiques ou artisanaux (Lefkimmiatis and Unser, 2013; Ding et al., 2018; Li, Luisier, and Blu, 2016). La résolution des problèmes qui en résultent est généralement prohibitive pour les applications en temps réel, car une seule image reconstruite peut prendre plusieurs secondes. À cet égard, les réseaux profonds sont des candidats idéaux en raison de leur évaluation rapide. À ce titre, l'apprentissage profond a été utilisé pour l'imagerie d'un seul pixel en présence de bruit de Poisson (Liu et al., 2020), bien qu'aucune modification particulière n'ait été apportée au réseau neuronal pour tenir compte du bruit de Poisson.

De nombreuses études ont exploré de nombreuses architectures de réseaux neuronaux différentes ; cependant, elles considèrent généralement des couches convolutionnelles qui agissent dans le domaine de l'image, alors que les données brutes sont dans le domaine de la mesure. En outre, l'interprétation de la sortie d'un réseau neuronal reste une question ouverte, notamment en présence de bruit où des problèmes de robustesse peuvent survenir, comme l'indique (Kim et al., 2018). En particulier, dans le domaine de l'imagerie à un seul pixel, un réseau neuronal qui n'est pas finement adapté au système peut être surpassé par des reconstructeurs linéaires (Jiao et al., 2020).

Contribution

Tout d'abord, nous explorons la meilleure façon de mapper les données brutes dans le domaine de l'image, avant d'appliquer une cascade de couches convolutives.

Cette mise en correspondance est traditionnellement apprise ou calculée par le pseudo-inverse de Moore-Penrose. Nous décrivons une cartographie linéaire qui peut être interprétée en termes de reconstruction d'image traditionnelle. Nous décrivons une application linéaire qui peut être interprétée en termes de reconstruction d'image traditionnelle. Cette application linéaire est implémentée comme la première couche d'un réseau neuronal profond (non linéaire). Nous avons introduit cette idée dans (Ducros, Lorente Mur, and Peyrin, 2020) pour les données non bruitées, et ici nous l'étendons aux données bruitées. En particulier, nous proposons d'estimer l'intensité de l'image, qui est inconnue en pratique, et d'utiliser cette estimation pour approximer la matrice de covariance des mesures.

Ensuite, nous décrivons un mécanisme qui permet d'entraîner un réseau en présence d'un bruit de Poisson. Nous fournissons un schéma de normalisation qui permet de considérer des données brutes avec différents ordres de grandeur, ce qui est obligatoire pour des scénarios réalistes où l'intensité de l'image n'est pas connue.

Enfin, nous validons notre approche par la reconstruction d'un ensemble de données expérimentales que nous avons acquises avec des temps d'intégration et des flux lumineux variables. Après acceptation, les jeux de données seront mis à disposition avec les implémentations de nos méthodes de reconstruction dans la boîte à outils Python (SPyRiT (Lorente Mur and Ducros, 2020)).

F.6.1 Problème et limitations

F.6.1.1 Modèle d'acquisition

Comme les motifs Hadamard contiennent des valeurs négatives, nous adoptons une stratégie différentielle (Lorente Mur et al., 2019). Nous désignons par $\hat{\mathbf{m}}_{+}^{\alpha}$ les mesures des parties positives des motifs, et par $\hat{\mathbf{m}}_{-}^{\alpha}$ les mesures des parties négatives. Nous modélisons le bruit comme un mélange de distributions de Poisson et de Gauss (Foi et al., 2008; Rosenberger et al., 2016). Le bruit de Poisson dépend du signal et provient de la nature discrète de la charge électronique, tandis que le bruit gaussien indépendant du signal tient compte des fluctuations de la lecture et du circuit. Pour les mesures positives et négatives, nous avons

$$\hat{\mathbf{m}}_{+,-}^{\alpha} \sim K\,\mathcal{P}(\alpha \boldsymbol{H}_1^{+,-}\boldsymbol{f}) + \mathcal{N}(\mu_{\text{dark}}, \sigma_{\text{dark}}^2) \tag{F.46}$$

où $\mathcal{P}$ et $\mathcal{N}$ sont les distributions de Poisson et Gaussienne, K est une constante qui représente le gain global du système (en comptes/électron), α est l'intensité (en photons) de l'image (qui est proportionnelle au temps d'intégration), μ_{dark} est le courant d'obscurité (en comptes), et σ_{dark} est le bruit d'obscurité (en comptes). Nous supposons en outre que μ_{dark} et σ_{dark} sont indépendants de l'intensité de l'image α.

Les mesures normalisées $\mathbf{m}^{\alpha}$ sont finalement définies comme suit

$$\mathbf{m}^{\alpha} = (\hat{\mathbf{m}}_{+}^{\alpha} - \hat{\mathbf{m}}_{-}^{\alpha})/(\alpha K), \tag{F.47}$$

F.6.2 Projection des mesures brutes dans le domaine image

Nous proposons d'utiliser une solution interprétable régularisée par Tikhonov (Tarantola, 2005) pour mapper les données brutes dans le domaine des images. En particulier, nous choisissons

$$\tilde{\boldsymbol{f}} = \mathcal{G}^{1}(\hat{\boldsymbol{m}}^{\alpha}) = \boldsymbol{H}^{\top}\boldsymbol{y}^{*}, \tag{F.48}$$

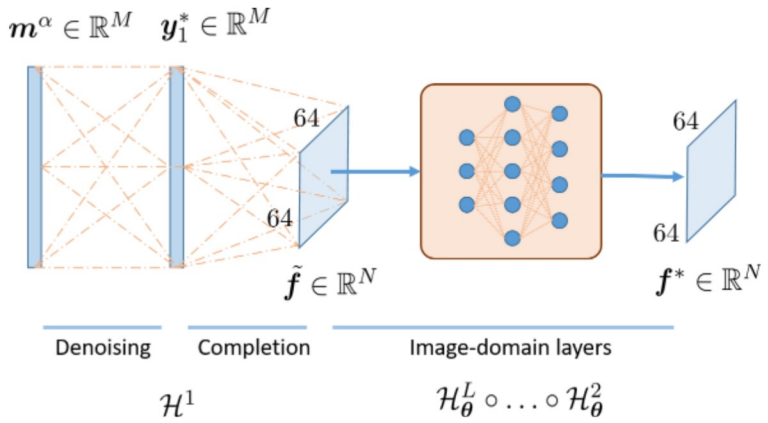


FIGURE F.9 – Réseau proposé. Les deux premières couches transposent les mesures dans le domaine de l'image selon la solution analytique donnée par l'équation (F.49). Ces deux couches peuvent être interprétées comme une couche qui débruite l'équation (F.49a) des mesures brutes, suivie d'une couche qui estime les coefficients manquants à partir des mesures débruitées de l'équation (F.49b). L'image brute $\tilde{f}$ est corrigée par une cascade de couches de convolution du domaine de l'image (CL) et de couches non linéaires, telles que les couches ReLU.

où les données régularisées y^* sont telles que $y^* = [y_1^*, y_2^*]^\top$, où $y_1^* \in \mathbb{R}^M$ et $y_2^* \in \mathbb{R}^{(N-M)}$ sont des versions régularisées des coefficients acquis et manquants, respectivement.

Soit $\boldsymbol{\Sigma_\alpha}$ la variance de nos mesures bruitées, $\boldsymbol{\mu}$ et $\boldsymbol{\Sigma}$ sont respectivement la moyenne et la variance de notre modèle préalable dans le domaine Hadamard. Nous avons dérivé la solution analytique régularisée de Tikhonov donnée par

$$\boldsymbol{y}_1^*(\boldsymbol{m}^\alpha) = \boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_1[\boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_\alpha]^{-1}(\boldsymbol{m}^\alpha - \boldsymbol{\mu}_1), \tag{F.49a}$$

$$\boldsymbol{y}_2^*(\boldsymbol{m}^\alpha) = \boldsymbol{\mu}_2 + \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_1^{-1}[\boldsymbol{y}_1^*(\boldsymbol{m}^\alpha) - \boldsymbol{\mu}_1]. \tag{F.49b}$$

où $vec\Sigma_1 \in \mathbb{R}^{M\times M}$, $\boldsymbol{\Sigma}_{21} \in \mathbb{R}^{(N-M)\times M}$ et $\boldsymbol{\Sigma}_2 \in \mathbb{R}^{(N-M)\times(N-M)}$ sont les blocs de la covariance $\boldsymbol{\Sigma}$ dans le domaine de mesure, $\boldsymbol{\mu}_1 \in \mathbb{R}^M$ et $\boldsymbol{\mu}_2 \in \mathbb{R}^{(N-M)}$ sont les blocs de $\boldsymbol{\mu}$. Il est intéressant de noter que l'équation (F.49a) peut être interprétée comme le débruitage des données brutes dans le domaine de la mesure, tandis que l'équation (F.49b) est l'estimation des coefficients manquants à partir des coefficients acquis débarrassés.

Pour contourner la difficulté d'inverser la matrice dépendant du signal dans l'équation (F.49a), nous choisissons de négliger les termes non diagonaux de $\boldsymbol{\Sigma}_1$; En dénotant $\boldsymbol{\sigma}_1^2 = \mathsf{diag}\,(\boldsymbol{\Sigma}_1)$, nous obtenons

$$\boldsymbol{y}_1^*(\boldsymbol{m}^\alpha) = \boldsymbol{\mu}_1 + \boldsymbol{\sigma}_1^2/(\boldsymbol{\sigma}_1^2 + \boldsymbol{\sigma}_\alpha^2)(\boldsymbol{m}^\alpha - \boldsymbol{\mu}_1), \tag{F.50}$$

où la division et la multiplication s'appliquent par élément.

La méthode finale proposée est résumée dans la Fig. F.9, où la sortie des étapes de débruitage et de complétion est ensuite corrigée par un réseau neuronal profond dans le domaine de l'image.

F.6.3 Résultats sur données expérimentales

Comme le montre la première colonne de la Fig. F.10, nous acquérons trois objets différents : la lampe LED directement (première rangée), un chat du jeu d'essai STL-10 imprimé sur une feuille transparente (rangée du milieu) et la cible de résolution

Siemens Star (rangée du bas) (*Photography — Electronic still picture imaging — Resolution and spatial frequency responses* 2019). L'image de référence est calculée à partir d'un vecteur de mesure entièrement échantillonné qui est acquis avec le rapport signal/bruit le plus élevé (c'est-à-dire un éclairage à flux élevé, un temps d'acquisition long). Plus précisément, nous considérons qu'il n'y a pas de densité neutre, et le temps d'intégration à 1 ms par motif pour la lampe, à 4 ms par motif pour le chat STL-10, et à 8 ms par motif pour la cible Siemens,

Pour chaque objet, nous acquérons également un ensemble de données à haut bruit en plaçant une densité optique neutre derrière la lampe, afin de réduire le flux lumineux. Nous considérons des densités optiques de 1,3 pour la lampe LED, de 0,6 pour le chat STL-10 et de 0,3 pour la cible Siemens. Nous fixons le temps d'intégration à 4 ms pour la lampe LED et le chat STL-10, et à 8 ms pour la cible Siemens, nous ne retenons que $M = 512$ mesures pour la lampe LED et le chat STL-10, ce qui accélère l'acquisition d'un facteur 8. Pour la cible Siemens, dont le contenu en fréquences spatiales est plus riche, nous conservons plus de mesures ($M = 2048$; facteur d'accélération de 2).

Nos paramètres instrumentaux estimés étaient : $\mu_{\text{dark}} = 1070$ comptes, $K = 1{,}54$ compte/électron, et $\sigma_{\text{dark}} = 53$ photons.

Notre méthode permet des reconstructions visuellement convaincantes et de bonnes performances en termes de PSNR contre plusieurs niveaux de bruit, lorsqu'elle est entraînée pour de faibles niveaux de bruit (valeurs élevées de α). Elle présente donc de bonnes propriétés pour le traitement des données expérimentales. Contrairement aux études précédentes sur l'apprentissage profond à un seul pixel, telles que (Higham et al., 2018; Li, 2010; Rizvi et al., 2020; Hoshi et al., 2020), nous avons montré la robustesse de notre méthode à différentes conditions d'éclairage, et donné une signification interprétable à la première couche de notre réseau neuronal.

Le niveau de bruit de l'image en cours d'acquisition est une caractéristique clé dans les expériences réelles. Ce paramètre peut être estimé en premier lieu, puis le réseau qui s'adapte au niveau de bruit réel peut être évalué. Cependant, comme le niveau de bruit est inconnu, cela nécessite le chargement de plusieurs réseaux, ce qui peut être une limitation sévère pour les applications en temps réel, et en particulier si les modèles sont trop grands pour être tous stockés sur l'unité de processeur graphique. La méthode que nous proposons est robuste à différents niveaux de bruit tant qu'elle a été entraînée avec de faibles niveaux de bruit (valeurs élevées de α). Par conséquent, le même réseau peut être utilisé de manière quasi optimale dans une grande variété de situations. La cartographie Tikhonov-régularisée que nous proposons permet de débruiter les données brutes, afin de fournir une reconstruction approximative aux couches convolutives qui peuvent se concentrer sur l'apprentissage des caractéristiques spatiales.

Une limitation de notre réseau profond par rapport à des approches plus classiques telles que la (Lefkimmiatis and Unser, 2013; Luisier, Blu, and Unser, 2011; Li, Luisier, and Blu, 2016) est qu'il n'y a aucune garantie théorique qu'il fonctionnera pour n'importe quelle image ; en particulier, si l'image en cours d'acquisition diffère significativement de celles de notre ensemble d'entraînement. Il s'agit d'une préoccupation commune pour les approches d'apprentissage profond qui, cependant, sont considérées comme fonctionnant bien dans la pratique. Bien que cela tende à être confirmé par nos résultats expérimentaux où notre approche a fonctionné sur le secteur Siemens Star et sur une lumière LED (tous deux très différents des images de la base de données stl-10), il n'y a aucune garantie théorique que ce sera toujours le cas.

