A study of unrolled algorithms for dictionary learning and inverse problems, and contributions to M/EEG signal processing

Benoît Malézieux

université
PARIS-SACLAY

A study of unrolled algorithms for dictionary learning and inverse problems, and contributions to M/EEG signal processing

Etude des algorithmes déroulés pour l'apprentissage de dictionnaire et les problèmes inverses, et contributions au traitement des signaux M/EEG

Thèse de doctorat de l'université Paris-Saclay

École doctorale n°580 : sciences et technologies de l'information et de la communication (STIC)
Spécialité de doctorat: Informatique mathématique
Graduate School : Informatique et sciences du numérique
Référent : Faculté des sciences d'Orsay

Thèse préparée dans l'unité de recherche **Inria Saclay-Île-de-France** (Université Paris-Saclay, Inria), sous la direction de Matthieu KOWALSKI, Maître de Conférences, et le co-encadrement de Thomas MOREAU, Chargé de Recherche

Thèse soutenue à Paris-Saclay, le 20 Septembre 2023, par

Benoît MALEZIEUX

Composition du jury

Membres du jury avec voix délibérative

Rémi GRIBONVAL Directeur de Recherche, Inria, ENS Lyon	Président
Thomas OBERLIN Professeur, ISAE-SUPAERO, Université de Toulouse	Rapporteur et Examinateur
Emilie CHOUZENOUX Chargée de Recherche, Inria Saclay	Examinatrice
Bruno TORRESANI Professeur, Université Aix-Marseille	Examinateur

THESE DE DOCTORAT
NNT : 2023UPASG047

université PARIS-SACLAY

ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Titre: Etude des algorithmes déroulés pour l'apprentissage de dictionnaire et les problèmes inverses, et contributions au traitement des signaux M/EEG

Mots clés: algorithmes déroulés, apprentissage de dictionnaire, optimisation, parcimonie, problèmes inverses, M/EEG, traitement du signal

Résumé: La magnétoencéphalographie et l'électroencéphalographie (M/EEG) sont des techniques non invasives d'enregistrement de l'activité électrique du cerveau. Les données consistent en des séries temporelles multivariées qui fournissent des informations sur les processus cognitifs ainsi que sur l'état biologique d'un sujet. Dans le cadre de cette thèse, nous étudions trois problèmes qui se posent lors du traitement de signaux M/EEG: l'extraction de motifs, la résolution de problèmes inverses, et l'apprentissage statistique à partir de signaux M/EEG. Tout d'abord, nous proposons une analyse des algorithmes déroulés dans le cadre de l'apprentissage de dictionnaire, et en particulier de leur efficacité computationelle par rapport aux méthodes d'optimisation traditionnelles. Nous proposons également une étude de leur comportement pour la résolution des problèmes inverses. Dans un deuxième temps, nous mettons l'accent sur l'aspect applicatif. Nous revisitons l'apprentissage de dictionnaire convolutif pour l'extraction de motifs dans des signaux MEG de grande taille, et présentons une implémentation qui pourrait avoir des perspectives d'application en étude de populations. Enfin, nous introduisons une méthode d'apprentissage à partir des signaux M/EEG fondée sur l'analyse des matrices de covariance et le transport optimal.

Title: A study of unrolled algorithms for dictionary learning and inverse problems, and contributions to M/EEG signal processing

Keywords: unrolling, dictionary learning, optimization, sparsity, inverse problems, M/EEG, signal processing

Abstract: Magnetoencephalography and electroencephalography (M/EEG) are non-invasive techniques for recording the electrical activity of the brain. The data consist of multivariate time series that provide information about cognitive processes as well as the biological state of a subject. In this thesis, we study three problems that arise when processing M/EEG signals: pattern extraction, inverse problems resolution, and statistical learning from M/EEG signals. First, we propose an analysis of unrolled algorithms in the context of dictionary learning, and in particular of their computational efficiency compared to traditional optimization methods. We also propose a study of their behavior for solving inverse problems. In a second step, we focus on applications in M/EEG. We revisit convolutional dictionary learning for pattern extraction in large MEG signals, and present an implementation that could have potential applications in population studies. Finally, we introduce a method to learn from M/EEG signals based on covariance matrices analysis and optimal transport.

