



**HAL**  
open science

# Learning strategies for computational MRI

Alban Gossard

► **To cite this version:**

Alban Gossard. Learning strategies for computational MRI. Medical Imaging. Université Paul Sabatier - Toulouse III, 2022. English. NNT : 2022TOU30209 . tel-04075522

**HAL Id: tel-04075522**

**<https://theses.hal.science/tel-04075522>**

Submitted on 20 Apr 2023

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Université Fédérale



Toulouse Midi-Pyrénées

# THÈSE

En vue de l'obtention du

**DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE**

Délivré par : *l'Université Toulouse 3 Paul Sabatier (UT3 Paul Sabatier)*

---

---

Présentée et soutenue le *12/12/2022* par :

**Alban GOSSARD**

**Méthodes d'apprentissage pour l'IRM computationnelle**

---

---

## JURY

PETER OCHS	Professor	Rapporteur
NICOLAS PAPADAKIS	Directeur de Recherche	Rapporteur
MARIE CHABERT	Professeure	Examinatrice
LAURENT CONDAT	Senior Research Scientist	Examineur
JULIE DELON	Professeure	Présidente
PIERRE WEISS	Chargé de Recherche	Directeur de thèse
FRÉDÉRIC DE GOURNAY	Maître de Conférences	Co-directeur de thèse

---

**École doctorale et spécialité :**

*MITT : Mathématiques et Applications*

**Unité de Recherche :**

*Institut de Mathématiques de Toulouse (UMR 5219)*

**Directeur(s) de Thèse :**

*Pierre WEISS et Frédéric de GOURNAY*

**Rapporteurs :**

*Peter OCHS et Nicolas PAPADAKIS*



*“Le changement climatique, de même que l’extinction des espèces ou l’augmentation des déchets plastiques, que l’on qualifie de problèmes, sont en fait des symptômes. La limitation du changement climatique est utile, mais revient à donner une aspirine à quelqu’un atteint d’un cancer. Cela l’aidera seulement à se sentir mieux temporairement. [...]”*

(Dennis Meadows)





## Avant-propos

Ces trois années de thèse furent à la fois longues de par les longs moments de creux, et courtes avec l'impression d'avoir commencé hier. Elles clôturent mes années d'études et à ce titre je me permets ici une petite digression.

Ne mentons pas, seul quelqu'un étant déjà passé par l'expérience de la thèse peut comprendre cette expérience unique. Des moments de déprime car rien ne fonctionne aux moments d'euphorie, c'est un véritable ascenseur avec une alternance qui semble ne jamais prendre fin. Malgré les déboires, cette expérience reste positive et je ne saurais trop recommander à des étudiants curieux voulant améliorer leur rigueur et leurs compétences scientifiques de se lancer dans cette épreuve. L'expérience sera longue, pavée de difficultés mais une fois au bout, la satisfaction est grande.

La recherche offre un cadre pour comprendre les choses, sans l'impératif d'avoir à développer un produit ou un service. Cependant à ressources finies et qui a priori vont décroître au cours du temps, il y a fort à parier que les moyens de la recherche nécessitent à l'avenir de prioriser les thématiques et de se concentrer sur de la recherche permettant de répondre aux impératifs de l'époque. Un véritable enjeu s'impose alors aux chercheurs en maths: les mathématiques sont utiles partout mais c'est souvent un combat pour en convaincre les décideurs, et surtout le grand public. J'espère que le lecteur profane aux mathématiques prendra autant de plaisir à parcourir ce manuscrit que j'en ai eu pour l'écrire, et que ces pages, à leur modeste échelle, lui permettront de trouver quelques éléments de réponse sur à quoi bon les maths peuvent bien servir.

## Remerciements

### Aux personnes qui font que cette thèse voit le jour

Tout d'abord je tiens à adresser mes sincères remerciements aux rapporteurs de cette thèse, Peter Ochs et Nicolas Papadakis, pour leurs relectures et retours ayant permis d'améliorer ce manuscrit de thèse. Je suis aussi reconnaissant envers les examinateurs, Marie Chabert, Laurent Condat et Julie Delon, d'avoir accepté de consacrer du temps à ma soutenance.

Cette thèse n'aurait bien entendu pas vu le jour sans le duo de choc que forment mes deux directeurs de thèse. Vous m'avez formé et appris tant de choses en recherche. Loin de vouloir énumérer toutes les leçons tirées, je vous remercie d'avoir toujours placé la rigueur scientifique au coeur de nos travaux, quitte à repousser une publication de plusieurs mois pour améliorer ou mieux comprendre des résultats. Merci Pierre pour ta clairvoyance dans les travaux de recherche et ton leadership, toujours apprécié que ce soit en recherche ou pour organiser des activités à l'extérieur du travail. Tu n'es pas le genre de directeur de thèse à laisser dépérir ses doctorants dans un coin et ta présence a été précieuse pendant ces trois années de thèse. Je te remercie aussi de t'être démené pour l'achat de machines de calcul qui nous permettent de mener à bien nos recherches. Fred, ton dynamisme

m’inspire et j’aime ton état d’esprit, toujours enclin à rigoler. Malgré tes apparences bordéliques, tu es probablement l’une des personnes les plus intègres que j’ai eu la chance de rencontrer. J’ai beaucoup appris de tes intuitions et de ta capacité à voir géométriquement les choses, avant en tant qu’étudiant de Master et ensuite pendant cette thèse. Pour conclure ce paragraphe, je rappelle une de tes citations et te remercie de m’avoir permis de relever ce défi: “Tu ne soutiendras pas ta thèse avant que j’ai soutenu mon HDR.” (Frédéric de Gournay, 23/02/2020).

### Aux collègues

Mes remerciements vont aussi aux autres matelots-doctorants du navire GMM, qui contre vents et marées, ont toujours été là pour rigoler de notre sort commun.

Clément, je ne sais pas qui de nous deux est le plus bavard et je garderai longtemps les souvenirs de nos nombreuses discussions sur tout et n’importe quoi. Je te remercie pour tes conseils, ta bienveillance et je suis persuadé que tu t’épanouiras dans l’enseignement supérieur et la recherche. Merci aussi à Jessica pour tes précieux conseils, d’avant la thèse jusqu’à ta soutenance. Je te souhaite de faire des choses qui te plaisent. Je remercie aussi tous les amis du bureau 120 pour leur soutien pendant le covid ainsi que pour les raclettes illégales dans le bureau en période de confinement ! Merci aussi à Morgane et Franck, et je vous souhaite bonne chance pour la suite. Evgeniia merci pour les discussions que nous avons eues et ton apport précieux par ta connaissance de la Russie, ce qui permet de mieux déchiffrer le monde. Les temps ne sont pas faciles et je te souhaite tout le meilleur pour la suite.

Au bureau d’en face, Julie et Iain. Merci Julie pour ces parties d’ultimatum en conférence (navré que nous n’ayons pas pu retrouver ton disque !). Merci Iain pour tout ces échanges intéressants et plus particulièrement sur les processus Gaussiens qui m’ont été bien utiles.

Merci à Mahmoud, partenaire de galère des procédures Adum. Hippolyte, avec qui je partage la passion de l’optimisation, je te souhaite bon courage pour ta dernière année de thèse, qui je suis sûr se déroulera très bien.

Une pensée aussi aux frères de thèse : Valentin, Corbi, Léo et Clément. Merci Valentin pour ton regard bienveillant et tes conseils. Corbi, conserve ton état d’esprit qui te rend si social. Léo, je te remercie pour tous tes conseils avant la thèse. Nathanaël, curieux et de bonne humeur, je te souhaite bon courage pour ta thèse. Et tente de canaliser Pierre dans son addiction aux échecs ! Je remercie aussi Wessim, toujours vaillant et plein d’énergie, pour l’expérience d’encadrement que m’a offert ton stage et je te souhaite bon courage pour la suite. Je pense à Manu pour les quelques échanges que nous avons eus, toujours constructifs. Je remercie aussi Jonas pour ta personnalité unique.

Je remercie aussi les membres du GMM. Tout d’abord Violaine pour m’avoir permis de rejoindre le bateau GMM en tant qu’étudiant. Merci à Charles pour les échanges sur des sujets aussi divers qu’intéressants. Merci à Aude pour ta bonne humeur éternelle et ton exceptionnelle pédagogie. Je remercie Romain de

ne pas m'en avoir voulu le jour où j'ai écrit sur une de mes copies "Je ne suis pas venu à l'INSA pour de tels calculs, on a des ordinateurs pour ça...". Loïc, c'est un plaisir d'écouter tes anecdotes, toujours plus abracadabrantes les unes que les autres. Merci Florent pour tous les afterworks au Dubliners dont tu es toujours l'instigateur. Je te remercie Béatrice pour ta bienveillance et ta sagesse inspirante. Une pensée pour Jean-Yves, si tu lis ces lignes depuis le soleil de Casablanca. Merci à Pascal, qui brille par ses qualités d'enseignant comme par son efficacité à gérer un département. Je remercie aussi Sandrine, une gestionnaire en or, toujours de bonne humeur et tu rends service avant même qu'on te demande. Tu es une des clés de voûte du GMM. Merci à Olivier M. de te démener avec toutes les procédures de recrutement. Je salue ton intégrité et tu es une des rares personnes avec qui il est possible de discuter d'absolument de tout sans forcément être d'accord pour peu que les arguments soient valables.

Je remercie aussi, surtout pour leur aide sur la fin de ma thèse, les femmes de l'ombre de l'école doctorale, toujours réactives, Marie-Hélène et Agnès.

### **Aux institutions**

Je remercie l'Institut de Mathématiques de Toulouse de m'avoir accueilli pendant ces trois années. L'allure des bâtiments contraste avec la concentration d'esprits brillants, parfois aussi décalés socialement qu'intelligents. Il est indéniable que les supports en calcul numérique fournis par l'IMT m'ont été très utiles...

La thèse a été ponctuée de périodes de confinements avec quelques allègements des contraintes et je remercie le gouvernement pour la distraction croissante procurée au fur et à mesure des crises, de par son amateurisme et le déni de démocratie dont ses représentants font preuve.

### **Aux amis**

Une très forte pensée à Bastien, je te dois tant. Merci pour ta joie de vivre mais aussi pour toutes tes mésaventures qui alimentent nos longues discussions. Je garde un excellent souvenir de nos séances de travail quand nous étions étudiants en Master et même si pendant ces trois années, nous nous sommes peu vus, j'espère sincèrement qu'on se retrouvera un jour, et pourquoi pas que l'on travaillera ensemble ! Dans tous les cas je compte sur toi pour m'inviter un jour sur ton yacht.

Alexis, tu sembles avoir trouvé ta voie et c'est toujours avec grand plaisir que je lis de tes nouvelles. Quand tu veux on peut se retrouver à Paris, Londres ou Toulouse.

Sans pouvoir lister tout le monde, je remercie aussi tous les camarades de l'INSA et du début de thèse d'avoir été là: Mathieu, Claire, Marc, Solène, Adrien.

Merci aux para'potes, Korantin et Louis, pour ces journées à la montagne. Quel plaisir de s'échapper, de voler comme un oiseau, et de se poser avec un grand sourire, content d'avoir volé, même 5 minutes. Merci à Adélie, "l'adulte responsable", pour avoir veillé sur nous pendant nos randonnées spatiales. Je remercie aussi Robin pour ces sessions monocycles improbables à la Daurade jusqu'à pas d'heure.

Je pense aussi au fablab, ce lieu plein de vie à l'atmosphère unique, dans lequel j'aime passer du temps depuis 5 ans et demi maintenant. Vous répondez toujours présents quand il s'agit de faire des projets complètement stupides. Je tiens en particulier à remercier Xavier, alias *ChampX*, de m'avoir dès le début initié au fablab. Merci aussi pour tes nouvelles et tes photos magnifiques du fin fond de l'Antarctique qui m'ont fait rêver. Merci à Guillaume de m'avoir formé et épaulé, au fablab mais aussi en dehors.

Bien entendu Romain tu as une place importante dans ces remerciements de thèse. Je te remercie pour les appels téléphoniques interminables au sujet de la thèse, de nos projets, de la vie et de tant d'autres choses. Des bancs de la classe prépa à la thèse, tu as toujours été là pour te lancer dans les projets les plus fous, mais aussi pour les conseils personnels. Travailler ensemble est un véritable plaisir, surtout quand il s'agit de planter un drone dans un arbre ou dans une ligne électrique de nuit ! Je garde espoir que nous puissions un jour résider dans la même ville.

Je te remercie Xavier, ami d'enfance. Je suis content qu'au fil des années, nous ne nous soyons pas perdus de vue et j'aime toujours faire des après-midi vélo avec toi (même si elles se font plus rares !). Merci pour ta curiosité et tes questions sur mes thématiques de recherche, qui forcent à expliquer simplement et à faire un travail de pédagogie.

Je remercie aussi Bianca pour ton calme et ta sagesse. Les journées randonnées avec toi m'ont été très ressourçantes sur cette fin de thèse.

## **A la famille**

Je dois aussi largement cette thèse, plus généralement mon parcours, et qui je suis à ma famille. Je ne serai jamais assez reconnaissant envers mon père et ma mère pour l'éducation qu'ils m'ont dispensée. Merci de ne m'avoir jamais bridé dans tous mes projets, y compris quand il s'agissait de transformer le salon en véritable atelier de bricolage. Merci d'avoir accepté seulement tard les écrans à la maison et de m'avoir offert mon premier ordinateur sur lequel j'ai commencé en informatique. Je remercie tout particulièrement ma mère de s'être acharnée pour que je me mette au travail, et pourtant enfant j'étais (et je suis toujours ?) têtue ! Tu peux être fière ! Une pensée aussi à Opa, à défaut que je sois médecin, tu as (pour le moment) un petit-fils docteur ! Merci aussi à Mamie de s'être autant occupée de moi avec Audrey. Tu peux toi aussi être fière du parcours de tes petits-enfants. Enfin merci à ma petite soeur Audrey, nous partageons tant de centres d'intérêts communs, et il semblerait bien que cela continue avec la recherche en maths.

# Contents

<b>Introduction</b>	<b>1</b>
<b>1 General introduction</b>	<b>13</b>
1.1 Introduction to Magnetic Resonance Imaging . . . . .	14
1.1.1 History overview . . . . .	14
1.1.2 The physics behind the acquisition . . . . .	16
1.1.3 A simplified model . . . . .	19
1.1.4 Different issues . . . . .	21
1.2 Image reconstruction for linear inverse problems . . . . .	27
1.2.1 Introduction to inverse problems for imaging . . . . .	27
1.2.2 Hand-crafted priors . . . . .	31
1.2.3 Training deep networks . . . . .	39
1.2.4 Learned priors . . . . .	41
1.3 Introduction à l'IRM en français . . . . .	52
1.3.1 Historique . . . . .	52
1.3.2 La physique derrière l'acquisition . . . . .	52
1.3.3 Un modèle simplifié . . . . .	57
<b>2 Spurious minimizers in non-uniform Fourier sampling optimization</b>	<b>59</b>
2.1 Introduction . . . . .	60
2.2 Notation . . . . .	62
2.3 Preliminaries . . . . .	62
2.3.1 The setting . . . . .	62
2.3.2 Elementary observations . . . . .	64
2.4 Theoretical issues . . . . .	65
2.4.1 Spurious minimizers . . . . .	65
2.4.2 Numerical illustration of Theorem 1 . . . . .	68
2.4.3 Flatness for high frequencies . . . . .	70
2.5 Escaping the minimizers . . . . .	71
2.5.1 The effect of using a large dataset . . . . .	71
2.5.2 Stochastic gradient descent . . . . .	73
2.5.3 Variable metric . . . . .	73
2.5.4 Numerical illustrations . . . . .	73
2.6 Conclusion . . . . .	75
2.7 Proofs . . . . .	77
2.7.1 Proof of Proposition 11 . . . . .	77
2.7.2 Proof of Theorem 1 . . . . .	78
2.7.3 Proof of Proposition 12 . . . . .	79
2.7.4 Proof of Theorem 2 . . . . .	80

<b>3</b>	<b>Bayesian Optimization of Sampling Densities in MRI</b>	<b>83</b>
3.1	Introduction . . . . .	84
3.1.1	Some sampling theory . . . . .	85
3.1.2	Data-driven sampling schemes . . . . .	86
3.1.3	Our contribution . . . . .	88
3.2	The proposed approach . . . . .	88
3.2.1	Preliminaries . . . . .	88
3.2.2	The challenges of sampling scheme optimization . . . . .	90
3.2.3	Regularization and dimensionality reduction . . . . .	91
3.2.4	The optimization routine . . . . .	96
3.3	Numerical experiments and results . . . . .	99
3.3.1	The experimental setting . . . . .	99
3.3.2	Choosing a kernel for the discrepancy . . . . .	101
3.3.3	Bayesian optimization: database size and numerical complexity	102
3.3.4	Comparing optimization routines for the total variation re- constructor . . . . .	102
3.3.5	Comparing optimization routines for a neural network recon- structor . . . . .	106
3.4	Conclusion . . . . .	108
3.5	Implementation details . . . . .	110
3.5.1	TV reconstruction algorithm . . . . .	110
3.5.2	The unrolled neural network . . . . .	110
3.5.3	Training the reconstruction network for a family of operators	111
3.5.4	Joint optimization . . . . .	111
3.5.5	Computational details . . . . .	112
3.6	Sampling density and Shannon’s condition . . . . .	114
3.6.1	How to control the sampling density? . . . . .	114
3.6.2	Performance variance for a fixed sampling density . . . . .	115
3.7	Resampling from estimated density . . . . .	115
3.7.1	Density estimation . . . . .	115
3.7.2	Resampling . . . . .	117
<b>4</b>	<b>Training Adaptive Reconstruction Networks for Blind Inverse Problems</b>	<b>119</b>
4.1	Introduction . . . . .	120
4.2	Related works . . . . .	123
4.3	Preliminaries . . . . .	126
4.3.1	The forward models . . . . .	127
4.3.2	The model-based reconstruction networks . . . . .	128
4.4	Training on a family of operators . . . . .	129
4.4.1	What is different? . . . . .	130
4.4.2	Choosing distributions of operators . . . . .	130
4.5	Solving blind inverse problems . . . . .	131
4.6	Numerical experiments . . . . .	132

4.6.1	The setting . . . . .	132
4.6.2	The benefits of training on a family in MRI . . . . .	133
4.6.3	Blind inverse problems . . . . .	136
4.7	Conclusion . . . . .	139
<b>5</b>	<b>Adaptive scaling of the learning rate by second order automatic differentiation</b>	<b>147</b>
5.1	Introduction . . . . .	148
5.1.1	Foreword . . . . .	149
5.1.2	Related works . . . . .	150
5.1.3	Our contributions . . . . .	152
5.2	Rescaling the learning rate . . . . .	152
5.2.1	Presentation and guideline for rescaling . . . . .	152
5.2.2	Analysis of the rescaling . . . . .	154
5.3	Computing the curvature . . . . .	156
5.3.1	Main results . . . . .	156
5.3.2	Proof of Algorithm 4 . . . . .	157
5.3.3	Proof of Theorem 3 . . . . .	160
5.4	Numerical experiments . . . . .	163
5.4.1	Convergence/exploration trade off . . . . .	163
5.4.2	Hyperexploration mode . . . . .	166
5.4.3	Hyperconvergence mode . . . . .	167
5.4.4	Influence of the averaging factor of the curvature . . . . .	168
5.5	Conclusion and discussion . . . . .	170
5.6	Second order computation . . . . .	170
5.6.1	Hessian-vector dot product . . . . .	170
5.6.2	Structure of the layers . . . . .	171
5.7	Description of the numerical experiments . . . . .	172
5.8	Additional numerical experiments . . . . .	175
5.8.1	Dealing with momentum . . . . .	175
5.8.2	Batch reduction on CIFAR . . . . .	177
5.8.3	Effect of the $L^2$ regularization . . . . .	178
5.8.4	Comparison with BB and Robbins-Monro . . . . .	178
	<b>Conclusion</b>	<b>183</b>
	<b>A Résumé en Français</b>	<b>187</b>
	<b>Bibliography</b>	<b>189</b>





# Introduction

## Introduction in english

The progresses made in imaging since the second half of the XX<sup>th</sup> century are impressive. They are the result of advances in various fields, including the design of hardware equipment, the increase in computing capacity, theoretical advances with the modelling of inverse problems, information theory and optimization. Today, all these developments are combined in imaging tools such as Magnetic Resonance Imaging (MRI), ultrasound imaging, tomography imaging and microscopy. The use of all these methods falls within the framework of *computational imaging*.

Image processing and especially image reconstruction has a long history, and the first work on reconstruction for inverse problems dates back to the 1970s. Throughout the second half of the XX<sup>th</sup> century, the methods developed were limited by computational power. A simple model evaluation was expensive and could take up to several minutes. The methods were therefore based on a few model evaluations such as the so-called *back projection* methods. These reconstructions used simple approximations of the inverse, computable affordable, as the adjoint of the acquisition model. These inversion models were eventually corrected by taking into account the physics if available. The methods of *filtered back-projection* in *computed tomography* are a perfect example [Ramachandran 1971]. It has also been used in MRI for uniform sampling [Lauterbur 1973].

These methods were then replaced by more efficient reconstructions, based on variational formulations and solved with iterative algorithms. These variational formulations involve a regularization term. The most common regularization, which has been widely used in the field of image processing for decades, is that of Tikhonov, which consists of penalizing the data attachment term by a term of the form  $\mathcal{R}(x) = \|Lx\|_2^2$  with  $L$  a linear operator. This regularization has been the subject of numerous studies.

The development of variational formulations has been made possible by advances in algorithmic theory, making it possible to compute operations with a  $O(N^2)$  complexity in  $O(N \log(N))$  through *divide and conquer* algorithms and factorization methods among others. The most famous example is that of the FFT (*Fast Fourier Transform*) which is used in many problems such as Fourier sampling, tomography or deblurring.

At the end of the XX<sup>th</sup> century, engineers have realized that the number of measurements could be drastically reduced, allowing significant gains in acquisition time, while maintaining good reconstruction quality through the use of non-linear methods. These reconstructors also raised theoretical questions from a mathematical point of view. The formal framework then appeared with compressed acquisition at the beginning of the XXI<sup>th</sup> century giving theoretical guarantees on the reconstruction of the signals. In parallel, advances in convex optimization allowed the

development of efficient algorithms, making the use of these non-linear formulations possible in practice thanks to FISTA [Beck 2009] and *splitting* methods such as ADMM [Fortin 2000, Boyd 2011] or Douglas-Rachford [Eckstein 1992].

In parallel to these works, learning has developed significantly. Initially derived from neuroscience, the field has been enriched by other communities, such as theoretical computer science with automatic differentiation, information theory, optimal transport and stochastic optimization. This field now represents an important part of the research with 3.9% of the articles in 2019 [Zhang 2021a].

Imaging has also been influenced and it has been benefiting for the last ten years from the revolutions offered by learning, by first tackling the question of image reconstruction. The approaches have attempted an appealing task in terms of modelling: get free from the physical models, which are sometimes imperfect, and learn them from a set of data [Zhu 2018]. This approach has recently shown instability problems. Hybrid approaches that focus the use of learned models on specific tasks were developed in parallel. They use learned methods on problems for which physics-driven methods perform poorly: removing noise, removing artefacts related to an operator or removing the residue from a numerical solver. These approaches are now state-of-the-art and they have led to significant gains in image quality. Among these methods that combine physics and learned models, several approaches stand out, some of which can be cited: *unrolled* networks, *Plug & Play* approaches or PINNs (*Physics-Informed Neural Networks*).

After improving image reconstruction, learning has been extended to all the elements involved in an imaging device. It is now applied to the design of acquisition models which parameters are optimized to reduce acquisition times or improve image quality. Optimized jointly with the reconstruction method, this is called *co-design*. This disruption opens up challenging aspects: some of the theories previously established in the XX<sup>th</sup> and early XXI<sup>st</sup> centuries for classical reconstruction methods are no longer valid for learned algorithms and this opens the way to a new field of mathematics. This contribution of learning has of course also been made possible by the democratization of massively parallel architectures, which costs have decreased and which power has continued to increase.

The success of these learned models is also due to advances in stochastic optimization. The large number of parameters involved in the models has made it necessary to develop optimizers that can handle very high dimensional models. A step back of fifty years shows the vertiginous progress made both from a theoretical and hardware point of views with the computing architectures. Indeed, the high dimension used to qualify spaces with a dimension of the order of a hundred; today the very high dimensional models go up to a hundred billion parameters. The works in this field have led to the development of optimizers that work more or less well in practice and with varying theoretical guarantees. Most of them involve hyperparameters which values must be adjusted to train the models correctly, but above all, from a practical point of view, to obtain the highest possible performance. The increasing use of learning in imaging alone and the deployment of increasingly numerically heavy models raise the issue of reducing the numerical cost of these

trainings.

### **What are the current challenges in computational imaging?**

Several areas of improvement are at the heart of the next technical challenges in computational imaging:

**Stability** Currently the learned methods are very dependent on the training setting. Improving the stability of the reconstructors is a major challenge to allow the methods to move from the academic setting to concrete and certifiable medical applications. The improvement of co-design methods is also a challenge to use improve imaging techniques in specific applications.

**Automate** The hyperparameters involved in the training of the models have a major impact on the performance of the methods. These hyperparameters often require manual adjustment, which is a bottleneck both in terms of resources used and in terms of human time spent on unrewarding tasks. This point is not specific to computational imaging and it affects learning in general. Automating training methods is of importance for future researchs.

**Reducing the numerical cost of training and the dependence to large training datasets** Training models in computational imaging can take days to several weeks. In practice, this makes the application to real problems prohibitive when the model has to be trained again for each different case of application. Reducing the numerical cost of training is a crucial issue to allow improvements in computational imaging to be applied in industry. The large size of the models can also be a problem, as a large volume of data is needed to train them. For many applications, particularly in the biomedical field, datasets are limited and computational imaging will require a reduced reliance on this large volume of data.

### **Contributions**

Guided by the above issues, I have dealt in this thesis with some aspects of computational imaging with a focus on the methodology. The objective is to propose improvements in the field of co-design, in image reconstruction and in the optimization of learned models with recent tools of learning and parallel computing. Throughout this manuscript, several application cases are studied: computed tomography (CT), deblurring and a particular attention is given to MRI. The contributions are summarized below.

#### **Co-design**

Recently several works have been proposed for off-the-grid optimization of Fourier schemes for optimal sampling in MRI. From a practical point of view, the iterates

produced require many iterations to converge and are highly dependent on the initialization.

#### Contributions

- An analysis of the spurious minimizers in optimal and non-uniform Fourier sampling schemes is given (Chapter 2).
- Motivated by existing works in sampling theory and using Bayesian optimization, a method to globalize the convergence for MRI sampling schemes is proposed (Chapter 3).

### Reconstruction using neural networks

The unrolled networks are trained for specific applications and show experience severe instabilities as soon as the forward operator of the data attachment term is changed.

#### Contributions

- We show that not only does the network gain in stability, but for well-constructed reconstructors there is no significant loss in training a reconstruction method on a set of operators.
- Training on a family of operators allows solving several blind inverse problems using unrolled neural networks.

### Neural networks optimization

Some of the recent works in stochastic optimization consists in applying a corrective term on the step size of optimization methods for deep networks. These methods either require a step given by the user to the algorithm, or compute a step automatically but only at each pass over the data set.

#### Contributions

- An algorithm to compute the curvature of a composition of functions at a lower cost than existing methods is proposed.
- A physical interpretation of the step of this algorithm is given.
- This allows introducing a method that performs an automatic rescaling of the step at each iteration.

### Outline of this manuscript

This manuscript is divided into 5 chapters. The first chapter is an introduction to computational imaging and it illustrates through the case of MRI the devel-

opments that have guided this field. It also contains a pedagogical introduction to inverse problems and the associated reconstruction methods. This introduction traces the early linear reconstruction methods, the emergence of non-linear methods and recent advances in reconstruction methods that are learned with neural networks. The following chapters are based on different publications or preprints and, although links are made between the different chapters, they can be read independently of each other. The second chapter deals with spurious minimizers in the optimization of non-uniform Fourier sampling schemes. The motivation is the optimization of MRI sampling schemes for a chosen reconstruction method and for a specific image database. This chapter shows that this type of problem has a combinatorial number of minimizers that can disappear with the large number of images but that classical MRI databases do not contain enough images to expect this phenomenon to appear. The third chapter proposes a method to globalize the convergence for the optimization of Fourier sampling schemes. This drastically reduces the numerical cost of the optimization while maintaining a significant gain in the image quality. The fourth chapter deals with the training of neural networks that are adaptive to changes in the physics of the acquisition. This formalism allows to solve several blind inverse problems. Finally, the fifth chapter tackles the optimization of neural networks. It proposes a method to scale the learning rate and this opens the way to automate the choice of the hyperparameters during the training phase.

## Introduction en français

Les progrès réalisés en imagerie depuis la seconde moitié du XX<sup>ème</sup> siècle sont impressionnants. Ils relèvent de progrès dans des domaines variés, allant de la conception matérielle des appareils, de l'augmentation des capacités de calcul aux avancées théoriques avec la modélisation des problèmes inverses, la théorie de l'information et l'optimisation. Aujourd'hui, toutes ces évolutions se rejoignent et sont combinées sur des outils d'imagerie comme en Imagerie par Résonance Magnétique (IRM ou MRI en anglais), en imagerie à ultrasons, en imagerie par tomographie ou bien en microscopie. L'ensemble de ces méthodes rentre dans le cadre de *l'imagerie computationnelle*.

Le traitement d'images et plus particulièrement la reconstruction ont une longue histoire, et les premiers travaux sur la reconstruction d'images dans le cadre de problèmes inverses datent des années 1970. Durant toute la seconde moitié du XX<sup>ème</sup> siècle, les méthodes développées étaient limitées par la puissance de calcul, une simple évaluation du modèle étant coûteuse et pouvant prendre jusqu'à plusieurs minutes. Les méthodes reposaient donc sur quelques évaluations du modèle comme les méthodes dites de *retroprojection*. Ces reconstructions utilisaient des approximations simples de l'inverse, facilement calculables, comme l'adjoint du modèle d'acquisition. Ces modèles d'inversion étaient éventuellement corrigés en prenant en compte les éléments de physique à disposition. Les méthodes de *filtered back-projection* en *computed tomography* en sont un parfait exemple [Ramachandran 1971]. Cela a aussi été utilisé en IRM pour de l'échantillonnage uniforme [Lauterbur 1973].

Ces méthodes ont ensuite été remplacées par des reconstructions plus performantes, reposant sur des formulations variationnelles et résolues avec des algorithmes itératifs. Ces formulations variationnelles font intervenir un terme de régularisation. La régularisation la plus courante et qui a largement été utilisée en traitement d'image pendant des décennies est celle de Tikhonov, qui consiste à pénaliser le terme d'attache aux données par un terme de la forme  $\mathcal{R}(x) = \|Lx\|_2^2$  avec  $L$  un opérateur linéaire. Le choix de cette régularisation a fait l'objet de nombreuses études.

Le développement de ces formulations variationnelles a été permis grâce aux progrès en algorithmique, permettant de calculer des opérations de complexité  $O(N^2)$  en  $O(N \log(N))$  à travers des méthodes de *diviser pour régner* et de factorisation entre autres. L'exemple le plus parlant est celui de la FFT (*Fast Fourier Transform*) qui est utilisé dans de nombreux problèmes comme l'échantillonnage de Fourier, la tomographie ou le défloutage.

A la fin du XX<sup>ème</sup> siècle, les ingénieurs se sont aperçus que le nombre de mesures pouvait être drastiquement réduit, permettant des gains importants sur les temps d'acquisition, et tout en conservant une bonne qualité de reconstruction grâce à l'utilisation de méthodes non linéaires. Cette utilisation a aussi suscité des questions théoriques d'un point de vue mathématique. L'acquisition comprimée a ensuite fourni un cadre théorique au début du XXI<sup>ème</sup> avec des garanties sur la construc-

tion des signaux. En parallèle, les avancées en optimisation convexe ont permis le développement d’algorithmes efficaces, rendant l’utilisation de ces formulations non linéaires possible en pratique grâce à FISTA [Beck 2009] et les méthodes de *splitting* comme l’ADMM [Fortin 2000, Boyd 2011] ou Douglas-Rachford [Eckstein 1992].

En parallèle de ces travaux en optimisation, l’apprentissage s’est développé de façon importante. Au départ dérivé des neurosciences, le domaine s’est enrichi grâce à d’autres communautés, comme l’informatique théorique avec la différentiation automatique, la théorie de l’information, le transport optimal et l’optimisation stochastique. Ce domaine représente aujourd’hui une part importante de la recherche avec 3.9% des articles qui y sont dédiés en 2019 [Zhang 2021a].

L’imagerie n’a pas été épargnée par cette vague et bénéficie depuis une dizaine d’années des révolutions offertes par l’apprentissage, en attaquant d’abord la question de la reconstruction des images. Les approches ont notamment essayé une tâche attrayante en terme de modélisation : s’affranchir des modèles physiques, parfois imparfaits, et les apprendre à partir d’un ensemble de données [Zhu 2018]. Cette approche a récemment montré des problèmes de stabilité. Des approches hybrides qui concentrent l’utilisation de modèles appris sur des tâches spécifiques ont en parallèle été développées. Elles utilisent des méthodes apprises sur des problèmes pour lesquels les méthodes guidées par la physique ont de moindres performances : retirer le bruit, retirer des artefacts liés à un opérateur ou à une méthode de résolution imparfaite. Ces approches représentent maintenant l’état de l’art et ont permis des gains significatifs sur la qualité des images traitées. Parmi ces méthodes mélangeant la physique et les modèles appris, plusieurs approches se distinguent, dont quelques-unes peuvent être citées : les réseaux *unrolled*, les approches *Plug & Play* ou encore les PINNs (*Physics-Informed Neural Networks*).

Après avoir permis d’améliorer l’étape de reconstruction de l’image, l’apprentissage s’est étendu à l’ensemble des éléments qui interviennent dans un appareil d’imagerie. Désormais, il est appliqué jusqu’à la conception de modèles d’acquisition dont les paramètres sont optimisés pour réduire les temps d’acquisition ou améliorer la qualité des images. Optimisés conjointement avec la méthode de reconstruction, cela s’appelle le *co-design*. Cette évolution pose des questions stimulantes : une partie des théories précédemment établies au XX<sup>ème</sup> et début du XXI<sup>ème</sup> siècles pour des méthodes de reconstruction classiques ne sont plus valides pour des algorithmes appris et cela ouvre la voie à un nouveau champ des mathématiques. Cette vague de l’apprentissage a bien entendu aussi été rendue possible grâce à la démocratisation des architectures massivement parallèles dont les coûts n’ont cessé de diminuer et la puissance n’a cessé d’augmenter.

Le succès de ces modèles appris est aussi dû aux avancées en optimisation stochastique. Le grand nombre de paramètres qui interviennent dans les modèles a nécessité de développer des optimiseurs capables de gérer la très grande dimension [Tieleman 2012, Kingma 2015]. Un recul d’une cinquantaine d’années montre le vertigineux progrès fait tant d’un point de vue théorique, que matériel avec les architectures de calcul, la grande dimension qualifiait autrefois des espaces de dimension de l’ordre de la centaine; aujourd’hui les modèles en très grande dimension vont



jusqu'à la centaine de milliards de paramètres. Les travaux ont mené au développement d'optimiseurs marchant plus ou moins bien en pratique avec des garanties théoriques variables. Le plupart d'entre eux font intervenir des hyperparamètres dont les valeurs doivent être ajustées pour entraîner correctement les modèles, mais surtout, d'un point de vue pratique, pour obtenir des performances les plus élevées possibles. L'utilisation croissante de l'apprentissage rien que dans l'imagerie et le déploiement de modèles de plus en plus lourds numériquement posent la question de réduire le coût numérique de ces entraînements.

### Quels sont les enjeux actuels en imagerie computationnelle ?

Plusieurs axes d'amélioration sont au coeur des prochains défis techniques à relever en imagerie computationnelle :

**Stabilité** Actuellement les méthodes apprises pour des cas d'application spécifique sont très dépendantes du cadre d'entraînement. Améliorer la stabilité des reconstruteurs est un enjeu majeur pour permettre aux méthodes de passer du cadre universitaire à des applications médicales concrètes et certifiables. L'amélioration des méthodes de co-design est aussi un enjeu pour mettre en place des techniques d'imagerie spécifiques aux cas d'application étudiés.

**Automatisation** Les hyperparamètres entrant en jeu lors de l'entraînement des modèles impactent grandement la performance des méthodes. Ces hyperparamètres demandent souvent un ajustement manuel, ce qui est un point bloquant à la fois en terme de ressources utilisées, qu'en terme de temps humain passé sur des tâches peu gratifiantes. Ce point touche des domaines beaucoup plus larges que l'imagerie computationnelle et automatiser les méthodes d'entraînement sera au coeur des recherches futures.

### Réduction du temps d'entraînement et du volume de données

L'entraînement de modèles en imagerie computationnelle peut prendre de plusieurs jours à plusieurs semaines. En pratique, cela rend l'application à des problèmes réels prohibitif quand le modèle doit être réentraîné pour chaque cas d'application. Réduire les temps d'entraînement est un enjeu crucial pour permettre l'application dans l'industrie des méthodes développées. La grande dimension des modèles peut aussi poser problème puisque pour les entraîner, il est nécessaire d'avoir à disposition un grand volume de données. Pour bon nombre d'applications, et particulièrement dans le domaine biomédical, les jeux de données sont limités et l'imagerie computationnelle nécessitera de réduire la dépendance à ce grand volume de données pour l'entraînement des modèles.

## Contributions

Guidé par les enjeux précédemment exposés, j'ai traité dans cette thèse quelques aspects de l'imagerie computationnelle avec un accent sur la méthodologie. L'objectif est de proposer des améliorations dans le domaine du co-design, de la reconstruction et de l'optimisation de modèles appris avec des outils récents de l'apprentissage et du calcul parallèle. Tout au long de ce manuscrit, plusieurs cas d'application sont étudiés : la tomographie computationnelle (*computed tomography*, CT en anglais), le défloutage et une attention particulière est portée à l'IRM. Les contributions sont résumées ci-après.

## Co-design

Récemment plusieurs travaux ont été proposés pour l'optimisation hors grille de points dans l'espace de Fourier pour de l'échantillonnage optimal en IRM. D'un point de vue pratique, les itérées produites demandent beaucoup d'itérations pour converger et sont très dépendantes de l'initialisation.

### Contributions

- Une analyse des minimiseurs dans le cadre de l'échantillonnage optimal non-uniforme de Fourier (Chapitre 2).
- Motivé par les travaux existants reposant sur la théorie de l'échantillonnage, et à l'aide de l'optimisation bayésienne, une méthode de globalisation de l'optimisation de schémas pour l'IRM est proposée (Chapitre 3).

## Reconstruction par réseaux de neurones

Les réseaux *unrolled* sont entraînés pour des cas d'application spécifique et montrent une forte instabilité dès lors que l'opérateur du terme d'attache aux données est changé.

### Contributions

- Nous montrons que non seulement le réseau gagne en stabilité, mais que pour des reconstituteurs bien construits, il n'y a pas de perte significative à entraîner une méthode de reconstruction sur un ensemble d'opérateurs.
- L'entraînement sur une famille d'opérateurs permet de résoudre plusieurs problèmes inverses aveugles avec des reconstituteurs *unrolled*.

## Optimisation de réseaux de neurones

Une partie des travaux récents en optimisation stochastique consiste à appliquer un terme correctif sur le pas de méthodes d'optimisation pour des réseaux profonds. Ces méthodes nécessitent un pas donné par l'utilisateur à l'algorithme, ou calculent un pas automatiquement mais seulement à chaque passage sur l'ensemble des données.

### Contributions

- Un algorithme permettant de calculer la courbure d'une composition de fonctions en un coût plus faible que les méthodes existantes est proposé.
- Dans le cadre de cet algorithme, un paramètre de pas intervient dont une interprétation physique est donnée.
- Cela permet d'introduire une méthode réalisant une mise à l'échelle automatique du pas à chaque itération.

## Plan du manuscrit

Ce manuscrit est divisé en 5 chapitres. Le premier chapitre est une introduction à l'imagerie computationnelle et illustre à travers le cas de l'IRM les évolutions ayant guidé ce domaine. Il contient aussi une introduction pédagogique aux problèmes inverses et aux méthodes de reconstruction associées. Cette introduction retrace les premières méthodes de reconstruction linéaires, l'apparition de méthodes non linéaires et les méthodes récentes de reconstruction apprises à l'aide de réseaux de neurones. Les chapitres suivants reposent sur différentes publications ou prépublications et, bien que des liens soient faits dans ces chapitres entre eux, ils peuvent se lire indépendamment les uns des autres. Le second chapitre traite des minimiseurs parasites dans l'optimisation de schémas d'échantillonnage de Fourier dont la motivation est l'optimisation de schémas d'échantillonnage pour l'IRM pour une méthode de reconstruction choisie et pour une base de données d'images spécifique. Ce chapitre montre que ce type de problème a un nombre combinatoire de minimiseurs qui peuvent disparaître avec le grand nombre d'images mais que les bases de données classiques d'IRM ne contiennent pas assez d'images pour espérer voir apparaître ce phénomène. Le troisième chapitre propose une méthode de globalisation de la convergence pour l'optimisation de schémas de Fourier. Cela permet de grandement réduire le coût numérique de l'optimisation tout en conservant un gain dans l'amélioration des images. Le quatrième chapitre traite de l'entraînement de réseaux de neurones *unrolled* adaptatifs à des changements dans la physique de l'acquisition. Ce formalisme permet de résoudre plusieurs problèmes inverses aveugles. Enfin, le cinquième chapitre traite des méthodes d'optimisation pour des réseaux de neurones de manière générale. Il propose une méthode permettant d'introduire une mise à l'échelle du pas pour l'optimisation de réseaux de neurones. Cela ouvre la voie à une automatisation du choix des hyperparamètres

lors de l'entraînement.

## List of Publications

### Published

- de Gournay, F., **Gossard, A.**, & Weiss, P. (2020, December). Off-the-grid data-driven optimization of sampling schemes in MRI. In ITWIST 2020.
- **Gossard, A.**, Weiss, P., & de Gournay, F. (2022). Spurious minimizers in non uniform Fourier sampling optimization. *Inverse Problems*, 38(2022), 105003.

### Submitted

- **Gossard, A.**, de Gournay, F., & Weiss, P. (2022). Bayesian Optimization of Sampling Densities in MRI.
- **Gossard, A.**, & Weiss, P. (2022). Training Adaptive Reconstruction Networks for Blind Inverse Problems.
- de Gournay, F., & **Gossard, A.** (2022). Adaptive scaling of the learning rate by second order automatic differentiation.

# General introduction

---

**Résumé** Ce chapitre introduit les principaux outils intervenant dans les problèmes inverses à travers l'exemple de l'imagerie par résonance magnétique (IRM). La première partie traite du fonctionnement d'un scanner IRM, elle donne une description de l'histoire des scanners IRM, un modèle simplifié et les défis computationnels pour la reconstruction d'images IRM ainsi que pour les problèmes de co-design. La seconde partie est une introduction pédagogique aux problèmes inverses et aux méthodes de reconstruction qui y sont associées avec des illustrations numériques et des liens renvoyant vers les codes Python. Elle explique les principales tendances dans le choix de bons a priori pour régulariser les problèmes inverses, leurs avantages et inconvénients. Les algorithmes d'optimisation utilisés pour entraîner les réseaux de neurones profonds sont aussi présentés et quelques enjeux autour du choix des hyperparamètres. Des questions ouvertes sont posées, dont certaines sont traitées dans les chapitres suivants.

**Abstract** This chapter gives the main tools involved in inverse problems for imaging through the example of Magnetic Resonance Imaging (MRI). The first part deals with the operation of an MRI scanner, it gives an history overview, a simplified model and the computational challenges in MR reconstruction as well as in co-design problems. The second part is a pedagogical introduction to inverse problems and their associated reconstruction methods with numerical illustrations and links to Python code. It explains the main trends in the choice of good priors to regularize inverse problems, their pros and cons. It also introduces the optimization algorithms used to train deep neural networks and some of the issues with the choice of their hyperparameters. Open questions are stated and some of them are tackled in the following chapters.

## Contents

---

<b>1.1</b>	<b>Introduction to Magnetic Resonance Imaging</b>	<b>14</b>
1.1.1	History overview	14
1.1.2	The physics behind the acquisition	16
1.1.3	A simplified model	19
1.1.4	Different issues	21
<b>1.2</b>	<b>Image reconstruction for linear inverse problems</b>	<b>27</b>
1.2.1	Introduction to inverse problems for imaging	27
1.2.2	Hand-crafted priors	31
1.2.3	Training deep networks	39
1.2.4	Learned priors	41
<b>1.3</b>	<b>Introduction à l'IRM en français</b>	<b>52</b>
1.3.1	Historique	52
1.3.2	La physique derrière l'acquisition	52
1.3.3	Un modèle simplifié	57

---

## 1.1 Introduction to Magnetic Resonance Imaging

This first part gives the operation of an MRI scanner, a simplified model and the computational challenges. The two first sub-sections are self-contained and the hurried reader can skip it.

### 1.1.1 History overview

Over the past four decades, MRI scanners have become a crucial tool in the diagnosis of many diseases and in cognitive research. Due to their non-invasive and harmless nature, they have enabled many scientific advances by allowing to image the interior of a body without the need for surgery. The MRI scanners used in medical imaging today are the culmination of a huge amount of works, the theoretical foundations of which came from discoveries in Nuclear Magnetic Resonance (NMR) in the first half of the 20<sup>th</sup> century. The first works to tackle the issue of imaging biological tissue were published in the 1970s. Raymond Vahan Damadian was the first to propose to improve cancer diagnosis by using a device to characterize healthy tissue from tumor tissue. This method was based on the different magnetic response of cancerous tissue from that of healthy tissue. Subsequently, other works addressed the problem of observing an image of the human body [Lauterbur 1973, Mansfield 1977]. It was for these works, which made it possible to localize information in space, that Paul Lauterbur and Peter Mansfield were awarded the 2003 Nobel Prize in Physiology or Medicine. From 1975 onwards, the technology used in contemporary MRI

scanners was introduced by Richard Ernst. It is based on the frequency and the phase encoding. This technology, which is described in the Section 1.1.2, makes it possible to construct acquisition sequences locating information in 3D. Finally, the following decade saw the commercialization of the first scanners. For research which foundations are complex, the industrial-scale application of scanners is remarkable, with only 10 years separating the founding work from its application in commercial scanners.



Figure 1.1: Raymond Vahan Damadian presenting his invention at a press conference in 1977. Credit: Copyright Bettmann/Corbis /AP Images (<http://cen.acs.org>)



(a) A conventional 1.5T MRI scanner (Philips scanner). Credit: Jan Ainali, (CC BY 3.0), <https://commons.wikimedia.org/wiki/File:MRI-Philips.JPG>



(b) A new generation of MRI scanner that fits on wheels and use a low power magnetic field of 64mT (Hyperfine Swoop scanner). Credit: Hyperfine <https://hyperfine.io/>

Figure 1.2: Two types of contemporary MRI scanners.



### 1.1.2 The physics behind the acquisition

The operation of MRI scanners is based on subtle concepts of Nuclear Magnetic Resonance. Although there are many ways of performing the acquisition, the simplest way of doing it is outlined below. Some details are omitted, the aim being to introduce the simplified mathematical framework. For more information, the excellent website “Questions and Answers in MRI”<sup>1</sup> gives all the details and the physics associated with MRI. A good explanation of the exploitation of the resonance phenomenon in MRI is also given in [Idy-Peretti 2009].

Before describing how an MRI scanner works, let us recall some basic principles of Nuclear Magnetic Resonance. Nuclei have a spin, an electromagnetic property that describes the orientation of their polarization. This spin is represented as a vector in 3D and its strength has different states which, at the scale of these particles, are all discrete. When a magnetic field is applied to nuclei, the spins align themselves along the direction of the magnetic field. It is this effect that is widely exploited by MRI scanners to measure the response of a sample to magnetic field stresses.

MRI scanners use the magnetization of hydrogen atoms (single proton) because they are present in large numbers in living tissue, and their proportion varies according to the nature of the tissue. Once the density of these atoms can be measured, it is possible to characterize the tissues making up an image. The whole purpose of an MRI scan is to find the density of the protons which are excited. These protons are excited with magnetic fields and it is the response over the whole, or part of the volume, that is measured. Of course, the response of the entire volume is of no use since it is not possible to spatially localize the information. This is where all the engineering and complexity of MRI scanners enters the game to allow the image associated with the imaged volume to be recovered from the measured signals. The remainder of this sub-section is dedicated to explaining the main components involved in the physical phenomenon exploited by MRI scanners, and to give a schematic explanation of how the signals are measured.

#### 1.1.2.1 Main components of an MRI scanner

In a simplified form, an MRI scanner has 4 main components: a main magnetic field, gradient coils, radiofrequency (RF) coils and receiving antennas.

**Primary magnetic field** A main magnetic field, denoted  $B_0$ , is applied to the whole scanned volume and it is aligned in the  $e_z$  direction. This uniform magnetic field aligns the spins in the  $e_z$  direction. Its typical strength is of the order of 1T on conventional scanners and on a new generation of scanners it is decreased to  $\sim 60\text{mT}$  (see Figure 1.2).

---

<sup>1</sup><https://www.mriquestions.com/>

**Gradient coils** There are several groups of gradient coils – in a simplified form three, each corresponding to an axis. The magnetic fields induced by each of these groups of coils is aligned with the corresponding axis and the strength varies along each axis (typically 15 to 45mT/m on conventional scanners). The  $z$  gradient coils create an additional magnetic field aligned with  $B_0$  and with an amplitude increasing linearly along the  $z$  axis. The superposition of these two fields gives a total field with an intensity that varies according to the  $z$  axis. The next section explains how this varying magnetic field allows selecting a slice to excite. This is called *frequency encoding*. The other gradient coils on the  $x$  and  $y$  axes allow the phase to be modified. These coils are involved in the *phase encoding*, as opposed to the *frequency encoding* performed by the  $z$  gradient coils.

**Radiofrequency coils** The RF coils generate an oscillating magnetic field  $B_1$  which is orthogonal to the  $B_0$  magnetic field. These radiofrequency coils emit signals for a short period of time to change the orientation of the protons spin. The field  $B_1$  oscillates at a frequency  $\omega_{RF}$ . The strength of such field is of the order of  $\sim 10\text{mT}$ .

**Receiving antennas** Traditionally, RF coils also acted as receivers to measure the response. They are now tending to be replaced by coils dedicated to receiving the signal and are placed all around the volume to be scanned.

### 1.1.2.2 Acquisition

Now that the various elements involved in the signal acquisition of an MRI scanner have been introduced, we describe how the acquisition is performed. Figure 1.3 gives schematically the main elements of an MRI scanner and an example of a phase shift profile in the  $(x, y)$  plane.

**Slice selection** First an MRI scanner selects a slice to image. It takes advantage of a phenomenon called *precession* which occurs when the spins are oscillating and that they are subject to a constant magnetic field. This precessional movement is similar to a spinning top that spins fast but which axis of rotation varies over time, describing circles. The frequency of this motion is called the Larmor frequency. It is proportional to the strength of the magnetic field through the relation:

$$\omega_L = \gamma B \tag{1.1}$$

with  $\omega_L$  the Larmor frequency [Hz],  $\gamma$  the gyromagnetic ratio [Hz/T] which depends on the type of nuclei that is excited and  $B$  the magnetic field strength [T]. As the intensity of the magnetic field  $B$  varies along the  $z$  axis, the Larmor frequency  $\omega_L$  also varies. When the RF coils generate the  $B_1$  magnetic field, only the spins that have a Larmor frequency  $\omega_L$  close to  $\omega_{RF}$  are excited and align with  $B_1$ . This allows performing a *slice selection* and measuring the response only in a  $(x, y)$

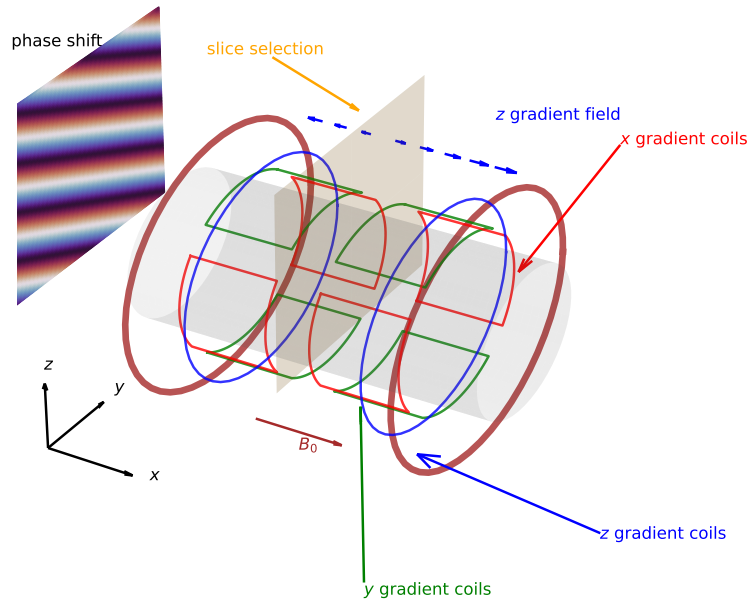


Figure 1.3: Main components of an MRI scanner. The primary field  $B_0$  is uniform (brown), and the space varying field is generated using the gradient coils in  $z$  (blue). This allows selecting a slice which protons inside are precessing at a given frequency  $\omega_L$ . The  $x$  and  $y$  gradient coils create a phase shift in this plane (green and red).

plane. Depending on the  $z$  gradient field that is generated, the selected slice can be displaced along the  $z$  axis.

**Relaxation** Once the spins are aligned in the  $(x, y)$  plane, the signal of the RF coils stops. The spin of the protons then perform the precessional movement. This movement generates a magnetic field that is measured by the receiving antennas located around the scanner. Note that in practice, during the excitation and relaxation, it is never all the spins that align with the magnetic field but only a tiny portion. As in the absence of a magnetic field, the spins are disordered and as it is the average over the whole volume that is measured, the contributions of the spins that do not align with the magnetic field are on average null.

**Phase encoding** During the relaxation, the  $x$  and  $y$  gradient coils are used to change the phase shift of the spins inside the slice. As the magnetic field generated by the  $x$  and  $y$  gradient coils varies in space, it allows changing the strength of the total magnetic field with respect to the position in the  $(x, y)$  plane. This change of the strength of  $B$  makes the Larmor frequency change with respect to the position in the  $(x, y)$  plane. The spins are not precessing at the same frequency and it induces a phase shift between the spins in the plane. Once the  $x$  and  $y$  gradient fields are stopped, the spins recover their initial precessing frequency and the phase shifts no longer change. This allows measuring the response of the slice for a given phase shift. This process is repeated between each measure during the relaxation in order

to change the phase shift over time. It provides different informations which are then used for the reconstruction. In the next section we explain that this phase shift corresponds exactly to the term of a Fourier transform and that varying this phase over time corresponds to varying the frequency measured by the Fourier transform.

**Shot** The process of excitation followed by relaxation is called one *shot*. During an MRI scan, this procedure is repeated several times to get enough samples to reconstruct the image. It is also repeated for the different slices in order to image a 3D volume with a set of 2D images.

### 1.1.3 A simplified model

The objective of this part is, from the previously introduced physical representation, to give the relations between the signals measured by each antenna around the scanner and the image to be recovered, i.e. the density of the protons over space. Effects such as the power decrease over time of the received signal [Fessler 2010] are deliberately left aside.

**Continuous model** If we let  $\phi(x, y, t) \in \mathbb{C}$  denote the phase shift at time  $t$  with respect to the position in space  $(x, y)$ , and if we index by  $1 \leq i \leq I$  the antennas, the  $i$ -th antenna receives the signal

$$y_i(t) \stackrel{\text{def}}{=} \int_{\mathbb{R}^2} f(x, y) \sigma_i(x, y) \phi(x, y, t) dx dy \quad (1.2)$$

with  $f$  the unknown image and  $y_i(t)$  the signal measured at time  $t$ . The term  $\sigma_i(x, y)$  is called a *sensitivity map* and it expresses the sensitivity of the  $i$ -th receiving coil over space. Each antenna receives more or less signal associated with a position  $(x, y)$ , depending on the distance at which the coil is placed. This map is typically a very regular function with a modulus decreasing with the distance to the antennas. As the sensitivity of the antennas decreases with the distance, and as scanners are often equipped with a Faraday cage preventing external signals from penetrating inside, the integration domain can be reduced to a rectangle around the measured volume  $\Omega \subset \mathbb{R}^2$ .

We recall that the phase shift  $\phi(x, y, t)$  is generated by the  $x$  and  $y$  gradient coils. It takes the form of a complex exponential and it is therefore written

$$\phi(x, y, t) = e^{-i(x\xi^{(x)}(t) + y\xi^{(y)}(t))} \quad (1.3)$$

with  $\xi^{(x)}(t)$  and  $\xi^{(y)}(t)$  the frequencies with respect to the axes  $x$  and  $y$  at time  $t$ . The functions  $\xi^{(x)}$  and  $\xi^{(y)}$  describe the *trajectories* in the Fourier domain. These trajectories satisfy constraints related to the physics of the scanner which are detailed in Chapter 3.

**Discrete model** In practice the signal  $y_i(t)$  is integrated by the scanner and it is discretized over time in such a way that  $M$  measurements are available  $(y_i[m])_{m \leq M}$ . By discretizing (1.2) at the different time steps, the model is then written

$$y_i[m] = \int_{\Omega} f(x, y) \sigma_i(x, y) e^{-i(x\xi_m^{(x)} + y\xi_m^{(y)})} dx dy. \quad (1.4)$$

The frequencies  $(\xi_m^{(x)}, \xi_m^{(y)})$  correspond to the discretization of the trajectory  $t \mapsto (\xi^{(x)}(t), \xi^{(y)}(t))$ . An analysis of the effects of the discretization and the integration is given in [Lazarus 2020a].

For the moment the quantities  $f$  and  $\sigma_i$  are continuous. However, from a numerical point of view, they are represented by matrices of size  $N_x \times N_y$  denoted respectively  $u$  and  $s_i$ . We therefore choose to model them using an interpolation function  $\psi : \mathbb{R}^2 \rightarrow \mathbb{R}$  and the relationship between the continuous and discrete version reads

$$f = \left( \sum_{n=1}^N \delta_{p_n} u[n] \right) \star \psi \quad (1.5)$$

$$\sigma_i = \left( \sum_{n=1}^N \delta_{p_n} s_i[n] \right) \star \psi \quad (1.6)$$

where  $N = N_x \times N_y$  and  $u \in \mathbb{C}^N$  (resp.  $s_i \in \mathbb{C}^N$ ) is the discrete representation of  $f$  (resp.  $\sigma_i$ ). The vector  $p_n$  corresponds to the positions on the 2D grid, i.e.  $p_n \in \left[ -\frac{N_x}{2}, \frac{N_x}{2} - 1 \right] \times \left[ -\frac{N_y}{2}, \frac{N_y}{2} - 1 \right]$  and the pixels of  $u$  and  $s_i$  are aligned on this grid.

By replacing the continuous terms by their discrete version in (1.4) and letting  $\xi_m = (\xi_m^{(x)}, \xi_m^{(y)})$  denote the  $m$ -th sampling point, we get

$$y_i[m] = \kappa(\xi_m) \sum_{n=1}^N u[n] s_i[n] e^{-i\langle p_n, \xi_m \rangle}. \quad (1.7)$$

The function  $\kappa$  is the Fourier transform of  $\psi$ . In practice the constant interpolating function is chosen in this manuscript. It aligns the points  $p_n$  on the bottom-left of each pixel:

$$\psi(x, y) \stackrel{\text{def}}{=} \mathbf{1}_{0 \leq x \leq 1} \times \mathbf{1}_{0 \leq y \leq 1} \quad (1.8)$$

which yields

$$\kappa(x, y) = \tilde{\kappa}(x) \tilde{\kappa}(y) \quad \text{with} \quad \tilde{\kappa}(x) \stackrel{\text{def}}{=} -e^{-ix/2} \text{sinc}(x/2). \quad (1.9)$$

To generate phase shifts with a wavelength of the order of a pixel (distance between the adjacent points  $p_n$ ) in the image space, we can see in (1.7) that the magnitude of the frequencies must go up to  $\pi$ . The frequencies being included in  $[-\pi, \pi]^2$ , the term  $\kappa$  does not vary much, but above all it can be easily corrected

on the measured signals so as to eliminate it in the equations. Henceforth we will ignore it as it does not drastically change the model. By removing this dependence, we identify in the forward model (1.7) a Non-Uniform Fourier Transform (NUFT) and the equation can be written in matrix form as

$$y_i = A(\xi)S_i u \quad (1.10)$$

where  $S_i$  is a diagonal matrix associated to the  $i^{\text{th}}$  sensitivity map  $S_i = \text{diag}(s_i[n])_{1 \leq n \leq N}$  and  $A(\xi) : \mathbb{C}^N \rightarrow \mathbb{C}^M$  is the non-uniform Fourier transform at frequencies  $\xi \in ([-\pi, \pi]^2)^M \subset \mathbb{R}^{2M}$  with

$$\xi \stackrel{\text{def}}{=} (\xi_m)_{1 \leq m \leq M}. \quad (1.11)$$

#### 1.1.4 Different issues

From a mathematical point of view, the problem of acquiring an image from an MRI scanner can be broken down into two main parts:

- The choice of the frequencies  $(\xi_m)_{m \leq M}$ , also called k-space sampling.
- The reconstruction of the image  $\tilde{u} \in \mathbb{C}^N$  from the measurements  $y \in \mathbb{C}^M$ .

These two problems are briefly described below from a historical perspective before being detailed later in this manuscript. Chapter 3 contains explanations on how existing methods [Lazarus 2019] generate sampling schemes for MRI scanners while taking into account the constraints of the scanner physics. This method is then used as a reference for comparisons in Chapter 3. The issue of image reconstruction is addressed in Section 1.2 from a more general perspective of inverse problems and several applications are given.

##### 1.1.4.1 Computation of the forward model

Depending on the application, the computation of the matrix-vector product  $A(\xi)$  may be expensive. Advances in algorithmic theory as well as hardware technological progress have allowed the development of computational imaging by processing more and more resolved images. The Fast Fourier Transform (FFT) is a perfect example of this. The Cooley and Tukey's algorithm made it possible to calculate applications that took  $O(N^2)$  operations in only  $O(N \log(N))$  thanks to a *divide and conquer* algorithm and an optimized implementation [Cooley 1965]. Subsequently, its variants have allowed computational imaging to expand its fields of application, for example by handling the case of non-uniform Fourier samples with a complexity approaching that of the FFT. In the case of MRI, these advances make it possible to develop trajectories that take full advantage of the potential offered by the scanner by using off-the-grid trajectories. In this subsection, we compare some of the existing libraries for performing Non-Uniform Fourier Transforms (NUFT).

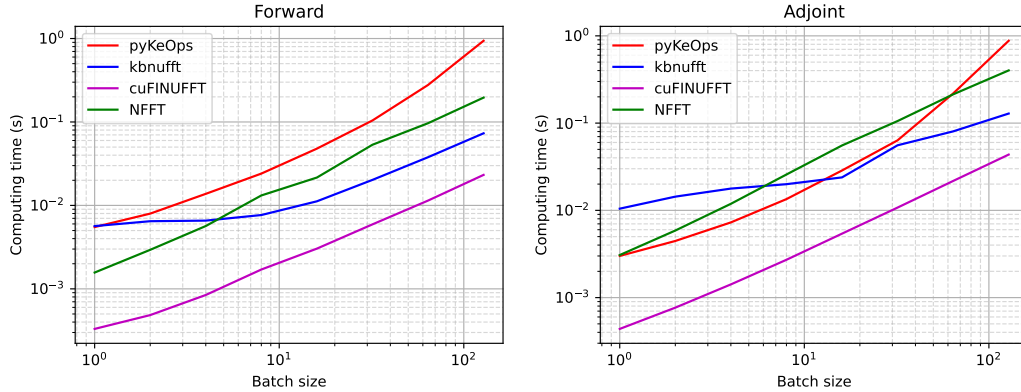


Figure 1.4: Computing time of a NUFFT using different libraries on  $320 \times 320$  images with 10% undersampling ( $\sim 10^4$  points) and different batch sizes. This benchmark is performed on a workstation that has 28 Intel Xeon W-2275 CPU @ 3.30GHz and an Nvidia Quadro RTX 5000 GPU. The NFFT [Keiner 2009] runs on CPU and the cuFINUFFT [Shih 2021], kbnuft [Muckley 2020b] and PyKeOps [Charlier 2021b] based NUFFT run on GPU. The bindings are available at <https://github.com/albangossard/Bindings-NUFFT-pytorch>.

Several implementations of the NUFFT exist. The first effective implementations date back to the work of [Fessler 2003] and [Greengard 2004]. Subsequently [Keiner 2009] proposed another implementation which was officialized through a library widely distributed on computer systems. This work was based on CPU computation, limiting the application to a limited number of images. Afterwards, GPUs opened the way to parallelization and GPU implementations of [Fessler 2003] were proposed in [Lin 2018, Muckley 2020b]. Another implementation with a different interpolation was also proposed in [Shih 2021].

All these implementations rely on careful choices of interpolants and propose an approximation of the NUFFT to a precision chosen by the user. The democratization of GPUs has been followed by an intensive development of libraries allowing to develop GPU codes on high-level languages [Okuta 2017, Lam 2015]. Recently, [Charlier 2021b] has proposed a high-level library allowing GPU computation through symbolic operations. This library takes advantage of reduction operations to exploit the full potential of GPUs. In practice, this allows the implementation of a wide range of operations with an  $O(N^2)$  complexity, but which, thanks to massively parallel computing, run at a speed that is slightly worse than other libraries using more complex strategies. More importantly, this makes it possible to implement complex operations very easily and it also supports automatic differentiation. Figure 1.4 gives a comparison of the computing time of different existing libraries [Keiner 2009, Shih 2021, Muckley 2020b] and of this naive implementation.

### 1.1.4.2 Image reconstruction

The first algorithms used to reconstruct images were inspired by CT-scan (*computed tomography*) and used back-projection  $A(\xi)^*$  to map signals in the Fourier domain to the image space [Lauterbur 1973].

This back-projection makes sense when the frequencies are located on a grid (see Section 1.2.1.2) and therefore the columns of  $A(\xi)$  are orthogonal. As soon as this property is no longer verified, the performance of this reconstruction method collapses and it is then necessary to consider variational approaches [Fessler 2010]. The latter consist in minimizing the sum of a data fidelity term and a regularization term given by the prior on the image to be reconstructed. Non-linear reconstructors such as the ones promoting sparsity fall into this category. They are detailed in Section 1.2.1.

From the second half of the 2010s onwards, another class of reconstructors has emerged: *learned reconstructors*. These are reconstruction methods that use a neural network to denoise the image or remove reconstruction artefacts. Perhaps the most advertised of these in MRI is AUTOMAP [Zhu 2018] which builds a mapping from the sensor data to the reconstructed image. Other approaches consisting in using a coarse inversion of the operator followed by a denoising neural network also exist. These types of approaches were then supplanted in terms of performance by other methods inspired by classical variational methods called *unrolled networks*. They consist in replacing the denoising term, usually built from a prior on the image, by a neural network whose weights are optimized [Diamond 2017, Adler 2017]. This approach is detailed in Section 1.2.4.4.

Through this manuscript we will consider reconstruction algorithms. We define them in the following definition.

**Definition 1** (Reconstructor). *Let  $R(y, A(\xi), \theta)$  denote a reconstruction algorithm which inputs are*

1. *the vector of measurements  $y$ ,*
2. *the forward operator  $A(\xi)$ ,*
3. *the parameters  $\theta$  of this reconstructor.*

*The output is the reconstructed image  $\tilde{x} = R(y, A(\xi), \theta)$ .*

This framework applies both to classical non-learned priors and to learned reconstructors that involve neural networks (see Section 1.2).

### 1.1.4.3 The choice of the k-space sampling

As for each slice of a 3D image, a scanner measures the Fourier transform of a 2D image for a set of  $M$  frequencies  $(\xi_m)_{m \leq M}$ , the question of choosing these frequencies can be tackled using information theory. For integer frequencies (i.e. frequencies located on a grid), Shannon's theory guarantees that a signal can be recovered as soon as it is sampled at twice its maximum frequency



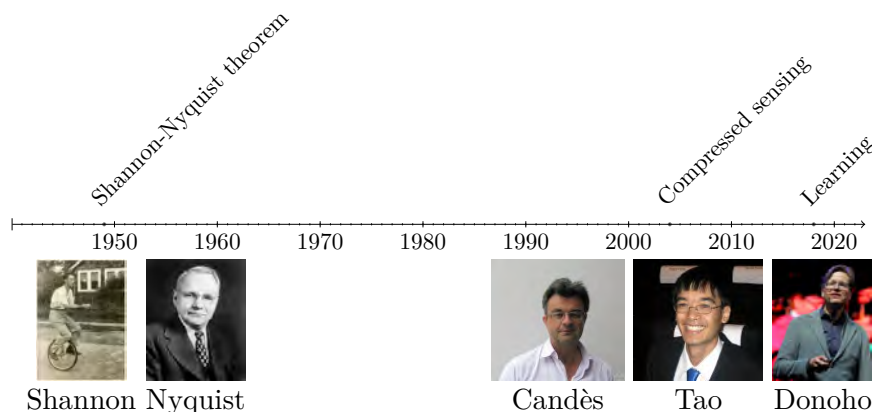


Figure 1.5: Main trends in sampling theory since the 1940s: it started with the Shannon-Nyquist theorem at the end of the 1940s, compressed sensing has emerged in the 2000s and nowadays learning challenges the existing theories.

[Kotelnikov 1933, Shannon 1949, Shannon 1948]. This theory was developed at the end of the 1940s and the main trends in sampling theory are summarized in Figure 1.5. It has motivated the development of acquisition sequences that allow the generation of phase shifts with spatial frequencies that are aligned on a grid, such as the *Echo Planar Imaging* (EPI) sequence [Schmitt 2012]. From a theoretical point of view, progress has been made on non-uniform sampling [Feichtinger 1994] and generalizations of Shannon’s theorem have been obtained [Marvasti 2012]. This theory is now mature and it gives theoretical results on how to choose an optimal sampling scheme for bandlimited signals<sup>2</sup> and linear reconstructors.

One of the major challenge in MRI is to reduce acquisition times without degrading the quality of the reconstructed image. As the total scan time is related to the number of spin excitation cycles, the aim is to reduce the number of frequencies measured for each slice. From the 1980s onwards, it was observed that acquisition times could be decreased by reducing the number of acquisition points without compromising too much the reconstruction. In the early 2000s, the field of compressed sensing provided a theoretical framework for these observations, giving a major boost to research in this area [Candès 2006, Lustig 2005]. The seminal works were based on the restricted isometry property (RIP) or on the incoherence between the measurements. However it soon became evident that these concepts were not suited to the practice of MRI and that the important gains were actually based on the local coherence principle [Adcock 2017, Boyer 2019]. The main conclusion of these works is that a good sampling scheme should satisfy a certain density that varies with the position in the  $k$ -space. In particular, low frequencies must be sampled densely enough to meet the Shannon criterion at the center and

<sup>2</sup>A bandlimited signal is defined as a signal with a Fourier transform that has bounded support. With a slight abuse of language, in MRI it is the Fourier transform of the signal that is bandlimited. This hypothesis is respected as long as the scanner receivers are not exposed to responses outside the scanned volume.

the density is allowed to decrease in the high frequencies.

These theories have led to the development of methods that generate sampling schemes by minimizing the distance between two probability measures: the first is a target density  $\rho$  and the second is a sum of Dirac masses at the sampling positions  $\xi_m$ . Mathematically this reads

$$\inf_{\xi \in \Xi_C([- \pi, \pi]^D)^M} \text{dist} \left( \rho, \frac{1}{M} \sum_{m=1}^M \delta_{\xi_m} \right) \quad (1.12)$$

where  $D$  is the dimension, here  $D = 2$ . The set  $\Xi$  is the constraints set associated with the positions of the points along the trajectories. These constraints are detailed in Chapter 3.

This formalism has led to several works [Boyer 2016, Chauffert 2017, Teuber 2011] and has been developed up to the implementation on MRI scanners [Lazarus 2019, Lazarus 2020b]. Note that other authors also proposed a similar approach to optimize the distance between measures on manifolds [Teuber 2011, Gräf 2012] and this formalism is also known in image processing as *electrostatic halftoning* [Schmaltz 2010, Gwosdek 2014]. In (1.12), the Wasserstein distance could also be used and in this case the problem fits into the class of optimal transport problems [Lebrat 2019]. The choice of the density  $\rho$  is motivated by theoretical considerations such as the respect of the Shannon criterion in the low frequencies.

A new trend consists in changing the distance in (1.12) by a metric governed by the data. The acquisition points are no longer optimized to meet a certain density but directly *learned* from the output of the reconstruction algorithm. The problem (1.12) is replaced by

$$\inf_{\xi \in \Xi_C([- \pi, \pi]^D)^M} \mathbb{E} [\eta(x, R(A(\xi)x, A(\xi), \theta))] \quad (1.13)$$

where  $\eta$  is an image quality metric and  $x$  is a random variable representing the reference images that comes from an image database. We recall that the vector  $\theta$  represents the parameters of the image reconstruction mapping.

This approach has been at the heart of many works since the mid-2010s. They can be broken down into three main categories: methods based on subset selections, mask optimization and gradient based methods that optimize the position of the sampling locations continuously. The first approaches were based on the selection of subsets from a set of fixed patterns. The combination of trajectories leading to the best reconstruction is selected by a greedy algorithm that builds the k-space iteratively [Baldassarre 2016, Gözcü 2018]. This type of algorithm has the huge drawback of having a bad scaling with the dimension. Other works were then proposed to improve the scalability using stochastic optimization [Sanchez 2020],  $\ell^1$  penalization and bi-level programming [Sherry 2020] as well as bias-accelerated subset selection [Zibetti 2021]. The first work to then extend the optimization of sampling schemes to joint optimization with the reconstruction method is [Jin 2019].

Paper	Scaling	Off-the-grid	Points	Lines	$\ell^1$ recon.	Deep recon.	Linear recon.	Joint optim.
Self-Supervised Deep Active Accelerated MRI [Jin 2019]	~	✗	✗	✓	○	✓	○	✓
Greedy approaches [Baldassarre 2016, Gözcü 2018]	✗	✗	✓	✓	✓	✓	✓	✗
Fast greedy approaches [Sanchez 2020, Zibetti 2021]	✓	✗	✓	✓	✓	✗	✓	✗
Bilevel MRI [Sherry 2020]	~	✗	✓	✓	✓	✗	○	✗
J-MoDL [Aggarwal 2020]	~	✗	✓	✓	○	✓	○	✓
LOUPE [Bahadir 2020]	✗	✗	✓	✓	○	✓	○	✓
PILOT [Weiss 2021]	✗	✓	✓	✓	○	✓	○	✓
BJORK [Wang 2022a]	✗	✓	✓	✓	○	✓	○	✓

Table 1.1: Comparison of different data-driven sampling schemes optimization. The symbol ~ means questionable. The symbol ○ indicates that no results were reported with this approach, but that it can be easily incorporated within the framework.

This consists in modifying the problem (1.13) to also minimize with respect to the parameters  $\theta$  involved in the reconstruction algorithm  $R$ . In a similar spirit to [Sherry 2020], [Bahadir 2020] proposes to learn a sampling pattern by optimizing a mask allowing the selection of the frequencies in the Fourier domain but in addition to existing works, they use a neural network in the reconstruction method.

Another class of methods is based on the optimization of the position of the points by making them evolve *off-the-grid*. The first paper to propose this approach is [Aggarwal 2020] where the constraints are handled by fixing patterns on a specific structure (aligned on lines for example). Other works then investigated the optimization of the points position by letting them evolve freely [Weiss 2021, Wang 2022a]. The constraints are satisfied by solving a travelling salesman problem for [Weiss 2021] and by using a penalty in the cost function for [Wang 2022a]. A summary of the different existing works and their comparison is given in Table 1.1.

What is important to remember is that in most recent works, the authors propose to learn the sampling scheme jointly with the weights of a neural network that makes up the reconstructor  $R$ . Although the formulation seems simple, solving this problem is a real computational challenge. Indeed, if a gradient-based optimization method is used, the reconstruction method  $R$  has to be differentiated w.r.t.  $\xi$  and all the intermediate variables that depend on  $A(\xi)$  have to be stored. Even if these computational problems are solved with modern computing capabilities, such for-

mulation may be impossible to solve due to a combinatorial number of minimizers, which is the subject of Chapter 2. Chapter 3 proposes a method that drastically reduces the numerical cost of the resolution of (1.13).