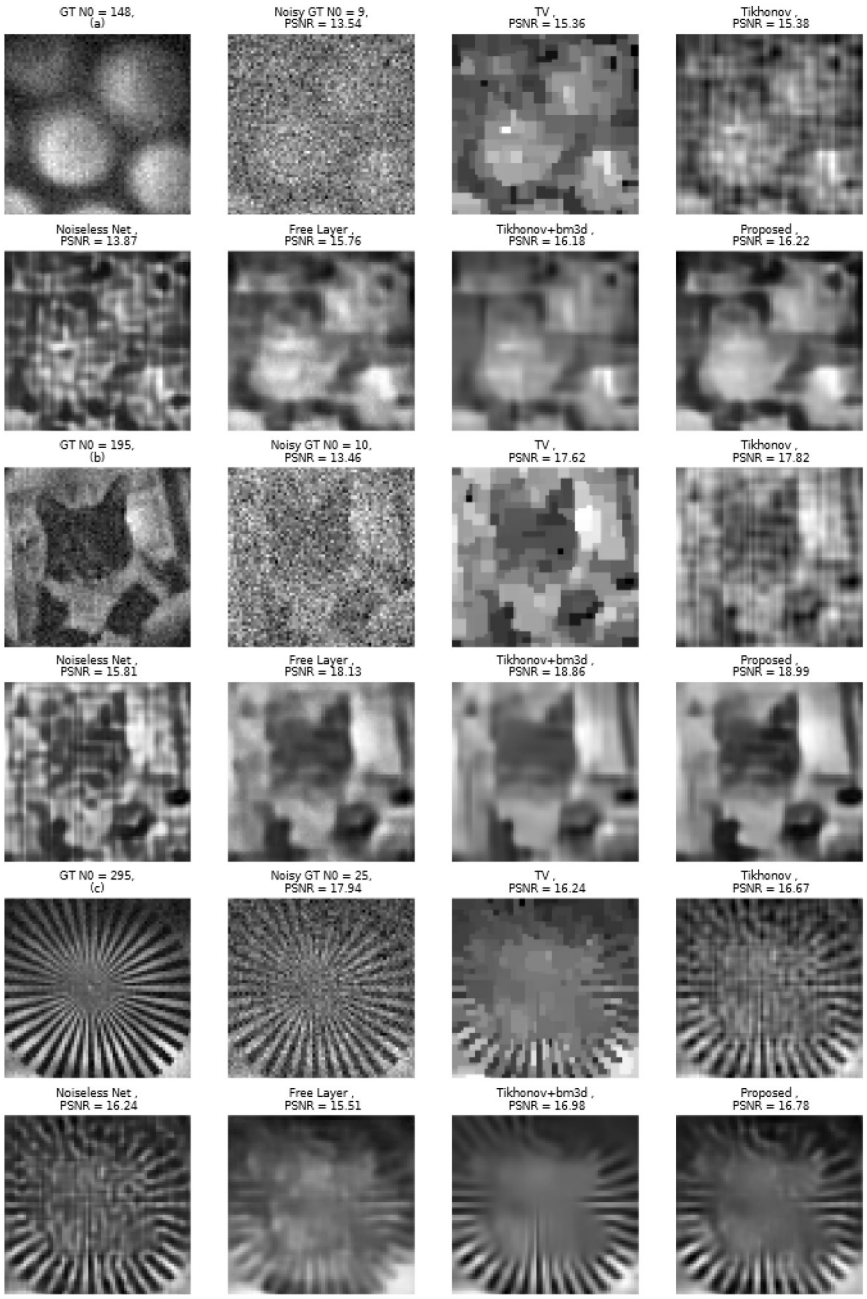


FIGURE F.10 – Reconstructions de trois ensembles de données expérimentales par les différentes méthodes (rangée du haut : Lampe LED avec $M = 512$; ligne du milieu : Chat STL-10 avec $M = 512$; rangée du bas : Cible de résolution Siemens Star avec $M = 1024$). Nous affichons les images reconstruites à partir d'un ensemble de données entièrement échantillonnées (ground-truth ; GT) acquis avec une intensité d'image élevée (première colonne, $\alpha = 148$ photons, $\alpha = 195$ photons et $\alpha = 295$ photons pour la lampe LED, le chat STL-10 et la cible de résolution Siemens Star, respectivement) et une intensité d'image plus faible ('Noisy GT' deuxième colonne, $\alpha = 9$ photons, $\alpha = 10$ photons et $\alpha = 25$ photons pour la lampe LED, le chat STL-10 et la cible de résolution Siemens Star, respectivement). Les colonnes suivantes montrent les reconstructions réalisées à l'aide de la solution régularisée de la variation totale (Li, 2010), de la solution régularisée de Tikhonov de l'équation (F.50), du réseau sans bruit proposé (Noiseless Net) (réseau entraîné sans bruit et où nous supposons que $\boldsymbol{\Sigma}_\alpha = \mathbf{0}$), un réseau neuronal profond avec la même architecture que notre méthode proposée, où le mapping est appris comme dans (Higham et al., 2018) (couche libre), la méthode régularisée de Tikhonov combinée au débruitage BM3D (Dabov et al., 2007) et le réseau proposé Section F.6.2 (entraîné avec $\mu_\alpha = 50$ photons et $\sigma_\alpha = 0,5\mu_\alpha$).

Conclusion

Nous présentons une méthode d'apprentissage profond pour récupérer une image dans l'imagerie à un seul pixel en introduisant une cartographie des données brutes dans le domaine de l'image. Ce mappage est implémenté comme la première couche d'un réseau neuronal, qui fournit une reconstruction approximative d'une cascade traditionnelle de couches convolutionnelles. Nous décrivons également un cadre permettant d'entraîner un réseau en utilisant uniquement des données simulées. Nous décrivons comment exploiter tous les paramètres expérimentaux pour traiter les données expérimentales, et aussi comment normaliser la mesure, ce qui est fondamental lorsqu'on envisage un reconstructeur non linéaire (par exemple, un réseau neuronal).

Bien que notre réseau soit entraîné en utilisant uniquement des simulations, il est très performant sur les données expérimentales, même lorsque le niveau de bruit est inconnu. L'estimation de la covariance du bruit couplée à un processus d'entraînement approprié semble être efficace dans une grande variété de scénarios. Ceci est particulièrement intéressant dans les configurations en temps réel où il n'est pas possible d'évaluer de nombreux reconstructeurs différents. Dans les travaux futurs, nous explorerons des architectures de réseaux interprétables plus complexes.

F.7 Réseau de complétion 3D pour la reconstruction profonde d'images hyperspectrales mono-pixel

L'imagerie hyperspectrale présente un intérêt pour les applications biomédicales. La caméra mono-pixel peut être utilisée comme un imageur hyperspectral en focalisant la lumière sur une fibre optique reliée à un spectromètre. Cette caméra hyperspectrale mono-pixel hérite de la résolution spectrale d'un spectromètre. En ajoutant une dimension supplémentaire aux données d'entrée et en exploitant les corrélations à travers cette dimension, nous pouvons régulariser les images hyperspectrales pour améliorer la qualité de la reconstruction.

Ce chapitre rend compte de nos expériences avec des réseaux neuronaux qui utilisent des convolutions de niveau supérieur pour reconstruire des images hyperspectrales à partir de données provenant d'une caméra hyperspectrale à un seul pixel. Les éléments de ce chapitre ont été réalisés en collaboration avec V. Pronina de l'Institut des sciences et technologies de Skolkovo et ont été soumis à *Optics Express*.

abstract

L'imagerie mono-pixel acquiert une image en mesurant ses coefficients dans un domaine de transformation, grâce à un modulateur spatial de lumière. Cependant, comme les mesures sont séquentielles, seuls quelques coefficients peuvent être mesurés dans les applications en temps réel. Par conséquent, la reconstruction d'un seul pixel est généralement un problème inverse sous-déterminé qui nécessite une régularisation pour obtenir une solution appropriée.

Combiné à un détecteur spectral, le concept d'imagerie mono-pixel permet de réaliser une imagerie hyperspectrale. Alors que chaque canal peut être reconstruit indépendamment, nous proposons d'exploiter la redondance spectrale entre les canaux pour régulariser le problème de reconstruction. En particulier, nous introduisons un réseau de complétion déboisé qui inclut des filtres de convolution 3D. Contrairement aux approches de type boîte noire, notre réseau combine la théorie classique de Tikhonov avec la méthodologie de l'apprentissage profond, ce qui conduit à un réseau explicable.

En considérant à la fois des données simulées et expérimentales, nous démontrons que l'approche proposée produit des images hyperspectrales avec des rapports signal-bruit de pointe et une similarité structurelle plus élevés que les approches développées pour les images en niveaux de gris.

F.7.1 Introduction

L'imagerie mono-pixel est une technique de calcul qui permet de reconstruire une image à partir d'un détecteur à point unique (Gibson, Johnson, and Padgett, 2020). Elle a trouvé des applications en microscopie à fluorescence (Studer et al., 2012), en chirurgie guidée par l'image (Aguénounon et al., 2019), en tomographie optique diffuse (Pian et al., 2017), en imagerie infrarouge à ondes courtes (Zhang et al., 2020), en imagerie à travers des milieux diffusants (Li et al., 2020a). En considérant un détecteur spectral, le concept d'imagerie à un seul pixel s'étend à l'imagerie générique. L'imagerie multispectrale a été démontrée avec une unité dispersive suivie de tubes photomultiplicateurs (Rousset et al., 2018; Tao et al., 2021). L'imagerie hyperspectrale à un seul pixel a été démontrée avec un spectromètre à fibre compact (Peller, Farahi, and Trammell, 2018; Lorente Mur et al., 2020) ou en exploitant un interféromètre de Michelson (Jin et al., 2017).

Les mesures sur un seul pixel peuvent être modélisées comme des produits scalaires entre une image sous-jacente et certains motifs bidimensionnels mis en œuvre par un modulateur spatial de lumière (SLM) (Edgar, Gibson, and Padgett, 2019). Pour limiter les temps d'acquisition, il est hautement souhaitable de réduire le nombre de motifs lumineux, ce qui conduit à un problème inverse mal posé et sous-déterminé. Les approches classiques pour résoudre de tels problèmes sont basées sur la détection compressive (Duarte et al., 2008) ; cependant, l'apprentissage profond (DL) s'est également avéré efficace dans de nombreux problèmes optiques, notamment le débrouillage d'images (Pronina et al., 2020), la reconstruction d'images (Wang et al., 2019), le débruitage d'images (Jiang et al., 2016), et le démêlage spectral et de durée de vie (Yao et al., 2019; Smith, Ochoa, and Intes, 2020). Il n'est pas surprenant que l'imagerie mono-pixel ait également bénéficié de la DL. Dans (Higham et al., 2018), un réseau auto-encodeur convolutif profond s'est avéré plus performant que la détection comprimée dans une tâche de reconstruction d'image. Contrairement à la conception empirique du réseau dans (Higham et al., 2018), (Ducros, Lorente Mur, and Peyrin, 2020) a introduit une architecture où la première couche a été montrée pour estimer l'espérance conditionnelle de l'image compte tenu des mesures sans bruit. Dans la (Lorente Mur et al., 2021), le réseau a été généralisé au cas de données bruyantes et à différents niveaux de bruit. Cependant, alors que les autres approches hyperspectrales computationnelles (*e.g.*, CASSI (Wang et al., 2019) ou CS MUSI (Gedalin, Oiknine, and Stern, 2019)) exploitent des réseaux convolutifs 3D complets pour engager la dimension hyperspectrale, les réseaux de reconstruction *simple pixel* existants sont toujours limités à des convolutions 2D, le problème de la reconstruction hyperspectrale attendant toujours d'être abordé.

Contribution

Dans cet article, nous proposons d'abord de généraliser le réseau de reconstruction proposé dans (Lorente Mur et al., 2021) pour l'imagerie en niveaux de gris à un seul pixel au problème de l'imagerie hyperspectrale à un seul pixel. En particulier, nous considérons des convolutions d'ordre supérieur pour apprendre la régularisation à travers la dimension hyperspectrale. À notre connaissance, il s'agit du premier effort

de développement d'un reconstructeur 3D pour l'imagerie mono-pixel hyperspectrale. Nos réseaux combinent le schéma d'optimisation classique pour le débruitage dans le domaine de la mesure avec l'approche d'apprentissage profond pour la projection du domaine de la mesure au domaine de l'image et une régularisation finale de la solution dans le domaine de l'image. Pour entraîner notre réseau, nous créons un ensemble de données HSI à partir de l'ensemble des images RVB. Nous validons notre approche sur des données hyperspectrales simulées et expérimentales. Si ce manuscrit est accepté, notre réseau de reconstruction hyperspectrale sera intégré dans le paquetage open source `spyrit` (Lorente Mur and Ducros, 2020).

F.7.2 Algorithme de reconstruction hyperspectrale par apprentissage profond proposé

Dans le cas spécifique de l'imagerie hyperspectrale, le processus de formation de l'image (sans bruit) peut être écrit comme suit

$$\boldsymbol{m}_\lambda^\alpha = \alpha \boldsymbol{H}_1 \boldsymbol{f}_\lambda, \quad 1 \leq \lambda \leq \Lambda, \tag{F.51}$$

où $\boldsymbol{m}_\lambda$ et $\boldsymbol{f}_\lambda$ sont le λ-ième canal d'une mesure et son image correspondante, respectivement. Nous désignons également le vecteur de mesures complet par $\boldsymbol{m} = [\boldsymbol{m}_1^\top \dots \boldsymbol{m}_\Lambda^\top]^\top$ et l'image hyperspectrale complète par $\boldsymbol{f} = [\boldsymbol{f}_1^\top \boldsymbol{f}_\Lambda^\top]^\top$.

Dans les sections suivantes, nous explorons les approches de reconstruction de l'hyperspectral $\boldsymbol{f}$ à partir du vecteur de mesure $\boldsymbol{m}$. L'architecture finale du réseau d'achèvement de débruitage 3D proposé pour la reconstruction d'images hyperspectrales mono-pixel est présentée à la Fig. F.11a.

F.7.2.1 Réseaux de complétion débruitée en 2D

Une approche naïve consiste à traiter chaque canal spectral indépendamment, ce qui peut être résumé par

$$\tilde{\boldsymbol{f}}_\lambda = \mathcal{G}_{\boldsymbol{\theta}}^1(\boldsymbol{m}_\lambda^\alpha), \qquad 1 \leq \lambda \leq \Lambda, \tag{F.52a}$$

$$\hat{\boldsymbol{f}}_\lambda = (\mathcal{G}_{\boldsymbol{\theta}}^L \circ \dots \circ \mathcal{G}_{\boldsymbol{\theta}}^2)(\tilde{\boldsymbol{f}}_\lambda), \qquad 1 \leq \lambda \leq \Lambda, \tag{F.52b}$$

où $\mathcal{G}^1$ met en œuvre la solution de complétion débruitée donnée par (F.49). Notez que chaque canal spectral est traité indépendamment. En particulier, les couches de $(\mathcal{G}_{\boldsymbol{\theta}}^L \circ \dots \circ \mathcal{G}_{\boldsymbol{\theta}}^2)$ reposent sur des noyaux de convolution 2D qui agissent uniquement dans le domaine spatial et qui sont entraînés à l'aide d'images en niveaux de gris.

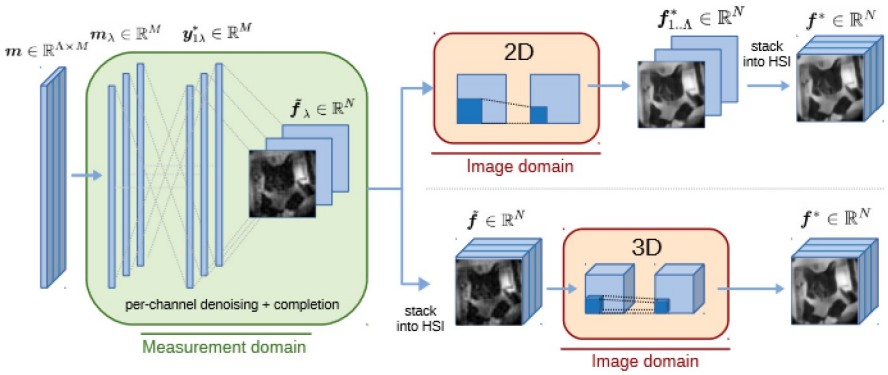
F.7.3 Réseaux de complétion débruitée 3D

Pour exploiter la corrélation entre les canaux spectraux d'une image hyperspectrale, nous introduisons un réseau neuronal 3D. Après la reconstruction par canal en utilisant Eq.F.49, nous introduisons des convolutions d'ordre supérieur dans le domaine de l'image.