Résumé

La magnétoencéphalographie et l'électroencéphalographie (M/EEG) sont des modalités de neuroimagerie non invasives qui enregistrent l'activité électrique du cerveau. Concrètement, quelques dizaines ou centaines de capteurs sont placés autour de la tête d'un sujet, et la machine enregistre l'intensité du champ électrique ou magnétique à haute résolution temporelle, de l'ordre de 1000Hz. Cela permet de suivre l'évolution temporelle et spatiale des courants électriques circulant dans les dendrites des neurones lors de la transmission synaptique. Les données obtenues sont constituées d'autant de séries temporelles qu'il y a de capteurs et peuvent contenir un volume important d'échantillons, variant avec la durée de l'enregistrement. L'objectif de ce processus est de fournir des informations sur le processus cognitif ou l'état biologique d'un sujet. Il existe de nombreuses façons d'extraire des informations utiles de ces données, en fonction de la tâche à accomplir en aval. Chacune d'entre elles s'accompagne de défis spécifiques.

Premier défi: extraction efficace de motifs dans les signaux M/EEG. Une méthode d'automatisation de l'analyse des signaux M/EEG consiste à extraire des motifs cognitifs correspondant à des activités physiologiques, telles que des artefacts récurrents comme les battements de cœur ou les clignements d'yeux, ou des stimuli auditifs ou visuels externes présentés à un sujet. En fait, l'identification de formes d'ondes prototypiques et de leurs occurrences dans les données permet de concevoir une représentation événementielle de la dynamique temporelle du signal et de relier les stimuli aux réponses évoquées qu'ils déclenchent. Cette représentation permet de développer des outils statistiques pour déterminer le temps de réponse, ainsi que son intensité.

Une façon d'y parvenir est d'exploiter l'apprentissage de dictionnaire convolutif, qui est particulièrement efficace pour les tâches d'apprentissage de motifs sur des séries temporelles ou des images. Toutefois, les méthodes d'optimisation de l'apprentissage de dictionnaire convolutif ont du mal à s'adapter aux grandes quantités de données à traiter dans les applications telles que l'analyse des signaux MEG. En particulier, ces méthodes fonctionnent bien sur de petits enregistrements d'un sujet spécifique, mais sont difficilement applicables à des cohortes de dizaines ou de centaines d'individus. Ainsi, leur adaptation à l'analyse des enregistrements M/EEG pour l'étude de population présente un défi majeur: nous devons d'abord améliorer l'efficacité de calcul de l'apprentissage de dictionnaire convolutionnel.

Pour ce faire, nous devons trouver des algorithmes plus rapides ou faire des concessions acceptables qui préservent la qualité du résultat final.

Dans ce qui suit, nous étudions dans quelle mesure l'utilisation de la différentiation automatique et des algorithmes déroulés - un type spécial de réseau de neurones basé sur des algorithmes d'optimisation itératifs - permet d'accélérer l'apprentissage de dictionnaire pour les données volumineuses. Nous fournissons également des outils pour rendre l'apprentissage par dictionnaire convolutif plus rapide dans la pratique, ainsi que des perspectives d'applications dans l'analyse de données MEG.

Deuxième défi: apprentissage d'a priori pour la résolution de problèmes inverses. La mesure de l'activité électrique à la surface de la tête par magnétoencéphalographie s'accompagne d'une perte d'information potentiellement critique concernant la localisation des sources électriques à l'intérieur du cerveau. En effet, chaque capteur enregistre le signal émis par chaque source dont l'intensité dépend de la distance qui les sépare et des propriétés physiques des tissus cérébraux que l'onde électromagnétique doit traverser. En raison de la linéarité des équations de Maxwell, le signal observé est obtenu comme une transformation linéaire des sources, corrompue par le bruit de mesure. La combinaison des méthodes à éléments finis, des données du scan IRM de la tête du sujet, et de la connaissance des tissus cérébraux permet de calculer une approximation numérique de cette transformation linéaire. Étant donné que nous connaissons à la fois les observations et la transformation linéaire, nous sommes intéressés par la récupération des sources qui ont généré ces observations. C'est ce qu'on appelle le problème de la localisation des sources, un type particulier de problème inverse linéaire.