#### Open questions

- Are there issues in off-the-grid data-driven optimization of Fourier sampling schemes?
- How to globalize the convergence in Fourier sampling optimization?

## 1.2 Image reconstruction for linear inverse problems

This section is a gentle introduction to inverse problems for imaging and the existing reconstruction methods. It starts from the very beginning of image reconstruction and goes to the recent advances that combine knowledge of physics and learned priors. First, we recall the definition of an inverse problem and we give three application cases: deblurring, reconstruction of MR images and CT scans. Then we explain how to recover the unknown quantity from the observations with classical approaches that involve hand-crafted priors. In a third part, we explain how to train deep neural networks and we give some of the main stochastic optimizers with open questions on the choice of the hyperparameters. This section ends with learned priors: operator and parameters learning through bilevel programming, *plug&play* (P&P) approaches, approximation inverse, unrolled networks and generative priors.

### 1.2.1 Introduction to inverse problems for imaging

Inverse problems consist in finding a quantity  $x$  from a vector of measurements  $y$ . This quantity and its associated measurement are related through an operator which in many applications is linear. This type of problem appears in many different domains, of course in medical imaging (microscopy, MRI, CT scans), but also in geology (seismic wave measurement) and in space observation (measurement of a quantity on the ground from satellite observations). Throughout this thesis, we focus on applications for biomedical imaging where  $x$  represents a 2D image and where the forward operator that maps the quantity  $x$  to the measurement  $y$  is linear.

#### 1.2.1.1 Inverse problem

Many image acquisition systems can be modelled by a linear operator:

$$y = A(\xi)x + \epsilon \quad (1.14)$$

where  $x$  is the image or volume to be measured and  $\epsilon$  is a random variable modelling the noise. The operator  $A$  is parameterized by a vector  $\xi$  which corresponds to the physical characteristics of the acquisition system. It can be for example the

frequencies acquired in the Fourier domain for MRI or the angles in CT-scan. The question then arises of recovering  $x$  from  $y$  which is an *inverse problem*.

**Definition 2** (Inverse problem).

*Given the observation  $y \in \mathbb{C}^M$ , we seek to find  $\tilde{x} \in \mathbb{C}^N$  such that  $y = A(\xi)\tilde{x}$ .* (1.15)

In general, the quantity to recover  $\tilde{x} \in \mathbb{C}^N$  lives in a space of much greater dimension than the observation vector  $y \in \mathbb{C}^M$ . The noise can also be outside of the range of  $A(\xi)$ . This leads to an under-determined system and the solution, if it exists, is not unique. This problem is difficult, especially as the measurements are often contaminated by noise.

**Definition 3** (Well-posed problem). *An inverse problem is said to be well-posed if it verifies the three following conditions:*

- Existence: *there exists a solution to problem (1.15).*
- Uniqueness: *the solution of the problem (1.15) is unique.*
- Stability to small perturbations: *a small perturbation of  $y$  induces a small perturbation of  $\tilde{x}$ .*

In practice the problem (1.15) is seldom well-posed, due to either the dimensionality issue  $M \neq N$ , or the conditioning of  $A(\xi)$  (stability). These issues can be solved by regularizing the problem. The problem of recovering  $x$  from  $y$  can be formalized as a minimization problem, and regularizing consists in adding a penalization term  $\mathcal{R} : \mathbb{C}^N \rightarrow \mathbb{R} \cup \{+\infty\}$  to the function which is minimized. This regularization can be equal to  $+\infty$  to enforce constraints on the solution  $\tilde{x}$ .

$$\tilde{x} \stackrel{\text{def}}{=} \arg \min_{x \in \mathbb{C}^N} F(x) + \mathcal{R}(x) \quad \text{with } F(x) \stackrel{\text{def}}{=} \frac{1}{2} \|A(\xi)x - y\|_2^2. \quad (1.16)$$

The function  $F$  is the data fidelity term. The regularization term  $\mathcal{R}$  is called a *prior* since, depending on the term chosen, it influences the image obtained when solving (1.16). The minimization of the data fidelity term alone enforces the existence and a good choice of regularizer ensures the uniqueness. The choice of the regularization term has been the subject of an impressive amount of works over the last 30 years. Before giving common regularizations, we introduce hereafter examples that drive this section.

### 1.2.1.2 Examples of forward operator

Throughout this chapter we introduce three different inverse problems: deblurring, MR image reconstruction and computed tomography (CT).

**Deblurring** We first recall the definition of the convolution in the continuous setting.

**Definition 4** (Convolution). *For  $t \in \mathbb{R}$ , the convolution product between two functions  $\xi$  and  $x$  is defined by*

$$y(t) = \int_{\mathbb{R}} \xi(u)x(t-u) du. \quad (1.17)$$

In the discrete setting, the *non-periodic convolution* for a kernel  $\xi$  of size  $J$  writes

$$y[i] = \sum_{j=1}^J \xi[j]\bar{x}[i-j]$$

with  $\bar{x}$  the signal which is equal to  $x$  on the interval where  $x$  is defined, and which is eventually padded with zeros to handle the boundaries. Letting  $\%$  denote the division remainder, the *periodic convolution* for a kernel of size  $J$  is defined as

$$y[i] = \sum_{j=1}^J \xi[j]x[(i-j)\%N]. \quad (1.18)$$

**Proposition 1.** *If  $\mathcal{F}$  denotes the Fast Fourier Transform (FFT) on vectors, the periodic convolution can be computed using 3 FFT:*

$$y = v \star u = \mathcal{F}^{-1}(\mathcal{F}(v)\mathcal{F}(u))$$

This is of great importance in deblurring. Indeed, there are different kinds of blur depending on the physical system that is modelled (for example fixed blur or space varying blur [Sawchuk 1972]). In the simplest case of a fixed blur, the forward model consists in convolving the image  $x$  with a kernel  $\xi$ :

$$y = A(\xi)x = \xi \star x$$

which can be efficiently computed in the discrete case using the FFT.

**MRI** We have seen in Section 1.1.3 that an MRI scanner allows measuring the Fourier transform of an image along trajectories in the Fourier domain. In the particular case where the frequencies are aligned on a grid, the application of  $A(\xi)$  to an image can be computed using the FFT. The scanner measures a set of  $M$  frequencies on a grid defined by  $\xi$  among the  $N$  possible frequencies of the standard FFT:

$$A(\xi) = M(\xi)\mathcal{F}$$

where  $M(\xi)$  is a diagonal matrix which plays the role of a mask and with diagonal element  $i$  equals to 1 if the  $i$ -th frequency lies in  $\xi$  and 0 otherwise.

In practice, on most MR technologies the constraints on the trajectories in the Fourier domain prevents the measured frequencies to be aligned on a grid. We

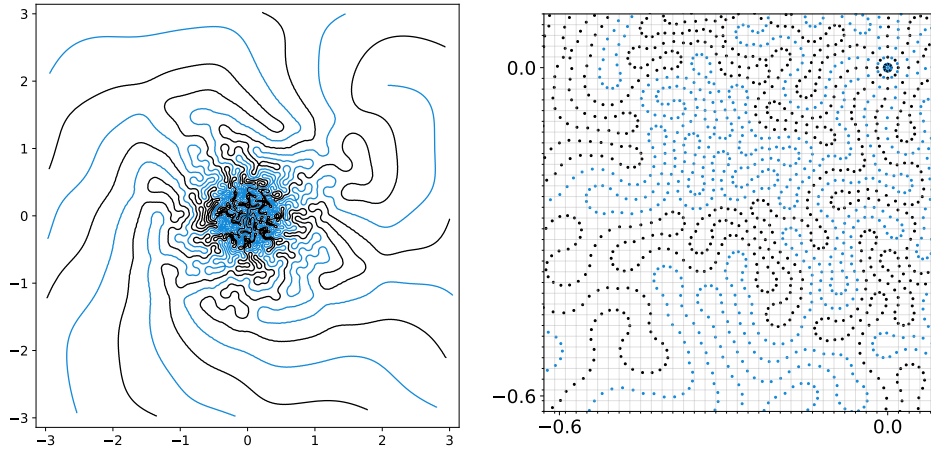


Figure 1.6: Example of trajectories for MRI with 10% undersampling (in comparison to the fully sampled scheme on the Shannon grid). On the left the scheme in the Fourier domain and on the right a zoom. The grid represents the fully sampled Shannon grid (on-the-grid frequencies).

say that the frequencies are *off-the-grid*. We refer the reader to Section 1.1.4.1 for details concerning the computation of  $A(\xi)$  in this case. In Figure 1.6 an example of trajectories in the Fourier domain are given: the measured frequencies are far away from being defined on a grid.

**Computed Tomography (CT)** A CT scanner works as follows: rays are sent through tissues which internal compositions are to be measured and the intensities of the rays on the other side of the scanned volume are measured by a sensor array. There are several technologies for tomography and two of them stand out:

- Parallel-beam: the rays are emitted in a parallel way and the intensity of the received signal is measured by a line of sensors in a plane.
- Fan-beam: the rays are emitted by a point source and propagate in a radial way, the intensity is measured on the other side of the volume by a line of sensors which can be in a plane or distributed on an arc of circle and which center is the origin of the source.

In these two cases, the received signal corresponds to what is called a *sinogram*: it is an image in which one of the axes corresponds to the position along the line of sensor and the other corresponds to the angle formed by the source and the sensors. Figure 1.7 gives an illustration of these two technologies.

For a given ray of index  $m$ , angle  $\theta_m$  and distance to the zero point  $s_m$ , the relation between the measured signal  $y_m$  and the image to measure is modelled as

$$y_m = \int_{u_x \cos \theta_m + u_y \sin \theta_m = s_m} x(u_x, u_y) \, du_x du_y \quad (1.19)$$

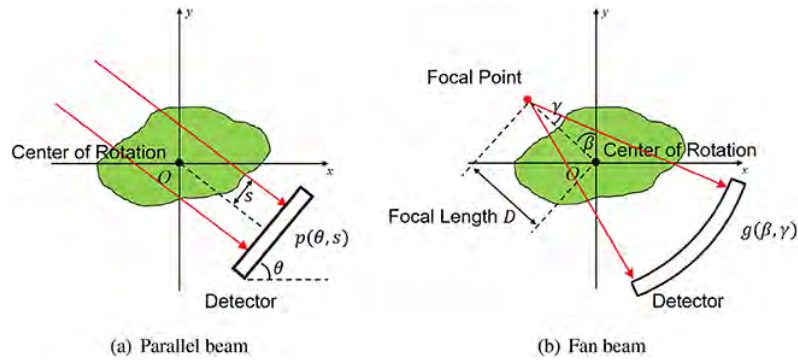


Figure 1.7: Illustration of two configurations for CT imaging: on the left parallel-beam and on the right fan-beam. Credit: [Wang 2019]

where  $u_x$  and  $u_y$  are the space variables.

## 1.2.2 Hand-crafted priors

### 1.2.2.1 Tikhonov regularization

Tikhonov regularizers have the form  $\mathcal{R}(x) = \frac{\lambda}{2} \|Lx\|_2^2$  with  $L$  a matrix and  $\lambda \in \mathbb{R}^+$ . In this section we focus on the most simple Tikhonov regularisation:  $\mathcal{R}(x) = \frac{\lambda}{2} \|x\|_2^2$ . This regularization has the appealing advantage of making problem (1.16) easy to solve since minimizing this problem boils down to solve the symmetric definite positive linear system

$$x = [A(\xi)^* A(\xi) + \lambda \text{Id}]^{-1} A(\xi)^* y. \quad (1.20)$$

This problem can thus be solved with a conjugate gradient algorithm, and depending on the spectrum of the forward operator  $A(\xi)$ , a relatively small number of iterations may be enough to approximate the solution of the variational problem faithfully [Tyrtysnikov 1997].

When adding a regularization, the minimum is a trade-off between the minimization of the data fidelity term and the minimization of the regularization. Hence the reconstructed image can be biased. In the case of the Tikhonov regularization, this impact is well studied with the mathematical properties of problem (1.16). It uses the singular value decomposition (SVD) which we define below.

**Singular value decomposition** Singular value decomposition is the generalization of the eigenvalue decomposition for non-square complex matrices.

**Definition 5.** *If  $A$  is a real (resp. complex) matrix of size  $M \times N$ , then there exists a decomposition:*

$$A = U \Sigma V^* \quad (1.21)$$

with  $\Sigma$  a matrix of size  $M \times N$  with extra-diagonal elements are 0 and diagonal coefficients are non-negative. The matrices  $U$  and  $V$  are unitary with real (resp.



complex) coefficients of size  $M \times M$  for  $U$  and  $N \times N$  for  $V$ .

**Remark 1.** *The decomposition is not unique and only  $\Sigma$  is unique if we assume that its coefficients are ordered.*

**Definition 6** (Pseudo-inverse of a matrix). *The pseudo-inverse  $A^\dagger$  of a matrix  $A \in \mathbb{C}^{M \times N}$  is the unique matrix defined by the following equalities:*

$$\begin{aligned} (i) \quad AA^\dagger A &= A & (iii) \quad (AA^\dagger)^* &= AA^\dagger \\ (ii) \quad A^\dagger AA^\dagger &= A^\dagger & (iv) \quad (A^\dagger A)^* &= A^\dagger A \end{aligned}$$

**Proposition 2.** *The pseudo-inverse is the limit when  $\lambda \rightarrow 0$  of the matrix in the right hand side term of (1.20):*

$$A^\dagger = \lim_{\lambda \rightarrow 0} (A^* A + \lambda \text{Id})^{-1} A^* \quad (1.22)$$

This means that when  $\lambda$  tends towards zero, the reconstructed signal using the Tikhonov regularization is exactly the one obtained using the pseudo-inverse.

**Proposition 3.** *The pseudo-inverse of a matrix  $A \in \mathbb{C}^{M \times N}$  can be computed using its SVD:*

$$A^\dagger = V \Sigma^\dagger U^* \quad (1.23)$$

with  $\Sigma^\dagger$  a matrix with extra-diagonal elements equal to 0 and

$$\left(\Sigma^\dagger\right)_{i,i} = \begin{cases} \frac{1}{\Sigma_{i,i}} & \text{if } \Sigma_{i,i} > 0 \\ 0 & \text{otherwise} \end{cases} \quad \forall i \leq \min(M, N). \quad (1.24)$$

**Proposition 4.** *If a signal  $x$  is decomposed into an eigenvectors basis  $(v_i)_i$  that form the matrix  $V$  and that are associated to the eigenvalues  $(\mu_i)_i$  of  $A(\xi)^* A(\xi)$ ,*

$$x = \sum_{i=1}^N u_i v_i \quad \text{with } u_i \in \mathbb{C}, v_i \in \mathbb{C}^N,$$

then, the solution associated to the Tikhonov reconstructor (1.20) can be written in the basis of eigenvectors:

$$\tilde{x} = \sum_{i=1}^N \frac{\mu_i}{\mu_i + \lambda} u_i v_i \quad (1.25)$$

Proposition 4 means that the reconstructed signal  $\tilde{x}$  is the original signal  $x$  but with an amplitude in the eigenvectors basis that is modulated by  $\frac{\mu_i}{\mu_i + \lambda}$ . The components associated to large eigenvalues are less impacted by the regularization term than the components associated to smaller eigenvalues which can be highly diminished. As for the components associated with the zero eigenvalues, they are

of course also zero. Note that when  $\lambda$  tends to 0, the reconstructed signal can be expressed as

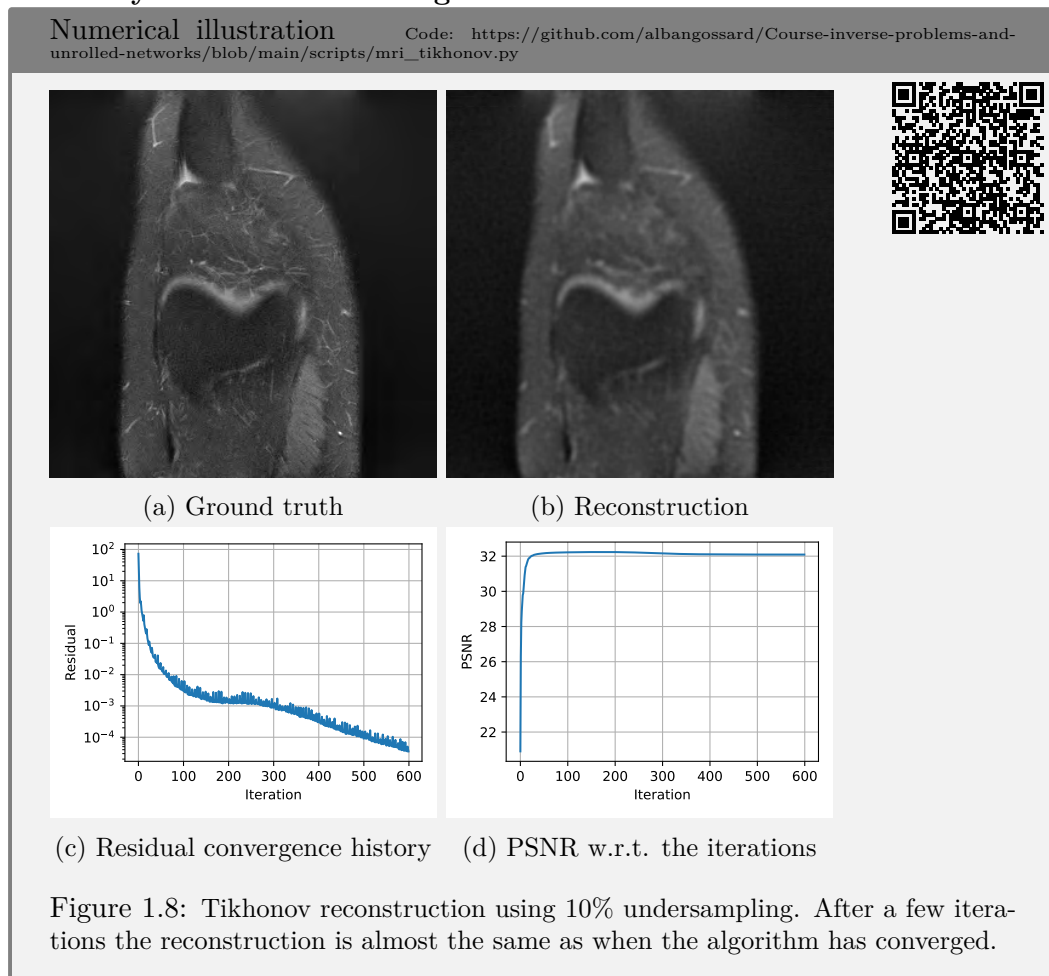
$$\tilde{x} = \sum_{i=1}^N \mathbb{1}_{\mu_i=0} u_i v_i. \quad (1.26)$$

This corresponds exactly to project the signal onto the family of eigenvectors with non-zero eigenvalues or to use the pseudo-inverse.

**Proposition 5.** *In the noiseless setting, the reconstructor associated to the pseudo-inverse corresponds to project onto the range of  $A(\xi)^*$ :*

$$\tilde{x} = \Pi_{\text{Im}(A(\xi)^*)}(x). \quad (1.27)$$

### Summary of the Tikhonov regularization



#### Pros

- Theoretical properties well studied

**Cons**

- Poor reconstruction performance for images with discontinuous edges
- Outperformed by learned priors on natural images (see Section 1.2.4)

**1.2.2.2 Beyond Tikhonov regularization**

The Tikhonov regularization introduced before yields linear reconstructor. These reconstructors have poor reconstruction performance with images that have discontinuous edges. In this section we describe some of the main non-linear reconstructors used in inverse problems for imaging.

**Sparsity in a dictionary** An image can be sparsely represented within a family of well chosen vectors. It is thus tempting to design a regularization that promotes sparsity. Given a dictionary  $D \in \mathbb{C}^{N \times P}$ , one possible parametrization is to synthesize an image  $\tilde{x} = Dz$  and to minimize a function that balances between the data fidelity term and a sparse penalization of  $z$ :

$$z = \arg \min_{z \in \mathbb{R}^P} \frac{1}{2} \|A(\xi)Dz - y\|_2^2 + \lambda \|z\|_1 \quad (1.28)$$

with  $\lambda \in \mathbb{R}^+$ . The matrix  $D$  can be a dictionary of wavelets for example (see [Frazier 2006] for an introduction) and  $z$  is the sparse representation of the image in the synthesis family.

**Total variation** A common regularization term in (1.16) is the total variation (TV) regularization [Rudin 1992]. It has been the state-of-the-art for years in biomedical imaging and its use is now declining with the advent of neural networks. Most of the observed images in CT and MRI are piecewise constant. It is therefore natural to look for a solution that has a low number of jumps and large constant parts. If we let  $\nabla : \mathbb{C}^N \rightarrow \mathbb{C}^{D \times N}$  denote the discrete gradient for signals in dimension  $D$ , the regularization term  $\mathcal{R}(x) = \lambda \|\nabla x\|_1$  promotes piecewise constant images. The  $\ell^1$  norm promotes sparsity of the term  $\nabla x$  which corresponds to having a few jumps for a well chosen value of  $\lambda$ . This kind of problem can be solved using a proximal gradient descent or splitting methods.

**1.2.2.3 Reconstruction algorithms**

When the regularization term is non differentiable, as for the regularization terms introduced in the previous section, one can resort to proximal gradient descent or splitting algorithms like Douglas-Rachford [Eckstein 1992] or its dual version the Alternating Direction Method of Multipliers (ADMM) [Parikh 2014, Boyd 2011]. All these algorithms use the proximal mapping of the regularization term.

**Definition 7** (Proximal mapping). *Given a function  $g : \mathbb{C}^N \rightarrow \mathbb{R} \cup \{+\infty\}$  that is lower semi-continuous and convex, the proximal mapping is defined implicitly by a minimization problem:*

$$\text{prox}_g(x) \stackrel{\text{def}}{=} \arg \min_{y \in \mathbb{C}^N} g(y) + \frac{1}{2} \|y - x\|_2^2. \quad (1.29)$$

For specific functions, this proximal mapping can be computed in closed form. Otherwise, we have to resort to an iterative algorithm to compute one evaluation.

**Proximal gradient descent** A proximal gradient descent can be applied to solve the variational formulation associated with the sparse dictionary and with TV regularization. The sequence of iterates are then

$$x^{(k+1)} = \text{prox}_{\gamma \mathcal{R}} \left( x^{(k)} - \gamma A(\xi)^* \left( A(\xi) x^{(k)} - y \right) \right) \quad (1.30)$$

Choosing a value for the step  $\gamma$  small enough, this sequence converges towards a solution of the problem (1.16) [Facchinei 2003]. The value of  $\gamma$  depends on the spectral norm of  $A(\xi)$ , more precisely we should have  $\gamma \leq \frac{2}{\|A(\xi)\|_{2 \rightarrow 2}^2}$ .

However, for the TV regularization, the proximal mapping has no closed form and we resort to use the ADMM in the following paragraph.

**Alternating Direction Method of Multipliers (ADMM)** This method relies on a splitting with respect to the different terms in the function that has to be minimized. We introduce the method in the case of a two variables splitting. The function to minimize is

$$\min_{x \in \mathbb{C}^N} f(Ax) + g(Lx) \quad (1.31)$$

with  $f$  a function related to the data fidelity term,  $A$  and  $L$  are linear operators and  $g$  is the function in the regularization term.

Using a splitting, the problem (1.31) is equivalent to

$$\min_{\substack{x, z_1, z_2 \\ \gamma_1 Ax = z_1 \\ \gamma_2 Lx = z_2}} f\left(\frac{z_1}{\gamma_1}\right) + g\left(\frac{z_2}{\gamma_2}\right). \quad (1.32)$$

Letting  $z = \begin{pmatrix} z_1 \\ z_2 \end{pmatrix}$ ,  $\mu = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}$ ,  $B = \begin{pmatrix} \gamma_1 A \\ \gamma_2 L \end{pmatrix}$  and  $\gamma_1, \gamma_2$  some positive constants, the augmented Lagrangian [Bertsekas 2014] is defined as

$$\mathcal{L}(x, z_1, z_2, \mu_1, \mu_2) \stackrel{\text{def}}{=} f\left(\frac{z_1}{\gamma_1}\right) + g\left(\frac{z_2}{\gamma_2}\right) + \frac{\beta}{2} \|Bx - z\|_2^2 + \langle \mu, Bx - z \rangle. \quad (1.33)$$

The ADMM consists in alternatively minimizing the augmented Lagrangian with respect to the different splitted variables and then to update the Lagrange

multipliers:

$$\begin{cases} x^{(k+1)} = \min_x \mathcal{L}(x, z^{(k)}, \mu^{(k)}) \\ z^{(k+1)} = \min_z \mathcal{L}(x^{(k+1)}, z, \mu^{(k)}) \\ \mu^{(k+1)} = \mu^{(k)} + \beta (Bx^{(k+1)} - z^{(k+1)}) \end{cases} \quad (1.34)$$

In the case of the TV reconstructor, we have  $f = \frac{1}{2} \|\cdot - y\|_2^2$ ,  $A = A(\xi)$ ,  $L = \nabla$  and  $g : v \in \mathbb{C}^{D \times N} \mapsto \lambda \|v\|_1$  and the minimization steps are the following.

**Minimization w.r.t.  $x$ :** The optimality equations boil down to solving

$$B^* Bx^{(k+1)} = B^* \left( y - \frac{\mu^{(k)}}{\beta} \right) \quad (1.35)$$

which can be done using a conjugate gradient algorithm [Tyrtshnikov 1997].

**Minimization w.r.t.  $z_1$ :** This operation is done element-wise

$$z_1^{(k+1)} = \frac{1}{\beta + \frac{1}{\gamma_1^2}} \left( \frac{y}{\gamma_1} + \beta \gamma_1 A(\xi) x^{(k)} + \mu_1^{(k)} \right). \quad (1.36)$$

**Minimization w.r.t.  $z_2$ :** As the  $\ell^1$  norm is separable, this operation is also performed element-wise

$$z_2^{(k+1)} = \text{prox}_{\frac{\lambda}{\gamma_2 \beta} \|\cdot\|_1} \left( \gamma_2 \nabla x^{(k)} + \frac{\mu_2^{(k)}}{\beta} \right). \quad (1.37)$$

The interested reader can refer to [Parikh 2014, Combettes 2011, Boyd 2011] for more informations on splitting algorithms.

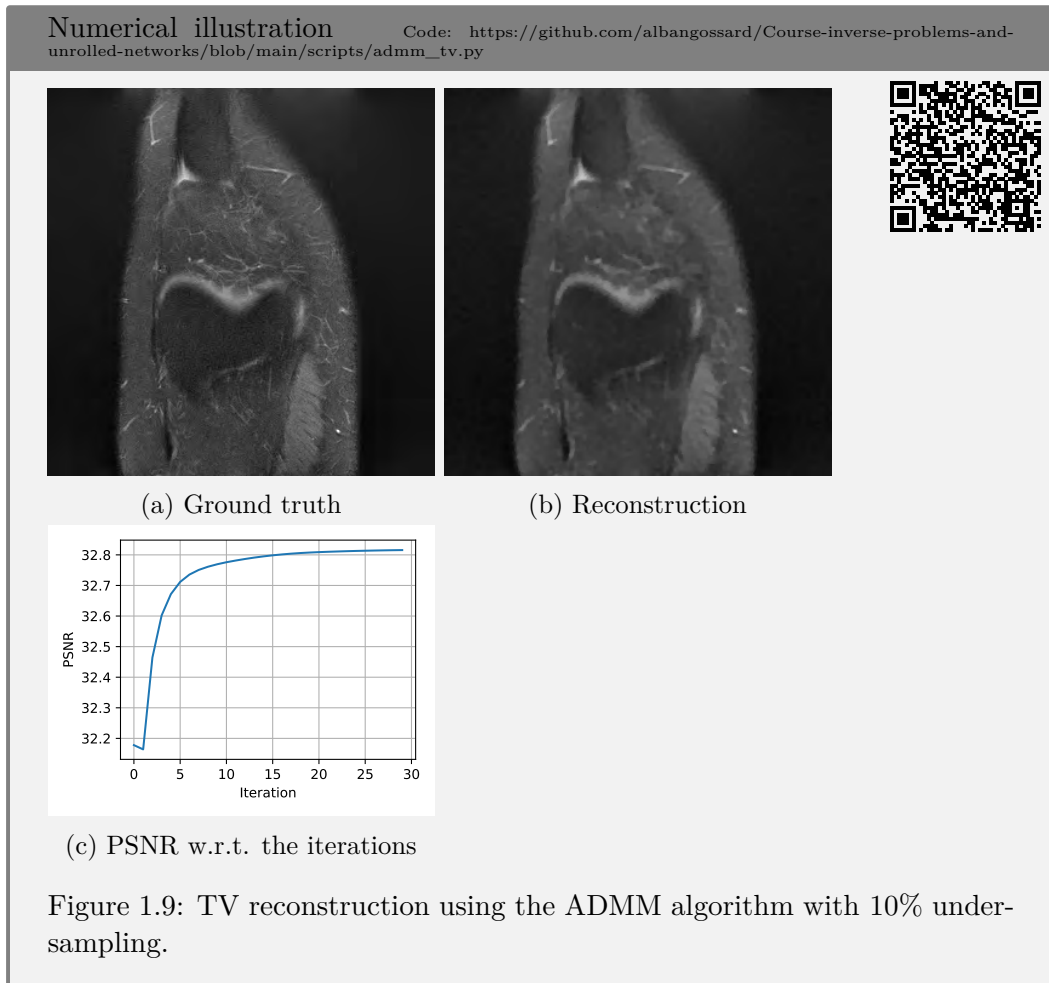
### Summary of the non-linear reconstructors

#### Pros

- Theoretical properties well studied
- Good reconstruction performance

#### Cons

- Outperformed by learned priors
- Potential high computational cost depending on the problem



### 1.2.2.4 Interpretations of the regularizer

In this section several interpretations of the regularizer are given to motivate the learned priors presented in Section 1.2.4.

**Proximal operator as a denoiser** Let us consider problem (1.28) with  $P = N$  and  $D = \text{Id}$

$$\min_{x \in \mathbb{R}^N} \frac{1}{2} \|A(\xi)x - y\|_2^2 + \lambda \|x\|_1 \tag{1.38}$$

and solve this problem with a proximal gradient descent:

$$x^{(k+1)} = \text{prox}_{\gamma\lambda\|\cdot\|_1} \left( x^{(k)} - \gamma A(\xi)^* (A(\xi)x^{(k)} - y) \right). \tag{1.39}$$

The proximal operator associated with the  $\ell^1$  norm is the *soft thresholding* and it writes:

$$\left( \text{prox}_{\alpha\|\cdot\|_1} (x) \right)_i = \text{sign}(x_i) \max(|x_i| - \alpha, 0) \tag{1.40}$$

Figure 1.10 gives an illustration of the soft thresholding, with the value  $\alpha = 1$ .

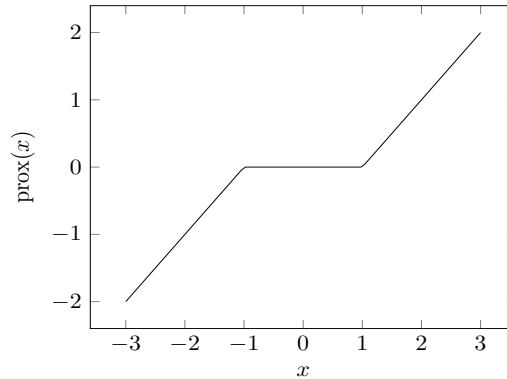


Figure 1.10: Illustration of the soft thresholding which is the proximal mapping of the  $\ell^1$  norm ((1.40) with  $\alpha = 1$ ). This thresholding acts as a denoiser on synthesis problems (1.28).

This operator can be seen as a denoiser since it shrinks the small values to 0 and the value of  $\alpha$  sets the noise threshold that we wish to remove.

**Bayesian interpretation of the regularizer** The solution of (1.16) can be interpreted as a Maximum A Posteriori (MAP) estimate [Ji 2008]. Indeed, assume that the noise is a Gaussian random variable  $\epsilon \sim \mathcal{N}(0, \sigma^2 \text{Id})$  of variance  $\sigma^2$ . Then, the we have  $y|x \sim \mathcal{N}(A(\xi)x, \sigma^2 \text{Id})$  and the probability of observing  $y$  given  $x$  writes

$$p(y|x) = \frac{1}{(2\pi\sigma^2)^{M/2}} \exp\left(\frac{-1}{2\sigma^2} \|A(\xi)x - y\|_2^2\right).$$

Then we write the reconstructed image as a MAP estimation:

$$\tilde{x} = \arg \max_x p(x|y). \quad (1.41)$$

Using the Bayes' rule

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)},$$

we obtain that

$$\begin{aligned} \tilde{x} &= \arg \min_x -\log(p(y|x)) - \log(p(x)) \\ &= \arg \min_x \frac{1}{2\sigma^2} \|A(\xi)x - y\|_2^2 - \log(p(x)). \end{aligned}$$

The term  $-\log(p(x))$  can be identified as the regularizer  $\mathcal{R}(x) \propto -\log(p(x))$ . Hence the regularizer encodes the prior on the image  $x$ .

These two interpretations advocate two main ingredients that are at the core of recent data-driven learned methods:

- *Replace the proximal mapping* by a more general denoiser in the foremen-

tioned iterative algorithms. In this case there is no reason that the denoiser is the proximal mapping of a well-defined function.

- *Learn priors* that perform well for specific tasks using data.

These motivations led to different approaches that are detailed in the following sections. More precisely, *Plug&Play* (P&P) consists in replacing the proximal mapping by an existing denoiser. This denoiser can be hand-crafted or trained to perform a specific task *without* prior knowledge of the iterative algorithm it will be used in. The *unrolled networks* consists in training a deep denoiser inside an iterative algorithm that is known during the training phase in opposition to plug&play. These two approaches are discussed in Section 1.2.4.

### 1.2.3 Training deep networks

In Section 1.2.4, we aim at optimizing the parameters  $\theta$  of a reconstructor in such a way that the output of the reconstructor  $\tilde{x} = R(y, A(\xi), \theta)$  is as close as possible to the ground truth image  $x$ . The purpose of this section is twofold:

- We describe the main aspects of optimizing deep networks for image reconstruction. This provides the technical tools for the following section where an overview of learned priors is given.
- We address some of the aspects of choosing the learning rate and we give an outline of the literature on this subject. This introduction motivates Chapter 5.

**Training deep networks for image reconstruction** If  $x$  is a random vector, minimizing in expectation is formalized as

$$\inf_{\theta} \mathbb{E} \left( \|R(A(\xi)x + \epsilon, A(\xi), \theta) - x\|_2^2 \right) \quad (1.42)$$

where the expectation is taken w.r.t.  $x$  and to  $\epsilon$ . In the above formulation (1.42), the noise term is critical. It is drawn randomly at each iteration and its role is to ensure that the network is stable to small perturbations including adversarial perturbations [Genzel 2022b]. In practice, the density of  $x$  is not available and it is replaced by an empirical distribution over a set of images. This empirical expectation requires computing the gradient over the entire dataset, which is often not possible due to the large number of images. Hence we resort to stochastic optimization. A few images are selected for which the value of the cost function and its gradient are calculated, and the parameters are then updated using this information and, eventually, that of the previous iterations. The data is therefore changed at each iteration until the entire dataset has been processed. This corresponds to do one *epoch*.

**Optimizers** The design of efficient stochastic optimization procedures has been the subject of intensive works over the last decade. The performance obtained are



impressive and they allow training high dimensional models. There is a multitude of stochastic solvers available, some of them are quite robust in practice and given a problem involving stochastic optimization like (1.42), a set of restricted optimizers can be considered. The most common are SGD, RMSProp [Tieleman 2012], Adagrad [Duchi 2011], Adadelata [Zeiler 2012] and Adam [Kingma 2015] (generalization of RMSProp with momentum). These optimizers often rely on hyperparameters such as the step size or the momentum. For each optimizer, these parameters should be tuned with care to obtain the best convergence results. The interested reader can refer to [Kochenderfer 2019] for a complete book on optimization including the stochastic optimizers.

**The learning rate** The choice of the learning rate is of high importance to provide satisfactory performance. This choice can be time consuming since a good learning rate is task dependent. The convergence of stochastic optimization is ensured using Robbins-Monro algorithm [Robbins 1951]. While Robbins-Monro conditions guarantee convergence for an interval of values of the hyperparameters, in practice the best performance is achieved using fine tuning of the learning rate decay. Some methods have been proposed to provide a step that is adapted at each epoch.

In deterministic optimization the scaling of the step can be inferred either from second order information, or by using numerical approximation such as the Barzilai-Borwein (BB) method [Barzilai 1988, Raydan 1997, Dai 2002, Xiao 2010, Biglari 2013, Li 2019a]. This method approximates the curvature of a function with numerical differences using past gradient evaluations. In the stochastic convex setting, the BB method was introduced in [Tan 2016] and it has been extended in [Ma 2018] to non-convex problems and in [Liang 2019] for deep neural networks. Due to the variance of the gradient and possibly to a poor estimation of the curvature by numerical differences, these methods allow prescribing a new step at each epoch only. In [Yang 2018, Castera 2022], the step is prescribed at each iteration at the cost of computing two mini-batch gradients per iteration. Moreover, in [Yang 2018] the gradient over all the data needs to be computed at the beginning of each epoch whereas [Castera 2022] maintains an exponential moving average to avoid this extra computation. The downside of [Castera 2022] is that they still need to tune the learning rate and its decay factor. From all these works, there is still a lack of understanding of how to properly scale the gradient.

The other option is to compute second order information using automatic differentiation. The theory that allows computing the matrix-vector product of the Hessian with a certain direction is well-studied [Walther 2008, Christianson 1992, Griewank 2008, Pearlmutter 1994].

#### Open questions

- Is it possible to reduce the numerical cost of computing second order information?

- Is it possible to find a good scaling of the learning rate?
- How does the convergence and exploration regimes behave with respect to the learning rate?

### 1.2.4 Learned priors

At the beginning of the development of neural networks for inverse problems, numerous networks architectures that learn to invert  $A(\xi)$  were proposed, including the famous AUTOMAP [Zhu 2018]. This class of network takes the vector  $y$  as an input and returns the reconstructed image  $x$ . They have the advantage of being very quick to evaluate once the training is done. However, on problems like MR image reconstruction where the signal  $y$  is in a different space than the one of  $x$ , the network has to learn the physics that maps the two spaces and it can experience instabilities [Antun 2020]. Moreover, the physics is known and it should be used to invert the forward operator. For a few years now, alternative approaches that combine knowledge of the physics of the acquisition system and neural networks have been proposed [Gregor 2010, Sun 2016, Diamond 2017, Adler 2017, Zhang 2018, Dong 2018, Adler 2018, Gilton 2019, Hammernik 2019]. We describe below the main trends.

#### 1.2.4.1 Learning parameters and operators in hand-crafted priors

Before the advent of neural networks, priors were primarily based on weighting of different regularization terms. The choice of the values for the regularization weights is crucial and for more than two variables it becomes intractable. In the 2010s, works started considering the optimization of these parameters in variational formulations like (1.16). This falls into the class of bilevel programming [Kunisch 2013] and it involves implicit differentiation of the solution yielded by the minimization of (1.16). Subsequent works then considered the differentiation of the iterates of an algorithm [Ochs 2015, Ochs 2016]. In [Ochs 2015] the authors propose to differentiate the iterates of a primal-dual algorithm. This idea is at the core of many recent learning based reconstruction approaches and it was extended to the learning of operators and specific operations like denoising. This latter point is discussed in the following parts of this section.

The TV regularization introduced in [Rudin 1992] (see Section 1.2.2.2) has the drawback of being anisotropic. Indeed, it misbehaves for the reconstruction of circular arcs [Chen 2013, Condat 2017, Chambolle 2021a, Chambolle 2021b]. There have been extensive work on the design of discrete isotropic regularizations including but not limited to [Condat 2017, Chambolle 2021a]. Recently, [Chambolle 2021b] proposed to learn consistent discretization for the TV regularization. It formalizes the learning problem as minimizing the reconstruction error of the result given by a primal-dual algorithm. The TV discretization is learned by deriving the gradient of the solution with respect to the weights of the TV discretization.

**Summary****Pros**

- Allows tuning hyperparameters of variational models

**Cons**

- Outperformed by neural networks

**1.2.4.2 Plug & Play priors**

Plug & Play (P&P) methods were first motivated by hand-crafted priors and empirical validation that have shown good results. The idea was introduced in [Venkatakrishnan 2013] as P&P-ADMM where the proximal mapping inside an ADMM algorithm was replaced by an off-the-shelf image denoiser. It corresponds to replace the proximal mapping in Section 1.2.2.3 by this denoiser. It was then extended to a FISTA algorithm in [Gu 2014] with denoisers like BM3D [Dabov 2007] or WNNM [Gu 2014]. This latter formulation has the appealing advantage of not requiring to invert the data fidelity term at each iteration.

Different works then started considering learned priors by using pre-trained neural networks as plug&play priors among which we can cite [Ryu 2019, Zhang 2021b]. Motivated by the MAP estimation of Section 1.2.2.4, the goal of plug&play is to decouple the development of the data updates from the denoising part in an algorithm. In particular, with the advances of learning, one wishes to build a denoiser that does not require training every time the forward model  $A(\xi)$  is changed. Recently, driven by the performance of learned plug&play denoisers, several works investigated the theory behind plug&play algorithms in order to build stable and convergent schemes. This question of convergence was first addressed by [Chan 2016] with a boundedness assumption. A work on equilibrium [Buzzard 2018] then provided an interpretation of plug&play methods as the balance between multiple operators rather than the solution of a minimization problem. The convergence was then tackled by [Ryu 2019] where the authors require a Lipschitz condition on the network that is used. This condition is critical as, to comply with such constraint, it requires to change the network by normalizing each layer [Miyato 2018] or use a penalization [Yoshida 2017]. In parallel, an alternative formulation called Regularization by Denoising (RED) was proposed in [Romano 2017] and it introduces a penalization term of the function in (1.16) that relies on the output of a denoiser. Such a formulation comes with interesting properties under assumptions on the denoising mapping. Indeed, the idea was to derive traditional optimization algorithms to minimize the function while keeping a proximal mapping that is well-defined through a regularization term. This keeps the interpretation of the denoiser as a prior on the image. However, [Reehorst 2018] has revealed that a fixed point interpretation is more appropriate for the RED formulation on most practical denoisers. Then, [Liu 2021] filled the gap between plug&play and RED by providing

conditions under which these methods yield the same converging sequence. More recently, [Hurault 2022a] proposed a plug&play approach where the denoising step is expressed as the gradient of the quadratic residual of a neural network. This allows obtaining convergence of the algorithm to a stationary point of an explicit function. The authors then show in [Hurault 2022b] that the proposed denoising step derives from the proximal operator of a scalar function. Note the interesting theoretical analysis of plug&play priors in the setting of MAP estimation in [Laumont 2022].

A major drawback of plug&play approaches is the need for manual tuning at evaluation. Indeed, parameters such as the noise level, the number of iterations or the stopping criteria should be tuned to obtain high quality results. Illustrations of the effect of the hyperparameters on the image quality can be found in [Wei 2022]. There have been several works to determine hyperparameters that yield good reconstruction. Amongst them, [Wei 2020, Wei 2022] propose to learn a policy network that determines the hyperparameters of the algorithm where the plug&play denoiser is used. The interested reader can refer to [Kamilov 2022] for a complete review of plug&play methods.

### Summary

#### Pros

- Interpretability of the denoising mapping
- Good study of the convergence properties
- Prior that once trained can be used for a wide range of applications

#### Cons

- Outperformed by unrolled networks trained for a specific task
- Manual tuning of the hyperparameters at evaluation

#### 1.2.4.3 Approximate inversion

In particular cases (such as MRI for well-spread sampling schemes), the matrix  $A(\xi)$  can be unitary, or a submatrix of a matrix that is almost unitary in the sense that its singular values are either equal to 0 or close to 1 [Aubel 2019]. In this case, the adjoint matrix is close to its pseudo-inverse  $A(\xi)^* \simeq A(\xi)^\dagger$ . A natural way to get a rough reconstruction is to consider  $\tilde{x} = A(\xi)^* y$ . It will then be contaminated by structured noise which is specific to the measurement operator  $A(\xi)$ . In CT imaging, this was a common practice and these artifacts were removed using the so-called filtered back-projection [Jin 2017]. However, the noise is still present in the image, and an option to remove it is to simply add a neural network at the output of this reconstructor.

**Definition 8** (Image to image neural network). Let  $\mathcal{D}_\theta : \mathbb{C}^N \mapsto \mathbb{C}^N$  denote a neural network that takes as an input an image in  $\mathbb{C}^N$  and returns another image of the same size. The parameters of this network are concatenated in the vector  $\theta$ .

This network is placed after the inversion and its role is to remove the artifacts present on this image and to denoise it:

$$\tilde{x} = \mathcal{D}_\theta(A(\xi)^*y). \quad (1.43)$$

Attention must be paid to the normalization of the operator when using the adjoint reconstructor. Indeed in the case of a neural network, this aspect is all the more important as the performance of a neural network highly depends on the normalization of the input data.

A more natural way to recover the image and denoise it is to perform an inversion of the forward operator  $A(\xi)$  and to use a neural network to denoise the resulting image by using the following reconstruction:

$$\tilde{x} = \mathcal{D}_\theta \left( (A(\xi)^*A(\xi) + \lambda\text{Id})^{-1} A(\xi)^*y \right). \quad (1.44)$$

This formulation has the advantage of providing a natural inversion of the forward model. The precision can be chosen by the user through the number of iterations in the conjugate gradient algorithm. It also pushes away the need to automatically differentiate the solver of the data fidelity term to train the neural network  $\mathcal{D}_\theta$  w.r.t.  $\theta$ . However, as the inversion problem is not solved exactly or as the inverse problem is ill-posed, the image before the denoiser can present artifacts. These are compensated by  $\mathcal{D}_\theta$  and this makes the learned denoiser highly dependent on the forward operator  $A(\xi)$ .

### Summary of the approximate inverse

#### Pros

- Computationally affordable

#### Cons

- The denoiser is operator dependent (requires training for each application)

#### 1.2.4.4 Unrolled networks

A natural approach in *deep learning* would be to learn a regularizer from a collection of training images. Of course, deriving a proximal mapping from a regularization based on neural networks is far from being trivial and unrolled networks only draw their inspiration from classical iterative algorithm. Instead of choosing a regularizer and computing its proximal operator to perform the optimization, the prox-

imal operator is directly replaced by a neural network [Sun 2016, Diamond 2017, Adler 2017, Zhang 2018, Dong 2018, Adler 2018, Hammernik 2019, Li 2019b]. It has been empirically proven that convolutional neural networks (CNNs) are excellent denoisers. As we want this network to be invariant to shifts in the image, it is natural to take a CNN. The structure of the network as well as the learned parameters and the optimizer used give the prior for the type of images recovered (see implicit regularization [Neyshabur 2015, Soudry 2018]). Note that there are approaches that learn the regularization term with specific structure and derive a proximal operator that is used in the unrolled network (see RED in Section 1.2.4.2).

In this section, we consider a list of ordered mappings that can be neural networks. We let  $(\mathcal{D}_\theta^k(x))_{k \leq K}$  denote this list of  $K$  networks.

The unrolled methods consist in unrolling a classical optimization algorithm with  $K$  iterations into  $K$  successions of operations. These operations depend on the parameters  $\theta$  and all these operations can therefore be seen as layers of a neural network. Figure 1.11 gives an illustration of a proximal gradient descent algorithm as  $K$  steps of an iterative algorithm parameterized by  $\theta$  where the number of iterations  $K$  is fixed once for all. In the classical framework where the proximal mapping is the soft thresholding, the parameter  $\theta$  corresponds to the weight of the  $\ell^1$  regularization. This algorithm takes as inputs:

- the vector of measurements  $y$ , which is involved in the expression of the gradient of  $F$ ;
- the forward operator  $A(\xi)$ ;
- the parameters  $\theta$ ;
- eventually an initialization of the unrolled algorithm  $x^{(0)}$ .

This algorithm returns the reconstructed image  $x^{(K)}$  which corresponds to the  $K$ -th iterate.

We give below two widely used unrolled algorithms although many variants of unrolled networks could be considered: the unrolled proximal gradient descent and the unrolled ADMM [Sun 2016, Dong 2018]. This section ends with a recent extension of unrolled networks to deep equilibrium networks.

**Unrolled proximal gradient descent** By replacing the proximal mapping, the iterative scheme of the unrolled proximal gradient descent is

$$x^{(k+1)} = \mathcal{D}_\theta^k \left( x^{(k)} - \gamma \nabla F(x^{(k)}) \right) \quad (1.45)$$

where we recall that  $F(x) = \frac{1}{2} \|A(\xi)x - y\|_2^2$  and  $\nabla F(x) = A(\xi)^* (A(\xi)x - y)$ .

Depending on the operator  $A(\xi)$ , a good step size  $\gamma$  can be set either manually by the user, computed analytically, or using numerical computations with the power iteration method to compute  $\|A(\xi)\|_{2 \rightarrow 2}$ :

$$\gamma = \frac{1}{\|A(\xi)\|_{2 \rightarrow 2}}.$$

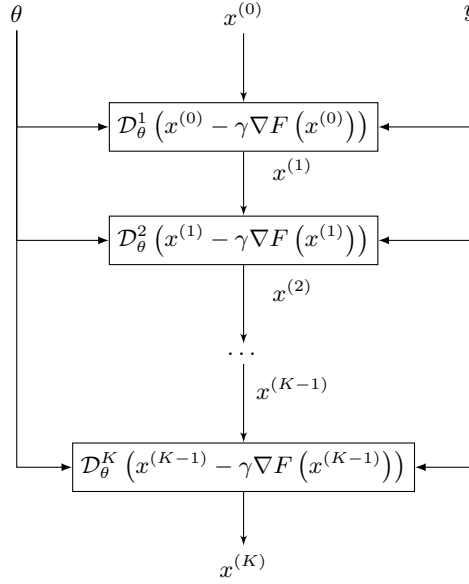


Figure 1.11: Illustration of an unrolled proximal gradient descent. The  $k$ -th proximal mapping is replaced by a denoiser  $\mathcal{D}_\theta^k$ .

There are also alternatives that consider the accelerated version of the proximal gradient descent, namely FISTA [Beck 2009], and unrolled FISTA has been proposed in [Xiang 2021].

The disadvantage of this method is that a proximal gradient descent can have a slow convergence speed and it can therefore require a large number of iterations to get a good solution. For classical approaches where the denoiser is not learned, this is only an issue in terms of computing time. When the denoiser is learned its gradient has to be computed during the training phase, which requires storing all the intermediate variables of each iteration. This bottleneck makes it impossible to use the unrolled proximal gradient descent with a large number of iterations in practice. On images of a few hundreds of pixels and on a modern GPU, memory issues start to appear after a few dozens of iterations.

### Unrolled ADMM

In order to introduce the unrolled ADMM, we first derive the classical ADMM algorithm in a general case. Once the proximal mapping that acts as a denoiser is identified, it is replaced by a neural network. We emphasize that once the proximal operator is replaced by another arbitrary mapping, the variational formulation falls apart. The problem we aim to solve is

$$\min_x \frac{1}{2} \|A(\xi)x - y\|_2^2 + \mathcal{R}_\theta(x).$$

Introducing the splitting  $x = z$ , this formulation is equivalent to

$$\min_{\substack{x, z \\ x=z}} \frac{1}{2} \|A(\xi)x - y\|_2^2 + \mathcal{R}_\theta(z). \quad (1.46)$$

In order to minimize (1.46), we use the augmented Lagrangian as in Section 1.2.2.2:

$$\mathcal{L}(x, z, \mu) = \frac{1}{2} \|A(\xi)x - y\|_2^2 + \mathcal{R}_\theta(z) + \frac{\beta}{2} \|x - z\|_2^2 + \langle \mu, x - z \rangle. \quad (1.47)$$

By repeating the same machinery of (1.34), the minimization steps yield an optimality equation that corresponds to the data fidelity term, and the proximal mapping of  $\mathcal{R}_\theta$ :

$$z^{(k+1)} = \arg \min_{z \in \mathbb{C}^N} \mathcal{R}_\theta(z) + \frac{\beta}{2} \left\| x^{(k+1)} + \frac{\mu^{(k)}}{\beta} - z \right\|_2^2$$

By replacing this proximal operator by a denoiser  $\mathcal{D}_\theta^k$ , we derive the unrolled ADMM:

$$\begin{cases} x^{(k+1)} = [A(\xi)^* A(\xi) + \beta \text{Id}]^{-1} (A(\xi)^* y + \beta z^{(k)} - \mu^{(k)}) \\ z^{(k+1)} = \mathcal{D}_\theta^k \left( x^{(k+1)} + \frac{\mu^{(k)}}{\beta} \right) \\ \mu^{(k+1)} = \mu^{(k)} + \beta (x^{(k+1)} - z^{(k+1)}) \end{cases} \quad (1.48)$$

and the reconstructed image is  $\tilde{x} = z^{(K)}$ .

As the whole algorithm is automatically differentiated with respect to the parameters of the networks  $\theta$ , it is important that the number of iterations in the conjugate gradient solving the first step of (1.48) remains fixed. Otherwise, it would be equivalent to changing the model at each call of the unrolled ADMM.

Notice that in the unrolled ADMM, each update w.r.t.  $x$  can be as costly as solving an entire Tikhonov formulation. In practice, if the parameter  $\beta$  is well chosen, the ADMM provides satisfactory results after only a few iterations provided that the problem w.r.t.  $x$  is solved with a good accuracy. This parameter  $\beta$  can vary with respect to the iterations to have a fast convergence. See [Zhang 2021b] for practical considerations and [Chan 2016] for theoretical convergence under assumptions on the denoiser  $\mathcal{D}_\theta^k$ .

**Practical limitations** From the construction of unrolled networks, the mappings used at each iteration are supposed to only act as denoisers and they do not invert the operator since all the information about  $A(\xi)$  is included in the steps where the data fidelity term is minimized. However, after one inversion of the data fidelity term, the noise term  $\epsilon$  results in structured noise in the recovered image. Hence when training unrolled networks, the denoisers learn to remove these operator specific artifacts. The main difference with plug&play priors is that, from the MAP estimation of Section 1.2.2.4, the denoisers of plug&play only encodes the image



prior while the denoisers of unrolled networks learn to remove structured noise. Numerical experiments reveal that when the operator  $A(\xi)$  is perturbed between training and evaluation in unrolled networks, the performance collapses. This performance drop is illustrated in Chapter 4.

From a practical point of view, it is important to bear in mind that the physical system is never perfectly modelled and that when real data  $y$  are taken, the reconstruction may not work well because the training was not done with the real measurement operator. This effect is called *an inverse crime* [Colton 1998]. Finally, if the physical acquisition system changes, so does the operator and the training must be performed again. On most practical problems, this training can take several weeks which is a bottleneck for many applications.

### Summary of the unrolled networks

#### Pros

- State-of-the-art results

#### Cons

- Computationally intensive
- Memory bottleneck
- The denoiser is operator dependent because of the structured noise (requires training for each application)

#### Open questions

- How to make unrolled networks robust to changes in the forward operator  $A(\xi)$ ?
- How does taking into account these changes impact the performance of these networks?
- If the mappings become robust to changes in  $A(\xi)$ :
  - How does this behave in comparison to P&P priors?
  - Is it possible to easily solve blind inverse problems (i.e.  $\xi$  is unknown and has to be retrieved)?

These questions are investigated in Chapter 4. In particular, we propose to train unrolled networks on a family of operators to make this network adaptive on different applications.

**Deep equilibrium networks** A major burden in unrolled networks is the memory requirement that constrains the number of iterations to be below a few dozens.

Inspired by works on plug&play methods, [Gilton 2021a] propose to view the solution given by an unrolled network as the solution of a fixed point scheme. The idea is to leverage the memory bottleneck by solving exactly the fixed point equation and using implicit differentiation to compute the gradient w.r.t. the parameters of the network. Mathematically, we seek to find the fixed point of an iterative scheme:

$$v^{(k+1)} = g(v^{(k+1)}, y, \theta, \xi, \theta) \quad (1.49)$$

with  $\theta$  the parameters of the denoiser. These schemes can be for example an unrolled ADMM or an unrolled proximal gradient descent. In the case of the deep equilibrium ADMM, the update vector is

$$v^{(k)} = \begin{pmatrix} x^{(k)} \\ z^{(k)} \\ \mu^{(k)} \end{pmatrix} \quad (1.50)$$

and the iterative scheme is the one of (1.48). Once an approximate solution  $\tilde{v}$  of the true solution  $v^{(\infty)}$  is found, the output can be plug into any differentiable function to compute the loss function. In order to optimize the parameters  $\theta$ , one needs the Jacobian of the solution  $\tilde{v}$  w.r.t.  $\theta$ . This can be computed analytically without using automatic differentiation through all the iterates  $v^{(k)}$ . For that purpose, we differentiate the equation  $v^{(\infty)} = g(v^{(\infty)}, y, \theta, \xi)$  using  $\tilde{v} \simeq v^{(\infty)}$ :

$$\frac{\partial \tilde{v}}{\partial \theta}(\tilde{v}, y, \theta, \xi) \simeq \frac{\partial g}{\partial v}(\tilde{v}, y, \theta, \xi) \frac{\partial \tilde{v}}{\partial \theta}(\tilde{v}, y, \theta, \xi) + \frac{\partial g}{\partial \theta}(\tilde{v}, y, \theta, \xi), \quad (1.51)$$

and then factorizing yields

$$\frac{\partial \tilde{v}}{\partial \theta}(\tilde{v}, y, \theta, \xi) \simeq \left[ \text{Id} - \frac{\partial g}{\partial v}(\tilde{v}, y, \theta, \xi) \right]^{-1} \frac{\partial g}{\partial \theta}(\tilde{v}, y, \theta, \xi). \quad (1.52)$$

Using Neumann series, the above term can be computed by truncating the following infinite sum:

$$\left[ \text{Id} - \frac{\partial g}{\partial v}(\tilde{v}, y, \theta, \xi) \right]^{-1} = \sum_{n=0}^{\infty} \left( \frac{\partial g}{\partial v}(\tilde{v}, y, \theta, \xi) \right)^n.$$

Once  $\frac{\partial \tilde{v}}{\partial \theta}(\tilde{v}, y, \theta, \xi)$  is approximated, computing the gradient of the loss function w.r.t.  $\theta$  is straightforward.

In [Gilton 2021a] the authors also propose to accelerate the fixed point scheme using Anderson acceleration. We refer the reader to their paper for these explanations.

This formulation allows using different number of iterations between the training and the evaluation modes which is not possible with unrolled networks without a performance drop. This formalism also looks similar to bilevel programming where the solution of a lower level problem is differentiated using implicit differentiation [Kunisch 2013] (see Section 1.2.4.1). One major drawback is that the implicit differentiation involves the true solution  $v^{(\infty)}$  of the fixed point scheme. If this fixed point equation is not solved with sufficient accuracy, it induces a bias in the gradient

estimation (1.52).

### Summary of the deep equilibrium networks

#### Pros

- Solves the memory bottleneck of unrolled networks
- Allows varying the number of iterations between training and evaluation

#### Cons

- Computationally intensive
- Potential misbehavior of the gradient if the solution is not computed with sufficient accuracy
- Recent approach and lack of insight on this model

#### 1.2.4.5 Generative models

We have seen that inspired by classical variational problems, it is possible to regularize and unroll a network using a denoiser as a proximal mapping. The structure of this regularization is essential for the performance of the unrolled network. If the optimization and the dataset are crucial, the architecture is even more so in the sense that it encodes the prior. A recent approach consists in using a neural network as a prior for an image reconstructor without learning this network beforehand: the Deep Image Prior (DIP) [Ulyanov 2018]. It is the convolutional architecture of the network that enforces the image to have a particular structure. The reconstruction consists in optimizing the weights of the network so that the output image corresponds to the observations through the measurement operator  $A(\xi)$ :

$$\min_{\theta} \frac{1}{2} \|A(\xi)\mathcal{D}_{\theta}(z) - y\|_2^2 \quad (1.53)$$

with  $z$  a vector in small dimension generated randomly once and for all. This allows solving a large class of problems including denoising, inpainting and super-resolution. Although giving impressive results, this method has three main disadvantages:

- Each reconstruction requires optimizing the weights of a neural network over several hundreds to thousands of iterations.
- The architecture of the network encodes the type of image that can be reconstructed and for each application, the structure of the network must be adapted. This requires an important optimization step of the hyperparameters that define the structure. If the network behaves well for one image, there is no guarantee that it will also work for a slightly modified image.

- The idea at the core of DIP is that when optimizing the weights, the structure of the image is fitted before the noise. If the number of iterations is not chosen with care and if the optimization is ran for too long, on applications like denoising the reconstructed image can have artifacts that increase with the number of iterations. This requires using early stopping.

These three disadvantages make this reconstruction method difficult to apply in practice. It advocates to construct a model that reduces the size of the parameter space to avoid the early stopping issue. This model should be also adapted to a particular application and encode the image prior by restricting the set of admissible images. If the type of image to be reconstructed is known (e.g. brain images in MRI) and the diversity of images is relatively low, there is a low-dimensional manifold on which the images live. It is therefore possible to train an image generator to map a low dimensional vector to realistic images. This approach is at the core of the Generative Adversarial Networks (GANs) which was first introduced in [Goodfellow 2014]. In this setting there are two networks in competition with each other. The first, called the *generative network*, generates images from a random vector. These images are then given to another network which role is to *discriminate* the true images in the training database from the fake images generated by the first network.

Once trained, the generative network generates images corresponding to the training database. For each reconstruction, the input vector of the network is optimized so as to generate an image which transformation through  $A(\xi)$  matches the observations:

$$\min_z \frac{1}{2} \|A(\xi)\mathcal{D}_\theta(z) - y\|_2^2 \quad (1.54)$$

This method was first proposed in [Bora 2017]. It has been investigated both with learned generative priors  $\mathcal{D}_\theta$  [Asim 2020a] and untrained priors [Asim 2020b]. This formalism allows solving blind inverse problems [Asim 2020b]. A major drawback of this method is that learned priors generate images that are highly dependent on the training dataset. It is therefore difficult to apply this type of method in concrete applications, particularly in the biomedical field where only limited datasets are accessible and where the images of main interest such as tumours are poorly represented in the datasets. Finally, the images are often outside of the span of  $\mathcal{D}_\theta$ . In practice this issue is resolved by using a splitting between the image and the prior  $\mathcal{D}_\theta(z)$  and regularization terms such as Total Variation are added [Asim 2020b]. In the end, this introduces a hand-crafted prior.

### Summary of the generative models

#### Pros

- State-of-the-art results for specific tasks
- Allows to solve blind inverse problems

**Cons**

- Potential heavy computations at evaluation
- For learned generative priors, highly dependent on the training dataset
- For learned generative priors, difficult to train

### 1.3 Introduction à l'IRM en français

Cette partie est une traduction en français du début de la Section 1.1 et présente le fonctionnement d'un scanner IRM et un modèle simplifié.

#### 1.3.1 Historique

Au cours des quatre dernières décennies, les scanners IRM sont devenus un outil crucial pour le diagnostic de nombreuses maladies et pour la recherche cognitive. Grâce à leur caractère non invasif et inoffensif, ils ont permis de nombreuses avancées scientifiques en permettant d'imager l'intérieur d'un corps sans avoir recours à la chirurgie. Les scanners IRM utilisés aujourd'hui en imagerie médicale sont l'aboutissement d'un très grand nombre de travaux dont les fondements théoriques sont issus des découvertes en Résonance Magnétique Nucléaire (RMN) dans la première moitié du 20<sup>ème</sup> siècle. Les premiers travaux abordant la question d'imager des tissus biologiques ont été publiés dans les années 1970. Raymond Vahan Damadian a été le premier à proposer d'améliorer le diagnostic du cancer en utilisant un dispositif permettant de caractériser le tissu sain du tissu tumoral. Cette méthode était basée sur la réponse magnétique différente du tissu cancéreux par rapport à celle du tissu sain. Par la suite, d'autres travaux ont abordé le problème de l'observation d'une image du corps humain [Lauterbur 1973, Mansfield 1977]. C'est pour ces travaux, qui ont permis de localiser l'information dans l'espace, que Paul Lauterbur et Peter Mansfield ont reçu le prix Nobel de physiologie ou médecine 2003. A partir de 1975, la technologie utilisée dans les IRM contemporains a été introduite par Richard Ernst. Elle est basée sur le codage par la fréquence et par la phase. Cette technologie, décrite dans la Section 1.3.2, permet de construire des séquences d'acquisition localisant l'information en 3D. Enfin, la décennie suivante voit la commercialisation des premiers scanners. Pour une recherche dont les fondements sont complexes, l'application à l'échelle industrielle des scanners est remarquable, puisque seulement 10 ans séparent les travaux fondateurs de leur application dans les scanners commerciaux.

#### 1.3.2 La physique derrière l'acquisition

Le fonctionnement des scanners IRM est basé sur les concepts subtils de Résonance Magnétique Nucléaire. Bien qu'il existe de nombreuses façons d'effectuer l'acquisition, la manière la plus simple de procéder est décrite ci-dessous. Certains



Figure 1.12: Raymond Vahan Damadian présentant son invention lors d'une conférence de presse en 1977. Credit: Copyright Bettmann/Corbis /AP Images (<http://cen.acs.org>)

détails sont omis, le but étant d'introduire le cadre mathématique simplifié. Pour plus d'informations, l'excellent site "Questions et réponses en IRM"<sup>3</sup> donne tous les détails et la physique associée à l'IRM. Une bonne explication de l'exploitation du phénomène de résonance en IRM est également donnée dans [Idy-Peretti 2009].

Avant de décrire le fonctionnement d'un scanner IRM, nous rappelons quelques principes de base de la Résonance Magnétique Nucléaire. Les noyaux possèdent un spin, une propriété électromagnétique qui décrit l'orientation de leur polarisation. Ce spin est représenté comme un vecteur en 3D et sa force possède différents états qui, à l'échelle de ces particules, sont tous discrets. Lorsqu'un champ magnétique est appliqué aux noyaux, les spins s'alignent dans la direction du champ magnétique. C'est cet effet qui est largement exploité par les scanners IRM pour mesurer la réponse d'un échantillon aux contraintes du champ magnétique.

Les scanners IRM utilisent la magnétisation des atomes d'hydrogène (proton seul) car ils sont présents en grand nombre dans les tissus vivants et leur proportion varie selon la nature du tissu. Dès lors que l'on peut mesurer la densité de ces atomes, il est possible de caractériser les tissus qui composent une image. L'objectif d'une IRM est de trouver la densité des protons qui sont excités. Ces protons sont excités par des champs magnétiques et c'est la réponse sur tout ou partie du volume qui est mesurée. Bien entendu, la réponse de l'ensemble du volume n'est d'aucune utilité puisqu'il n'est pas possible de localiser spatialement l'information. C'est là que toute l'ingénierie et la complexité des scanners IRM entrent en jeu pour permettre de récupérer l'image associée au volume imagé à partir des signaux mesurés. Le reste de cette sous-section est consacré à l'explication des principaux composants impliqués dans le phénomène physique exploité par les scanners IRM, et à l'explication schématique de la façon dont les signaux sont mesurés.

---

<sup>3</sup><https://www.mriquestions.com/>



(a) Un scanner IRM conventionnel d'une puissance de 1.5T (Philips scanner). Credit: Jan Ainali, (CC BY 3.0), <https://commons.wikimedia.org/wiki/File:MRI-Philips.JPG>



(b) Une nouvelle génération de scanner IRM sur roues et qui utilise un faible champ magnétique d'une puissance de 64mT (Hyperfine Swoop scanner). Credit: Hyperfine <https://hyperfine.io/>

Figure 1.13: Deux types de scanners IRM contemporains.

### 1.3.2.1 Principaux composants d'un scanner MRI

De manière simplifiée, un scanner IRM comporte 4 composants principaux : un champ magnétique principal, des bobines de gradient, des bobines de radiofréquence (RF) et des antennes de réception.

**Champ magnétique principal** Un champ magnétique principal, noté  $B_0$ , est appliqué à l'ensemble du volume scanné et il est aligné dans la direction  $e_z$ . Ce champ magnétique uniforme aligne les spins dans la direction  $e_z$ . Son intensité est de l'ordre de 1T sur les scanners conventionnels et sur une nouvelle génération de scanners elle est diminuée à  $\sim 60\text{mT}$  (voir Figure 1.13).

**Bobines de gradient** Il y a plusieurs groupes de bobines de gradient – sous une forme simplifiée trois, chacun correspondant à un axe. Les champs magnétiques induits par chacun de ces groupes de bobines sont alignés avec l'axe correspondant et l'intensité varie le long de chaque axe (typiquement 15 à 45mT/m sur les scanners conventionnels). Les bobines de gradient en  $z$  créent un champ magnétique supplémentaire aligné sur  $B_0$  et dont l'amplitude augmente linéairement le long de l'axe  $z$ . La superposition de ces deux champs donne un champ total dont l'intensité varie en fonction de l'axe  $z$ . La section suivante explique comment ce champ magnétique variable permet de sélectionner une coupe à exciter. C'est ce qu'on appelle le *codage par la fréquence*. Les autres bobines de gradient sur les axes  $x$  et  $y$  permettent de modifier la phase. Ces bobines sont impliquées dans le *codage par la phase*, par opposition au *codage par la fréquence* effectué par les bobines de gradient  $z$ .

**Bobines de radiofréquence** Les bobines RF génèrent un champ magnétique oscillant  $B_1$  qui est orthogonal au champ magnétique  $B_0$ . Ces bobines de radiofréquence émettent des signaux pendant une courte période de temps pour

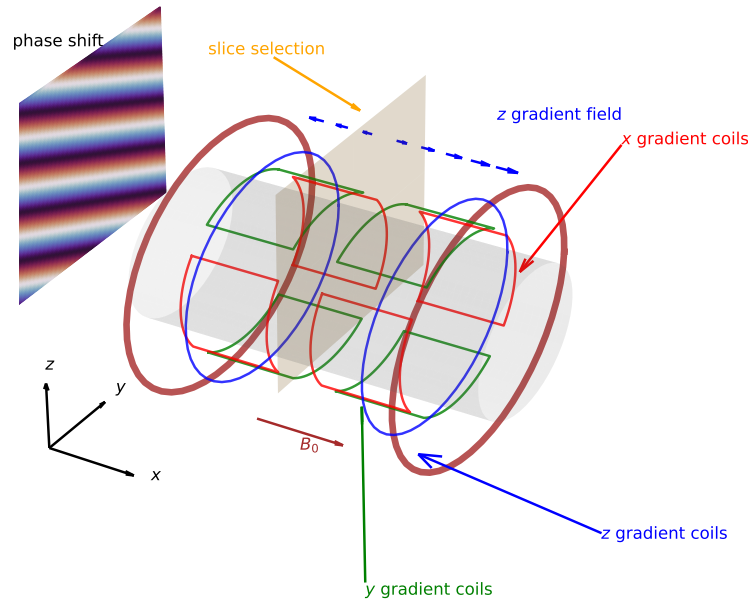


Figure 1.14: Principaux composants d'un scanner IRM. Le champ primaire  $B_0$  est uniforme (marron), et le champ variable dans l'espace est généré par les bobines de gradient en  $z$  (bleu). Cela permet de sélectionner une coupe dont les protons à l'intérieur font un mouvement de précession à une fréquence  $\omega_L$ . Les bobines de gradient  $x$  et  $y$  créent un déphasage dans ce plan (vert et rouge).

changer l'orientation du spin des protons. Le champ  $B_1$  oscille à une fréquence  $\omega_{RF}$  et l'intensité d'un tel champ est de l'ordre de  $\sim 10\text{mT}$ .

**Antennes de réception** Traditionnellement, les bobines RF faisaient également office de récepteurs pour mesurer la réponse. Elles tendent maintenant à être remplacées par des bobines dédiées à la réception du signal et sont placées tout autour du volume à scanner.

### 1.3.2.2 Acquisition

Maintenant que les différents éléments impliqués dans l'acquisition du signal d'un scanner IRM ont été présentés, nous décrivons comment l'acquisition est réalisée. La Figure 1.14 présente de manière schématique les principaux éléments d'un scanner IRM et un exemple de profil de déphasage dans le plan  $(x, y)$ .

**Sélection de coupe** Tout d'abord, un scanner IRM sélectionne une coupe à imager. Il tire parti d'un phénomène appelé *précession* qui se produit lorsque les spins oscillent et qu'ils sont soumis à un champ magnétique constant. Ce mouvement de précession est similaire à une toupie qui tourne vite mais dont l'axe de rotation varie dans le temps, décrivant des cercles. La fréquence de ce mouvement s'appelle la fréquence de Larmor. Elle est proportionnelle à l'intensité du champ magnétique



à travers la relation :

$$\omega_L = \gamma B \quad (1.55)$$

avec  $\omega_L$  la fréquence de Larmor [Hz],  $\gamma$  le rapport gyromagnétique [Hz/T] qui dépend du type de noyau qui est excité et  $B$  l'intensité du champ magnétique [T]. Comme l'intensité du champ magnétique  $B$  varie le long de l'axe  $z$ , la fréquence de Larmor  $\omega_L$  varie également. Lorsque les bobines RF génèrent le champ magnétique  $B_1$ , seuls les spins dont la fréquence de Larmor  $\omega_L$  est proche de  $\omega_{RF}$  sont excités et s'alignent sur  $B_1$ . Cela permet d'effectuer une *sélection de coupe* et de mesurer la réponse uniquement dans un plan  $(x, y)$ . Selon le champ de gradient en  $z$  qui est généré, la tranche sélectionnée peut être déplacée le long de l'axe  $z$ .

**Relaxation** Une fois que les spins sont alignés dans le plan  $(x, y)$ , le signal des bobines RF s'arrête. Les spins des protons effectuent alors le mouvement de précession. Ce mouvement génère un champ magnétique qui est mesuré par les antennes réceptrices situées autour du scanner. Notons qu'en pratique, lors de l'excitation et de la relaxation, ce ne sont jamais tous les spins qui s'alignent avec le champ magnétique mais seulement une infime partie. Comme en l'absence de champ magnétique, les spins sont désordonnés et comme c'est la moyenne sur tout le volume qui est mesurée, les contributions des spins qui ne s'alignent pas avec le champ magnétique sont en moyenne nulles.

**Codage par la phase** Pendant la relaxation, les bobines de gradient en  $x$  et  $y$  sont utilisées pour modifier la phase des spins à l'intérieur de la coupe. Comme le champ magnétique généré par les bobines de gradient en  $x$  et  $y$  varie dans l'espace, il permet de modifier l'intensité du champ magnétique total par rapport à la position dans le plan  $(x, y)$ . Cette modification de l'intensité de  $B$  fait varier la fréquence de Larmor par rapport à la position dans le plan  $(x, y)$ . Les spins n'entrent pas en précession à la même fréquence et cela induit un décalage de phase entre les spins dans le plan. Une fois que les champs de gradient en  $x$  et  $y$  sont arrêtés, les spins retrouvent leur fréquence de précession initiale et les décalages de phase ne changent plus. Cela permet de mesurer la réponse de la coupe pour un déphasage donné. Ce processus est répété entre chaque mesure pendant la relaxation afin de faire évoluer le déphasage dans le temps. Il fournit différentes informations qui sont ensuite utilisées pour la reconstruction. Dans la section suivante, nous expliquons que ce déphasage correspond exactement au terme d'une transformée de Fourier et que faire varier cette phase dans le temps correspond à faire varier la fréquence mesurée par la transformée de Fourier.

**Shot** Le processus d'excitation suivi de la relaxation est appelé un *shot*. Au cours d'un examen IRM, cette procédure est répétée plusieurs fois afin d'obtenir suffisamment d'échantillons pour reconstruire l'image. Elle est également répétée pour les différentes coupes afin d'imager un volume 3D avec un ensemble d'images 2D.

### 1.3.3 Un modèle simplifié

L'objectif de cette partie est, à partir de la représentation physique introduite précédemment, de donner les relations entre les signaux mesurés par chaque antenne autour du scanner et l'image à retrouver, c'est-à-dire la densité des protons dans l'espace. Les effets tels que la diminution au cours du temps de la puissance du signal reçu [Fessler 2010] sont volontairement laissés de côté.