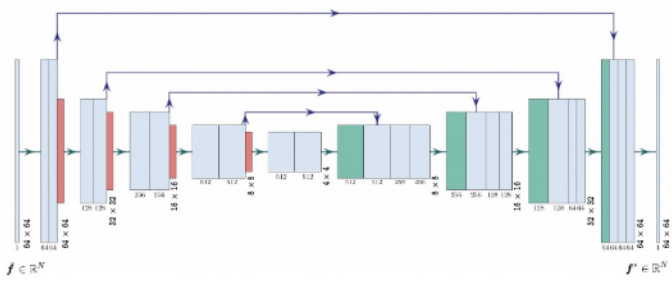
$$\tilde{\boldsymbol{f}}_\lambda = \mathcal{G}_{\boldsymbol{\theta}}^1(\boldsymbol{m}_\lambda^\alpha), \quad 1 \leq \lambda \leq \Lambda, \tag{F.53a}$$

$$\hat{\boldsymbol{f}} = \mathcal{G}_{\boldsymbol{\theta}}(\tilde{\boldsymbol{f}}), \tag{F.53b}$$

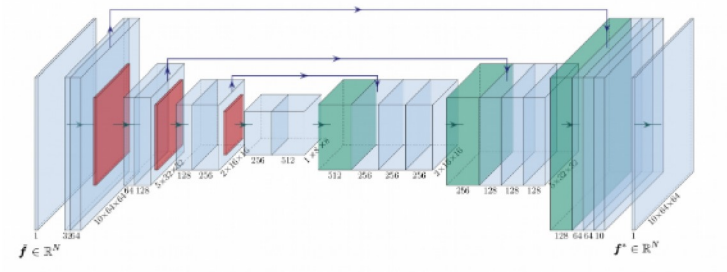
où les couches apprenantes de $\mathcal{G}_{\boldsymbol{\theta}}$ agissent sur l'ensemble des trois dimensions d'un hypercube, régularisant à la fois les dimensions spatiale et spectrale contrairement au réseau 2D.



(A)



(B)

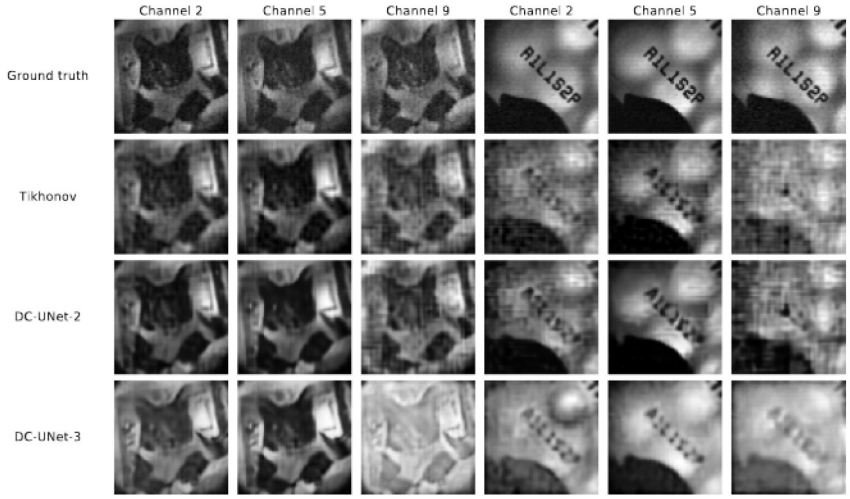


(C)

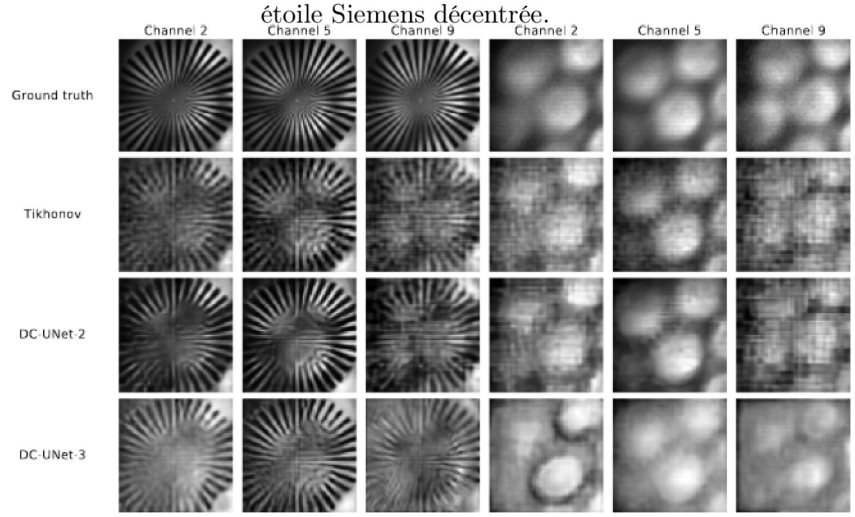
FIGURE F.11 – Overview of the deep reconstruction networks. (a) Architecture of the 2D and 3D denoised completion networks (DC-Net). Denoising and completion occur in the measurement domain, for all channels independently in both the 2D and 3D cases. In the 2D case (top part of the diagram), the learnable layers ($\mathcal{G}_{\boldsymbol{\theta}}^{L} \circ \ldots \circ \mathcal{G}_{\boldsymbol{\theta}}^{2}$) correspond to 2D UNet; in the 3D case (bottom part of the diagram), to a 3D UNet. (b) Architecture of the 2D UNet. (c) Architecture of the 3D UNet.

F.7.4 Résultats sur données expérimentales

Nous observons dans la figure F.12 que DC-UNet-2 et DC-UNet-3 fournissent des images de sortie plus lisses et moins bruyantes que la méthode de Tikhonov, grâce au réseau de correction des artefacts agissant dans le domaine de l'image. Malgré la persistance de certains artefacts, DC-UNet-3 permet de restaurer des détails d'image plus nets dans tous les canaux spectraux. Ceci est particulièrement visible sur les canaux limites : le visage du chat STL-10 et les lettres de l'étoile excentrée de Siemens sont plus proches de ceux des images de référence. Ce résultat est obtenu grâce à la nature de la régularisation spatiale-spectrale du réseau 3D de post-traitement.



(a) Reconstruction de deux images expérimentales - chat STL-10 et cible de résolution en étoile Siemens décentrée.



(b) Reconstruction de deux images expérimentales - cible étoilée Siemens centrée et lampe LED.

FIGURE F.12 – Résultats de la reconstruction de quatre images hyperspectrales expérimentales. Dans les deux sous-figures, la ligne supérieure affiche les images de vérité du sol pour les 2e, 5e et 9e canaux ; la deuxième ligne affiche les résultats du débruitage et de l'achèvement avec la solution de régularisation généralisée de Tikhonov ; la troisième ligne affiche la reconstruction avec le réseau d'achèvement déboisé avec la régularisation 2D UNet ; la ligne inférieure affiche la reconstruction avec le réseau d'achèvement déboisé avec la régularisation 3D UNet.

DC-UNet-3 obtient des résultats supérieurs en termes de réduction du bruit par rapport à Tikhonov et une meilleure préservation des détails par rapport à DC-UNet-2. Nous observons que la prise en compte de la corrélation spectrale entre les canaux adjacents permet de récupérer des canaux qui contiennent initialement moins d'informations. Ceci est clairement visible sur la cible excentrée à résolution étoilée de Siemens, où l'inscription est préservée dans tous les canaux spectraux, contrairement aux résultats obtenus avec Tikhonov et DC-UNet-2.

La Fig. F.13 présente la reconstruction de la cible étoilée Siemens en couleur à l'aide des méthodes Tikhonov, DC-UNet-2 et DC-UNet-3. La solution de Tikhonov présente certains artefacts tels que les structures carrées à la frontière des branches qui

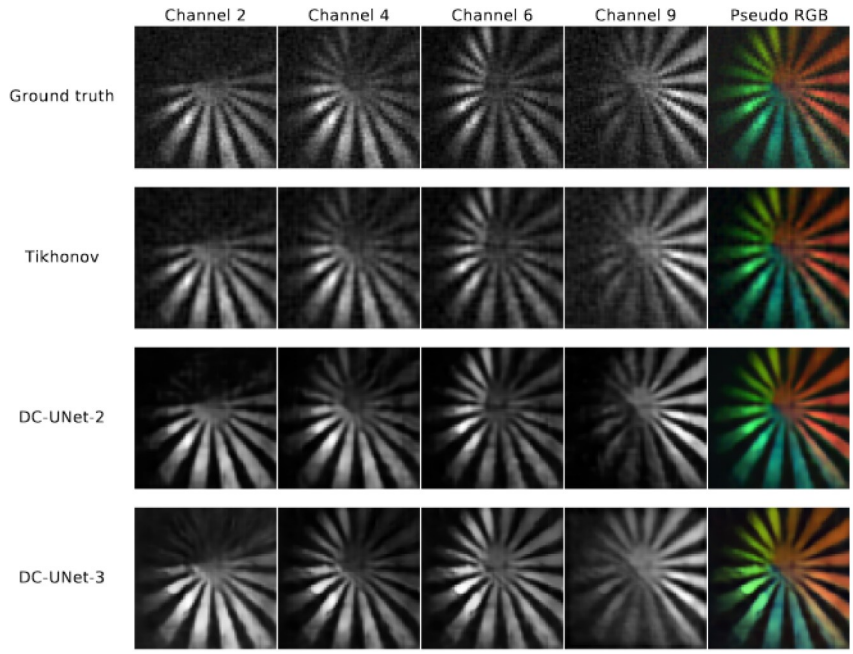


FIGURE F.13 – Reconstruction de la cible Siemens couleur expérimentale. Rangée supérieure : images de vérité du sol dans les 2e, 4e, 6e et 9e canaux. Deuxième rangée : Solution de Tikhonov. Troisième ligne : Solution DC-Net utilisant un UNet 2D. Rangée du bas : Solution DC-Net utilisant un UNet 3D. La colonne de droite représente l'image pseudo-couleur calculée avec le résultat de la récupération de la méthode correspondante selon la méthode (Magnusson et al., 2020).

proviennent du sous-échantillonnage dans la base Hadamard, ou l'aspect granuleux du bruit basé sur les photons, tandis que la reconstruction avec DC-UNet-2 et DC-UNet-3 donne des images plus lisses. Comparé à DC-UNet-2, DC-UNet-3 est capable de conserver certains détails qui sont perdus lors de la suppression des artefacts dans certains cas. Par exemple, ceci peut être observé dans les canaux 4 et 6, où les branches supérieures de l'étoile ont été effacées et ont perdu leur alignement lors de la reconstruction avec DC-UNet-2, mais sont intactes après la récupération avec DC-UNet-3. De plus, dans le 'Canal 9', DC-UNet-2 a presque effacé les branches inférieures gauches, contrairement aux résultats fournis par DC-UNet3. Cela semble indiquer que DC-UNet-3 exploite la redondance spectrale des canaux. Pour obtenir une comparaison visuelle des images hyperspectrales, nous calculons également une image pseudo couleur comme décrit dans (Magnusson et al., 2020). En comparant les images pseudo-couleur dans la colonne de droite de la Fig.F.13, on peut voir que les trois méthodes réussissent à rendre les couleurs dans une certaine mesure. Cependant, toutes les méthodes semblent donner un léger décalage dans le spectre des couleurs. Les branches rouges reconstruites via Tikhonov et DC-UNet-2 semblent être biaisées vers les longueurs d'onde rouges, tandis que la récupération avec DC-UNet-3 entraîne un décalage des branches rouges vers les longueurs d'onde orange. Une situation similaire est observée dans les branches bleues de l'étoile, où les applications Tikhonov et DC-UNet-2 entraînent un décalage vers les longueurs d'onde bleues inférieures, tandis que la reconstruction via DC-UNet-3 décale les branches vers les longueurs d'onde bleues supérieures ou vertes inférieures. Cependant, en termes de résolution spatiale, les branches produites par DC-UNet-3D semblent avoir une résolution plus élevée, en particulier dans les branches supérieures gauches.

Conclusion

Nous avons proposé le premier réseau neuronal profond pour la reconstruction d'images hyperspectrales à partir d'un petit nombre de mesures bruitées d'un seul pixel. Les couches du réseau proposé combinent la solution classique de Tikhonov avec des filtres de convolution apprenables. Pour exploiter la corrélation entre les canaux spectraux, il considère les filtres qui agissent à la fois dans les dimensions spatiale et spectrale. Nos expériences montrent qu'un réseau UNet 3D améliore la qualité de la reconstruction par rapport à un réseau UNet 2D qui agit uniquement dans le domaine de l'image. Le réseau 3D proposé démontre une performance supérieure en termes de métriques quantitatives ainsi que visuellement. Ceci est particulièrement évident dans les canaux à faible nombre de photons, *i.e.*, avec un bruit écrasant. Dans les travaux futurs, nous prévoyons de considérer les discontinuités spectrales, les dimensions supplémentaires et d'autres architectures du réseau de base.

F.8 Espérance-maximisation par apprentissage profond pour la reconstruction d'images mono-pixel avec un bruit dépendant du signal

Les architectures d'apprentissage profond basées sur des modèles pour la reconstruction d'images sont une famille de méthodes d'apprentissage profond dont la structure émule les schémas d'optimisation MAP classiques. Ces architectures utilisent la connaissance de l'opérateur direct et du modèle de bruit pour alterner entre une couche de cohérence des données et une couche de débruitage d'image à image. La couche de cohérence des données assure la cohérence de l'image reconstruite par rapport à l'opérateur direct et aux niveaux de bruit, tandis que la couche de débruitage image à image assure la cohérence de l'image reconstruite par rapport à l'opérateur direct et aux niveaux de bruit.

Dans ce chapitre, nous avons proposé une architecture d'apprentissage profond basée sur l'algorithme d'expectation-maximisation. Cette architecture a été conçue pour s'attaquer aux problèmes d'imagerie à un seul pixel où les données sont corrompues par un bruit mixte Skellam-Gaussien. Le contenu de ce chapitre est basé sur un compte-rendu de conférence de l'IEEE ISBI 2020 (Lorente Mur et al., 2021). En s'appuyant sur cette conférence, le contenu de ce chapitre a été soumis à *IEEE Transactions on computational imaging.*

F.8.1 abstract

La reconstruction d'images à partir d'une séquence de quelques mesures linéaires qui sont corrompues par un bruit normalement distribué dépendant du signal est un problème inverse avec de nombreuses applications d'imagerie biomédicale telles que la tomographie informatisée, la tomographie par émission de positrons et la microscopie optique. Dans cette étude, nous nous concentrons sur l'imagerie à un seul pixel, où le dispositif acquiert une transformée Hadamard sous-échantillonnée de l'image de la scène.

L'apprentissage profond est un cadre très efficace pour résoudre les problèmes inverses en imagerie. Plusieurs architectures de réseaux neuronaux permettent de faire le lien entre les méthodes de reconstruction d'images basées sur l'apprentissage profond et l'optimisation. Ces méthodes d'apprentissage profond, qui reposent sur un opérateur direct, conduisent à des réseaux plus faciles à interpréter. Nous proposons

ici une nouvelle architecture de réseau obtenue en déroulant l'algorithme d'espérance-maximisation. En particulier, nous calculons l'estimation maximale a posteriori de l'image inconnue compte tenu des mesures corrompues par un bruit dépendant du signal normalement distribué. Nous montrons que le soi-disant EM-Net s'applique aux modèles de bruit mixtes Skellam-Gaussien qui sont courants dans l'imagerie à un seul pixel.

Nous présentons des résultats de reconstruction à partir d'acquisitions simulées et expérimentales d'un seul pixel. Nous montrons que l'EM-Net se généralise très bien aux niveaux de bruit non vus pendant la formation, malgré le fait qu'il a moins de paramètres appris que les autres méthodes. Le réseau EM proposé reconstruit généralement des images avec moins d'artefacts et des rapports signal/bruit plus élevés, en particulier dans des situations de bruit élevé.