**Modèle continu** Si on note  $\phi(x, y, t) \in \mathbb{C}$  le décalage de phase à l'instant  $t$  par rapport à la position  $(x, y)$  dans l'espace, et si on indice par  $1 \leq i \leq I$  les antennes, la  $i$ -ème antenne reçoit le signal

$$y_i(t) \stackrel{\text{def}}{=} \int_{\mathbb{R}^2} f(x, y) \sigma_i(x, y) \phi(x, y, t) dx dy \quad (1.56)$$

avec  $f$  l'image inconnue et  $y_i(t)$  le signal mesuré à l'instant  $t$ . Le terme  $\sigma_i(x, y)$  est appelé *carte de sensibilité* et il exprime la sensibilité de la  $i$ -ème bobine de réception dans l'espace. Chaque antenne reçoit plus ou moins de signal associé à une position  $(x, y)$ , en fonction de la distance à laquelle la bobine est placée. Cette carte est typiquement une fonction très régulière dont le module décroît avec la distance aux antennes. Comme la sensibilité des antennes diminue avec la distance, et comme les scanners sont souvent équipés d'une cage de Faraday empêchant les signaux extérieurs de pénétrer à l'intérieur, le domaine d'intégration peut être réduit à un rectangle autour du volume mesuré que l'on note  $\Omega \subset \mathbb{R}^2$ .

Nous rappelons que le décalage de phase  $\phi(x, y, t)$  est généré par les bobines de gradient en  $x$  et  $y$ . Il prend la forme d'une exponentielle complexe et s'écrit donc

$$\phi(x, y, t) = e^{-i(x\xi^{(x)}(t) + y\xi^{(y)}(t))} \quad (1.57)$$

avec  $\xi^{(x)}(t)$  et  $\xi^{(y)}(t)$  les fréquences par rapport aux axes  $x$  et  $y$  à l'instant  $t$ . Les fonctions  $\xi^{(x)}$  et  $\xi^{(y)}$  décrivent les *trajectoires* dans le domaine de Fourier. Ces trajectoires satisfont des contraintes liées à la physique du scanner qui sont détaillées dans le Chapitre 3.

**Modèle discret** En pratique, le signal  $y_i(t)$  est intégré par le scanner et il est discrétisé dans le temps de telle sorte que  $M$  mesures sont disponibles  $(y_i[m])_{m \leq M}$ . En discrétisant (1.56) aux différents pas de temps, le modèle s'écrit alors

$$y_i[m] = \int_{\Omega} f(x, y) \sigma_i(x, y) e^{-i(x\xi_m^{(x)} + y\xi_m^{(y)})} dx dy. \quad (1.58)$$

Les fréquences  $(\xi_m^{(x)}, \xi_m^{(y)})$  correspondent à la discrétisation de la trajectoire  $t \mapsto (\xi^{(x)}(t), \xi^{(y)}(t))$ . Une analyse des effets de la discrétisation et de l'intégration est donnée dans [Lazarus 2020a].

Pour l'instant, les quantités  $f$  et  $\sigma_i$  sont continues. Cependant, d'un point de vue numérique, elles sont représentées par des matrices de taille  $N_x \times N_y$  notées

respectivement  $u$  et  $s_i$ . Nous choisissons donc de les modéliser à l'aide d'une fonction d'interpolation  $\psi : \mathbb{R}^2 \rightarrow \mathbb{R}$  et la relation entre la version continue et discrète est la suivante

$$f = \left( \sum_{n=1}^N \delta_{p_n} u[n] \right) \star \psi \quad (1.59)$$

$$\sigma_i = \left( \sum_{n=1}^N \delta_{p_n} s_i[n] \right) \star \psi \quad (1.60)$$

avec  $N = N_x \times N_y$  et  $u \in \mathbb{C}^N$  (resp.  $s_i \in \mathbb{C}^N$ ) est la représentation discrète de  $f$  (resp.  $\sigma_i$ ). Le vecteur  $p_n$  correspond aux positions sur la grille 2D, i.e.  $p_n \in \left[ \left[ -\frac{N_x}{2}, \frac{N_x}{2} - 1 \right] \times \left[ \left[ -\frac{N_y}{2}, \frac{N_y}{2} - 1 \right] \right]$  et les pixels de  $u$  et  $s_i$  sont alignés sur cette grille.

En remplaçant les termes continus par leur version discrète dans (1.58) et en posant  $\xi_m = (\xi_m^{(x)}, \xi_m^{(y)})$  le  $m$ -ème point d'échantillonnage, on obtient

$$y_i[m] = \kappa(\xi_m) \sum_{n=1}^N u[n] s_i[n] e^{-i \langle p_n, \xi_m \rangle}. \quad (1.61)$$

La fonction  $\kappa$  est la transformée de Fourier de  $\psi$ . En pratique, la fonction d'interpolation constante est choisie dans ce manuscrit. Elle aligne les points  $p_n$  en bas à gauche de chaque pixel :

$$\psi(x, y) \stackrel{\text{def}}{=} \mathbf{1}_{0 \leq x \leq 1} \times \mathbf{1}_{0 \leq y \leq 1} \quad (1.62)$$

ce qui donne

$$\kappa(x, y) = \tilde{\kappa}(x) \tilde{\kappa}(y) \quad \text{with} \quad \tilde{\kappa}(x) \stackrel{\text{def}}{=} -e^{-ix/2} \text{sinc}(x/2). \quad (1.63)$$

Pour générer des décalages de phase d'une longueur d'onde de l'ordre du pixel (distance entre les points adjacents de  $p_n$ ) dans l'espace image, on voit dans (1.61) que l'amplitude des fréquences doit aller jusqu'à  $\pi$ . Les fréquences étant comprises dans  $[-\pi, \pi]^2$ , le terme  $\kappa$  ne varie donc pas beaucoup, mais surtout il peut être facilement corrigé sur les signaux mesurés de façon à l'éliminer dans les équations. Par la suite, nous l'ignorons car il ne modifie pas grandement le modèle. En supprimant cette dépendance, nous identifions dans le modèle (1.61) une Transformée de Fourier Non-Uniforme (NUFT) et l'équation peut être écrite sous forme de matrice comme suit

$$y_i = A(\xi) S_i u \quad (1.64)$$

où  $S_i$  est une matrice diagonale associée à la  $i$ -ème carte de sensibilité  $S_i = \text{diag}(s_i[n])_{1 \leq n \leq N}$  et  $A(\xi) : \mathbb{C}^N \rightarrow \mathbb{C}^M$  est la transformée de Fourier non-uniforme aux fréquences  $\xi \in ([-\pi, \pi]^2)^M \subset \mathbb{R}^{2M}$  avec

$$\xi \stackrel{\text{def}}{=} (\xi_m)_{1 \leq m \leq M}. \quad (1.65)$$

# Spurious minimizers in non-uniform Fourier sampling optimization

---

**Résumé** Une tendance récente en traitement du signal et des images est l'optimisation des schémas d'échantillonnage de Fourier pour des ensembles spécifiques de signaux. Dans ce chapitre, nous expliquons pourquoi le choix optimal de schémas d'échantillonnage de Fourier non cartésiens est un problème non convexe difficile en révélant deux problèmes d'optimisation. Le premier est l'existence d'un nombre combinatoire de minimiseurs parasites pour une classe générique de signaux. Le second est un effet de gradient évanescent pour les hautes fréquences. Nous concluons ce chapitre en montrant comment l'utilisation de grands ensembles de signaux peut atténuer le premier effet et nous illustrons expérimentalement les avantages de l'utilisation d'algorithmes de gradient stochastique avec une métrique variable.

**Abstract** A recent trend in the signal and image processing literature is the optimization of Fourier sampling schemes for specific datasets of signals. In this chapter, we explain why choosing optimal non Cartesian Fourier sampling patterns is a difficult nonconvex problem by bringing to light two optimization issues. The first one is the existence of a combinatorial number of spurious minimizers for a generic class of signals. The second one is a vanishing gradient effect for the high frequencies. We conclude this chapter by showing how using large datasets can mitigate the first effect and illustrate experimentally the benefits of using stochastic gradient algorithms with a variable metric.

This chapter is based on the publication [Gossard 2022b]:

**Gossard, A.**, de Gournay, F. & Weiss, P. (2022). Spurious minimizers in non uniform Fourier sampling optimization. *Inverse Problems*, 38(2022), 105003.

**Contents**


---

<b>2.1</b>	<b>Introduction</b>	<b>60</b>
<b>2.2</b>	<b>Notation</b>	<b>62</b>
<b>2.3</b>	<b>Preliminaries</b>	<b>62</b>
2.3.1	The setting	62
2.3.2	Elementary observations	64
<b>2.4</b>	<b>Theoretical issues</b>	<b>65</b>
2.4.1	Spurious minimizers	65
2.4.2	Numerical illustration of Theorem 1	68
2.4.3	Flatness for high frequencies	70
<b>2.5</b>	<b>Escaping the minimizers</b>	<b>71</b>
2.5.1	The effect of using a large dataset	71
2.5.2	Stochastic gradient descent	73
2.5.3	Variable metric	73
2.5.4	Numerical illustrations	73
<b>2.6</b>	<b>Conclusion</b>	<b>75</b>
<b>2.7</b>	<b>Proofs</b>	<b>77</b>
2.7.1	Proof of Proposition 11	77
2.7.2	Proof of Theorem 1	78
2.7.3	Proof of Proposition 12	79
2.7.4	Proof of Theorem 2	80

---

**2.1 Introduction**

Finding efficient Fourier sampling schemes is a critical issue in communications and imaging. This led to various theories including the celebrated Shannon-Nyquist theorems for bandlimited signals and compressed sensing for sparse signals. Unfortunately - in most practical cases - the signals to reconstruct are quite loosely described by these generic classes. For instance, magnetic resonance images of brains or knees have a rich structure due to the underlying object. It is therefore tempting to optimize a sampling scheme directly for a given dataset rather than relying on a rough mathematical model. The recent progresses in Graphical Processing Units (GPU) programming, automatic differentiation and machine learning make this idea even more tantalizing. In the sole field of Magnetic Resonance Imaging (MRI), the following list of references [Gözcü 2018, Jin 2019, Sherry 2020, Bahadir 2019, Zibetti 2021, Wang 2022a, Weiss 2021, Gossard 2012, Loktyushin 2021, Peng 2022, Aggarwal 2020] illustrates this novel trend.

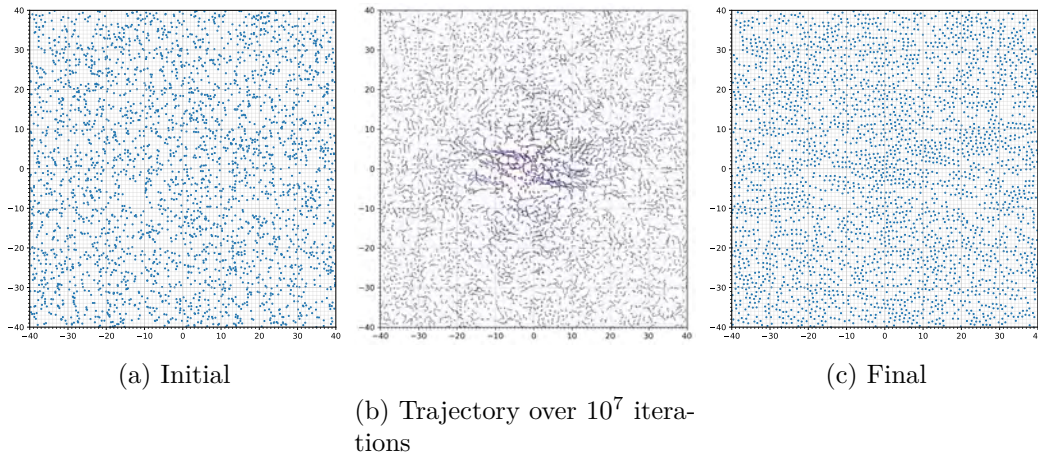


Figure 2.1: A typical sampling optimization trajectory. Starting from the sampling configuration on the left (uniform point process), we obtain the sampling scheme on the right after  $10^7$  iterations. The trajectory in the center corresponds to the  $10^7$  iterations of a gradient descent with fixed step size. Notice that the points clusters have disappeared, but that the scheme is still essentially uniform, while we would expect the low frequencies to be sampled more densely.

Unfortunately, most of the above works report (more or less explicitly) optimization issues. Fig. 2.1 illustrates one of them. In this example, we tried to optimize a sampling scheme for a single image from the fastMRI challenge [Zbontar 2018]. To this end, we minimize the  $\ell^2$  reconstruction error using a simple back-projection reconstructor with a subsampling factor of 2. The trajectory of a gradient descent is displayed in Fig. 2.1b. As can be seen, the final sampling set covers approximately uniformly the Fourier domain, while we would expect the low frequencies to be sampled more densely. This likely highlights the presence of a spurious minimizer.

The aim of this paper is to explain this phenomenon from a mathematical perspective and to bring some solutions to mitigate the difficulties. We focus on linear reconstruction methods, which simplifies the analysis and we highlight the critical role of the non-uniform Fourier transform as an oscillation generator. We expect that some of the arguments can be reused for more complex nonlinear reconstruction methods, which suffer from the same experimental issues. We also focus on optimization schemes that continuously optimize the positions of some sampling locations. These techniques have the advantage of not relying on a grid, which is an essential feature for various applications such as magnetic resonance imaging or radio-interferometry. In addition, they spark the hope of avoiding the curse of dimensionality encountered in combinatorial problems. We show that this dream is not realistic, but that the situation improves by considering large signals datasets and specific variable metric techniques. We conclude the paper by illustrating our findings on 1D experiments.

## 2.2 Notation

In this paper, we will focus on discrete 1D signals, for the ease of exposition. However, the main arguments apply to arbitrary dimensions and continuous signals as well.

We consider a signal  $u$  as a vector of  $\mathbb{C}^N$  with  $N \in 2\mathbb{N}$ . We let  $\mathcal{N} = \llbracket -\frac{N}{2}, \frac{N}{2} - 1 \rrbracket$ . An alternative way to represent a signal  $u \in \mathbb{C}^N$  is to use a discrete measure  $\mu$  of the form:

$$\mu = \sum_{n \in \mathcal{N}} u_n \delta_{\frac{n}{N}}. \quad (2.1)$$

Given a location  $\xi \in \mathbb{R}$ , we define:

$$\hat{u}(\xi) \stackrel{\text{def}}{=} \frac{1}{\sqrt{N}} \sum_{n \in \mathcal{N}} u_n e^{-2i\pi \langle \xi, \frac{n}{N} \rangle}, \quad (2.2)$$

which can be seen as the continuous Fourier transform of the measure  $\mu$ . We consider  $\Xi = [\xi_1, \dots, \xi_M] \in \mathbb{R}^M$  a set of  $M$  locations. The Fourier transform  $\hat{u}(\Xi) \in \mathbb{C}^M$  at the locations  $\Xi$  can be written as a matrix-vector product of the form  $\hat{u}(\Xi) = A(\Xi)^* u$  with the normalized Vandermonde matrix  $A(\Xi) \in \mathbb{C}^{N \times M}$  defined by

$$A(\Xi)_{n,m} \stackrel{\text{def}}{=} \frac{1}{\sqrt{N}} e^{2i\pi \langle \xi_m, \frac{n}{N} \rangle}.$$

In what follows, we let  $a(\xi) \in \mathbb{C}^N$  denote the vector defined for all  $n \in \mathcal{N}$  by

$$a(\xi)[n] \stackrel{\text{def}}{=} \frac{1}{\sqrt{N}} e^{2i\pi \langle \xi, \frac{n}{N} \rangle},$$

so that

$$A(\Xi) = [a(\xi_1), \dots, a(\xi_M)].$$

The matrix  $A(\Xi)^*$  can be seen as the nonuniform Fourier transform [Oppenheim 1971] from the grid to the set of sampling locations  $\Xi$ . We let  $(A(\Xi)^*)^\dagger$  denote the pseudo-inverse of  $A(\Xi)^*$ .

## 2.3 Preliminaries

Below, we first describe the precise mathematical setting and then turn to some preliminary results.

### 2.3.1 The setting

Let  $u \in \mathbb{C}^N$  denote a signal. We assume that a sampling device allows picking  $M$  frequencies  $\xi_1, \dots, \xi_M$  in  $\mathbb{R}$ , yielding the set of measurements  $y = A(\Xi)^* u + w$  with  $w \sim \mathcal{N}(0, \sigma^2 \text{Id})$  a white Gaussian noise. A vast amount of reconstruction techniques have been designed in the literature to reconstruct  $u$  from  $y$ . A generic reconstructor can be defined as a mapping  $\mathcal{R} : (\mathbb{C}^M \times \mathbb{R}^M) \rightarrow \mathbb{C}^N$  that takes as an input a

measurement  $y \in \mathbb{C}^M$  and a sampling scheme  $\Xi \in \mathbb{R}^M$  and outputs a reconstructed signal  $\mathcal{R}(y, \Xi)$ . Given a collection of signals  $u_1, \dots, u_P$  and a reconstructor  $\mathcal{R}$ , a natural framework to find the best sampling scheme  $\Xi$  is to solve the following optimization problem:

$$\inf_{\Xi \in \mathbb{R}^M} \frac{1}{2P} \sum_{p=1}^P \mathbb{E}_w(\|\mathcal{R}(A(\Xi)^* u_p + w, \Xi) - u_p\|_2^2). \quad (2.3)$$

This problem can be attacked with first order methods that continuously optimize the sampling locations  $\xi_m$ , see for instance [Weiss 2021, Gossard 2012, Wang 2022a]. In this work, we will concentrate on three simple linear reconstruction methods of the form  $\mathcal{R}(y, \Xi) = R(\Xi)y$ :

**The back-projection method** which consists in defining the reconstructor as  $R_1(\Xi) = A(\Xi)$  or

$$\mathcal{R}_1(y, \Xi) \stackrel{\text{def}}{=} A(\Xi)y. \quad (2.4)$$

**The pseudo-inverse method** where the reconstructor is defined with  $R_2(\Xi) = (A(\Xi)^*)^\dagger$  or

$$\mathcal{R}_2(y, \Xi) \stackrel{\text{def}}{=} (A(\Xi)^*)^\dagger y. \quad (2.5)$$

**The Tikhonov method** (or regularized inverse) which consists in solving the following quadratic problem:

$$\mathcal{R}_3(y, \Xi) \stackrel{\text{def}}{=} (1 + \lambda) \arg \min_{f \in \mathbb{C}^N} \frac{1}{2} \|A(\Xi)^* f - y\|_2^2 + \frac{\lambda}{2} \|f\|_2^2 \quad (2.6)$$

for  $\lambda > 0$ . Hence

$$R_3(\Xi) = (1 + \lambda) (A(\Xi)A(\Xi)^* + \lambda \text{Id})^{-1} A(\Xi). \quad (2.7)$$

The multiplication by  $(1 + \lambda)$  is there to compensate the bias introduced by the regularization and will later simplify the expressions. A similar analysis can be carried out for the more standard solver  $R_3(\Xi) = (A(\Xi)A(\Xi)^* + \lambda \text{Id})^{-1} A(\Xi)$ , but it leads to significantly more complicated formulas, which we prefer avoiding for the sake of readability.

These techniques are quite popular in the actual practice. We restrict our analysis to linear reconstructors of the type (2.4), (2.5) and (2.6) for simplicity reasons. Note that (2.5) corresponds to the limit case of (2.6) when  $\lambda$  tends to zero. Numerical experiments reveal that the optimization issues raised in Theorems 1 and 2 also apply to nonlinear reconstructors such as sparsity promoting convex penalties. However, the techniques used in the proofs do not directly extend to this framework.

We first analyze the problem with a single image  $u$  in the dataset, i.e.  $P = 1$ . Let us define three cost functions  $J_1$ ,  $J_2$  and  $J_3$  which respectively correspond to



the back-projection, the pseudo-inverse and the regularized inverse.

**Definition 9** (Cost function). *Given a signal  $u$ , a sampling scheme  $\Xi$  and a reconstruction method  $R(\Xi)$ , the cost function reads*

$$J(\Xi) \stackrel{\text{def}}{=} \mathbb{E}_w \left( \frac{1}{2} \|R(\Xi)(A(\Xi)^*u + w) - u\|_2^2 \right) \quad (2.8)$$

where  $w \sim \mathcal{N}(0, \sigma^2 \text{Id})$  is white Gaussian noise.

### 2.3.2 Elementary observations

We will make use of the following definitions.

**Definition 10** (The min distance). *Given a set of sampling points  $\Xi$ , the min distance  $\text{md}(\Xi)$  is defined by*

$$\text{md}(\Xi) \stackrel{\text{def}}{=} \min_{m \neq m'} \text{dist}(\xi_m, \xi_{m'})$$

where  $\text{dist}$  is the distance on the torus defined for  $(\xi_1, \xi_2) \in \mathbb{R}^2$  as

$$\text{dist}(\xi_1, \xi_2) \stackrel{\text{def}}{=} \inf_{k \in \mathbb{Z}} \|\xi_1 - \xi_2 - kN\|_\infty. \quad (2.9)$$

**Definition 11** (Subgrid). *Throughout the paper, we say that  $\Xi \in [-N/2, N/2]^M$  is a subgrid if  $\xi_m - \xi_{m'} \in \mathbb{Z}^*$  for all  $m \neq m'$ .*

**Proposition 6** ( $J$  is  $N$ -periodic). *We have*

$$J(\Xi \bmod N) = J(\Xi). \quad (2.10)$$

*Proof.* Let  $n = kN$  with  $k \in \mathbb{N}$ . The proof simply stems from the fact that  $a(\xi + n) = a(\xi)$ .  $\square$

The previous proposition shows that we can restrict our attention to frequencies  $\xi$  belonging to the set  $[-N/2, N/2[$ .

**Proposition 7** (Existence of minimizers). *For any  $M \in \mathbb{N}$  and any  $u \in \mathbb{C}^N$ , there exists at least one minimizer of  $J$  on  $[-N/2, N/2]^M$ .*

*Proof.* We start by noticing that  $J$  is a  $C^\infty$  function since it is defined as a composition of  $C^\infty$  functions. Hence it is also continuous on  $[-N/2, N/2]^M$ . This yields the existence of at least one minimizer.  $\square$

Now we proceed to a reformulation of the problem by rearranging the terms involved in the definition of  $J$ .

**Proposition 8.** *The reconstructors associated to  $J_1$ ,  $J_2$  and  $J_3$  defined in (2.8) can be expressed as*

$$R(\Xi) = A(\Xi)Q(\Xi) \quad (2.11)$$

(i.e. the solution lives in  $\text{ran}(A)$ ) with:

- $Q_1(\Xi) = \text{Id}$
- $Q_2(\Xi) = (A(\Xi)^*A(\Xi))^\dagger$
- $Q_3(\Xi) = (1 + \lambda)(A(\Xi)^*A(\Xi) + \lambda\text{Id})^{-1}$ .

*Proof.* For  $Q_1$ , there is nothing to prove. For  $Q_2$ , we use one of the standard properties of the pseudo-inverse. For  $Q_3$ , we use the equality  $A(A^*A + \lambda\text{Id}) = (AA^* + \lambda\text{Id})A$  and then left multiply by  $(AA^* + \lambda\text{Id})^{-1}$  and right multiply by  $(A^*A + \lambda\text{Id})^{-1}$ .  $\square$

**Proposition 9.** *Letting  $\hat{u}(\Xi) = A(\Xi)^*u$ , we have*

$$\begin{aligned} J(\Xi) &= \frac{1}{2}\|u\|_2^2 - \langle Q(\Xi)\hat{u}(\Xi), \hat{u}(\Xi) \rangle \\ &\quad + \frac{1}{2}\|R(\Xi)\hat{u}(\Xi)\|_2^2 + \frac{1}{2}\mathbb{E}_w \left( \|R(\Xi)w\|_2^2 \right) \end{aligned} \quad (2.12)$$

*Proof.* We drop the dependency in  $\Xi$  to simplify the notation.

$$\begin{aligned} 2J &= \|u\|_2^2 + \|RA^*u\|_2^2 + \mathbb{E}_w \left( \|Rw\|_2^2 \right) + 2\mathbb{E}_w \left( \text{Re}\langle RA^*u - u, Rw \rangle \right) - 2\text{Re}\langle RA^*u, u \rangle \\ &= \|u\|_2^2 + \|RA^*u\|_2^2 + \mathbb{E}_w \left( \|Rw\|_2^2 \right) - 2\langle Q\hat{u}, \hat{u} \rangle \end{aligned}$$

where we used  $Q(\Xi)^* = Q(\Xi)$  and  $\mathbb{E}_w(w) = 0$ .  $\square$

Equation (2.12) greatly simplifies when  $\Xi$  is a subgrid. Let us define the following function

$$\tilde{J}(\Xi) \stackrel{\text{def}}{=} \frac{1}{2}\|u\|_2^2 - \frac{1}{2}\|\hat{u}(\Xi)\|_2^2 + \frac{\sigma^2 M}{2}. \quad (2.13)$$

**Proposition 10.** *When  $\Xi$  is a subgrid,  $J(\Xi) = \tilde{J}(\Xi)$ .*

*Proof.* When  $\Xi$  is a subgrid, we have  $A(\Xi)^*A(\Xi) = \text{Id}$  by the orthogonality of the basis associated with the FFT. We use the decomposition of Proposition 9 with  $Q(\Xi) = \text{Id}$ ,  $R(\Xi)^*R(\Xi) = \text{Id}$ .  $\square$

## 2.4 Theoretical issues

In this section, we give the main theoretical results of the paper.

### 2.4.1 Spurious minimizers

The aim of this section is to illustrate a common situation where the function  $J$  possesses a combinatorial number of minimizers. We construct examples where the function  $\tilde{J}$  defined in (2.13) is very oscillatory, while  $J - \tilde{J}$  is of small amplitude.

The function  $J$  is close to  $\tilde{J}$  not only for subgrids as in Proposition 10 but also for well-spread schemes. Following the proof of Proposition 10, and the decomposition of Proposition 9, it is sufficient to control how close  $Q(\Xi)$  and  $R(\Xi)^*R(\Xi)$  are to  $\text{Id}$ . This is the aim of the following proposition.

**Proposition 11** (Bound on  $Q$  and  $R^*R$ ). *Consider a sampling pattern  $\Xi$  such that  $\text{md}(\Xi) > 1$  and set  $\epsilon = 1/\text{md}(\Xi)$ . Then*

$$-a_i \text{Id} \preceq Q_i - \text{Id} \preceq a_i \text{Id} \quad (2.14)$$

$$-b_i \text{Id} \preceq R_i^* R_i - \text{Id} \preceq b_i \text{Id}, \quad (2.15)$$

with

$$\begin{aligned} a_1 &= 0, & a_2 &= \frac{\epsilon}{1-\epsilon}, & a_3 &= \frac{\epsilon}{1-\epsilon} \\ b_1 &= \epsilon, & b_2 &= \frac{\epsilon}{1-\epsilon}, & b_3 &= \frac{4\epsilon}{(1-\epsilon)^2} \end{aligned}$$

The proof is postponed to Section 2.7.1.

**Theorem 1** (A combinatorial number of minimizers). *Set a number of samples  $M \in \mathbb{N}$  and consider a vector  $u \in \mathbb{C}^N$  such that the following properties are verified*

- i) *The modulus  $|\hat{u}|^2$  possesses a subset of  $K \geq M$  local maximizers  $Z = \{\zeta_1, \dots, \zeta_K\}$  separated by a distance at least  $\delta = \text{md}(Z)$  with  $\delta > 1 + 2r$  for some  $r > 0$ .*
- ii) *The modulus  $|\hat{u}|^2$  is locally strictly concave for each  $\zeta_k$ :*

$$|\hat{u}|^2(\zeta_k + h) \leq |\hat{u}|^2(\zeta_k) - \frac{c}{2}h^2, \forall h \in [-r, r]$$

for some  $c > 0$ .

- iii) *For any subset  $\bar{\Xi}$  of  $M$  distinct points in  $Z$ , we have*

$$\frac{cr^2}{2} > (b + 2a) \|\hat{u}(\bar{\Xi})\|_2^2 + bM\sigma^2 \quad (2.16)$$

where  $a$  and  $b$  are given in Proposition 11 with  $\epsilon = \frac{1}{\delta - 2r}$ .

Then, the function  $J$  possesses at least  $\binom{K}{M} \cdot M!$  local minimizers.

The proof of Theorem 1 is postponed to Section 2.7.2. The conditions in Theorem 1 may look cryptic at first sight. We first show a simple example of a function  $u$  that verifies the hypotheses and leads to a huge number of critical points.

**Corollary 1.** *Assume that  $N \in 4\mathbb{N}$  and define  $u \in \mathbb{C}^N$  as follows*

$$u[n] = \begin{cases} \sqrt{N}/2 & \text{if } n = \pm N/4, \\ 0 & \text{otherwise.} \end{cases} \quad (2.17)$$

Let  $M = \lfloor \eta \sqrt{N} \rfloor$  with  $\eta = \frac{\pi^2 \sqrt{2}}{256 \cdot (20 + 16\sigma^2)}$  then all the functions  $J_i$  possess a number of minimizers larger than  $M! \cdot \left(\frac{1}{2\eta}\right)^{\eta \sqrt{N}}$ .

For  $\sigma \leq 1$ , the bound holds for  $\eta = 3 \cdot 10^{-3}$ . For  $\sigma = 0$  and  $J_1$  the bound can be increased to  $\eta = 1.09 \cdot 10^{-1}$ .

*Proof.* The choice of  $u$  in (2.17) leads to the oscillatory function  $\hat{u}(\xi) = \cos\left(\frac{\pi}{2}\xi\right)$ . The modulus  $|\hat{u}|$  is maximal at every point  $\xi \in 2\mathbb{N}$ . Let  $\xi_0 \in 2\mathbb{N}$  and set  $r = \frac{1}{4}$ . For any  $\xi \in [\xi_0 - r, \xi_0 + r]$ , we have

$$\begin{aligned} (|\hat{u}|^2)''(\xi) &= \frac{\pi^2}{2} \left( \sin^2\left(\frac{\pi}{2}\xi\right) - \cos^2\left(\frac{\pi}{2}\xi\right) \right) \\ &\leq -\frac{\pi^2}{2\sqrt{2}}. \end{aligned}$$

Let  $p \in \mathbb{N}$ . The conditions i) and ii) of Theorem 1 are satisfied with  $Z = 2p\mathbb{N} \cap [-N/2, N/2[$ ,  $K = \lfloor N/(2p) \rfloor$ ,  $r = 1/4$ ,  $c = \frac{\pi^2 \sqrt{2}}{8}$ ,  $\delta = 2p$ . Further notice that for every set  $\bar{\Xi} \in Z^M$ ,  $\|\hat{u}(\bar{\Xi})\|_2^2 = M$ .

For this example, the condition (2.16) therefore reads

$$M < \frac{\pi^2 \sqrt{2}}{256} \cdot \left( \frac{1}{b + 2a + b\sigma^2} \right). \quad (2.18)$$

As long as this condition is satisfied, Theorem 1 allows concluding on the existence of  $\binom{\lfloor N/2p \rfloor}{M} \cdot M!$  maximizers.

Now, if  $\delta - 2r \geq 2$ , we can coarsely simplify the bounds in Proposition 11 as

$$a \leq \frac{2}{\delta - 2r} \quad \text{and} \quad b \leq \frac{16}{\delta - 2r}.$$

Hence, a combinatorial number of minimizers is granted given that

$$M < \frac{\pi^2 \sqrt{2}}{256} \cdot \left( \frac{\delta - 2r}{20 + 16\sigma^2} \right). \quad (2.19)$$

Now, take  $p = \lfloor \sqrt{N} \rfloor$  and  $M = \lfloor \eta \cdot \sqrt{N} \rfloor$  with  $\eta = \frac{\pi^2 \sqrt{2}}{128 \cdot (20 + 16\sigma^2)}$ . Then Theorem 1 yields a number of minimizers larger than  $\binom{\lfloor \sqrt{N}/2 \rfloor}{\lfloor \eta \cdot \sqrt{N} \rfloor} \cdot M!$ . Using the standard bound

$$\binom{n}{k} \geq \left(\frac{n}{k}\right)^k \quad (2.20)$$

yields a number of minimizers larger than  $\left(\frac{1}{2\eta}\right)^{\eta \sqrt{N}}$ .

In particular for  $\sigma < 1$  this yields  $\eta = 0.003$ . The bound can be increased to  $\eta = 0.109$  for  $\sigma = 0$  and  $J_1$ .  $\square$

### 2.4.2 Numerical illustration of Theorem 1

In this section we illustrate Theorem 1 through numerical examples in Fig. 2.2. We first consider the noiseless settings  $\sigma = 0$  and illustrate the existence of spurious minimizers for the back-projection and the pseudo-inverse methods.

We introduce the following function

$$F(\Xi) \stackrel{\text{def}}{=} \frac{1}{2} \sum_{m=1}^M |\hat{u}(\xi_m)|^2 = \frac{1}{2} \|\hat{u}(\Xi)\|_2^2, \quad (2.21)$$

which somehow measures the energy captured within a sampling scheme  $\Xi$ . We also introduce the functions  $G_1$  and  $G_2$  such that  $J_1 = \frac{1}{2}\|u\|_2^2 - F + G_1$  and  $J_2 = \frac{1}{2}\|u\|_2^2 - F + G_2$ . Using Proposition 9, we have

$$G_1(\Xi) = \frac{1}{2} \langle (A(\Xi)^* A(\Xi) - \text{Id}) \hat{u}(\Xi), \hat{u}(\Xi) \rangle, \quad (2.22)$$

and

$$G_2(\Xi) = \frac{1}{2} \langle (\text{Id} - (A(\Xi)^* A(\Xi))^\dagger) \hat{u}(\Xi), \hat{u}(\Xi) \rangle. \quad (2.23)$$

From the left to the right, we used three different 1D signals: a high frequency cosine, a low frequency sine and a Gaussian. We plot the different energy landscapes, for  $M = 2$  measurements at locations  $\Xi = \{\xi_1, \xi_2\}$  and  $N = 16$ . From the top to the bottom, we display the functions  $J_1, J_2, G_1, G_2, -F$  and the modulus of the Fourier transform  $\xi \mapsto |\hat{u}(\xi)|$ . In order to understand the effect of the signal's structure, the local minima of  $J_1, J_2, G_1, G_2$  and  $-F$  are represented with red dots.

First notice that the cost functions are symmetric with respect to the diagonal. This simply reflects the fact that permutation of points lead to the same energy, and this illustrates the factor  $M!$  in Theorem 1.

As can be seen in all cases, the functions  $G_1$  and  $G_2$  vanish far away from the diagonal (see Proposition 11). These point configurations correspond to well-spread sampling schemes. On the contrary, the function  $-F$  can have a large amplitude even outside the diagonal. These two properties are the main ingredients to prove Theorem 1.

The left column (high frequency cosine), corresponds to the example in Corollary 1. We see a number of minimizers that seems quadratic in  $N$  for  $M = 2$ . The center column (low frequency sine) shows that the number of minimizers decreases with a higher regularity of the signal, by reducing the oscillations in  $F$ . On the right (Gaussian function), we illustrate a case where  $F$  has only one local maximum. Even in this case, the function  $J$  still has valleys with shallow local minima. The same phenomenon appears in the center (low frequency sine). Notice that this phenomenon is not captured by Theorem 1, which only relies on local maximizers of  $F$ . In these two examples, the oscillations are induced by the function  $G$ , which we do not explore in this paper.

In Fig. 2.3, the energy profile of  $J_3$  is displayed with the low frequency signal (see Fig. 2.2 center column) for  $M = 2$  and for various noise levels  $\sigma$  and regularization

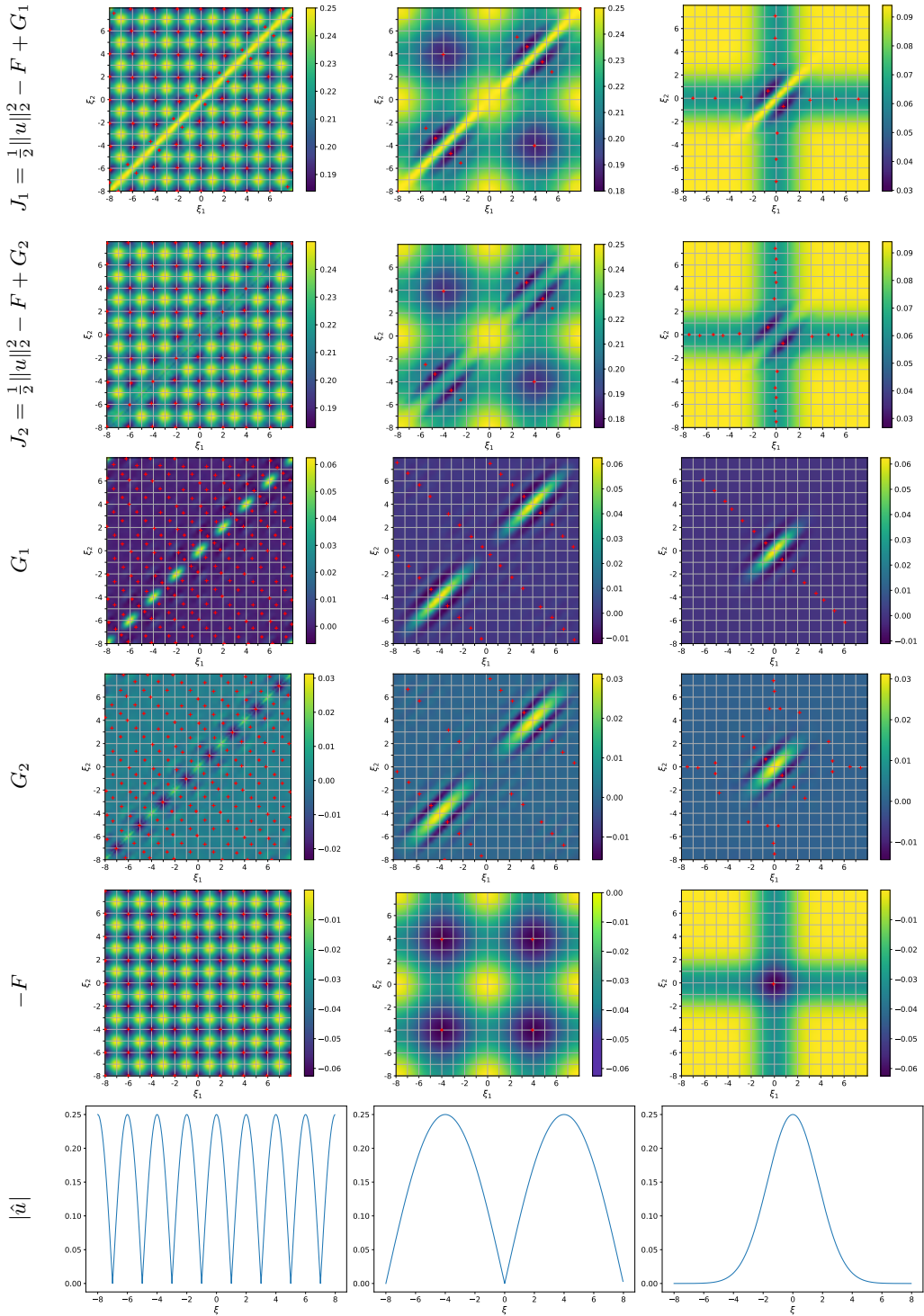


Figure 2.2: The energy profile for  $M = 2$  and three different signals  $\hat{u}$ : a high frequency cosine, a low frequency sine and a Gaussian (from left to right). From top to bottom, we represent  $J_1$ ,  $J_2$ ,  $G_1$ ,  $G_2$ ,  $F$  and  $|\hat{u}|$ . The red dots represent local minima.

parameters  $\lambda$ .

Neither the noise, nor the regularization parameter  $\lambda$  are able to remove the local minimizers of  $J_3$  which are displayed by red dots. The last column is a critical case where the noise prevails over the signal and the reconstruction error is high (typically  $\sim 0.18$  in the noiseless setting and  $\sim 0.5$  with  $\sigma = 5 \times 10^{-1}$ ).

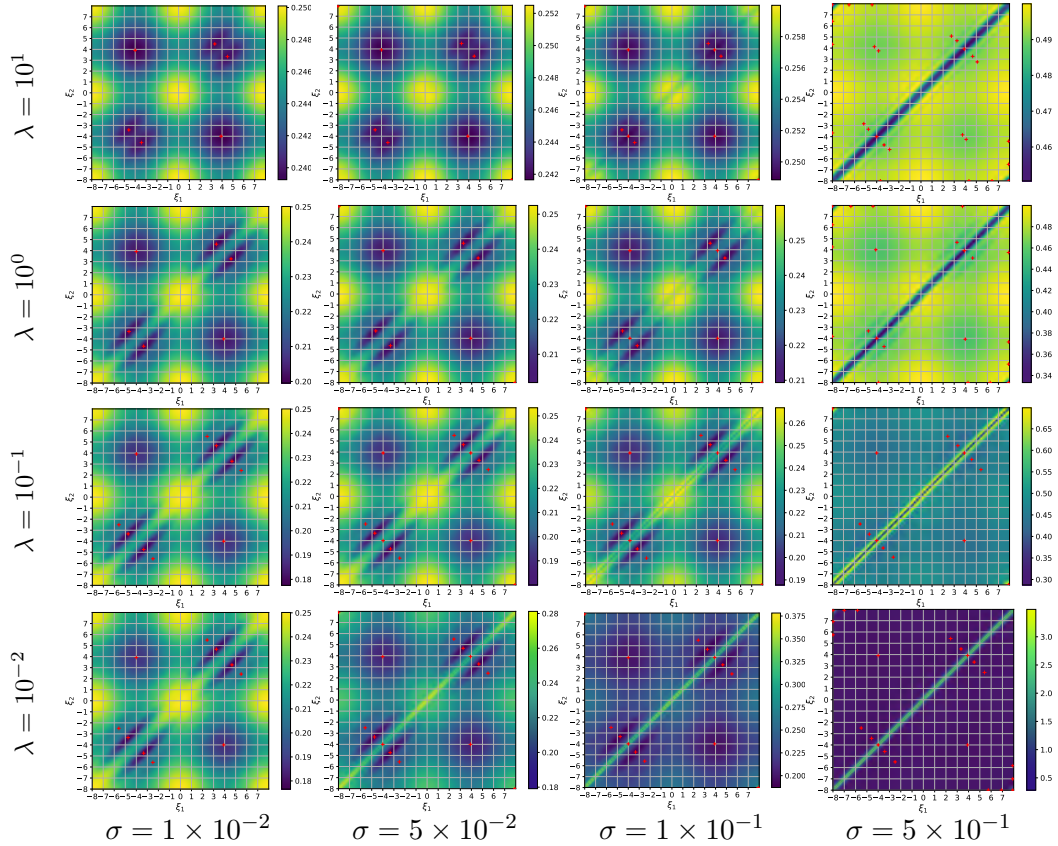


Figure 2.3: The energy profile of  $J_3$  for  $M = 2$  with the low frequency signal (see center column of Fig. 2.2) for different noise levels  $\sigma$  and regularization  $\lambda$ . The red dots represent local minima.

### 2.4.3 Flatness for high frequencies

In this paragraph we show that the partial derivatives of the cost function may vanish, for indexes corresponding to high frequencies. This explains another practical difficulty in Fourier sampling optimization: without using variable metric techniques, the sampling points located in the high frequencies move very slowly. Though our proof only applies to the function  $J_1$ , this effect also seems to occur for  $J_2$ . See for instance the four corners of Fig. 2.2, right.

**Proposition 12.** *Letting  $r$  denote the residual error function*

$$r(\Xi) = A(\Xi)A(\Xi)^*u - u, \quad (2.24)$$

the gradient of the cost function  $J_1$  reads:

$$\nabla J_1(\Xi) = \operatorname{Re} \left( \nabla \left( \hat{u}(\Xi) \odot \overline{\hat{r}(\Xi)} \right) \right), \quad (2.25)$$

where  $\nabla$  in the right-hand-side denotes the usual derivative in 1D or the gradient in higher dimension and where  $\odot$  is the coordinate-wise (Hadamard) product.

The proof of Proposition 12 is postponed to Section 2.7.3.

**Theorem 2** (Vanishing gradients for high frequencies). *Consider a signal  $u \in \mathbb{C}^N$  and a point configuration  $\Xi \in \mathbb{R}^M$ . Under the decay assumptions*

$$|\hat{u}(\xi)| \lesssim \frac{1}{|\xi|^\alpha} \quad \text{and} \quad |\hat{u}'(\xi)| \lesssim \frac{1}{|\xi|^\alpha}, \quad (2.26)$$

with  $\alpha > 0$ , we have

$$\left| \frac{\partial J_1(\Xi)}{\partial \xi_m} \right| \lesssim \frac{\|\hat{u}(\Xi)\|_1}{\operatorname{md}(\Xi) |\xi_m|^\alpha}. \quad (2.27)$$

The decay assumption appears naturally in the continuous setting, when considering signals  $u$  from Sobolev spaces  $H^k$  with  $k$  derivatives in  $L^2$ .

## 2.5 Escaping the minimizers

In this section we propose some solutions to mitigate the issues raised in Section 2.4 and we illustrate them numerically.

### 2.5.1 The effect of using a large dataset

In Theorem 1, we proved existence of many local minimizers in the case  $P = 1$ , which corresponds to a unique signal. Let us now assume that we have access to  $P$  signals  $u_1, \dots, u_P$  in  $\mathbb{C}^N$ . The analysis carried out to prove Theorem 1 can be replicated verbatim. The only difference being that every occurrence of  $|\hat{u}|^2$  must be replaced by  $\rho_P \stackrel{\text{def}}{=} \frac{1}{P} \sum_{p=1}^P |\hat{u}_p|^2$ . The function  $\rho_P$  can be understood as the average power spectral density of the family  $u_1, \dots, u_P$ . As highlighted in Theorem 1, two important factors that can create spurious minimizers are i) the number  $K$  of strict maximizers of  $\rho_P$  and ii) the curvature  $c$  at these maximizers.

As  $P$  increases, we typically expect the density  $\rho_P$  to become smoother. This effect is illustrated for a simple family of shifted and dilated rectangular functions in Fig. 2.4. As can be seen, both the number of maxima and the curvature  $c$  of  $\rho_P$  in Theorem 1 decay with  $P$ . For  $N = 128$ , we display the average power spectral density for  $P$  ranging from 1 to  $10^3$ . Each signal is defined by

$$u[n] = \int_{n-\frac{1}{2}}^{n+\frac{1}{2}} \mathbb{1}_{[a,b]}(x) \, dx, \quad (2.28)$$

where  $a$  and  $b$  are drawn uniformly in the range  $[-N/2 + 1, N/2 - 1]$ . The discrete signals are then renormalized so that  $\|u\|_2 = 1$ .



The same experiment can be reproduced in a more relevant framework from a practical viewpoint. The average power spectral density for 2D knee images of the fastMRI database [Zbontar 2018] are represented in Fig. 2.5. The image are of size  $320 \times 320$ . The local maximizers are computed and displayed with red dots in Fig. 2.5. In that case, increasing the family size  $P$  reduces the number of maximizers at a slow rate. Indeed they slightly increase from 13k points in the case  $P = 1$  to 14k in the case  $P = 100$  and then start to decrease to 11k for  $P = 10000$ . However, the curvature  $c$  decays much faster. As a conclusion, we see that *using large families of signals can reduce asymptotically the number and the size of the basins of attraction of some spurious minimizers.*

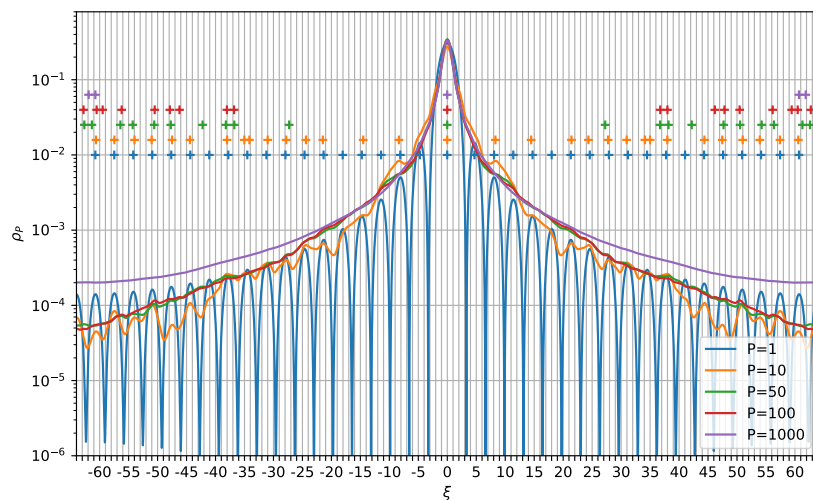
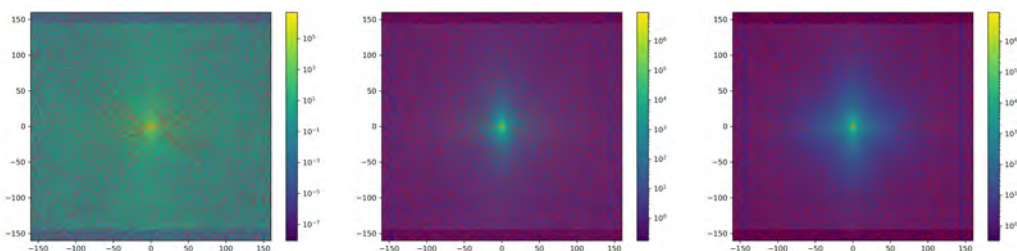


Figure 2.4: Average power spectral density  $\rho_P$  for families of rectangular functions with different sizes  $P$ . The dots represent local maxima of  $\rho_P$  for different values of  $P$ .



(a)  $P = 10^0$  – 13746 maxima (b)  $P = 10^2$  – 14888 maxima (c)  $P = 10^4$  – 11592 maxima

Figure 2.5: Average power spectral density  $\rho_P$  for a subset of images from the knee dataset of fastMRI. The image size is  $N = 320$  and the red dots represent local maximizers.

### 2.5.2 Stochastic gradient descent

When using a large family of signals, the cost function (2.3) naturally lends itself to the use of stochastic gradient descents (SGD), see [Wang 2022a, Weiss 2021] that address large MRI datasets. Contrarily to a deterministic gradient descent, which is known to converge to critical points under mild regularity conditions, the stochastic gradient with a fixed step size does not converge. The method is known to end up frolicking in the neighborhood of local critical points [Bottou 2010]. The radius of the neighborhood depends on the stochastic gradient variance and on the step-size. Intuitively, *using stochastic gradients algorithms should therefore allow escaping local minimizers*. We will showcase this effect in the forthcoming numerical experiments.

### 2.5.3 Variable metric

In Section 2.4.3, Theorem 2 states that the gradient of  $J_1$  might vanish in the high frequency domain. Using second order information is a well known remedy to mitigate this effect. In this work, we propose a simple method which corresponds to a variable diagonal metric with well-chosen coefficients.

As shown in Theorem 2, the gradient vanishes with a rate depending on the Fourier transform magnitude  $|\hat{u}|$ . For a dataset, this decay is somewhat captured by the average power spectral density  $\rho_P(\xi) \stackrel{\text{def}}{=} \frac{1}{P} \sum_{p=1}^P |\hat{u}_p(\xi)|^2$ . Hence, we propose to compute  $\rho_P$  once and for all on a fine grid ( $20 \times N$  discretization points in our example). The function  $\rho_P$  is then linearly interpolated in between the grid points during the gradient descent. At each gradient iteration we replace  $\frac{\partial J_1(\Xi)}{\partial \xi_m}$  by

$$\frac{1}{\rho_P(\xi_m)^\beta} \frac{\partial J_1(\Xi)}{\partial \xi_m}, \quad (2.29)$$

where  $\beta$  is a constant that has to be set empirically. From numerical experiments  $\beta \in [1, 2]$  shows good performance. In all the experiments presented hereafter we use  $\beta = 1$ . We will see later in the numerical experiments, that *this variable metric significantly accelerates the convergence for sampling points located in high frequencies*.

### 2.5.4 Numerical illustrations

In this section, we aim at illustrating numerically the different results established previously. We aim at reconstructing 1D signals of size  $N = 128$  from  $M = 64$  measurements in the Fourier domain. We suppose that  $P$  rectangular signals generated using (2.28) are given. We illustrate our findings with the back-projection reconstructor associated to the cost function  $J_1$ , but similar results have been obtained with the pseudo-inverse. As we are working in 1D with small dimensions  $N$  and  $M$ , at each iteration, the whole matrix  $A(\Xi)^*$  is evaluated and the gradient  $\nabla J_1$  is computed directly from the analytic expression (2.25). We first use a fixed step

gradient descent algorithm in order to showcase the convergence dynamics of the algorithm. The initialization of  $\Xi$  is a subgrid with a constant spacing of 2. The following experiments are conducted:

**Effect of the dataset size  $P$**  We first vary the number of signals by taking  $P = 1$  and  $P = 1000$ . The evolution of  $\Xi$  is displayed in Fig. 2.6, respectively top-left and top-center. The history of the cost function is given in Fig. 2.7. For this experiment, we expect that a good sampling scheme consists of low frequencies sampled at the Shannon-Nyquist rate. In this regard, the sampling scheme obtained in Fig. 2.6 for  $P = 1000$  is more satisfactory than the one obtained for  $P = 1$ . In the case  $P = 1000$ , the displacement of  $\Xi$  is more important, suggesting that some local minima have been discarded.

**Variable metric** We then study, for  $P = 1000$  the effect of a variable metric gradient descent as described in Section 2.5.3. We also compare this approach to an L-BFGS algorithm [Goldfarb 1970] with a line search and with a Hessian estimated using the last 8 gradients. In Fig. 2.6, the usual gradient algorithm is at the top-center, the variable metric gradient descent is at the bottom-center and the L-BFGS algorithm is at the bottom-left. The cost function evolution is displayed in Fig. 2.7. Using a variable metric results in a huge speed-up of the algorithm. This is particularly visible for points  $\xi$  located at high frequencies, which is another illustration of Theorem 2. For this example, the L-BFGS algorithm converges slightly faster than the variable metric gradient descent in the early iterations. However, its per-iteration cost is much higher since it uses a line search and a non diagonal metric. Since the L-BFGS algorithm can be seen as a state-of-the-art quasi-Newton method, the proposed empirical metric (2.29) seems remarkably efficient.

**Stochastic gradient descent** Finally in Fig. 2.6 right column, we investigate the use of a fixed-step stochastic gradient descent algorithm with a batch size of 1. In that experiment, a new random signal is generated at every iteration using the model (2.28) and the stochastic gradient is computed with respect to that signal only. The trajectory of the vanilla SGD is comparable with the one obtained using a deterministic gradient descent for  $P = 1000$  in Fig. 2.6 top-center. The variable metric trick significantly improves the convergence speed and more importantly, the final points configuration. As a conclusion, the variable metric SGD algorithm seems to be able to escape spurious minimizers and to take advantage of the averaging effect of the large dataset without the struggle of computing the gradient over a large dataset.

**Comparison of the sampling schemes** The final sampling schemes are not directly comparable in terms of cost function because the objective function is computed over different datasets. In Table 2.1, we therefore report the cost function computed on a specific set of signals. This set contains the  $P = 1000$  signals that are

used in the numerical illustrations of Fig. 2.6 center column. When tested against a large dataset, the final configuration obtained for  $P = 1$  seems highly sub-optimal. This effect is most likely due to a convergence to a local minimizer and also to the fact that the sampling scheme is not adapted to a whole family but only to a single signal. The remarkable observation that can be made from Table 2.1 is that the optimal configuration obtained with the variable metric SGD performs better on the dataset of  $P = 1000$  signals than the experiment conducted in Fig. 2.6 which is tailored for this dataset. This shows that the usual deterministic algorithms are stuck in local minima even with large datasets. On the contrary, the variable metric SGD algorithm seems effective.

These numerical results highlight the effectiveness of the different tricks suggested in this section: the use of a variable metric to handle high frequencies and a stochastic optimization to avoid local minima.

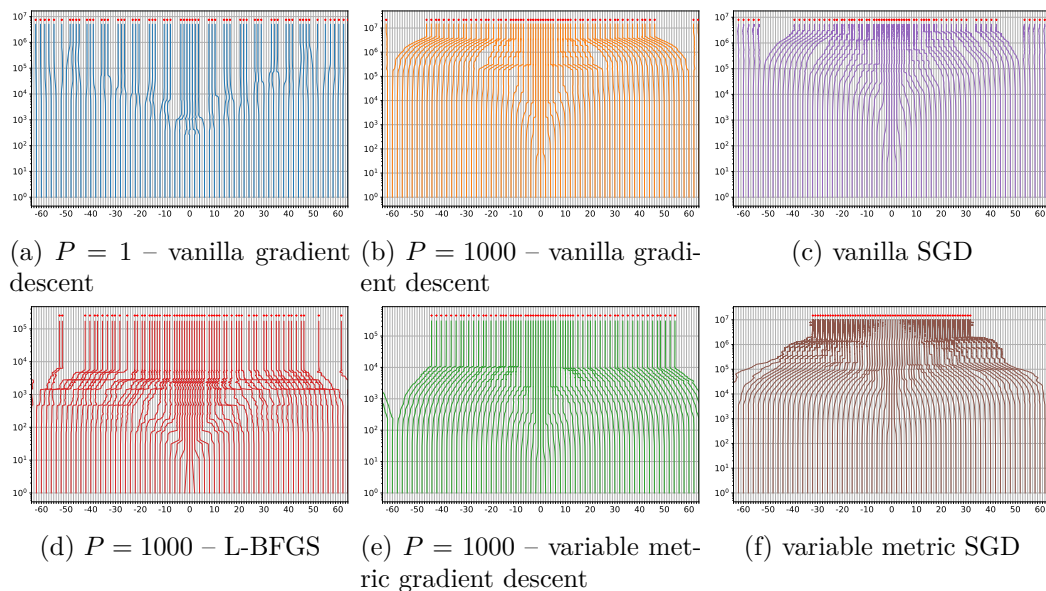


Figure 2.6: Trajectories of  $\Xi$  the back-projection reconstructor  $J_1$  and a fixed-step gradient descent. The iterations are represented on the vertical axis, and the horizontal axis corresponds to  $\xi$  and is periodic. The initialization is a uniform subgrid and is seen on the axis  $y = 0$  of the top and middle figures. Left and center: trajectories of  $\Xi$  for different sizes of signals families. The objective function is given in Fig. 2.7. The right column represents trajectories of  $\Xi$  using a stochastic gradient descent with one signal in the batch that is different at each iteration. The trajectories in the stochastic case have been averaged over the last 10000 iterations.

## 2.6 Conclusion

We highlighted two obstacles to the convergence of gradient based algorithms for Fourier sampling schemes optimization. The first one is a high number of local

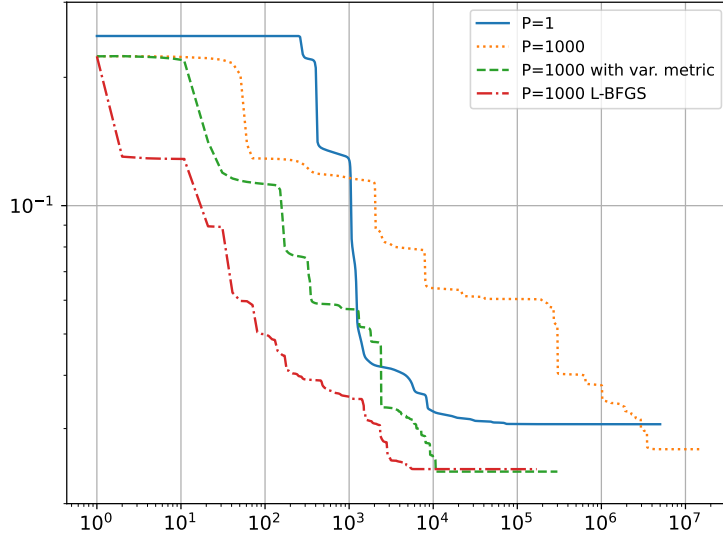


Figure 2.7: Objective function  $J_1$  (back-projection) for the different experiments in Fig. 2.6 in the deterministic case.

Test case	$P = 1$	$P = 1000$	$P = 1000$ with var. metric	L-BFGS	SGD	SGD with var. metric
Eff.	$9.07 \times 10^{-2}$	$2.68 \times 10^{-2}$	$2.38 \times 10^{-2}$	$2.41 \times 10^{-2}$	$6.63 \times 10^{-2}$	$1.00 \times 10^{-2}$

Table 2.1: Effectiveness of the sampling schemes obtained with different strategies on a dataset of 1000 signals. The table contains the average reconstruction error  $J_1$  over the dataset. This dataset is the one used in the case  $P = 1000$ , see Fig. 2.6 center column.

minimizers and the second one is a vanishing gradient phenomenon for high frequencies. As far as we know, this is the first theoretical study explaining why optimizing sampling patterns with modern automatic differentiation tools might result in algorithms being stucked at unsatisfactory locations. We also proposed three approaches to mitigate these effects. First, the number of spurious minimizers, the width and the depth of their basins of attraction can be reduced by considering large databases of signals. This acts as a regularization by averaging. Second, the vanishing gradient effect can be attacked with variable metric gradient descents. Finally, the use of a stochastic gradient instead of a deterministic gradient approach seems to allow escaping the narrow basins in a simplified 1D setting. These remarks may help explaining why the recent approaches in the literature based on the Adam optimizer manage to slightly improve the sampling pattern efficiency. Our work suggests that increasing the database sizes may help further easing the numerical resolution of the sampling pattern optimization by further smoothing the energy profiles. Many state-of-the-art reconstructors are based on a quadratic data fidelity term and we expect that some of the techniques used in this paper in the

linear case can be reused even in a nonlinear setting. This is left for future research.

## 2.7 Proofs

### 2.7.1 Proof of Proposition 11

Significant progress have been made lately in the control of the extreme eigenvalues of Vandermonde matrices, which play a pivotal role in algebraic number theory [Bombieri 1984, Moitra 2015, Batenkov 2020, Aubel 2019]. The tightest results for well separated schemes was recently obtained in [Aubel 2019]. Rewriting their result in our formalism, we obtain the following inequality.

**Proposition 13** (Conditioning of Vandermonde matrices [Aubel 2019]). *Let  $\Xi = (\xi_1, \dots, \xi_M)$  denote a set of distinct sampling points. The following inequalities hold*

$$\left(1 - \frac{1}{\text{md}(\Xi)}\right) \text{Id} \preceq A(\Xi)^* A(\Xi) \preceq \left(1 + \frac{1}{\text{md}(\Xi)}\right) \text{Id} \quad (2.30)$$

*Proof.* Relation (2.30) is a direct consequence of [Aubel 2019, eq. (31)] up to renormalizations.  $\square$

*Proof.* For  $i = 1$ , recall that  $Q_1 = \text{Id}$  and  $R_1^* R_1 - \text{Id} = A^* A$ . Then (2.14) is trivial and (2.15) follows from Proposition 13.

Let  $\tau_m$  denote the eigenvalues of  $A(\Xi)^* A(\Xi)$ . By Proposition 13,  $|\tau_m - 1| \leq \epsilon < 1$ .

For  $i = 2$ ,  $R_2^* R_2 = Q_2^* A^* A Q_2 = (A^* A)^\dagger$ ,  $Q_2 = (A^* A)^\dagger$ . With  $\epsilon < 1$ ,  $A(\Xi)^* A(\Xi)$  is invertible and we have

$$\frac{1}{1 + \epsilon} \text{Id} \preceq (A(\Xi)^* A(\Xi))^{-1} \preceq \frac{1}{1 - \epsilon} \text{Id}. \quad (2.31)$$

And we finally get

$$\frac{-\epsilon}{1 + \epsilon} \text{Id} \preceq (A(\Xi)^* A(\Xi))^{-1} - \text{Id} \preceq \frac{\epsilon}{1 - \epsilon} \text{Id}. \quad (2.32)$$

For  $i = 3$ ,  $Q_3 = (1 + \lambda)(A^* A + \lambda \text{Id})^{-1}$ , so that

$$\frac{1 + \lambda}{1 + \lambda + \epsilon} \text{Id} \preceq Q_3 \preceq \frac{1 + \lambda}{1 + \lambda - \epsilon} \text{Id}.$$

This gives

$$\frac{-\epsilon}{1 + \lambda + \epsilon} \text{Id} \preceq Q_3 - \text{Id} \preceq \frac{\epsilon}{1 + \lambda - \epsilon} \text{Id}$$

and using that  $\epsilon > 0$  and  $\lambda \geq 0$  allows concluding.

In order to prove (2.15), Proposition 13 yields  $1 - \epsilon \leq \tau_m \leq 1 + \epsilon$ . In addition,

$$R_3^* R_3 = (1 + \lambda)^2 (A^* A + \lambda \text{Id})^{-1} A^* A (A^* A + \lambda \text{Id})^{-1}$$

can be diagonalized and its eigenvalues are therefore of the form

$$(1 + \lambda)^2 \frac{\tau_m}{(\tau_m + \lambda)^2}.$$

By taking the upper-bound on the numerator and the lower-bound on the denominator, we obtain the following bound:

$$R_3^* R_3 \preceq (1 + \lambda)^2 \frac{1 + \epsilon}{(1 - \epsilon + \lambda)^2} \text{Id}.$$

We can continue as follows:

$$\begin{aligned} (1 + \lambda)^2 \frac{1 + \epsilon}{(1 - \epsilon + \lambda)^2} &= (1 + \lambda)^2 \frac{1 + \epsilon}{(1 + \lambda)^2 (1 - \epsilon / (1 + \lambda))^2} \\ &\leq \frac{1 + \epsilon}{(1 - \epsilon)^2}. \end{aligned}$$

By a similar reasoning with respect to the smallest eigenvalue of  $R_3^* R_3$ , we get:

$$\frac{1 - \epsilon}{(1 + \epsilon)^2} \text{Id} \preceq R_3^* R_3 \preceq \frac{1 + \epsilon}{(1 - \epsilon)^2} \text{Id}.$$

Subtracting the identity on both sides and using the fact that  $\epsilon^2 < \epsilon$  since  $\epsilon < 1$  yields

$$-\frac{4\epsilon}{(1 - \epsilon)^2} \text{Id} \preceq R_3^* R_3 - \text{Id} \preceq \frac{4\epsilon}{(1 - \epsilon)^2} \text{Id}.$$

□

## 2.7.2 Proof of Theorem 1

Under the hypotheses of Theorem 1, first notice that any set  $\Xi \in Z^M$  is a local maximizer of  $\Xi \mapsto \|\hat{u}(\Xi)\|_2^2$ . Indeed any perturbation of the individual sampling locations  $\xi_m$  results in a decay of the captured energy.

There are  $\binom{K}{M}$  possible sampling configurations when all the points belong to  $Z$ . Let  $\bar{\Xi} = \{\bar{\xi}_1, \dots, \bar{\xi}_M\}$  denote one of them. The idea of the proof is to show that there is a local minimizer of  $J$  in the following neighborhood  $B = [\bar{\xi}_1 - r, \bar{\xi}_1 + r] \times \dots \times [\bar{\xi}_M - r, \bar{\xi}_M + r]$ . A sufficient condition for the set  $B$  to contain a local minimizer of  $J$  is that  $J(\bar{\Xi}) < J(\Xi)$  for all  $\Xi \in \partial B$  (the boundary of  $B$ ) since  $J$  is continuous. Throughout this proof, we use  $\epsilon = \frac{1}{\delta - 2r}$  since we always have for all  $\Xi$  under consideration,  $\text{md}(\Xi) \geq \delta - 2r$ .

Using Proposition 9, the expression of (2.13) and the bounds of Proposition 11, we obtain

$$\left| J(\Xi) - \tilde{J}(\Xi) \right| \leq \left( \frac{b}{2} + a \right) \|\hat{u}(\Xi)\|_2^2 + \frac{b}{2} \sigma^2 M. \quad (2.33)$$

For all  $\Xi \in \partial B$ , at least one index  $m$  must verify  $\bar{\xi}_m - \xi_m = r$  and we have by

strict concavity of  $|\hat{u}|$  around  $\bar{\xi}_m$

$$\|\hat{u}(\bar{\Xi})\|_2^2 - \|\hat{u}(\Xi)\|_2^2 \geq \frac{cr^2}{2} \quad (2.34)$$

$$\|\hat{u}(\bar{\Xi})\|_2^2 + \|\hat{u}(\Xi)\|_2^2 \leq 2\|\hat{u}(\bar{\Xi})\|_2^2. \quad (2.35)$$

Hence for  $\Xi \in \partial B$ , using (2.34) yields

$$\tilde{J}(\Xi) - \tilde{J}(\bar{\Xi}) \geq \frac{cr^2}{2}. \quad (2.36)$$

Combining the previous inequalities yields

$$\begin{aligned} J(\Xi) - J(\bar{\Xi}) &= J(\Xi) - \tilde{J}(\Xi) + \tilde{J}(\Xi) - \tilde{J}(\bar{\Xi}) + \tilde{J}(\bar{\Xi}) - J(\bar{\Xi}) \\ &\stackrel{(2.36)}{\geq} \frac{cr^2}{2} + J(\Xi) - \tilde{J}(\Xi) + \tilde{J}(\bar{\Xi}) - J(\bar{\Xi}) \\ &\stackrel{(2.33)}{\geq} \frac{cr^2}{2} - \left[ \left( \frac{b}{2} + a \right) \left( \|\hat{u}(\Xi)\|_2^2 + \|\hat{u}(\bar{\Xi})\|_2^2 \right) + bM\sigma^2 \right] \\ &\stackrel{(2.35)}{\geq} \frac{cr^2}{2} - \left[ (b + 2a) \|\hat{u}(\bar{\Xi})\|_2^2 + bM\sigma^2 \right]. \end{aligned}$$

Therefore, the condition

$$\frac{cr^2}{2} > (b + 2a) \|\hat{u}(\bar{\Xi})\|_2^2 + bM\sigma^2 \quad (2.37)$$

suffices to conclude on the existence of a minimizer of  $J$  in the interior of  $B$ . The multiplicative factor  $M!$  is related to the fact that for a given minimizer, all the possible permutations of indices give rise to different minimizers.

### 2.7.3 Proof of Proposition 12

*Proof.* Let us consider a point configuration  $\Xi \in \mathbb{R}^M$  and a perturbation  $\epsilon \in \mathbb{R}^M$ .

Given a vector of measurements  $\hat{u}(\Xi) \in \mathbb{C}^M$ , we let  $\nabla \hat{u}(\Xi) = \begin{pmatrix} \hat{u}'(\xi_1) \\ \vdots \\ \hat{u}'(\xi_M) \end{pmatrix}$  denote the

vector of derivatives at the sampling locations. We recall that  $\hat{u}(\Xi) = A(\Xi)^*u$  and we define  $\hat{r}(\Xi) = A(\Xi)^*r$ . Elementary calculus leads to the following identities for every  $\epsilon$  direction of variation:

$$\begin{aligned} (\text{Jac}_A(\Xi)\epsilon)^* &= \text{Jac}_{A^*}(\Xi)\epsilon \\ \nabla \hat{u}(\Xi) \odot \epsilon &= \text{Jac}_{A^*}(\Xi)\epsilon \cdot u. \end{aligned}$$



Then, we apply standard calculus of variations:

$$\begin{aligned}
J_1(\Xi + \epsilon) &= J_1(\Xi) + \operatorname{Re}\langle \operatorname{Jac}_A(\Xi)\epsilon \cdot \hat{u}(\Xi), r(\Xi) \rangle \\
&\quad + \operatorname{Re}\langle A(\Xi)\operatorname{Jac}_{A^*}(\Xi)\epsilon \cdot u, r(\Xi) \rangle + o(\|\epsilon\|_2^2) \\
&= J_1(\Xi) + \operatorname{Re}\langle \hat{u}(\Xi), (\operatorname{Jac}_A(\Xi)\epsilon)^* r(\Xi) \rangle + \operatorname{Re}\langle \nabla \hat{u}(\Xi) \odot \epsilon, \hat{r}(\Xi) \rangle + o(\|\epsilon\|_2^2) \\
&= J_1(\Xi) + \operatorname{Re}\langle \hat{u}(\Xi), \nabla \hat{r}(\Xi) \odot \epsilon \rangle + \operatorname{Re}\langle \epsilon, \overline{\nabla \hat{u}(\Xi)} \odot \hat{r}(\Xi) \rangle + o(\|\epsilon\|_2^2) \\
&= J_1(\Xi) + \operatorname{Re}\langle \overline{\nabla \hat{r}(\Xi)} \odot \hat{u}(\Xi), \epsilon \rangle + \operatorname{Re}\langle \epsilon, \overline{\nabla \hat{u}(\Xi)} \odot \hat{r}(\Xi) \rangle + o(\|\epsilon\|_2^2).
\end{aligned}$$

Hence, by identification

$$\begin{aligned}
\nabla J_1(\Xi) &= \operatorname{Re}\left(\overline{\nabla \hat{r}(\Xi)} \odot \hat{u}(\Xi) + \nabla \hat{u}(\Xi) \odot \overline{\hat{r}(\Xi)}\right) \\
&= \operatorname{Re}\left(\nabla\left(\hat{u}(\Xi) \odot \overline{\hat{r}(\Xi)}\right)\right).
\end{aligned}$$

□

#### 2.7.4 Proof of Theorem 2

In order to simplify the notation, let  $L(\Xi) \stackrel{\text{def}}{=} A(\Xi)^*A(\Xi)$ . By Proposition 12, we have

$$\left| \frac{\partial J_1(\Xi)}{\partial \xi_m} \right| \leq |\hat{u}'(\xi_m)| \cdot |\hat{r}(\xi_m)| + |\hat{u}(\xi_m)| \cdot |\hat{r}'(\xi_m)|.$$

By definition, we have  $\hat{r}(\Xi) = (L(\Xi) - \operatorname{Id})\hat{u}(\Xi)$ , hence

$$|\hat{r}(\xi_m)| \leq \|\hat{r}(\Xi)\|_2 \leq \frac{\|\hat{u}(\Xi)\|_2}{\operatorname{md}(\Xi)}, \quad (2.38)$$

where we used Proposition 13 to obtain the last inequality. Now, we also wish to control  $|\hat{r}'(\xi_m)|$ . To this end, first notice that

$$\begin{aligned}
\hat{r}'(\xi_m) &= \sum_{m'=1}^M \left( \frac{\partial L(\Xi)_{m,m'}}{\partial \xi_m} \hat{u}(\xi_{m'}) \right. \\
&\quad \left. + L(\Xi)_{m,m'} \hat{u}'(\xi_{m'}) \mathbf{1}_{m=m'} \right) - \hat{u}'(\xi_m) \\
&= \sum_{m'=1}^M \frac{\partial L(\Xi)_{m,m'}}{\partial \xi_m} \hat{u}(\xi_{m'}).
\end{aligned}$$

We start with an analytical expression of the matrix  $L(\Xi)$ .

**Proposition 14** (The expression of  $A^*A$ ). *Let  $L(\Xi) \stackrel{\text{def}}{=} A(\Xi)^*A(\Xi)$ . We have*

$$[L(\Xi)]_{m,m'} = \begin{cases} 1 & \text{if } m = m', \\ \frac{1}{N} \exp\left(\frac{\iota\pi(\xi_m - \xi_{m'})}{N}\right) \times \frac{\sin(\pi(\xi_m - \xi_{m'}))}{\sin\left(\frac{\pi(\xi_m - \xi_{m'})}{N}\right)} & \text{otherwise.} \end{cases} \quad (2.39)$$

*Proof.* We have:

$$\begin{aligned}
[L(\Xi)]_{m,m'} &= \frac{1}{N} \sum_n e^{2\iota \frac{\pi}{N} \langle \xi_{m'} - \xi_m, n \rangle} \\
&= \frac{1}{N} e^{-\iota \pi (\xi_{m'} - \xi_m)} \frac{1 - e^{2\iota \pi (\xi_{m'} - \xi_m)}}{1 - e^{2\iota \frac{\pi}{N} (\xi_{m'} - \xi_m)}} \\
&= \frac{1}{N} e^{-\iota \pi (\xi_{m'} - \xi_m)} \frac{e^{\iota \pi (\xi_{m'} - \xi_m)}}{e^{\iota \frac{\pi}{N} (\xi_{m'} - \xi_m)}} \times \frac{e^{-\iota \pi (\xi_{m'} - \xi_m)} - e^{\iota \pi (\xi_{m'} - \xi_m)}}{e^{-\iota \frac{\pi}{N} (\xi_{m'} - \xi_m)} - e^{\iota \frac{\pi}{N} (\xi_{m'} - \xi_m)}} \\
&= \frac{1}{N} e^{-\iota \frac{\pi}{N} (\xi_{m'} - \xi_m)} \frac{\sin(\pi (\xi_{m'} - \xi_m))}{\sin(\frac{\pi}{N} (\xi_{m'} - \xi_m))}.
\end{aligned}$$

□

Now, we will use the following lemma.

**Lemma 1.** *The following bound holds:*

$$\left| \frac{\partial L(\Xi)_{m,m'}}{\partial \xi_m} \right| \leq \frac{\pi}{N} + \frac{4}{\text{dist}(\xi_{m'}, \xi_m)} \leq \frac{\pi}{N} + \frac{4}{\text{md}(\Xi)}.$$

*Proof.* Letting  $\delta = \xi_m - \xi_{m'}$ , we have

$$\frac{\partial L(\Xi)_{m,m'}}{\partial \xi_m} = \frac{\pi}{N^2} \times \frac{\iota e^{\iota \frac{\pi}{N} \delta} \sin(\pi \delta)}{\sin(\frac{\pi}{N} \delta)} + \frac{\pi}{N} \times \frac{e^{-\iota \frac{\pi}{N} \delta}}{\sin(\frac{\pi}{N} \delta)} \left( \cos(\pi \delta) - \frac{\sin(\pi \delta)}{N} \times \frac{\cos(\frac{\pi}{N} \delta)}{\sin(\frac{\pi}{N} \delta)} \right).$$

Without loss of generality we consider the case  $0 \leq \delta \leq N/2$ . Using  $\left| \frac{\sin(\pi \delta)}{N \sin(\frac{\pi}{N} \delta)} \right| \leq 1$  let us remark that

$$\left| \frac{\partial L(\Xi)_{m,m'}}{\partial \xi_m} \right| \leq \frac{\pi}{N} + \frac{\pi}{N} \left| \frac{1}{\sin(\frac{\pi}{N} \delta)} \left( \frac{\sin(\pi \delta) \cos(\frac{\pi}{N} \delta)}{N \sin(\frac{\pi}{N} \delta)} - \cos(\pi \delta) \right) \right|.$$

Using the inequality  $\left| \frac{\sin(\pi \delta)}{N \sin(\frac{\pi}{N} \delta)} \right| \leq 1$  again, we obtain

$$\left| \frac{\sin(\pi \delta) \cos(\frac{\pi}{N} \delta)}{N \sin(\frac{\pi}{N} \delta)} - \cos(\pi \delta) \right| \leq \left| \cos\left(\frac{\pi}{N} \delta\right) \right| + 1 \leq 2.$$

Finally, using the inequality  $\sin(x) \geq x/2$  for  $x \in (0, \pi/2)$ , we get  $\left| \frac{\partial L(\Xi)_{m',m}}{\partial \xi_{m'}} \right| \leq \frac{\pi}{N} + \frac{4}{\delta}$ . □

Lemma 1 and a Cauchy-Schwarz inequality provides the following bound:

$$|\hat{r}'(\xi_m)| \leq \left( \frac{\pi}{N} + \frac{4}{\text{md}(\Xi)} \right) \|\hat{u}(\Xi)\|_1.$$

Combining everything finally yields:

$$\left| \frac{\partial J_1(\Xi)}{\partial \xi_m} \right| \leq |\hat{u}'(\xi_m)| \cdot \frac{\|\hat{u}(\Xi)\|_2}{\text{md}(\Xi)} + |\hat{u}(\xi_m)| \cdot \|\hat{u}(\Xi)\|_1 \cdot \left( \frac{\pi}{N} + \frac{4}{\text{md}(\Xi)} \right).$$

Under the decay assumptions of Theorem 2, we obtain

$$\left| \frac{\partial J_1(\Xi)}{\partial \xi_m} \right| \lesssim \frac{\|\hat{u}(\Xi)\|_1}{\text{md}(\Xi)|\xi_m|^\alpha}.$$

# Bayesian Optimization of Sampling Densities in MRI

---

**Résumé** L’optimisation des schémas d’échantillonnage en IRM en utilisant une métrique guidée par les données a récemment fait l’objet d’une attention particulière. Suite aux observations du Chapitre 2 sur le nombre combinatoire de minimiseurs dans l’optimisation hors grille de schémas de Fourier, nous proposons un cadre pour optimiser globalement les densités d’échantillonnage en utilisant l’Optimisation Bayésienne. En utilisant une technique de réduction de dimension, nous optimisons les trajectoires d’échantillonnage plus de 20 fois plus rapidement que les méthodes conventionnelles hors grille, avec un nombre restreint d’images d’entraînement. Cette méthode – entre autres avantages – permet de s’affranchir de la nécessité d’utiliser de la différentiation automatique. Ses performances sont légèrement inférieures à celles des trajectoires apprises de l’état de l’art, car elle réduit l’espace des trajectoires admissibles, mais elle présente des avantages considérables en terme de temps de calcul. Les autres contributions comprennent : i) une évaluation précise de l’influence de la distance dans la formulation variationnelle pour générer des trajectoires ; ii) une méthode d’apprentissage spécifique sur des familles d’opérateurs pour les réseaux *unrolled* ; et iii) une méthode de gradient projeté pour l’optimisation des trajectoires.

**Abstract** Data-driven optimization of sampling patterns in MRI has recently received a significant attention. Following the observations of Chapter 2 on the combinatorial number of minimizers in off-the-grid optimization, we propose a framework to globally optimize the sampling densities using Bayesian optimization. Using a dimension reduction technique, we optimize the sampling trajectories more than 20 times faster than conventional off-the-grid methods, with a restricted number of training samples. This method – among other benefits – discards the need of automatic differentiation. Its performance is slightly worse than state-of-the-art learned trajectories since it reduces the space of admissible trajectories, but comes with significant computational advantages. Other contributions include: i) a careful evaluation of the distance in probability space to generate trajectories ii) a specific training procedure on families of operators for unrolled reconstruction networks and iii) a gradient projection based scheme for trajectory optimization.

This chapter is based on the preprint [Gossard 2022a]:

**Gossard, A.**, de Gournay, F., & Weiss, P. (2022). Bayesian Optimization of Sampling Densities in MRI.

---

**Contents**

<b>3.1</b>	<b>Introduction</b>	<b>84</b>
3.1.1	Some sampling theory	85
3.1.2	Data-driven sampling schemes	86
3.1.3	Our contribution	88
<b>3.2</b>	<b>The proposed approach</b>	<b>88</b>
3.2.1	Preliminaries	88
3.2.2	The challenges of sampling scheme optimization	90
3.2.3	Regularization and dimensionality reduction	91
3.2.4	The optimization routine	96
<b>3.3</b>	<b>Numerical experiments and results</b>	<b>99</b>
3.3.1	The experimental setting	99
3.3.2	Choosing a kernel for the discrepancy	101
3.3.3	Bayesian optimization: database size and numerical complexity	102
3.3.4	Comparing optimization routines for the total variation reconstructor	102
3.3.5	Comparing optimization routines for a neural network reconstructor	106
<b>3.4</b>	<b>Conclusion</b>	<b>108</b>
<b>3.5</b>	<b>Implementation details</b>	<b>110</b>
3.5.1	TV reconstruction algorithm	110
3.5.2	The unrolled neural network	110
3.5.3	Training the reconstruction network for a family of operators	111
3.5.4	Joint optimization	111
3.5.5	Computational details	112
<b>3.6</b>	<b>Sampling density and Shannon’s condition</b>	<b>114</b>
3.6.1	How to control the sampling density?	114
3.6.2	Performance variance for a fixed sampling density	115
<b>3.7</b>	<b>Resampling from estimated density</b>	<b>115</b>
3.7.1	Density estimation	115
3.7.2	Resampling	117

---

### 3.1 Introduction

The quest for efficient acquisition and reconstruction mechanisms in Magnetic Resonance Imaging (MRI) has been ongoing since its invention in the 1970’s. This led to a few major breakthrough, which comprise the design of efficient pulse sequences [Bernstein 2004], the use of parallel imaging [Roemer 1990, Blaimer 2004],

the theory and application of compressed sensing [Lustig 2008] and its recent improvements thanks to the progresses in learning and GPU computing [Knoll 2020]. While the first attempts to use neural networks in this field were primarily focused on the efficient design of reconstruction algorithms [Jacob 2020], some recent works began investigating the design of efficient sampling schemes or joint sampling/reconstruction schemes. The aim of this paper is to make progress in the numerical analysis of this nascent and challenging field.

### 3.1.1 Some sampling theory

In a simplified way, an MRI scanner measures values of the Fourier transform of the image to reconstruct at different locations  $(\xi_m)_{1 \leq m \leq M}$  in the so-called k-space. The locations  $(\xi_m)$  are obtained by sampling a continuous trajectory defined through a gradient sequence. The problem we tackle in this paper is: how to choose the points  $(\xi_m)$  or the underlying trajectories in an efficient or optimal way?

**Shannon-Nyquist** This question was first addressed using Shannon-Nyquist theorem, which certifies that sampling the k-space on a sufficiently fine Euclidean grid provides exact reconstructions using linear reconstructors. This motivated the design of many trajectories, such as the ones in echo-planar imaging (EPI) [Schmitt 2012]. Progresses on non-uniform sampling theory [Feichtinger 1994] then provided guidelines to produce efficient sampling/reconstruction schemes for linear reconstructors. This theory is now mature for the reconstruction of bandlimited functions. In a nutshell, it advocates the use of a sampling set which covers the k-space sufficiently densely with well spread samples.

**Compressed sensing theory** Shannon-Nyquist theory requires sampling the k-space densely, resulting in long scanning times. It was observed in the 1980's that subsampling the high frequencies using variable density radial patterns did not compromise the image quality too much [Ahn 1986, Jackson 1992]. The first theoretical elements justifying this evidence were provided by the theory of compressed sensing, when using nonlinear reconstructors. This seminal theory is based on concepts such as the restricted isometry property (RIP) or the incoherence between the measurements [Candès 2006, Lustig 2005]. However it soon became evident that these concepts were not suited to the practice of MRI and a refined theory based on local coherence appeared in [Adcock 2017, Boyer 2019]. The main teaching is that a good sampling scheme for  $\ell^1$ -based reconstruction methods must have a variable density that depends on the sparsity basis and on the sparsity pattern of the images. To the best of our knowledge, this theory is currently the one that provides the best explanation of the success of sub-sampling. In particular, analytical expressions of the optimal densities [Adcock 2020] can be derived and fit relatively well with the best empirical ones.

**The main teachings** To date, there is still a significant discrepancy between the theory and practice of sampling in MRI. A mix between theory and common sense however provides the following main insights. A good sampling scheme should [Boyer 2016]:

- have a variable density, decaying with the distance to the center of the k-space,
- have a sufficiently high density in the center to comply with the Shannon-Nyquist criterion, and sufficiently low to avoid dense clusters which would not bring additional information,
- have a locally uniform coverage of the k-space. In particular, nearby samples are detrimental to the reconstruction since they are highly correlated and increase the condition number of partial Fourier matrices.

These considerations are all satisfied when using Poisson disk sampling with an adequate density [Vasanawala 2011] for pointwise sampling. They also led to the development of the SPARKLING trajectories [Chauffert 2017, Lazarus 2019], which incorporate additional trajectory constraints in the design.

**What can still be optimized?** Given the previous remarks, an important question remains open: how to choose the sampling density? An axiomatic approach leads to choosing radial densities with a plateau (constant value) at the center. The radial character ensures rotation invariance, which seems natural to image organs in arbitrary orientations. The plateau enforces Nyquist rate at the center. However, it may still be possible to improve the results for specific datasets.

### 3.1.2 Data-driven sampling schemes

The first attempts to learn a sampling density [Knoll 2011, Zhang 2014] were based on the average energy of the k-space coefficients on a collection of reference images. While this principle is valid for linear reconstructions, it is not supported by a theoretical background when using nonlinear reconstructors. Motivated by the recent breakthroughs of learning and deep learning, many authors recently proposed to learn either the reconstructor [Hammernik 2018], the sampling pattern [Baldassarre 2016, Gözcü 2018, Zibetti 2021, Sherry 2020], or both [Jin 2019, Bahadir 2020, Weiss 2021, Aggarwal 2020, Wang 2022a]. Data-driven optimization has emerged as a promising approach to tailor the sampling schemes with respect to the reconstructor and to the image structure. In [Baldassarre 2016, Gözcü 2018, Sherry 2020, Sanchez 2020, Zibetti 2021], the authors look for an optimal subset of a fixed set of k-space positions. The initial algorithms are based on simple greedy approaches that generated a sampling pattern by iteratively selecting a discrete horizontal line that minimizes the residual error of the reconstructed image. This approach is limited to low dimensional sets of parallel lines. Some efforts have been spent on finding better and more scalable solutions to this hard combinatorial problem using stochastic greedy algorithms [Sanchez 2020],

$\ell^1$ -relaxation and bi-level programming [Sherry 2020] or bias-accelerated subset selection [Zibetti 2021]. This method is reported to provide results over 3D images and seems to have solved some of the scalability issues.

To the best of our knowledge, the first work investigating the joint optimization of a sampling pattern and a reconstruction algorithm was proposed in [Jin 2019]. In this work, the authors use a Monte Carlo Tree Search which allows them to optimize a policy that determines the positions to sample. This sampling relies on lines and the reconstruction process is an image to image domain with an inverse Fourier transform performed on the data before the denoising step. In the same spirit, [Bahadir 2020] proposes to learn MRI trajectories by optimizing a binary mask over a Cartesian grid with some sparsity constraint. The reconstruction is decomposed into two steps: a regridding using an inverse Fourier transform and a U-NET for de-aliasing. Finally, a new class of reconstruction methods called *algorithm unrolling*, mimicking classical variational approaches have emerged. These approaches improve the interpretability of deep learning based methods. Optimizing the weights of a CNN that plays the role of a denoiser in a conjugate gradient descent has been investigated in [Aggarwal 2020]. The authors jointly optimize the sampling pattern and a denoising network based on an unrolled conjugate gradient scheme. The sampling scheme is expressed as the tensor product of 1D sampling patterns which significantly restricts the possible sampling schemes.

Overall, the previous works suffer from some limitations: the sampling points are required to live on a Cartesian grid, which may be non physical and lead to combinatorial problems; the methods cannot incorporate advanced constraints on the sampling trajectory and therefore focus on “rigid” constraints such as selecting a subset of horizontal lines.

To address these issues, some recent works propose to optimize points that can move freely in a continuous domain [Weiss 2021, Wang 2022a]. This approach allows handling real kinematic constraints. In [Weiss 2021], the authors propose to reconstruct an image using a rough inversion of the partial Fourier transform, followed by a U-NET to eliminate the residual artifacts. They optimize jointly the weights of the U-NET together with the k-space positions using a stochastic gradient method. The physical kinematic constraints are handled using two different ingredients. First, the k-space points are regularly ordered by solving a traveling salesman problem, ensuring a low distance between consecutive points. Second, the constraints are promoted using a penalization function. This re-ordering step was then abandoned in [Wang 2022a], where the authors use a B-spline parameterization of the trajectories with a penalization over the constraints in the cost function. Instead of using a rough inversion with a U-NET, the authors opted for an unrolled ADMM reconstructor where the proximal operator is replaced by a DIDN CNN [Yu 2019]. The k-space locations and the CNN weights are optimized jointly. In both works, long computation times and memory requirements are reported. We also observed significant convergence issues related to the existence of spurious minimizers [Gossard 2022b].



### 3.1.3 Our contribution

The purpose of this work is to improve the process of optimizing sampling schemes from a methodological perspective. We propose a framework that optimizes the sampling density using Bayesian Optimization (BO). Our method has a few advantages compared to recent learning based approaches: i) it globalizes the convergence by reducing the dimensionality of the optimization problem, ii) it reduces the computing times drastically, iii) it requires only a small number of reference images and iv) it works off-the-grid and handles arbitrary physical constraints. The first three features are essential to make sampling scheme optimization tractable in a wide range a different MRI scanners. The last one allows more versatility in the sampling patterns that can take advantage of all the degrees of freedom offered by an MRI scanner.

## 3.2 The proposed approach

In this section, we describe the main ideas of this work after having introduced the notation.

### 3.2.1 Preliminaries

**Images** Let  $\mathcal{X}$  denote the set of  $K$  training images  $\mathcal{X} = \{x_1, \dots, x_K\}$ . A  $D$ -dimensional image is a vector of  $\mathbb{C}^N$ , where  $N = N_1 \dots N_D$  and  $N_d \in 2\mathbb{N}$  denotes the number of pixels in the  $d$ -th direction. In this work, each index  $n \in \llbracket 1, N \rrbracket$ , is associated to a position  $p_n \in \llbracket -\frac{N_1}{2}, \frac{N_1}{2} - 1 \rrbracket \times \dots \times \llbracket -\frac{N_D}{2}, \frac{N_D}{2} - 1 \rrbracket$  on Euclidean grid. It describes the location of the  $n$ -th pixel in the k-space. With a slight abuse of notation, we associate to each discrete image  $x_k \in \mathbb{C}^N$ , a function still denoted  $x_k$ , defined by

$$x_k = \left( \sum_{n=1}^N x_k[n] \delta_{p_n} \right) \star \psi,$$

where  $\star$  denotes the convolution-product and where  $\psi$  is an interpolation function. For instance, we can set  $\psi$  as the indicator of a grid element to generate piece-wise constant images.

**Image quality** To measure the reconstruction quality, we consider an image quality metric  $\eta : \mathbb{R}^N \times \mathbb{R}^N \rightarrow \mathbb{R}_+$ . The experiments in this work are conducted using the squared  $\ell^2$  distance  $\eta(\tilde{x}, x) = \frac{1}{2} \|\tilde{x} - x\|_2^2$ . Any other metric could be used instead with the proposed approach.

**The Non-Uniform Fourier Transform** Throughout the paper, we let  $\xi = (\xi_1, \dots, \xi_M) \in (\mathbb{R}^D)^M$  denote a set of locations in the k-space (or Fourier domain). Let  $A(\xi) \in \mathbb{C}^{M \times N}$  denote the forward non-uniform Fourier transform defined for

all  $m \in \llbracket 1, M \rrbracket$  and  $x \in \mathbb{C}^N$  by

$$\begin{aligned} [A(\xi)(x)]_m &= \int_{t \in \mathbb{R}^D} \exp(-i\langle t, \xi_m \rangle) x(t) dt \\ &= \Psi(\xi_m) \cdot \sum_{n=1}^N x[n] \exp(-i\langle p_n, \xi_m \rangle), \end{aligned} \quad (3.1)$$

where  $\Psi$  is the Fourier transform of the interpolation function  $\psi$ .

**Image reconstruction** We let  $R : \mathbb{C}^M \times (\mathbb{R}^D)^M \times \mathbb{R}^J \rightarrow \mathbb{C}^N$  denote an image reconstruction mapping. For a measurement vector  $y \in \mathbb{C}^M$ , a sampling scheme  $\xi \in (\mathbb{R}^D)^M$ , and a parameter  $\lambda \in \mathbb{R}^J$ , we let  $\tilde{x} = R(\xi, y, \lambda)$  denote the reconstructed image. In this paper, we will consider two different reconstructors:

- A Total Variation (TV) reconstructor [Lustig 2008], which is a standard baseline:

$$R_1(\xi, y, \lambda) = \arg \min_{x \in \mathbb{C}^N} \frac{1}{2} \|A(\xi)x - y\|_2^2 + \lambda \|\nabla x\|_1, \quad (3.2)$$

where  $\lambda \geq 0$  is a regularization parameter. The approximate solution of this problem is obtained with an iterative algorithm run for a fixed number of iterations. We refer the reader to Section 3.5.1 for the algorithmic details. This allows us to use the automatic differentiation of PyTorch as described in [Ochs 2015].

- An unrolled neural network  $R_2(\xi, y, \lambda)$ , where  $\lambda$  denotes the weights of the neural network. There is now a multitude of such reconstructors available in the literature [Muckley 2020a]. They draw their inspiration from classical model-based reconstructors with hand-crafted priors. The details are provided in Section 3.5.2.

**Constraints on the sampling scheme** As mentioned in the introduction, the sampling positions  $\xi = (\xi_1, \dots, \xi_M)$  correspond to the discretization of a k-space trajectory subject to kinematic constraints. Throughout the paper, we let  $\Xi \subset (\mathbb{R}^D)^M$  denote the constraint set for  $\xi$ . A sampling set consists of  $N_s \in \mathbb{N}$  trajectories (shots) with  $P$  measurements per shot. We consider realistic kinematic constraints on these trajectories. Let  $\alpha$  denote the maximal speed of a discrete trajectory and  $\beta$  denote its maximal acceleration (the slew rate). We let

$$Q_P^{\alpha, \beta} = \left\{ \xi \in ([-\pi, \pi]^D)^P, \|\dot{\xi}\|_\infty \leq \alpha, \|\ddot{\xi}\|_\infty \leq \beta, C\xi = b \right\}, \quad (3.3)$$

where

$$\begin{aligned} \|\dot{\xi}\|_\infty &= \max_{1 \leq p \leq P-1} \|\xi_{p+1} - \xi_p\|_2 \\ \|\ddot{\xi}\|_\infty &= \max_{2 \leq p \leq P-1} \|\xi_{p+1} + \xi_{p-1} - 2\xi_p\|_2, \end{aligned}$$

where  $b$  is a vector and  $C$  a matrix encoding some position constraints. For instance, we enforce the first point of each trajectory to start at the origin. Since the sampling schemes consists of  $N_s$  trajectories, the constraint set on the sampling is  $\Xi = (Q_P^{\alpha,\beta})^{N_s}$ . The total number of measurements  $M$  equal to  $M = N_s \cdot P$ . We refer the reader to [Chauffert 2016] for more details on these constraints.

### 3.2.2 The challenges of sampling scheme optimization

In this paper, we consider the optimization of a sampling scheme for a fixed reconstruction mapping  $R$ . A good sampling scheme should reconstruct the images in the training set  $\mathcal{X}$  efficiently in average. Hence, a natural optimization criterion is

$$\min_{\xi \in \Xi} \mathbb{E} \left( \frac{1}{K} \sum_{k=1}^K \eta(R(\xi, A(\xi)x_k + n, \lambda), x_k) \right). \quad (3.4)$$

The term  $A(\xi)x_k$  corresponds to the measurements of the image  $x_k$  associated to the sampling scheme  $\xi$ . The expectation is taken with respect to the term  $n \in \mathbb{C}^N$  which models noise on the measurements. More elaborate forward models can be designed to account for sensibility matrices in multi-coil imaging or for trajectory errors. We will not consider these extensions in this paper. Their integration is straightforward – at least at an abstract level.

Even if problem (3.4) is simple to state (and very similar to [Weiss 2021, Wang 2022a]), the practical optimization looks extremely challenging for the following reasons:

- The computation of the cost function is very costly.
- Computing the derivative of the cost function using backward differentiation requires differentiating a Non-uniform Fast Fourier Transform (NFFT). It also requires a consequent quantity of memory that limits the complexity of the reconstruction mapping.
- The energetic landscape of the functional is usually full of spurious minimizers [Gossard 2022b].
- The minimization of an expectation calls for the use of stochastic gradient descent, but the additional presence of a constraint set  $\Xi$  reduces the number of solvers available.

Hence, the design of efficient computational solutions is a major issue. It will be the main focus of this paper. The following sections are dedicated to the simplification of (3.4) and to the design of a lightweight solver. We also propose a home-made solver that attacks (3.4) directly. Since similar ideas were proposed in [Wang 2022a], we describe the main ideas and differences in Section 3.5.5.1 only.

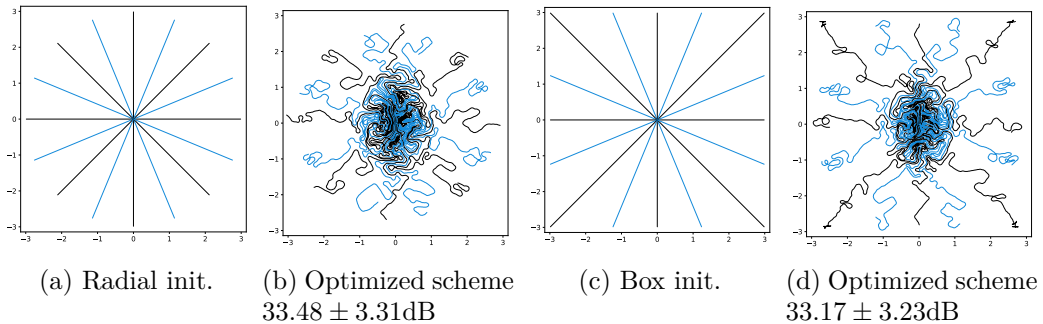


Figure 3.1: The globalization issue: optimizing a scheme with an advanced multi-scale approach yields different average PSNR when starting from different point configurations. In this experiment, we used a total variation reconstruction algorithm and 10% undersampling.

### 3.2.3 Regularization and dimensionality reduction

The non-convexity of (3.4) is a major hurdle inducing spurious minimizers [Gossard 2022b]. We discuss the existing solutions to mitigate this problem and give our solution of choice.

#### 3.2.3.1 Existing strategies and their limitation

In [Weiss 2021, Wang 2022a], the authors propose to avoid local minima by using a multi-scale optimization approach starting from a trajectory described through a small number of control points and progressively getting more complex through the iterations. The use of the stochastic Adam optimizer can also allow escaping from narrow basins of attraction. In addition, Adam optimizer can be seen as a preconditioning technique, which can accelerate the convergence, especially for the high frequencies [Gossard 2022b]. This optimizer together with a multi-scale approach can yield sampling schemes with improved reconstruction quality at the cost of a long training process. However, despite heuristic approaches to globalize the convergence, we experienced significant difficulties in getting reproducible results.

To illustrate this fact, we conducted a simple experiment in Fig. 3.1. Starting from two similar initial sampling trajectories, we let a multi-scale solver run for 14 epochs and 85 hours on the fastMRI knee database. We then evaluate the average reconstruction peak signal-to-noise ratio (PSNR) on the validation set. As can be seen, the final point configuration and the average performance varies significantly.

#### 3.2.3.2 Optimizing a sampling density

The key idea in this paper is to regularize the problem by optimizing a sampling density rather than the point positions directly. To formalize this principle we need to introduce two additional ingredients:

1. A *probability density generator*  $\rho : \mathbb{R}^L \rightarrow \mathcal{P}$ , where  $\mathcal{P}$  is the set of probability

distributions on  $\mathbb{R}^D$ . In this paper,  $\rho$  will be defined as a simple affine mapping, but we could also consider more advanced generators such as Generative Adversarial Networks.

2. A *trajectory sampler*  $\mathcal{S}_M : \mathcal{P} \rightarrow (\mathbb{R}^D)^M$ , which maps a density  $\rho$  to a point configuration  $\mathcal{S}_M(\rho) \in (\mathbb{R}^D)^M$ . Various possibilities could be considered such as Poisson point sampling, Poisson disk sampling. In this paper, we will use discrepancy based methods [Boyer 2016].

Instead of minimizing (3.4), we propose to work directly with the density. Letting  $\xi : \mathbb{R}^L \rightarrow (\mathbb{R}^D)^M$  denote the mapping defined by

$$\xi(z) \stackrel{\text{def}}{=} \mathcal{S}_M(\rho(z)), \quad (3.5)$$

we propose to minimize:

$$F(z) \stackrel{\text{def}}{=} \min_{z \in \mathcal{C} \subset \mathbb{R}^L} \frac{1}{K} \sum_{k=1}^K \mathbb{E}(\eta[R(\xi(z), A(\xi(z))x_k + n, \lambda), x_k]), \quad (3.6)$$

where the expectation is taken with respect to the noise term  $n$ . A schematic illustration of this approach is proposed in Fig. 3.2.

### 3.2.3.3 The density generator

Various approaches could be used to define a density generator  $\rho$ . In this work, we simply define  $\rho(z)$  as an affine mapping, i.e.

$$\rho(z) \stackrel{\text{def}}{=} \mu_0 + \sum_{l=1}^L z_l \mu_l, \quad (3.7)$$

where  $z$  belongs to a properly defined convex set  $\mathcal{C}$ . We describe hereafter how the eigen-elements  $(\mu_l)_l$  and the set  $\mathcal{C}$  are constructed.

**A candidate space of densities** The general idea of our construction is to define a family of elementary densities and to enrich it by taking convex combinations of its elements.

Let  $\theta \in [0, \pi[$  denote a rotation angle,  $\sigma_x, \sigma_y$  denote lengths,  $r > 0$  denote a density at the center and  $\gamma > 0$  a decay rate. For  $(x, y) \in \mathbb{R}^2$ , let  $x_\theta = x \cos(\theta) + y \sin(\theta)$ ,  $y_\theta = -\sin(\theta)x + \cos(\theta)y$ . We define

$$g(x, y; \sigma_x, \sigma_y, \theta, r, \gamma) = \frac{1}{c} \min \left( r, \frac{1}{\left( \left( \frac{x_\theta}{\sigma_x} \right)^2 + \left( \frac{y_\theta}{\sigma_y} \right)^2 + \epsilon \right)^\gamma} \right), \quad (3.8)$$

where  $c$  is a normalizing constant such that  $\int_{\mathbb{R}^2} g = 1$ . We then smooth the function

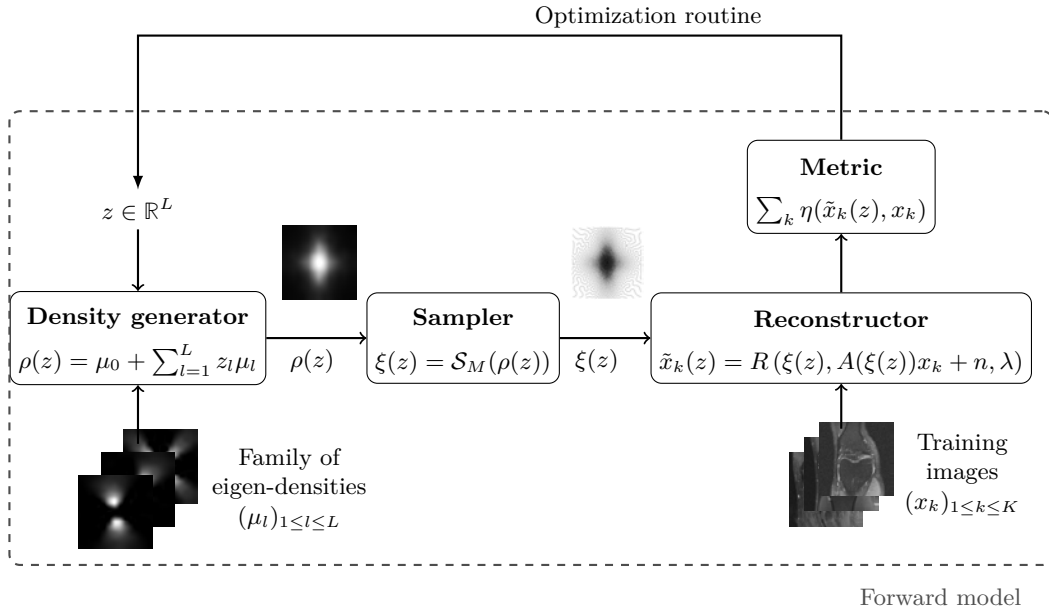


Figure 3.2: A schematic illustration of the proposed algorithm. We generate a sampling density  $\rho(z)$  through an affine combination of eigen-elements  $(\mu_l)$ . The density is then used in a sampling pattern generator  $\mathcal{S}_M$  which yields a sampling trajectory  $\xi(z)$ . A set of training images are then reconstructed using this scheme. This allows computation of the (batch) average error. A zero-th, or first order (automatic differentiation) optimization routine optimizes the sampling density iteratively.

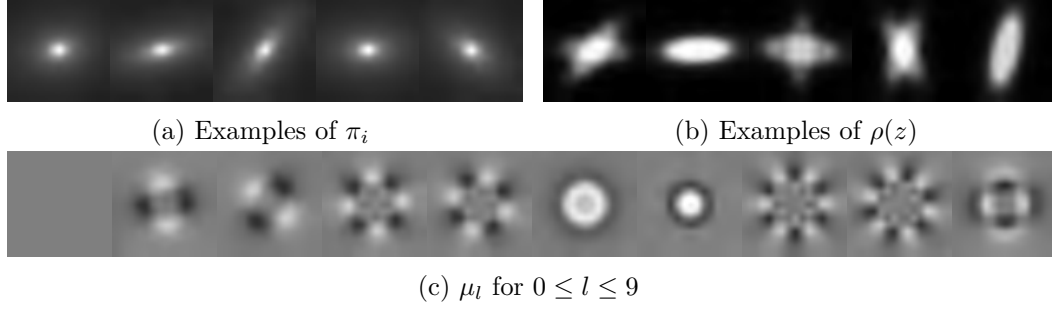


Figure 3.3: Examples of densities using the proposed parameterization.

$g$  by convolving it with a Gaussian function  $G_\kappa$  of standard deviation  $\kappa > 0$ :

$$\pi = G_\kappa \star g. \quad (3.9)$$

The elements in this family are good candidates for sampling densities: i) they are nearly constant and approximately equal to  $r$  at the center of the  $k$ -space, ii) they can be anisotropic to accommodate for specific image orientations and iii) they have various decay rates, allowing sampling the high frequencies more or less densely. Some examples of such densities are displayed in Fig. 3.3a. However, the family of densities generated by this procedure is quite poor. For instance, it is impossible to sample densely both the  $x$  and  $y$  axes simultaneously. In order to enrich it, we propose to consider the set of convex combinations of these elementary densities. This allows us to construct more general multi-modal densities, see Fig. 3.3b for examples of such convex combinations.

**Dimensionality reduction** In order to construct the family  $(\mu_0, \dots, \mu_L)$ , we first draw a large family of  $I \gg L$  densities  $(\pi_i)_{1 \leq i \leq I}$ . They are generated at random by uniform draws of the parameters  $(\sigma_x, \sigma_y, \theta, t, \gamma)$  inside a box. We then perform a principal component analysis (PCA) on this family to generate some eigen-elements  $(\nu_l)_{0 \leq l \leq L}$ . We set  $\mu_0 = \nu_0 / \langle \nu_0, \mathbb{1} \rangle$ . Since probability densities must sum to 1, we orthogonalize the family  $(\nu_l)$  with respect to the vector  $\mu_0$ . Thereby, we obtain a second family  $(\mu_l)_{0 \leq l \leq L}$  that satisfies  $\langle \mu_0, \mathbb{1} \rangle = 1$  and  $\langle \mu_l, \mathbb{1} \rangle = 0$  for all  $1 \leq l \leq L$ . This procedure discards one dimension. The resulting PCA basis is illustrated in Fig. 3.3c.

Let  $\mathcal{E}$  denote the intersection of the span of  $(\mu_l)_{l \leq L}$  with the probability densities and  $\Pi_{\mathcal{E}}$  the orthogonal projection on  $\mathcal{E}$ . The space of densities is the convex hull of the family  $(\pi_i)_i$  projected on  $\mathcal{E}$ :

$$\mathcal{C} \stackrel{\text{def}}{=} \text{Conv}(\Pi_{\mathcal{E}}(\pi_i), 1 \leq i \leq I). \quad (3.10)$$

As illustrated in Fig. 3.3b, this process overall provides a rather rich and natural family with a low dimensionality (here  $L = 20$ ).

### 3.2.3.4 The sampler

The sampler  $\mathcal{S}_M : \mathcal{P} \rightarrow (\mathbb{R}^D)^M$  is based on discrepancy minimization [Schmaltz 2010, Gräf 2012, Chauffert 2017]. It is defined as an approximate solution of

$$\mathcal{S}_M(\rho) = \arg \min_{\xi \in \Xi} \text{dist} \left( \frac{1}{n} \sum_{m=1}^M \delta_{\xi^{[m]}}, \rho \right), \quad (3.11)$$

where  $\Xi \subset (\mathbb{R}^D)^M$  takes into account the trajectory constraints and  $\text{dist}$  is a discrepancy defined by

$$\text{dist}(\mu, \nu) = \sqrt{\langle h \star (\mu - \nu), (\mu - \nu) \rangle_{L^2(\mathbb{R}^D)}},$$

where  $h$  is a positive definite kernel (i.e. a function with a real positive Fourier transform). Other metrics on the set of probability distributions could be used such as the transportation distance [Lebrat 2019]. The formulation (3.11) has already been proposed in [Chauffert 2017] and it is at the core of the Sparkling scheme generation [Lazarus 2019]. We will discuss the choice of the kernel  $h$  in the numerical experiments: it turns out to play a critical role.

In practice (3.11) is not solved exactly: an iterative solver [Chauffert 2016] is ran for a fixed number of iterations. This allows the use of automatic differentiation in order to compute the Jacobian of  $\xi$  w.r.t.  $z$ . Technical details about the implementation of this sampler are provided in Section 3.5.5.

### 3.2.3.5 The pros and cons of this strategy

The optimization problem (3.6) presents significant advantages compared to the original one (3.4):

- The number of optimization variables is considerably reduced: instead of working with  $D \cdot M$  variables, we now only work with  $L \ll D \cdot M$  variables defining a continuous density. In this paper we set  $L = 20$  which is considerably smaller in comparison to the  $M = 25801$  2D sampling points for the formulation of (3.4) with 25% undersampling on  $320 \times 320$  images. This allows resorting to global optimization routines. Hereafter, we will describe a Bayesian optimization approach.
- The point configurations generated by this algorithm are always locally uniform since they correspond to the minimizers of a discrepancy. Clusters are therefore naturally discarded, which can be seen as a natural regularization scheme.
- As discussed in the numerical experiments, the regularization effect allows optimizing the sampling density with a small dataset with a similar performance. Optimizing the function with as little as 32 reference images yields a near optimal density. This aspect might be critical for small databases.



On the negative side, notice that we considerably constrained the family of achievable trajectories, thereby reducing the maximal achievable gain. We will show later that the trajectories obtained by minimizing (3.6) are indeed slightly less efficient than those obtained with (3.4). This price might be affordable if we compare it to the advantages of having a significantly faster and more robust solver requiring only a fraction of the data needed for solving (3.4).

### 3.2.4 The optimization routine

In this section, we describe an algorithmic approach to attack the problem (3.6).

#### 3.2.4.1 The non informativeness of the gradient

A natural approach to solve (3.6) is to optimize the coefficients  $z \in \mathbb{R}^L$  using a gradient based algorithm. Unfortunately, the reparameterization of the cost function with a density still makes the energy profile full of spurious minimizers. The presence of these oscillations would trap a gradient based algorithm in local minimizers. Fig. 3.4 illustrates this fact. In the “Shift” row, we display the energy profile when shifting the sampling pattern on the top-left by 4 pixels in the horizontal and vertical direction. In the “Density” row, we display the energy profile with a family of  $L = 2$  eigen-elements  $(\mu_1, \mu_2)$ . The  $3 \times 3$  red dots on the energy profiles corresponds to the  $3 \times 3$  sampling densities on the top-right.

Overall, this experiment shows that the gradient direction is not meaningful: it oscillates in an erratic way. This advocates for the use of 0<sup>th</sup> order optimization methods. A significant advantage of this observation is that it allows discarding the memory and time issues related to automatic differentiation.

#### 3.2.4.2 Bayesian optimization

As can be seen from the energy profiles in Fig. 3.4, the cost function seems to be decomposable as a smooth function plus an oscillatory one of low amplitude. This calls for the use of algorithms that i) sample the function at a few scattered points, ii) construct a smooth surrogate approximation, iii) find a minimizer of the surrogate and add it to the explored samples, iv) go back to ii).

Bayesian Optimization (BO) [Frazier 2018b] is a principled approach that follows these steps. It seems particularly adequate since it models uncertainty on the function evaluations and comes with advanced solvers [Balandat 2020]. Its application is nonetheless nontrivial and requires some care in our setting. We describe some technical details hereafter.

Consider an objective function of the form

$$\inf_{z \in \mathcal{C}} \mathbb{E} (F(z, V)),$$

where  $V$  is a random vector. In our setting,  $V$  models both the noise  $n$  and the input images  $x_k$ . We consider  $V$  to be a random vector taken uniformly inside a

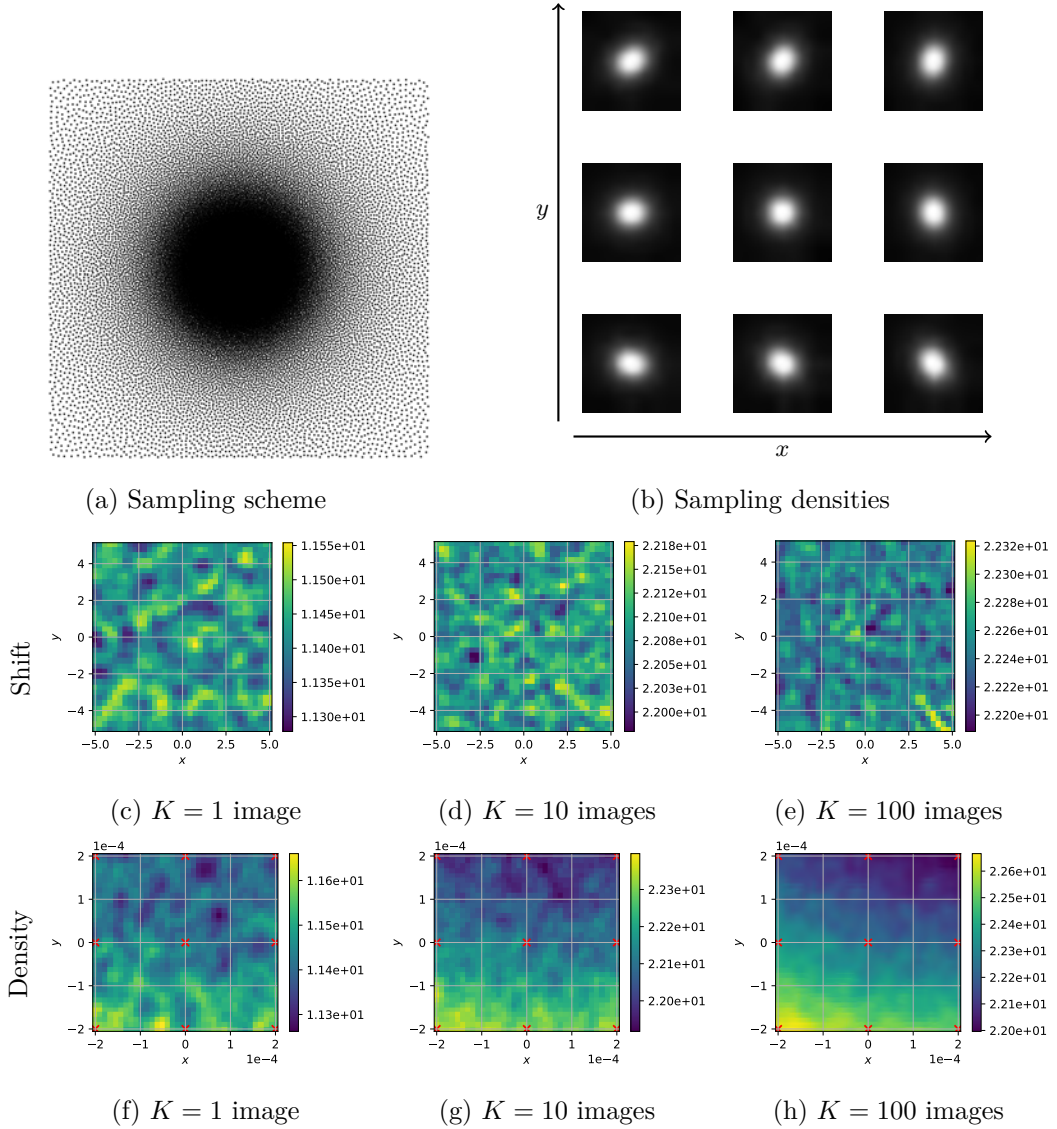


Figure 3.4: Illustration of the spurious minimizers. Here, we consider a total variation reconstructor and 25% under-sampling. Second row: cost function when the sampling scheme on the top-left is shifted along the  $x$  and  $y$  axes (grid size = 1 pixel). Last row: energy profiles when sampling using interpolation of the densities on the top-right. The  $3 \times 3$  red dots correspond to the densities on the top-right. Observe that the oscillation amplitude decays with the number of images, but spurious minimizers are present whatever the number of images.

database. In that setting, Bayesian optimization requires the following ingredients:

1. An initial sampling set.
2. A black-box evaluation routine of  $F(z, V)$ .
3. A family of interpolation functions together with a regression routine.
4. A solver that minimizes the regression function.

Hereafter, each choice made in this work is described.

**The initial sampling set** To initialize the algorithm, we need the convex set  $\mathcal{C}$  to be covered as uniformly as possible in order to achieve a good uniform approximation of the energy profile. In this work, we used a maximin space covering design [Pronzato 2017]. The idea is to construct a discrete set  $\mathcal{Z} = \{z_1, \dots, z_P\}$  that solves approximately

$$\max_{\mathcal{Z} \in \mathcal{C}^P} \min_{p' \neq p} \|x_p - x_{p'}\|_2. \quad (3.12)$$

In words, we want the minimal distance between pairs of points in  $\mathcal{Z}$  to be as large as possible. This problem is known to be hard. In this work we used the recent solver proposed in [Debarnot 2022] together with the Faiss library [Johnson 2019].

**The evaluation routine** Evaluating the cost function (3.6) is not an easy task. For just one realization of the noise  $n$  and image  $x_k$ , we need a fast reconstruction method and a fast way to evaluate the non-uniform Fourier transform. The technical details are provided in the appendix 3.5. Second,  $K$  might be very large. For instance, the fastMRI knee training database contains more than 30 000 slices of size  $320 \times 320$ . Hence, it is impossible to compute the complete function and it is necessary to either pick a random, but otherwise fixed subset of the images, or to consider random batches that would vary from one iteration to the next. A similar comment holds for the noise term  $n$ .

While Bayesian optimization allows the use of random functions, it requires evaluating integrals with Monte-Carlo methods, which is computationally costly. Hence, in all the forthcoming experiments, we will fix a subset of  $K$  images. In practice, we observed that using random batches increases the computational load without offering perceptible advantages.

**The interpolation process** In Bayesian optimization, a Gaussian process is used to model the underlying unknown function. This random process models both the function and the uncertainty associated with each prediction. This uncertainty is related to the fact that the function  $F$  is evaluated only at a finite number of points hence leading to an unknown behavior when getting distant from the samples. It is also related to the fact that the function evaluations might be noisy. Every sampled point has a zero variance when using a fixed realization or a low variance when using

random noise and batches. The variance increases with the distance between the sampled points.

In our experiments, the Gaussian process is constructed using a Matern kernel of parameter  $5/2$ , which is a popular choice for dimensions in the range  $[5, 20]$ . It is defined as

$$\Phi(z_1, z_2) = \left( 1 + \frac{\sqrt{5}\|z_1 - z_2\|_2}{\nu} + \frac{5\|z_1 - z_2\|_2^2}{3\nu^2} \right) \exp\left( \frac{-\sqrt{5}\|z_1 - z_2\|_2}{\nu} \right),$$

where  $\nu$  is a scaling parameter that controls the smoothness of the interpolant and its point-wise variance. In practice, the value of  $\nu$  is a parameter that is optimized at each iteration when fitting the Gaussian process to the sampled data.

The interpolant mean and its variance are then constructed by solving a linear system constructed using the kernel  $\Phi$  and the sampled points  $z_1, \dots, z_P$ . We refer to [Frazier 2018b] for more details.

**Sampling new points** Bayesian optimization works by iteratively sampling new points. The point in the sampling set with lowest function value, is an approximation of the minimizer. To choose a new point, there is a trade-off between finding a better minimizer in the neighborhood of this point and space exploration. Indeed, big gaps in between the samples could hide a better minimizer. This trade-off is managed through a so-called utility function. In this work, we chose the expected improvement [Frazier 2018b], resulting in a new function  $\mathcal{L}(z)$ . The new sampled point is found by solving a constrained non-convex problem:

$$\inf_{z \in \mathcal{C}} \mathcal{L}(z)$$

Since the function  $\mathcal{L}$  is non-convex, we use a multi-start strategy. We first sample 1000 points evenly in  $\mathcal{C}$  using a maximin design. Then, we launch many projected gradient descents on  $\mathcal{L}$  in parallel, starting from those points. The best critical point is chosen and added as a new sample.

This process requires projecting  $z$  on  $\mathcal{C}$  defined in (3.10). To this end, we designed an efficient first order solver.

## 3.3 Numerical experiments and results

### 3.3.1 The experimental setting

**Database and computing power** Throughout this section, we used the fastMRI database [Zbontar 2018]. It contains MRI images of size  $320 \times 320$ . We focused on the single coil and fully sampled knee images. The training set is composed of 973 3D volumes, which represents a total of 34 742 slices. The validation set has 199 volumes and 7135 slices.

Some images in the dataset have a significant amount of noise. This presents two significant drawbacks: i) the signal-to-noise-ratio of the reconstructed images is

increased artificially and ii) we have shown that noise can dramatically impact the convergence of off-the-grid Fourier sampling optimization [Gossard 2022b]. To mitigate these effects, we pre-processed all the slices using a non-local mean denoising algorithm [Buades 2011].

The experiments are conducted on the Jean-Zay HPC facility. For each task we use 10 cores and an Nvidia Tesla V100 with 16GB of memory.

**Sampling** The bounds of the constraint sets in (3.3) are given by:

$$\alpha = \Delta t \gamma \frac{G_{max}}{K_{max}} \quad \text{and} \quad \beta = \Delta t^2 \gamma \frac{S_{max}}{K_{max}}, \quad (3.13)$$

where  $\Delta t$  is the sampling step of the scanner. Following [Chaithya 2022], we used the following realistic hardware constraints:  $G_{max} = 40\text{mT/m}$ ,  $S_{max} = 180\text{T/m/s}$ ,  $K_{max} = 2\pi$  and  $\gamma = 42.57\text{MHz/T}$ . The value of  $\Delta t$  is fixed to ensure that at maximal speed, the distance between two consecutive points equals the Shannon-Nyquist rate [Lazarus 2020a].

We consider two different scenarii: 25% and 10% undersampling. Each shot consists of 646 acquisition points and we use  $N_s = 40$  shots and  $N_s = 16$  shots respectively for the 25% and the 10% undersampling. Each shot is constrained to start at the center of the k-space. The first few points of each trajectory are fixed to be radial, see Section 3.5.5.4 for the technical details.

The family of densities is generated using the process described in Section 3.2.3.3 with  $10^4$  densities generated at random.

**Sampling baseline** All the optimized schemes are compared to a state-of-the-art handcrafted baseline: the SPARKLING method described in [Lazarus 2019]. There, the attraction-repulsion problem (3.11) is solved with a radial density  $\rho$ . Its value at the center has been optimized to yield the best possible signal-to-noise ratio on the validation set in a way similar to [Chaithya 2021]. The corresponding point configuration is given in Fig. 3.6a and Fig. 3.6b for the 25% and 10% undersampling rates respectively. It provides a 7dB improvement compared to the usual radial lines commonly found in the literature (see the first two rows of Table 3.2).

**Image reconstruction** The experiments are conducted with two reconstruction models:

- a total variation reconstruction method with 120 iterations of Algorithm 1 in Section 3.5.1 and with a regularization parameter  $\lambda = 10^2$  and,
- an unrolled network (NN) with 6 iterations of ADMM and a DruNet as the denoising step [Zhang 2021b], 30 iterations of the CG algorithm that initializes the ADMM and 10 iterations of CG to solve the data-consistency equations at each iteration.

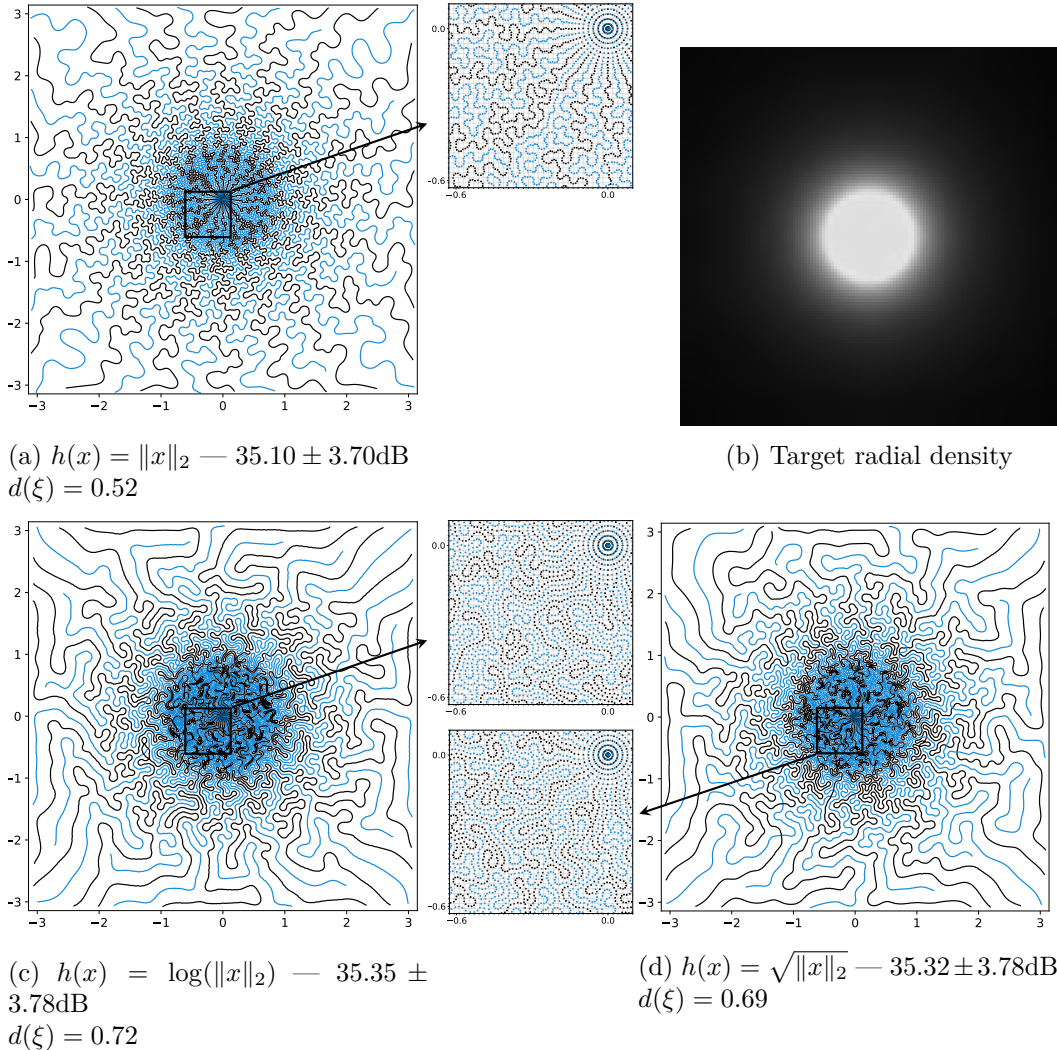


Figure 3.5: On the importance of the discrepancy’s kernel  $h(x)$ . The same density is sampled with different kernels. The average PSNR of the reconstructed images on the validation set is displayed with its standard deviation. The average distance between contiguous points on the trajectories is displayed as  $d(\xi)$ .

### 3.3.2 Choosing a kernel for the discrepancy

In all the previous “SPARKLING” papers [Lazarus 2019, Chauffert 2017], the kernel function  $h(x) = \|x\|_2$  was used. This choice seems like the most natural alternative since it is the only one which is scale invariant in the unconstrained setting. This means that if  $\Xi = (\mathbb{R}^D)^M$  and if a density  $\rho$  is dilated by a certain factor, then so is the optimal sampling scheme. However, this property is not true anymore when constraints are added. In that case, the choice of kernel turns out to be of importance.

To illustrate this fact, we considered the three different radial kernels  $h(x) =$

$\|x\|_2$ ,  $h(x) = \sqrt{\|x\|_2}$  and  $h(x) = \log(\|x\|_2)$ . As can be seen on Fig. 3.5, performance variations of more than 0.2dB are obtained depending on the kernel. The reason is that contiguous points on the trajectories are spaced more or less depending on this choice. For instance, observe that the points on the zoom of Fig. 3.5a are more packed along the trajectories than on Fig. 3.5c. To compensate for this higher longitudinal density, the sampler then increases the distance between adjacent trajectories, thereby creating holes in the sampling set. This is detrimental, since low frequency information is lost in the process. This effect can be quantified by evaluating the distances between contiguous points in the k-space center. As can be seen, it goes from 0.52 for the usual kernel  $h(x) = \|x\|_2$  to a significantly higher value 0.72 for the logarithmic kernel. The latter kernel creates a higher repulsion for neighboring points.

### 3.3.3 Bayesian optimization: database size and numerical complexity

In this section, we aim at evaluating the computational complexity of the Bayesian optimization routine. To this end, we study the impact of the number of images  $K$  in the dataset, the size of the initial sampling set and the number of iterations, which are governing the algorithm's complexity. Table 3.1 summarizes our main findings for the total variation reconstruction and unrolled neural network.

There, we see that the number of images  $K$  in the dataset has nearly no influence on the quality of the final sampling density. Taking  $K = 32$  or  $K = 512$  images yields an identical PSNR on the validation set. This holds both for the 25% and 10% undersampling rates. As can be seen in the Tables, reconstructing as little as  $200 \times 32$  images is enough to reach the best possible density in the family. The same conclusion holds for the 10% undersampling rate. This represents 18% of a single epoch.

We also see that the initial sampling set of the convex  $\mathcal{C}$  plays a marginal role on the quality of the final result. In addition, taking a small number of initial points allows to reduce the overall complexity of the algorithm to reach a given PSNR.

### 3.3.4 Comparing optimization routines for the total variation reconstructor

In what follows, we aim at comparing two different sampling optimization approaches:

**Trajectory optimization** The minimization of (3.4) in the space of trajectories.

We use a modified version of the multi-scale approach in [Wang 2022a], see Section 3.5.5.1.

**BO density** The Bayesian approach to minimize (3.6) globally.

To compare these approaches, we conduct various experiments. The corresponding results are shown in Table 3.3, Table 3.2 and Fig 3.6. Below, we summarize our



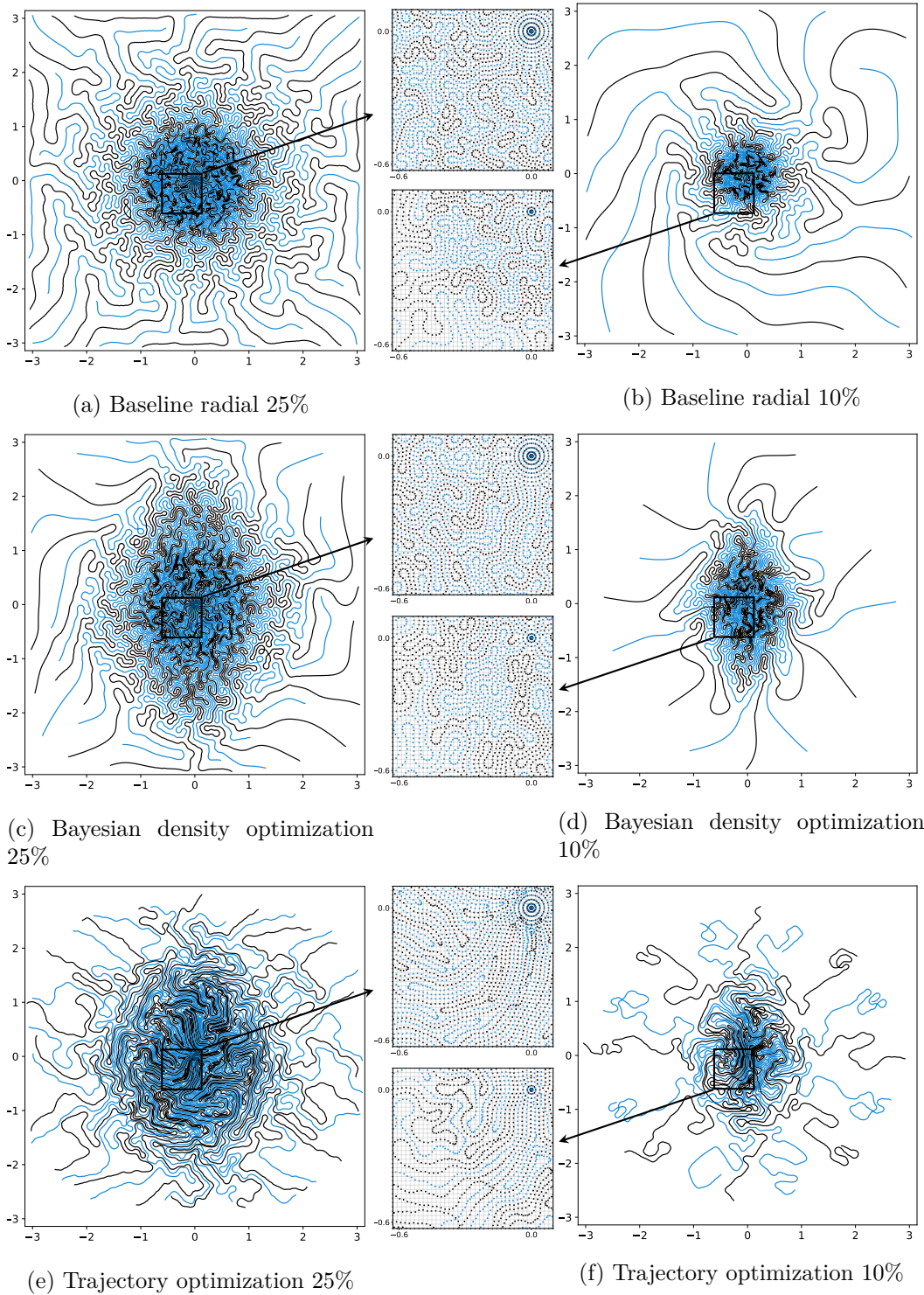


Figure 3.6: Optimized sampling schemes with various optimization approaches. A total variation reconstruction algorithm is used.



Method	# init. points	# eval.	$K = 32$ imgs.	$K = 128$ imgs.	$K = 512$ imgs.
TV	20	200	$35.64 \pm 3.82$	$35.65 \pm 3.82$	$35.65 \pm 3.82$
TV	100	300	$35.63 \pm 3.81$	$35.66 \pm 3.82$	$35.66 \pm 3.82$
TV	200	300	$35.65 \pm 3.81$	$35.66 \pm 3.82$	$35.66 \pm 3.82$
NN	20	200	$38.14 \pm 4.77$	$38.10 \pm 4.75$	$38.09 \pm 4.76$
NN	100	300	$38.17 \pm 4.79$	$38.05 \pm 4.73$	$38.08 \pm 4.75$
NN	200	300	$38.20 \pm 4.80$	$38.08 \pm 4.75$	$38.10 \pm 4.76$

Table 3.1: Bayesian optimization on a convex set  $\mathcal{C}$  of dimension  $L = 20$  using a total variation reconstruction algorithm and an unrolled network for 25% undersampling. The PSNR is evaluated for the optimized density on the validation dataset containing 7135 images. The total number of cost function evaluations is given in the second column.

main findings.

**Qualitative comparison of the sampling schemes** In this paragraph, we compare our method with existing works [Wang 2022a, Weiss 2021]. The optimized sampling schemes are shown in Fig. 3.6 for the TV reconstructor. In Fig. 3.6, we see the results of the different optimization routines.

The two optimization methods yield anisotropic sampling schemes with a higher density along the vertical axis. However the trajectories present significant differences.

The Bayesian optimization yields a sampling scheme which covers the space more uniformly. The trajectories have a significantly higher curvature at the k-space center. These features are somehow hard-coded within the sampling generator  $\mathcal{S}_M$  described in Section 3.2.3.4.

The trajectory optimization yields trajectories which are locally linear and aligned at a distance of about a pixel. This suggests that the trajectory optimization favors Shannon’s sampling rate at the center of the k-space. A potential explanation is as follows. When the sampling points are close to a subgrid [Gossard 2022b], the adjoint of the forward operator  $A(\xi)^*$  is roughly the pseudo-inverse. Using a points configuration close to a subgrid therefore helps iterative reconstruction algorithms to converge.

Finally, at the bottom-left of the zoomed region on the 25% undersampling rate, it seems that Bayesian optimization (Fig. 3.6c) yields a density slightly higher than trajectory optimization (Fig. 3.6e). This density is critical for the reconstruction quality and might explain a part of the quantitative differences observed in the next section.

**Performance comparison** Table 3.2 reveals that the trajectory optimization yields better performance than the Bayesian optimization approach both for the

Method	25%	10%
Radial scheme	$27.87 \pm 2.75\text{dB}$ $0.66 \pm 0.12$	$24.28 \pm 2.67\text{dB}$ $0.57 \pm 0.12$
Sparkling radial (baseline)	$35.35 \pm 3.78\text{dB}$ $0.85 \pm 0.11$	$32.94 \pm 3.20\text{dB}$ $0.79 \pm 0.14$
Bayesian optim. $K = 32$	$35.66 \pm 3.82\text{dB}$ (+0.31dB) $0.86 \pm 0.11$	$33.41 \pm 3.26\text{dB}$ (+0.47dB) $0.80 \pm 0.14$
Trajectory optim. $K = 34742$	$35.92 \pm 3.89\text{dB}$ (+0.57dB) $0.87 \pm 0.11$	$33.48 \pm 3.31\text{dB}$ (+0.54dB) $0.80 \pm 0.14$
Trajectory optim. $K = 32$	$35.67 \pm 3.88\text{dB}$ (+0.32dB) $0.86 \pm 0.11$	$32.84 \pm 3.19\text{dB}$ (-0.10dB) $0.79 \pm 0.14$
Trajectory optim. $K = 128$	$35.67 \pm 3.85\text{dB}$ (+0.32dB) $0.86 \pm 0.11$	$32.89 \pm 3.17\text{dB}$ (-0.05dB) $0.79 \pm 0.14$

Table 3.2: Comparison of different optimization procedures for the TV reconstructor with different numbers of images  $K$  in the training set. For each test case, the first line is the PSNR and the second line is the SSIM. The number after  $\pm$  indicates the standard deviation.

25% (+0.26dB) and 10% (+0.07dB) undersampling rates. This was to be expected since the density optimization is much more constrained. The difference is however mild.

We also report some of the best (resp. worst) PSNR increase (resp. decrease) in Fig. 3.7. For each case, we selected 3 images that are representative among the top 10 best (resp. worst) images of the test dataset. For both the trajectory optimization and the Bayesian optimization method, the images that have the largest PSNR increase have large vertical structures. This increase might be due to the anisotropy of the optimized schemes that are more adapted to the center slices of the 3D knee images. On the contrary, the images having the largest PSNR decrease are outliers that are not prevalent in the dataset such as extreme slices. Notice that images that have the best (resp. worst) PSNR increase (resp. decrease) are the same for the Bayesian optimization and for the trajectory optimization.

**Computing times** Table 3.3 gives the computation times for each method with the total variation reconstruction method. The proposed approach has the significant advantage of giving an optimized sampling scheme with guarantees on the underlying density with a reduced computational budget and with a reduced number of images. As can be seen, our approach requires only 32 images and 3 hours. This has to be compared to the 85 hours (3 days and a half) needed by the trajectory optimization routine.

This feature is a significant advantage of our approach. It could be key element when targeting high resolution images or 3D data.

Method	Computational time
Trajectory optimization	85h
Bayesian optimization	
Optimization $K = 32$	3h
Optimization $K = 128$	4h

Table 3.3: Computational cost of the different optimization procedures (25% undersampling and TV reconstructor) with an NVIDIA Quadro RTX 5000 GPU.

**Size of the training set** As advertised, the Bayesian optimization approach works even for small datasets. The trajectory optimization routine also provides competitive results with only 32 images in the training set. However, the performance collapses for the 10% undersampling rate. Increasing the size of the training set to  $K = 128$  does not improve the situation. This feature is in strong favor of our approach, when having access to a limited dataset.

### 3.3.5 Comparing optimization routines for a neural network reconstructor

The aim of this section, is to compare three different sampling optimizers:

- The Bayesian density optimization solver proposed in this paper.
- The trajectory optimization solver with a fixed unrolled neural network trained on a family of sampling schemes, see Section 3.5.3. This is a novelty of this paper.
- An optimization routine minimizing the trajectories and the unrolled network weights simultaneously, as proposed in [Wang 2022a, Weiss 2021].

**Qualitative comparisons** The differences between the density optimization and the trajectory optimization can be observed on Fig. 3.8. They are much more pronounced than for the total variation reconstructor. Surprisingly, the trajectory optimized sampling schemes leave large portions of the low frequencies unexplored. Hence, it seems that the unrolled network is able to infer low frequency information better than the traditional total variation prior. This suggests that the existing compressed sampling theories designed for the Fourier-Wavelet system have to be revised significantly to account for the progress in neural network reconstructions. The optimization of a trajectory for a fixed sampling scheme or the joint optimization yield qualitatively similar trajectories, with perhaps larger unexplored parts of the k-space for the fixed reconstruction method.

**Quantitative comparisons** Table 3.4 allows comparing the different methods quantitatively. BO yields a PSNR increase almost twice lower than the multi-scale optimization (+0.94dB VS +1.83dB for 25% and +0.64dB VS +1.16dB for 10%).

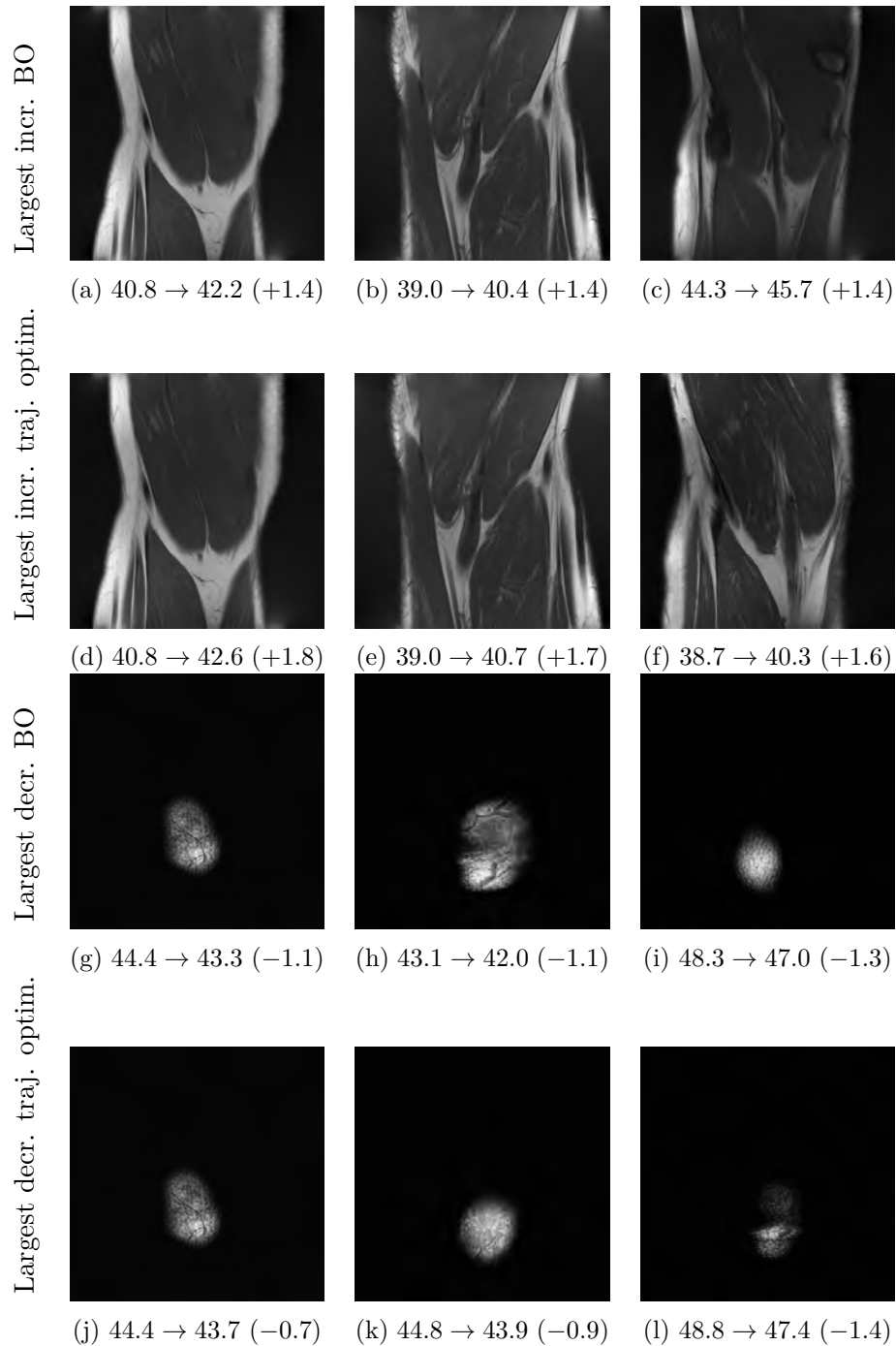


Figure 3.7: Sample of images that have the largest increase (resp. decrease) of the PSNR for the different optimization methods with the TV reconstructor and 25% undersampling. The numbers below the images are the PSNR using the baseline sampling scheme and the PSNR using optimized trajectories.

Method	25%	10%
Baseline with unrolled net	37.26 ± 4.57dB 0.89 ± 0.09	34.49 ± 3.71dB 0.83 ± 0.13
BO scheme $K = 128$ with unrolled net	38.20 ± 4.80dB (+0.94dB) 0.91 ± 0.08	35.13 ± 3.92dB (+0.64dB) 0.86 ± 0.13
Traj. optim. with fixed unrolled net	39.09 ± 5.06dB (+1.83dB) 0.92 ± 0.07	35.65 ± 4.18dB (+1.16dB) 0.85 ± 0.12
Joint optim. multi-scale	39.03 ± 4.87dB (+1.77dB) 0.92 ± 0.07	35.53 ± 4.05dB (+1.04dB) 0.85 ± 0.12

Table 3.4: Comparison of different optimization procedures for the unrolled ADMM reconstructor. For each test case, the first line is the PSNR and the second line is the SSIM. The increase compared to the baseline scheme is shown in parentheses.

This can likely be explained by the fact that the chosen family of densities (dimension 20) is unable to reproduce the complexity of the optimized trajectories. It is possible that richer sampling densities could reduce the gap between both approaches. However, Bayesian optimization is known to work only in small dimension and it is currently unclear how to extend the method to this setting.

Interestingly, the trajectory optimized with a fixed unrolled neural network trained on a family provides slightly better results ( $\approx +0.1\text{dB}$ ) than the joint optimization. This suggests that the joint optimization gets trapped in a local minimizer since it can only be better if optimized jointly with the reconstructor.