F.8.2 Introduction

L'imagerie monopixel est une configuration d'imagerie informatique dans laquelle un détecteur à point unique est utilisé pour récupérer une image (Duarte et al., 2008). Elle a été appliquée avec succès à la microscopie à fluorescence (Studer et al., 2012), à l'imagerie hyperspectrale (Rousset et al., 2018; Arce et al., 2014), à la tomographie optique diffuse (Pian et al., 2017), à la chirurgie guidée par l'image (Aguénounon et al., 2019), à l'imagerie infrarouge à ondes courtes (Zhang et al., 2020). Une caméra monopixel acquiert le produit scalaire entre l'image d'une scène et certains motifs lumineux bidimensionnels qui sont affichés séquentiellement à l'aide d'un modulateur de lumière spatial (Edgar, Gibson, and Padgett, 2018). L'image de la scène est ensuite reconstruite à partir des mesures brutes. Pour limiter les temps d'acquisition, il est fortement souhaitable de réduire le nombre de motifs lumineux, ce qui conduit à un problème inverse sous-déterminé.

Les récentes avancées en apprentissage profond ont révolutionné la reconstruction d'images (Wang et al., 2018; Arridge et al., 2019). En particulier, les réseaux de neurones convolutifs (CNN) se sont révélés très efficaces pour résoudre le problème de la tomographie assistée par ordinateur, soit par l'apprentissage d'une cartographie inverse directe (McCann, Jin, and Unser, 2017), soit par l'utilisation de réseaux de neurones adversaires (Kang et al., 2019). De nombreux efforts sont consacrés à combler le fossé entre les approches plus traditionnelles basées sur des modèles pour la reconstruction d'images et les approches basées sur l'apprentissage profond pour les problèmes inverses. Les CNN peuvent fournir des transformées de sparification (Chun and Fessler, 2020), modéliser le collecteur des images naturelles (Aggarwal, Mani, and Jacob, 2019), ou un projecteur sur cet espace (Gupta et al., 2018). Les prieurs CNN sont ensuite insérés dans un algorithme d'optimisation non roulé (Adler and Öktem, 2018; Reader et al., 2021) ou utilisés comme dans des expansions, comme les séries de Neumann (Gilton, Ongie, and Willett, 2020).

Cette tendance profite aussi grandement à l'optique computationnelle (Barbastathis, Ozcan, and Situ, 2019; Kellman et al., 2019). Dans l'imagerie à un seul pixel, en particulier, une cartographie entièrement apprise de l'espace de mesure à l'espace image peut surpasser les reconstructions par détection compressée (Higham et al., 2018). Dans (Dave et al., 2019), les auteurs utilisent un modèle autorégressif profond qui est branché dans un algorithme de méthode des multiplicateurs à direction alternée pour les images à un seul pixel. Dans (Ducros, Lorente Mur, and Peyrin, 2020), nous proposons un CNN de reconstruction simple où la première couche calcule l'espérance conditionnelle de l'image inconnue à partir de mesures sans bruit. Nous généralisons cette idée au cas de données bruyantes avec des niveaux de bruit variables dans

(Lorente Mur et al., 2021), tandis que nous introduisons une architecture itérative dans (Lorente Mur et al., 2021).

Les mesures à un seul pixel résultent de la différence de deux variables de Poisson. Par conséquent, les données à un seul pixel sont corrompues par un bruit mixte Skellam-Gaussien, qui n'a pas d'expression sous forme fermée. Pour des problèmes similaires tels que le débruitage mixte poissonnien-gaussien, cette difficulté est contournée en utilisant des transformées stabilisatrices de variance telles que la transformée d'Anscombe généralisée (Anscombe, 1948) avant de traiter le bruit gaussien (Murtagh, Starck, and Bijaoui, 1995). Cependant, cela introduit un biais (Makitalo and Foi, 2011). D'autres approches reposent sur des approximations. Dans (Benvenuto et al., 2008; Chouzenoux et al., 2015) et (Ghulyani and Arigovindan, 2021), la vraisemblance de Poisson-Gaussien mixte est approximée comme une somme finie. L'estimateur sans biais du risque de Poisson-Gaussien - expansion linéaire des seuils (Li, Luisier, and Blu, 2018) cherche à approximer une estimation sans biais de l'erreur quadratique moyenne avec des combinaisons linéaires de filtres de Wiener suivies d'un seuillage des coefficients d'ondelettes de Haar. D'autres approches simplifient le problème soit en approximant la contribution de Poisson par un bruit dépendant du signal normalement distribué (Li et al., 2015), soit en négligeant la distribution normale. On peut alors calculer le maximum *a posteriori* en utilisant l'algorithme de maximisation de l'espérance (Dempster, Laird, and Rubin, 1977). Cependant, le problème d'optimisation qui en résulte est généralement résolu à l'aide de prieurs élaborés à la main (par exemple, pénalisation basée sur la hessienne ou la variation totale).

Contribution

Dans cet article, nous dérivons et analysons une architecture de réseau neuronal profond basée sur l'algorithme d'expectation-maximisation. Ce réseau, appelé EM-Net, est capable de résoudre des problèmes inverses avec un bruit dépendant du signal, tels que ceux impliqués dans l'optique computationnelle. Nous étendons de manière significative nos résultats préliminaires (Lorente Mur et al., 2021) dans trois directions. Premièrement, nous modélisons la reconstruction d'un seul pixel comme un problème de reconstruction mixte Skellam-Gaussien, pour lequel nous dérivons la vraisemblance correspondante. Comme il est impossible de résoudre ce problème numériquement, nous proposons une approximation normale qui conduit à une covariance dépendant du signal. Ensuite, nous proposons un algorithme itératif en quatre étapes dans lequel la priorité de l'image est apprise à partir d'exemples d'entraînement et nous montrons comment réduire le nombre de paramètres apprenables, conformément à la théorie EM. Nous proposons également une stratégie d'apprentissage où la plupart des paramètres de notre réseau EM peuvent être précalculés. Troisièmement, nous effectuons des comparaisons numériques approfondies avec des méthodes de reconstruction basées sur les données, conçues pour l'imagerie à un seul pixel et d'autres modalités. Comme le niveau de bruit est généralement inconnu dans les expériences pratiques, nous nous concentrons sur la robustesse des méthodes en présence d'un bruit dont le niveau diffère de celui utilisé pendant l'apprentissage. Nous considérons à la fois des mesures simulées corrompues par un bruit mixte Skellam-Gaussien et des données expérimentales.

Après acceptation, notre méthode de reconstruction sera mise à disposition dans la boîte à outils Python SPyRiT(Lorente Mur and Ducros, 2020).

F.8.3 Imagerie mono-pixel compressive

F.8.3.1 Imagerie mono-pixel

Soit $\boldsymbol{f} \in [0,1]^N$ l'image à acquérir. L'idée principale de l'imagerie monopixel compressive est de mesurer $\boldsymbol{y} = \boldsymbol{H}\boldsymbol{f}$ en utilisant le matériel et de récupérer $\boldsymbol{f}$ en utilisant le logiciel. La matrice $\boldsymbol{H} = [\boldsymbol{h}_1, \ldots, \boldsymbol{h}_N] \in \mathbb{R}^{N\times N}$ rassemble les motifs qui sont chargés séquentiellement dans un modulateur spatial de lumière. Ces motifs peuvent être choisis comme matrice de base, comme les bases de Fourier, de cosinus discret, d'ondelettes et de Hadamard (Ochoa et al., 2018). En pratique, seuls quelques motifs d'une base sont acquis afin d'accélérer les temps d'acquisition. De plus, comme le modulateur spatial de lumière ne peut pas implémenter de valeurs négatives, les motifs $\boldsymbol{H}$ sont divisés en motifs positifs et négatifs $\boldsymbol{H}^+$ et $\boldsymbol{H}^-$, de sorte que $(\boldsymbol{H}^+)_{i,n} = \max((\boldsymbol{H})_{i,n}, 0)$ et $(\boldsymbol{H}^+)_{i,n} = \max(-(\boldsymbol{H})_{i,n}, 0)$ (voir (Lorente Mur et al., 2019) pour plus de détails).

Nous modélisons la mesure brute comme un bruit mixte Poisson-Gaussien (Foi et al., 2008)

$$\hat{\mathbf{m}}^{\alpha}_{+,-} \sim K\,\mathcal{P}(\alpha \boldsymbol{S}\boldsymbol{H}^{+,-}\boldsymbol{f}) + \mathcal{N}(\mu_{\text{dark}}, \sigma^2_{\text{dark}}) \tag{F.54}$$

où $\mathcal{P}$ et $\mathcal{N}$ sont les distributions de Poisson et Gaussienne, $\boldsymbol{S} = [\boldsymbol{I}_M, \boldsymbol{0}] \in \mathbb{R}^{M\times N}$ est une matrice de sous-échantillonnage (où $M \leq N$), K est une constante qui représente le gain global du système (en nombre de comptes/électron), α est l'intensité (en photons) de l'image, μ_{dark} est le courant d'obscurité (en nombre de comptes), et σ_{dark} est le bruit d'obscurité (en nombre de comptes). Nous supposons en outre que μ_{dark} et σ_{dark} sont indépendants de l'intensité de l'image α.

Les mesures brutes sont enfin combinées et normalisées. Cette étape standard préposée est

$$\mathbf{m}^{\alpha} = (\hat{\mathbf{m}}^{\alpha}_{+} - \hat{\mathbf{m}}^{\alpha}_{-})/(\alpha K). \tag{F.55}$$

L'insertion du modèle de bruit des mesures brutes donné par (F.56) dans l'équation précédente conduit au modèle de bruit mixte Skellam-Gaussien

$$\mathbf{m}^{\alpha} \sim \frac{1}{\alpha}\,\mathcal{S}_{\text{k}}(\alpha \boldsymbol{S}\boldsymbol{H}^{+}\boldsymbol{f}, \alpha \boldsymbol{S}\boldsymbol{H}^{-}\boldsymbol{f}) + \frac{2}{\alpha K}\,\mathcal{N}(0, \sigma^2_{\text{dark}}), \tag{F.56}$$

où la distribution de Skellam (Skellam, 1946) $\mathcal{S}_{\text{k}}(\boldsymbol{\mu}_1, \boldsymbol{\mu}_2)$ est la distribution de la différence de deux variables distribuées par Poisson avec une moyenne de $\boldsymbol{\mu}_1$ et $\boldsymbol{\mu}_2$. L'espérance des mesures est $\mathbf{m}^{\alpha} = \boldsymbol{S}\boldsymbol{H}\boldsymbol{f}$.

F.8.3.2 Approximation normale du modèle de bruit mixte Skellam-Gaussien

La vraisemblance du modèle de bruit mixte Skellam-Gaussien implique une somme infinie d'exponentielles multipliées par des fonctions de Bessel modifiées. L'exploitation de cette expression analytique est numériquement intraitable car elle implique une série infinie et l'évaluation des fonctions de Bessel.

$$p(\boldsymbol{m}|\boldsymbol{f}) \propto \prod_{i=1}^{M} \sum_{n\in\mathbb{N}} e^{-\alpha(\boldsymbol{h}_{i,+}^{\top}\boldsymbol{f} + \boldsymbol{h}_{i,-}^{\top}\boldsymbol{f})} \left(\frac{\boldsymbol{h}_{i,+}^{\top}\boldsymbol{f}}{\boldsymbol{h}_{i,-}^{\top}\boldsymbol{f}}\right)^{\frac{n-\alpha\boldsymbol{h}_i^{\top}\boldsymbol{f}}{2\alpha}} \mathcal{I}_{|\frac{n}{\alpha}-\boldsymbol{h}^{\top}\boldsymbol{f}|}\left(2\alpha\sqrt{(\boldsymbol{h}_{i,+}^{\top}\boldsymbol{f})(\boldsymbol{h}_{i,-}^{\top}\boldsymbol{f})}\right) e^{-\frac{(\boldsymbol{m}_i - \frac{n}{\alpha})^2}{\beta^2}}. \tag{F.57}$$

Comme pour les travaux traitant du bruit mixte Poisson-Gaussien, nous proposons d'introduire une approximation. Comme les distributions gaussiennes sont une bonne approximation des distributions de Poisson avec des moyennes supérieures à 5 (voir (Huang, 2001)) –les mesures sur un seul pixel sont généralement plusieurs ordres de grandeur au-dessus de cette valeur–, nous utilisons l'approximation normale des distributions de Poisson dans (F.56). Nous obtenons

$$\mathbf{m}^\alpha = \boldsymbol{SHf} + \boldsymbol{\epsilon}^\alpha, \text{with} \quad \boldsymbol{\epsilon}^\alpha \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_\alpha) \tag{F.58}$$

où la covariance du bruit $\boldsymbol{\Sigma}_\alpha$ est donnée par

$$\boldsymbol{\Sigma}_\alpha = \mathsf{Diag}\left(\boldsymbol{\sigma}_\alpha^2\right) = \mathsf{Diag}\left(\frac{1}{\alpha}\boldsymbol{H}_1^+\boldsymbol{f} + \frac{1}{\alpha}\boldsymbol{H}_1^-\boldsymbol{f}\right) + \frac{2\sigma_{\text{dark}}^2}{K^2\alpha^2}\boldsymbol{I}_M. \tag{F.59}$$

Notez que la variance du bruit $\boldsymbol{\sigma}_\alpha^2$ est inconnue car elle dépend à la fois de l'image inconnue $\boldsymbol{f}$ et de l'intensité inconnue α. Par conséquent, comme dans (Lorente Mur et al., 2021), nous exploitons la mesure brute pour estimer $\boldsymbol{\sigma}_\alpha^2$. Nous remarquons d'abord à partir de (F.56) que la variance et l'espérance de la mesure brute sont liées par $\mathbb{V}\text{ar}\left(\hat{\mathbf{m}}_{+,-}^\alpha\right) = K\,\mathbb{E}\left(\hat{\mathbf{m}}\right)_{+,-}^\alpha - K\mu_{\text{dark}} + \sigma_{\text{dark}}^2$. Ensuite, nous approximons l'espérance inconnue par l'échantillon bruité, c'est-à-dire, $\mathbb{E}\left(\hat{\mathbf{m}}_{+,-}^\alpha\right) \approx \hat{\boldsymbol{m}}_{+,-}^\alpha$, ce qui conduit à la variance suivante pour les mesures pré-traitées

$$\boldsymbol{\sigma}_\alpha^2 \approx \tilde{\boldsymbol{\sigma}}_\alpha^2 = \frac{1}{\alpha^2 K}(\hat{\boldsymbol{m}}_+^\alpha + \hat{\boldsymbol{m}}_-^\alpha - 2\mu_{\text{dark}}\mathbf{1}) + \frac{2\sigma_{\text{dark}}^2}{K^2\alpha^2}\mathbf{1}. \tag{F.60}$$

F.8.4 Méthode proposée

Nous adoptons un cadre bayésien et reconstruisons $\boldsymbol{f}$ à partir de $\boldsymbol{m}^\alpha$ en calculant un estimateur ponctuel de $p(\boldsymbol{f}|\boldsymbol{m}^\alpha)$. En particulier, nous cherchons à calculer la solution a posteriori maximale qui résout les problèmes suivants

$$\underset{\boldsymbol{f}}{\text{argmax}} \quad \log p(\boldsymbol{m}^\alpha|\boldsymbol{f}) + \log p(\boldsymbol{f}), \tag{F.61}$$

où nous supposons la fonction de densité de probabilité $p(\boldsymbol{m}^\alpha|\boldsymbol{f}) \propto \exp\left(-\frac{1}{2}\|\boldsymbol{SH})\boldsymbol{f} - \boldsymbol{m}^\alpha\|^2_{\boldsymbol{\Sigma}_\alpha^{-1}}\right)$ selon (F.58), tandis que $p(\boldsymbol{f})$ est une fonction de densité de probabilité inconnue.