### 3.4 Conclusion

In this work, we designed efficient optimization algorithms that either optimize trajectories directly or learn a sampling density and an associated sampling pattern in MRI. Overall, the main highlights of this work are:

- The compressed sensing theories designed for the Fourier-Wavelet system with  $\ell^1$  reconstruction (e.g. [Adcock 2021]) seem nearly optimal from an experimental point of view. Sampling schemes can be designed based on a density that is close to Shannon’s rate at the k-space center and that decays towards the high frequencies. The precise shape of the density depends on the images structure.
- In that context, the Bayesian optimization of densities is an attractive method to design sampling schemes. It works with small datasets, ensures the convergence to a global minimizer. Its performance is close to much heavier trajectory optimizers and is from one to two orders of magnitude faster.
- In the case of unrolled neural network reconstructions, the proposed Bayesian optimization framework is still interesting with gains of up to 1dB in average

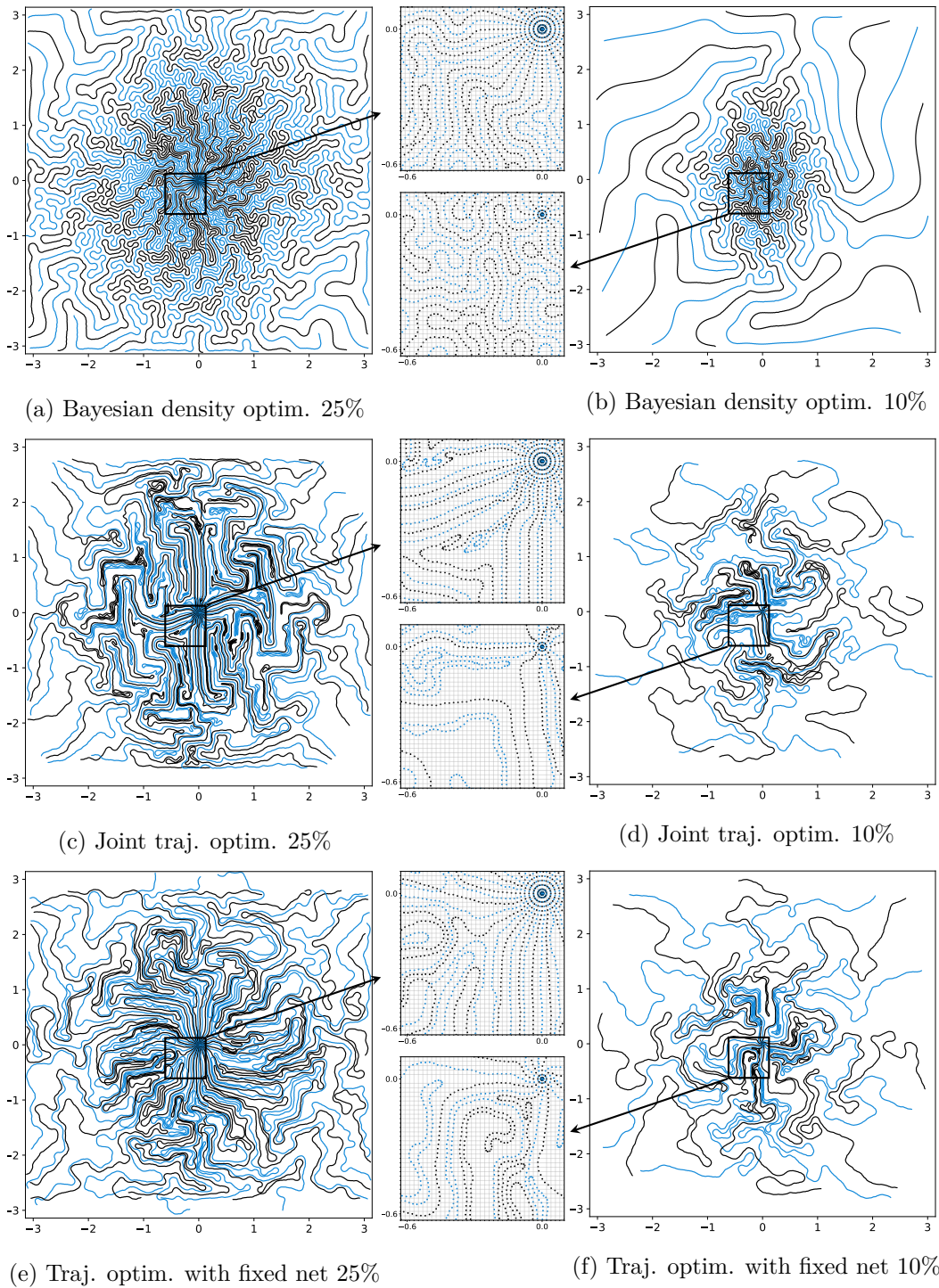


Figure 3.8: Optimized sampling schemes with the various optimization approaches for a neural network reconstruction.

on the fastMRI knee validation set. However, the gain can be nearly doubled with a direct optimization of the trajectories. A possible explanation for this fact is that the family of densities is too poor to describe the best convoluted trajectories.

- We also improved the Sparkling trajectories [Lazarus 2019], by changing the discrepancies.
- We also provided various improvements to the direct optimization of trajectories by using the Extra-Adam algorithm to handle hard constraints and by training reconstruction networks on families of operators.

## 3.5 Implementation details

### 3.5.1 TV reconstruction algorithm

In this part we detail the TV iterative reconstruction algorithm that is used in this paper. We consider a regularized version of the total variation of the form

$$TV_\epsilon(x) = \sum_{n=1}^N \sqrt{\|(\nabla x)[n]\|_2^2 + \epsilon^2}.$$

Given  $y \in \mathbb{C}^M$ , the solver of problem (3.2) is given in Algorithm 1. The parameter  $\alpha$  drives the acceleration and  $D$  is the dimension, here  $D = 2$ . It corresponds to a Nesterov accelerated gradient descent [Nesterov 1983] with a regularized version of the  $\ell^1$  norm. A critical point is the choice of the step  $\tau$  in Algorithm 1. This step is

---

**Algorithm 1** A TV minimization algorithm

---

**Require:** Number of iterations  $Q$ .

Set  $z^{(0)} = x^{(0)} = 0$ ,  $\tau = \frac{1}{\|A(\xi)\|_{2 \rightarrow 2}^2 + 4D\lambda/\epsilon}$ .

**for all**  $q = 0$  to  $Q - 1$  **do**

$r^{(q)} = A(\xi)^*(A(\xi)z^{(q)} - y)$

$x^{(q+1)} = z^{(q)} - \tau [r^{(q)} + \lambda \nabla TV_\epsilon(z^{(q)})]$

$z^{(q+1)} = x^{(q+1)} + \alpha(x^{(q+1)} - x^{(q)})$

**end for**

**return**  $x^{(Q)}$ .

---

computed using the spectral norm of the data fidelity term which can be computed using a power iteration method for each point configuration  $\xi$ . The resulting step is taken into account in the computation of the gradient with respect to the locations  $\xi$  of the cost function in (3.4).

### 3.5.2 The unrolled neural network

The neural network based reconstruction is an unrolled network. The one used in this work is based on the ADMM (Alternative Descent Method of Multipliers)

[Ng 2010]. It consists in alternating a regularized inverse followed by a denoising step with a neural network. If  $\mathcal{D}_{\lambda^{(p)}}$  denotes the denoiser used at iteration  $p$ , the unrolled ADMM can be expressed through the sequence:

$$\begin{cases} x^{(p+1)} = (A(\xi)^* A(\xi) + \beta \text{Id})^{-1} (A(\xi)^* y + \beta z^{(p)} - \mu^{(p)}) \\ z^{(p+1)} = \mathcal{D}_{\lambda^{(p)}} \left( x^{(p+1)} + \frac{\mu^{(p)}}{\beta} \right) \\ \mu^{(p+1)} = \mu^{(p)} + \beta (x^{(p+1)} - z^{(p+1)}) \end{cases}$$

with a pseudo-inverse initialization  $z^{(0)} = A(\xi)^\dagger y$ .

In this work, we use the DruNet network [Zhang 2021b] to define the denoising mappings  $\mathcal{D}_{\lambda^{(p)}}$ . We choose an ADMM algorithm for the following reasons:

1. for well-spread sampling schemes, the matrix  $A(\xi)^* A(\xi)$  has a good conditioning and the linear system that has to be inverted can be solved in less than a dozen iterations,
2. it has demonstrated great performance to solve linear inverse problems in imaging, including image reconstruction from Fourier samples [Wang 2022a].

We opted for a different network at each iteration instead of a network that shares its weights across all iterations. This leads to slightly higher performance at the price of a slightly harder to interpret architecture (see e.g. [Genzel 2022a] for a similar discussion in CT reconstruction).

### 3.5.3 Training the reconstruction network for a family of operators

Following [Gossard 2022c], we trained our network in a non usual way. Instead of training the denoising networks  $\mathcal{D}_{\lambda^{(p)}}$  for a single operator  $A(\xi_0)$ , we actually trained it for a whole family of operators  $\mathcal{A} = \{A(\xi), \xi \in \mathcal{F}\}$ , where  $\mathcal{F}$  is a large family of sampling schemes. We showed in [Gossard 2022c], that this simple approach yields a much more robust network, which is adaptive to the forward operator.

In our experiments, the network is trained on a family of  $10^3$  sampling schemes that are generated using the attraction-repulsion minimization problem (3.11). These schemes are parameterized by densities that are within  $\mathcal{C}$ . This pretraining step consists of 32 epochs with a batch of 8 images using the Adam optimizer with default parameters ( $\beta_1 = 0.9$  and  $\beta_2 = 0.999$ ). The step for the CNN weights is set to  $10^{-4}$  with a multiplicative update of 0.95 after each epoch. The measurements are perturbed by an additive white noise (see  $n$  in (3.4)).

### 3.5.4 Joint optimization

Instead of optimizing the sampling scheme for a fixed network, we can also optimize jointly the sampling locations together with the network weights. This approach



was proposed in [Wang 2022a, Weiss 2021]. Due to memory requirements, we set the batch size to 7 for the unrolled network in our training procedure. The step size for the CNN weights in this experiment is also set to  $10^{-4}$  with the default Adam parameters.

### 3.5.5 Computational details

In this paragraph, we describe the main technical tools used to optimize the reconstruction process.

#### 3.5.5.1 Solving the particle problem (3.4)

Problem (3.4) is a highly non-trivial problem. Two different computational solutions were proposed in [Weiss 2021, Wang 2022a]. In this work, we re-implemented a solver with some differences outlined below.

First, the optimization problem (3.4) involves a nontrivial constraint set  $\Xi$ . While the mentioned works use a penalization over the constraints, we enforce the constraints by using a projection at each iteration. Handling constraints in stochastic optimization was first dealt with stochastic mirror-prox algorithms [Juditsky 2011]. This approach turned out to be inefficient in practice. We therefore resorted to an extension of Adam in the constrained case called Extra-Adam [Gidel 2019]. The step size was set to  $10^{-3}$  and the default Adam parameters  $\beta_1 = 0.9$  and  $\beta_2 = 0.999$ . We observed no significant difference by tuning these last two parameters. We also use a step decay of 0.9 each fourth of epoch and batch size of 13, which is the largest achievable by our GPU.

Similarly to [Weiss 2021, Wang 2022a], we use a multi-scale strategy. The trajectories are defined through a small number of control points, that progressively increases across iterations. We simply use a piecewise linear discretization (contrarily to higher order splines in [Wang 2022a]). The initial decimation factor is  $2^7$  and is divided by two every two epochs. This results in a total number of epochs equal to 14 and takes about 86 hours for a total variation solver. In comparison [Wang 2022a], reports a total of 40 epochs.

#### 3.5.5.2 Implementing the Non-uniform Fourier Transform (NUFT)

Various fast implementations of the Non-uniform Fourier Transform (3.1) are now available [Keiner 2009, Fessler 2003, Shih 2021, Muckley 2020b]. In this work, we need a pyTorch library capable of backward differentiation. Evaluating the gradient of the cost function in (3.4) or in (3.6) indeed requires computing the differential of the forward operator  $A(\xi)$  with respect to  $\xi$ . This can be done by computing  $D$  non-uniform Fourier transforms (see [Wang 2022a, Gossard 2022b, Wang 2021]). Different packages were tested and we finally opted for the cuFINUFFT implementation [Shih 2021]. The bindings for different kind of NUFT are available at <https://github.com/albangossard/Bindings-NUFFT-pytorch/>.

### 3.5.5.3 Minimizing the discrepancy

The minimization of the discrepancy (3.11) is achieved with a gradient descent, as was proposed in the original paper [Schmaltz 2010], see Algorithm 2. The input parameters are the initial sampling set  $\xi^{\text{ini}}$ , the target density  $\rho$  and a step-size  $\tau > 0$ . The step size needs to be carefully chosen to ensure a fast convergence. The optimal choice can be shown to be related to the minimal distance between adjacent points. In our experiments, it was tuned by hand and fixed respectively to  $2 \times 10^4$  and  $5 \times 10^3$  for the 25% and 10% undersampling schemes.

The expression (3.11) can be splitted in two terms: an attraction term and a repulsion term (see [Chauffert 2017]). The attraction term is a convolution and an approximation can be computed efficiently using the FFT. Computing the repulsion term is more challenging numerically. In fact, the repulsion term involves pairwise interactions between the set of points which can have up to hundred thousands of points. A naive method have a quadratic complexity which would be a bottleneck on CPU. This term often appears in electrostatic halftoning and the design of efficient procedures to compute such terms has drawn a lot of attention. This term can be computed using the Fast Multiple Method (FMM) which relies on approximations to group the interactions of points that are close to each other and compute an approximation of the repulsion term efficiently. It can also be computed using a NUFT [Potts 2003] and there are now mature implementations with  $O(N \log(N))$  complexity. However, with the recent advances in GPU computing, operations with  $O(N^2)$  complexity which were a bottleneck in the past, are now becoming more and more feasible with reasonable number of particles (a few dozens of thousands) and they achieve similar results as more intricate algorithms (see Section 1.1.4.1). For this kind of application, the bottleneck is the bandwidth usage of the GPU and some libraries are starting to propose efficient and easy to use tools to implement reduction operations that take advantage of the massive parallel structure of GPUs [Charlier 2021a]. Computing the gradient requires to compute pairwise interactions between all particles: in our codes, it is achieved using PyKeOps [Charlier 2021a]. This approach presents the advantages of being fast, adapting to arbitrary kernels  $h$  and to natively allow backward differentiation within PyTorch. For a number of particles  $M$  above  $10^6$ , fast multipole methods might become preferable [Chaithya 2022].

---

**Algorithm 2** Gradient descent to minimize (3.11).

---

```

Set  $\zeta^{(0)} = \xi^{\text{ini}}$ 
for  $j = 1 \dots J$  do
     $\zeta^{(j)} = \Pi_{\Xi} \left( \zeta^{(j-1)} - \tau \nabla_1 \text{dist}(\zeta^{(j-1)}, \rho) \right)$ 
end for
Set  $\xi^{(n)} = \zeta^{(J)}$ 

```

---

#### 3.5.5.4 Handling the mass at 0

An important issue is related to the fact that all trajectories start at the k-space origin. This creates a large mass for the sampling scheme at 0. When minimizing a discrepancy between the sampling scheme and a target density, the sampling points are therefore repulsed from the origin, creating large holes at the center. To avoid this detrimental effect, we fix rectilinear radial trajectories at the origin at maximal acceleration until a distance of 0.5 pixel between adjacent trajectories samples is reached. This creates a fixed pattern in the k-space center, which can easily be seen in the zoom of Fig. 3.6a and Fig. 3.6b. We compute the discrepancy only at the exterior of a disk centered at the origin containing this fixed pattern.

#### 3.5.5.5 Projection onto the constraint set

The projector onto the constraint set is used twice in this work. First the Extra-Adam algorithm requires a Euclidean projector on the constraint set  $\Xi$  to solve (3.4). This projector is also needed to compute one evaluation of the sampler in (3.11). In this project, we used the dual approach proposed in [Chaffert 2016], implemented on a GPU. This algorithm can be implemented in PyTorch, and can be differentiated. This allows computing the gradient of the overall function in (3.6).

## 3.6 Sampling density and Shannon’s condition

In this section, we aim at illustrating the claims in Section 3.1.1 for a total variation reconstructor.

### 3.6.1 How to control the sampling density?

Let  $\omega$  denote a pixel in the Fourier domain and  $\rho : \mathbb{R}^D \rightarrow \mathbb{R}$  denote a sampling density. If the sampling points could be placed arbitrarily in space, satisfying Shannon-Nyquist rate would be met by imposing  $M \int_{x \in \omega} \rho(x) dx = 1$ . This equation indeed means that the pixel  $\omega$  should contain in average one point. Unfortunately, the existence of trajectory constraints makes the condition more intricate. Indeed, they impose a maximal distance between adjacent samples. Hence the condition above would result in distances smaller than Nyquist rate along trajectories, and distances larger along distant pieces of trajectories to compensate for the closeness of adjacent samples.

A simple solution to remedy this problem is to increase the sampling density in the center of the k-space. This is actually what was done in [Lazarus 2019, Chaithya 2022, Chaithya 2021], perhaps without a clear justification.

In order to evaluate whether Shannon-Nyquist rate is satisfied at the k-space center, we can measure the average distance between adjacent points in the sampling scheme. To this end, let  $C_m = \{x \in \mathbb{R}^D, \|x - \xi_m\|_2 \leq \|x - \xi_{m'}\|_2, \forall m' \neq m\}$  denote

the Voronoi cell associated to  $\xi_m$ . We define the radius of the  $m$ -th Voronoi cell as

$$r_m = \sqrt{2} \max_{x \in C_m} \|x - \xi_m\|_2. \quad (3.14)$$

If the radius is larger than 1, it means that Shannon’s condition is not met. In the forthcoming experiments, we will analyze the distribution of the Voronoi radii for all sampling points  $\xi_m$  that are within a disk  $S$  centered in the Fourier domain.

### 3.6.2 Performance variance for a fixed sampling density

In this experiment, we aim at showing that the sampling efficiency does not vary significantly, under the assumption that the k-space center is sampled at Shannon’s rate. The experiments are conducted with a total variation reconstruction algorithm.

The conclusions are as follows:

1. For a fixed sampling density, the standard deviation of the PSNR w.r.t. the sampling schemes increases gradually as the radii  $r_m$  increases.
2. In addition, this standard deviation is negligible when the radii are not above 1, justifying our claim.

Table 3.5 gives the average PSNR computed on 512 images for different sampling densities. For each density, 10 different sampling schemes are generated by using different random initializations (see [Boyer 2016]). For each sampling scheme, we compute the average PSNR and the standard deviation  $\hat{\sigma}$  of the average PSNR w.r.t. to the sampling schemes. The density at the center is controlled by the scalar  $r$  in equation (3.8). The repartition of the Voronoi cells radii is shown in Fig. 3.9.

Table 3.5 shows that for values of  $r$  below 1.1, the variance increases significantly. This means that below this value, the actual samples position matter. When  $r > 1.1$ , its influence seems negligible. The positions of the points from 2 different sampling schemes with fixed density can be different, as shown on the third row of Table 3.5.

## 3.7 Resampling from estimated density

In this section, we investigate the choice of the density parametrization. We propose to resample the density by solving (3.11) with a density that has been estimated from the trajectory optimization.

### 3.7.1 Density estimation

First we validate a method to estimate the density given a sampling scheme. Starting from the generated trajectories from (3.11), the density is estimated using a kernel density estimation. We then compare this estimation to the ground truth.






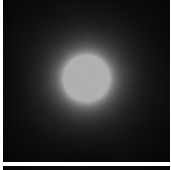
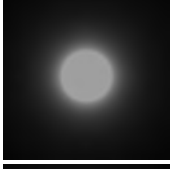
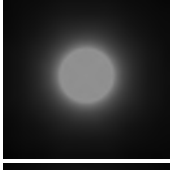

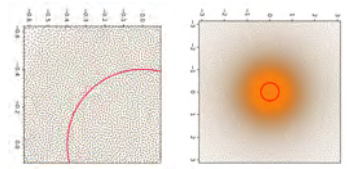
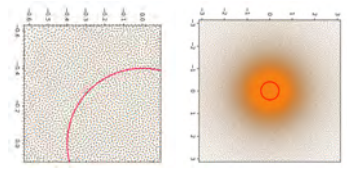
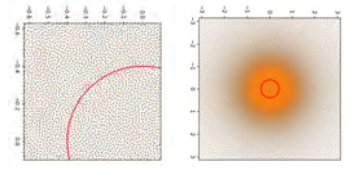
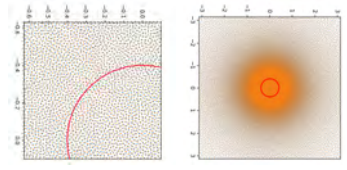
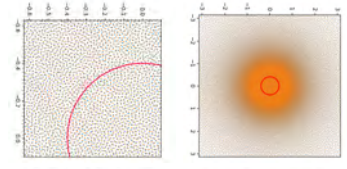
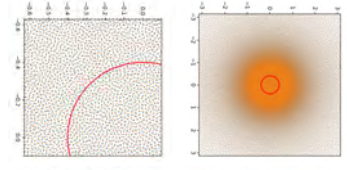
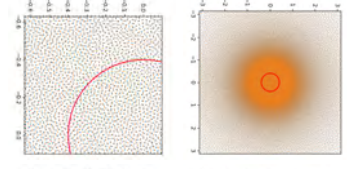
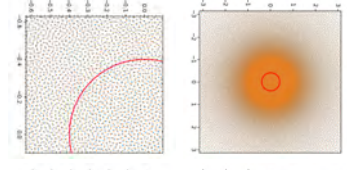
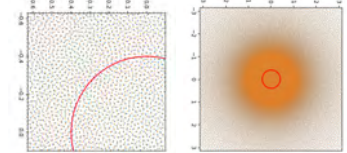
Density at the center	$r = 1.6$	$r = 1.5$	$r = 1.4$	$r = 1.3$	$r = 1.2$	$r = 1.1$	$r = 1.0$	$r = 0.9$	$r = 0.8$
Density									
Sampling scheme									
Average PSNR	34.97	35.02	35.06	35.12	35.16	35.15	34.89	26.49	21.51
$\hat{\sigma}$	0.0028	0.0027	0.0048	0.0098	0.0298	0.0679	0.5011	1.3610	2.0165
Average radius $r_m$	0.78	0.81	0.84	0.87	0.91	0.95	1.00	1.06	1.12

Table 3.5: Average PSNR evaluated on 512 images and its standard deviation for 10 different sampling schemes. Different densities with different sampling rates are being used. For readability, for each density only 2 out of the 10 sampling schemes are displayed. The red circle indicates the area where the distribution of  $r_m$  is computed.

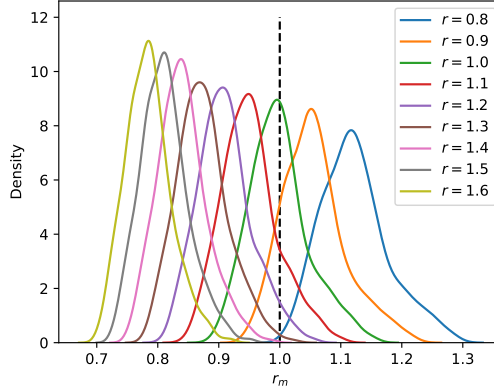


Figure 3.9: Empirical probability density function of the Voronoi cells radii  $r_m$  for different densities at the k-space center.

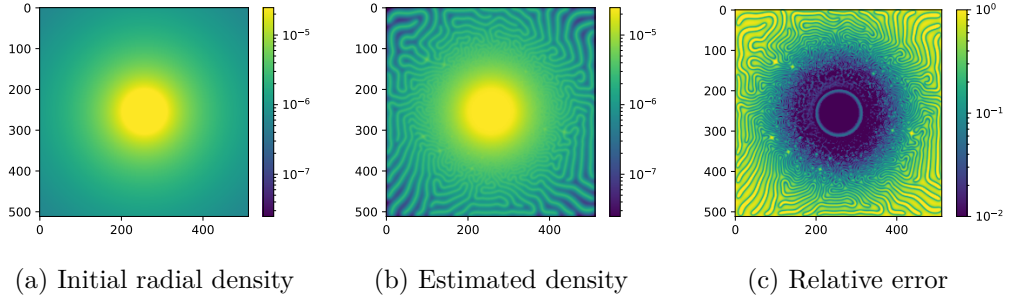


Figure 3.10: Illustration of the estimation of the density using a kernel density estimation with an exponential kernel. The colors are in log-scale for ease of comparison in the center of the k-space.

This estimation involves a kernel and it appears from numerical experiments that the kernel that yields the best estimation is the exponential. Its bandwidth parameter has been optimized manually as to recover the initial radial density. It is set to twice the size of one pixel in the Fourier domain, i.e.  $w = \frac{4\pi}{N_x}$ .

In Fig. 3.10 are given the initial density, the estimated density using the sampling points and the relative error between both. In the center of the k-space, the error made by the estimation is below 1% and on the borders of the plateau at the center it is below 6%. From this experiment, we validate this density estimation and it can be used to recover the density from a sampling scheme.

### 3.7.2 Resampling

We now use the density estimation to get a density from the optimized scheme from the trajectory optimization. Given this estimated density, a new sampling scheme is generated by solving (3.11) with a multi-scale approach and with the logarithmic kernel. Finally, the average PSNR is computed on the fastMRI validation dataset

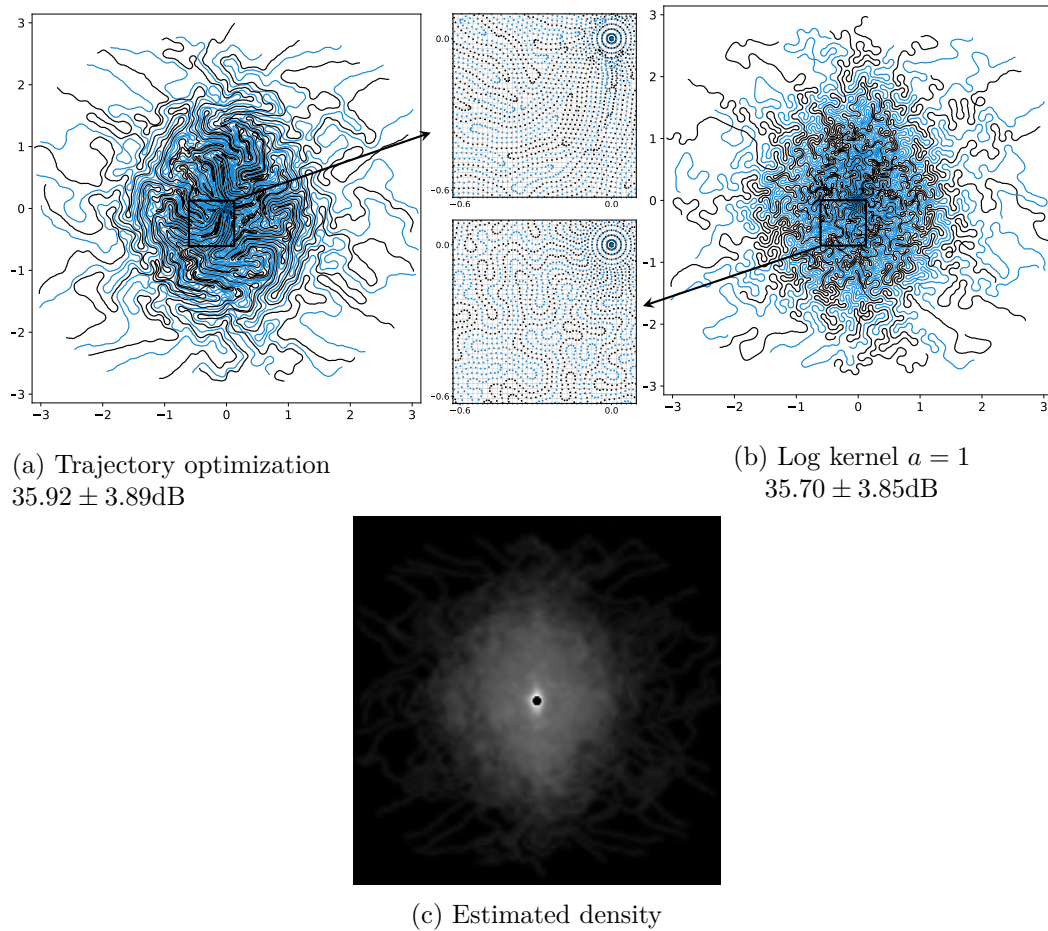


Figure 3.11: Sampling schemes resampled from a density estimated using the optimized trajectories.

with the TV reconstructor. The results are summarized in Fig. 3.11. The drop between the scheme of trajectory optimization (35.92dB) and the scheme resampled from the estimated density (35.70dB) is of only  $-0.22\text{dB}$ . We recall that the BO scheme yields an average PSNR of 35.66dB ( $-0.26\text{dB}$  in comparison to trajectory optimization). This shows that the proposed parametrization is expressive enough to allow generating densities that are as close to the density of trajectory optimization in terms of performance. However, it seems that the  $-0.22\text{dB}$  drop is due to the fact that the density parametrization cannot generate as complex trajectories as when optimizing the points position directly.

# Training Adaptive Reconstruction Networks for Blind Inverse Problems

---

**Résumé** Les réseaux de neurones ont récemment permis de résoudre de nombreux problèmes inverses mal posés avec des performances sans précédent. Les approches basées sur la physique remplacent déjà progressivement les algorithmes de reconstruction non appris dans les applications pratiques. Cependant, ces réseaux souffrent d'un défaut majeur : lorsqu'ils sont entraînés sur un opérateur donné, ils ne se généralisent pas bien à un opérateur différent. L'objectif de ce chapitre est double. Premièrement, nous montrons à travers diverses applications que l'entraînement de réseaux avec une famille d'opérateurs permet de résoudre le problème d'adaptabilité sans compromettre de manière significative la qualité de la reconstruction. Deuxièmement, nous montrons que cette procédure d'entraînement permet de s'attaquer à des problèmes inverses aveugles difficiles. Nos expériences incluent des problèmes d'échantillonnage de Fourier survenant en imagerie par résonance magnétique (IRM), en imagerie par tomographie (CT) et en défloutage d'images.

**Abstract** Neural networks have recently allowed solving many ill-posed inverse problems with unprecedented performance. Physics informed approaches already progressively replace carefully hand-crafted reconstruction algorithms in real applications. However, these networks suffer from a major defect: when trained on a given forward operator, they do not generalize well to a different one. The aim of this work is twofold. First, we show through various applications that training the network with a family of forward operators allows solving the adaptivity problem without compromising the reconstruction quality significantly. Second, we illustrate that this training procedure allows tackling challenging blind inverse problems. Our experiments include partial Fourier sampling problems arising in magnetic resonance imaging (MRI), computerized tomography (CT) and image deblurring.

This chapter is based on the preprint [Gossard 2022c]:

**Gossard, A., & Weiss, P. (2022).** Training Adaptive Reconstruction Networks for Blind Inverse Problems.



---

**Contents**

<b>4.1</b>	<b>Introduction</b>	<b>120</b>
<b>4.2</b>	<b>Related works</b>	<b>123</b>
<b>4.3</b>	<b>Preliminaries</b>	<b>126</b>
4.3.1	The forward models	127
4.3.2	The model-based reconstruction networks	128
<b>4.4</b>	<b>Training on a family of operators</b>	<b>129</b>
4.4.1	What is different?	130
4.4.2	Choosing distributions of operators	130
<b>4.5</b>	<b>Solving blind inverse problems</b>	<b>131</b>
<b>4.6</b>	<b>Numerical experiments</b>	<b>132</b>
4.6.1	The setting	132
4.6.2	The benefits of training on a family in MRI	133
4.6.3	Blind inverse problems	136
<b>4.7</b>	<b>Conclusion</b>	<b>139</b>

---

## 4.1 Introduction

The primary contribution of this paper is the design of model-based neural networks to solve *families* of blind linear inverse problems. Many sensing devices like cameras, magnetic resonance imaging (MRI) or computerized tomography (CT) systems measure a signal  $\mathbf{x} \in \mathbb{C}^N$  through a linear operator  $\mathbf{A}(\boldsymbol{\theta}) \in \mathbb{C}^{M \times N}$ . The parameter  $\boldsymbol{\theta} \in \mathbb{R}^P$  characterizes the sensing operator. For instance, it can encode the point spread function in image deblurring, the projection angles in CT or the Fourier sampling locations in MRI. This leads to measurements of the form:

$$\mathbf{y} = \mathcal{P}(\mathbf{A}(\boldsymbol{\theta})\mathbf{x}), \quad (4.1)$$

where  $\mathcal{P} : \mathbb{C}^M \rightarrow \mathbb{C}^M$  is a perturbation (e.g. additive Gaussian noise, quantization). A model based inverse problem consists in recovering an estimate  $\hat{\mathbf{x}}$  of  $\mathbf{x}$  from  $\mathbf{y}$  and  $\mathbf{A}(\boldsymbol{\theta})$ . If the parameter  $\boldsymbol{\theta}$  is unknown, then we speak of a blind inverse problem.

In this paper, we focus on neural network based reconstruction. We consider mappings of the form:

$$\begin{aligned} \mathcal{N} : \mathbb{R}^D \times \mathbb{R}^P \times \mathbb{C}^M &\rightarrow \mathbb{R}^N \\ (\mathbf{w}, \boldsymbol{\theta}, \mathbf{y}) &\mapsto \mathcal{N}[\mathbf{w}, \boldsymbol{\theta}, \mathbf{y}]. \end{aligned} \quad (4.2)$$

Given a weight  $\mathbf{w} \in \mathbb{R}^D$ , a forward operator parametrization  $\boldsymbol{\theta}$  and a measurement vector  $\mathbf{y}$ , the network  $\mathcal{N}$  outputs an estimate  $\hat{\mathbf{x}} = \mathcal{N}[\mathbf{w}, \boldsymbol{\theta}, \mathbf{y}]$ . Given a forward

operator  $\mathbf{A}(\boldsymbol{\theta}_0)$ , the traditional procedure to optimize the weights  $\mathbf{w}$ , is to minimize the following empirical risk:

$$\inf_{\mathbf{w} \in \mathbb{R}^D} \frac{1}{2I} \sum_{i=1}^I \|\mathcal{N}[\mathbf{w}, \boldsymbol{\theta}_0, \mathbf{y}_i] - \mathbf{x}_i\|_2^2, \quad (4.3)$$

where  $(\mathbf{x}_i)_{1 \leq i \leq I}$  is a collection of training images and  $(\mathbf{y}_i)$  is the corresponding collection of measurements generated using (4.1). That is, we wish the reconstruction mapping to output images close in average to the true underlying signals. In this paper, we explore a seemingly minor variation of this principle by solving:

$$\inf_{\mathbf{w} \in \mathbb{R}^D} E(\mathbf{w}) \stackrel{\text{def}}{=} \mathbb{E} \left[ \frac{1}{2I} \sum_{i=1}^I \|\mathcal{N}[\mathbf{w}, \boldsymbol{\theta}, \mathbf{y}_i] - \mathbf{x}_i\|_2^2 \right], \quad (4.4)$$

where the expectation is taken with respect to the parameter  $\boldsymbol{\theta}$  considered as a random variable. That is, we train our reconstruction mapping on a *family of operators*. The main motivation for this modification is twofold. First, we want to address a lack of adaptivity for the standard training procedure. Second, we want to use the resulting reconstruction mapping to solve blind inverse problems. Let us discuss these two points in more depth.

**The adaptivity issue** While model-based reconstruction networks provide state-of-the-art results in a large panel of applications, it is now well established that they suffer from a *lack of adaptivity*. This means that a network trained for a specific operator  $\mathbf{A}(\boldsymbol{\theta}_0)$  may have a significant performance drop if used for another operator  $\mathbf{A}(\boldsymbol{\theta}_1)$ . This drop can be evaluated as follows. Let  $\boldsymbol{\theta}_0 \neq \boldsymbol{\theta}_1$  denote two different operator parametrizations. Let  $\mathbf{y}_0 = \mathcal{P}(\mathbf{A}(\boldsymbol{\theta}_0)\mathbf{x})$  and  $\mathbf{y}_1 = \mathcal{P}(\mathbf{A}(\boldsymbol{\theta}_1)\mathbf{x})$ . Assume that  $\mathbf{w}_0^*$  and  $\mathbf{w}_1^*$  are the weights of a reconstruction network optimized for  $\mathbf{A}(\boldsymbol{\theta}_0)$  and  $\mathbf{A}(\boldsymbol{\theta}_1)$  respectively. We compare the quality of  $\mathcal{N}(\mathbf{w}_1^*, \boldsymbol{\theta}_0, \mathbf{y}_0)$  and  $\mathcal{N}(\mathbf{w}_0^*, \boldsymbol{\theta}_0, \mathbf{y}_0)$  in the second and third rows of Fig. 4.1. Observe the significant performance difference.

To avoid this pitfall, we propose to train the network by minimizing (4.4). We will carefully evaluate the performance of the resulting networks in Section 4.6 for MR image reconstruction from under-sampled data, CT imaging with limited angles and image deblurring. We conclude that this learning approach yields a reconstruction network which is significantly more stable to variations of the forward operator. In addition, the performance of an unrolled network trained on a restricted family is only marginally worse than that of a network that would be trained and used for a single operator. It therefore provides a satisfactory answer to the adaptivity issue. We also address several questions raised by our methodology. Can the unrolled network trained on a family extrapolate to unseen operators? How to sample the space of admissible operators  $\mathcal{A}$ ? What is the gain of our approach in comparison to more “universal approaches” such as plug&play priors?

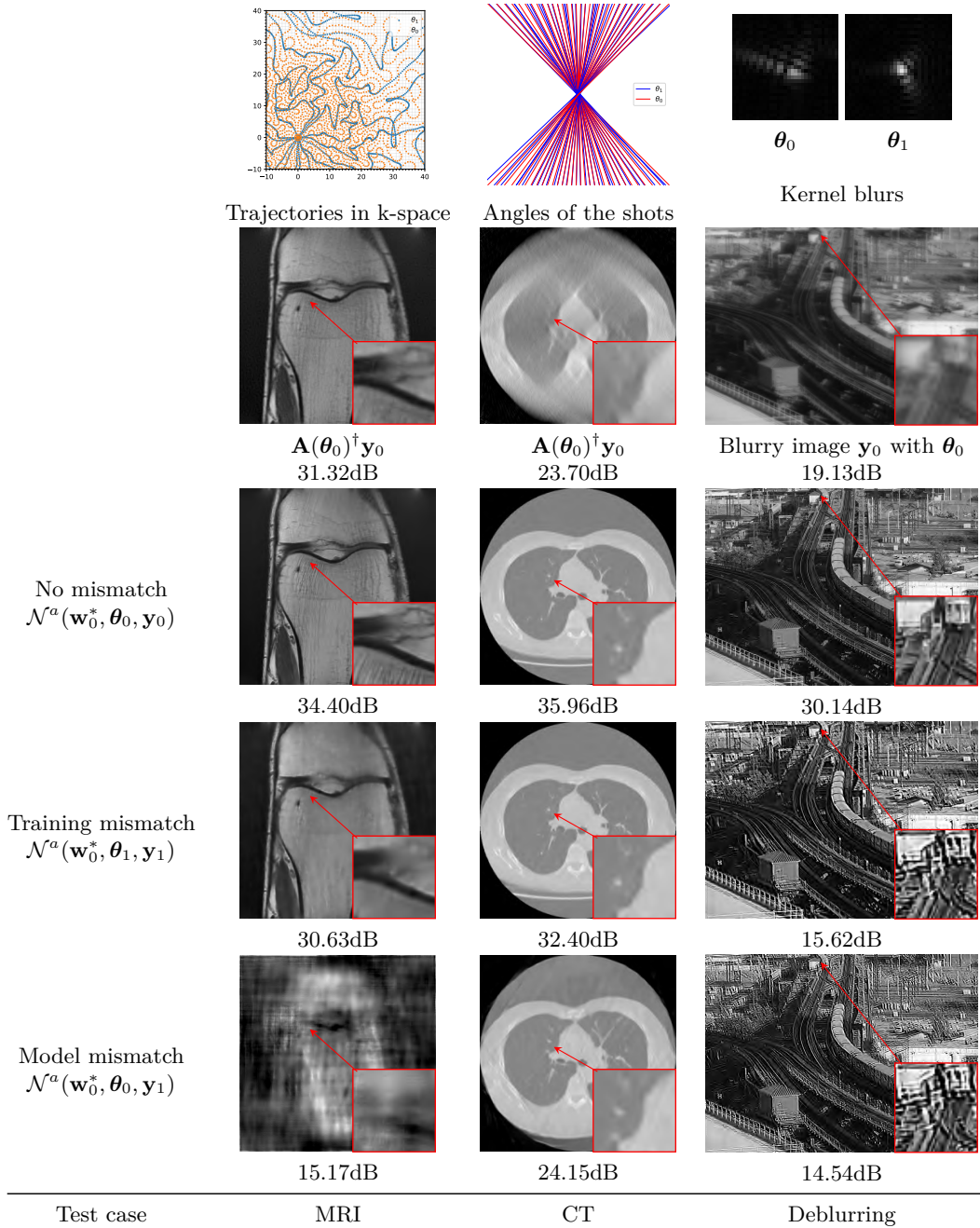


Figure 4.1: Examples of the issues addressed in this paper. *1st row*: description of the forward operators parameterized by  $\boldsymbol{\theta}_0$  and  $\boldsymbol{\theta}_1$ . *2nd row*: pseudo-inverse reconstruction of  $\mathbf{y}_0 = \mathbf{A}(\boldsymbol{\theta}_0)\mathbf{x}$ . *3rd row*: reconstruction with no model or training mismatch. *4th row*: reconstruction with a training mismatch. *Last row*: reconstruction with a model mismatch (blind). All the models are an unrolled ADMM trained on  $\mathbf{A}(\boldsymbol{\theta}_0)$ . The reconstruction PSNR is provided below each image.

**Blind inverse problems** Assume that we observe  $\mathbf{y}_1 = \mathcal{P}(\mathbf{A}(\boldsymbol{\theta}_1)\mathbf{x})$ . Unfortunately, we only have access to an approximate knowledge  $\mathbf{A}(\boldsymbol{\theta}_0)$  of the forward model. This can be due to an imprecise calibration of the sensing device or to the motion of a patient in a scanner for instance. A problem solved with a model mismatch (i.e. with the operator  $\mathbf{A}(\boldsymbol{\theta}_0)$  in place of  $\mathbf{A}(\boldsymbol{\theta}_1)$ ), can lead to catastrophic reconstruction results, as illustrated in the last row of Fig. 4.1.

The second contribution of this work is to propose a systematic approach to recover an estimate  $\hat{\boldsymbol{\theta}}_1$  of  $\boldsymbol{\theta}_1$  from the observation  $\mathbf{y}_1$ . We show that unrolled networks trained on a family of forward models provide a powerful tool to solve several blind inverse problems. The idea is simply to minimize the data consistency error

$$\hat{\boldsymbol{\theta}} \in \arg \min_{\boldsymbol{\theta} \in \Theta} \frac{1}{2} \|\mathbf{A}(\boldsymbol{\theta})\mathcal{N}[\mathbf{w}, \boldsymbol{\theta}, \mathbf{y}] - \mathbf{y}\|_2^2. \quad (4.5)$$

The reconstructed image  $\hat{\mathbf{x}} \stackrel{\text{def}}{=} \mathcal{N}[\mathbf{w}, \hat{\boldsymbol{\theta}}, \mathbf{y}]$  is defined as the output of the unrolled neural network.

This consistency principle is spread massively in the literature of blind inverse problems. The main contribution here is to plug it with a specific training procedure on a family of forward operators.

## 4.2 Related works

Let us contextualize this work.

**Regularization theory** From a historical perspective, the first inverse problem solvers were based on simple inverses or approximate inverses of  $\mathbf{A}(\boldsymbol{\theta})$ . This approach provides low quality results when the matrix  $\mathbf{A}(\boldsymbol{\theta})$  has a non trivial kernel or when the conditioning number of  $\mathbf{A}^*(\boldsymbol{\theta})$  is high. In those cases, it is critical to use regularization terms. For long ( $\sim 1960$ -2000), simple quadratic terms (Tikhonov) dominated the scientific landscape. Around 1990, a second research trend appeared with convex, nonlinear regularizers such as total variation [Rudin 1992]. This area culminated with the development of the compressed sensing theory [Candès 2006, Lustig 2005]. Starting from 2015, impressive performance gains have occurred with the advent of neural networks. They seem to be likely to replace the initial methods in a growing number of technologies [Wang 2018].

**Learned reconstruction** There are two main approaches to attack reconstruction problems using machine learning [Arridge 2019]. A first solution is *end-to-end networks* where the neural network is agnostic to the operator  $\mathbf{A}(\boldsymbol{\theta})$ . It gets trained through pairs  $(\mathbf{y}_i, \mathbf{x}_i)$  generated with the model (4.1). A popular example is AUTOMAP [Zhu 2018]. In this technology, the network needs to infer the forward model from the training data. This usually requires a huge amount of training data for large  $M$  and  $N$ .

The other possibility is *model-based* reconstruction networks that are defined as mappings of the form (4.2). They are often praised for the fact that they require less training data and benefit from a higher interpretability. Two popular approaches among this class are:

- *Denoising nets*: There, the reconstruction network performs a rough inversion followed by a denoising network such as a U-Net, to remove the remaining artifacts, see e.g. [Jin 2017].
- *Unrolled nets*: Many efficient iterative methods have been developed to solve convex optimization problems (proximal gradient descent, Douglas-Rachford, ADMM, Primal-Dual, ...) [Combettes 2011]. They have the general form:

$$\mathbf{x}_{k+1} = \text{prox}_R(M(\mathbf{A}(\boldsymbol{\theta}), \mathbf{y}, \mathbf{x}_k)), \quad (4.6)$$

for  $k = 1$  to  $K \in \mathbb{N}$ . The mapping  $M$  is linear and can be interpreted as a crude way to invert the operator, in the sense that  $\mathbf{A}(\boldsymbol{\theta})M(\mathbf{A}(\boldsymbol{\theta}), \mathbf{y}, \mathbf{x}_k) \simeq \mathbf{y}$ . The term  $\text{prox}_R$  can be interpreted as a way to regularize (denoise) the remaining artifacts. The so-called plug&play priors [Venkatakrishnan 2013] fit in this category.

The unrolled networks draw their inspiration from (4.6). They consist in replacing the handcrafted or learned proximal operator  $\text{prox}_R$  by a sequence of neural networks  $(\mathcal{N}_k[\mathbf{w}_k])_{1 \leq k \leq K}$  promoting an output  $\mathbf{x}_K$  similar to the training images. The difference with the plug&play priors is that the weights  $\mathbf{w}_k$  are trained specifically for a given operator  $\mathbf{A}$ . Examples of approaches in this category include [Sun 2016, Diamond 2017, Adler 2017, Zhang 2018, Dong 2018, Adler 2018, Aggarwal 2018, Hammernik 2019, Li 2019b]. These algorithms are currently among the most efficient for MRI reconstruction [Muckley 2021].

For completeness, let us mention that a popular alternative consists in synthesizing the images  $\mathbf{x}$  with generative models [Bora 2017, Asim 2020b]. Compared to the approaches mentioned above, it typically suffers from a higher computational cost. Indeed, a gradient descent in the latent space needs to be performed. Hence, we will not consider this approach further in this work.

**Adaptivity** Neural network reconstructions can suffer from severe instabilities. This issue was notably discussed in [Antun 2020], where it was shown that well chosen additive noise (an adversarial attack) or modifications of the forward operator could lead to disastrous hallucinations for some specific architectures. This problem was studied with care in [Genzel 2022b]. There, the authors have shown that careful training procedures could fix this issue and yield robust and state-of-the-art reconstruction results.

A paper closely related to our work is [Gilton 2021b]. The authors study the same robustness issue to model mismatches. The authors propose two distinct

algorithmic approaches to attack it. The first one is called *parametrize & perturb* by the authors. It suffers from an important drawback, which is the need to re-optimize the network weights for every new operator. It can therefore be slow at runtime and we do not compare it in this paper. The other approach is called *Reuse & Regularize (R&R)*. It consists in training a network for a given operator  $\mathbf{A}(\boldsymbol{\theta}_0)$ , and then use this network for another operator  $\mathbf{A}(\boldsymbol{\theta}_1)$ . This is done in an iterative procedure, accounting for the data consistency term  $\|\mathbf{A}(\boldsymbol{\theta}_1)\mathbf{x} - \mathbf{y}\|_2^2$ . The approach we propose in this paper is significantly lighter at run-time, since we just train the network with a family of operators.

An older and popular alternative consists in replacing the proximal operator in (4.6) by a denoiser. This approach is often called a *plug&play (P&P)* prior [Venkatakrisnan 2013]. It was first used with hand-crafted priors [Gu 2014] and a significant performance boost occurred with the use of pre-trained neural networks among which we can cite [Ryu 2019, Zhang 2021b]. This approach has the huge asset of adapting painlessly to arbitrary inverse problems. We propose some comparisons and discuss the pros and cons of each approach in Section 4.6.

**Blind inverse problems** Blind inverse problems are spread massively in applications and it is impossible to provide a comprehensive overview of the existing works. The review papers [Kundur 1996, Campisi 2017] provide a good idea of the wealth of results for the sole field of blind deconvolution and super-resolution.

A possibility is to design a two-step method. First an estimate of the forward operator is built. Second, this estimate is used in conjunction with the methods from the previous section. In some cases, it is possible to exploit some redundancy in the data to estimate the operator parameters. This is the case in parallel Magnetic Resonance Imaging (MRI), where the coil sensitivity maps can be estimated using only the low frequencies [Sodickson 1997, Pruessmann 1999, Griswold 2002]. When no redundancy is available, estimating the operator can be achieved by minimizing the discrepancy between the statistics of the acquired measurement and the statistics of the measurements generated by applying an operator to a “natural” signal. A good example in blind deblurring is the Goldstein-Fattal approach [Goldstein 2012], which analyzes the power spectrum of the blurry image. Recently, a few authors proposed to build an identification network that learns to identify the blur kernel [Schuler 2015] or a blur parametrization [Sun 2015, Yan 2016, Chakrabarti 2016, Debarnot 2022] from the blurry-noisy image. While this approach is cheap computationally, it requires an application specific design and we will not consider it further in this work.

One of the most popular alternatives consists in minimizing a combination of a data fidelity term and a regularizing prior. This can be addressed through an alternate minimization between the image and the operator parametrization. Most of the literature suggests the use of hand-crafted priors on the unknown operator or on the image to recover (see e.g. [Chan 1998, Fergus 2006, Krishnan 2009, Krishnan 2011, Xu 2013, Ahmed 2013, Pan 2014, Michaeli 2014,

Pan 2016, Ren 2016, Bai 2018, Ljubenočić 2019, Chen 2019, Zhang 2022] for blind deblurring, or [Riis 2021, Wang 2022b] in CT imaging).

While these approaches can provide excellent results, they are likely to be outperformed by neural network based approaches in a near future. Indeed, impressive performance has already been reached recently thanks to neural network based regularizers. Different strategies have been suggested, going from untrained networks (see [Bostan 2020] for an application in optics), generative models (see [Asim 2020b] for an application in blind deblurring), or unrolled networks (see [Lecouat 2022, Lecouat 2021] for an application to super-resolution from an image sequence).

The method advocated in our paper is close in spirit to the works in this latest category. It differs in the way the training is performed. Here, we first train an unrolled network on a family of forward operators, which allows fixing the weights once for all. We then minimize (4.5) in the space of parameters of the forward model. This methodology has various advantages:

- Compared to untrained networks [Bostan 2020], the method does not optimize the network weights to solve the problem, which is typically quite heavy computationally. It is therefore faster at runtime. In addition, it is adapted to a clearly defined image dataset.
- Methods based on generative models [Bora 2017, Asim 2020b] may suffer from a significant drawback: the produced images necessary live in the range of the generator. To avoid this issue, a possibility is to add hand-crafted regularization terms such as total variation that allow extending the span of possible images [Asim 2020b].
- In [Lecouat 2022, Lecouat 2021], the neural network weights are trained directly to solve the blind inverse problem. This significantly limit the number of weights and iterations within the iterative procedure. In this paper, we propose to train the network beforehand, allowing to use arbitrary solvers and as many iterations as desired to find the parameter  $\boldsymbol{\theta}$ .

### 4.3 Preliminaries

In this paper, we consider forward models

$$\mathbf{y} = \mathbf{A}(\boldsymbol{\theta})\mathbf{x} + \mathbf{b} \quad (4.7)$$

where  $\mathbf{A}(\boldsymbol{\theta}) \in \mathbb{K}^{M \times N}$  is a linear mapping either real ( $\mathbb{K} = \mathbb{R}$ ) or complex ( $\mathbb{K} = \mathbb{C}$ ). In our experiments, we consider additive white Gaussian noise (complex for MRI)  $\mathbf{b} \sim \mathcal{N}(0, \sigma^2 \text{Id})$ . The dependency of  $\mathbf{A}$  with respect to its parameter  $\boldsymbol{\theta}$  can be linear or nonlinear. We let  $N \in \mathbb{N}$  denote the number of pixels of the image  $\mathbf{x}$  with  $N = N_x \times N_y$  for 2D images and  $M$  is the number of measurements.

### 4.3.1 The forward models

To illustrate our problem, we consider three important biomedical applications: parallel magnetic resonance imaging, computerized tomography and microscopy/astronomy. Let us describe these applications more precisely.

**Parallel Magnetic Resonance Imaging** Our aim here is to reconstruct images from under-sampled Fourier samples with unknown sensitivity maps associated to  $J \in \mathbb{N}$  reception coils, and with inaccurate trajectories. The parameter  $\boldsymbol{\theta}$  can be decomposed as  $\boldsymbol{\theta} = (\boldsymbol{\tau}, \boldsymbol{\omega})$ , where  $\boldsymbol{\tau}$  is the parameter describing the sensitivity maps and  $\boldsymbol{\omega}$  describes a perturbation of the sampling locations. To the best of our knowledge, these two problems have not been treated jointly in the literature yet.

Let  $\mathcal{F}(\boldsymbol{\xi})$  denote the non-uniform Fourier transform (NUFT) [Potts 2001] at frequencies  $\boldsymbol{\xi}$ , defined by

$$[\mathcal{F}(\boldsymbol{\xi})]_{m,n} = e^{-i\langle \mathbf{p}_n, \boldsymbol{\xi}_m \rangle}$$

where  $(\mathbf{p}_n)_{1 \leq n \leq N}$  is a set of 2D positions on a grid. We construct a family of forward operators  $\mathcal{A} = \{\mathbf{A}(\boldsymbol{\xi}, \boldsymbol{\theta}), \boldsymbol{\xi} \in \Xi, \boldsymbol{\theta} \in \Theta\}$ , where  $\Xi \subset \mathbb{R}^{2 \times M}$  is a set of 2D sampling schemes with  $M$  sampling points. The parameter space  $\Theta = \mathcal{T} \times \Omega$  describes the set of admissible parameters for the sensitivity maps  $\mathcal{T}$  and for the perturbation  $\Omega$ . The measured signal  $\mathbf{y} = (\mathbf{y}^{(1)}, \dots, \mathbf{y}^{(J)})$  is acquired through  $J$  coils. The  $m$ -th measurement acquired by the  $j$ -th coil is defined by

$$y_m^{(j)} = [\mathbf{A}(\boldsymbol{\xi}, \boldsymbol{\theta})\mathbf{x}]_{m,j} + \mathbf{b}_{m,j} = \left[ \mathcal{F}(\mathbf{h}(\boldsymbol{\omega}) \star \boldsymbol{\xi}) \left( \mathbf{x} \odot \mathbf{s}(\boldsymbol{\tau}^{(j)}) \right) \right]_m + \mathbf{b}_{m,j}. \quad (4.8)$$

The mapping  $\mathbf{s} : \boldsymbol{\tau}^{(j)} \in \mathbb{R}^T \mapsto \mathbf{s}(\boldsymbol{\tau}^{(j)}) \in \mathbb{C}^N$  parametrizes the coil sensitivity maps. Since the sensitivity maps are smooth, we use a parametrization based on thin plate splines (TPS) [Duchon 1977]. The total number of parameters that encode the sensitivity map is  $T = 104$ . It consists of the TPS coefficients using  $7 \times 7$  regularly spaced control points plus the coefficients of a first degree polynomial. This has to be multiplied by two for the real and imaginary parts.

Following [Vannesjo 2016, Dietrich 2016], we assume that the trajectory  $\boldsymbol{\xi}$  is perturbed by a convolution with an impulse response  $\mathbf{h}(\boldsymbol{\omega})$ . The symbol  $\star$  in (4.8) corresponds to a discrete convolution. Evaluating the convolution filter  $\mathbf{h}(\boldsymbol{\omega})$  is known as a challenging problem that can be addressed with (expensive) field cameras [Dietrich 2016]. Here, in the spirit of [Vannesjo 2016], we will rather treat it as a blind inverse problem. We parametrize  $\mathbf{h}$  as a linear combination of the form  $\mathbf{h}(\boldsymbol{\omega}) = \sum_{o=1}^O \omega_o \mathbf{h}_o$ , where  $(\mathbf{h}_o)_{1 \leq o \leq O}$  is an orthogonal basis. In practice, we simply use compactly supported filters of size  $O = 32$  and  $(\mathbf{h}_o)_{1 \leq o \leq O}$  corresponds to the first 32 elements of the canonical basis.

**Computerized Tomography** Our aim is to reconstruct images from parallel beam computerized tomography. The parameter  $\boldsymbol{\theta}$  describes the projection angles (or the patient motion).



We assume that the CT scan uses parallel beams and that it performs  $J$  acquisitions with a receptor that has  $M$  sensors, resulting in  $J \times M$  measurements. In this application, the parameter  $\boldsymbol{\theta} = (\boldsymbol{\alpha}, \mathbf{s})$  represents the angles  $\boldsymbol{\alpha} \in \mathbb{R}^J$  and the shifts at the origin  $\mathbf{s} \in \mathbb{R}^J$  that describe the beams trajectories. If  $m$  corresponds to the  $m$ -th pixel of the receptor and if we index the acquisitions by  $1 \leq j \leq J$ , we get

$$y_m^{(j)} = \iint_{\Omega} \mathbf{x}(u_x, u_y) \delta_{u_x \cos(\alpha_j) + u_y \sin(\alpha_j) = \mathbf{p}_m + \mathbf{s}_j} du_x du_y + b_m^{(j)}, \quad (4.9)$$

with  $\Omega = [-N_x/2, N_x/2] \times [-N_y/2, N_y/2]$  and  $\mathbf{p} = \llbracket -M/2, \dots, M/2 - 1 \rrbracket$ . A perfect model would correspond to  $\boldsymbol{\alpha}$  being equispaced angles and  $\mathbf{s} = \mathbf{0}$ . The forward model can be computed using the Fourier slice theorem. This corresponds to performing a 2D NUFT and we resort to the same library used as for MRI (see <https://github.com/albangossard/Bindings-NUFFT-pytorch>).

**Deblurring in optics** In this application, we wish to solve problems appearing in optics, especially microscopy or astronomy. The parameter  $\boldsymbol{\theta}$  describes the point spread function through the theory of diffraction. The acquisition model in this application simply reads

$$\mathbf{y} = \mathbf{h} \star \mathbf{x} + \mathbf{b}, \quad (4.10)$$

where  $\mathbf{h}$  is the kernel blur. We consider blurs generated by Fresnel diffraction theory [Goodman 1996]. The kernel blur is parametrized by a vector  $\boldsymbol{\theta} \in \mathbb{R}^7$  and the convolution kernel is expressed as

$$\mathbf{h}(\boldsymbol{\theta}) = c \left| \int_{\|w\|_2 \leq f_c} \exp \left( 2i\pi \left[ \sum_{k=1}^K \theta_k Z_k + \langle \mathbf{u}, \mathbf{w} \rangle \right] \right) dw_b \right|^2. \quad (4.11)$$

In this expression,  $f_c$  is a cutoff frequency and  $c$  is a scaling parameter such that  $\|\mathbf{h}\|_1 = 1$ . The expansion  $\sum_{k=1}^K \theta_k Z_k$  describes the pupil function of an objective. The functions  $Z_k$  are Zernike polynomials and the vector  $\boldsymbol{\theta}$  therefore parametrizes the pupil function.

### 4.3.2 The model-based reconstruction networks

We consider three convolutional neural networks all based on a fixed convolutional neural network architecture  $\mathcal{D}$  which is a DruNet network [Zhang 2021b]. This network is the current state-of-the-art when used within plug&play algorithms. One of its important assets is its ability to accommodate for different noise levels. The idea is to set one of the input channels as a constant image with a value equal to the standard deviation of the noise.

### 4.3.2.1 Denoising network

The denoising network is denoted  $\mathcal{N}^d$ . It is of the form

$$\mathcal{N}^d[\mathbf{w}, \boldsymbol{\theta}, \mathbf{y}] = \mathcal{D}[\mathbf{w}, \mathbf{A}(\boldsymbol{\theta})^\dagger(\mathbf{y})].$$

### 4.3.2.2 Unrolled proximal gradient network

Letting  $F(\mathbf{x}) = \frac{1}{2}\|\mathbf{A}(\boldsymbol{\theta})\mathbf{x} - \mathbf{y}\|_2^2$ , the unrolled proximal gradient descent takes the sequential form:

$$\begin{aligned} \mathbf{x}_0 &= \mathbf{A}(\boldsymbol{\theta})^\dagger \mathbf{y} \\ \mathbf{x}_{k+1} &= \mathcal{D}[\mathbf{w}_k, \mathbf{x}_k - \gamma \nabla F(\mathbf{x}_k)]. \end{aligned}$$

where  $\gamma = \frac{1}{\|\mathbf{A}(\boldsymbol{\theta})\|_{2 \rightarrow 2}^2}$  is a step-size. The reconstruction network runs for  $K$  iterations and is denoted  $\mathcal{N}^p : (\mathbf{w}, \boldsymbol{\theta}, \mathbf{y}) \mapsto \mathbf{x}_K$ .

### 4.3.2.3 Unrolled ADMM

It takes the form (see e.g. [Sun 2016]):

$$\begin{aligned} \mathbf{x}_0 &= \mathbf{A}(\boldsymbol{\theta})^\dagger \mathbf{y} \quad \text{and} \quad \boldsymbol{\mu}_0 = 0 \\ \mathbf{z}_{k+1} &= [\mathbf{A}(\boldsymbol{\theta})^* \mathbf{A}(\boldsymbol{\theta}) + \beta \text{Id}]^{-1} (\mathbf{A}(\boldsymbol{\theta})^* \mathbf{y} + \beta \mathbf{x}_k - \boldsymbol{\mu}_k) \\ \mathbf{x}_{k+1} &= \mathcal{D} \left[ \mathbf{w}_k, \mathbf{z}_{k+1} + \frac{\boldsymbol{\mu}_k}{\beta} \right] \\ \boldsymbol{\mu}_{k+1} &= \boldsymbol{\mu}_k + \beta (\mathbf{z}_{k+1} - \mathbf{x}_{k+1}). \end{aligned}$$

This sequence runs for  $K$  iterations and the result is denoted  $\mathcal{N}^a : (\mathbf{w}, \boldsymbol{\theta}, \mathbf{y}) \mapsto \mathbf{x}_K$ . The parameter  $\beta$  is a penalty parameter, which is fixed in our experiments.

In the two unrolled algorithms, the weights  $\mathbf{w}$  to be trained are  $\mathbf{w} = [\mathbf{w}_0, \dots, \mathbf{w}_{K-1}]$  and they are not shared across iterations.

## 4.4 Training on a family of operators

Traditionally, networks are trained by minimizing the empirical risk for a given forward model  $\boldsymbol{\theta}_0$ . Our first contribution is to train *end-to-end* networks by minimizing the risk over a set of forward operators:

$$\inf_{\mathbf{w} \in \mathbb{R}^D} \frac{1}{2} \mathbb{E} \|\mathcal{N}[\mathbf{w}, \boldsymbol{\theta}, \mathbf{y}] - \mathbf{x}\|_2^2, \quad (4.12)$$

where the expectation is taken with respect to the noise  $\mathbf{b}$ , to the images  $\mathbf{x}$  and to the forward models  $\boldsymbol{\theta}$ .

#### 4.4.1 What is different?

Let us provide a rough theoretical explanation of the difference between a training on a single operator and an operators family. To this end, we focus on the simplest denoising network. The pseudo-inverse  $\mathbf{A}(\boldsymbol{\theta})^\dagger$  applied to  $\mathbf{y} = \mathbf{A}(\boldsymbol{\theta}) + \mathbf{b}$  yields a vector  $\hat{\mathbf{x}}$  of the form

$$\hat{\mathbf{x}} = \mathbf{x} + \mathbf{k} + \mathbf{n}, \quad (4.13)$$

where  $\mathbf{k} \in \ker(\mathbf{A}(\boldsymbol{\theta}))$  and  $\mathbf{n}$  is a correlated Gaussian noise living in  $\text{ran}(\mathbf{A}(\boldsymbol{\theta}))$ . Hence the denoising network  $\mathcal{D}$  serves two purposes: 1) recover the missing data  $\mathbf{k}$  in the kernel of  $\mathbf{A}(\boldsymbol{\theta})$  and 2) remove the correlated noise  $\mathbf{n}$ . Each of these two tasks is clearly highly dependent on  $\boldsymbol{\theta}$ . If trained with a single scheme, the network may get specialized very well for these specific statistical patterns and not extrapolate to other ones. We will explore in the numerical section 4.6.2, whether a training with a larger variety of operators mitigates the performance drop.

#### 4.4.2 Choosing distributions of operators

In this section, we focus on designing distributions for  $\boldsymbol{\theta}$ , for the different applications listed above.

##### 4.4.2.1 Magnetic Resonance Imaging

In this modality, the family of forward operators is constructed by considering different sampling schemes, sensitivity maps and trajectory perturbations.

**Sampling schemes** We propose to generate random sampling schemes  $\boldsymbol{\xi}$  following the ideas from [Boyer 2016, Chauffert 2017, Lazarus 2019]. The principle is to design a scheme that fits a target probability measure  $\rho : \mathbb{R}^2 \rightarrow \mathbb{R}_+$ . To this end, we define

$$\boldsymbol{\xi}(\rho) \stackrel{\text{def}}{=} \arg \min_{\boldsymbol{\xi} \in \Xi} \text{dist} \left( \frac{1}{M} \sum_{m=1}^M \delta_{\boldsymbol{\xi}_m}, \rho \right), \quad (4.14)$$

where  $\text{dist}$  is a discrepancy between probability measures and  $\Xi \subseteq \mathbb{R}^{2 \times M}$  is a set that describes the admissible trajectories from the scanner.

Following [Gossard 2022a], we generate random target densities  $\rho$  as anisotropic power decaying distributions. They are parametrized by a random vector  $\boldsymbol{\lambda}$  that encodes the density at origin, the anisotropy and the power decay law. To avoid solving (4.14) at training time, we have pre-computed 1000 sampling patterns. The corresponding vectors  $\boldsymbol{\lambda}$  have been generated by using a max-min sampling (see [Pronzato 2017, Debarnot 2022]) of a set of an admissible set of parameters  $\Lambda$ . We refer to [Gossard 2022a] for more details. Examples of densities and sampling patterns  $\boldsymbol{\xi}(\rho(\boldsymbol{\lambda}))$  are displayed in Fig. 4.2a-4.2e.

**Sensitivity maps** As for the sensitivity maps, we used real estimates generated using the fastMRI database [Zbontar 2018]. We first estimate them using a stan-

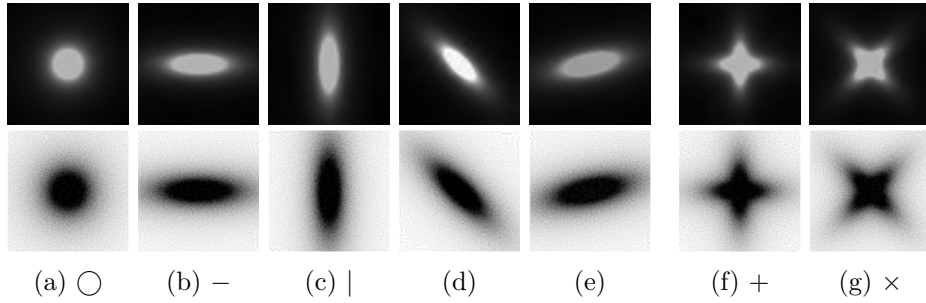


Figure 4.2: Densities (top) and corresponding sampling schemes (bottom). Fig. 4.2a, 4.2b, 4.2c, 4.2d and 4.2e belong to the family  $\mathcal{A}$ . Fig. 4.2f and 4.2g (crosses) do not. Notice that the sampling patterns are diverse with significant differences from one to the other.

dard approach [Griswold 2002] and then project the estimates onto the span of the proposed parametrization with thin plate splines. At training time, they are associated to the corresponding training pairs.

**Trajectories filtering** We did not include the trajectory perturbation effect (convolution with  $\mathbf{h}(\boldsymbol{\omega})$ ) at training time.

#### 4.4.2.2 Computerized tomography

In this modality, we assume that the distribution of projection angles follows a uniform distribution centered on a vector of regularly spaced angles  $\boldsymbol{\alpha}_0 = (-\pi/4, -\pi/4 + \pi/J, \dots, \pi/4)$  (see the red lines in first row of Fig. 4.1) and shift at origin  $\mathbf{s}_0 = 0$ . We only sample the angles in the range  $[-\pi/4, \pi/4]$  to account for the missing cone in computerized tomography.

Hence, we have  $\boldsymbol{\alpha} = \boldsymbol{\alpha}_0 + \boldsymbol{\alpha}_\delta$  with  $\boldsymbol{\alpha}_\delta \sim \mathcal{U}([-1.37^\circ, 1.37^\circ]^J)$  and the random shifts are  $\mathbf{s} \sim \mathcal{U}([-2, 2]^J)$ . These perturbations may reflect movements of the patient inside the scanner during the scan.

#### 4.4.2.3 Deblurring

In this application, we vary the blur kernel by changing only the 4-th to the 10-th Zernike polynomials. We set  $\theta_1 = \theta_2 = \theta_3 = 0$  in (4.11) and we let the coefficients  $\theta_4$  to  $\theta_{10}$  follow a uniform distribution in  $[-0.15, 0.15]$ .

## 4.5 Solving blind inverse problems

After the training procedure on a family of operators, we get a reconstruction mapping  $\mathcal{N}[\mathbf{w}^*, \boldsymbol{\theta}, \mathbf{y}]$ . Now, if  $\mathbf{y} = \mathcal{P}(\mathbf{A}(\bar{\boldsymbol{\theta}})\mathbf{x})$ , with an unknown parameter  $\bar{\boldsymbol{\theta}}$ , we propose to solve the optimization problem (4.5). As the function in (4.5) is deterministic over a small to moderate dimension, we can opt for basically any

standard optimization routine. The use of automatic differentiation techniques available in PyTorch or Tensorflow allows us to compute the Jacobian of  $\mathcal{N}[\mathbf{w}^*, \boldsymbol{\theta}, \mathbf{y}]$  with respect to the parameter  $\boldsymbol{\theta}$ . Possible optimization routines include:

- The L-BFGS optimizer [Liu 1989]. This quasi-Newton method estimates the Hessian of the function using first order information only and is known to converge rapidly when initialized close to a (local) minimizer. It therefore seems particularly adapted when the user has a good knowledge of the true parameter  $\bar{\boldsymbol{\theta}}$ .
- The RMSProp or ADAM optimizers [Tieleman 2012, Kingma 2015]. For the computerized tomography and blind deblurring problems, we observed significant issues with a convergence to bad local minimizers only. To avoid this phenomenon, a possibility is to resort to inertial methods, which are known to escape narrow basins of attraction. In our experiments, we used the RMSProp optimizer with a parameter  $\beta = 0.9$  (Adam with  $\beta_1 = 0$ ).
- Global optimization. If it turns out that the cost function is too chaotic, the gradient of the objective function does not provide a meaningful information on the location of the global minimizer. We can then resort to 0-th order methods such as Bayesian optimization [Frazier 2018a].

As is, the prior on the forward model is encoded by the physics of the acquisition system through the mapping  $\mathbf{A}(\cdot)$ . Hence, we add no regularization term on  $\boldsymbol{\theta}$ . It would also be possible to add one to promote specific solutions. This is the standard approach in Bayesian estimation. We did not explore this idea in this paper.

## 4.6 Numerical experiments

The numerical experiments are divided in two sections. In the first section 4.6.2 we compare the benefits and drawbacks of training model-based networks on a family of operators. We consider the training of a denoising network and of an unrolled proximal gradient network for MR image reconstruction. The experiments are conducted in a simplified framework where the constraints and the sensitivity maps are not taken into account.

In the second section 4.6.3, we illustrate that training model-based networks on a family of operators allows solving blind inverse problems. The experiments are carefully conducted on the three applications: MRI, CT and image deblurring with an unrolled ADMM [Sun 2016] with  $K = 5$  iterations. The MRI experiments in this part take into account both the constraints and the sensitivity maps.

### 4.6.1 The setting

All the models were trained using the Adam optimizer in PyTorch with the default parameters except the learning rate which was tuned for each experiment.

**Magnetic Resonance Imaging** The training database is the fastMRI knee training dataset [Zbontar 2018]. It contains 34,742 images of size  $320 \times 320$ . All evaluations were performed on the validation set of the fastMRI knee database containing 7,135 2D slices. We used the efficient cuFINUFFT transform [Shih 2021], which is the fastest available library in our experiments (see <https://github.com/albangossard/Bindings-NUFFT-pytorch> for comparisons).

For the experiments illustrating the advantages of training on a family of operators, we set  $M = N/4$ , i.e. a 4x downsampling rate. We used a single reception coil ( $J = 1$ ) with a known sensitivity map  $\mathbf{s} = 1$ . The denoising network  $\mathcal{N}^d$  was trained on 30 epochs with a learning rate of  $10^{-3}$  and an exponential step decay of 0.95 after each epoch. The unrolled network  $\mathcal{N}^p$  uses  $K = 10$  iterations and it was trained on 14 epochs with a learning rate of  $10^{-4}$  and an exponential step decay of 0.95 after each epoch. Both trainings took about 24h on an Nvidia V100, resulting in a total energy of  $\sim 70\text{kWh}$ .

The blind reconstruction experiments are conducted with  $M = N/10$  measurements and  $J = 15$  reception coils. The networks are trained for 8 epochs with a learning rate of  $10^{-4}$  and with an exponential step decay of 0.95 after each epoch.

**Computerized Tomography** The dataset used is the Lung Image Database Consortium [Armato III 2011] which has a total of 244,527 slices. We divided this dataset into a training and a validation dataset (80% and 20% respectively). As the blind inverse problem (4.5) requires differentiating the operator  $\mathbf{A}(\boldsymbol{\theta})$  with respect to its parameters  $\boldsymbol{\theta}$ , we cannot use standard GPU-based libraries to compute the Radon transform [Ronchetti 2020]. We thus resorted to an homemade implementation that relies on a NUFT through the Fourier slice theorem. In order to reduce the important numerical cost and energy consumption of the experiments with CT reconstruction, we downsized the images to  $256 \times 256$ .

**Deblurring** The image deblurring experiments use the MS COCO dataset [Lin 2014] (118,287/5,000 images for training/validation). During training we randomly cropped patches of size  $400 \times 400$  to accelerate the computation.

## 4.6.2 The benefits of training on a family in MRI

In this section, we show the advantages and drawbacks of training a reconstruction network on a family of operators. In this version of the work, we only display the results related to MRI with a single coil. We plan to conduct similar experiments for all imaging modalities for the final version of this work.

### 4.6.2.1 Training on fixed sampling schemes

In this section, we trained the two reconstruction networks (the denoising network  $\mathcal{N}^d$  and the unrolled proximal gradient descent  $\mathcal{N}^p$ ) on 5 different schemes: a radial one ( $\circ$ , Fig. 4.2a), an horizontal one ( $-$ , Fig. 4.2b) and a vertical one ( $|$ , Fig. 4.2c).

In addition, we used two crosses, which do not belong to the training family  $\mathcal{A}$ . The first one is aligned with the axes (+, Fig. 4.2f) and the other one with the diagonals ( $\times$ , Fig. 4.2g). In Table 4.1, we report the average peak signal-to-noise ratio, on the validation set for the two architectures. Table 4.1 illustrates various facts listed below.

**Lack of adaptivity** Without surprise, the values on the diagonal are higher than the off-diagonal terms. This means that the best way to reconstruct images for a given scheme is to train the network for this specific scheme. The drop of peak signal-to-noise ratio (PSNR) when using a network trained with the wrong operator can be as high as 9dB for the denoising net and 5dB for the unrolled net (see the pairs – and |). This is a striking illustration of the strong dependency of a reconstruction net to the operator used at the training stage. The corresponding images are shown in Fig. 4.3.

**The superiority of unrolled nets** The unrolled network provides better reconstruction results than the denoising net. The overall gain on the diagonal varies between 1.4dB and 1.7dB for this particular application, which is significant. This is in accordance with recent comparisons of both strategies [Muckley 2021].

**Partial adaptivity** While the training on the vertical scheme | provides catastrophic results when used on the horizontal scheme –, the networks trained on the radial  $\bigcirc$  scheme provide rather good results uniformly on the 4 other sampling schemes. Indeed, we see that the performance drop when used on a distinct scheme raises up to 3.6dB for the denoising net and 2.7dB for the unrolled network. Overall, the unrolled network provides a significantly better alternative when it comes to adaptivity.

**Optimal sampling scheme** Some sampling schemes make the reconstruction easier than others, which is in accordance with the compressed sensing theory. The knee images have large vertical and horizontal edges. This seems to favor the + sampling scheme, which was already observed in other works [Wang 2022a, Weiss 2021, Gossard 2022a].

#### 4.6.2.2 Training on an operator family

In this section, we trained the reconstruction networks by varying the forward operators, as in (4.12). In what follows, we let ID and IU denote the “ideal” denoising network and the “ideal” unrolled network respectively. By ideal, we mean that the networks have been trained and tested with the same operator. They serve as a benchmark that cannot be outperformed. We let FD and FU denote the “family” denoising and unrolled networks, which have been trained over a complete family. We also tested the plug&play approach. We used an unrolled proximal gradient for  $K = 10$  iterations with a DruNet network as an embedded denoiser. It

		Train					Train				
		○	-		+	×	○	-		+	×
Evaluation	○	36.30 ±4.30	31.54 ±2.62	33.15 ±2.91	35.52 ±3.71	35.19 ±3.58	38.04 ±5.13	37.17 ±4.60	36.72 ±4.48	37.91 ±5.05	37.87 ±5.02
	-	32.93 ±2.79	35.43 ±3.96	28.27 ±2.54	33.45 ±3.01	31.32 ±2.62	35.93 ±4.10	37.09 ±4.64	32.97 ±2.60	35.65 ±4.03	35.38 ±3.64
		32.68 ±2.84	26.71 ±2.50	36.20 ±4.20	34.31 ±3.20	30.15 ±2.62	35.37 ±4.09	32.13 ±3.06	37.61 ±4.82	37.09 ±4.53	36.50 ±4.18
	+	35.36 ±3.65	29.76 ±2.57	32.10 ±2.94	36.34 ±4.28	32.83 ±2.74	37.77 ±4.92	36.09 ±4.09	36.41 ±4.02	37.98 ±5.03	37.32 ±4.60
	×	35.21 ±3.63	32.77 ±2.72	33.01 ±2.75	33.77 ±2.95	36.04 ±4.19	37.56 ±4.90	36.96 ±4.47	35.74 ±4.17	36.96 ±4.50	37.74 ±4.98

(a) Denoising net

(b) Unrolled net

Table 4.1: Average PSNR and its standard deviation evaluated on the fastMRI validation dataset for the two architectures.

		Model					
		ID	FD	IU	FU	P&P	R&R
Evaluation	○	36.30 ±4.30	36.03 ±4.10	38.04 ±5.13	38.00 ±5.09	35.06 ±3.53	35.20 ±3.56
	-	35.43 ±3.96	35.06 ±3.74	37.09 ±4.64	36.97 ±4.57	33.95 ±3.25	32.88 ±2.92
		36.20 ±4.20	35.89 ±3.99	37.61 ±4.82	37.52 ±4.77	35.12 ±3.54	34.21 ±3.23
	+	36.34 ±4.28	35.35 ±3.71	37.98 ±5.03	37.88 ±4.96	35.18 ±3.56	34.13 ±3.17
	×	36.04 ±4.19	35.18 ±3.59	37.74 ±4.98	37.66 ±4.92	34.66 ±3.41	33.57 ±3.04

Table 4.2: Average PSNR and standard deviation (in dB) of various reconstruction approaches applied to various operators.

was trained specifically to denoise the images with various levels of white Gaussian noise. Finally, we implemented the regularize&reuse network (R&R) [Gilton 2021b] composed of  $K = 10$  iterations. The embedded inversion network consists of a pseudo-inverse, followed by a DruNet network trained for the  $\circ$  sampling scheme. The hyperparameters in the method (see [Gilton 2021b]) were tuned to produce the best results. Table 4.2 shows the performance of the different architectures. The following conclusions can be drawn.

**Denoising network** The denoising network trained on the whole family drops by less than 0.4dB compared to ID, when tested with schemes that belong to the training family.

However, the performance is altered (1dB) for the two schemes outside the training family. This suggests that a proper training of a denoising net should cover a sufficiently vast family of operators.

**Unrolled networks** The unrolled networks trained with a family shows a performance drop of at most 0.12dB compared to IU for schemes inside *and outside* the family. This is a remarkable feature: the scheme is able to extrapolates out-



side the training set to some extent. This illustrates one of the take home message of our paper: training unrolled networks on a family does not degrade much the performance while providing adaptivity of the networks.

In addition, the price of adaptivity seems affordable for most applications since a 0.12dB loss is marginal.

**Plug & Play (P&P)** When comparing Table 4.1 and 4.2, we see that the plug&play approach performs better uniformly than models trained on a single operator. However, its performance drops by 1–1.5dB compared to ID and 2.5–3.1dB compared to IU.

It is also significantly less accurate than FU for an identical computational cost. This suggests that for a given reconstruction architecture, it is beneficial to train the proximal networks for a specific task rather than using a *universal* denoiser, as is the case in plug&play.

Notice however, that FU does not extrapolate well to problems completely different from the ones it was trained for. Indeed, we trained FU for an MRI reconstruction problem and tested it for a deblurring application. There, the plug&play approach was considerably more consistent. In a sense, we can see the proposed training as an intermediate step between the plug&play approach (adaptable to all inverse problems) and the traditional training of reconstruction networks (perfectly adapted to a single operator).

**Reuse & Regularize (R&R)** Finally, the R&R approach does improve the results by up to 1.5dB compared to a model trained on a single operator. However, it seems that our simpler training approach provides significantly better results. Notice that R&R has a wider scope since it also deals with unknown perturbations of the forward operator (i.e. blind inverse problems), while we only consider the non blind case in this section.

### 4.6.3 Blind inverse problems

In this section, we illustrate how training on a family of operators allows us to solve different blind inverse problems by minimizing (4.5). Fig. 4.4, 4.6 and 4.5 illustrate some of the results for MRI, deblurring and CT imaging respectively. In these experiments, we assume that

$$\mathbf{y} = \mathbf{A}(\boldsymbol{\theta}_1)\mathbf{x} + \mathbf{b}, \quad (4.15)$$

for some unknown parameter  $\boldsymbol{\theta}_1$  describing the forward model. Starting from an initial guess  $\boldsymbol{\theta}_0$ , we then solve (4.5) with a first order method, resulting in an estimate  $\hat{\boldsymbol{\theta}}_1$  of  $\boldsymbol{\theta}_1$ . Fig. 4.4, 4.5, 4.6 show the performance of the solver for various applications. Let us analyze these results.

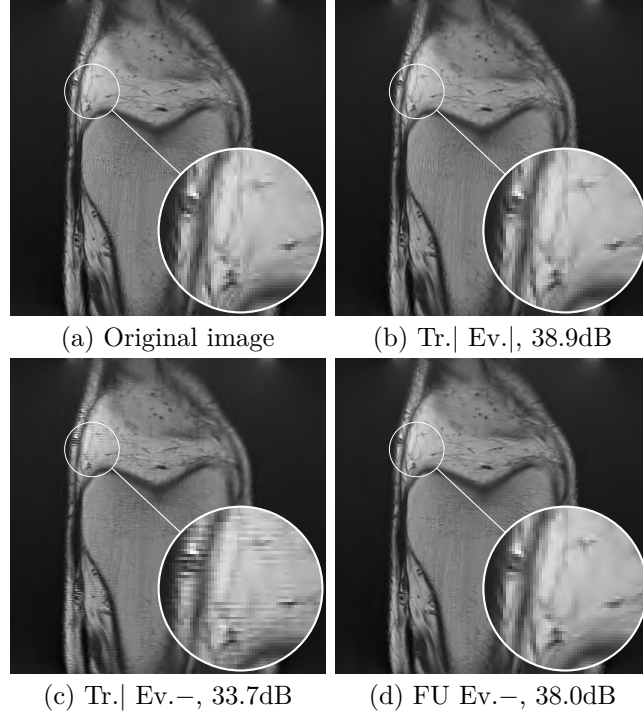


Figure 4.3: Examples of reconstructions using the unrolled network. We trained it on  $|$  (4.3b, 4.3c) and on a family (4.3d). We tested it on  $|$  (4.3b) and  $-$  (4.3c, 4.3d).

#### 4.6.3.1 Magnetic Resonance Imaging

This application provides the most impressive results for various reasons:

- To the best of our knowledge, no one yet attempted to estimate the sensitivity maps and trajectory errors jointly. Estimating divergence in trajectories might look hopeless at first sight, which may explain this fact. Indeed, looking at the differences between  $\xi_1$  and  $\xi_0$  (see top-right and the zoom on the right-most column of Fig. 4.4) we see that the frequency shifts are huge (up to 5 pixels).
- The total number of parameters to estimate is really large. Indeed, it consists in the  $104 \times 15$  parameters describing the sensitivity maps and the 32 parameters describing the convolution kernel that perturbs the trajectories, i.e. 1592 parameters.

If solved without any correction, the reconstruction results are disastrous (see the 2nd column). Solving the consistency problem (4.5) provides near perfect estimates of  $\hat{\theta}$  for all reconstruction mappings. For instance, the green  $\hat{\xi}_1$  and orange  $\xi_1$  trajectories cannot be distinguished on the right column. This may come as a surprise, and seems to suggest that this particular blind inverse problem is not as hard as may seem at first sight. This might be due to multiple redundancies in the data: the 15 reception coils associated to a slight oversampling of the  $k$ -space center (all the trajectories start exactly from the center) seem to ensure the identifiability

of the problem. A nice research perspective is to explain this phenomenon from a theoretical viewpoint.

The reconstruction result obtained with the neural network trained on a family is significantly better than the two other ones (more than +1.3dB compared to the one trained on  $\theta_0$  and to the plug&play approach). In particular, the bone texture is reconstructed with the proposed approach, while it is not for the two others.

#### 4.6.3.2 Computerized tomography

In this application, recall that the angle differences might be due to the motion of a patient in the scanner. As can be seen on the 2nd column, not accounting for this results in severe artifacts including some details loss and blur. This can be easily seen through the PSNR drop of more than 10dB from the 1st to the 2nd column.

A second observation is that the plug&play approach does not work for this application. The reason is likely the missing cone problem. The unrolled network is able to “learn” the invisible [Bubba 2019], that is to recover the kernel part  $\mathbf{k}$  in (4.13). In contrast, a neural network that would be trained only to remove additive Gaussian noise seems unable to do so. We see a clear advantage of the unrolled networks for this particular application.

The unrolled networks trained on the single operator  $\theta_0$  seems unable to recover the angles tilts: the average error only goes from  $0.12^\circ$  to  $0.09^\circ$ . On its side, the unrolled network trained on a family provides promising results, reducing the average tilts error to  $0.024^\circ$ . Recall that this training family consists of operators with random perturbations of the angles and positions. Both network correctly recover the shifts  $\mathbf{s}_1$ , by reducing their size by a factor larger than 10. The gain in performance by training on a family is significant with a difference of more than 2.3dB for the proposed approach.

#### 4.6.3.3 Blind deblurring

This application is one of the most studied in the literature, especially in computer vision. It has an important peculiarity: the basic block of the convolutional neural network is identical to the forward operator. Hence, when an unrolled network is trained, we can expect the networks  $\mathcal{D}[\mathbf{w}_k, \cdot]$  to not only act as “denoisers”, but also as deconvolution mappings.

**Known operator** This fact might explain the second row of Fig. 4.6, where the unrolled network actually degrades the image quality compared to the observation. We indeed go from 21.05dB to 15.62dB with a known operator! This also confirms the conclusions of the introductory example in Fig. 4.1.

The behavior of the unrolled network trained on a family is significantly more appealing with a gain of 5.6dB compared to the observation when knowing the true operator. The plug&play provides a significant increase of 2.6dB, which is not on par with the performance of the proposed approach.

**Unknown operator** When the operator is unknown, only the proposed approach is able to recover an approximate version of the blur kernel (12dB compared to less than 4.5dB for the other approaches). The estimate is not perfect, but it is still sufficient to significantly improve the image resolution (21dB to 25.2dB), with clearly enhanced details (see e.g. the railway).

#### 4.6.3.4 Additional experiments

Finally, to show that the approach is robust and versatile, we provide a few extra experiments in Fig. 4.7, 4.8 and 4.9.

## 4.7 Conclusion

In this work we studied the stability of model-based reconstruction networks to variations of the acquisition operator. We first illustrated significant performance drops when training the models on a single forward model. We then demonstrated on a realistic MRI reconstruction problem that a simple solution to mitigate this effect and ensure a good adaptivity is to train the model on a family of operators. It opens new interesting perspectives for computational imaging. A recent trend consists in optimizing the forward model and the reconstruction algorithm jointly (see e.g. [Weiss 2021, Gossard 2022b, Wang 2022a] for examples in MRI). With a reconstruction method capable of adapting to a vast family of operators, it becomes possible to restrict the attention to the optimization of the forward model only [Gossard 2022a].

We also showed that the design of networks being able to adapt to different forward models, allows solving a variety of blind inverse problems in a convincing way. This aspect is critical in most applications: the forward model is usually known at best approximately, and can even be completely unknown. The benefits of handling this issue are huge in terms of resolution and signal-to-noise-ratio gains.

This promising work opens many perspectives. First, our experiments were conducted with simulated measurements which can lead to both an inverse crime [Colton 1998] and a data crime [Shimron 2022]. Extensive validation should be carried out on real imaging devices. Another question that is not addressed in this paper is how the proposed blind inverse solver behaves if the true operator is not in the range of the admissible operators  $\theta \mapsto \mathbf{A}(\theta)$ ? This question is left for future work as well. Another interesting perspective would be to add motion correction for MRI. A motion in the image domain translates to a phase modulation in the Fourier domain. This is a critical issue in practice. The disconcerting ease with which we solved the estimation of trajectory shifts and sensitivity maps, sparks good hopes to solve this long resisting problem. Finally, a current weakness of the proposed approach, is a rather high computational complexity. Indeed, the model needs to be differentiated from 10 to 1000 times with respect to the parameter  $\theta$ . In practice, this took from 10 minutes to 1 hour in the experiments carried out in

this paper. This might be incompatible with real large scale applications. Hence, better optimization strategies should be developed.

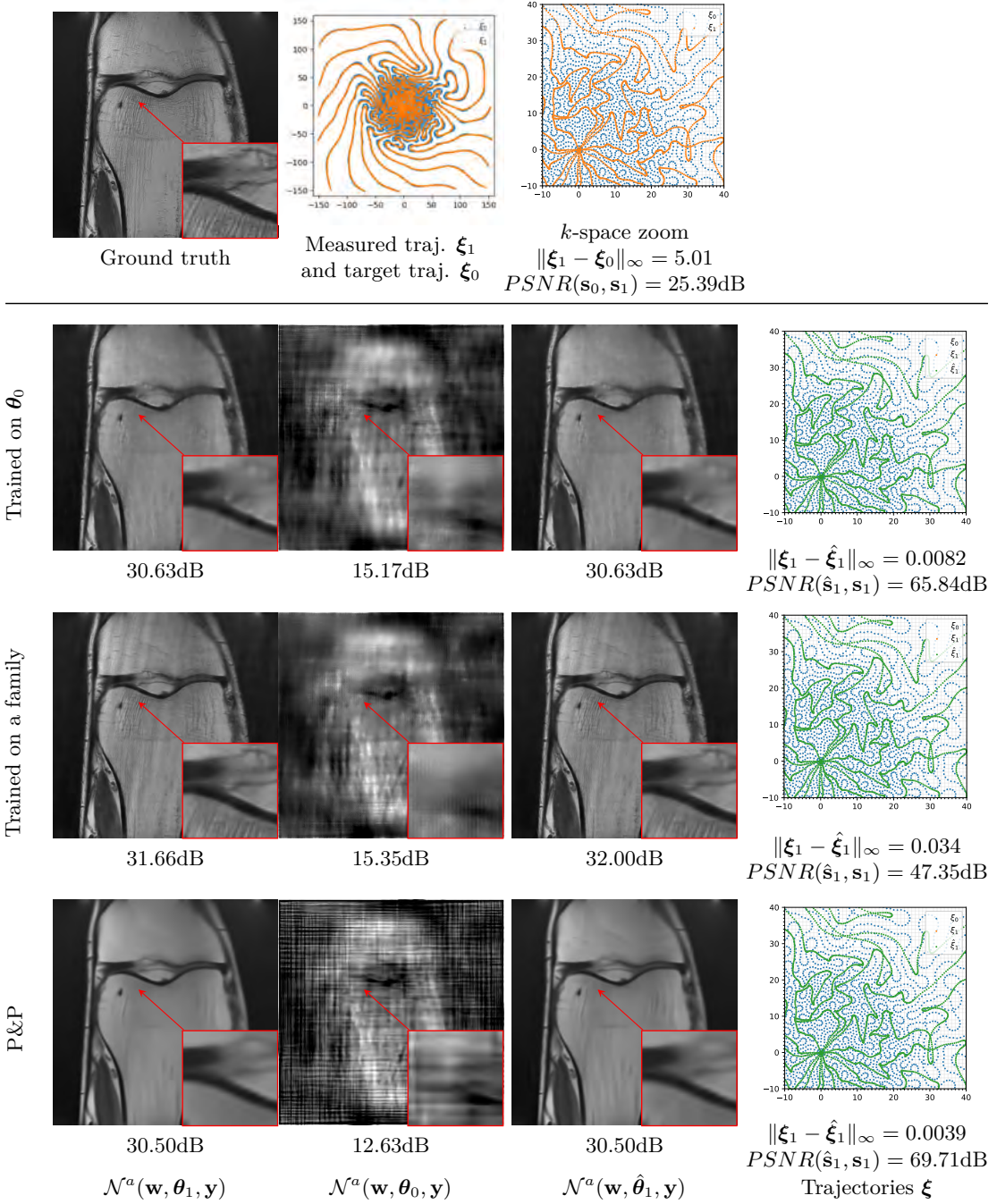


Figure 4.4: Self-calibrated MRI. *1st column*: reconstruction with a perfect knowledge of the forward model  $\theta_1$ . *2nd column*: reconstruction assuming the wrong forward model  $\theta_0$ . *3rd column*: reconstruction using the estimated forward model  $\hat{\theta}_1$ . *4th column*: estimate of the operator. We display the maximal distance between sampling points  $\|\xi_1 - \hat{\xi}_1\|_\infty$  as well as the PSNR of the estimated sensitivity maps  $\hat{s}_1$ . From top to bottom: different training strategies are compared. *2nd row*: trained on  $\theta_0$ . *3rd row*: trained on a family of operators. *4th row*: using a plug&play prior. The PSNR is indicated below each image. The noise level given to the P&P denoiser has been tuned as to yield the best PSNR on the non-blind problem. Other models do not require tuning at evaluation.

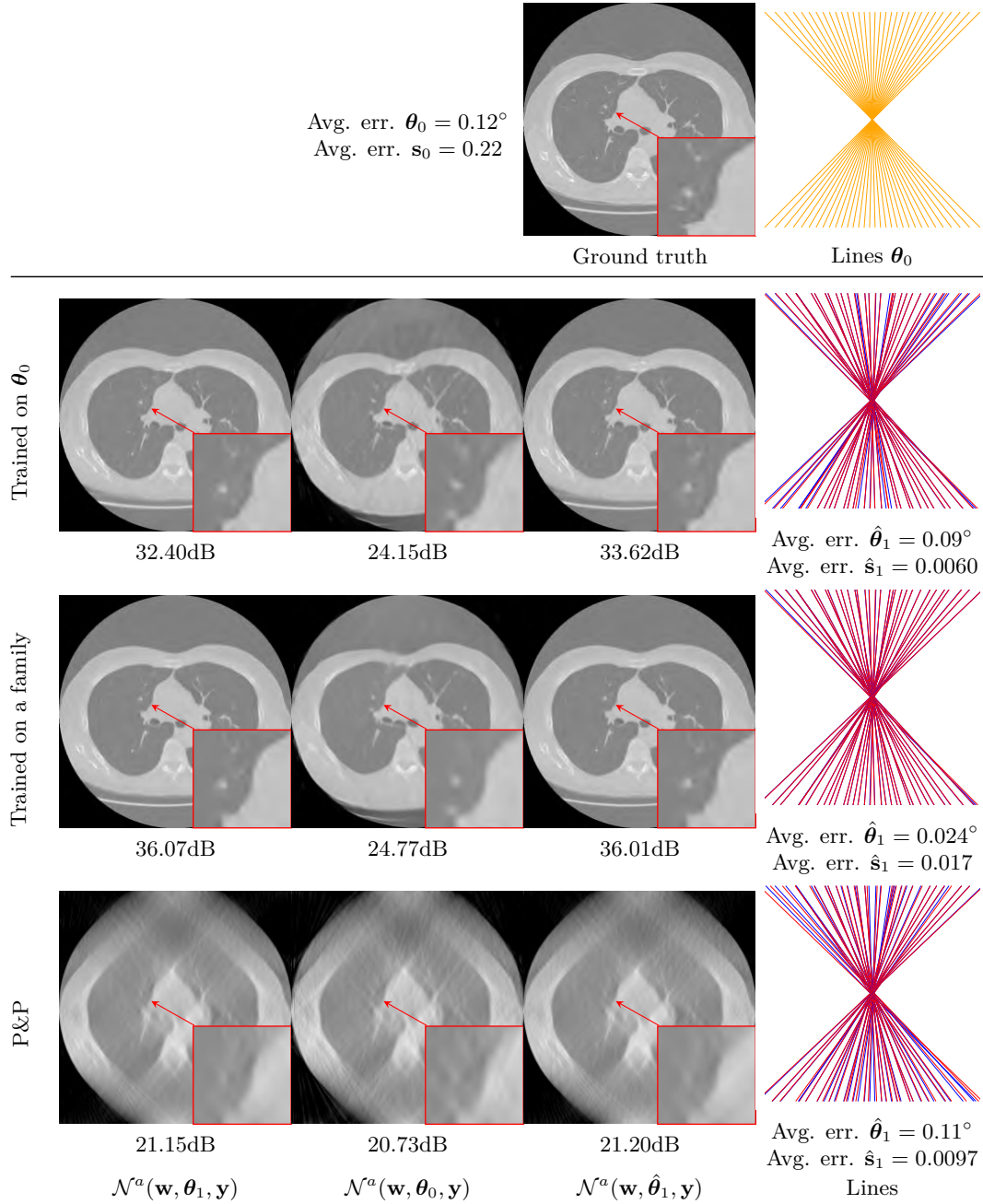


Figure 4.5: Self-calibrated computerized tomography. *1st column*: reconstruction with a perfect knowledge of the forward model  $\theta_1$ . *2nd column*: reconstruction assuming the wrong forward model  $\theta_0$ . *3rd column*: reconstruction using the estimated forward model  $\hat{\theta}_1$ . *4th column*: true  $\theta_1$  (blue) and estimated  $\hat{\theta}_1$  parameters (red) of the forward model. We display the average angle error and the average shift error. *2nd row*: trained on  $\theta_0$ . *3rd row*: trained on a family of operators. *4th row*: using a plug&play prior. The PSNR of the reconstructed image are indicated below each image. The noise level given to the P&P denoiser has been tuned as to yield the best PSNR on the non-blind problem. Other models do not require tuning at evaluation.



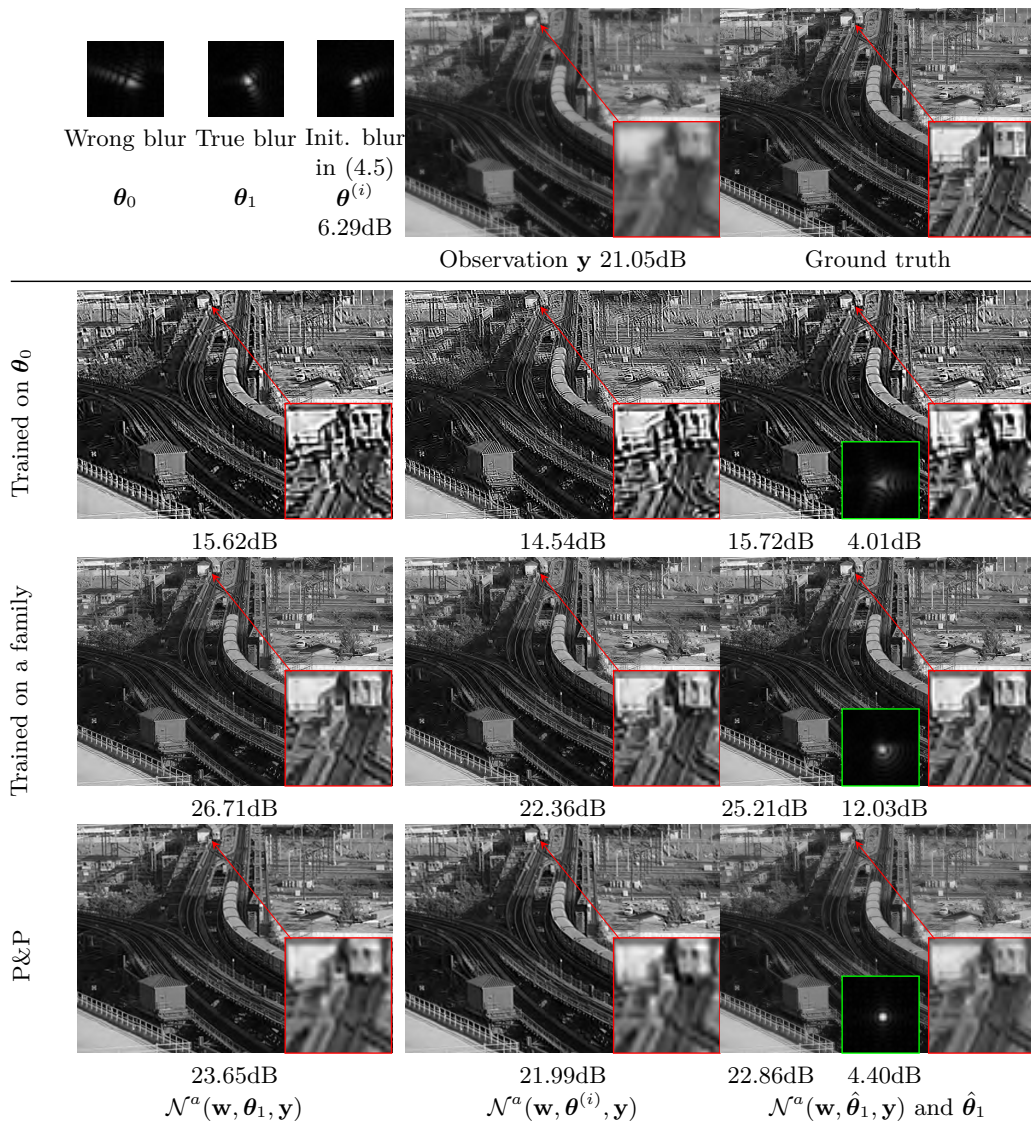


Figure 4.6: Blind deblurring. *1st column*: reconstruction with a perfect knowledge of the forward model  $\theta_1$ . *2nd column*: reconstruction assuming a wrong forward model  $\theta^{(i)}$ . *3rd column*: reconstruction using the estimated forward model  $\hat{\theta}_1$ . From top to bottom: different training strategies are compared. *2nd row*: trained on  $\theta_0$ . *3rd row*: trained on a family of operators. *4th row*: using a plug&play prior. The PSNR of the reconstructed image and the SNR of the reconstructed blur kernel are indicated below each image. The noise level given to the P&P denoiser has been tuned as to yield the best PSNR on the non-blind problem. Other models do not require tuning at evaluation. The measured signal is  $\mathbf{y} = \mathbf{A}(\theta_1)\mathbf{x} + \mathbf{b}$  and the blur of the experiments in the 3rd column is initialized with the blur given in the first row  $\theta^{(i)}$ .



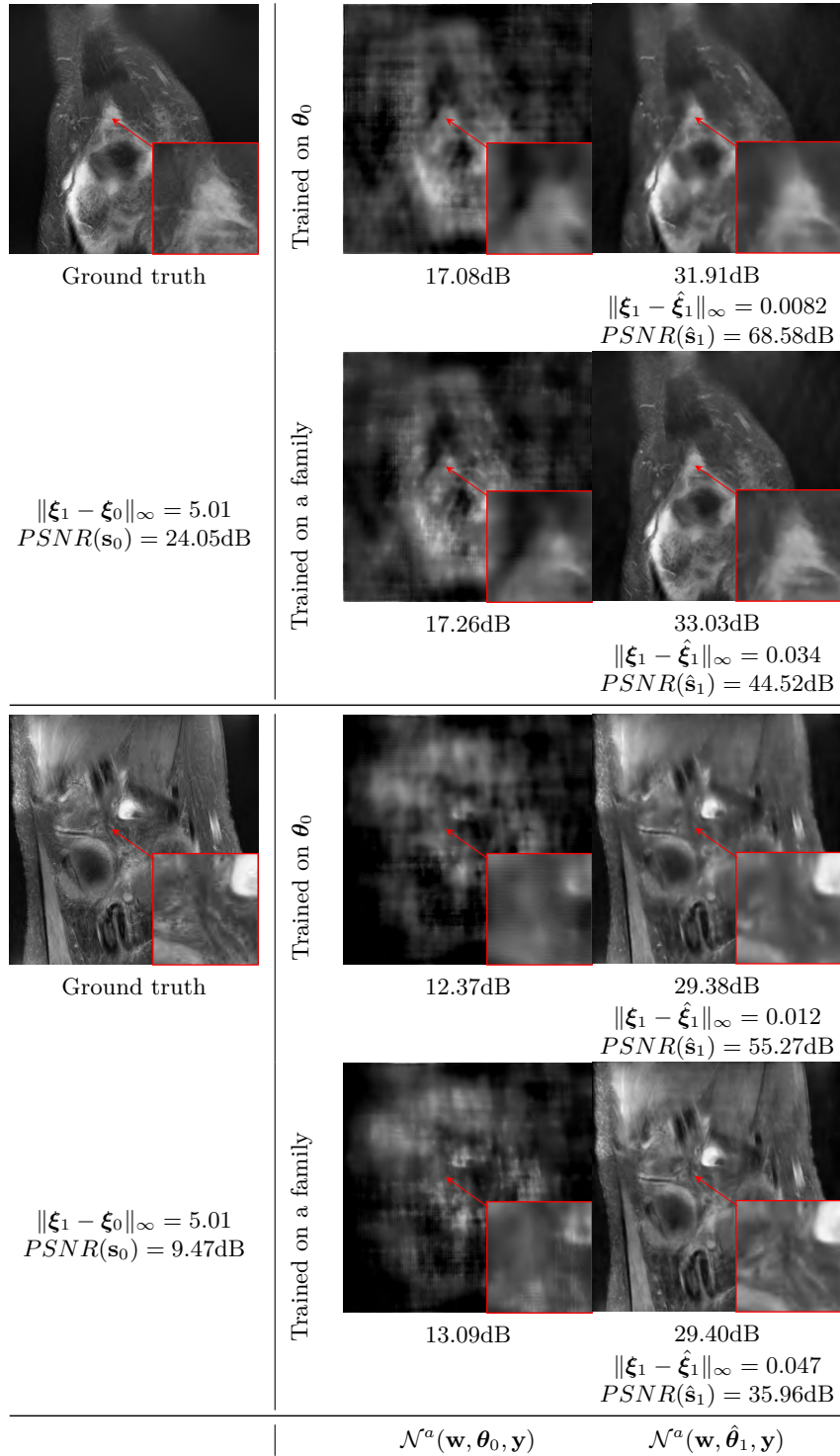


Figure 4.7: Additional experiments for self-calibrated MRI with different images. *1st column*: ground truth. *2nd column*: reconstruction assuming the wrong forward model  $\theta_0$ . *3rd column*: reconstruction using the estimated forward model  $\hat{\theta}_1$ . From top to bottom for each image: different training strategies are compared. *1st row*: trained on  $\theta_0$ . *2nd row*: trained on a family of operators. The PSNR is indicated below each image.

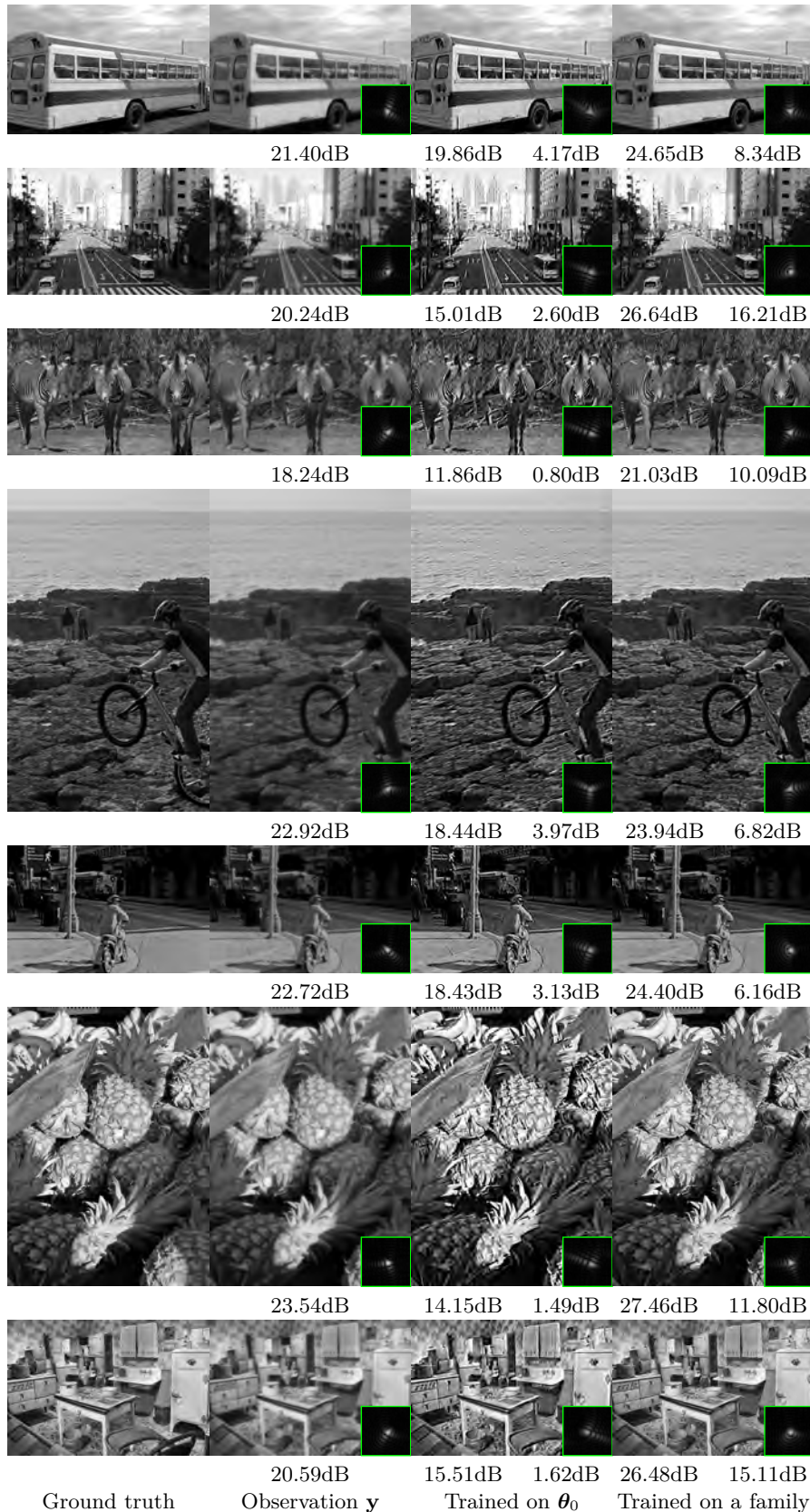


Figure 4.8: Additional experiments for blind deblurring with different images. *1st column:* ground truth. *2nd column:* observation  $y$  and the true blur  $\theta_1$ . *3rd column:* reconstruction using the estimated forward model  $\hat{\theta}_1$  with the network trained on  $\theta_0$ . *4th column:* reconstruction using the estimated forward model  $\hat{\theta}_1$  with the network trained on a family of operators. The PSNR of the reconstructed image and the SNR of the reconstructed blur kernel are indicated below each image.

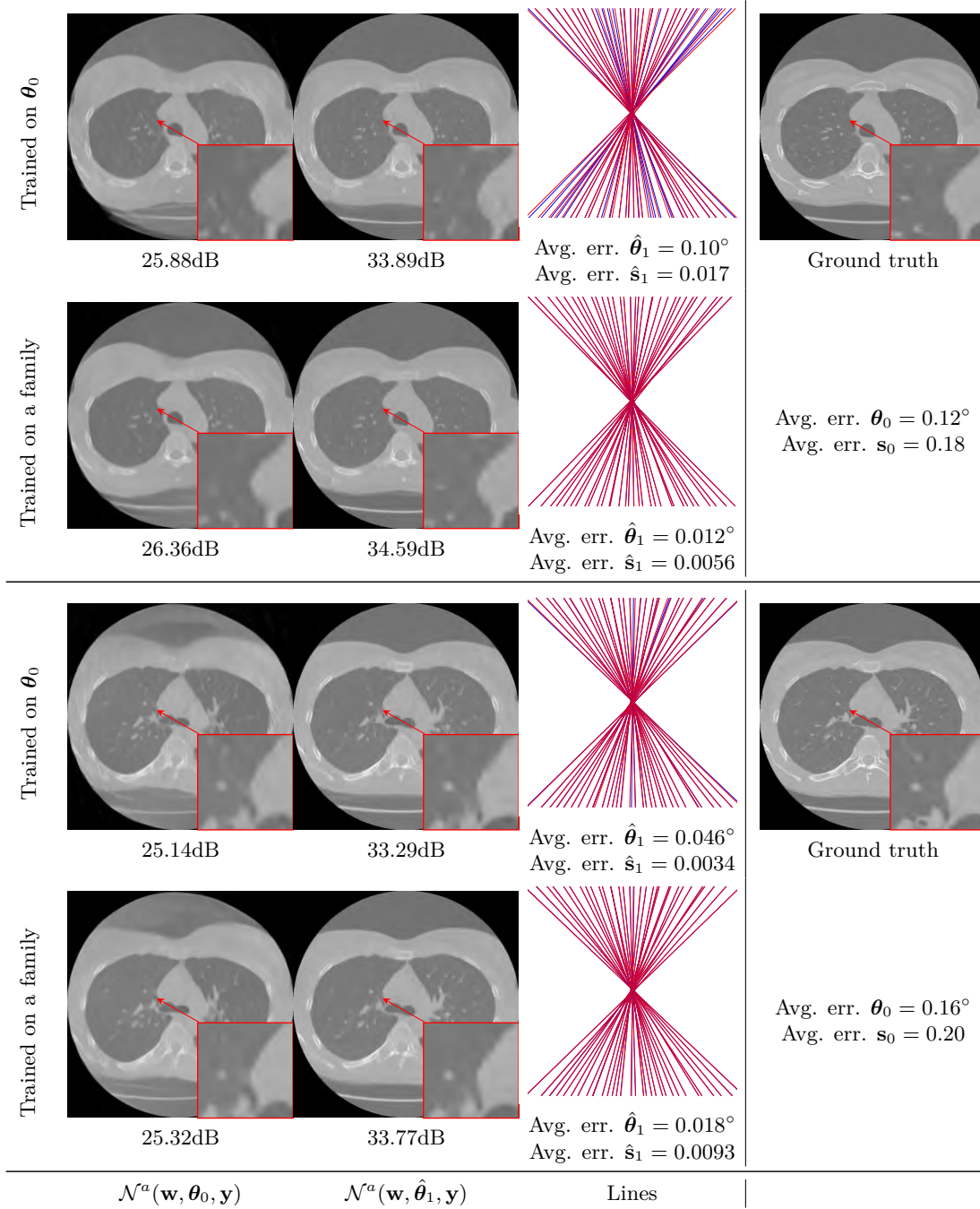


Figure 4.9: Additional experiments for self-calibrated CT with different images. *1st column:* reconstruction assuming the wrong forward model  $\theta_0$ . *2nd column:* reconstruction using the estimated forward model  $\hat{\theta}_1$ . *3rd column:* true  $\theta_1$  (blue) and estimated  $\hat{\theta}_1$  parameters (red) of the forward model. We display the average angle error and the average shift error. *4th column:* ground truth. From top to bottom for each image: different training strategies are compared. *1st row:* trained on  $\theta_0$ . *2nd row:* trained on a family of operators. The PSNR of the reconstructed image are indicated below each image.

# Adaptive scaling of the learning rate by second order automatic differentiation

---

**Résumé** Dans le contexte de l'optimisation des réseaux de neurones profonds, nous proposons une nouvelle méthode de différenciation automatique qui permet de mettre à l'échelle le pas de méthodes d'optimisation. Cette technique repose sur le calcul de la *courbure*, une information de second ordre dont la complexité de calcul se situe entre le calcul du gradient et celui du produit matrice-vecteur avec la hessienne. Si  $(1C, 1M)$  représente respectivement le temps de calcul et l'empreinte mémoire de la méthode du gradient, la technique proposée augmente le coût total à  $(1.5C, 2M)$  ou  $(2C, 1M)$ . Cette remise à l'échelle a l'avantage d'avoir une interprétation naturelle, elle permet à l'utilisateur de choisir entre l'exploration du jeu de paramètres et la convergence de l'algorithme. La remise à l'échelle est adaptative, elle dépend des données et de la direction de la descente. Les expériences numériques mettent en évidence les différents régimes d'exploration et de convergence.

**Abstract** In the context of the optimization of Deep Neural Networks, we propose to rescale the learning rate using a new technique of automatic differentiation. This technique relies on the computation of the *curvature*, a second order information whose computational complexity is in between the computation of the gradient and the one of the Hessian-vector product. If  $(1C, 1M)$  represents respectively the computational time and memory footprint of the gradient method, the new technique increase the overall cost to either  $(1.5C, 2M)$  or  $(2C, 1M)$ . This rescaling has the appealing characteristic of having a natural interpretation, it allows the practitioner to choose between exploration of the parameters set and convergence of the algorithm. The rescaling is adaptive, it depends on the data and on the direction of descent. The numerical experiments highlight the different exploration/convergence regimes.

This chapter is based on the preprint [de Gournay 2022]:  
de Gournay, F., & Gossard, A. (2022). Adaptive scaling of the learning rate by second order automatic differentiation.

## Contents

---

<b>5.1</b>	<b>Introduction</b>	<b>148</b>
5.1.1	Foreword	149
5.1.2	Related works	150
5.1.3	Our contributions	152
<b>5.2</b>	<b>Rescaling the learning rate</b>	<b>152</b>
5.2.1	Presentation and guideline for rescaling	152
5.2.2	Analysis of the rescaling	154
<b>5.3</b>	<b>Computing the curvature</b>	<b>156</b>
5.3.1	Main results	156
5.3.2	Proof of Algorithm 4	157
5.3.3	Proof of Theorem 3	160
<b>5.4</b>	<b>Numerical experiments</b>	<b>163</b>
5.4.1	Convergence/exploration trade off	163
5.4.2	Hyperexploration mode	166
5.4.3	Hyperconvergence mode	167
5.4.4	Influence of the averaging factor of the curvature	168
<b>5.5</b>	<b>Conclusion and discussion</b>	<b>170</b>
<b>5.6</b>	<b>Second order computation</b>	<b>170</b>
5.6.1	Hessian-vector dot product	170
5.6.2	Structure of the layers	171
<b>5.7</b>	<b>Description of the numerical experiments</b>	<b>172</b>
<b>5.8</b>	<b>Additional numerical experiments</b>	<b>175</b>
5.8.1	Dealing with momentum	175
5.8.2	Batch reduction on CIFAR	177
5.8.3	Effect of the $L^2$ regularization	178
5.8.4	Comparison with BB and Robbins-Monro	178

---

## 5.1 Introduction

The optimization of Deep Neural Networks (DNNs) has received tremendous attention over the past years. Training DNNs amounts minimizing the expectation of non-convex random functions in a high dimensional space  $\mathbb{R}^d$ . If  $\mathcal{J} : \mathbb{R}^d \rightarrow \mathbb{R}$  denotes this expectation, the problem reads

$$\min_{\Theta \in \mathbb{R}^d} \mathcal{J}(\Theta), \quad (5.1)$$

with  $\Theta$  the parameters. Optimization algorithms compute iteratively  $\Theta_k$ , an approximation of a minimizer of (5.1) at iteration  $k$ , by the update rule

$$\Theta_{k+1} = \Theta_k - \tau_k \dot{\Theta}_k, \quad (5.2)$$

where  $\tau_k$  is the learning-rate and  $\dot{\Theta}_k$  is the update direction. The choice of  $\dot{\Theta}_k$  encodes the type of algorithm used. This work focuses on the choice of the learning rate  $\tau_k$ .

There is a trade-off in the choice of this learning rate. Indeed high values of  $\tau_k$  allows *exploration* of the parameters space and slowly decaying step size ensures *convergence* in accordance to the famous Robbins-Monro algorithm [Robbins 1951]. This decaying condition may be met by defining the step as  $\tau_k = \tau_0 k^{-\alpha}$  with  $\tau_0$  being the initial step size and  $\frac{1}{2} < \alpha < 1$  a constant. The choice of the initial learning rate and its decay are left to practitioners and these hyperparameters have to be tuned manually in order to obtain the best rate of convergence. For instance, they can be optimized using a grid-search or by using more intricated strategies [Smith 2018], but in all generality tuning the learning rate and its decay factor is difficult and time consuming. The main issue is that the learning rate has no natural scaling. The goal of this work is to propose an algorithm that, given a direction  $\dot{\Theta}_k$  finds automatically a scaling of the learning rate. This rescaling has the following advantages:

- The scaling is adaptive, it depends on the data and of the choice of direction  $\dot{\Theta}_k$ .
- The scaling expresses the convergence vs. exploration trade-off. Multiplying the rescaled learning rate by 1/2 enforces convergence whereas multiplying it by 1 allows for exploration of the space of parameters.

This rescaling comes at a cost and it has the following disadvantages:

- The computational costs and memory footprint of the algorithm goes from  $(1C, 1M)$  to  $(1.5C, 2M)$  or  $(2C, 1M)$ .
- The rescaling method is only available to algorithms that yield directions of descent, it excludes momentum method and notably Adam-flavored algorithm.
- Rescaling is theoretically limited to functions whose second order derivative exists and does not vanish. This non-vanishing condition can be compensated by  $L^2$ -regularization.

### 5.1.1 Foreword

First recall that second order methods for the minimization of a deterministic  $\mathcal{C}^2$  function  $\Theta \mapsto \mathcal{J}(\Theta)$ , with a Hessian that we denote  $\nabla^2 \mathcal{J}$ , are based on the second

order Taylor expansion at iteration  $k$ :

$$\mathcal{J}(\Theta_k - \tau_k \dot{\Theta}_k) \simeq \mathcal{J}(\Theta_k) - \tau_k \langle \dot{\Theta}_k, \nabla \mathcal{J}(\Theta_k) \rangle + \frac{\tau_k^2}{2} \langle \nabla^2 \mathcal{J}(\Theta_k) \dot{\Theta}_k, \dot{\Theta}_k \rangle. \quad (5.3)$$

If the Hessian of  $\mathcal{J}$  is positive definite, the minimization of the right-hand side leads to the choice

$$\dot{\Theta}_k = P_k^{-1} \nabla \mathcal{J}(\Theta_k) \text{ with } P_k \simeq \nabla^2 \mathcal{J}(\Theta_k). \quad (5.4)$$

Once a direction  $\dot{\Theta}_k$  is chosen, another minimization in  $\tau_k$  gives

$$\tau_k = \frac{\langle \dot{\Theta}_k, \nabla \mathcal{J}(\Theta_k) \rangle}{\|\dot{\Theta}_k\|^2 c(\Theta_k, \dot{\Theta}_k)}, \quad (5.5)$$

where  $c$  is the curvature of the function, and is defined as

$$c(\Theta_k, \dot{\Theta}_k) \stackrel{\text{def}}{=} \frac{\langle \nabla^2 \mathcal{J}(\Theta_k) \dot{\Theta}_k, \dot{\Theta}_k \rangle}{\|\dot{\Theta}_k\|^2}. \quad (5.6)$$

A second-order driven algorithm can be decomposed in two steps: i) the choice of  $P_k$  in (5.4), and if this choice leads to an update which is a direction of ascent, that is  $\langle \dot{\Theta}_k, \nabla \mathcal{J}(\Theta_k) \rangle > 0$ , ii) a choice of  $\tau_k$  by an heuristic inspired from (5.6) and (5.5).

In the stochastic setting, we denote as  $s \mapsto \mathcal{J}_s$  the mapping of the random function. At iteration  $k$ , only information on  $(\mathcal{J}_s)_{s \in \mathcal{B}_k}$  can be computed where  $(\mathcal{B}_k)_k$  is a sequence of mini-batches which are indepently drawn. If  $\mathbb{E}_{s \in \mathcal{B}_k}$  is the empirical average over the mini-batch, we define  $\mathcal{J}_{\mathcal{B}_k} = \mathbb{E}_{s \in \mathcal{B}_k}[\mathcal{J}_s]$ . Given  $\Theta$ , the quantity  $\mathcal{J}(\Theta)$  is deterministic, and  $\mathcal{J}$  is the expectation of  $\mathcal{J}_s$  w.r.t.  $s$ .

### 5.1.2 Related works

**Choice of  $P_k$ :** The choice  $P_k = \nabla^2 \mathcal{J}_{\mathcal{B}_k}(\Theta_k)$  in (5.4), leads to a choice  $\tau_k = 1$  and to the so-called Newton method. It is possible in theory to compute the Hessian by automatic differentiation if it is sparse [Walther 2008], but to our knowledge it has not been implemented yet. In [Martens 2010], the authors solve  $\dot{\Theta}_k = [\nabla^2 \mathcal{J}_{\mathcal{B}_k}(\Theta_k)]^{-1} \nabla \mathcal{J}_{\mathcal{B}_k}(\Theta_k)$  by a conjugate gradient method which requires only matrix-vector product which is affordable by automatic differentiation [Christianson 1992, Pearlmutter 1994]. This point of view, as well as some variants [Vinyals 2012, Krishnan 2018], suffer from high computational cost per batch and go through less data in a comparable amount of time, leading to slower convergence at the beginning of the optimization.

Another choice is to set  $P_k \simeq \nabla^2 \mathcal{J}_{\mathcal{B}_k}(\Theta_k)$  in (5.4) which is coined as the ‘‘Quasi-Newton’’ approach. These methods directly invert a diagonal, block-diagonal or low rank approximation of the Hessian [Becker 1988, Schaul 2013, Roux 2007, Ollivier 2015, Martens 2015, Yao 2021]. In most of these works, the Hessian is approximated by  $\mathbb{E}[\nabla \mathcal{J}_s(\theta_k) \nabla \mathcal{J}_s(\theta_k)^T]$ , the so-called Fisher-Information matrix, which leads to the natural gradient method [Amari 1998]. Note also the use of a low-rank approximation of the true Hessian for variance reduction in [Gower 2018].



Finally, there is an interpretation of adaptive methods as Quasi-Newton methods. Amongst the adaptive methods, let us cite RMSProp [Tieleman 2012], Adam [Kingma 2015], Adagrad [Duchi 2011] and Adadelta [Zeiler 2012]. For all these methods,  $P_k$  is as a diagonal preconditioner that reduces the variability of the step size across the different layers. This class of methods can be written

$$\dot{\Theta}_k = P_k^{-1} m_k, \quad m_k \simeq \nabla \mathcal{J}(\Theta_k) \quad \text{and} \quad P_k \simeq \nabla^2 \mathcal{J}(\Theta_k). \quad (5.7)$$

For instance, RMSProp and Adagrad use  $m_k = \nabla \mathcal{J}_{\mathcal{B}_k}(\Theta_k)$  whereas Adam maintains in  $m_k$  an exponential moving averaging from the past evaluations of the gradient. The RMSProp, Adam and Adagrad optimizers build  $P_k$  such that  $P_k^2$  is a diagonal matrix whose elements are exponential moving average of the square of the past gradients (see [Reddi 2018] for example). It is an estimator of the diagonal part of the Fisher-Information matrix.

All these methods can be incorporated in our framework as we consider the choice of  $P_k$  as a preconditioning technique whose step is yet to be found. In a nutshell, if  $P_k$  approximates the Hessian up to an unknown multiplicative factor, our method is able to find this multiplicative factor.

**Barzilai-Borwein:** The Barzilai-Borwein (BB) class of methods [Barzilai 1988, Raydan 1997, Dai 2002, Xiao 2010, Biglari 2013, Li 2019a] may be interpreted as methods which aim at estimating the curvature in (5.6) by numerical differences using past gradient computations. In the stochastic convex setting, the BB method was introduced in [Tan 2016] for the choice  $\dot{\Theta}_k = \nabla \mathcal{J}(\Theta_k)$  with variance-reducing methods [Johnson 2013]. It has been extended in [Ma 2018] to non-convex problems and in [Liang 2019] to DNNs. Due to the variance of the gradient and possibly to a poor estimation of the curvature by numerical differences, these methods allow prescribing a new step at each epoch only. In [Yang 2018, Castera 2022], the step is prescribed at each iteration at the cost of computing two mini-batch gradients per iteration. Moreover, in [Yang 2018] the gradient over all the data needs to be computed at the beginning of each epoch whereas [Castera 2022] maintains an exponential moving average to avoid this extra computation. The downside of [Castera 2022] is that they still need to tune the learning rate and its decay factor and that their method has not been tried on other choices than  $P_k = \text{Id}$ .

Our belief is that approximating by numerical differences in a stochastic setting suffers too much from variance from the data and from the approximation error. Hence we advocate in this study for exact computations of the curvature (5.6).

**Automatic differentiation:** The theory that allows to compute the matrix-vector product of the Hessian with a certain direction is well-studied [Walther 2008, Christianson 1992, Griewank 2008, Pearlmutter 1994] and costs 4 passes (2 forward and backward passes) and 3 memory footprint, when the computation of the gradient costs 2 passes (1 forward and backward pass) and 1 memory footprint. We study the cost of computing the curvature defined in (5.6), which to the best of our knowledge, has never been studied. Our method has a numerical cost that is always lower than the best BB method [Castera 2022].



### 5.1.3 Our contributions

We propose a change of point of view. While most of the methods presented above use first order information to develop second order algorithms, we use second order information to tune a first order method. The curvature (5.6) is computed using automatic differentiation in order to estimate the local Lipschitz constant of the gradient and to choose a step accordingly. Our contribution is threefold:

- We propose a method that automatically rescales the learning rate using curvature information in Section 5.2.1 and we discuss the heuristics of this method in Section 5.2.2. The rescaling allows the practitioner to choose between three different physical regimes coined as : hyperexploration, exploration/convergence trade-off and hyperconvergence.
- We study the automatic differentiation of the curvature in Section 5.3 and its computational costs.
- Numerical tests are provided in Section 5.4 with a discussion on the three different physical regimes introduced in Section 5.2.1.

## 5.2 Rescaling the learning rate

### 5.2.1 Presentation and guideline for rescaling

The second order analysis of Section 5.1 relies on the Taylor expansion

$$\mathcal{J}(\Theta_k - \tau_k \dot{\Theta}_k) \simeq \mathcal{J}(\Theta_k) - \tau_k \langle \dot{\Theta}_k, \nabla \mathcal{J}(\Theta_k) \rangle + \frac{\tau_k^2}{2} c(\Theta_k, \dot{\Theta}_k) \|\dot{\Theta}_k\|^2,$$

with  $c(\Theta_k, \dot{\Theta}_k)$  given by (5.6). This Taylor expansion yields the following algorithm: given  $\Theta_k$  and an update direction  $\dot{\Theta}_k$ , compute  $c(\Theta_k, \dot{\Theta}_k)$  by (5.6), the step  $\tau_k$  by (5.5) and finally update the parameters  $\Theta_k$  by (5.2). The first order analysis is slightly different. Starting with the second order exact Taylor expansion in integral form:

$$\mathcal{J}(\Theta_k - \tau_k \dot{\Theta}_k) = \mathcal{J}(\Theta_k) + \tau_k \langle \dot{\Theta}_k, \nabla \mathcal{J}(\Theta_k) \rangle + \int_{t=0}^{\tau_k} (\tau_k - t) c(\Theta_k - t \dot{\Theta}_k, \dot{\Theta}_k) \|\dot{\Theta}_k\|^2 dt,$$

we introduce the local directional Lipschitz constant of the gradient

$$L_k = \max_{t \in [0, \tau_k]} |c(\Theta_k - t \dot{\Theta}_k, \dot{\Theta}_k)|, \quad (5.8)$$

in order to bound the right-hand side of the Taylor expansion. One obtains

$$\mathcal{J}(\Theta_k - \tau_k \dot{\Theta}_k) \leq \mathcal{J}(\Theta_k) - \tau_k \langle \dot{\Theta}_k, \nabla \mathcal{J}(\Theta_k) \rangle + \frac{\tau_k^2}{2} L_k \|\dot{\Theta}_k\|^2. \quad (5.9)$$

Introducing the rescaling  $r_k$

$$r_k = \frac{\langle \dot{\Theta}_k, \nabla \mathcal{J}(\Theta_k) \rangle}{\|\dot{\Theta}_k\|^2 L_k} \quad (5.10)$$

and writing  $\tau_k = r_k \ell$ , Equation (5.9) turns into

$$\mathcal{J}(\Theta_k - \ell r_k \dot{\Theta}_k) \leq \mathcal{J}(\Theta_k) + (\ell^2 - \ell) \frac{L_k}{2} \|r_k \dot{\Theta}_k\|^2 \quad \forall \ell. \quad (5.11)$$

Any choice of  $\ell$  in  $]0, 1[$  leads to a decrease of  $\mathcal{J}$  in (5.11). The choice  $\ell = \frac{1}{2}$  allows faster decrease of the right-hand side of (5.11). We coin the choice  $\ell = 1$  in (5.11) as the *exploration choice* and the choice  $\ell = \frac{1}{2}$  as the *convergence choice*. The only difficulty in computing (5.10) lies in the computation of  $L_k$ . Indeed,  $L_k$  is a maximum over an unknown interval and, in the stochastic setting, we only estimate the function  $\mathcal{J}$  and its derivative on a batch  $\mathcal{B}_k$ .

We propose to build  $\tilde{L}_k$  an estimator of  $L_k$  by using the following rules.

- Replace the maximum over the unknown interval  $[0, \tau_k]$  in (5.8) by the value at  $t = 0$ . This is reminiscent of the Newton's method.
- Perform an exponential moving average on the past computations of  $r_k$  in order to average over the data previously seen.
- Use the maximum of this latter exponential moving average and the current estimate in order to stabilize  $\tilde{L}_k$ .

The algorithm reads as follows:

---

**Algorithm 3** Rescaling of the learning rate

---

- 1: **Hyperparameters**  $\beta_3 = 0.9$  (exponential moving average).
  - 2: **Initialization**  $\hat{c}_0 = 0$
  - 3: **Input (at each iteration  $k$ ):** a batch  $\mathcal{B}_k$ ,  $g_k = \mathbb{E}_{s \in \mathcal{B}_k} [\nabla \mathcal{J}_s(\Theta_k)]$  and  $\dot{\Theta}_k$  a direction that verifies  $\langle g_k, \dot{\Theta}_k \rangle > 0$ .
  - 4:  $c_k = \mathbb{E}_{s \in \mathcal{B}_k} \left[ \left| \langle \nabla^2 \mathcal{J}_s(\Theta_k) \dot{\Theta}_k, \dot{\Theta}_k \rangle \right| \right] / \|\dot{\Theta}_k\|^2$  ▷ local curvature
  - 5:  $\hat{c}_k = \beta_3 \hat{c}_{k-1} + (1 - \beta_3) c_k$  and  $\tilde{c}_k = \hat{c}_k / (1 - \beta_3^k)$  ▷ moving average
  - 6:  $\tilde{L}_k = \max(\tilde{c}_k, c_k)$  ▷ stabilization
  - 7:  $r_k = \langle \dot{\Theta}_k, g_k \rangle / (2 \|\dot{\Theta}_k\|^2 \tilde{L}_k)$  ▷ rescaling factor
  - 8: **Output (at each iteration  $k$ ):**  $r_k$  a rescaling of the direction  $\dot{\Theta}_k$ .
  - 9: **Usage of rescaling:** The practitioner should use the update rule  $\Theta_{k+1} = \Theta_k - \ell r_k \dot{\Theta}_k$ , where  $\ell$  follows the *Rescaling guidelines* (see below).
- 

Note that the curvature  $c_k$  is computed with the same batch that the one used to compute  $g_k$  and  $\dot{\Theta}_k$ .

**Rescaling guidelines** Given a descent direction  $\dot{\Theta}_k$ , the update rule is given by

$$\Theta_{k+1} = \Theta_k - \ell r_k \dot{\Theta}_k,$$

where  $\ell$  is the learning-rate that has to be chosen by the practitioner and  $r_k$  is the rescaling computed by Algorithm 3. In the deterministic case,  $\ell$  has a physical interpretation:

- $1 \geq \ell \geq \frac{1}{2}$  (*Convergence/exploration trade-off*). The choice  $\ell = 1$  (exploration) is the largest step that keeps the loss function non increasing. The choice  $\ell = 1/2$  (convergence) ensures the fastest convergence to the closest local minimum. It is advised to start from  $\ell = 1$  and decrease to  $\ell = 1/2$  (see Section 5.4.1).
- $\ell > 1$  (*Hyperexploration*). The expected behavior is a loss function increase and large variations of the parameters. This mode can be used to escape local basin of attraction in annealing methods (see Section 5.4.2).
- $0 < \ell < 1/2$  (*Hyperconvergence*). This mode slows down the convergence. In the stochastic setting, if the practitioner has to resort to setting  $\ell < 1/2$  in order to obtain convergence, then some stochastic effects are of importance in the optimization procedure (see Section 5.4.3).

## 5.2.2 Analysis of the rescaling

Several remarks are necessary to understand the limitations and applications of rescaling.

**The algorithm does not converge in the deterministic setting.** Note that in the deterministic one dimensional case, when  $\mathcal{J}$  is convex (i.e. the curvature is positive),  $\beta_3 = 0$  and  $\ell = 1/2$ , the algorithm boils down to the Newton method. It is known that the Newton method may fail to converge, even for strictly convex smooth functions. For example if we choose

$$\mathcal{J}(\Theta) = \sqrt{1 + \Theta^2},$$

the iterates of the Newton method are given by  $\Theta_{k+1} = -\Theta_k^3$ , which diverges as soon as  $|\Theta_0| > 1$ . This problem comes from the fact that the curvature  $c(\Theta_k - t\dot{\Theta}_k, \dot{\Theta}_k)$  has to be computed for each  $t \in [0, \tau_k]$  in order to estimate  $L_k$  in (5.8) but this maximum is estimated by its value at  $t = 0$ . In this example,  $c(\Theta_k, \dot{\Theta}_k)$  is a bad estimator for  $L_k$  as it is too small and the resulting step is too large.

Another issue in DNN is the massive use of piecewise linear activation functions which can make the Hessian vanish and in this case, the rescaled algorithm may diverge. For example if the loss function is locally linear, then  $c_k = 0$  in line 4 and if we choose  $\beta_3 = 0$  then  $r_k = +\infty$  in line 7.

**The algorithm does not converge in the stochastic setting.** Let  $X$  be a vector-valued random variable and  $\mathcal{J}$  is the function

$$\mathcal{J}(\Theta) = \frac{1}{2} \mathbb{E} \left[ \|\Theta - X\|^2 \right],$$

then for any value of  $\beta_3$  and for the choice  $\ell = \frac{1}{2}$ , the rescaled algorithm yields the update  $\Theta_k = \mathbb{E}_{\mathcal{B}_k}[X]$ , when the optimal value is  $\Theta^* = \mathbb{E}[X]$ . The algorithm oscillates around  $\Theta^*$ , with oscillations depending on the variance of the gradient. This oscillating stochastic effect is well known and is the basic analysis of SGD. Since the proposed rescaling analysis is performed in a deterministic setting, it is not designed to offer any solution to this problem.

**Enforcing convergence by Robbins-Monro conditions** In order to enforce convergence, we can use the results of [Robbins 1951]. It is then sufficient to sow instructions like

$$\alpha \leq \ell r_k k^\delta \leq \beta, \quad (5.12)$$

with fixed  $\alpha, \beta > 0$  and  $\delta \in ]1/2, 1[$ . This is the choice followed by [Castera 2022] for instance. Note that convergence analysis for curvature-dependent step is, to our knowledge, studied only in [Alvarez 2004], for the non-stochastic time-continuous setting.

**Fostering convergence with  $L_2$  regularization** In the deterministic case, the non-convergence of the algorithm is caused by vanishing eigenvalues of the Hessian. This issue can be fixed by adding a term  $\frac{\lambda}{2} \|\Theta\|^2$  to the function  $\Theta \mapsto \mathcal{J}(\Theta)$ , with  $\lambda > 0$ . This method is coined as  $L_2$  regularization with parameter  $\lambda$ . This method shifts the eigenvalues of the Hessian of  $\mathcal{J}$  by the parameter  $\lambda$ . Close to the minimum, every eigenvalue of the Hessian is then positive. Although  $L_2$  regularization does not guarantee convergence, it promotes it.

**Gradient preconditioning** In case of gradient preconditioning  $\dot{\Theta}_k = P_k^{-1} g_k$  with  $P_k \simeq \nabla^2 \mathcal{J}(\Theta_k)$ , the advantage of rescaling is that the practitioner is allowed to approximate the Hessian up to a multiplicative factor. Indeed suppose that instead of providing a good estimate of the Hessian, the practitioner multiplies it at each iteration by an arbitrary factor  $\alpha_k \in \mathbb{R}$ . In this case,  $\dot{\Theta}_k$  is multiplied by  $\alpha_k^{-1}$  but the curvature  $c_k$  does not change in Line 5 of Algorithm 3. This means that  $\hat{c}_k$  is independent of the previous  $(\alpha_s)_{s \leq k}$ . Finally, the rescaling  $r_k$  is multiplied by  $\alpha_k$  in Line 7. Hence the output of the algorithm  $r_k \dot{\Theta}_k$  is independent of the sequence  $(\alpha_s)_{s \leq k}$ . Therefore, the practitioner does not need to worry about finding the right multiplicative factor, it is accounted for by the rescaling method.

**Negativeness of the curvature (line 4)** The main difference between a first-order analysis (5.8) and a second-order (5.5) lies in handling the case when the curvature is negative. The first-order analysis, which we choose, relies on using

absolute value of the curvature, when second-order analysis relies on more intricate methods, see [Allen-Zhu 2018, Carmon 2017, Liu 2017, Curtis 2019]. Note that the absolute value is taken inside the batch average in line 4 and not outside. Otherwise data in the batch where  $\langle \nabla \mathcal{J}^2(\Theta_k) \dot{\Theta}_k, \dot{\Theta}_k \rangle$  is negative could compensate the data where it is positive, leading to a bad estimation of the curvature.

**Heuristics in the estimation of  $L_k$  (lines 5 and 6)** The estimator of  $L_k$  must comply with two antagonist requirements. The first one is to average the curvature over the different batches to effectively compute the true curvature of  $\mathcal{J}$ . The second one is to use the local curvature at point  $\Theta_k$  and in the direction  $\dot{\Theta}_k$  which requires to forget old iterations. This advocates the use of an exponential moving average in line 5 with the parameter  $\beta_3$ . The maximum in line 6 is reminiscent of the construction of AMSGrad [Reddi 2018] from Adam [Kingma 2015], and it stabilizes batches where  $c_k \gg \tilde{c}_k$ . In order to be consistent with the remark in *Gradient preconditioning*, the averaged quantity is the one which does not depend on the unknown multiplicative factor  $\alpha_k$ .

### 5.3 Computing the curvature

In this section, we focus on the computation of  $c(\Theta, \dot{\Theta})$  by automatic differentiation and its cost.

#### 5.3.1 Main results

A Neural Network  $\mathcal{N}$  is a directed acyclic graph and at each node of the graph, the data are transformed and fed to the rest of the graph. The data at the output  $x_n$  are then compared to  $y$ . Since there is no cycle in the graph, there is no mathematical restriction to turn such graph into a list. The set of parameters for layer  $s$  is denoted as  $\theta_s$ , and we denote  $\Theta = (\theta_s)_{s=0..n}$  the set of parameters of  $\mathcal{N}$ . The action of  $\mathcal{N}$  is expressed by the recurrence:

$$x_{s+1}(\Theta) = \mathcal{F}_s(x_s(\Theta), \theta_s), \quad 0 \leq s \leq n-1 \quad (5.13)$$

where  $\mathcal{F}_s$  is the action of the  $s^{\text{th}}$  layer of  $\mathcal{N}$ . The output  $x_n$  is then compared to a target via a loss function  $\mathcal{F}_n$  and we denote  $x_{n+1} \in \mathbb{R}$  the result of this loss function.

Let  $X(\Theta) = (x_s(\Theta))_{s=0..n+1}$  denote the set of data as it is transformed through the neural network. The intermediate data  $x_s(\Theta)$  (resp. parameter  $\theta_s$ ) are supposed to belong to an Hilbert space  $\mathcal{H}_s$  (resp.  $\mathcal{G}_s$ ). We then have for each  $0 \leq s \leq n$

$$\mathcal{F}_s : \mathcal{H}_s \times \mathcal{G}_s \rightarrow \mathcal{H}_{s+1} \text{ and } \mathcal{H}_{n+1} = \mathbb{R}.$$

The gradient of  $\mathcal{J}$  with respect to  $\Theta$  is computed using automatic differentiation. This requires to define the differentials of  $\mathcal{F}_s$  with respect to its variables. Let  $\partial_x \mathcal{F}_s : \mathcal{H}_s \rightarrow \mathcal{H}_{s+1}$ , resp.  $\partial_\theta \mathcal{F}_s : \mathcal{G}_s \rightarrow \mathcal{H}_{s+1}$ , be the differential of  $\mathcal{F}$  at the point

$(x_s(\Theta), \theta_s)$  w.r.t.  $x$ , resp.  $\theta$ . Denote  $(\partial_x \mathcal{F}_s)^* : \mathcal{H}_{s+1} \rightarrow \mathcal{H}_s$  and  $(\partial_\theta \mathcal{F}_s)^* : \mathcal{H}_{s+1} \rightarrow \mathcal{G}_s$  the adjoints of the differentials of  $\mathcal{F}_s$ . These adjoints are defined for all  $\phi \in \mathcal{H}_{s+1}$  as the unique linear mapping that verifies:

$$\begin{aligned} \langle \partial_x \mathcal{F}_s^* \phi, \psi \rangle_{\mathcal{H}_s} &= \langle \phi, \partial_x \mathcal{F}_s \psi \rangle_{\mathcal{H}_{s+1}} \quad \forall \psi \in \mathcal{H}_s \\ \langle \partial_\theta \mathcal{F}_s^* \phi, \psi \rangle_{\mathcal{G}_s} &= \langle \phi, \partial_\theta \mathcal{F}_s \psi \rangle_{\mathcal{H}_{s+1}} \quad \forall \psi \in \mathcal{G}_s. \end{aligned}$$

Denote by  $\nabla^2 \mathcal{F}_s$  the second order derivative tensor of  $\mathcal{F}_s$  at the point  $(x_s, \theta_s)$ . The backward of the data  $\hat{X} = (\hat{x}_s)_{s=1..n+1}$  and the backward-gradient  $\hat{\Theta} = (\hat{\theta}_s)_{s=0..n}$  are defined by:

$$\begin{cases} \hat{x}_s = (\partial_x \mathcal{F}_s)^* \hat{x}_{s+1} & \text{with } \hat{x}_{n+1} = 1 \\ \hat{\theta}_s = (\partial_\theta \mathcal{F}_s)^* \hat{x}_{s+1}. \end{cases} \quad (5.14)$$

In Algorithm 4, the standard backpropagation algorithm is given as well as the modifications needed to compute the curvature. The proof of this algorithm is given in Section 5.3.2.

---

**Algorithm 4** Backpropagation *with curvature computation*

---

- 1: Compute and store the data  $X = (x_s)_s$  with a forward pass (5.13).
- 2: Compute *and store* the backward  $\hat{X} = (\hat{x}_s)_s$  and  $\hat{\Theta} = (\hat{\theta}_s)_s$  using (5.14).
- 3: Then  $\nabla \mathcal{J}(\Theta) = \hat{\Theta}$ .
- 4: Choose any direction of update  $\dot{\Theta} = (\dot{\theta}_s)_s$ .
- 5: Compute the tangent  $\dot{X} = (\dot{x}_s)_s$  with the following forward pass:

$$\dot{x}_{s+1} = (\partial_x \mathcal{F}_s) \dot{x}_s + (\partial_\theta \mathcal{F}_s) \dot{\theta}_s, \quad \dot{x}_0 = 0. \quad (5.15)$$

- 6: Then  $\langle \nabla^2 \mathcal{J}(\Theta) \dot{\Theta}, \dot{\Theta} \rangle = \sum_s \langle \dot{x}_{s+1}, \nabla^2 \mathcal{F}_s(\dot{x}_s, \dot{\theta}_s) \otimes (\dot{x}_s, \dot{\theta}_s) \rangle_{\mathcal{H}_{s+1}}$ .
- 

By Algorithm 4, the computation of the curvature  $c(\theta, \dot{\theta})$  requires 3 passes in total and the storage of  $X$  and  $\hat{X}$  whereas the computation of the gradient requires 2 passes and the storage of  $X$ . Hence the memory footprint is multiplied by 2 and the computation time by 1.5. We show in Section 5.3.3 how to design a divide-and-conquer algorithm that changes this cost to  $(2C, 1M)$ .

**Theorem 3.** *If  $(1C, 1M)$  represents respectively the computational time and memory footprint of the standard backpropagation method, Algorithm 4 costs either  $(1.5C, 2M)$  or  $(2C, 1M)$ .*

This result is of importance since it states that computing the exact curvature is at least as cheap as using numerical differences of the gradient [Castera 2022].

### 5.3.2 Proof of Algorithm 4

The goal of this section is to analyse the complexity of computing the curvature term and to prove Algorithm 4.

**Forward pass** We recall that the forward pass is computed through the recurrence

$$x_{s+1}(\Theta) = \mathcal{F}_s(x_s(\Theta), \theta_s), \quad 0 \leq s \leq n.$$

Moreover the objective function is defined as  $\mathcal{J}(\Theta) = x_{n+1}(\Theta)$ . The computation of  $X$  through the recurrence (5.13) is denoted as the *forward pass*.

**Tangent pass** Given  $\Theta$ , a set of data  $X(\Theta)$ , and an arbitrary direction  $\dot{\Theta} = (\dot{\theta}_s)_{s=0..n}$ , the tangent  $\dot{X} = (\dot{x}_s)_{s=0..n}$  is defined as

$$\dot{x}_s = \lim_{\tau \rightarrow 0} \frac{x_s(\Theta + \tau \dot{\Theta}) - x_s(\Theta)}{\tau}.$$

For each layer  $s$ , recall that  $\partial_x \mathcal{F}_s$  (resp.  $\partial_\theta \mathcal{F}_s$ ) is the differential of  $\mathcal{F}_s$  with respect to the parameter  $x$  (resp.  $\theta$ ) at the point  $(x_s(\Theta), \theta_s)$ . From now, we omit the notation of the point at which the differential is taken in order to simplify the notations. By the chain rule theorem, we have that if  $\dot{x}_s$  exists, then the forward recurrence (5.13) yields

$$\dot{x}_{s+1} = (\partial_x \mathcal{F}_s) \dot{x}_s + (\partial_\theta \mathcal{F}_s) \dot{\theta}_s, \quad \dot{x}_0 = 0. \quad (5.16)$$

A recurrence on  $s$  allows obtaining existence of  $\dot{X}$  and the scaling

$$X(\Theta + \tau \dot{\Theta}) = X(\Theta) + \tau \dot{X} + O(\tau^2).$$

Hence, if  $X(\Theta)$  is computed and  $\dot{\Theta}$  is chosen, then  $\dot{X}$  – the tangent in direction  $\dot{\Theta}$  – can be computed via the forward recurrence (5.16) and we have

$$\langle \nabla \mathcal{J}(\Theta), \dot{\Theta} \rangle = \dot{x}_{n+1}. \quad (5.17)$$

The recurrence (5.16) which allows the computation of  $\dot{X}$  is coined as the *tangent pass*.