L'algorithme EM (Dempster, Laird, and Rubin, 1977) a été couramment utilisé pour estimer la MAP pour les tâches de reconstruction d'images (Zhou et al., 2007). Il s'agit d'un algorithme itératif qui produit une séquence d'estimations $\{\boldsymbol{f}^{(k)}\}$ telle que la séquence $\{\log p(\boldsymbol{f}^{(k)}|\boldsymbol{m}^\alpha)\}_k$ est monotone non décroissante. Chaque itération de l'algorithme EM est basée sur deux étapes : l'étape d'espérance et l'étape de maximisation. L'étape d'espérance calcule l'espérance conditionnelle de la log-vraisemblance de $\mathbf{f}$ par rapport à une variable aléatoire auxiliaire $\mathbf{x}$ étant donné l'estimation actuelle $\boldsymbol{f}^iter \in \mathbb{R}^N$ et les mesures $\boldsymbol{m}^\alpha \in \mathbb{R}^M$.

$$\mathcal{Q}(\boldsymbol{f}|\boldsymbol{f}^{(k)}) = \mathbb{E}_{\mathbf{x}}[\log p(\boldsymbol{x}|\boldsymbol{f})|\boldsymbol{m}^\alpha, \boldsymbol{f}^{(k)}] + p(\boldsymbol{f}). \tag{F.62}$$

Cette espérance conditionnelle est ensuite maximisée par rapport à $\boldsymbol{f}$ pendant l'étape de maximisation, pour produire l'itération suivante

$$\boldsymbol{f}^{(k+1)} = \underset{\boldsymbol{f}}{\text{argmax}}\, \mathcal{Q}(\boldsymbol{f}|\boldsymbol{f}^{(k)}). \tag{F.63}$$

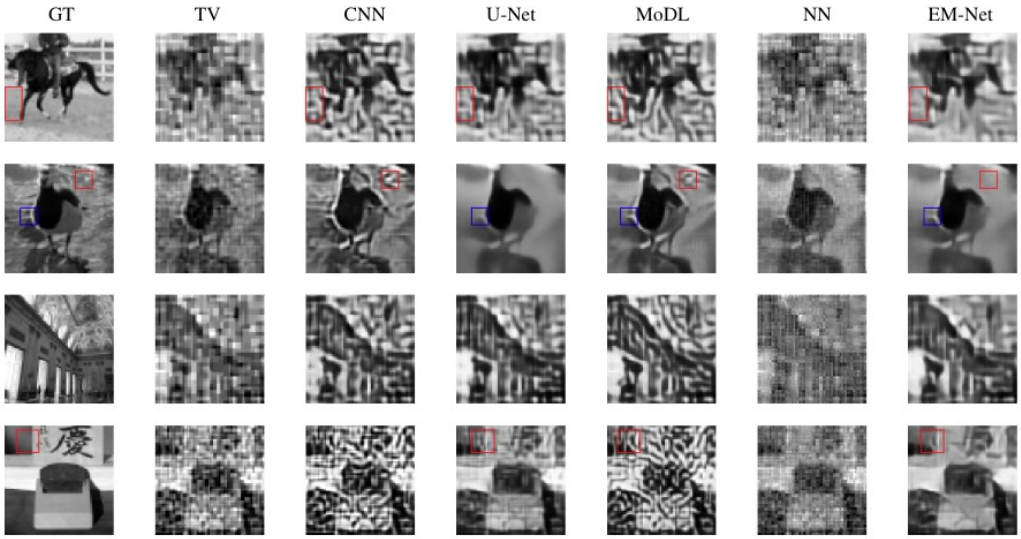


FIGURE F.14 – Reconstructions d'images simulées par les différentes méthodes. (Rangée supérieure) : image STL-10 d'un cheval avec $M = 512$ et $\alpha = 3$ ph. ; (Deuxième rangée) : image STL-10 d'un canard avec $M = 1024$ et $\alpha = 30$ ph. ; (Troisième rangée) : imageNet d'un hall avec $M = 512$ et $\alpha = 2$ ph. ; (Quatrième rangée) : imageNet d'un canapé avec $M = 1024$ et $\alpha = 5$ ph. Les images ont été reconstruites à partir de mesures simulées de l'image de vérité au sol (GT). Les colonnes suivantes montrent des reconstructions utilisant la variation totale ('TV'), le reconstructeur ConvNet -b ('CNN'), un reconstructeur U-Net (Ronneberger et al., 2015) ('U-net'), MoDL (Aggarwal, Mani, and Jacob, 2019) ('MoDL'), et la méthode proposée ('EM-Net').

Pour des lois gaussiennes, ceci se traduit par

$$\bar{\boldsymbol{x}}^{(k)} = \underset{\boldsymbol{x}}{\operatorname{argmin}} \|\boldsymbol{S}\boldsymbol{x} - \boldsymbol{m}^{\alpha}\|^2_{\tilde{\boldsymbol{\Sigma}}_{\alpha}^{-1}} + \|\boldsymbol{x} - \boldsymbol{H}\boldsymbol{f}^{(k)}\|^2_{\boldsymbol{\Sigma}^{-1}} \tag{F.64a}$$

$$\boldsymbol{f}^{(k+1)} = \underset{\boldsymbol{f}}{\operatorname{argmin}} \|\bar{\boldsymbol{x}}^{(k)} - \boldsymbol{H}\boldsymbol{f}\|^2_{\boldsymbol{\Sigma}^{-1}} - \log p(\boldsymbol{f}). \tag{F.64b}$$

Comme $p(\boldsymbol{f})$ est inconnue, nous proposons de remplacer (F.64b) par un modèle non-linéaire $\mathcal{D}_{\boldsymbol{\omega}}$

$$\bar{\boldsymbol{x}}^{(k)} = \underset{\boldsymbol{x}}{\operatorname{argmin}} \|\boldsymbol{S}\boldsymbol{x} - \boldsymbol{m}^{\alpha}\|^2_{\tilde{\boldsymbol{\Sigma}}_{\alpha}^{-1}} + \|\boldsymbol{x} - \boldsymbol{H}\boldsymbol{f}^{(k)}\|^2_{\boldsymbol{\Sigma}^{-1}} \tag{F.65a}$$

$$\boldsymbol{f}^{(k+1)} = \mathcal{D}_{\boldsymbol{\omega}}(\boldsymbol{H}^{\top}\bar{\boldsymbol{x}}^{(k)}) \tag{F.65b}$$

Ceci peut également s'écrire en quatre étapes

$$\boldsymbol{y}_1^{(k)} = \boldsymbol{\sigma}_1^2/(\boldsymbol{\sigma}_1^2 + \tilde{\boldsymbol{\sigma}}_{\alpha}^2)(\boldsymbol{m}^{\alpha} - \boldsymbol{S}\boldsymbol{H}\boldsymbol{f}^{(k)}) \tag{F.66a}$$

$$\boldsymbol{y}_2^{(k)} = \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_1^{-1}\boldsymbol{y}_1^{(k)} \tag{F.66b}$$

$$\bar{\boldsymbol{f}}^{(k)} = \boldsymbol{f}^{(k)} + \boldsymbol{H}^{\top}\boldsymbol{y}^{(k)} \tag{F.66c}$$

$$\boldsymbol{f}^{(k+1)} = \mathcal{D}_{\boldsymbol{\omega}}(\bar{\boldsymbol{f}}^{(k)}), \tag{F.66d}$$

Conclusion

Nous proposons un réseau itératif basé sur l'algorithme EM. Ce réseau EM profond peut résoudre des problèmes inverses linéaires sous-déterminés où les mesures sont

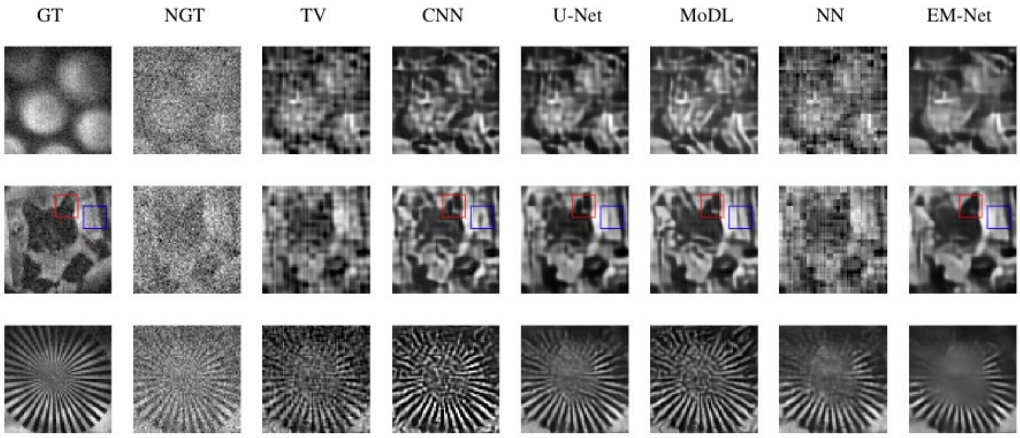


FIGURE F.15 – Reconstructions de trois ensembles de données expérimentales à l'aide de six méthodes différentes. (Rangée supérieure) Lampe LED avec $M = 512$ mesures ; (Deuxième rangée) Chat STL-10 avec $M = 512$ mesures ; (Troisième rangée) Étoile Siemens avec $M = 2048$ mesures. Les images ont été reconstruites à partir d'un ensemble de données entièrement échantillonnées (ground-truth ; GT) acquises avec une intensité d'image élevée (première colonne, $\alpha = 148$ ph., $\alpha = 195$ ph. et $\alpha = 295$ ph.) et d'un ensemble de données entièrement échantillonnées bruitées (noisy ground-truth ; NGT) acquises avec une intensité d'image plus faible (deuxième colonne, $\alpha = 9$ photons, $\alpha = 10$ photons et $\alpha = 14$). Les colonnes suivantes montrent les reconstructions à partir de l'image NGT sous-échantillonnée en utilisant la variation totale ('TV'), le reconstructeur ConvNet -b ('CNN'), un reconstructeur U-Net (Ronneberger et al., 2015) ('U-net'), MoDL (Aggarwal, Mani, and Jacob, 2019) ('MoDL'), et la méthode proposée ('EM-Net').

corrompues par un bruit dépendant du signal normalement distribué (par exemple, un bruit de Poisson, un bruit mixte Poisson-Gaussien, un bruit mixte Skellam-Gaussien). Le réseau EM alterne quatre étapes, qui peuvent être interprétées de manière simple : débruitage des mesures, achèvement des mesures, mise en correspondance des mesures avec le domaine de l'image et débruitage de l'image.

La plupart des réseaux non enroulés supposent un bruit gaussien additif à variance constante. Par conséquent, ils ont tendance à mal se généraliser à différents niveaux de bruit et peuvent être mal adaptés aux applications où le niveau de bruit ne peut être prédit à l'avance. Au contraire, la méthode proposée estime explicitement la covariance du bruit pour débruiter le domaine de mesure. En conséquence, nous constatons que EM-Net se généralise très bien à des niveaux de bruit différents de ceux utilisés pendant l'entraînement. En particulier, il surpasse toutes les méthodes évaluées en présence d'un bruit dont la variance est supérieure à celle utilisée pendant l'entraînement et est comparable à la meilleure méthode en présence d'un bruit dont la variance est inférieure.

F.9 Reconstruction par filtrage de Kalman et échantillonnage adaptatif par apprentissage profond pour la reconstruction vidéo en temps réel

Jusqu'à présent, au cours de cette thèse, nous avons considéré des cas statiques, où nous récupérons l'image courante de la scène en utilisant uniquement les mesures acquises courantes. Bien que ce type d'algorithme puisse être utilisé séquentiellement pour reconstruire des images, les images acquises n'utilisent pas les images acquises précédemment comme information préalable. De plus, cette approche séquentielle ne permettrait pas de réduire le nombre de coefficients acquis tout en gardant à peu près la même qualité d'image. Ce chapitre vise à développer des méthodes permettant de récupérer l'image courante en utilisant les images acquises précédemment. Pour ce faire, nous nous donnons un modèle prédictif pour les vidéos naturelles. Ce modèle prédictif permettra de déterminer l'image antérieure, de déterminer les coefficients que nous devons échantillonner, et de reconstruire notre image courante.

Nous proposons le schéma suivant de sous-échantillonnage adaptatif en temps basé sur la variance.

Algorithm 9 Algorithme de sous-échantillonnage adaptatif en fonction du temps basé sur la variance pour l'imagerie à un seul pixel.

Initialisation : Estimer la première frame $\boldsymbol{f}_0^+ = \mathcal{M}_0(\boldsymbol{m}_0^\alpha)$.
for $t \in \{1, \dots, T\}$ **do**
  1 - Étape de prédiction : Calculer
  $\boldsymbol{f}_t^- = \boldsymbol{f}_{t-1}^+$,
  $\boldsymbol{P}_t^- = \boldsymbol{P}_{t-1}^+ + \boldsymbol{Q}_{t-1}$.
  2 - Choisir les coefficients :
  Qui maximisent $\mathcal{G}_\theta(\boldsymbol{f}_t^-) + (\boldsymbol{f}_t^-)^2$,
  Acquérir le vecteur de mesures correspondant $\boldsymbol{m}_t^\alpha$.
  3 - Mise à jour : Calculer
  $\boldsymbol{f}_t^+ = \boldsymbol{f}_t^- + \boldsymbol{P}_t^- \boldsymbol{H}_t^\top (\boldsymbol{H}_t \boldsymbol{P}_t^- \boldsymbol{H}_t^\top + \boldsymbol{\Sigma}_{\alpha t})^{-1} [\boldsymbol{m}_t^\alpha - \boldsymbol{H}_t \boldsymbol{f}_t^-]$,
  $\boldsymbol{P}_t^+ = \boldsymbol{P}_t^- - \boldsymbol{P}_t^- \boldsymbol{H}_t^\top (\boldsymbol{H}_t \boldsymbol{P}_t^- \boldsymbol{H}_t^\top + \boldsymbol{\Sigma}_{\alpha t})^{-1} \boldsymbol{H}_t \boldsymbol{P}_t^-$.
end for

Dans la Fig. F.16, nous comparons les résultats obtenus par cet algorithme 'DL Adaptive Kalman' (Deep learned based adaptive Kalman) avec des méthodes concurrentes. Comme nous pouvons le voir, l'image reconstruite est similaire à celle obtenue par échantillonnage adaptatif avec des estimations exactes de la variance. Cette méthode est capable de capturer le mouvement des objets pour les 10 premières images. Après 15 images, l'arrière-plan commence à être corrompu par des artefacts, et nous devrions probablement rafraîchir l'image avec de nouvelles acquisitions plus longues, comme celle de la première image.

Conclusion

Dans ce chapitre, nous explorons les algorithmes de reconstruction basés sur des méthodes d'estimation d'état bayésiennes. Cela nous a permis de développer des schémas de reconstruction qui visent à réduire le nombre de mesures acquises en utilisant les informations des trames précédemment acquises. En particulier, nous avons développé un schéma de sous-échantillonnage adaptatif dans le temps basé sur l'estimation de la variance en utilisant l'apprentissage profond. Nos expériences indiquent que cette méthode permettrait de réduire pour environ 10 trames de 512 mesures à 200 mesures tout en conservant une qualité d'image similaire.

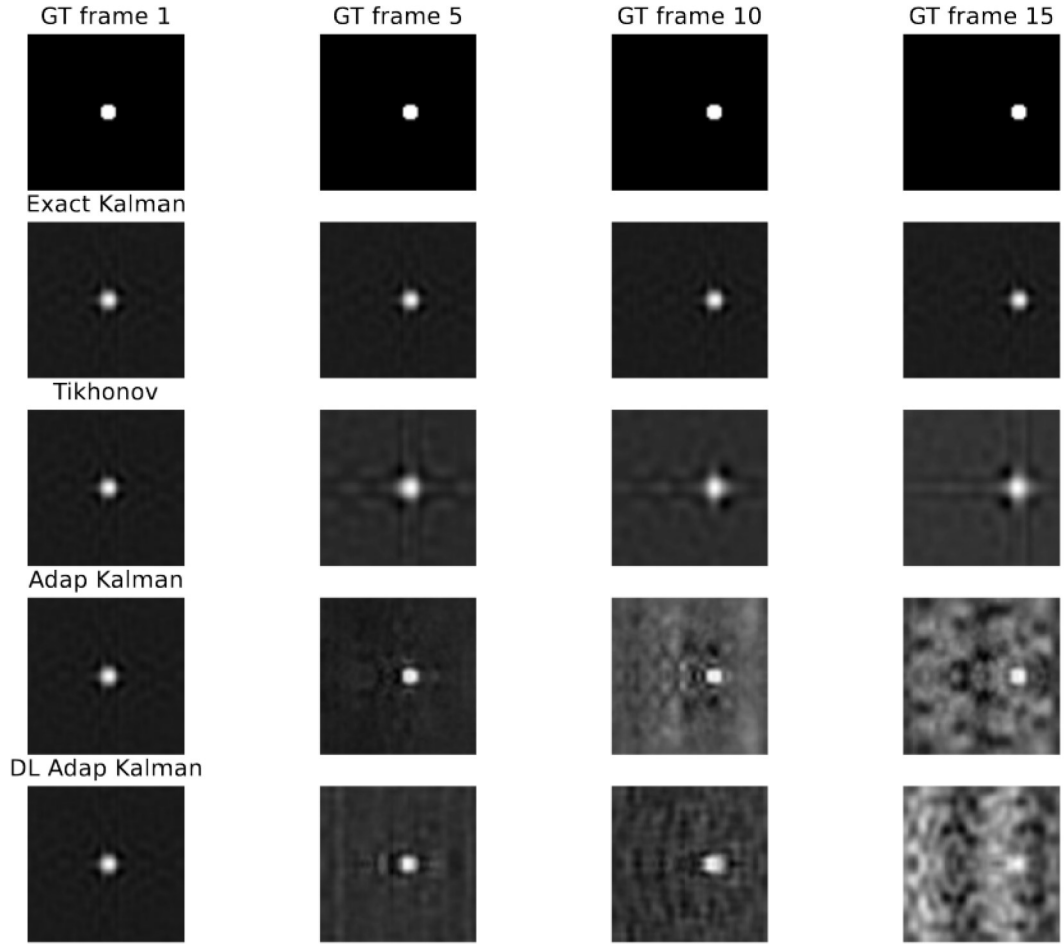


FIGURE F.16 – Méthodes de reconstruction. La première image est estimée via la régularisation de Tikhovov avec 512 mesures. Pour le reste des images, nous estimons l'image reconstruite en considérant 200 mesures avec 'Exact Kalman' : les équations de filtrage de Kalman lorsque le modèle prédictif est parfait (comme une traduction) ; 'Tikhonov' : la solution régularisée de Tikhonov généralisée ; 'Adap Kalman' : les équations de filtrage de Kalman mais où les coefficients échantillonnés sont acquis sur la base de l'estimation exacte de la variance ; 'DL Adap Kalman' : les équations de filtrage de Kalman sont utilisées mais où les coefficients échantillonnés sont acquis sur la base de l'estimation de la variance donnée par un réseau neuronal profond.

List of Publications

Ducros, N., A Lorente Mur, and F Peyrin (2020). "A Completion Network for Reconstruction from Compressed Acquisition". In: *2020 IEEE 17th International Symposium on Biomedical Imaging (ISBI)*, pp. 619–623. DOI: 10.1109/ISBI45749.2020.9098390.

Lorente Mur, A., F. Peyrin, and N. Ducros (2020). "Recurrent Neural Networks for Compressive Video Reconstruction". In: *2020 IEEE 17th International Symposium on Biomedical Imaging (ISBI)*, pp. 1651–1654. DOI: 10.1109/ISBI45749.2020.9098327.

Lorente Mur, A. et al. (2019). "Handling negative patterns for fast single-pixel lifetime imaging". In: *Molecular-Guided Surgery: Molecules, Devices, and Applications V*. Ed. by B. W. Pogue and S. Gioux. Vol. 10862. International Society for Optics and Photonics. SPIE, pp. 16 –25. DOI: 10.1117/12.2511123.

Lorente Mur, A. et al. (2020). "Deep neural networks for single-pixel compressive video reconstruction". In: *Unconventional Optical Imaging II*. Ed. by C. Fournier, M. P. Georges, and G. Popescu. Vol. 11351. International Society for Optics and Photonics. SPIE, pp. 71 –80. DOI: 10.1117/12.2553326.

Lorente Mur, A. et al. (2021). "Deep Expectation-Maximization For Image Reconstruction From Under-Sampled Poisson Data". In: *2021 IEEE 18th International Symposium on Biomedical Imaging (ISBI)*, pp. 1535–1539. DOI: 10.1109/ISBI48211.2021.9433805.

Software and datasets

Ducros, N. and A. Lorente Mur (2020). *Single-Pixel Hyperspectral Imaging dataset Version 1.0*. https://github.com/openspyrit/spihim.

Lorente Mur, A. and N. Ducros (2020). *Single-Pixel pYthon Image Reconstruction Toolbox (spyrit) Version 1.0*. https://github.com/openspyrit/spyrit.

Bibliography

Adler, J. and O. Öktem (2018). "Learned Primal-Dual Reconstruction". In: *IEEE Transactions on Medical Imaging* 37.6, pp. 1322–1332. DOI: 10.1109/TMI.2018.2799231.

Aggarwal, H. K., M. P. Mani, and M. Jacob (2019). "MoDL: Model-Based Deep Learning Architecture for Inverse Problems". In: *IEEE Transactions on Medical Imaging* 38.2, pp. 394–405. DOI: 10.1109/TMI.2018.2865356.

Aguénounon, E. et al. (2019). "Single snapshot imaging of optical properties using a single-pixel camera: a simulation study". In: *Journal of Biomedical Optics* 24.7, pp. 1 –6. DOI: 10.1117/1.JBO.24.7.071612.

Aigner, S. and M. Körner (2018). "FutureGAN: Anticipating the Future Frames of Video Sequences using Spatio-Temporal 3d Convolutions in Progressively Growing Autoencoder GANs". In: *ArXiv* abs/1810.01325.

Alston, L. (2017). "Spectroscopie de fluorescence et imagerie optique pour l'assistance à la résection de gliomes : conception et caractérisation de systèmes de mesure et modèles de traitement des données associées, sur fantômes et au bloc opératoire". PhD thesis. Université Claude Bernard Lyon 1.

Anscombe, F. J. (Dec. 1948). "The transformation of Poisson, binomial and negative-binomial data". In: *Biometrika* 35.3-4, pp. 246–254. ISSN: 0006-3444. DOI: 10.1093/biomet/35.3-4.246. eprint: https://academic.oup.com/biomet/article-pdf/35/3-4/246/785684/35-3-4-246.pdf.

Arce, G. R. et al. (2014). "Compressive Coded Aperture Spectral Imaging: An Introduction". In: *IEEE Signal Processing Magazine* 31.1, pp. 105–115. DOI: 10.1109/MSP.2013.2278763.

Arridge, S. et al. (2019). "Solving inverse problems using data-driven models". In: *Acta Numerica* 28, pp. 1–174.

Baldassarre, L. et al. (2016). "Learning-Based Compressive Subsampling". In: *IEEE Journal of Selected Topics in Signal Processing* 10.4, pp. 809–822. ISSN: 1941-0484. DOI: 10.1109/JSTSP.2016.2548442.

Ballas, N. et al. (2015). "Delving deeper into convolutional networks for learning video representations". In: *arXiv preprint arXiv:1511.06432*.

Barbastathis, G., A. Ozcan, and G. Situ (2019). "On the use of deep learning for computational imaging". In: *Optica* 6.8, pp. 921–943. DOI: 10.1364/OPTICA.6.000921.

Bengio, Y. (2012). "Practical Recommendations for Gradient-Based Training of Deep Architectures". In: *Neural Networks: Tricks of the Trade: Second Edition*. Ed. by G. Montavon, G. B. Orr, and K.-R. Müller. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 437–478. ISBN: 978-3-642-35289-8. DOI: 10.1007/978-3-642-35289-8_26.

Benvenuto, F et al. (2008). "The study of an iterative method for the reconstruction of images corrupted by Poisson and Gaussian noise". In: *Inverse Problems* 24.3, p. 035016. DOI: 10.1088/0266-5611/24/3/035016.

Bishop, C. (1995). *Neural networks for pattern recognition*. Oxford New York: Clarendon Press Oxford University Press. ISBN: 978-0-19-853864-6.

Boissy, J., J.-F. Giovannelli, and P. Minvielle (2020). "An Insight Into the Gibbs Sampler: Keep the Samples or Drop Them?" In: *IEEE Signal Processing Letters* 27, pp. 2069–2073. DOI: 10.1109/LSP.2020.3038136.

Boyd, S. et al. (Jan. 2011). "Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers". In: *Foundations and Trends in Machine Learning* 3, pp. 1–122. DOI: 10.1561/2200000016.

Bugeau, A. et al. (2010). "Coherent background video inpainting through Kalman smoothing along trajectories". In: *VMV 2010-15th International Workshop on Vision, Modeling, and Visualization Workshop*, pp. 123–130.

Burge, J. et al. (2020). "Convolutional LSTM Neural Networks for Modeling Wildland Fire Dynamics". In: *CoRR* abs/2012.06679. arXiv: 2012.06679.

Chambolle, A. and T. Pock (Dec. 2010). "A First-Order Primal-Dual Algorithm for Convex Problems with Applications to Imaging". In: *Journal of Mathematical Imaging and Vision* 40.1, pp. 120–145. DOI: 10.1007/s10851-010-0251-1.

Chapman, S. (May 1928). "On the Brownian displacements and thermal diffusion of grains suspended in a non-uniform fluid". In: *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character* 119.781, pp. 34–54. DOI: 10.1098/rspa.1928.0082.

Chen, Q. et al. (2018). "Imaging of hidden object using passive mode single pixel imaging with compressive sensing". In: *Laser Physics Letters* 15.12, p. 126201. DOI: 10.1088/1612-202x/aae216.

Cho, K. et al. (2014). "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation". In: *CoRR* abs/1406.1078. arXiv: 1406.1078.

Chouzenoux, E. et al. (2015). "A Convex Approach for Image Restoration with Exact Poisson–Gaussian Likelihood". In: *SIAM Journal on Imaging Sciences* 8.4, pp. 2662–2682. DOI: 10.1137/15M1014395. eprint: https://doi.org/10.1137/15M1014395.

Chrabaszcz, P., I. Loshchilov, and F. Hutter (2017). "A Downsampled Variant of ImageNet as an Alternative to the CIFAR datasets". In: *CoRR* abs/1707.08819. arXiv: 1707.08819.

Chun, I. Y. and J. A. Fessler (2020). "Convolutional Analysis Operator Learning: Acceleration and Convergence". In: *IEEE Transactions on Image Processing* 29, pp. 2108–2122.

Chung, J. et al. (2014). "Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling". In: *CoRR* abs/1412.3555. arXiv: 1412.3555.

Coates, A., A. Ng, and H. Lee (2011). "An Analysis of Single-Layer Networks in Unsupervised Feature Learning". In: *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. Ed. by G. Gordon, D. Dunson, and M. Dudík. Vol. 15. Proceedings of Machine Learning Research. Fort Lauderdale, FL, USA: PMLR, pp. 215–223.

Creath, K. (1988). "V Phase-Measurement Interferometry Techniques". In: ed. by E. Wolf. Vol. 26. Progress in Optics. Elsevier, pp. 349–393. DOI: https://doi.org/10.1016/S0079-6638(08)70178-1.

Czajkowski, K. M., A. Pastuszczak, and R. Kotyński (2018). "Real-time single-pixel video imaging with Fourier domain regularization". In: *Opt. Express* 26.16, pp. 20009–20022. DOI: 10.1364/OE.26.020009.

Dabov, K. et al. (2007). "Image Denoising by Sparse 3-D Transform-Domain Collaborative Filtering". In: *IEEE Transactions on Image Processing* 16.8, pp. 2080–2095. DOI: 10.1109/TIP.2007.901238.

Dave, A. et al. (2019). "Solving Inverse Computational Imaging Problems Using Deep Pixel-Level Prior". In: *IEEE Transactions on Computational Imaging* 5, pp. 37–51.

Dempster, A. P., N. M. Laird, and D. B. Rubin (Sept. 1977). "Maximum Likelihood from Incomplete Data Via theEMAlgorithm". In: *Journal of the Royal Statistical Society: Series B (Methodological)* 39.1, pp. 1–22. DOI: 10.1111/j.2517-6161.1977.tb01600.x.

Deng, J. et al. (2009). "Imagenet: A large-scale hierarchical image database". In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee, pp. 248–255.

Ding, Q. et al. (2018). "Statistical Image Reconstruction Using Mixed Poisson-Gaussian Noise Model for X-Ray CT". In: *arXiv: Medical Physics*.

Ding, X., W. Chen, and I. Wassell (2014). "Generalized-KFCS: Motion estimation enhanced Kalman filtered compressive sensing for video". In: *2014 IEEE International Conference on Image Processing (ICIP)*, pp. 1297–1301. DOI: 10.1109/ICIP.2014.7025259.

Dong, C. et al. (2015). "Image Super-Resolution Using Deep Convolutional Networks". In: *CoRR* abs/1501.00092. arXiv: 1501.00092.

Donoho, D. L. (2006). "Compressed sensing". In: *IEEE Transactions on Information Theory* 52.4, pp. 1289–1306. DOI: 10.1109/TIT.2006.871582.

Doucet, A. (2001). *Sequential Monte Carlo methods in practice*. New York: Springer. ISBN: 978-0-387-95146-1.

Duarte, M. F. et al. (2008). "Single-pixel imaging via compressive sampling". In: *IEEE Signal Processing Magazine* 25.2, pp. 83–91. DOI: 10.1109/MSP.2007.914730.

Ducros, N. (2017). *Fast Single-Pixel Biomedical Imaging – Armoni*. https://anr.fr/Project-ANR-17-CE19-0003.

Ducros, N. and A. Lorente Mur (2020). *Single-Pixel Hyperspectral Imaging dataset Version 1.0*. https://github.com/openspyrit/spihim.

Ducros, N., A Lorente Mur, and F Peyrin (2020). "A Completion Network for Reconstruction from Compressed Acquisition". In: *2020 IEEE 17th International Symposium on Biomedical Imaging (ISBI)*, pp. 619–623. DOI: 10.1109/ISBI45749.2020.9098390.

Duda, R. O. and P. E. Hart (1973). "Pattern classification and scene analysis". In: *A Wiley-Interscience publication*.

Eckstein, J. and W. Yao (Oct. 2015). "Understanding the convergence of the alternating direction method of multipliers: theoretical and computational perspectives". In: *Pacific Journal of Optimization* 11, pp. 619–644.

Edgar, M. P., G. M. Gibson, and M. J. Padgett (Dec. 2018). "Principles and prospects for single-pixel imaging". In: *Nature Photonics* 13.1, pp. 13–20. DOI: 10.1038/s41566-018-0300-7.

— (Jan. 2019). "Principles and prospects for single-pixel imaging". In: *Nature Photonics* 13.1, pp. 13–20. ISSN: 1749-4893.

Fessler, J. and A. Hero (1994). "Space-alternating generalized expectation-maximization algorithm". In: *IEEE Transactions on Signal Processing* 42.10, pp. 2664–2677. DOI: 10.1109/78.324732.

Foi, A. et al. (2008). "Practical Poissonian-Gaussian Noise Modeling and Fitting for Single-Image Raw-Data". In: *IEEE Transactions on Image Processing* 17.10, pp. 1737–1754. DOI: 10.1109/TIP.2008.2001399.