**Adjoint/backward pass** In order to compute the gradient, one resorts to the backpropagation algorithm which allows reversing the recurrence (5.16) that defines the tangent and computing directly  $\hat{\Theta} = (\hat{\theta}_s)_{s=0..n}$  such that

$$\langle \nabla \mathcal{J}(\Theta), \hat{\Theta} \rangle = \dot{x}_{n+1} = \sum_s \langle \dot{\theta}_s, \hat{\theta}_s \rangle_{\mathcal{G}_s}.$$

The vector  $\hat{\Theta}$  is then equal to  $\nabla \mathcal{J}(\Theta)$ , provided that one uses the scalar product induced by the sum of the scalar products of all  $\mathcal{G}_s$ . To compute  $\hat{\Theta}$ , we use  $\partial_x \mathcal{F}_s^* : \mathcal{H}_{s+1} \rightarrow \mathcal{H}_s$  and  $\partial_\theta \mathcal{F}_s^* : \mathcal{H}_{s+1} \rightarrow \mathcal{G}_s$  the adjoints of the differentials of  $\mathcal{F}_s$ . The backward of the data  $\hat{X} = (\hat{x}_s)_{s=1..n+1}$  and the backward-gradient  $\hat{\Theta} = (\hat{\theta}_s)_{s=0..n}$  are defined by the reversed recurrence:

$$\begin{cases} \hat{x}_s = (\partial_x \mathcal{F}_s)^* \hat{x}_{s+1} & \text{with } \hat{x}_{n+1} = 1 \\ \hat{\theta}_s = (\partial_\theta \mathcal{F}_s)^* \hat{x}_{s+1}. \end{cases} \quad (5.18)$$

The definition of the adjoint and the formula of the tangent (5.16) give the following equality:

$$\begin{aligned}
\langle \dot{x}_{s+1}, \hat{x}_{s+1} \rangle_{\mathcal{H}_{s+1}} &= \langle (\partial_x \mathcal{F}_s) \dot{x}_s + (\partial_\theta \mathcal{F}_s) \dot{\theta}_s, \hat{x}_{s+1} \rangle_{\mathcal{H}_{s+1}} \\
&= \langle \dot{x}_s, (\partial_x \mathcal{F}_s)^* \hat{x}_{s+1} \rangle_{\mathcal{H}_{s+1}} + \langle \dot{\theta}_s, (\partial_\theta \mathcal{F}_s)^* \hat{x}_{s+1} \rangle_{\mathcal{H}_{s+1}} \\
&= \langle \dot{x}_s, \hat{x}_s \rangle_{\mathcal{H}_s} + \langle \dot{\theta}_s, \hat{\theta}_s \rangle_{\mathcal{G}_s}.
\end{aligned} \tag{5.19}$$

Summing up the above equations for every  $s$ , we obtain:

$$\dot{x}_{n+1} = \langle \dot{x}_{n+1}, \hat{x}_{n+1} \rangle_{\mathcal{H}_{n+1}} = \langle \dot{x}_0, \hat{x}_0 \rangle_{\mathcal{H}_{n+1}} + \sum_s \langle \dot{\theta}_s, \hat{\theta}_s \rangle_{\mathcal{G}_s} = \sum_s \langle \dot{\theta}_s, \hat{\theta}_s \rangle_{\mathcal{G}_s},$$

where we use  $\dot{x}_0 = 0$  and  $\hat{x}_{n+1} = 1$ . We then obtain the celebrated backward propagation formula

$$\nabla \mathcal{J}(\Theta) = \hat{\Theta}.$$

The complexity analysis of the computation of the gradient by the backward formula shows that it requires the computation and the storage of the forward pass in order to be able to evaluate  $(\partial_x \mathcal{F}_s)^*$  and  $(\partial_\theta \mathcal{F}_s)^*$  at the point  $(x_s(\Theta), \theta_s)$ .

**Computing the curvature** Equation (5.17) is the implicit definition of  $\nabla \mathcal{J}$ , where  $\dot{X}$  is defined by the recurrence (5.16). The trick of automatic differentiation is to use the the backward  $\hat{X}$  defined in recurrence (5.18) to reverse (5.16). This inversion is performed in (5.19) and it allows not computing the tangent  $\dot{X}$ . We show in this paragraph that the backward  $\hat{X}$  also reverses the recurrence defining the second order term  $\ddot{X}$  defined in (5.20) below. Once the direction  $\hat{\Theta}$  is chosen, the curvature term can be computed by only a forward pass. To this end, for any direction  $\hat{\Theta}$ , introduce  $\ddot{X} = (\ddot{x}_s)_{s=0..n+1}$  as:

$$\ddot{x}_s = \lim_{\tau \rightarrow 0} \frac{x_s(\Theta + \tau \hat{\Theta}) - x_s(\Theta) - \tau \dot{x}_s}{\tau^2}, \tag{5.20}$$

where  $\dot{x}_s$  is the tangent defined in (5.16). Recall that  $\nabla^2 \mathcal{F}_s : \mathcal{H}_s \times \mathcal{G}_s \rightarrow \mathcal{H}_{s+1}$  is the bilinear symmetric mapping that represents the second order differentiation of  $\mathcal{F}_s$  at point  $(x_s(\Theta), \theta_s)$ . It is defined as the only bilinear symmetric mapping that verifies for every  $(h_x, h_\theta)$  the relation

$$\begin{aligned}
\mathcal{F}_s(x_s(\Theta) + h_x, \theta_s + h_\theta) &= \mathcal{F}_s(x_s(\Theta), \theta_s) + \partial_x \mathcal{F}_s h_x + \partial_\theta \mathcal{F}_s h_\theta \\
&\quad + \frac{1}{2} \nabla^2 \mathcal{F}_s(h_x, h_\theta) \otimes (h_x, h_\theta) + o(\|h_x\|^2 + \|h_\theta\|^2).
\end{aligned}$$

It is easy to prove that  $\ddot{x}_s$  exists and verifies:

$$\ddot{x}_{s+1} = (\partial_x \mathcal{F}_s) \ddot{x}_s + \frac{1}{2} \nabla^2 \mathcal{F}_s(\dot{x}_s, \dot{\theta}_s) \otimes (\dot{x}_s, \dot{\theta}_s), \quad \text{with } \ddot{x}_0 = 0. \tag{5.21}$$



Indeed, denote  $\xi_s = x_s(\Theta + \tau\dot{\Theta}) - x_s(\Theta) - \tau\dot{x}_s$  so that

$$\ddot{x}_s = \lim_{\tau \rightarrow 0} \frac{\xi_s}{\tau^2},$$

we have

$$\begin{aligned} \xi_{s+1} &= \mathcal{F}_s(x_s(\Theta + \tau\dot{\Theta}), \theta_s + \tau\dot{\theta}_s) - \mathcal{F}_s(x_s(\Theta), \theta_s) - \tau(\partial_x \mathcal{F}_s)\dot{x}_s - \tau(\partial_\theta \mathcal{F}_s)\dot{\theta}_s \\ &= \mathcal{F}_s(\xi_s + x_s(\Theta) + \tau\dot{x}_s, \theta_s + \tau\dot{\theta}_s) - \mathcal{F}_s(x_s(\Theta), \theta_s) - \tau(\partial_x \mathcal{F}_s)\dot{x}_s - \tau(\partial_\theta \mathcal{F}_s)\dot{\theta}_s \\ &= (\partial_x \mathcal{F}_s)\xi_s + \frac{\tau^2}{2} \nabla^2 \mathcal{F}_s \left( \frac{\xi_s}{\tau} + \dot{x}_s, \dot{\theta}_s \right) \otimes \left( \frac{\xi_s}{\tau} + \dot{x}_s, \dot{\theta}_s \right) + o(\tau^2 + \|\xi_s\|^2). \end{aligned} \quad (5.22)$$

By a forward recurrence on (5.22), starting with  $\xi_0 = 0$ , we have that  $\xi_s = O(\tau^2)$  so that  $\ddot{x}_s$  exists. Dividing (5.22) by  $\tau^2$  and taking the limit yields (5.21).

Upon replacing  $\dot{X}$  by  $\ddot{X}$ , the trick used in (5.19) can be applied and translates into:

$$\begin{aligned} \langle \ddot{x}_{s+1}, \hat{x}_{s+1} \rangle_{\mathcal{H}_{s+1}} &= \langle (\partial_x \mathcal{F}_s)\ddot{x}_s + \frac{1}{2} \nabla^2 \mathcal{F}_s(\dot{x}_s, \dot{\theta}_s) \otimes (\dot{x}_s, \dot{\theta}_s), \hat{x}_{s+1} \rangle_{\mathcal{H}_{s+1}} \\ &= \langle \ddot{x}_s, (\partial_x \mathcal{F}_s)^* \hat{x}_{s+1} \rangle_{\mathcal{H}_{s+1}} + \frac{1}{2} \langle \nabla^2 \mathcal{F}_s(\dot{x}_s, \dot{\theta}_s) \otimes (\dot{x}_s, \dot{\theta}_s), \hat{x}_{s+1} \rangle_{\mathcal{H}_{s+1}} \\ &= \langle \ddot{x}_s, \hat{x}_s \rangle_{\mathcal{H}_s} + \frac{1}{2} \langle \nabla^2 \mathcal{F}_s(\dot{x}_s, \dot{\theta}_s) \otimes (\dot{x}_s, \dot{\theta}_s), \hat{x}_{s+1} \rangle_{\mathcal{H}_{s+1}} \end{aligned}$$

Summing up these equations in  $s$  and using  $\ddot{x}_0 = 0$  and  $\hat{x}_{n+1} = 1$ , we obtain

$$\begin{aligned} \ddot{x}_{n+1} &= \langle \ddot{x}_{n+1}, \hat{x}_{n+1} \rangle_{\mathcal{H}_{n+1}} \\ &= \langle \ddot{x}_0, \hat{x}_0 \rangle_{\mathcal{H}_{n+1}} + \sum_s \frac{1}{2} \langle \nabla^2 \mathcal{F}_s(\dot{x}_s, \dot{\theta}_s) \otimes (\dot{x}_s, \dot{\theta}_s), \hat{x}_{s+1} \rangle_{\mathcal{H}_{s+1}} \\ &= \sum_s \frac{1}{2} \langle \nabla^2 \mathcal{F}_s(\dot{x}_s, \dot{\theta}_s) \otimes (\dot{x}_s, \dot{\theta}_s), \hat{x}_{s+1} \rangle_{\mathcal{H}_{s+1}}. \end{aligned}$$

In order to conclude and prove Algorithm 4, it is sufficient to remark that  $\mathcal{J}(\Theta) = x_{n+1}(\Theta)$  so that  $\frac{1}{2} \langle \nabla^2 \mathcal{J}(\Theta) \dot{\Theta}, \dot{\Theta} \rangle = \ddot{x}_{n+1}$ .

**More on automatic differentiation** In Section 5.6.1, the reader will find a method to compute the matrix-vector product with the Hessian. This method is not new and is known as the *Pearlmutter's trick* [Christianson 1992, Pearlmutter 1994]. We prove this trick in our setting in order to link our computations with other automatic-differentiation techniques. Moreover, we also give some of the expression of  $\nabla^2 \mathcal{F}_s(\dot{x}_s, \dot{\theta}_s) \otimes (\dot{x}_s, \dot{\theta}_s)$  for standard layers in Section 5.6.2 to settle the notations.

### 5.3.3 Proof of Theorem 3

Recall that  $(1C, 1M)$  is the complexity of a gradient computation, we show how to change the overall cost of computing the curvature from  $(1.5C, 2M)$  to  $(2C, 1M)$

by a divide-and-conquer algorithm. In order to simplify the analysis, several simplifications are made.

- There are three kind of passes, the forward pass in (5.13) that computes  $X$ , the backward pass in (5.18) that computes  $\hat{X}$  and  $\hat{\Theta}$  and the tangent-curvature pass described in Algorithm 4 that computes the curvature. We suppose that each of these passes have roughly the same computational cost  $C/2$ . This assumption is subject to discussion. In one hand the backward and tangent passes require each twice as much matrix multiplication as the forward pass. On the other hand, soft activation functions are harder to compute in the forward pass.
- We assume that storing  $X$  or  $\hat{X}$  has the same memory footprint  $1M$ . Notably, we suppose that the cost of storing the parameters  $\Theta$  or the gradient  $\hat{\Theta}$  is negligible with respect to the storage of the data through the network. This assumption can only be made for optimization with large enough batches  $\mathcal{B}_k$ .
- We suppose that we can divide the neural network in two pieces that each costs half the memory and half the computational time. This means that we are able to find  $L$ , such that the storage of  $(x_s)_{s \leq L}$  and the storage of  $(x_s)_{s \geq L}$  have same memory footprint  $M/2$ . Moreover we suppose that performing a pass for  $s \geq L$  or for  $s \leq L$  costs  $C/4$  computational time. This assumption is reasonable and simplifies the analysis but it is of course possible to exhibit pathological networks that won't comply with this assumption.
- We suppose that the only cost in data transfer comes from the initialization of the parameters  $\Theta$ , the initial data  $x_0$  and the direction of descent  $\dot{\Theta}$ . Note that the computation of  $\hat{\Theta}$  requires the computation of the gradient  $\dot{\Theta}$ .

We now describe how to compute the curvature with  $(2C, 1M)$  and no extra data transfer. We display the current memory load and the elapsed computational time at the end of each phase. A visual illustration of this algorithm is proposed in Figure 5.1.

0. Transfer the data  $x_0$  and  $\Theta$ .
1. Compute  $X = (x_s)_s$  and store it. For  $s \geq L$ , compute the backward via (5.14) without storing it. Cost is  $(\frac{3}{4}C, 1M)$
2. Flush from memory  $(x_s)_{s \geq L}$ . Cost is  $(\frac{3}{4}C, \frac{1}{2}M)$
3. For  $s \leq L$ , compute the backward via (5.14) and store it. Cost is  $(1C, 1M)$
4. Choose the descent direction and transfer the data  $\dot{\Theta}$ . Compute the tangent via (5.16) for  $s \geq L$ . Cost is  $(\frac{5}{4}C, M)$
5. Flush from memory  $(\hat{x}_s)_s$  and  $(x_s)_{s < L}$ . Cost is  $(\frac{5}{4}C, 0M)$
6. For  $s \geq L$ , compute the forward, the backward and store them. Compute the tangent for  $s \geq L$ . Cost is  $(2C, 1M)$

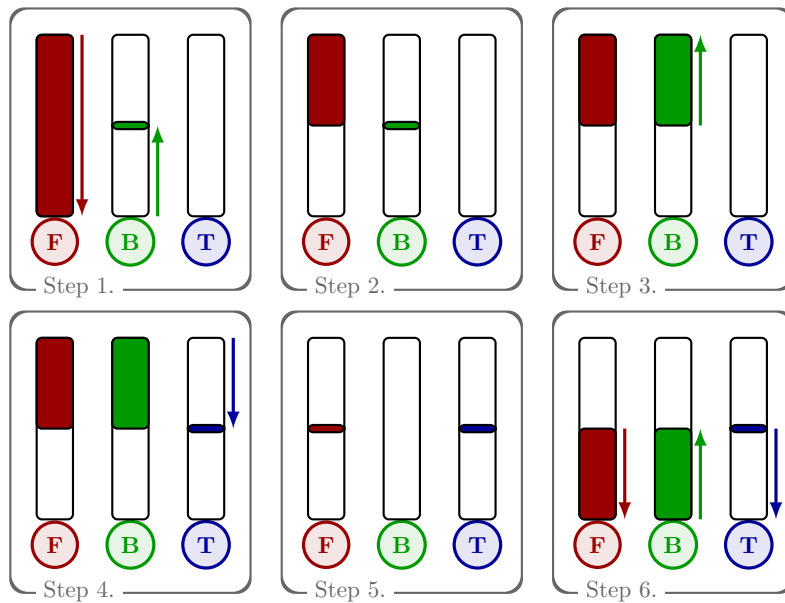


Figure 5.1: Illustration of the divide-and-conquer algorithm that changes the cost of computing the curvature from  $(1.5C, 2M)$  to  $(2C, 1M)$ . The rectangles above the letters **F**, **B**, **T** represent the three different passes (in order: forward, backward and tangent). The memory usage is represented by color-filling in the rectangles, the computations are represented by arrows on the right of the passes. In total, the filled area never exceeds 1 rectangle, hence memory usage is  $1M$ . The total length of the arrows is 4 times the length of a rectangle, this represents 4 passes. The computational time is then twice the computational time of the standard backward algorithm.

## 5.4 Numerical experiments

### 5.4.1 Convergence/exploration trade off

#### 5.4.1.1 The RED algorithm

In order to test the convergence/exploration trade-off, we reproduce the benchmark of [Castera 2022]. We set ourselves in the case where the initial parameters are randomly chosen, so that the practitioner wants a smooth transition from exploration ( $\ell = 1$ ) to convergence ( $\ell = \frac{1}{2}$ ). We choose in Algorithm 5 a simple, per epoch, exponential decay rule of the learning rate  $\ell$  from 1 to  $1/2$ . This algorithm is coined as RED (Rescaled with Exponential Decay). We purposely unplug any other tricks of the trade, notably Robbins-Monro convergence conditions. Indeed, a Robbins-Monro decay rule would interfere with our analysis. Algorithm RED is not a production algorithm, it serves at testing the “natural” convergence properties of rescaling. In Section 5.8.4, we provide a comparison of RED with a standard SGD that has Robbins-Monro decaying conditions. Due to the remark in Section 5.2.2, we make clear that  $L^2$ -regularization is used. If  $\Theta \mapsto \mathcal{L}_s(\Theta)$  is the original loss function, then the function  $\mathcal{J}_s$  is defined as  $\mathcal{J}_s(\Theta) \stackrel{\text{def}}{=} \mathcal{L}_s(\Theta) + \frac{\lambda}{2}\|\Theta\|^2$ .

---

**Algorithm 5** RED (rescaled-exponential-decay) for SGD or RMSProp preconditioning **and no convergence guaranty**

---

- 1: **Input parameters**  $\beta_2 = 0.999$  (RMSProp parameter), RMSProp (boolean),  $\lambda > 0$  ( $L^2$ -regularization),  $N$  (total number of epochs),  $\varepsilon = 10^{-8}$  (numerical stabilization).
  - 2: **Initialization**  $\hat{v}_0 = 0$ ,  $\Theta_0$  random,  $\ell = 1$  initial learning rate and  $\eta = 1/2$  the step multiplicative factor between the first and the last iterations.
  - 3: **for**  $k = 1, 2, \dots$  **do**
  - 4:      $g_k = \mathbb{E}_{s \in \mathcal{B}_k} [\nabla \mathcal{J}_s(\Theta_k)]$  ▷ gradient
  - 5:     **if** RMSProp **then** ▷ RMSProp preconditioning
  - 6:          $\hat{v}_k = \beta_2 \hat{v}_{k-1} + (1 - \beta_2) g_k^2$    and    $\tilde{v}_k = \hat{v}_k / (1 - \beta_2^k)$  and  $P_k = \text{diag}(\sqrt{\tilde{v}_k} + \varepsilon)$
  - 7:     **else**
  - 8:          $P_k = \text{Id}$
  - 9:     **end if**
  - 10:      $\dot{\Theta}_k = P_k^{-1} g_k$  ▷ direction of update
  - 11:     **Use Algorithm 3 and compute**  $r_k$  ▷ rescaling
  - 12:      $\Theta_{k+1} = \Theta_k - \ell r_k \dot{\Theta}_k$  ▷ parameters update
  - 13:     At the end of each epoch  $\ell \leftarrow \eta^{\frac{1}{N}} \ell$
  - 14: **end for**
- 

The numerical experiments are done on the benchmark of [Castera 2022]. It consists of four test cases, a MNIST classifier [LeCun 2010], a CIFAR-10 classifier [Krizhevsky 2009] with VGG11 [Simonyan 2015] architecture, a CIFAR-100 classifier with VGG19 and the classical autoencoder of MNIST described in [Hinton 2006]. The ReLU units are replaced by smooth versions ( $\mathcal{C}^2$  functions) in order to compute the curvature term, and  $L^2$  regularization is added to each test.

The models are trained with a batch size of 256 and the number of epochs is set to 200 for MNIST classification and 500 for the others. The precise set of parameters that allows reproducibility is described in Section 5.7. We also give indications of the computational time on an NVIDIA Quadro RTX 5000. Each experiment is run 3 times with different random seeds and we display the average of the tests with a bold line, the limits of the shadow area are given by the maximum and the minimum over the runs. When displaying the training loss or the step histories, an exponential moving average with a factor 0.99 is applied in order to smooth the curves and gain in visibility. Note that the training and testing loss functions are displayed with the  $L^2$  regularization term. On all figures the  $x$ -axis is the number of epochs. Remember however that the computational cost is not the same for the different optimizers, see Theorem 3.

#### 5.4.1.2 Interpretation of the RED experiments

In this first set of experiments, we compare the RED method given in Algorithm 5 with standard SGD and RMSProp. In order to recover these two latter algorithms, set  $r_k = 1$  in line 11 of Algorithm 5. The hyperparameters, namely the initial learning rate  $\ell$  and its decay factor  $\eta$ , are optimized on the training loss with a grid search over the 20% first epochs, these algorithms are coined as “standard algorithms”. The results are displayed in Figure 5.2 for the standard algorithms (orange for SGD, blue for RMSProp) and their RED version (red for SGD, green for RMSProp).

**Training loss** The analysis of the training loss shows that RED is competitive to the standard SGD and RMSProp methods. Note however that the hyperparameters of the standard methods have been chosen as to optimize the behavior of the training loss, hence we cannot expect the RED method to outperform the manually-tuned methods.

**Step** We always observe an increase in the step for the first few epochs (50 for MNIST, 10 for CIFAR). This step increase coincides with the important decrease of the training and testing loss functions. We interpret this behavior as a search for a basin of attraction of a local minimum. It should be noted that the step of the standard algorithms on CIFAR100 and on the autoencoder is an order of magnitude smaller than their RED counterpart. Indeed larger steps on these methods cause the algorithm to diverge. This seems to indicate that the stage of the first 10 epochs where the step is small is of importance and is well captured by the RED algorithm. Note that this behavior is the one that is implemented when using warm-up techniques [Loshchilov 2017]. The analysis of the step seems to showcase the power of adaptive rescaling and indicates that warm-up techniques can be handled by the rescaling. This potential is investigated in Section 5.4.2.

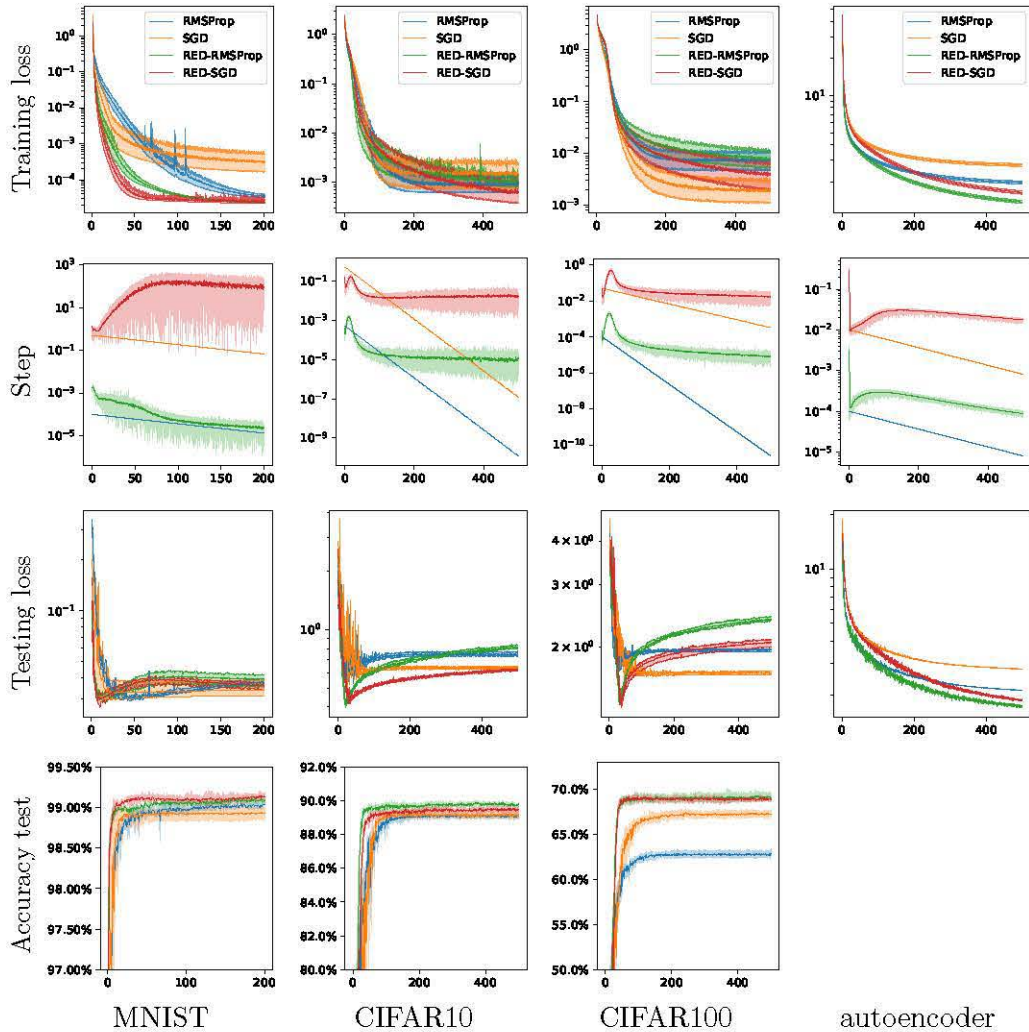


Figure 5.2: Training loss, step size, testing loss and test accuracy for the RED and manually-tuned SGD and RMSProp optimizers. Each column gives the different test cases (resp. MNIST, CIFAR10, CIFAR100 and autoencoder). The RED method which has no tuning gives competitive results in comparison with the manually-tuned SGD and RMSProp optimizers.

**Testing loss and accuracy** The rescaling aims at minimizing quickly the training loss, no conclusions can be drawn from the analysis of the test dataset. Nevertheless, on the CIFAR experiments, an overfitting phenomenon starts from the 25<sup>th</sup> epoch approximatively. The overfitting is clearer and more pronounced on the RED method. This is in accordance with the analysis of the step size: the rescaled method seems to have converged to the maximum of the expressivity of the network at the 50<sup>th</sup> epoch. Concerning the accuracy, it is well known that adaptive methods have poor generalization performances in the overparameterized setting in comparison to SGD [Wilson 2017]. Indeed the standard RMSProp achieves lower performance on the test dataset of CIFAR100. Surprisingly, the RED-RMSProp algorithm does not have this property.

As a conclusion of these tests, RED, which is a naive implementation of convergence/exploration trade-off works surprisingly well on this benchmark. We purposely disconnected Robbins-Monro decay rule and let the algorithm run way past overfitting. It still exhibits good convergence properties.

### 5.4.1.3 Other numerical tests

In Section 5.8.1 we investigate the use of momentum with SGD and Adam on the CIFAR100 classifier. The proposed method is only available to deal with direction of descent and the directions of update given by momentum based algorithms are not necessarily direction of descent, yielding poor convergence results.

In Section 5.8.2, we perform tests with smaller batches and we exhibit pathological cases where the rescaled method is highly impacted by stochasticity. The main conclusion is that the performance of the method collapses when the batch is too small compared to the number of classes. This problem in the curvature computation arises at the last layer of the neural network (linear classifier).

In Section 5.8.3 we study the effect of the  $L_2$  regularization on the CIFAR10 classifier, showing numerically that the potential theoretical issues raised in Section 5.2.2 do not impede convergence.

Finally, in Section 5.8.4 we perform some comparisons with an existing BB method [Castera 2022] and with a SGD with Robbins-Monro decay condition.

## 5.4.2 Hyperexploration mode

In order to showcase hyperexploration, we propose a vanilla annealing method. We replace in Algorithm 5 (RED) the line 13 (update of the parameter  $\ell$ ) by setting periodically  $\ell = 1$  for 5 epochs,  $\ell = \frac{1}{2}$  for 13 epochs and  $\ell = 2$  for 2 epochs. These three phases are coined respectively as *exploration*, *convergence* and *hyperexploration*. We favor sharp changes when letting  $\ell$  oscillate in order to easily interpret the results. This simple annealing method is coined RAn (Rescaled Annealing). We display in Figure 5.3 the results for CIFAR10 and CIFAR100. On Figure 5.3 the shift between the choice  $\ell = \frac{1}{2}$  and  $\ell = 2$  is represented by a vertical gray line. We also display the results for the RED algorithm for comparison. Of

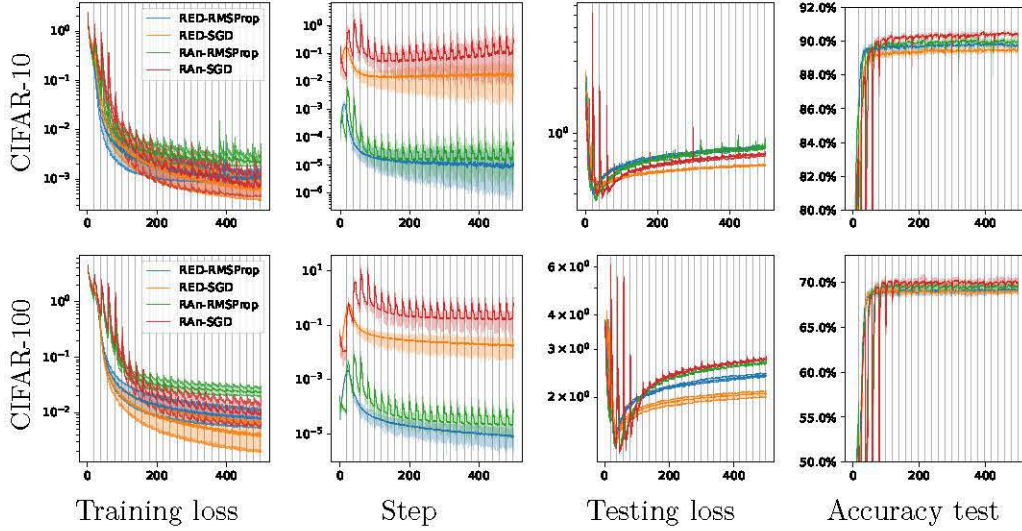


Figure 5.3: Annealing (RAn) vs Exponential decay (RED) method. The annealing method increases the loss functions during the hyperexploration ( $\ell = 2$ ) phase (after the vertical gray lines). This empirically proves that the factor  $\ell = 1$  is the limiting factor that allows exploration without increasing the loss function. The basin of attraction of the RAn method is different of the one of RED, except possibly for CIFAR10 with RMSProp.

importance in Figure 5.3 is the behavior of the loss function. The latter increases at each *hyperexploration* phase, and converges during the *exploration* and *convergence* phase. A similar effect is also present but less pronounced on the testing loss and accuracy. The increase of the training loss function for  $\ell = 2$  is in accordance with the theory, and is at the core of annealing methods that aim at escaping local minima. These tests validate the fact that  $\ell = 1$  is an upper-bound for the *exploration* choice.

### 5.4.3 Hyperconvergence mode

In this example, we wish to study a more realistic dataset for which stochastic issues are of essence. To that end, we use the ImageNet 1K database and load a state-of-the-art pretrained ResNet-50. This network achieves a 80.858% top-1 accuracy and a 95.434% top-5 accuracy. From the study of Section 5.8.2, summarized in Section 5.4.1.3, we know that important stochastic problems will occur in the last layer (Linear Classifier or LC) of the DNN. Hence, we erase the parameters of the linear classifier and aim at re-training it while freezing the weights of the feature extractor (upstream section of the network). This setting is reminiscent of a *toy transfer learning* problem and aims at training a simple neural network with a state-of-the-art dataset.

From the coupon collector’s problem with 1000 classes, we know that the expectation of  $T$ , the smallest batch size that obtains at least one element in each



class, is approximatively 7.3K when classes are drawn independently and uniformly. We expect stochastic issues to appear when batches are of size smaller than 7.3K. The batch size used to pretrain the network is 1K, hence we test the rescaling for batch of size 1K, 2K, 4K, 8K and 16K. Since stochastic effects should be seriously mitigated for the 8K and 16K cases, these two cases represent a baseline for the training.

We first discard every trick and test the rescaling for the different batch size. We adopt a fixed rescaled learning rate strategy of  $\ell = \frac{1}{2}$  in order to converge as fast as possible. The result is given in Figure 5.4 top line and referred as *plain training*. Of importance in the top line of Figure 5.4 are the oscillations in the training loss, which are less pronounced as the batch size increases. Note also the stability of the top-1 and top-5 accuracies around a value that depends on the batch size. For the 16K experiment, the linear classifier achieves the top-1 and top-5 accuracies of the pre-trained weights.

We then implement several tricks of the trade, namely repeated augmentation (RA) [Hoffer 2019] and label smoothing (LS) [Szegedy 2016]. The result are displayed in Figure 5.4, middle line. These two tricks do not seem to have any effect on the training of the linear classifier.

In the bottom line of Figure 5.4, we implement a decrease of the learning rate with a Cosine annealing (Cos) technique, in addition to (RA) and (LS). The (Cos) technique reduces the learning rate and enforces the *hyperconvergence mode*. As far as the accuracies are concerned, reducing the learning rate allows the algorithm to converge when the batches are small and is useless when the batch size is greater than 8K. This test corroborates the findings of [Smith 2018] and the tests of Section 5.8.2.

In this benchmark, one of the important advantages of rescaling is to be able to perform several tests (batch reduction, repeated augmentation, label smoothing) without having to set the learning rate for each test.

#### 5.4.4 Influence of the averaging factor of the curvature

This section is dedicated to the study of the impact of the averaging factor of the curvature  $\beta_3$  on the algorithm. A low value  $\beta_3 \simeq 0$  yields an estimation of the curvature that is less dependent of the past iterations at the expense of having a higher variance. A value close to 1 results in a low variance estimation but that has a bias due to old iterations. In Figure 5.5, the CIFAR10 classifier is optimized using RED-SGD with values of  $\beta_3 \in \{0, 0.5, 0.9, 0.99\}$ . Interestingly, the parameter that gives the fastest increase of the test accuracy is  $\beta_3 = 0$  at the cost of more instabilities. Although higher values of  $\beta_3$  lead to an underestimation of the step size, the difference of performance on the training loss is insignificant. Overall, a value  $\beta_3 \in [0.5, 0.99]$  has little impact on the convergence rate of the algorithm and a default value of  $\beta_3 = 0.9$  can be considered.

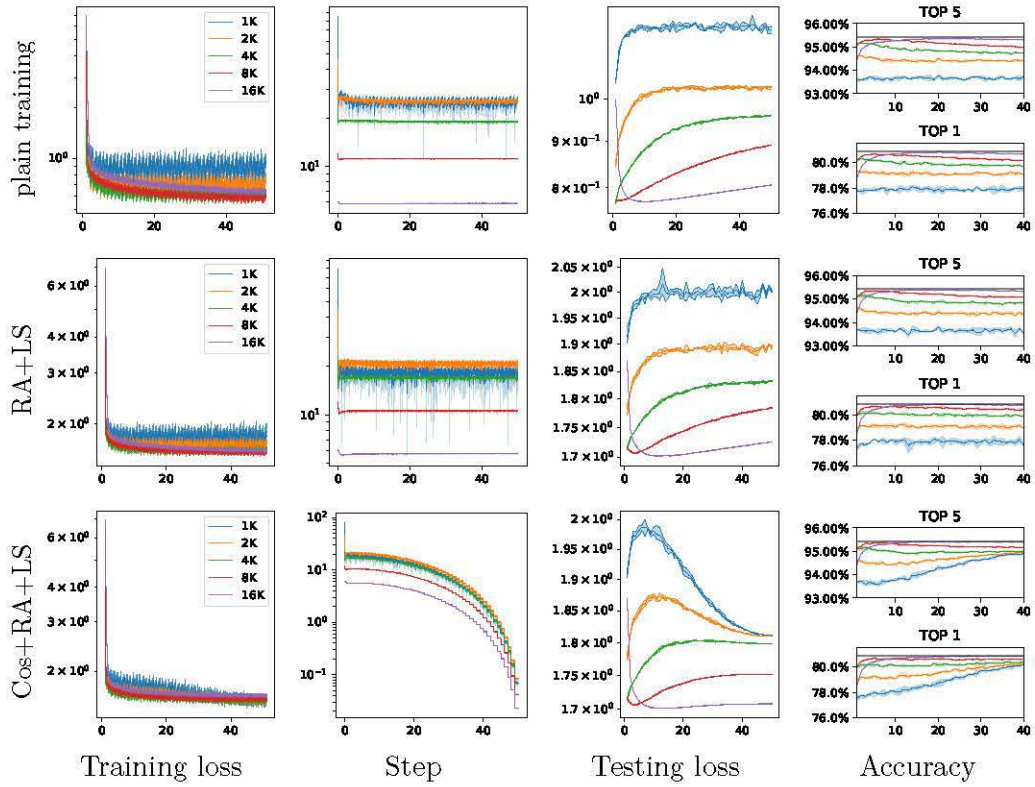


Figure 5.4: Training a linear classifier on ImageNet 1K with a ResNet-50 feature extractor and with different batch size for SGD.

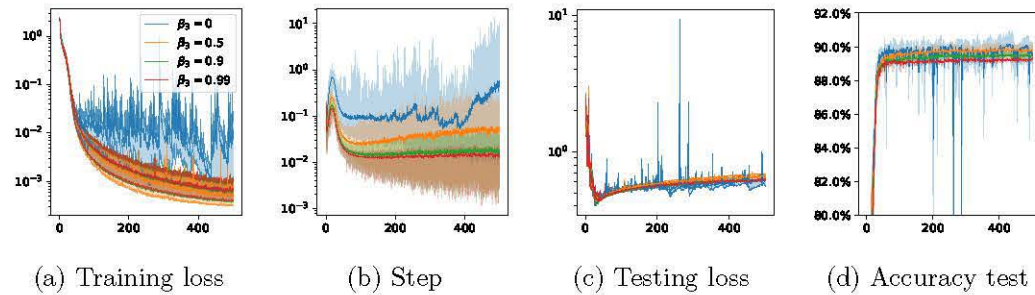


Figure 5.5: Training loss, step size, testing loss and test accuracy on the CIFAR10 classifier with RED-SGD. The tests are conducted with different values of the curvature averaging parameter  $\beta_3$ . A value  $\beta_3 = 0$  yields instability and  $\beta_3 \in [0.5, 0.99]$  has little impact on the convergence rate.

## 5.5 Conclusion and discussion

We developed a framework that allows automatic rescaling of the learning rate of a descent method with the use of the curvature, which is an easily affordable second order information computed by automatic differentiation. This rescaling yields a data and direction adapted learning rate with a physical meaning. The practitioner can choose the behavior of the algorithm by setting the value of this rescaled learning rate. A value between 1/2 and 1 results in convergence, a value above 1 yields hyperexploration of the space of parameters and a value below 1/2 enforces convergence when stochasticity is of importance.

In the numerical examples of Section 5.4.1 a choice of exponential decrease is competitive to simple manual tuning of the learning rate in the case of SGD and RMSProp preconditioning. In Section 5.4.2, we show that the choice  $\ell > 1$  allows escaping basin of attraction of local minima. The more intricate benchmark of Section 5.4.3 show that rescaling doesn't save us from reducing the learning rate but that it allows to control the environment and compare different experiments.

The main limitation of this method is that it does not allow use of momentum. Indeed momentum methods do not necessarily yield directions of descent and do rely on per-iteration minimization of Lyapunov functions [Polyak 2017]. Implementing momentum methods with curvature computation is a challenge reserved for future works. Another drawback is the need to use  $\mathcal{C}^2$  activation functions, notably excluding ReLU. Finally, the curvature computation, also affordable in theory, requires additional implementations on top of ready-to-use machine learning libraries, which restricts, for now, our method to rather simple networks.

## 5.6 Second order computation

### 5.6.1 Hessian-vector dot product

In this section, we turn our attention to showing how to compute  $\nabla^2 \mathcal{J}(\Theta) \dot{\Theta}$  in our setting. The results are known as the *Pearlmutter's trick* [Christianson 1992, Pearlmutter 1994]. We emphasize that the computation of the curvature is simpler than the Hessian-vector product. In our setting, the trick that allows the computation of the Hessian-vector product is based on the following ideas

- The mapping  $\dot{\Theta} \mapsto \frac{1}{2} \langle \nabla^2 \mathcal{J}(\Theta) \dot{\Theta}, \dot{\Theta} \rangle$  is bilinear. If we differentiate with automatic differentiation this mapping with respect to  $\dot{\Theta}$ , we retrieve  $\nabla^2 \mathcal{J}(\Theta) \dot{\Theta}$ .
- Because  $X(\Theta)$  is fixed, the aforementioned mapping is defined by a single forward recurrence. Hence, only one additional backward recurrence should be sufficient to compute  $\nabla^2 \mathcal{J}(\Theta) \dot{\Theta}$ .

In order to make explicit this backward recurrence, we need to introduce two vectors  $A_s \in \mathcal{H}_s$  and  $B_s \in \mathcal{G}_s$  that are defined by the implicit equation:

$$\langle A_s, a \rangle_{\mathcal{H}_s} + \langle B_s, b \rangle_{\mathcal{G}_s} = \langle \nabla^2 \mathcal{F}_s(\hat{x}_s, \hat{\theta}_s) \otimes (a, b), \hat{x}_{s+1} \rangle_{\mathcal{H}_{s+1}} \quad \forall (a, b) \in \mathcal{H}_s \times \mathcal{G}_s.$$

The existence and uniqueness of  $(A_s, B_s)$  is just Riesz theorem applied to the linear form on  $\mathcal{H}_s \times \mathcal{G}_s$ :

$$(a, b) \mapsto \langle \nabla^2 \mathcal{F}_s(\dot{x}_s, \dot{\theta}_s) \otimes (a, b), \hat{x}_{s+1} \rangle_{\mathcal{H}_{s+1}}.$$

Construct  $\tilde{X} = (\tilde{x}_s)_s$  and  $\tilde{\Theta} = (\tilde{\theta}_s)_s$  by a backward recurrence using

$$\begin{cases} \tilde{x}_s = (\partial_x \mathcal{F})^* \tilde{x}_{s+1} + A_s & \text{with } \tilde{x}_{n+1} = 0 \\ \tilde{\theta}_s = (\partial_\theta \mathcal{F})^* \tilde{x}_{s+1} + B_s. \end{cases} \quad (5.23)$$

Then we have

$$\nabla^2 \mathcal{J}(\Theta) \dot{\Theta} = \tilde{\Theta}. \quad (5.24)$$

In order to prove (5.24), we show that for any other direction  $\dot{\Theta}'$ , we have

$$\langle \nabla^2 \mathcal{J}(\Theta) \dot{\Theta}, \dot{\Theta}' \rangle = \langle \tilde{\Theta}, \dot{\Theta}' \rangle.$$

First consider  $\dot{X}'$  the tangent associated with direction  $\dot{\Theta}'$ . We have by bilinearity of  $\nabla^2 \mathcal{F}_s$  and by Algorithm 4 that

$$\langle \nabla^2 \mathcal{J}(\Theta) \dot{\Theta}, \dot{\Theta}' \rangle = \sum_s \langle \nabla^2 \mathcal{F}_s(\dot{x}_s, \dot{\theta}_s) \otimes (\dot{x}'_s, \dot{\theta}'_s), \hat{x}_{s+1} \rangle_{\mathcal{H}_{s+1}} = \sum_s \langle A_s, \dot{x}'_s \rangle + \langle B_s, \dot{\theta}'_s \rangle \quad (5.25)$$

By definition of  $\tilde{X}$  and  $\tilde{\Theta}$  in (5.23) and by formula (5.16) for the tangent  $\dot{X}'$ , the following equality holds:

$$\begin{aligned} \langle \tilde{x}_s, \dot{x}'_s \rangle + \langle \tilde{\theta}_s, \dot{\theta}'_s \rangle - \langle A_s, \dot{x}'_s \rangle - \langle B_s, \dot{\theta}'_s \rangle \\ = \langle (\partial_x \mathcal{F})^* \tilde{x}_{s+1}, \dot{x}'_s \rangle + \langle (\partial_\theta \mathcal{F})^* \tilde{x}_{s+1}, \dot{\theta}'_s \rangle = \langle \tilde{x}_{s+1}, \dot{x}'_{s+1} \rangle \end{aligned}$$

Summing up the above equation for every  $s$ , using  $\tilde{x}_{n+1} = 0$ ,  $\dot{x}'_0 = 0$  and (5.25) yields (5.24)

### 5.6.2 Structure of the layers

In this section, we explain how to compute the curvature for some of the standard layers used in DNNs. First, we make clear the different kind of layers we use:

- **Loss layers** are parameter-free layers from  $\mathcal{H}_n$  to  $\mathbb{R}$ , they are denoted by  $\mathcal{L}$

$$\mathcal{F}_n(x, \theta) = \mathcal{L}(x).$$

- **Smooth activation layers** do not have parameters and are such that  $\mathcal{H}_{s+1} = \mathcal{H}_s$ . They are defined coordinate-wise through a smooth function  $\Phi_s : \mathbb{R} \rightarrow \mathbb{R}$  with

$$\mathcal{F}_s(x, \theta)[i] = \Phi_s(x[i]) \quad \forall i.$$

- **Linear layers or convolutional layers.** The set of parameters are the weights (or kernel) denoted  $\theta$ . We suppose that these layers have no bias. They are abstractly defined as

$$\mathcal{F}_s(x, \theta)[i] = \sum_{k,j} \theta[k]x[j]\mathbb{1}_{ijk}$$

where  $i$  (resp  $j, k$ ) denotes the sets of indices of the outputs (resp. the input, the weights). The function  $(i, j, k) \mapsto \mathbb{1}_{ijk}$  represents the assignment of the multi-index  $(k, j)$  to  $i$ . This affectation is either equal to 1 or 0, that is  $(\mathbb{1}_{ijk})^2 = \mathbb{1}_{ijk}$ .

- **Bias layers** are layers where  $\mathcal{H}_s = \mathcal{H}_{s+1}$  and are defined by

$$\mathcal{F}_s(x, \theta)[i] = x_s[i] + \sum_k \mathbb{1}_{ik}\theta[k].$$

They are often concatenated with linear or convolutional layers. There is no restriction to split a biased linear layer into the composition of a linear layer and a bias layer.

- **Batch normalization layers.** We split a batch normalization layer into the composition of four different layers, the centering layer, the normalizing layer, a linear layer with diagonal weight matrix and a bias layer. For each output index  $i$ , the centering and normalizing layers are defined by an expectation over the batch and some input indices. This expectation is denoted as  $\mathbb{E}_i$ . The centering layer can be written as

$$\mathcal{F}_s(x, \theta)[i] = x_s[i] - \mathbb{E}_i(x_s).$$

The normalizing layer is defined as

$$\mathcal{F}_s(x, \theta)[i] = \frac{x[i]}{\sqrt{\mathbb{E}_i(x^2) + \varepsilon}}.$$

For the different layers, we give the formula for the different recurrences in Table 5.1. We begin with the classic backward computations, they are mainly given here to settle the notations.

## 5.7 Description of the numerical experiments

All the experiments were conducted and timed using Python 3.8.11 and PyTorch 1.9 on an Intel(R) Xeon(R) W-2275 CPU @ 3.30GHz with an NVIDIA Quadro RTX 5000 GPU. We also used the Jean-Zay HPC facility for additional runs.

The models are trained with a batch size of 256 so that one epoch corresponds to 235 iterations for MNIST and 196 for CIFAR. The number of epochs is set to

Name	$x_{s+1}[i]$	$\hat{x}_s[j]$	$\hat{\theta}_s[k]$
Activation	$\Phi(x_s[i])$	$\Phi'(x_s[j])\hat{x}_{s+1}[j]$	N.A.
Linear	$\sum_{k,j} \theta[k]x_s[j]\mathbf{1}_{ijk}$	$\sum_{k,i} \theta[k]\hat{x}_{s+1}[i]\mathbf{1}_{ijk}$	$\sum_{j,i} \hat{x}_{s+1}[j]x_s[i]\mathbf{1}_{ijk}$
Bias	$x_s[i] + \sum_k \mathbf{1}_{ik}\theta[k]$	$\hat{x}_{s+1}[j]$	$\sum_i \mathbf{1}_{ik}\hat{x}_{s+1}[i]$
Centering	$x_s[i] - \mathbb{E}_i(x_s)$	$\hat{x}_{s+1}[j] - \mathbb{E}_j(\hat{x}_{s+1})$	N.A.
Normalizing	$\begin{cases} \gamma = (\mathbb{E}_i(x_s^2) + \varepsilon)^{-1/2} \\ x_{s+1}[i] = \gamma x_s[i] \end{cases}$	$\gamma \hat{x}_{s+1}[j] - x_s \mathbb{E}_j[\gamma^3 x_s \hat{x}_{s+1}]$	N.A.

---

Name	$\dot{x}_{s+1}[i]$	$r_s = \frac{1}{2} \langle \nabla^2 \mathcal{F}_s(\dot{x}_s, \dot{\theta}_s) \otimes (\dot{x}_s, \dot{\theta}_s), \hat{x}_{s+1} \rangle_{\mathcal{H}_{s+1}}$
Activation	$\Phi'(x_s[i])\dot{x}_s[i]$	$\frac{1}{2} \sum_i \Phi''(x_s[i])\dot{x}_s^2[i]\hat{x}_{s+1}[i]$
Linear	$\sum_{k,j} (\theta[k]\dot{x}_s[j] + \dot{\theta}[k]x_s[j]) \mathbf{1}_{ijk}$	$\sum_{k,j,i} \dot{\theta}[k]\dot{x}_s[i]\hat{x}_{s+1}[j]\mathbf{1}_{ijk}$
Bias	$\dot{x}_s[i] + \sum_k \mathbf{1}_{ik}\dot{\theta}[k]$	0
Centering	$\dot{x}_s[i] - \mathbb{E}_i(\dot{x}_s)$	0
Normalizing	$\begin{cases} \dot{\gamma} = -\mathbb{E}_i(\dot{x}_s x_s)\gamma^3 \\ \dot{x}_{s+1}[i] = \gamma \dot{x}_s[i] + \dot{\gamma} x_s[i] \end{cases}$	$\begin{cases} \dot{\gamma} = -\mathbb{E}_i(\dot{x}_s^2)\gamma^3 + 3\mathbb{E}_i(\dot{x}_s x_s)\gamma^5 \\ r_s = \sum_i \frac{1}{2} (\dot{\gamma} \dot{x}_s[i] + \dot{\gamma} x_s[i]) \hat{x}_{s+1}[i] \end{cases}$

Table 5.1: Quantities needed in the forward, backward and second order passes for standard layers.

200 for the MNIST classification and 500 for the others. Table 5.2 summarizes the characteristics of the datasets used.

Concerning the tuning of the standard methods, the step size and its decay factor were searched on a grid for the SGD and RMSProp methods. The learning rate is constant per epoch and its value at the  $n^{\text{th}}$  epoch is given by

$$\tau_n = \tau_0 d^n.$$

We searched amongst the values  $\tau_0 \in \{1 \times 10^n, 5 \times 10^n\}_{-5 \leq n \leq 1}$  for the step size and  $d \in \{0.97, 0.98, 0.99, 1\}$  for the step decay on MNIST classification and  $d \in \{0.99, 0.995, 1\}$  for the others. After 20% of the total number of epochs, the couple  $(\tau_0, d)$  that achieves the best training loss decrease is chosen.

For the CIFAR experiments we used data augmentation with a random crop and an horizontal flip. In the CIFAR100 training we added a random rotation of at most  $\pm 15^\circ$ .

For reproducibility, the values used in the experiments are summarized in Table 5.3. Unless explicitly stated, these are the default values used in the experiments of this work. The computing time per epoch is reported in Table 5.3 for each method. The codes of RED are not optimized, especially for the convolution layers where the backward with respect to the parameters is implemented by an additional run of the forward. This explains why RED is twice slower than the standard methods on the CIFAR classifiers which make intensive use of convolution layers.

Dataset	MNIST	CIFAR10	CIFAR100
License	CC BY-SA 3.0	MIT License	Unknown
Size of the training set	60000	50000	50000
Size of the testing set	10000	10000	10000
Number of channels	1	3	3
Size of the images	$28 \times 28$	$32 \times 32$	$32 \times 32$
Number of classes	10	10	100

Table 5.2: Summary of the datasets used.

Type of problem	MNIST classification	CIFAR10 classification	CIFAR100 classification	MNIST autoencoder
Type of network	LeNet Dense	VGG11 Convolutional	VGG19 Convolutional	Dense
Activation functions	Tanh	SoftPlus $\beta = 5$	SoftPlus $\beta = 5$	ELU
$L^2$ regularization	$\lambda = 10^{-7}$	$\lambda = 10^{-7}$	$\lambda = 10^{-7}$	$\lambda = 10^{-7}$
Loss function	Cross entropy	Cross entropy	Cross entropy	MSE
Number of epoch	200	500	500	500
Batch size	256	256	256	256
Number of epoch for tuning	40	100	100	100
Iteration per epoch	196	235	235	196
Computing time per epoch with the standard SGD / RMSProp	5.3s / 5.4s	16.4s / 16.8s	36.3s / 36.9s	5.1s / 5.3s
Computing time per epoch with RED-SGD / RED-RMSProp	7.6s / 7.7s	30.1s / 30.5s	65s / 65s	5.7s / 5.9s

Table 5.3: Summary of the experiment parameters.

## 5.8 Additional numerical experiments

### 5.8.1 Dealing with momentum

In Section 5.4.1, only stochastic optimizers without momentum are presented. In this section, we discuss the extension of our algorithm to momentum based update directions, notably momentum with RMSProp preconditioning which is the celebrated Adam algorithm [Kingma 2015].

Incorporating momentum consists in replacing the gradient by an exponential moving average of the past iterates of the gradients with a parameter  $\beta_1 \in [0, 1[$ . In our setting, it amounts replacing line 10 of Algorithm 5 by lines 4 and 5 of Algorithm 6.

---

**Algorithm 6** Adding momentum to RED
 

---

```

1: Initialization  $\hat{g}_0 = 0$ .
2: for  $k = 1..$  do
3:   ...
4:    $\hat{g}_k = \beta_1 \hat{g}_{k-1} + (1 - \beta_1) g_k$    and    $\tilde{g}_k = \hat{g}_k / (1 - \beta_1^k)$ 
5:    $\dot{\Theta}_k = P_k^{-1} \tilde{g}_k$ 
6:   ...
7: end for

```

---

Momentum was introduced by Polyak [Polyak 1964] in the convex non-stochastic setting. It can be interpreted as an adaptation of a convex method to a non-convex stochastic problem. We coin this explanation as the *heavy-ball* analysis. Another point of view, which we denote as *variance reduction*, is that the exponential moving average  $\tilde{g}_k$  is a better estimator of  $\nabla \mathcal{J}(\Theta_k)$  than  $g_k$ . Indeed all the previous batches  $(\mathcal{B}_m)_{m \leq k}$  are taken into account in the computation of  $\tilde{g}_k$ . The downside is that the averaged quantity is  $\nabla \mathcal{J}_{\mathcal{B}_m}(\Theta_m)$  and not  $\nabla \mathcal{J}_{\mathcal{B}_m}(\Theta_k)$ , this introduces a bias in the estimation of  $\nabla \mathcal{J}(\Theta_k)$ . With this interpretation in mind, the parameter  $\beta_1$  which drives the capacity of the exponential moving average to forget the previous iterations has to be tuned between the mini-batches gradient variance (high variance leads to high  $\beta_1$ ) and the convergence (high values of  $\|\Theta_k - \Theta_{k-1}\|$  lead to low choice of  $\beta_1$ ). In [Kingma 2015], the authors propose to solve this dilemma by taking decaying values of  $\beta_1$ , although in practice, the parameter  $\beta_1$  is constant.

**Momentum: heavy ball or variance reduction?** When momentum is understood as an heavy-ball method, at iteration  $k$  there are no reasons for  $-\dot{\Theta}_k$  to be a direction of descent. Because our algorithm relies on the assumption that  $-\dot{\Theta}_k$  is a direction of descent to choose a step, our analysis falls apart and RED should be used with care. On the other hand, if momentum is a variance reduction technique, the step has to be taken small enough in order not to bias the gradient estimation. With this latter assumption, RED can be applied.

In order to determine if, in our case, momentum acts as an heavy ball method or as a variance reduction technique, we study numerically when the standard Adam



and SGD with momentum optimizers yield a direction of descent. In Figure 5.6 first row, the test of CIFAR100 in Section 5.4.1.2 is performed with a momentum  $\beta_1 = 0.9$  and the hyperparameters were tuned using the same policy (see Section 5.7). We display in the last column of Figure 5.6 first row, the percentage of direction of descent per epoch with respect to the current batch  $\mathcal{B}_k$ . If  $n$  is the epoch number and  $\mathcal{K}_n$  the set of the iterations that are in epoch  $n$ , this percentage is given by:

$$q_n = \frac{1}{|\mathcal{K}_n|} \sum_{k \in \mathcal{K}_n} \mathbf{1}_{\langle g_k, \dot{\Theta}_k \rangle \geq 0} \quad (5.26)$$

We observe that on classification problems, SGD with momentum and more particularly Adam yield directions of update that are not direction of descent for  $\mathcal{J}_{\mathcal{B}_k}$ .

**Step choice** The RED algorithm needs a rule to deal with update directions which are not directions of descent. One possibility is to allow negative steps, which we discard since this would annihilate the heavy-ball effect. Another possibility, which we retain, is to take the absolute value of  $\tau_k^*$  in line 12. In a nutshell, compute the step for the opposite direction (which is a direction of descent) and use this step in the current direction. This choice is arbitrary and to properly tackle the momentum case, interpretations using Lyapunov functions should be considered. The choice of such functions is not clear and we defer such an analysis to future work.

In Figure 5.6 first row the results of the optimization using RED on CIFAR100 with momentum are displayed. The parameters for the initial learning rate and its decay factor are the default ones  $\ell = 1$  and  $\eta = 1/2$ . The RED algorithm has difficulties to converge both on the training and testing losses. We observed that the steps chosen by RED are several orders of magnitude higher than the ones obtained by manual tuning. On classification problems, RED follows directions of update that are not direction of descent.

**Learning rate multiplication** The impediment to using RED with momentum is that directions of update are not directions of descent. This can be solved by reducing the initial learning rate  $\ell$  to take smaller steps  $\tau_k$  so that  $\|\Theta_k - \Theta_{k-1}\|$  remains small.

We propose to diminish the initial learning rate by using  $\ell = 1 - \beta_1$ . This choice may seem arbitrary but it is inspired by the proofs of convergence of [Défossez 2022] that have bounds which scale as  $1 - \beta_1$ . The experiments of Figure 5.6 last row are performed with the same set of parameters except for the initial learning rate which is set to  $\ell = 0.1$ . With this smaller learning rate, the algorithm is stable and converges. Of importance, RED always yield direction of descent as seen from the bottom-right of Figure 5.6. As  $\ell$  was decreased, the exploration is lost, explaining these poor convergence results.

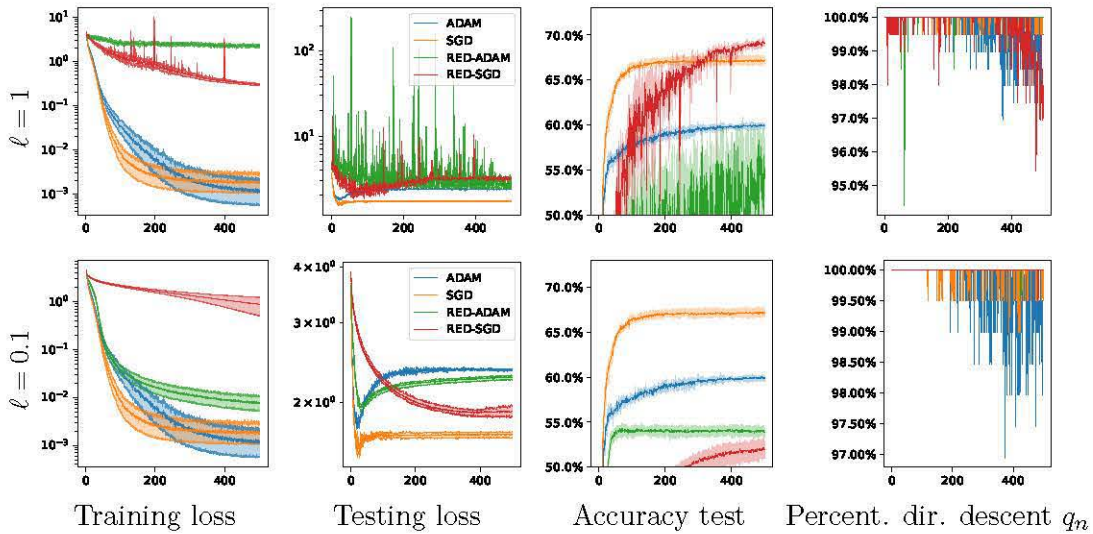


Figure 5.6: Tests with momentum ( $\beta_1 = 0.9$ ) with and without the learning rate stabilization ( $\ell = 1$  or  $\ell = 0.1$ ) for RED on CIFAR100. Manually-tuned algorithms (orange for SGD, blue for Adam) and their RED version (red for SGD, green for Adam) are given. Lower learning rate in RED ensures that momentum yields direction of descent at the expense of losing the exploration of the set of parameters.

**Conclusion** When using momentum, decreasing the learning rate makes the experiments fit in the framework the algorithm was proposed for. As a consequence, this causes the loss of the exploration which is critical to speed-up the convergence. The correct way of dealing with momentum would be to identify the Lyapunov function that has to be minimized, which is left for future work.

### 5.8.2 Batch reduction on CIFAR

In this section, we study batch dependence on the RED method for the CIFAR datasets. Reducing the batch size mimicks harder stochastic problems while keeping the experiment in a controlled environment. In Figure 5.7 (column 1 and 3), we provide the results obtained for different batch size and the evolution of the training and testing loss functions per epoch. The RED parameters are an initial learning rate  $\ell = 1$  and a target learning rate  $\ell = \frac{1}{2}$  after 100 epochs. A striking phenomenon in Figure 5.7 (column 1 and 3) is the loss of performance of the algorithm when the batch size is smaller than the number of classes.

Because of the relationship between the batch size and the number of classes, we wish to study if the last layer – the Linear Classifier (LC) – is the layer the most impacted by the batch size reduction. The LC optimizes the parameters of hyperplanes (one per class) which separate the information given by the remaining of the network, the Feature Extractor (FE). When the batch size is too small, some classes are not represented in the batch. The corresponding hyperplanes receive update information which is oblivious to the data of their class. We believe that

this effect explains the loss of performance of the LC and a lack of precision in the computation of the curvature.

In order to verify our assumption, we implement a *memory* layer, which is set between the FE and the LC. This memory layer stores the last 256 data given by the FE. We coin this trick a memory-DNN. Because the LC is fed with this memory, it should behave as if the batch size was 256, although the memory suffers from a slight delay, due to the fact that it is not updated for the current parameters of the FE. The memory footprint and computational load of the memory-DNN is increased by a small factor, since the FE is fed with small batches and is responsible for most of the computational load and memory footprint. In Figure 5.7 (column 2 and 4), we collect the results of memory-DNN. Of striking importance is a better behavior of memory-DNN compared to the standard DNN when the batch size is smaller than the number of classes.

In this test, we provide a simple remedy to avoid stochasticity issues in the training of the LC in a classification problem. More important than the memory trick is the fact that rescaling the learning rate allows us to provide a unified environment to test the method. The learning rate does not have to be adapted for each experiment, which would eventually prevent us from drawing any conclusions.

### 5.8.3 Effect of the $L^2$ regularization

According to the paragraph on the  $L^2$  regularization in Section 5.2.2, a regularization is introduced in our algorithm to counteract the effect of a potentially vanishing Hessian in the direction of update. This is a theoretical limitation and we study in this section the influence of this regularization on the performance of RED. We conduct the experiments of Section 5.4.1.2 for the CIFAR10 dataset with different values of the regularization  $\lambda \in \{10^{-7}, 10^{-4}\}$ . The hyperparameters of the standard SGD and RMSProp optimizers are tuned for each value of  $\lambda$ . We report in Figure 5.8 the different results, including the ones that are shown in Section 5.4.1.2. The grid search on the training loss that led to the choice of parameters for RMSProp and  $\lambda = 10^{-4}$  yielded large step size at the cost of instabilities in the test metrics. On all test cases, we observe that a value of regularization close to zero ( $\lambda = 10^{-7}$ ) gives good convergence results. In the considered tests, the need of a regularization seems to be more of a theoretical limitation than a practical one.

### 5.8.4 Comparison with BB and Robbins-Monro

In this section we compare our algorithm with the closest existing approach [Castera 2022], named *step-tuned*, where the authors approximate the curvature with a BB method. We also compare it with a Robbins-Monro decay rule of the learning rate for SGD. We did not compare with the BB method of [Yang 2018] as this method requires the computation of the gradient over the whole dataset at each epoch.

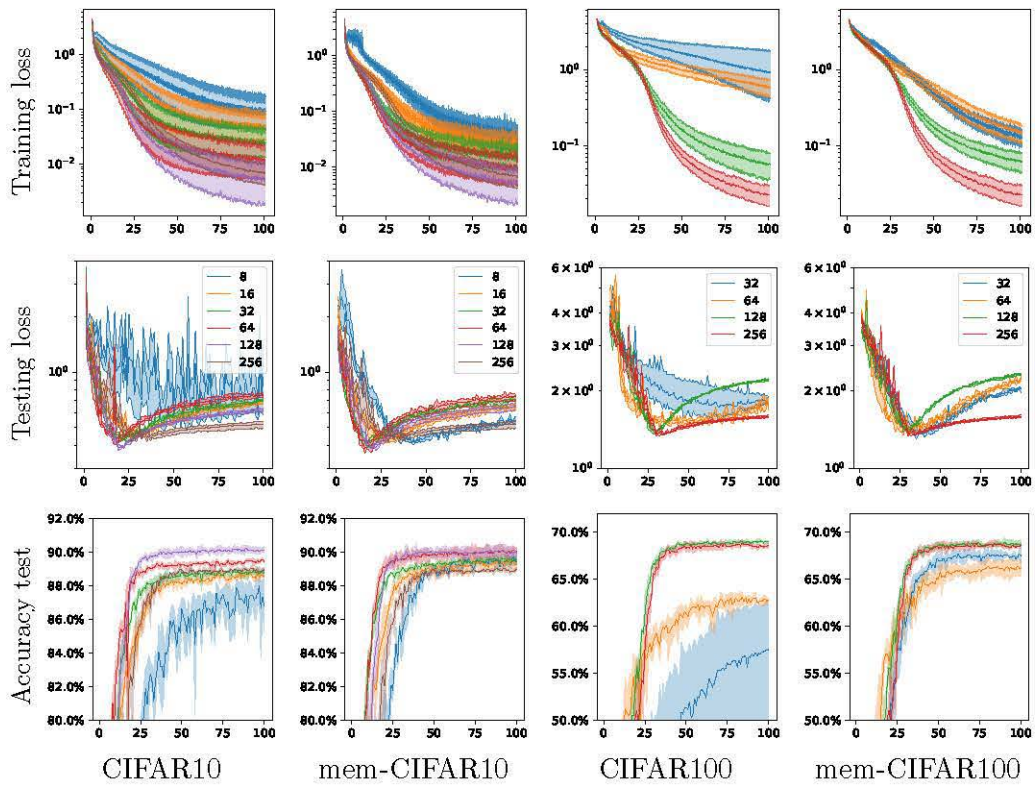


Figure 5.7: Batch reduction on the CIFAR10 and CIFAR100 datasets. The columns 1 and 3 are the vanilla RED-algorithm and the column 2 and 4 are the patches that (partially) solve the problem when the batch size is smaller than the number of classes.

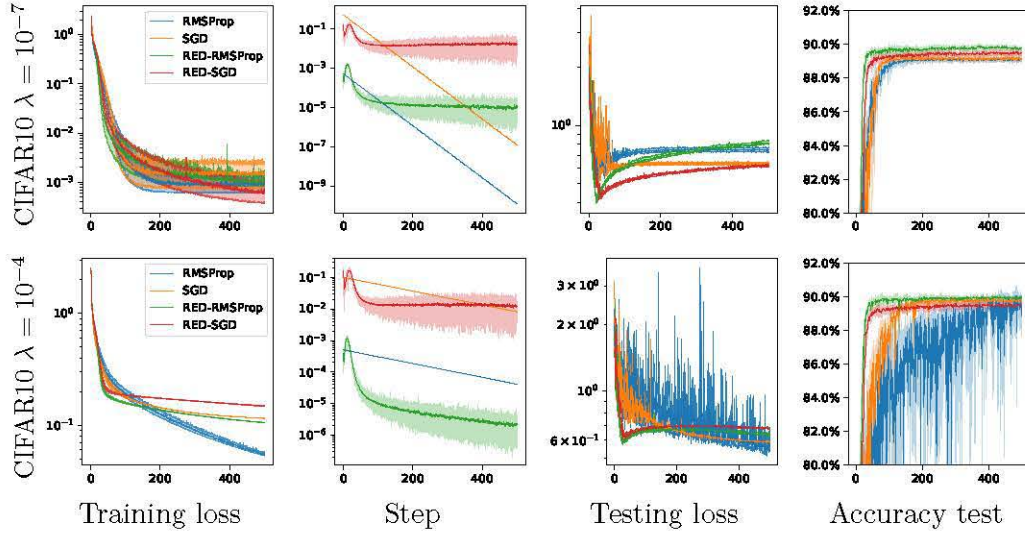


Figure 5.8: Influence of the  $L^2$  regularization  $\lambda$ . Manually-tuned algorithms (orange for SGD, blue for RMSProp) and their RED version (red for SGD, green for RMSProp) are given. For these tests, the smaller the regularization, the best the convergence of RED.

The step-tuned optimizer has several hyperparameters and we use the default ones except the learning rate as advised in [Castera 2022]. The initial learning rate of [Castera 2022] is searched on the same grid than the standard SGD (see Section 5.7).

In Figure 5.9, the results of the optimization on the CIFAR10 classifier and on the autoencoder are given for two values of the  $L^2$  regularization  $\lambda \in \{10^{-7}, 10^{-4}\}$ . RED algorithm is outperformed by step-tuned only on the training loss but the learning rate of step-tuned has been optimized for the training loss and we cannot expect better performance than step-tuned on this criterion. Finally, RED is more stable on every test metrics and has better generalization than step-tuned. Step-tuned [Castera 2022] requires the optimization of the learning rate and because RED does not need any hyperparameter adjustment, our method is competitive with this existing work. Note also that step-tuned is not available with the RMSProp preconditioner, when RED handles any kind of preconditioning technique.



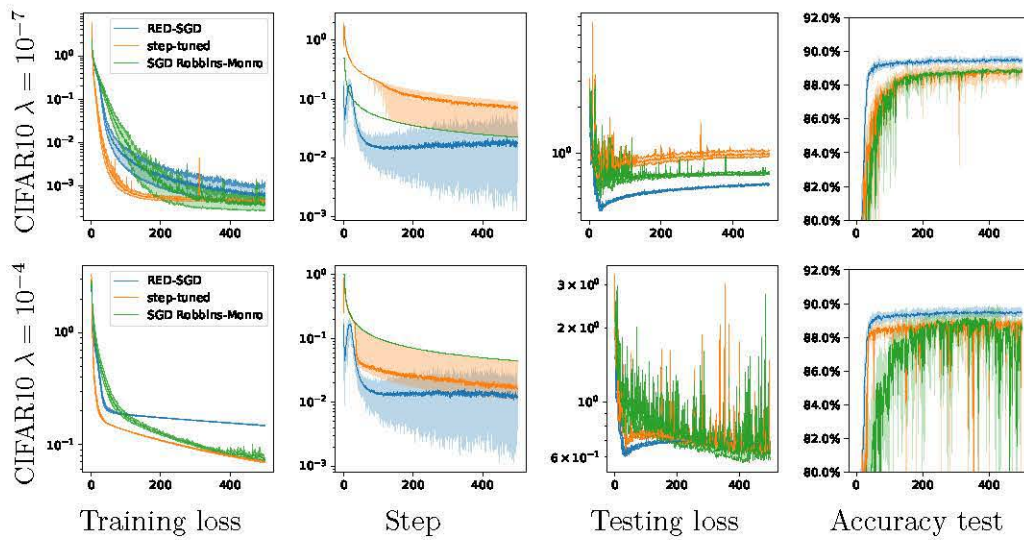


Figure 5.9: Comparison with step-tuned [Castera 2022] method on the CIFAR10 classifier. RED (blue), step-tuned (orange) and standard SGD (green) with Robbins-Monro conditions are given. RED is competitive with step-tuned on the accuracy but not on the training loss of the CIFAR10 classifier for which step-tuned is optimized.



# Conclusion

## Conclusion in english

The works presented in this manuscript address different issues of learning for computational MRI. They are articulated around adaptive image reconstruction networks, the optimization of Fourier sampling and the choice of hyperparameters for neural networks training.

Problems in Fourier sampling optimization that seem simple to resolve using deep learning tools turn out to be infeasible to solve using gradient based approaches. We resort to use a reparametrization in the density space to reduce the dimensionality of the problem. This allows using 0-th order optimization procedures to globalize the convergence. It both significantly reduces the computational cost of the training and let us use very limited datasets.

The training of neural networks for inverse problems in imaging is long and often application specific. This is a bottleneck for a large scale use in the industry. We have shown in this manuscript that training neural networks on a family of operators is not detrimental to the reconstruction in comparison to end-to-end training with a fixed operator. It appears that the unrolled networks are more robust to changes than approximate inversion networks. Having an unrolled network that is robust to changes in the forward model also allows solving different blind inverse problems.

The manual tuning of the learning rate when training deep neural networks is time consuming. We presented a new algorithm that allows computing second order information with a reduced computational cost in comparison to existing methods. This second order information is used in a rescaling of the learning rate that exhibits a natural interpretation between exploration and convergence.

In the works presented in this manuscript, although real dataset were used, the measurements were all simulated. This can lead to both an inverse crime [Colton 1998] and a data crime [Shimron 2022] for Chapters 3 and 4. A more complete benchmark of the proposed methods with experiments on real scanners should be conducted to move towards practical use.

Three years is both long and the end comes quickly. We ran out of time to explore several directions of research that deserve more studies. In particular, using the second order information of Chapter 5 to train neural networks for imaging should be investigated in the future.



## Conclusion en français

Les travaux présentés dans ce manuscrit abordent différentes problématiques d'apprentissage pour l'IRM computationnelle. Ils s'articulent autour des réseaux de neurones adaptatifs pour la reconstruction d'images, de l'optimisation de l'échantillonnage de Fourier et du choix des hyperparamètres pour l'entraînement des réseaux de neurones.

Les problèmes d'optimisation de schémas de Fourier qui semblent simples à résoudre à l'aide d'outils d'apprentissage profond s'avèrent être impossibles à résoudre à l'aide de méthodes de descente de gradient. Nous avons eu recours à une reparamétrisation dans l'espace de densité pour réduire la dimensionnalité du problème. Cela permet d'utiliser des procédures d'optimisation d'ordre 0 pour globaliser la convergence. Cela réduit aussi considérablement le coût de calcul de l'apprentissage et nous permet d'utiliser des jeux de données très limités.

L'apprentissage des réseaux de neurones pour les problèmes inverses en imagerie est long et souvent spécifique à l'application considérée. C'est un point critique pour une utilisation à grande échelle dans l'industrie de ces méthodes. Nous avons montré dans ce manuscrit que l'entraînement des réseaux neuronaux sur une famille d'opérateurs ne nuit pas à la reconstruction par rapport à l'entraînement pour un opérateur fixe. Il semble que les réseaux unrolled soient plus robustes aux changements dans la physique de l'acquisition que les réseaux reposant sur une inversion approximative de l'opérateur. Le fait de disposer d'un réseau unrolled robuste aux changements de modèle permet également de résoudre différents problèmes inverses aveugles.

Le choix manuel du pas dans l'entraînement de réseaux de neurones profonds prend beaucoup de temps. Nous avons présenté un nouvel algorithme qui permet de calculer l'information à l'ordre deux avec un coût de calcul réduit par rapport aux méthodes existantes. Cette information de second ordre est utilisée pour mettre à l'échelle le pas à travers un facteur multiplicatif qui présente une interprétation naturelle entre l'exploration et la convergence.

Dans les travaux présentés dans ce manuscrit, bien que des jeux de données réels aient été utilisés, les mesures ont toutes été simulées. Cela peut conduire à un *inverse crime* [Colton 1998] et à un *data crime* [Shimron 2022] pour les Chapitres 3 et 4. Une comparaison plus complète des méthodes proposées, avec des expériences sur des scanners réels, devrait être réalisée pour permettre de passer à une utilisation en pratique.

Trois ans sont à la fois longs et la fin approche vite. Nous avons manqué de temps pour explorer plusieurs directions de recherche qui méritent d'être davantage étudiées. En particulier, l'utilisation de l'information de second ordre du Chapitre 5 pour entraîner des réseaux de neurones pour les problèmes inverses en imagerie est une piste à explorer.