Foster, D. (2010). "Chromatic Function of the Cones". In: *Encyclopedia of the Eye*. Ed. by D. Dartt et al. Academic Press, pp. 266–274.

Gedalin, D., Y. Oiknine, and A. Stern (2019). "DeepCubeNet: reconstruction of spectrally compressive sensed hyperspectral images with deep neural networks." In: *Optics express* 27 24, pp. 35811–35822.

Gelman, A., W. R. Gilks, and G. O. Roberts (1997). "Weak convergence and optimal scaling of random walk Metropolis algorithms". In: *The Annals of Applied Probability* 7.1, pp. 110 –120. DOI: 10.1214/aoap/1034625254.

Ghulyani, M. and M. Arigovindan (2021). "Fast Roughness Minimizing Image Restoration Under Mixed Poisson–Gaussian Noise". In: *IEEE Transactions on Image Processing* 30, pp. 134–149. DOI: 10.1109/TIP.2020.3032036.

Gibson, G., S. Johnson, and M. Padgett (2020). "Single-pixel imaging 12 years on: a review." In: *Optics express* 28 19, pp. 28190–28208.

Gibson, G. M. et al. (2017). "Real-time imaging of methane gas leaks using a single-pixel camera." In: *Optics express* 25 4, pp. 2998–3005.

Gilton, D., G. Ongie, and R. Willett (2020). "Neumann Networks for Linear Inverse Problems in Imaging". In: *IEEE Transactions on Computational Imaging* 6, pp. 328–343.

Giovannelli, J.-F. (2008). "Unsupervised Bayesian Convex Deconvolution Based on a Field With an Explicit Partition Function". In: *IEEE Transactions on Image Processing* 17.1, pp. 16–26. DOI: 10.1109/TIP.2007.911819.

Glorot, X. and Y. Bengio (2010). "Understanding the difficulty of training deep feedforward neural networks". In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. Ed. by Y. W. Teh and M. Titterington. Vol. 9. Proceedings of Machine Learning Research. Chia Laguna Resort, Sardinia, Italy: PMLR, pp. 249–256.

Godsill, S. J., A. Doucet, and M. West (2004). "Monte Carlo Smoothing for Nonlinear Time Series". In: *Journal of the American Statistical Association* 99.465, pp. 156–168. DOI: 10.1198/016214504000000151. eprint: https://doi.org/10.1198/016214504000000151.

Goldstein, T. and S. Osher (Jan. 2009). "The Split Bregman Method for L1-Regularized Problems". In: *SIAM Journal on Imaging Sciences* 2.2, pp. 323–343. DOI: 10.1137/080725891.

Gordon, N., D. Salmond, and A. F. M. Smith (1993). "Novel approach to nonlinear/non-Gaussian Bayesian state estimation". In:

Graves, A., A. Mohamed, and G. E. Hinton (2013). "Speech Recognition with Deep Recurrent Neural Networks". In: *CoRR* abs/1303.5778. arXiv: 1303.5778.

Graves, A. et al. (2009). "A Novel Connectionist System for Unconstrained Handwriting Recognition". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 31.5, pp. 855–868. DOI: 10.1109/TPAMI.2008.137.

Gupta, H. et al. (2018). "CNN-Based Projected Gradient Descent for Consistent CT Image Reconstruction". In: *IEEE Transactions on Medical Imaging* 37.6, pp. 1440–1453. DOI: 10.1109/TMI.2018.2832656.

Gurrola-Ramos, J., O. Dalmau, and T. E. Alarcón (2021). "A Residual Dense U-Net Neural Network for Image Denoising". In: *IEEE Access* 9, pp. 31742–31754. DOI: 10.1109/ACCESS.2021.3061062.

Hadamard, J. (1893). "Résolution d'une question relative aux determinants." In: *Bulletin des Sciences Mathematiques* 17, pp. 240–246.

Hastings, W. K. (Apr. 1970). "Monte Carlo sampling methods using Markov chains and their applications". In: *Biometrika* 57.1, pp. 97–109. ISSN: 0006-3444. DOI: 10.1093/biomet/57.1.97. eprint: https://academic.oup.com/biomet/article-pdf/57/1/97/23940249/57-1-97.pdf.

Higham, C. et al. (2018). "Deep learning for real-time single-pixel video". In: *Scientific Reports*. 8. DOI: 10.1038/s41598-018-20521-y..

Hochreiter, S. and J. Schmidhuber (Dec. 1997). "Long Short-term Memory". In: *Neural computation* 9, pp. 1735–80. DOI: 10.1162/neco.1997.9.8.1735.

Hornik, K., M. Stinchcombe, and H. White (1989). "Multilayer feedforward networks are universal approximators". In: *Neural Networks* 2.5, pp. 359–366. ISSN: 0893-6080. DOI: https://doi.org/10.1016/0893-6080(89)90020-8.

Hoshi, I. et al. (2020). "Single-pixel imaging using a recurrent neural network combined with convolutional layers". In: *Opt. Express* 28.23, pp. 34069–34078. DOI: 10.1364/OE.410191.

Huang, K. (2001). *Introduction to statistical physics*. London New York: Taylor & Francis. ISBN: 0-7484-0941-6.

J. Candès, E., J. K. Romberg, and T. Tao (Aug. 2006). "Stable Signal Recovery from Incomplete and Inaccurate Measurements". In: 59.

Jazwinski, A. (1970). *Stochastic processes and filtering theory*. New York: Academic Press. ISBN: 9780080960906.

Jiang, C. et al. (2016). "Hyperspectral Image Denoising with a Combined Spatial and Spectral Weighted Hyperspectral Total Variation Model". In: *Canadian Journal of Remote Sensing* 42, pp. 53 –72.

Jiao, S. et al. (2020). "Does deep learning always outperform simple linear regression in optical imaging?" In: *Opt. Express* 28.3, pp. 3717–3731. DOI: 10.1364/OE.382319.

Jin, K. H. et al. (2017). "Deep Convolutional Neural Network for Inverse Problems in Imaging". In: *IEEE Transactions on Image Processing* 26.9, pp. 4509–4522. DOI: 10.1109/TIP.2017.2713099.

Julier, S., J. Uhlmann, and H. Durrant-Whyte (1995). "A new approach for filtering nonlinear systems". In: *Proceedings of 1995 American Control Conference - ACC'95* 3, 1628–1632 vol.3.

Kalman, R. E. et al. (1960). "Contributions to the theory of optimal control". In: *Bol. soc. mat. mexicana* 5.2, pp. 102–119.

Kang, E., J. Min, and J. C. Ye (Oct. 2017). "A deep convolutional neural network using directional wavelets for low-dose X-ray CT reconstruction". In: *Medical Physics* 44.10, e360–e375. DOI: 10.1002/mp.12344.

Kang, E. et al. (2019). "Cycle-consistent adversarial denoising network for multiphase coronary CT angiography". In: *Medical Physics* 46, 550–562.

Kellman, M. R. et al. (2019). "Physics-Based Learned Design: Optimized Coded-Illumination for Quantitative Phase Imaging". In: *IEEE Transactions on Computational Imaging* 5.3, pp. 344–353.

Kim, S. Y. et al. (2018). "3DSRnet: Video Super-resolution using 3D Convolutional Neural Networks". In: *CoRR* abs/1812.09079. arXiv: 1812.09079.

Kingma, D. P. and J. Ba (2014). "Adam: A Method for Stochastic Optimization". In: *CoRR* abs/1412.6980.

Kitagawa, G. (Mar. 1996). "Monte Carlo Filter and Smoother for Non-Gaussian Nonlinear State Space Models". In: *Journal of Computational and Graphical Statistics* 5.1, p. 1. DOI: 10.2307/1390750.

Kiwiel, K. C. (Mar. 2001). "Convergence and efficiency of subgradient methods for quasiconvex minimization". In: *Mathematical Programming* 90.1, pp. 1–25. DOI: 10.1007/pl00011414.

Kuhn, H. W. and A. W. Tucker (1951). "Nonlinear programming". In: *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability, 1950*. Berkeley and Los Angeles: University of California Press, pp. 481–492.

Le Cun, Y., I. Kanter, and S. Solla (1991). "Eigenvalues of covariance matrices: Application to neural-network learning". English (US). In: *Physical Review Letters* 66.18, pp. 2396–2399. ISSN: 0031-9007. DOI: 10.1103/PhysRevLett.66.2396.

LeCun, Y. (1989). "Generalization and network design strategies". In: *Connectionism in perspective* 19, pp. 143–155.

LeCun, Y. A. et al. (1998). *Neural networks : tricks of the trade*. Berlin New York: Springer. ISBN: 978-3-540-65311-0.

Lee, A. X. et al. (2018). "Stochastic Adversarial Video Prediction". In: *arXiv preprint arXiv:1804.01523*.

Lefkimmiatis, S. and M. Unser (2013). "Poisson Image Reconstruction With Hessian Schatten-Norm Regularization". In: *IEEE Transactions on Image Processing* 22.11, pp. 4314–4327.

Li, C. (2010). "An efficient algorithm for total variation regularization with applications to the single pixel camera and compressive sensing". PhD thesis. Rice University.

Li, F. et al. (2020a). "Compressive ghost imaging through scattering media with deep learning". In: *Opt. Express* 28.12, pp. 17395–17408. DOI: 10.1364/OE.394639.

Li, J., F. Luisier, and T. Blu (2016). "Deconvolution of poissonian images with the PURE-LET approach". In: *2016 IEEE International Conference on Image Processing (ICIP)*, pp. 2708–2712.

Li, J. et al. (2015). "A reweighted l^2 method for image restoration with Poisson and mixed Poisson-Gaussian noise". In: *Inverse Problems and Imaging* 9.3, pp. 875–894. DOI: 10.3934/ipi.2015.9.875.

Li, J. et al. (2020b). "Single-pixel compressive optical image hiding based on conditional generative adversarial network". In: *Opt. Express* 28.15, pp. 22992–23002. DOI: 10.1364/OE.399065.

Li, J., F. Luisier, and T. Blu (2018). "PURE-LET Image Deconvolution". In: *IEEE Transactions on Image Processing* 27.1, pp. 92–105. DOI: 10.1109/TIP.2017.2753404.

Liang, X. et al. (2017). "Dual Motion GAN for Future-Flow Embedded Video Prediction". In: *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 1762–1770. DOI: 10.1109/ICCV.2017.194.

Liu, X. et al. (2020). "Photon-limited single-pixel imaging". In: *Opt. Express* 28.6, pp. 8132–8144. DOI: 10.1364/OE.381785.

Lorente Mur, A. and N. Ducros (2020). *Single-Pixel pYthon Image Reconstruction Toolbox (spyrit) Version 1.0*. https://github.com/openspyrit/spyrit.

Lorente Mur, A., F. Peyrin, and N. Ducros (2020). "Recurrent Neural Networks for Compressive Video Reconstruction". In: *2020 IEEE 17th International Symposium on Biomedical Imaging (ISBI)*, pp. 1651–1654. DOI: 10.1109/ISBI45749.2020.9098327.

Lorente Mur, A. et al. (2019). "Handling negative patterns for fast single-pixel lifetime imaging". In: *Molecular-Guided Surgery: Molecules, Devices, and Applications V*. Ed. by B. W. Pogue and S. Gioux. Vol. 10862. International Society for Optics and Photonics. SPIE, pp. 16 –25. DOI: 10.1117/12.2511123.

Lorente Mur, A. et al. (2020). "Deep neural networks for single-pixel compressive video reconstruction". In: *Unconventional Optical Imaging II*. Ed. by C. Fournier, M. P. Georges, and G. Popescu. Vol. 11351. International Society for Optics and Photonics. SPIE, pp. 71 –80. DOI: 10.1117/12.2553326.

Lorente Mur, A. et al. (2021). "Deep Expectation-Maximization For Image Reconstruction From Under-Sampled Poisson Data". In: *2021 IEEE 18th International Symposium on Biomedical Imaging (ISBI)*, pp. 1535–1539. DOI: 10.1109/ISBI48211.2021.9433805.

Lorente Mur, A. et al. (2021). "Single-pixel image reconstruction from experimental data using neural networks". In: *Opt. Express* 29.11, pp. 17097–17110. DOI: 10.1364/OE.424228.

Lu, G. et al. (2018). "Deep Kalman Filtering Network for Video Compression Artifact Reduction". In: *Computer Vision – ECCV 2018*. Ed. by V. Ferrari et al. Cham: Springer International Publishing, pp. 591–608. ISBN: 978-3-030-01264-9.

Lu, G. et al. (2020). "Deep Non-Local Kalman Network for Video Compression Artifact Reduction". In: *IEEE Transactions on Image Processing* 29, pp. 1725–1737. DOI: 10.1109/TIP.2019.2943214.

Luisier, F., T. Blu, and M. Unser (2011). "Image Denoising in Mixed Poisson–Gaussian Noise". In: *IEEE Transactions on Image Processing* 20.3, pp. 696–708.

Magalhaes, F. et al. (2012). "High-resolution hyperspectral single-pixel imaging system based on compressive sensing". In: *Optical Engineering* 51.7, pp. 1 –7. DOI: 10.1117/1.OE.51.7.071406.

Magnusson, M. et al. (2020). "Creating RGB Images from Hyperspectral Images Using a Color Matching Function". In: *IGARSS 2020 - 2020 IEEE International Geoscience and Remote Sensing Symposium*, pp. 2045–2048.

Makitalo, M. and A. Foi (2011). "Optimal Inversion of the Anscombe Transformation in Low-Count Poisson Image Denoising". In: *IEEE Transactions on Image Processing* 20.1, pp. 99–109. DOI: 10.1109/TIP.2010.2056693.

Mallat, S. (1999). *A wavelet tour of signal processing*. Academic press.

McCann, M. T., K. H. Jin, and M. Unser (2017). "Convolutional Neural Networks for Inverse Problems in Imaging: A Review". In: *IEEE Signal Processing Magazine* 34.6, pp. 85–95. DOI: 10.1109/MSP.2017.2739299.

Murtagh, F., J. luc Starck, and A. Bijaoui (1995). "Image Restoration with Noise Suppression Using a Multiresolution Support". In: *Astronomy and Astrophysics, Suppl. Ser* 112, pp. 179–189.

Nathanson, H. (US patent 3746911, 1973-7-17). *ELECTROSTATICALLY DEFLECTABLE LIGHT VALVES FOR PROJECTION DISPLAYS*.

Ochoa, M. et al. (2018). "Assessing patterns for compressive fluorescence lifetime imaging". In: *Opt. Lett.* 43.18, pp. 4370–4373. DOI: 10.1364/OL.43.004370.

Oliu, M., J. Selva, and S. Escalera (2017). "Folded Recurrent Neural Networks for Future Video Prediction". In: *CoRR* abs/1712.00311. arXiv: 1712.00311.

Oprea, S. et al. (2020). "A Review on Deep Learning Techniques for Video Prediction". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 01, pp. 1–1. ISSN: 1939-3539. DOI: 10.1109/TPAMI.2020.3045007.

Parikh, N. (Jan. 2014). "Proximal Algorithms". In: *Foundations and Trends in Optimization* 1, pp. 127–239. DOI: 10.1561/2400000003.

Paszke, A. et al. (2019). "Pytorch: An imperative style, high-performance deep learning library". In: *Advances in neural information processing systems* 32, pp. 8026–8037.

Peller, J., F. Farahi, and S. Trammell (2018). "Hyperspectral imaging system based on a single-pixel camera design for detecting differences in tissue properties." In: *Applied optics* 57 27, pp. 7651–7658.

Peng, J. et al. (Apr. 2021). "Fourier microscopy based on single-pixel imaging for multi-mode dynamic observations of samples". In: *APL Photonics* 6.4, p. 046102. DOI: 10.1063/5.0042779.

Penrose, R. (July 1955). "A generalized inverse for matrices". In: *Mathematical Proceedings of the Cambridge Philosophical Society* 51.3, pp. 406–413. DOI: 10.1017/s0305004100030401.

Photography — Electronic still picture imaging — Resolution and spatial frequency responses (2019). https://www.iso.org/standard/71696.html. International Organization for Standardization. Geneva, CH.

Pian, Q. et al. (2017). "Compressive hyperspectral time-resolved wide-field fluorescence lifetime imaging." In: *Nature photonics* 11, pp. 411–414. ISSN: 1749-4885. DOI: 10.1038/NPHOTON.2017.82.

Pronina, V. et al. (2020). "Microscopy Image Restoration with Deep Wiener-Kolmogorov filters". In: *ECCV*.

Radwell, N. et al. (2014). "Single-pixel infrared and visible microscope". In: *Optica* 1.5, pp. 285–289. DOI: 10.1364/OPTICA.1.000285.

Radwell, N. et al. (Dec. 2019). "Deep learning optimized single-pixel LiDAR". In: *Applied Physics Letters* 115.23, p. 231101. DOI: 10.1063/1.5128621.

Reader, A. J. et al. (2021). "Deep Learning for PET Image Reconstruction". In: *IEEE Transactions on Radiation and Plasma Medical Sciences* 5.1, pp. 1–25. DOI: 10.1109/TRPMS.2020.3014786.

Rizvi, S. et al. (2020). "Deringing and denoising in extremely under-sampled Fourier single pixel imaging". In: *Opt. Express* 28.5, pp. 7360–7374. DOI: 10.1364/OE.385233.

Rodríguez, A. et al. (2016). "Dual-mode optical microscope based on single-pixel imaging". In: *Optics and Lasers in Engineering* 82, pp. 87–94. ISSN: 0143-8166. DOI: https://doi.org/10.1016/j.optlaseng.2016.02.004.

Ronneberger, O. et al. (2015). "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *CoRR* abs/1505.04597. arXiv: 1505.04597.

Rosenberger, M. et al. (Dec. 2016). "EMVA 1288 Camera characterisation and the influences of radiometric camera characteristics on geometric measurements". In: *ACTA IMEKO* 5, p. 81. DOI: 10.21014/acta_imeko.v5i4.356.

Rosenblatt, F. (1961). *Principles of neurodynamics. perceptrons and the theory of brain mechanisms*. Tech. rep. Cornell Aeronautical Lab Inc Buffalo NY.

Rousset, F., F. Peyrin, and N. Ducros (2018). "A Semi Nonnegative Matrix Factorization Technique for Pattern Generalization in Single-Pixel Imaging". In: *IEEE Transactions on Computational Imaging* 4.2, pp. 284–294. ISSN: 2333-9403. DOI: 10.1109/TCI.2018.2811910.

Rousset, F. et al. (2017). "Adaptive Basis Scan by Wavelet Prediction for Single-Pixel Imaging". In: *IEEE Transactions on Computational Imaging* 3.1, pp. 36–46. ISSN: 2333-9403. DOI: 10.1109/TCI.2016.2637079.

Rousset, F. et al. (2018). "Time-resolved multispectral imaging based on an adaptive single-pixel camera". In: *Opt. Express* 26.8, pp. 10550–10558. DOI: 10.1364/OE.26.010550.

Rudin, L. I., S. Osher, and E. Fatemi (1992). "Nonlinear total variation based noise removal algorithms". In: *Physica D: Nonlinear Phenomena* 60.1, pp. 259–268. ISSN: 0167-2789. DOI: https://doi.org/10.1016/0167-2789(92)90242-F.

Rumelhart, D. E., G. E. Hinton, and R. J. Williams (Oct. 1986). "Learning representations by back-propagating errors". In: *Nature* 323.6088, pp. 533–536. DOI: 10.1038/323533a0.

Saiful Islam, M. and E. Hossain (2020). "Foreign Exchange Currency Rate Prediction using a GRU-LSTM Hybrid Network". In: *Soft Computing Letters*, p. 100009. ISSN: 2666-2221. DOI: https://doi.org/10.1016/j.socl.2020.100009.

Sainath, T. N. et al. (2015). "Convolutional, long short-term memory, fully connected deep neural networks". In: *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, pp. 4580–4584.

Schmidhuber, J., D. Wierstra, and F. J. Gomez (2005). "Evolino: Hybrid neuroevolution/optimal linear search for sequence prediction". In: *Proceedings of the 19th International Joint Conferenceon Artificial Intelligence (IJCAI)*.

Siam, M. et al. (2017). "Convolutional gated recurrent networks for video segmentation". In: *2017 IEEE International Conference on Image Processing (ICIP)*, pp. 3090–3094. DOI: 10.1109/ICIP.2017.8296851.

Skellam, J. G. (1946). "The Frequency Distribution of the Difference Between Two Poisson Variates Belonging to Different Populations". In: *Journal of the Royal Statistical Society* 109.3, p. 296. DOI: 10.2307/2981372.

Sloane, N. J. A. and M. Harwit (1976). "Masks for Hadamard transform optics, and weighing designs". In: *Appl. Opt.* 15.1, pp. 107–114. DOI: 10.1364/AO.15.000107.

Smith, J. T., M. Ochoa, and X. Intes (2020). "UNMIX-ME: spectral and lifetime fluorescence unmixing via deep learning". In: *Biomed. Opt. Express* 11.7, pp. 3857–3874. DOI: 10.1364/BOE.391992.

Srivastava, N., E. Mansimov, and R. Salakhutdinov (2015). "Unsupervised Learning of Video Representations Using LSTMs". In: *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*. ICML'15. Lille, France: JMLR.org, 843–852.

Stockman, A. and L. Sharpe (1999). "Cone spectral sensitivities and color matching". In: *Color vision: From Genes to Perception*. Ed. by K. Gegenfurtner and L. T. Sharpe. Cambridge: Cambridge University Press. Chap. 2, pp. 53 –87.

Studer, V. et al. (2012). "Compressive fluorescence microscopy for biological and hyperspectral imaging". In: *Proceedings of the National Academy of Sciences* 109.26, E1679–E1687.

Sun, L. et al. (2019). "Joint cs-mri reconstruction and segmentation with a unified deep network". In: *International conference on information processing in medical imaging*. Springer, pp. 492–504.

Särkkä, S. (2013). *Bayesian Filtering and Smoothing*. Institute of Mathematical Statistics Textbooks. Cambridge University Press. DOI: 10.1017/CBO9781139344203.

Takhar, D. et al. (2006). "A new compressive imaging camera architecture using optical-domain compression". In: *in Proc. of Computational Imaging IV at SPIE Electronic Imaging*, pp. 43–52.

Tao, C. et al. (2021). "Compressive single-pixel hyperspectral imaging using RGB sensors." In: *Optics express* 29 7, pp. 11207–11220.

Tarantola, A. (2005). *Inverse Problem Theory and Methods for Model Parameter Estimation*. Society for Industrial and Applied Mathematics. DOI: 10.1137/1.9780898717921. eprint: https://epubs.siam.org/doi/pdf/10.1137/1.9780898717921.

Tikhonov, A. N. (1995). *Numerical methods for the solution of ill-posed problems*. Dordrecht Boston: Kluwer Academic Publishers. ISBN: 079233583X.

Valdés, P. A. et al. (2015). "Quantitative fluorescence using 5-aminolevulinic acid-induced protoporphyrin IX biomarker as a surgical adjunct in low-grade glioma surgery". In: *Journal of neurosurgery* 123.3, pp. 771–780.

van der Schaaf, A. and J. van Hateren (1996). "Modelling the Power Spectra of Natural Images: Statistics and Information". In: *Vision Research* 36.17, pp. 2759–2770. ISSN: 0042-6989. DOI: https://doi.org/10.1016/0042-6989(96)00002-8.

Walt, S. van der et al. (June 2014). "scikit-image: image processing in Python". In: *PeerJ* 2, e453. ISSN: 2167-8359. DOI: 10.7717/peerj.453.

Wan, E. and R. V. D. Merwe (2001). "Chapter 7 The Unscented Kalman Filter". In: *Kalman Filtering and Neural Networks*. Wiley, pp. 221–280.

Wang, G. et al. (2018). "Image Reconstruction is a New Frontier of Machine Learning". In: *IEEE Transactions on Medical Imaging* 37.6, pp. 1289–1296.

Wang, L. et al. (2019). "Hyperspectral Image Reconstruction Using a Deep Spatial-Spectral Prior". In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8024–8033.

Werbos, P. (1994). *The roots of backpropagation : from ordered derivatives to neural networks and political forecasting*. New York: Wiley. ISBN: 0471598976.

Wiener, N. (1949). *Extrapolation, interpolation, and smoothing of stationary time series*. New York: John Wiley & Sons, Inc.

Wu, D. et al. (Aug. 2021). "Imaging biological tissue with high-throughput single-pixel compressive holography". In: *Nature Communications* 12.1. DOI: 10.1038/s41467-021-24990-0.

Xingjian, S. et al. (2015). "Convolutional LSTM network: A machine learning approach for precipitation nowcasting". In: *Advances in neural information processing systems*, pp. 802–810.

Xu, L. et al. (2014). "Deep convolutional neural network for image deconvolution". In: *Advances in neural information processing systems* 27, pp. 1790–1798.

Yao, R. et al. (Mar. 2019). "Net-FLICS: fast quantitative wide-field fluorescence lifetime imaging with compressed sensing – a deep learning approach". In: *Light: Science & Applications* 8.1. DOI: 10.1038/s41377-019-0138-x.

Yin, R. et al. (Jan. 2017). "A Tale of Two Bases: Local-Nonlocal Regularization on Image Patches with Convolution Framelets". In: *SIAM Journal on Imaging Sciences* 10.2, pp. 711–750. DOI: 10.1137/16m1091447.

Yoo, J., A. Wahab, and J. C. Ye (Jan. 2018). "A Mathematical Framework for Deep Learning in Elastic Source Imaging". In: *SIAM Journal on Applied Mathematics* 78.5, pp. 2791–2818. DOI: 10.1137/18m1174027.

Yu, W.-K. et al. (July 2014). "Complementary compressive imaging for the telescopic system". In: *Scientific Reports* 4.1. DOI: 10.1038/srep05834.

Yu, X. et al. (2020a). "Deep Compressive Single Pixel Imaging by Reordering Hadamard Basis: A Comparative Study". In: *IEEE Access* 8, pp. 55773–55784. DOI: 10.1109/ACCESS.2020.2981505.

— (2020b). "Deep Compressive Single Pixel Imaging by Reordering Hadamard Basis: A Comparative Study". In: *IEEE Access* 8, pp. 55773–55784. DOI: 10.1109/ACCESS.2020.2981505.

Zhang, K. et al. (2017a). "Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising". In: *IEEE Transactions on Image Processing* 26.7, pp. 3142–3155. DOI: 10.1109/TIP.2017.2662206.

Zhang, K. et al. (2017b). "Learning deep CNN denoiser prior for image restoration". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3929–3938.

Zhang, Y. et al. (2020). "Dual-band single-pixel telescope". In: *Opt. Express* 28.12, pp. 18180–18188. DOI: 10.1364/OE.392522.

Zhang, Z. et al. (2017c). "Fast Fourier single-pixel imaging using binary illumination". In: *Scientific Reports* 7, pp. 2045–2322. DOI: 10.1038/s41598-017-12228-3.

Zhang, Z. et al. (2017d). "Hadamard single-pixel imaging versus Fourier single-pixel imaging". In: *Opt. Express* 25.16, pp. 19619–19639. DOI: 10.1364/OE.25.019619.

Zhou, J. et al. (2007). "A Bayesian MAP-EM Algorithm for PET Image Reconstruction Using Wavelet Transform". In: *IEEE Transactions on Nuclear Science* 54.5, pp. 1660–1669. DOI: 10.1109/TNS.2007.901200.

Çiçek, Ö. et al. (2016). "3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation". In: *MICCAI*.

FOLIO ADMINISTRATIF

THESE DE L'UNIVERSITE DE LYON OPEREE AU SEIN DE L'INSA LYON

NOM : LORENTE MUR DATE de SOUTENANCE : 14 / 12 / 2021

Prénoms : Antonio, Tomas

TITRE : Single-pixel imaging : compressed video acquisition and reconstruction using Deep learning

NATURE : Doctorat Numéro d'ordre : 2021LYSEI113

Ecole doctorale : Electronique, Electrotechnique, Automatique

Spécialité : Traitement du Signal et de l'Image

RESUME :
La caméra mono-pixel est une caméra qui permet de faire l'acquisition d'images bidimensionnelles à partir d'un capteur ponctuel. Elle mesure au niveau du détecteur le produit scalaire d'image de la scène avec des fonctions définies par l'utilisateur. L'image est alors récupérée par le biais d'algorithmes de reconstruction dédiés. La caméra mono-pixel peut être utilisée dans des problèmes d'imagerie où il serait impossible d'utiliser des méthodes d'imagerie conventionnelle par matrice de capteurs. En particulier, la caméra mono-pixel peut être couplée avec un spectromètre pour en faire une caméra hyperspectrale. De telles caméras mono-pixel permettent notamment de réaliser l'analyse de la signature spectrale de certaines molécules.

La principale limitation de l'imagerie mono-pixel sont les temps d'acquisition et de reconstruction, qui sont trop lents pour l'application en temps réel. L'objectif de cette thèse est la réalisation d'algorithmes de reconstruction et de sous-échantillonage pour permettre l'acquisition et reconstruction d'images à hautes fréquences.

Dans cette thèse nous avons étudié l'usage d'algorithmes d'apprentissage profond en imagerie mono-pixel. Nous avons introduit l'usage de solutions de régularisation généralisée de Tikhonov dans des reconstructeurs par réseaux de neurones dans le but de les appliquer à des données expérimentales. Puis, nous avons développé des architectures de réseaux neuronaux qui combinent des réseaux neuronaux avec l'algorithme d'espérance-maximisation afin d'estimer le maximum à posteriori de notre problème inverse. Enfin, nous avons étudié des schémas de sous-échantillonage évoluant au cours du temps selon l'évolution prédite au cours du temps de la variance. Nous avons combiné ce sous-échantillonage avec des schémas de reconstruction prenant en compte les frames reconstruites précédemment pour estimer la frame actuelle.

Par rapport aux approches classiques, notre approche permet l'acquisition et la reconstruction en temps réel, avec une cadence de 10 images par seconde. Enfin, nous avons montré l'application des méthodes proposées à des données issues d'une caméra mono-pixel hyperspectrale.

MOTS-CLÉS : Imagerie mono-pixel, Imagerie computationnelle, Optique computationnelle, Problèmes inverses, Reconstruction d'images, Reconstruction de vidéos, Apprentissage profond

Laboratoire (s) de recherche : CREATIS, CNRS UMR 5220 – INSERM U1294 – Université Lyon 1 – INSA Lyon – Université Jean Monnet Saint-Etienne.

Directeur de thèse: Françoise Peyrin

Président de jury : Gigan, Sylvain

Composition du jury :

Giovannelli, Jean-François	Professeur des universités	Université de Bordeaux 1	Rapporteur
Arridge, Simon	Professeur des universités	University College London	Rapporteur
Blu, Thierry	Professeur des universités	The Chinese University of Hong Kong	Examinateur
Gigan, Sylvain	Professeur des universités	Sorbonne	Examinateur
Peyrin, Françoise	Directrice de recherche	INSERM Lyon	Directeur de thèse
Ducros, Nicolas	Maître de conférence	INSA Lyon	Co-directeur de thèse