## Supports financiers

Cette thèse a été financée par le *Ministère de l'Éducation Nationale de l'Enseignement Supérieur, de la Recherche et de l'Innovation* français. Les coûts opérationnels ont été supportés par l'ANR *JCJC Optimization on Measures Spaces*, ANR-17-CE23-0013-01, et l'ANR-3IA Artificial and Natural Intelligence Toulouse Institute. Les travaux ont aussi bénéficié de l'accès à la plateforme de calcul HPC de GENCI-IDRIS (Accès 2021-AD011012210R1).

## Enseignements

En tant qu'étudiant en thèse, j'ai eu l'opportunité de donner des cours à l'INSA de Toulouse pendant les trois années écoulées. Les interventions dans les différents cours sont résumées ci-après:

- Projet recherche (équivalent M2). J'ai encadré 3 projets recherche de 5<sup>ème</sup> année au département de mathématiques.
- Projet modélisation (équivalent L3). J'ai encadré plusieurs projets de 3<sup>ème</sup> année pour les étudiants en mathématiques appliquées. Sujets : rendu graphique d'images, SVM, problèmes inverses, modélisation d'un réseau électrique renouvelable pour la France en 2050. **30 heures pendant 2 ans**
- Travaux dirigés signal (équivalent L2). **11 heures 1 année**
- Travaux pratiques optimisation (équivalent M1). **5 heures pendant 2 ans**
- Travaux pratiques analyse numérique et optimisation (équivalent L3). **45 heures pendants 3 ans**
- Introduction aux problèmes inverses et réseaux *unrolled* (équivalent M2). J'ai créé un cours/TP de 6h sur les réseaux *unrolled* dans le cadre du cours de 5<sup>ème</sup> année d'image. **5 heures 1 année.**



# Résumé en Français

---

Nous fournissons ici un résumé en langue française des travaux présentés dans ce manuscrit de thèse.

## **Résumé :**

Cette thèse traite d'aspects liés à l'apprentissage pour l'Imagerie par Résonance Magnétique computationnelle. Le premier chapitre est une introduction à l'imagerie computationnelle et illustre à travers le cas de l'IRM les évolutions ayant guidé ce domaine. Il contient aussi une introduction pédagogique aux problèmes inverses et les méthodes de reconstruction associées. Cette introduction retrace les premières méthodes de reconstruction linéaires, l'apparition de méthodes non linéaires et les méthodes récentes de reconstruction apprises à l'aide de réseaux de neurones. Le second chapitre traite des minimiseurs parasites dans l'optimisation de schémas d'échantillonnage de Fourier dont la motivation est l'optimisation de schémas d'échantillonnage pour l'IRM pour une méthode de reconstruction choisie et pour une base de données d'images spécifique. Ce chapitre montre que ce type de problème a un nombre combinatoire de minimiseurs qui peuvent disparaître avec le grand nombre d'images dans la base de données mais que les bases de données classiques d'IRM ne contiennent pas assez d'images pour espérer voir apparaître ce phénomène. Le troisième chapitre propose une méthode de globalisation de la convergence pour l'optimisation de schémas de Fourier. Cela permet de grandement réduire le coût numérique de l'optimisation tout en conservant un gain dans l'amélioration des images. Le quatrième chapitre traite de l'entraînement de réseaux de neurones "unrolled" adaptatifs à des changements dans la physique de l'acquisition. Ce formalisme permet de résoudre plusieurs problèmes inverses aveugles. Enfin, le cinquième chapitre traite des méthodes d'optimisation pour des réseaux de neurones de manière générale. Il propose une méthode permettant d'introduire une mise à l'échelle du pas pour l'optimisation de réseaux de neurones. Cela ouvre la voie à une automatisation du choix des hyperparamètres lors de l'entraînement.

## **Résumé pour le grand public :**

Les scanners IRM tout comme les autres appareils d'imagerie médicale permettent d'obtenir des images de l'intérieur d'un corps. Ils nécessitent de lourds calculs numériques et des modèles poussés, à la fois à cause du bruit présent dans les mesures et du caractère complexe de retrouver l'image. Depuis quelques années, pour retrouver l'image de l'intérieur d'un patient, des méthodes apprises font leur apparition. Le principe est d'apprendre à retrouver l'image à partir d'un grand volume de données. Elles ont permis à la fois de réduire les temps d'acquisition des

IRM, de grandement améliorer la qualité des images, et de construire des séquences d'acquisition adaptées à l'image scannée (genou ou cerveau par exemple). Cette thèse propose des méthodes permettant de diminuer les temps d'entraînement et de réduire la dépendance au grand volume de données. L'adaptabilité des méthodes apprises et l'automatisation du choix des paramètres permettant l'entraînement y sont aussi traités.

# Bibliography

- [Adcock 2017] Ben Adcock, Anders C Hansen, Clarice Poon and Bogdan Roman. *Breaking the coherence barrier: A new theory for compressed sensing*. In Forum of Mathematics, Sigma, volume 5. Cambridge University Press, 2017. (Cited in pages 24 and 85.)
- [Adcock 2020] Ben Adcock, Claire Boyer and Simone Brugiapaglia. *On oracle-type local recovery guarantees in compressed sensing*. Information and Inference: A Journal of the IMA, 2020. (Cited in page 85.)
- [Adcock 2021] Ben Adcock and Anders C Hansen. *Compressive imaging: Structure, sampling, learning*. Cambridge University Press, 2021. (Cited in page 108.)
- [Adler 2017] Jonas Adler and Ozan Öktem. *Solving ill-posed inverse problems using iterative deep neural networks*. Inverse Problems, vol. 33, no. 12, page 124007, 2017. (Cited in pages 23, 41, 45, and 124.)
- [Adler 2018] Jonas Adler and Ozan Öktem. *Learned primal-dual reconstruction*. IEEE transactions on medical imaging, vol. 37, no. 6, pages 1322–1332, 2018. (Cited in pages 41, 45, and 124.)
- [Aggarwal 2018] Hemant K Aggarwal, Merry P Mani and Mathews Jacob. *MoDL: Model-based deep learning architecture for inverse problems*. IEEE transactions on medical imaging, vol. 38, no. 2, pages 394–405, 2018. (Cited in page 124.)
- [Aggarwal 2020] Hemant Kumar Aggarwal and Mathews Jacob. *J-MoDL: Joint model-based deep learning for optimized sampling and reconstruction*. IEEE journal of selected topics in signal processing, vol. 14, no. 6, pages 1151–1162, 2020. (Cited in pages 26, 60, 86, and 87.)
- [Ahmed 2013] Ali Ahmed, Benjamin Recht and Justin Romberg. *Blind deconvolution using convex programming*. IEEE Transactions on Information Theory, vol. 60, no. 3, pages 1711–1732, 2013. (Cited in page 126.)
- [Ahn 1986] CB Ahn, JH Kim and ZH Cho. *High-speed spiral-scan echo planar NMR imaging-I*. IEEE Transactions on Medical Imaging, vol. 5, no. 1, pages 2–7, 1986. (Cited in page 85.)
- [Allen-Zhu 2018] Zeyuan Allen-Zhu. *Natasha 2: Faster non-convex optimization than *sgd**. Advances in neural information processing systems, vol. 31, 2018. (Cited in page 156.)
- [Alvarez 2004] F Alvarez and A Cabot. *Steepest descent with curvature dynamical system*. Journal of optimization theory and applications, vol. 120, no. 2, pages 247–273, 2004. (Cited in page 155.)
- [Amari 1998] Shun-Ichi Amari. *Natural gradient works efficiently in learning*. Neural computation, vol. 10, no. 2, pages 251–276, 1998. (Cited in page 150.)
- [Antun 2020] Vegard Antun, Francesco Renna, Clarice Poon, Ben Adcock and Anders C Hansen. *On instabilities of deep learning in image reconstruction and the potential costs of AI*. Proceedings of the National Academy of Sciences, vol. 117, no. 48, pages 30088–30095, 2020. (Cited in pages 41 and 124.)

- [Armato III 2011] Samuel G Armato III, Geoffrey McLennan, Luc Bidaut, Michael F McNitt-Gray, Charles R Meyer, Anthony P Reeves, Binsheng Zhao, Denise R Aberle, Claudia I Henschke, Eric A Hoffman *et al.* *The lung image database consortium (LIDC) and image database resource initiative (IDRI): a completed reference database of lung nodules on CT scans*. *Medical physics*, vol. 38, no. 2, pages 915–931, 2011. (Cited in page 133.)
- [Arridge 2019] Simon Arridge, Peter Maass, Ozan Öktem and Carola-Bibiane Schönlieb. *Solving inverse problems using data-driven models*. *Acta Numerica*, vol. 28, pages 1–174, 2019. (Cited in page 123.)
- [Asim 2020a] Muhammad Asim, Max Daniels, Oscar Leong, Ali Ahmed and Paul Hand. *Invertible generative models for inverse problems: mitigating representation error and dataset bias*. In *International Conference on Machine Learning*, pages 399–409. PMLR, 2020. (Cited in page 51.)
- [Asim 2020b] Muhammad Asim, Fahad Shamshad and Ali Ahmed. *Blind image deconvolution using deep generative priors*. *IEEE Transactions on Computational Imaging*, vol. 6, pages 1493–1506, 2020. (Cited in pages 51, 124, and 126.)
- [Aubel 2019] Céline Aubel and Helmut Bölcskei. *Vandermonde matrices with nodes in the unit disk and the large sieve*. *Applied and Computational Harmonic Analysis*, vol. 47, no. 1, pages 53–86, 2019. (Cited in pages 43 and 77.)
- [Bahadir 2019] Cagla Deniz Bahadir, Adrian V Dalca and Mert R Sabuncu. *Learning-based optimization of the under-sampling pattern in MRI*. In *International Conference on Information Processing in Medical Imaging*, pages 780–792. Springer, 2019. (Cited in page 60.)
- [Bahadir 2020] Cagla Bahadir, Alan Wang, Adrian Dalca and Mert R Sabuncu. *Deep-learning-based Optimization of the Under-sampling Pattern in MRI*. *IEEE Transactions on Computational Imaging*, 2020. (Cited in pages 26, 86, and 87.)
- [Bai 2018] Yuanchao Bai, Gene Cheung, Xianming Liu and Wen Gao. *Graph-based blind image deblurring from a single photograph*. *IEEE transactions on image processing*, vol. 28, no. 3, pages 1404–1418, 2018. (Cited in page 126.)
- [Balandat 2020] Maximilian Balandat, Brian Karrer, Daniel R. Jiang, Samuel Daulton, Benjamin Letham, Andrew Gordon Wilson and Eytan Bakshy. *BoTorch: A Framework for Efficient Monte-Carlo Bayesian Optimization*. In *Advances in Neural Information Processing Systems 33*, 2020. (Cited in page 96.)
- [Baldassarre 2016] Luca Baldassarre, Yen-Huan Li, Jonathan Scarlett, Baran Gözcü, Ilija Bogunovic and Volkan Cevher. *Learning-based compressive subsampling*. *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 4, pages 809–822, 2016. (Cited in pages 25, 26, and 86.)
- [Barzilai 1988] Jonathan Barzilai and Jonathan M Borwein. *Two-point step size gradient methods*. *IMA journal of numerical analysis*, vol. 8, no. 1, pages 141–148, 1988. (Cited in pages 40 and 151.)
- [Batenkov 2020] Dmitry Batenkov, Laurent Demanet, Gil Goldman and Yosef Yomdin. *Conditioning of partial nonuniform Fourier matrices with clustered nodes*. *SIAM Journal on Matrix Analysis and Applications*, vol. 41, no. 1, pages 199–220, 2020. (Cited in page 77.)

- [Beck 2009] Amir Beck and Marc Teboulle. *A fast iterative shrinkage-thresholding algorithm for linear inverse problems*. SIAM journal on imaging sciences, vol. 2, no. 1, pages 183–202, 2009. (Cited in pages 2, 7, and 46.)
- [Becker 1988] Sue Becker and Yann Le Cun. *Improving the Convergence of Back-Propagation Learning with Second Order Methods*. Technical Report CRG-TR-88-5, Department of Computer Science, University of Toronto, 1988. (Cited in page 150.)
- [Bernstein 2004] Matt A Bernstein, Kevin F King and Xiaohong Joe Zhou. Handbook of mri pulse sequences. Elsevier, 2004. (Cited in page 84.)
- [Bertsekas 2014] Dimitri P Bertsekas. Constrained optimization and lagrange multiplier methods. Academic press, 2014. (Cited in page 35.)
- [Biglari 2013] Fahimeh Biglari and Maghsud Solimanpur. *Scaling on the spectral gradient method*. Journal of Optimization Theory and Applications, vol. 158, no. 2, pages 626–635, 2013. (Cited in pages 40 and 151.)
- [Blaimer 2004] Martin Blaimer, Felix Breuer, Matthias Mueller, Robin M Heidemann, Mark A Griswold and Peter M Jakob. *SMASH, SENSE, PILS, GRAPPA: how to choose the optimal method*. Topics in Magnetic Resonance Imaging, vol. 15, no. 4, pages 223–236, 2004. (Cited in page 84.)
- [Bombieri 1984] Enrico Bombieri. *On the large sieve*. In Goldbach Conjecture, pages 227–252. World Scientific, 1984. (Cited in page 77.)
- [Bora 2017] Ashish Bora, Ajil Jalal, Eric Price and Alexandros G Dimakis. *Compressed sensing using generative models*. In International Conference on Machine Learning, pages 537–546. PMLR, 2017. (Cited in pages 51, 124, and 126.)
- [Bostan 2020] Emrah Bostan, Reinhard Heckel, Michael Chen, Michael Kellman and Laura Waller. *Deep phase decoder: self-calibrating phase microscopy with an untrained deep neural network*. Optica, vol. 7, no. 6, pages 559–562, Jun 2020. (Cited in page 126.)
- [Bottou 2010] Léon Bottou. *Large-scale machine learning with stochastic gradient descent*. In Proceedings of COMPSTAT’2010, pages 177–186. Springer, 2010. (Cited in page 73.)
- [Boyd 2011] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein et al. *Distributed optimization and statistical learning via the alternating direction method of multipliers*. Foundations and Trends® in Machine learning, vol. 3, no. 1, pages 1–122, 2011. (Cited in pages 2, 7, 34, and 36.)
- [Boyer 2016] Claire Boyer, Nicolas Chauffert, Philippe Ciuciu, Jonas Kahn and Pierre Weiss. *On the generation of sampling schemes for magnetic resonance imaging*. SIAM Journal on Imaging Sciences, vol. 9, no. 4, pages 2039–2072, 2016. (Cited in pages 25, 86, 92, 115, and 130.)
- [Boyer 2019] Claire Boyer, Jérémie Bigot and Pierre Weiss. *Compressed sensing with structured sparsity and structured acquisition*. Applied and Computational Harmonic Analysis, vol. 46, no. 2, pages 312–350, 2019. (Cited in pages 24 and 85.)
- [Buades 2011] Antoni Buades, Bartomeu Coll and Jean-Michel Morel. *Non-local means denoising*. Image Processing On Line, vol. 1, pages 208–212, 2011. (Cited in page 100.)



- [Bubba 2019] Tatiana A Bubba, Gitta Kutyniok, Matti Lassas, Maximilian März, Wojciech Samek, Samuli Siltanen and Vignesh Srinivasan. *Learning the invisible: a hybrid deep learning-shearlet framework for limited angle computed tomography*. Inverse Problems, vol. 35, no. 6, page 064002, 2019. (Cited in page 138.)
- [Buzzard 2018] Gregory T Buzzard, Stanley H Chan, Suhas Sreehari and Charles A Bouman. *Plug-and-play unplugged: Optimization-free reconstruction using consensus equilibrium*. SIAM Journal on Imaging Sciences, vol. 11, no. 3, pages 2001–2020, 2018. (Cited in page 42.)
- [Campisi 2017] Patrizio Campisi and Karen Egiazarian. *Blind image deconvolution: theory and applications*. CRC press, 2017. (Cited in page 125.)
- [Candès 2006] Emmanuel J Candès, Justin Romberg and Terence Tao. *Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information*. IEEE Transactions on information theory, vol. 52, no. 2, pages 489–509, 2006. (Cited in pages 24, 85, and 123.)
- [Carmon 2017] Yair Carmon, John C Duchi, Oliver Hinder and Aaron Sidford. “*Convex Until Proven Guilty*”: *Dimension-Free Acceleration of Gradient Descent on Non-Convex Functions*. In International Conference on Machine Learning, pages 654–663. PMLR, 2017. (Cited in page 156.)
- [Castera 2022] Camille Castera, Jérôme Bolte, Cédric Févotte and Edouard Pauwels. *Second-order step-size tuning of SGD for non-convex optimization*. Neural Processing Letters, pages 1–26, 2022. (Cited in pages 40, 151, 155, 157, 163, 166, 178, 180, and 181.)
- [Chaithya 2021] GR Chaithya, Zaccharie Ramzi and Philippe Ciuciu. *Learning the sampling density in 2D SPARKLING MRI acquisition for optimized image reconstruction*. In 2021 29th European Signal Processing Conference (EUSIPCO), pages 960–964. IEEE, 2021. (Cited in pages 100 and 114.)
- [Chaithya 2022] GR Chaithya, Pierre Weiss, Guillaume Daval-Frérrot, Aurélien Massire, Alexandre Vignaud and Philippe Ciuciu. *Optimizing full 3D SPARKLING trajectories for high-resolution Magnetic Resonance Imaging*. IEEE Transactions on Medical Imaging, 2022. (Cited in pages 100, 113, and 114.)
- [Chakrabarti 2016] Ayan Chakrabarti. *A neural approach to blind motion deblurring*. In European conference on computer vision, pages 221–235. Springer, 2016. (Cited in page 125.)
- [Chambolle 2021a] Antonin Chambolle and Thomas Pock. *Approximating the total variation with finite differences or finite elements*. In Handbook of Numerical Analysis, volume 22, pages 383–417. Elsevier, 2021. (Cited in page 41.)
- [Chambolle 2021b] Antonin Chambolle and Thomas Pock. *Learning consistent discretizations of the total variation*. SIAM Journal on Imaging Sciences, vol. 14, no. 2, pages 778–813, 2021. (Cited in page 41.)
- [Chan 1998] Tony F Chan and Chiu-Kwong Wong. *Total variation blind deconvolution*. IEEE transactions on Image Processing, vol. 7, no. 3, pages 370–375, 1998. (Cited in page 126.)
- [Chan 2016] Stanley H Chan, Xiran Wang and Omar A Elgendy. *Plug-and-play ADMM for image restoration: Fixed-point convergence and applications*. IEEE Transactions on Computational Imaging, vol. 3, no. 1, pages 84–98, 2016. (Cited in pages 42 and 47.)

- [Charlier 2021a] Benjamin Charlier, Jean Feydy, Joan Alexis Glaunes, François-David Collin and Ghislain Durif. *Kernel Operations on the GPU, with Autodiff, without Memory Overflows*. J. Mach. Learn. Res., vol. 22, no. 74, pages 1–6, 2021. (Cited in page 113.)
- [Charlier 2021b] Benjamin Charlier, Jean Feydy, Joan Alexis Glaunès, François-David Collin and Ghislain Durif. *Kernel Operations on the GPU, with Autodiff, without Memory Overflows*. Journal of Machine Learning Research, vol. 22, no. 74, pages 1–6, 2021. (Cited in page 22.)
- [Chauffert 2016] Nicolas Chauffert, Pierre Weiss, Jonas Kahn and Philippe Ciuciu. *A projection algorithm for gradient waveforms design in Magnetic Resonance Imaging*. IEEE transactions on medical imaging, vol. 35, no. 9, pages 2026–2039, 2016. (Cited in pages 90, 95, and 114.)
- [Chauffert 2017] Nicolas Chauffert, Philippe Ciuciu, Jonas Kahn and Pierre Weiss. *A projection method on measures sets*. Constructive Approximation, vol. 45, no. 1, pages 83–111, 2017. (Cited in pages 25, 86, 95, 101, 113, and 130.)
- [Chen 2013] Zhiqiang Chen, Xin Jin, Liang Li and Ge Wang. *A limited-angle CT reconstruction method based on anisotropic TV minimization*. Physics in Medicine & Biology, vol. 58, no. 7, page 2119, 2013. (Cited in page 41.)
- [Chen 2019] Liang Chen, Faming Fang, Tingting Wang and Guixu Zhang. *Blind image deblurring with local maximum gradient prior*. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 1742–1750, 2019. (Cited in page 126.)
- [Christianson 1992] Bruce Christianson. *Automatic Hessians by reverse accumulation*. IMA Journal of Numerical Analysis, vol. 12, no. 2, pages 135–150, 1992. (Cited in pages 40, 150, 151, 160, and 170.)
- [Colton 1998] David L Colton, Rainer Kress and Rainer Kress. Inverse acoustic and electromagnetic scattering theory, volume 93. Springer, 1998. (Cited in pages 48, 139, 183, and 184.)
- [Combettes 2011] Patrick L Combettes and Jean-Christophe Pesquet. *Proximal splitting methods in signal processing*. In Fixed-point algorithms for inverse problems in science and engineering, pages 185–212. Springer, 2011. (Cited in pages 36 and 124.)
- [Condat 2017] Laurent Condat. *Discrete total variation: New definition and minimization*. SIAM Journal on Imaging Sciences, vol. 10, no. 3, pages 1258–1290, 2017. (Cited in page 41.)
- [Cooley 1965] James W Cooley and John W Tukey. *An algorithm for the machine calculation of complex Fourier series*. Mathematics of computation, vol. 19, no. 90, pages 297–301, 1965. (Cited in page 21.)
- [Curtis 2019] Frank E Curtis and Daniel P Robinson. *Exploiting negative curvature in deterministic and stochastic optimization*. Mathematical Programming, vol. 176, no. 1, pages 69–94, 2019. (Cited in page 156.)
- [Dabov 2007] Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik and Karen Egiazarian. *Image denoising by sparse 3-D transform-domain collaborative filtering*. IEEE Transactions on image processing, vol. 16, no. 8, pages 2080–2095, 2007. (Cited in page 42.)

- [Dai 2002] Yuhong Dai, Jinyun Yuan and Ya-Xiang Yuan. *Modified two-point stepsize gradient methods for unconstrained optimization*. Computational Optimization and Applications, vol. 22, no. 1, pages 103–109, 2002. (Cited in pages 40 and 151.)
- [de Gournay 2022] Frédéric de Gournay and Alban Gossard. *Adaptive scaling of the learning rate by second order automatic differentiation*. 2022. (Cited in page 147.)
- [Debarnot 2022] Valentin Debarnot and Pierre Weiss. *Deep-Blur: Blind Identification and Deblurring with Convolutional Neural Networks*. 2022. (Cited in pages 98, 125, and 130.)
- [Défossez 2022] Alexandre Défossez, Léon Bottou, Francis Bach and Nicolas Usunier. *A simple convergence proof of adam and adagrad*. Transactions on Machine Learning Research, 2022. (Cited in page 176.)
- [Diamond 2017] Steven Diamond, Vincent Sitzmann, Felix Heide and Gordon Wetzstein. *Unrolled optimization with deep priors*. arXiv preprint arXiv:1705.08041, 2017. (Cited in pages 23, 41, 45, and 124.)
- [Dietrich 2016] Benjamin E Dietrich, David O Brunner, Bertram J Wilm, Christoph Barmet, Simon Gross, Lars Kasper, Maximilian Haeberlin, Thomas Schmid, S Johanna Vannesjo and Klaas P Pruessmann. *A field camera for MR sequence monitoring and system analysis*. Magnetic resonance in medicine, vol. 75, no. 4, pages 1831–1840, 2016. (Cited in page 127.)
- [Dong 2018] Weisheng Dong, Peiyao Wang, Wotao Yin, Guangming Shi, Fangfang Wu and Xiaotong Lu. *Denoising prior driven deep neural network for image restoration*. IEEE transactions on pattern analysis and machine intelligence, vol. 41, no. 10, pages 2305–2318, 2018. (Cited in pages 41, 45, and 124.)
- [Duchi 2011] John Duchi, Elad Hazan and Yoram Singer. *Adaptive subgradient methods for online learning and stochastic optimization*. Journal of machine learning research, vol. 12, no. 7, 2011. (Cited in pages 40 and 151.)
- [Duchon 1977] Jean Duchon. *Splines minimizing rotation-invariant semi-norms in Sobolev spaces*. In Constructive theory of functions of several variables, pages 85–100. Springer, 1977. (Cited in page 127.)
- [Eckstein 1992] Jonathan Eckstein and Dimitri P Bertsekas. *On the Douglas—Rachford splitting method and the proximal point algorithm for maximal monotone operators*. Mathematical Programming, vol. 55, no. 1, pages 293–318, 1992. (Cited in pages 2, 7, and 34.)
- [Facchinei 2003] Francisco Facchinei and Jong-Shi Pang. *Finite-dimensional variational inequalities and complementarity problems*. Springer, 2003. (Cited in page 35.)
- [Feichtinger 1994] Hans G Feichtinger and Karlheinz Gröchenig. *Theory and practice of irregular sampling*. Wavelets: mathematics and applications, vol. 1994, pages 305–363, 1994. (Cited in pages 24 and 85.)
- [Fergus 2006] Rob Fergus, Barun Singh, Aaron Hertzmann, Sam T Roweis and William T Freeman. *Removing camera shake from a single photograph*. In Acm Siggraph 2006 Papers, pages 787–794. 2006. (Cited in page 126.)
- [Fessler 2003] Jeffrey A Fessler and Bradley P Sutton. *Nonuniform fast Fourier transforms using min-max interpolation*. IEEE transactions on signal processing, vol. 51, no. 2, pages 560–574, 2003. (Cited in pages 22 and 112.)

- [Fessler 2010] Jeffrey A Fessler. *Model-based image reconstruction for MRI*. IEEE signal processing magazine, vol. 27, no. 4, pages 81–89, 2010. (Cited in pages 19, 23, and 57.)
- [Fortin 2000] Michel Fortin and Roland Glowinski. *Augmented lagrangian methods: applications to the numerical solution of boundary-value problems*. Elsevier, 2000. (Cited in pages 2 and 7.)
- [Frazier 2006] Michael W Frazier. *An introduction to wavelets through linear algebra*. Springer Science & Business Media, 2006. (Cited in page 34.)
- [Frazier 2018a] Peter I Frazier. *Bayesian optimization*. In *Recent advances in optimization and modeling of contemporary problems*, pages 255–278. Informs, 2018. (Cited in page 132.)
- [Frazier 2018b] Peter I Frazier. *A tutorial on Bayesian optimization*. arXiv preprint arXiv:1807.02811, 2018. (Cited in pages 96 and 99.)
- [Genzel 2022a] Martin Genzel, Ingo Gühring, Jan Macdonald and Maximilian März. *Near-Exact Recovery for Tomographic Inverse Problems via Deep Learning*. In *International Conference on Machine Learning*, pages 7368–7381. PMLR, 2022. (Cited in page 111.)
- [Genzel 2022b] Martin Genzel, Jan Macdonald and Maximilian März. *Solving inverse problems with deep neural networks-robustness included*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2022. (Cited in pages 39 and 124.)
- [Gidel 2019] Gauthier Gidel, Hugo Berard, Gaëtan Vignoud, Pascal Vincent and Simon Lacoste-Julien. *A variational inequality perspective on generative adversarial networks*. In *International Conference on Learning Representations*, 2019. (Cited in page 112.)
- [Gilton 2019] Davis Gilton, Greg Ongie and Rebecca Willett. *Neumann networks for linear inverse problems in imaging*. IEEE Transactions on Computational Imaging, vol. 6, pages 328–343, 2019. (Cited in page 41.)
- [Gilton 2021a] Davis Gilton, Gregory Ongie and Rebecca Willett. *Deep equilibrium architectures for inverse problems in imaging*. IEEE Transactions on Computational Imaging, vol. 7, pages 1123–1133, 2021. (Cited in page 49.)
- [Gilton 2021b] Davis Gilton, Gregory Ongie and Rebecca Willett. *Model adaptation for inverse problems in imaging*. IEEE Transactions on Computational Imaging, vol. 7, pages 661–674, 2021. (Cited in pages 124 and 135.)
- [Goldfarb 1970] Donald Goldfarb. *A family of variable-metric methods derived by variational means*. Mathematics of computation, vol. 24, no. 109, pages 23–26, 1970. (Cited in page 74.)
- [Goldstein 2012] Amit Goldstein and Raanan Fattal. *Blur-kernel estimation from spectral irregularities*. In *European Conference on Computer Vision*, pages 622–635. Springer, 2012. (Cited in page 125.)
- [Goodfellow 2014] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville and Yoshua Bengio. *Generative adversarial nets*. *Advances in neural information processing systems*, vol. 27, 2014. (Cited in page 51.)

- [Goodman 1996] J.W. Goodman. Introduction to fourier optics. Electrical Engineering Series. McGraw-Hill, 1996. (Cited in page 128.)
- [Gossard 2012] Alban Gossard, Frédéric de Gournay and Pierre Weiss. *Off-the-grid data-driven optimization of sampling schemes in MRI*. In international Traveling Workshop on Interactions between low-complexity data models and Sensing Techniques (iTWIST) 2020, 2012. (Cited in pages 60 and 63.)
- [Gossard 2022a] Alban Gossard, Frédéric de Gournay and Pierre Weiss. *Bayesian Optimization of Sampling Densities in MRI*. arXiv preprint arXiv:2209.07170, 2022. (Cited in pages 83, 130, 134, and 139.)
- [Gossard 2022b] Alban Gossard, Frédéric de Gournay and Pierre Weiss. *Spurious minimizers in non uniform Fourier sampling optimization*. Inverse Problems, 2022. (Cited in pages 59, 87, 90, 91, 100, 104, 112, and 139.)
- [Gossard 2022c] Alban Gossard and Pierre Weiss. *Training Adaptive Reconstruction Networks for Blind Inverse Problems*. arXiv preprint arXiv:2202.11342, 2022. (Cited in pages 111 and 119.)
- [Gower 2018] Robert Gower, Nicolas Le Roux and Francis Bach. *Tracking the gradients using the hessian: A new look at variance reducing stochastic methods*. In International Conference on Artificial Intelligence and Statistics, pages 707–715. PMLR, 2018. (Cited in page 150.)
- [Gözcü 2018] Baran Gözcü, Rabeeh Karimi Mahabadi, Yen-Huan Li, Efe Ilıcak, Tolga Cukur, Jonathan Scarlett and Volkan Cevher. *Learning-based compressive MRI*. IEEE transactions on medical imaging, vol. 37, no. 6, pages 1394–1406, 2018. (Cited in pages 25, 26, 60, and 86.)
- [Gräf 2012] Manuel Gräf, Daniel Potts and Gabriele Steidl. *Quadrature errors, discrepancies, and their relations to halftoning on the torus and the sphere*. SIAM Journal on Scientific Computing, vol. 34, no. 5, pages A2760–A2791, 2012. (Cited in pages 25 and 95.)
- [Greengard 2004] Leslie Greengard and June-Yub Lee. *Accelerating the nonuniform fast Fourier transform*. SIAM review, vol. 46, no. 3, pages 443–454, 2004. (Cited in page 22.)
- [Gregor 2010] Karol Gregor and Yann LeCun. *Learning fast approximations of sparse coding*. In Proceedings of the 27th international conference on machine learning, pages 399–406, 2010. (Cited in page 41.)
- [Griewank 2008] Andreas Griewank and Andrea Walther. Evaluating derivatives: principles and techniques of algorithmic differentiation. SIAM, 2008. (Cited in pages 40 and 151.)
- [Griswold 2002] Mark A Griswold, Peter M Jakob, Robin M Heidemann, Mathias Nitka, Vladimir Jellus, Jianmin Wang, Berthold Kiefer and Axel Haase. *Generalized autocalibrating partially parallel acquisitions (GRAPPA)*. Magnetic Resonance in Medicine: An Official Journal of the International Society for Magnetic Resonance in Medicine, vol. 47, no. 6, pages 1202–1210, 2002. (Cited in pages 125 and 131.)
- [Gu 2014] Shuhang Gu, Lei Zhang, Wangmeng Zuo and Xiangchu Feng. *Weighted nuclear norm minimization with application to image denoising*. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 2862–2869, 2014. (Cited in pages 42 and 125.)

- [Gwosdek 2014] Pascal Gwosdek, Christian Schmaltz, Joachim Weickert and Tanja Teuber. *Fast electrostatic half-toning*. Journal of real-time image processing, vol. 9, no. 2, pages 379–392, 2014. (Cited in page 25.)
- [Hammernik 2018] Kerstin Hammernik, Teresa Klatzer, Erich Kobler, Michael P Recht, Daniel K Sodickson, Thomas Pock and Florian Knoll. *Learning a variational network for reconstruction of accelerated MRI data*. Magnetic resonance in medicine, vol. 79, no. 6, pages 3055–3071, 2018. (Cited in page 86.)
- [Hammernik 2019] Kerstin Hammernik, Jo Schlemper, Chen Qin, Jinming Duan, Ronald M Summers and Daniel Rueckert.  *$\Sigma$ -net: Systematic Evaluation of Iterative Deep Neural Networks for Fast Parallel MR Image Reconstruction*. arXiv preprint arXiv:1912.09278, 2019. (Cited in pages 41, 45, and 124.)
- [Hinton 2006] Geoffrey E Hinton and Ruslan R Salakhutdinov. *Reducing the dimensionality of data with neural networks*. science, vol. 313, no. 5786, pages 504–507, 2006. (Cited in page 163.)
- [Hoffer 2019] Elad Hoffer, Tal Ben-Nun, Itay Hubara, Niv Giladi, Torsten Hoeffler and Daniel Soudry. *Augment your batch: better training with larger batches*. arXiv preprint arXiv:1901.09335, 2019. (Cited in page 168.)
- [Hurault 2022a] Samuel Hurault, Arthur Leclaire and Nicolas Papadakis. *Gradient step denoiser for convergent plug-and-play*. In International Conference on Learning Representations, 2022. (Cited in page 43.)
- [Hurault 2022b] Samuel Hurault, Arthur Leclaire and Nicolas Papadakis. *Proximal denoiser for convergent plug-and-play optimization with nonconvex regularization*. arXiv preprint arXiv:2201.13256, 2022. (Cited in page 43.)
- [Idy-Peretti 2009] I Idy-Peretti. *Évolution de l’imagerie par résonance magnétique*. IRBM, vol. 30, no. 2, pages 53–59, 2009. (Cited in pages 16 and 53.)
- [Jackson 1992] John I Jackson, Dwight G Nishimura and Albert Macovski. *Twisting radial lines with application to robust magnetic resonance imaging of irregular flow*. Magnetic Resonance in Medicine, vol. 25, no. 1, pages 128–139, 1992. (Cited in page 85.)
- [Jacob 2020] Mathews Jacob, Jong Chul Ye, Leslie Ying and Mariya Doneva. *Computational MRI: Compressive Sensing and Beyond [From the Guest Editors]*. IEEE Signal Processing Magazine, vol. 37, no. 1, pages 21–23, 2020. (Cited in page 85.)
- [Ji 2008] Shihao Ji, Ya Xue and Lawrence Carin. *Bayesian compressive sensing*. IEEE Transactions on signal processing, vol. 56, no. 6, pages 2346–2356, 2008. (Cited in page 38.)
- [Jin 2017] Kyong Hwan Jin, Michael T McCann, Emmanuel Froustey and Michael Unser. *Deep convolutional neural network for inverse problems in imaging*. IEEE Transactions on Image Processing, vol. 26, no. 9, pages 4509–4522, 2017. (Cited in pages 43 and 124.)
- [Jin 2019] Kyong Hwan Jin, Michael Unser and Kwang Moo Yi. *Self-supervised deep active accelerated MRI*. arXiv preprint arXiv:1901.04547, 2019. (Cited in pages 25, 26, 60, 86, and 87.)

- [Johnson 2013] Rie Johnson and Tong Zhang. *Accelerating stochastic gradient descent using predictive variance reduction*. Advances in neural information processing systems, vol. 26, 2013. (Cited in page 151.)
- [Johnson 2019] Jeff Johnson, Matthijs Douze and Hervé Jégou. *Billion-scale similarity search with gpus*. IEEE Transactions on Big Data, vol. 7, no. 3, pages 535–547, 2019. (Cited in page 98.)
- [Juditsky 2011] Anatoli Juditsky, Arkadi Nemirovski and Claire Tauvel. *Solving variational inequalities with stochastic mirror-prox algorithm*. Stochastic Systems, vol. 1, no. 1, pages 17–58, 2011. (Cited in page 112.)
- [Kamilov 2022] Ulugbek S Kamilov, Charles A Bouman, Gregory T Buzzard and Brendt Wohlberg. *Plug-and-play methods for integrating physical and learned models in computational imaging*. arXiv preprint arXiv:2203.17061, 2022. (Cited in page 43.)
- [Keiner 2009] Jens Keiner, Stefan Kunis and Daniel Potts. *Using NFFT 3—a software library for various nonequispaced fast Fourier transforms*. ACM Transactions on Mathematical Software (TOMS), vol. 36, no. 4, pages 1–30, 2009. (Cited in pages 22 and 112.)
- [Kingma 2015] Diederik P Kingma and Jimmy Ba. *Adam: A method for stochastic optimization*. In Proceedings of the International Conference on Learning Representations (ICLR), 2015. (Cited in pages 7, 40, 132, 151, 156, and 175.)
- [Knoll 2011] Florian Knoll, Christian Clason, Clemens Diwoky and Rudolf Stollberger. *Adapted random sampling patterns for accelerated MRI*. Magnetic resonance materials in physics, biology and medicine, vol. 24, no. 1, pages 43–50, 2011. (Cited in page 86.)
- [Knoll 2020] Florian Knoll, Tullie Murrell, Anuroop Sriram, Nafissa Yakubova, Jure Zbontar, Michael Rabbat, Aaron Defazio, Matthew J Muckley, Daniel K Sodickson, C Lawrence Zitnick et al. *Advancing machine learning for MR image reconstruction with an open competition: Overview of the 2019 fastMRI challenge*. Magnetic Resonance in Medicine, 2020. (Cited in page 85.)
- [Kochenderfer 2019] Mykel J Kochenderfer and Tim A Wheeler. Algorithms for optimization. Mit Press, 2019. (Cited in page 40.)
- [Kotelnikov 1933] Vladimir A Kotelnikov. *On the carrying capacity of the "either" and wire in telecommunications*. In Material for the First All-Union Conference on Questions of Communication (Russian), Izd. Red. Upr. Svyzai RKKA, Moscow, 1933, 1933. (Cited in page 24.)
- [Krishnan 2009] Dilip Krishnan and Rob Fergus. *Fast image deconvolution using hyper-Laplacian priors*. Advances in neural information processing systems, vol. 22, 2009. (Cited in page 126.)
- [Krishnan 2011] Dilip Krishnan, Terence Tay and Rob Fergus. *Blind deconvolution using a normalized sparsity measure*. In CVPR 2011, pages 233–240. IEEE, 2011. (Cited in page 126.)
- [Krishnan 2018] Shankar Krishnan, Ying Xiao and Rif A Saurous. *Neumann optimizer: A practical optimization algorithm for deep neural networks*. In Proceedings of the International Conference on Learning Representations (ICLR), 2018. (Cited in page 150.)

- [Krizhevsky 2009] Alex Krizhevsky, Geoffrey Hinton *et al.* *Learning multiple layers of features from tiny images*. Technical Report, Pennsylvania State University, 2009. (Cited in page 163.)
- [Kundur 1996] Deepa Kundur and Dimitrios Hatzinakos. *Blind image deconvolution*. IEEE signal processing magazine, vol. 13, no. 3, pages 43–64, 1996. (Cited in page 125.)
- [Kunisch 2013] Karl Kunisch and Thomas Pock. *A bilevel optimization approach for parameter learning in variational models*. SIAM Journal on Imaging Sciences, vol. 6, no. 2, pages 938–983, 2013. (Cited in pages 41 and 49.)
- [Lam 2015] Siu Kwan Lam, Antoine Pitrou and Stanley Seibert. *Numba: A llvm-based python jit compiler*. In Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC, pages 1–6, 2015. (Cited in page 22.)
- [Laumont 2022] Rémi Laumont, Valentin De Bortoli, Andrés Almansa, Julie Delon, Alain Durmus and Marcelo Pereyra. *On Maximum-a-Posteriori estimation with Plug & Play priors and stochastic gradient descent*. Journal of Mathematical Imaging and Vision, 2022. (Cited in page 43.)
- [Lauterbur 1973] Paul C Lauterbur. *Image formation by induced local interactions: examples employing nuclear magnetic resonance*. nature, vol. 242, no. 5394, pages 190–191, 1973. (Cited in pages 1, 6, 14, 23, and 52.)
- [Lazarus 2019] Carole Lazarus, Pierre Weiss, Nicolas Chauffert, Franck Mauconduit, Loubna El Gueddari, Christophe Destrieux, Ilyess Zemmoura, Alexandre Vignaud and Philippe Ciuciu. *SPARKLING: variable-density k-space filling curves for accelerated T2\*-weighted MRI*. Magnetic resonance in medicine, vol. 81, no. 6, pages 3643–3661, 2019. (Cited in pages 21, 25, 86, 95, 100, 101, 110, 114, and 130.)
- [Lazarus 2020a] Carole Lazarus, Maximilian März and Pierre Weiss. *Correcting the side effects of ADC filtering in MR image reconstruction*. Journal of Mathematical Imaging and Vision, vol. 62, no. 6, pages 1034–1047, 2020. (Cited in pages 20, 57, and 100.)
- [Lazarus 2020b] Carole Lazarus, Pierre Weiss, Loubna El Gueddari, Franck Mauconduit, Aurélien Massire, Mathilde Ripart, Alexandre Vignaud and Philippe Ciuciu. *3D variable-density SPARKLING trajectories for high-resolution T2\*-weighted magnetic resonance imaging*. NMR in Biomedicine, vol. 33, no. 9, page e4349, 2020. (Cited in page 25.)
- [Lebrat 2019] Léo Lebrat, Frédéric de Gournay, Jonas Kahn and Pierre Weiss. *Optimal transport approximation of 2-dimensional measures*. SIAM Journal on Imaging Sciences, vol. 12, no. 2, pages 762–787, 2019. (Cited in pages 25 and 95.)
- [Lecouat 2021] Bruno Lecouat, Jean Ponce and Julien Mairal. *Lucas-Kanade Reloaded: End-to-End Super-Resolution from Raw Image Bursts*. In 2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10–17, 2021, pages 2350–2359. IEEE, 2021. (Cited in page 126.)
- [Lecouat 2022] Bruno Lecouat, Thomas Eboli, Jean Ponce and Julien Mairal. *High Dynamic Range and Super-Resolution from Raw Image Bursts*. ACM Trans. Graph., vol. 41, no. 4, jul 2022. (Cited in page 126.)
- [LeCun 2010] Yann LeCun, Corinna Cortes and CJ Burges. *Mnist handwritten digit database*. AT&T Labs, 2010. (Cited in page 163.)



- [Li 2019a] Ting Li and Zhong Wan. *New adaptive barzilai–borwein step size and its application in solving large-scale optimization problems*. The ANZIAM Journal, vol. 61, no. 1, pages 76–98, 2019. (Cited in pages 40 and 151.)
- [Li 2019b] Yuelong Li, Mohammad Tofghi, Vishal Monga and Yonina C Eldar. *An algorithm unrolling approach to deep image deblurring*. In ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 7675–7679. IEEE, 2019. (Cited in pages 45 and 124.)
- [Liang 2019] Jinxiu Liang, Yong Xu, Chenglong Bao, Yuhui Quan and Hui Ji. *Barzilai–Borwein-based adaptive learning rate for deep learning*. Pattern Recognition Letters, vol. 128, pages 197–203, 2019. (Cited in pages 40 and 151.)
- [Lin 2014] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár and C Lawrence Zitnick. *Microsoft coco: Common objects in context*. In European conference on computer vision, pages 740–755. Springer, 2014. (Cited in page 133.)
- [Lin 2018] Jyh-Miin Lin. *Python non-uniform fast Fourier transform (PyNUFFT): An accelerated non-Cartesian MRI package on a heterogeneous platform (CPU/GPU)*. Journal of Imaging, vol. 4, no. 3, page 51, 2018. (Cited in page 22.)
- [Liu 1989] Dong C Liu and Jorge Nocedal. *On the limited memory BFGS method for large scale optimization*. Mathematical programming, vol. 45, no. 1, pages 503–528, 1989. (Cited in page 132.)
- [Liu 2017] Mingrui Liu and Tianbao Yang. *On noisy negative curvature descent: Competing with gradient descent for faster non-convex optimization*. arXiv preprint arXiv:1709.08571, 2017. (Cited in page 156.)
- [Liu 2021] Jiaming Liu, Salman Asif, Brendt Wohlberg and Ulugbek Kamilov. *Recovery analysis for plug-and-play priors using the restricted eigenvalue condition*. Advances in Neural Information Processing Systems, vol. 34, pages 5921–5933, 2021. (Cited in page 42.)
- [Ljubenović 2019] Marina Ljubenović and Mário A. T. Figueiredo. *Plug-and-play approach to class-adapted blind image deblurring*. International Journal on Document Analysis and Recognition (IJDAR), vol. 22, no. 2, pages 79–97, March 2019. (Cited in page 126.)
- [Loktyushin 2021] Alexander Loktyushin, Kai Herz, Nam Dang, Felix Glang, Anagha Deshmane, Simon Weinmüller, Arnd Doerfler, Bernhard Schölkopf, Klaus Scheffler and Moritz Zaiss. *MRzero-Automated discovery of MRI sequences using supervised learning*. Magnetic Resonance in Medicine, vol. 86, no. 2, pages 709–724, 2021. (Cited in page 60.)
- [Loshchilov 2017] Ilya Loshchilov and Frank Hutter. *Sgdr: Stochastic gradient descent with warm restarts*. In International Conference on Learning Representations, 2017. (Cited in page 164.)
- [Lustig 2005] Michael Lustig, Jin Hyung Lee, David L Donoho and John M Pauly. *Faster imaging with randomly perturbed, under-sampled spirals and  $\ell_1$  reconstruction*. In Proceedings of the 13th annual meeting of ISMRM, page 685, Miami Beach, FL, USA, 2005. (Cited in pages 24, 85, and 123.)

- [Lustig 2008] Michael Lustig, David L Donoho, Juan M Santos and John M Pauly. *Compressed sensing MRI*. IEEE signal processing magazine, vol. 25, no. 2, pages 72–82, 2008. (Cited in pages 85 and 89.)
- [Ma 2018] Ke Ma, Jinshan Zeng, Jiechao Xiong, Qianqian Xu, Xiaochun Cao, Wei Liu and Yuan Yao. *Stochastic non-convex ordinal embedding with stabilized barzilai-borwein step size*. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 32, 2018. (Cited in pages 40 and 151.)
- [Mansfield 1977] Peter Mansfield. *Multi-planar image formation using NMR spin echoes*. Journal of Physics C: Solid State Physics, vol. 10, no. 3, page L55, 1977. (Cited in pages 14 and 52.)
- [Martens 2010] James Martens *et al.* *Deep learning via hessian-free optimization*. In International conference on machine learning (ICML), volume 27, pages 735–742, 2010. (Cited in page 150.)
- [Martens 2015] James Martens and Roger Grosse. *Optimizing neural networks with kronecker-factored approximate curvature*. In International conference on machine learning, pages 2408–2417. PMLR, 2015. (Cited in page 150.)
- [Marvasti 2012] Farokh Marvasti. *Nonuniform sampling: theory and practice*. Springer Science & Business Media, 2012. (Cited in page 24.)
- [Michaeli 2014] Tomer Michaeli and Michal Irani. *Blind deblurring using internal patch recurrence*. In European conference on computer vision, pages 783–798. Springer, 2014. (Cited in page 126.)
- [Miyato 2018] Takeru Miyato, Toshiki Kataoka, Masanori Koyama and Yuichi Yoshida. *Spectral normalization for generative adversarial networks*. arXiv preprint arXiv:1802.05957, 2018. (Cited in page 42.)
- [Moitra 2015] Ankur Moitra. *Super-resolution, extremal functions and the condition number of Vandermonde matrices*. In Proceedings of the forty-seventh annual ACM symposium on Theory of computing, pages 821–830, 2015. (Cited in page 77.)
- [Muckley 2020a] Matthew J Muckley, Bruno Riemenschneider, Alireza Radmanesh, Sunwoo Kim, Geunu Jeong, Jingyu Ko, Yohan Jun, Hyungseob Shin, Dosik Hwang, Mahmoud Mostapha *et al.* *State-of-the-art Machine Learning MRI reconstruction in 2020: Results of the second fastMRI challenge*. arXiv preprint arXiv:2012.06318, 2020. (Cited in page 89.)
- [Muckley 2020b] Matthew J Muckley, Ruben Stern, Tullie Murrell and Florian Knoll. *TorchKbNufft: a high-level, hardware-agnostic non-uniform fast Fourier transform*. In ISMRM Workshop on Data Sampling & Image Reconstruction, 2020. (Cited in pages 22 and 112.)
- [Muckley 2021] Matthew J Muckley, Bruno Riemenschneider, Alireza Radmanesh, Sunwoo Kim, Geunu Jeong, Jingyu Ko, Yohan Jun, Hyungseob Shin, Dosik Hwang, Mahmoud Mostapha *et al.* *Results of the 2020 fastmri challenge for machine learning mr image reconstruction*. IEEE transactions on medical imaging, vol. 40, no. 9, pages 2306–2317, 2021. (Cited in pages 124 and 134.)
- [Nesterov 1983] Yurii E Nesterov. *A method for solving the convex programming problem with convergence rate  $O(1/k^2)$* . In Dokl. akad. nauk Sssr, volume 269, pages 543–547, 1983. (Cited in page 110.)

- [Neyshabur 2015] Behnam Neyshabur, Ryota Tomioka and Nathan Srebro. *In search of the real inductive bias: On the role of implicit regularization in deep learning*. International Conference on Learning Representations, 2015. (Cited in page 45.)
- [Ng 2010] Michael K Ng, Pierre Weiss and Xiaoming Yuan. *Solving constrained total-variation image restoration and reconstruction problems via alternating direction methods*. SIAM journal on Scientific Computing, vol. 32, no. 5, pages 2710–2736, 2010. (Cited in page 111.)
- [Ochs 2015] Peter Ochs, René Ranftl, Thomas Brox and Thomas Pock. *Bilevel optimization with nonsmooth lower level problems*. In International Conference on Scale Space and Variational Methods in Computer Vision, pages 654–665. Springer, 2015. (Cited in pages 41 and 89.)
- [Ochs 2016] Peter Ochs, René Ranftl, Thomas Brox and Thomas Pock. *Techniques for gradient-based bilevel optimization with non-smooth lower level problems*. Journal of Mathematical Imaging and Vision, vol. 56, no. 2, pages 175–194, 2016. (Cited in page 41.)
- [Okuta 2017] Ryosuke Okuta, Yuya Unno, Daisuke Nishino, Shohei Hido and Crissman Loomis. *CuPy: A NumPy-Compatible Library for NVIDIA GPU Calculations*. In Proceedings of Workshop on Machine Learning Systems (LearningSys) in The Thirty-first Annual Conference on Neural Information Processing Systems (NIPS), 2017. (Cited in page 22.)
- [Ollivier 2015] Yann Ollivier. *Riemannian metrics for neural networks I: feedforward networks*. Information and Inference: A Journal of the IMA, vol. 4, no. 2, pages 108–153, 2015. (Cited in page 150.)
- [Oppenheim 1971] Alan Oppenheim, Don Johnson and Kenneth Steiglitz. *Computation of spectra with unequal resolution using the fast Fourier transform*. Proceedings of the IEEE, vol. 59, no. 2, pages 299–301, 1971. (Cited in page 62.)
- [Pan 2014] Jinshan Pan, Zhe Hu, Zhixun Su and Ming-Hsuan Yang. *Deblurring text images via  $L_0$ -regularized intensity and gradient prior*. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 2901–2908, 2014. (Cited in page 126.)
- [Pan 2016] Jinshan Pan, Deqing Sun, Hanspeter Pfister and Ming-Hsuan Yang. *Blind image deblurring using dark channel prior*. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 1628–1636, 2016. (Cited in page 126.)
- [Parikh 2014] Neal Parikh, Stephen Boyd et al. *Proximal algorithms*. Foundations and trends® in Optimization, vol. 1, no. 3, pages 127–239, 2014. (Cited in pages 34 and 36.)
- [Pearlmutter 1994] Barak A Pearlmutter. *Fast exact multiplication by the Hessian*. Neural computation, vol. 6, no. 1, pages 147–160, 1994. (Cited in pages 40, 150, 151, 160, and 170.)
- [Peng 2022] Wei Peng, Li Feng, Guoying Zhao and Fang Liu. *Learning Optimal  $K$ -space Acquisition and Reconstruction using Physics-Informed Neural Networks*. CVPR, 2022. (Cited in page 60.)

- [Polyak 1964] Boris T Polyak. *Some methods of speeding up the convergence of iteration methods*. USSR computational mathematics and mathematical physics, vol. 4, no. 5, pages 1–17, 1964. (Cited in page 175.)
- [Polyak 2017] Boris Polyak and Pavel Shcherbakov. *Lyapunov functions: An optimization theory perspective*. IFAC-PapersOnLine, vol. 50, no. 1, pages 7456–7461, 2017. (Cited in page 170.)
- [Potts 2001] Daniel Potts, Gabriele Steidl and Manfred Tasche. *Fast Fourier transforms for nonequispaced data: A tutorial*. Modern sampling theory, pages 247–270, 2001. (Cited in page 127.)
- [Potts 2003] Daniel Potts and Gabriele Steidl. *Fast summation at nonequispaced knots by NFFT*. SIAM Journal on Scientific Computing, vol. 24, no. 6, pages 2013–2037, 2003. (Cited in page 113.)
- [Pronzato 2017] Luc Pronzato. *Minimax and maximin space-filling designs: some properties and methods for construction*. Journal de la Société Française de Statistique, vol. 158, no. 1, pages 7–36, 2017. (Cited in pages 98 and 130.)
- [Pruessmann 1999] Klaas P Pruessmann, Markus Weiger, Markus B Scheidegger and Peter Boesiger. *SENSE: sensitivity encoding for fast MRI*. Magnetic Resonance in Medicine: An Official Journal of the International Society for Magnetic Resonance in Medicine, vol. 42, no. 5, pages 952–962, 1999. (Cited in page 125.)
- [Ramachandran 1971] GN Ramachandran and AV Lakshminarayanan. *Three-dimensional reconstruction from radiographs and electron micrographs: application of convolutions instead of Fourier transforms*. Proceedings of the National Academy of Sciences, vol. 68, no. 9, pages 2236–2240, 1971. (Cited in pages 1 and 6.)
- [Raydan 1997] Marcos Raydan. *The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem*. SIAM Journal on Optimization, vol. 7, no. 1, pages 26–33, 1997. (Cited in pages 40 and 151.)
- [Reddi 2018] Sashank J Reddi, Satyen Kale and Sanjiv Kumar. *On the convergence of adam and beyond*. In Proceedings of the International Conference on Learning Representations (ICLR), 2018. (Cited in pages 151 and 156.)
- [Reehorst 2018] Edward T Reehorst and Philip Schniter. *Regularization by denoising: Clarifications and new interpretations*. IEEE transactions on computational imaging, vol. 5, no. 1, pages 52–67, 2018. (Cited in page 42.)
- [Ren 2016] Wenqi Ren, Xiaochun Cao, Jinshan Pan, Xiaojie Guo, Wangmeng Zuo and Ming-Hsuan Yang. *Image deblurring via enhanced low-rank prior*. IEEE Transactions on Image Processing, vol. 25, no. 7, pages 3426–3437, 2016. (Cited in page 126.)
- [Riis 2021] Nicolai André Brogaard Riis, Yiqiu Dong and Per Christian Hansen. *Computed tomography reconstruction with uncertain view angles by iteratively updated model discrepancy*. Journal of Mathematical Imaging and Vision, vol. 63, no. 2, pages 133–143, 2021. (Cited in page 126.)
- [Robbins 1951] Herbert Robbins and Sutton Monro. *A stochastic approximation method*. The annals of mathematical statistics, pages 400–407, 1951. (Cited in pages 40, 149, and 155.)

- [Roemer 1990] Peter B Roemer, William A Edelstein, Cecil E Hayes, Steven P Souza and Otward M Mueller. *The NMR phased array*. Magnetic resonance in medicine, vol. 16, no. 2, pages 192–225, 1990. (Cited in page 84.)
- [Romano 2017] Yaniv Romano, Michael Elad and Peyman Milanfar. *The little engine that could: Regularization by denoising (RED)*. SIAM Journal on Imaging Sciences, vol. 10, no. 4, pages 1804–1844, 2017. (Cited in page 42.)
- [Ronchetti 2020] Matteo Ronchetti. *Torchradon: Fast differentiable routines for computed tomography*. arXiv preprint arXiv:2009.14788, 2020. (Cited in page 133.)
- [Roux 2007] Nicolas Roux, Pierre-Antoine Manzagol and Yoshua Bengio. *Topmoumoute online natural gradient algorithm*. Advances in neural information processing systems, vol. 20, 2007. (Cited in page 150.)
- [Rudin 1992] Leonid I Rudin, Stanley Osher and Emad Fatemi. *Nonlinear total variation based noise removal algorithms*. Physica D: nonlinear phenomena, vol. 60, no. 1-4, pages 259–268, 1992. (Cited in pages 34, 41, and 123.)
- [Ryu 2019] Ernest Ryu, Jialin Liu, Sicheng Wang, Xiaohan Chen, Zhangyang Wang and Wotao Yin. *Plug-and-play methods provably converge with properly trained denoisers*. In International Conference on Machine Learning, pages 5546–5557. PMLR, 2019. (Cited in pages 42 and 125.)
- [Sanchez 2020] Thomas Sanchez, Baran Gözcü, Ruud B van Heeswijk, Armin Eftekhari, Efe Ilıcak, Tolga Çukur and Volkan Cevher. *Scalable learning-based sampling optimization for compressive dynamic MRI*. In ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 8584–8588. IEEE, 2020. (Cited in pages 25, 26, and 86.)
- [Sawchuk 1972] Alexander A Sawchuk. *Space-variant image motion degradation and restoration*. Proceedings of the IEEE, vol. 60, no. 7, pages 854–861, 1972. (Cited in page 29.)
- [Schaul 2013] Tom Schaul, Sixin Zhang and Yann LeCun. *No more pesky learning rates*. In International conference on machine learning (ICML), pages 343–351. PMLR, 2013. (Cited in page 150.)
- [Schmaltz 2010] Christian Schmaltz, Pascal Gwosdek, Andrés Bruhn and Joachim Weickert. *Electrostatic halftoning*. In Computer Graphics Forum, volume 29, pages 2313–2327. Wiley Online Library, 2010. (Cited in pages 25, 95, and 113.)
- [Schmitt 2012] Franz Schmitt, Michael K Stehling and Robert Turner. *Echo-planar imaging: theory, technique and application*. Springer Science & Business Media, 2012. (Cited in pages 24 and 85.)
- [Schuler 2015] Christian J Schuler, Michael Hirsch, Stefan Harmeling and Bernhard Schölkopf. *Learning to deblur*. IEEE transactions on pattern analysis and machine intelligence, vol. 38, no. 7, pages 1439–1451, 2015. (Cited in page 125.)
- [Shannon 1948] Claude Elwood Shannon. *A mathematical theory of communication*. The Bell system technical journal, vol. 27, no. 3, pages 379–423, 1948. (Cited in page 24.)
- [Shannon 1949] Claude E Shannon. *Communication in the presence of noise*. Proceedings of the IRE, vol. 37, no. 1, pages 10–21, 1949. (Cited in page 24.)

- [Sherry 2020] Ferdia Sherry, Martin Benning, Juan Carlos De los Reyes, Martin J Graves, Georg Maierhofer, Guy Williams, Carola-Bibiane Schönlieb and Matthias J Ehrhardt. *Learning the sampling pattern for MRI*. IEEE Transactions on Medical Imaging, vol. 39, no. 12, pages 4310–4321, 2020. (Cited in pages 25, 26, 60, 86, and 87.)
- [Shih 2021] Yu-hsuan Shih, Garrett Wright, Joakim Andén, Johannes Blaschke and Alex H Barnett. *cuFINUFFT: a load-balanced GPU library for general-purpose nonuniform FFTs*. In 2021 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), pages 688–697. IEEE, 2021. (Cited in pages 22, 112, and 133.)
- [Shimron 2022] Efrat Shimron, Jonathan I Tamir, Ke Wang and Michael Lustig. *Implicit data crimes: Machine learning bias arising from misuse of public data*. Proceedings of the National Academy of Sciences, vol. 119, no. 13, page e2117203119, 2022. (Cited in pages 139, 183, and 184.)
- [Simonyan 2015] Karen Simonyan and Andrew Zisserman. *Very deep convolutional networks for large-scale image recognition*. In Proceedings of the International Conference on Learning Representations (ICLR), 2015. (Cited in page 163.)
- [Smith 2018] Samuel L Smith, Pieter-Jan Kindermans, Chris Ying and Quoc V Le. *Don't decay the learning rate, increase the batch size*. In International Conference on Learning Representations, 2018. (Cited in pages 149 and 168.)
- [Sodickson 1997] Daniel K Sodickson and Warren J Manning. *Simultaneous acquisition of spatial harmonics (SMASH): fast imaging with radiofrequency coil arrays*. Magnetic resonance in medicine, vol. 38, no. 4, pages 591–603, 1997. (Cited in page 125.)
- [Soudry 2018] Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar and Nathan Srebro. *The implicit bias of gradient descent on separable data*. The Journal of Machine Learning Research, vol. 19, no. 1, pages 2822–2878, 2018. (Cited in page 45.)
- [Sun 2015] Jian Sun, Wenfei Cao, Zongben Xu and Jean Ponce. *Learning a convolutional neural network for non-uniform motion blur removal*. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 769–777, 2015. (Cited in page 125.)
- [Sun 2016] Jian Sun, Huibin Li, Zongben Xu et al. *Deep ADMM-Net for compressive sensing MRI*. Advances in neural information processing systems, vol. 29, 2016. (Cited in pages 41, 45, 124, 129, and 132.)
- [Szegedy 2016] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens and Zbigniew Wojna. *Rethinking the inception architecture for computer vision*. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 2818–2826, 2016. (Cited in page 168.)
- [Tan 2016] Conghui Tan, Shiqian Ma, Yu-Hong Dai and Yuqiu Qian. *Barzilai-borwein step size for stochastic gradient descent*. Advances in neural information processing systems, vol. 29, 2016. (Cited in pages 40 and 151.)
- [Teuber 2011] Tanja Teuber, Gabriele Steidl, Pascal Gwosdek, Christian Schmalz and Joachim Weickert. *Dithering by differences of convex functions*. SIAM Journal on Imaging Sciences, vol. 4, no. 1, pages 79–108, 2011. (Cited in page 25.)

- [Tieleman 2012] Tijmen Tieleman and G Hinton. *Divide the gradient by a running average of its recent magnitude*. *COURSERA Neural Networks for Machine Learning*. Mach. Learn, vol. 6, pages 26–31, 2012. (Cited in pages 7, 40, 132, and 151.)
- [Tyrtysnikov 1997] Evgeniï Evgen'evich Tyrtysnikov. A brief introduction to numerical analysis. Springer Science & Business Media, 1997. (Cited in pages 31 and 36.)
- [Ulyanov 2018] Dmitry Ulyanov, Andrea Vedaldi and Victor Lempitsky. *Deep image prior*. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 9446–9454, 2018. (Cited in page 50.)
- [Vannesjo 2016] S Johanna Vannesjo, Nadine N Graedel, Lars Kasper, Simon Gross, Julia Busch, Maximilian Haeberlin, Christoph Barmet and Klaas P Pruessmann. *Image reconstruction using a gradient impulse response model for trajectory prediction*. *Magnetic resonance in medicine*, vol. 76, no. 1, pages 45–58, 2016. (Cited in page 127.)
- [Vasanawala 2011] SS Vasanawala, MJ Murphy, Marcus T Alley, P Lai, Kurt Keutzer, John M Pauly and Michael Lustig. *Practical parallel imaging compressed sensing MRI: Summary of two years of experience in accelerating body MRI of pediatric patients*. In 2011 IEEE international symposium on biomedical imaging: From nano to macro, pages 1039–1043. IEEE, 2011. (Cited in page 86.)
- [Venkatakrishnan 2013] Singanallur V Venkatakrishnan, Charles A Bouman and Brendt Wohlberg. *Plug-and-play priors for model based reconstruction*. In 2013 IEEE Global Conference on Signal and Information Processing, pages 945–948. IEEE, 2013. (Cited in pages 42, 124, and 125.)
- [Vinyals 2012] Oriol Vinyals and Daniel Povey. *Krylov subspace descent for deep learning*. In Artificial intelligence and statistics, pages 1261–1268. PMLR, 2012. (Cited in page 150.)
- [Walther 2008] Andrea Walther. *Computing sparse Hessians with automatic differentiation*. *ACM Transactions on Mathematical Software (TOMS)*, vol. 34, no. 1, pages 1–15, 2008. (Cited in pages 40, 150, and 151.)
- [Wang 2018] Ge Wang, Jong Chu Ye, Klaus Mueller and Jeffrey A Fessler. *Image reconstruction is a new frontier of machine learning*. *IEEE transactions on medical imaging*, vol. 37, no. 6, pages 1289–1296, 2018. (Cited in page 123.)
- [Wang 2019] Ge Wang, Yi Zhang, Xiaojing Ye and Xuanqin Mou. *Machine learning for tomographic imaging*. IOP Publishing, 2019. (Cited in page 31.)
- [Wang 2021] Guanhua Wang and Jeffrey A Fessler. *Efficient approximation of Jacobian matrices involving a non-uniform fast Fourier transform (NUFFT)*. arXiv preprint arXiv:2111.02912, 2021. (Cited in page 112.)
- [Wang 2022a] Guanhua Wang, Tianrui Luo, Jon-Fredrik Nielsen, Douglas C Noll and Jeffrey A Fessler. *B-spline parameterized joint optimization of reconstruction and k-space trajectories (BJORK) for accelerated 2d MRI*. *IEEE Transactions on Medical Imaging*, 2022. (Cited in pages 26, 60, 63, 73, 86, 87, 90, 91, 102, 104, 106, 111, 112, 134, and 139.)
- [Wang 2022b] Renke Wang, Roxana Alexandru and Pier Luigi Dragotti. *Perfect Reconstruction of Classes of Non-Bandlimited Signals from Projections with Unknown Angles*. In ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 5877–5881. IEEE, 2022. (Cited in page 126.)

- [Wei 2020] Kaixuan Wei, Angelica Aviles-Rivero, Jingwei Liang, Ying Fu, Carola-Bibiane Schönlieb and Hua Huang. *Tuning-free plug-and-play proximal algorithm for inverse imaging problems*. In International Conference on Machine Learning, pages 10158–10169. PMLR, 2020. (Cited in page 43.)
- [Wei 2022] Kaixuan Wei, Angelica I Avilés-Rivero, Jingwei Liang, Ying Fu, Hua Huang and Carola-Bibiane Schönlieb. *TFFnP: Tuning-free Plug-and-Play Proximal Algorithms with Applications to Inverse Imaging Problems*. J. Mach. Learn. Res., vol. 23, no. 16, pages 1–48, 2022. (Cited in page 43.)
- [Weiss 2021] Tomer Weiss, Ortal Senouf, Sanketh Vedula, Oleg Michailovich, Michael Zibulevsky and Alex Bronstein. *PILOT: Physics-informed learned optimal trajectories for accelerated MRI*. Journal of Machine Learning for Biomedical Imaging (MELBA), pages 1–23, 2021. (Cited in pages 26, 60, 63, 73, 86, 87, 90, 91, 104, 106, 112, 134, and 139.)
- [Wilson 2017] Ashia C Wilson, Rebecca Roelofs, Mitchell Stern, Nati Srebro and Benjamin Recht. *The marginal value of adaptive gradient methods in machine learning*. Advances in neural information processing systems, vol. 30, 2017. (Cited in page 166.)
- [Xiang 2021] Jinxi Xiang, Yonggui Dong and Yunjie Yang. *FISTA-net: Learning a fast iterative shrinkage thresholding network for inverse problems in imaging*. IEEE Transactions on Medical Imaging, vol. 40, no. 5, pages 1329–1339, 2021. (Cited in page 46.)
- [Xiao 2010] Yunhai Xiao, Qiuyu Wang and Dong Wang. *Notes on the Dai–Yuan–Yuan modified spectral gradient method*. Journal of computational and applied mathematics, vol. 234, no. 10, pages 2986–2992, 2010. (Cited in pages 40 and 151.)
- [Xu 2013] Li Xu, Shicheng Zheng and Jiaya Jia. *Unnatural l0 sparse representation for natural image deblurring*. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 1107–1114, 2013. (Cited in page 126.)
- [Yan 2016] Ruomei Yan and Ling Shao. *Blind image blur estimation via deep learning*. IEEE Transactions on Image Processing, vol. 25, no. 4, pages 1910–1921, 2016. (Cited in page 125.)
- [Yang 2018] Zhuang Yang, Cheng Wang, Zhemin Zhang and Jonathan Li. *Random Barzilai–Borwein step size for mini-batch algorithms*. Engineering Applications of Artificial Intelligence, vol. 72, pages 124–135, 2018. (Cited in pages 40, 151, and 178.)
- [Yao 2021] Zhewei Yao, Amir Gholami, Sheng Shen, Mustafa Mustafa, Kurt Keutzer and Michael W Mahoney. *Adahessian: An adaptive second order optimizer for machine learning*. In Proceedings of the AAAI Conference on Artificial Intelligence, 2021. (Cited in page 150.)
- [Yoshida 2017] Yuichi Yoshida and Takeru Miyato. *Spectral norm regularization for improving the generalizability of deep learning*. arXiv preprint arXiv:1705.10941, 2017. (Cited in page 42.)
- [Yu 2019] Songhyun Yu, Bumjun Park and Jechang Jeong. *Deep iterative down-up cnn for image denoising*. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, pages 0–0, 2019. (Cited in page 87.)



- [Zbontar 2018] Jure Zbontar, Florian Knoll, Anuroop Sriram, Tullie Murrell, Zhengnan Huang, Matthew J Muckley, Aaron Defazio, Ruben Stern, Patricia Johnson, Mary Bruno *et al.* *fastMRI: An open dataset and benchmarks for accelerated MRI*. arXiv preprint arXiv:1811.08839, 2018. (Cited in pages 61, 72, 99, 130, and 133.)
- [Zeiler 2012] Matthew D Zeiler. *Adadelta: an adaptive learning rate method*. arXiv preprint arXiv:1212.5701, 2012. (Cited in pages 40 and 151.)
- [Zhang 2014] Yudong Zhang, Bradley S Peterson, Genlin Ji and Zhengchao Dong. *Energy preserved sampling for compressed sensing MRI*. Computational and mathematical methods in medicine, vol. 2014, 2014. (Cited in page 86.)
- [Zhang 2018] Jian Zhang and Bernard Ghanem. *ISTA-Net: Interpretable optimization-inspired deep network for image compressive sensing*. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 1828–1837, 2018. (Cited in pages 41, 45, and 124.)
- [Zhang 2021a] Daniel Zhang, Saurabh Mishra, Erik Brynjolfsson, John Etchemendy, Deep Ganguli, Barbara Grosz, Terah Lyons, James Manyika, Juan Carlos Niebles, Michael Sellitto *et al.* *The AI index 2021 annual report*. arXiv preprint arXiv:2103.06312, 2021. (Cited in pages 2 and 7.)
- [Zhang 2021b] Kai Zhang, Yawei Li, Wangmeng Zuo, Lei Zhang, Luc Van Gool and Radu Timofte. *Plug-and-play image restoration with deep denoiser prior*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2021. (Cited in pages 42, 47, 100, 111, 125, and 128.)
- [Zhang 2022] Meina Zhang, Yingying Fang, Guoxi Ni and Tiejong Zeng. *Pixel screening based intermediate correction for blind deblurring*. In 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 01–09, 2022. (Cited in page 126.)
- [Zhu 2018] Bo Zhu, Jeremiah Z Liu, Stephen F Cauley, Bruce R Rosen and Matthew S Rosen. *Image reconstruction by domain-transform manifold learning*. Nature, vol. 555, no. 7697, pages 487–492, 2018. (Cited in pages 2, 7, 23, 41, and 123.)
- [Zibetti 2021] Marcelo VW Zibetti, Gabor T Herman and Ravinder R Regatte. *Fast data-driven learning of parallel MRI sampling patterns for large scale problems*. Scientific Reports, vol. 11, no. 1, pages 1–19, 2021. (Cited in pages 25, 26, 60, 86, and 87.)

---

**Abstract:**

This thesis addresses different aspects of learning for computational Magnetic Resonance Imaging. The first chapter is an introduction to computational imaging and it illustrates through the case of MRI the developments that have guided this field. It also contains a pedagogical introduction to inverse problems and the associated reconstruction methods. This introduction traces the early linear reconstruction methods, the emergence of non-linear methods and recent advances in reconstruction methods that are learned with neural networks. The following chapters are based on different publications or preprints and, although links are made between the different chapters, they can be read independently of each other. The second chapter deals with spurious minimizers in the optimization of non-uniform Fourier sampling schemes. The motivation is the optimization of MRI sampling schemes for a chosen reconstruction method and for a specific image database. This chapter shows that this type of problem has a combinatorial number of minimizers that can disappear with the large number of images in the training database but that classical MRI databases do not contain enough images to expect this phenomenon to appear. The third chapter proposes a method to globalize the convergence for the optimization of Fourier sampling schemes. This drastically reduces the numerical cost of the optimization while maintaining a significant gain in the image quality. The fourth chapter deals with the training of neural networks that are adaptive to changes in the physics of the acquisition. This formalism allows to solve several blind inverse problems. Finally, the fifth chapter tackles the optimization of neural networks. It proposes a method to scale the learning rate and this opens the way to automate the choice of the hyperparameters during the training phase.

**Abstract for general audience:**

MRI scanners, like other medical imaging devices, allow imaging the interior of a body. They require intensive numerical computations and complex models, both because of the noise in the measurements and the complexity of retrieving the image. In recent years, learning methods have emerged for recovering the image of the interior of a patient. The principle is to learn to reconstruct the image from a large volume of data. They have allowed to reduce MRI acquisition times, to greatly improve image quality, and to build acquisition sequences adapted to the scanned image (knee or brain for example). This thesis proposes methods to reduce training times and to reduce the dependence on the large volume of data needed. The adaptability of the learning methods and the automation of the choice of training parameters are also addressed.

**Keywords:** optimization, learning, inverse problems, MRI, neural networks

---

**Résumé :**

Cette thèse traite d'aspects liés à l'apprentissage pour l'Imagerie par Résonance Magnétique computationnelle. Le premier chapitre est une introduction à l'imagerie computationnelle et illustre à travers le cas de l'IRM les évolutions ayant guidé ce domaine. Il contient aussi une introduction pédagogique aux problèmes inverses et les méthodes de reconstruction associées. Cette introduction retrace les premières méthodes de reconstruction linéaires, l'apparition de méthodes non linéaires et les méthodes récentes de reconstruction apprises à l'aide de réseaux de neurones. Le second chapitre traite des minimiseurs parasites dans l'optimisation de schémas d'échantillonnage de Fourier dont la motivation est l'optimisation de schémas d'échantillonnage pour l'IRM pour une méthode de reconstruction choisie et pour une base de données d'images spécifique. Ce chapitre montre que ce type de problème a un nombre combinatoire de minimiseurs qui peuvent disparaître avec le grand nombre d'images dans la base de données mais que les bases de données classiques d'IRM ne contiennent pas assez d'images pour espérer voir apparaître ce phénomène. Le troisième chapitre propose une méthode de globalisation de la convergence pour l'optimisation de schémas de Fourier. Cela permet de grandement réduire le coût numérique de l'optimisation tout en conservant un gain dans l'amélioration des images. Le quatrième chapitre traite de l'entraînement de réseaux de neurones "unrolled" adaptatifs à des changements dans la physique de l'acquisition. Ce formalisme permet de résoudre plusieurs problèmes inverses aveugles. Enfin, le cinquième chapitre traite des méthodes d'optimisation pour des réseaux de neurones de manière générale. Il propose une méthode permettant d'introduire une mise à l'échelle du pas pour l'optimisation de réseaux de neurones. Cela ouvre la voie à une automatisation du choix des hyperparamètres lors de l'entraînement.

**Mot clés :** optimisation, apprentissage, problèmes inverses, IRM, réseaux de neurones