Comme la dimension des mesures (fournies par quelques centaines de capteurs) est généralement beaucoup plus petite que la dimension des sources (quelques milliers de voxels dans le cerveau), ce problème inverse est mal posé, ce qui signifie que plusieurs solutions pourraient être correctes compte tenu d'un ensemble d'observations. L'incertitude sur les mesures, corrompues par le bruit, augmente le nombre de solutions potentielles. Par conséquent, les praticiens s'appuient sur leur connaissance préalable des données pour sélectionner la solution la plus plausible parmi toutes les solutions possibles. Dans certains cas, des transformées efficaces et analytiques sont disponibles et produisent des résultats satisfaisants, comme les ondelettes pour les images ou les Gaborlets pour les signaux audio. Cependant, la complexité et la variabilité des signaux font qu'il est souvent difficile de s'appuyer sur des a priori ou des dictionnaires ad hoc, en particulier dans le scénario non supervisé de la M/EEG où aucune donnée de vérité de terrain n'est disponible. Un deuxième défi se pose donc : nous cherchons un moyen efficace et robuste de résumer la structure des données afin de résoudre un problème linéaire inverse sans avoir recours à des connaissances préalables.

Ici, nous nous concentrons sur l'utilisation possible des algorithmes déroulés comme méthode d'apprentissage d'a priori pour la résolution de problèmes inverses. Plus précisément, nous démontrons les limites pratiques de l'apprentissage d'a priori dans le cadre non supervisé et comparons plusieurs types d'algorithmes d'apprentissage de dictionnaires déroulés pour les problèmes inverses non supervisés et semi-supervisés.

Troisième défi: prédiction à partir des signaux M/EEG. Les données de magnétoencéphalographie peuvent contenir de nombreuses informations sur les processus cognitifs ou sur l'état biologique d'un sujet donné. Pour accéder à ces informations, il est très intéressant de concevoir des outils de prédiction capables de traiter ce type spécifique de données. Les applications de la M/EEG comprennent l'interface cerveau-ordinateur – qui consiste à établir une interface de communication entre le cerveau et un dispositif externe – la prédiction de l'âge du cerveau – qui peut aider à caractériser le vieillissement biologique et la gravité des maladies – ou le dépistage de la démence – où nous voulons détecter si un sujet présente des signes avant-coureurs d'une maladie neurodégénérative.

Une pratique courante de l'apprentissage statistique sur les données M/EEG consiste à considérer que la quantité d'intérêt est une fonction de la puissance des sources. Comme la résolution du problème inverse pour récupérer ces sources est coûteuse et incertaine, nous sommes confrontés à un troisième défi : nous devons concevoir des méthodes capables d'effectuer des prédictions précises directement à partir des mesures.

Dans ce travail, nous proposons une nouvelle méthode de traitement des signaux M/EEG basée sur les matrices de covariance des mesures. Nous nous appuyons sur des outils de transport optimal et de géométrie riemannienne pour concevoir une nouvelle distance sur les distributions de matrices symétriques définies positives et évaluons son efficacité sur plusieurs tâches de prédiction en M/EEG.

Remerciements

Je tiens d'abord à remercier le jury d'avoir accepté d'évaluer mes travaux de thèse, que ce soit au travers des rapports sur le manuscrit ou des questions et des retours sur la soutenance. J'en profite pour remercier chaleureusement Rémi et Bruno d'avoir accepté de nous dépanner malgré le court délai dont on disposait.

Ensuite, j'aimerais remercier Thomas et Matthieu, mes directeurs de thèse. Pendant ces trois ans, vous m'avez fait découvrir le monde de la recherche, vous m'avez fait progresser dans ma manière de raisonner, d'expérimenter, de coder, de communiquer, et vous m'avez soutenu aussi bien dans les bons moments que dans les moments de doute. Vous m'avez donné la liberté d'essayer, m'avez poussé à m'accrocher, à creuser, et à comprendre les choses. Durant tous nos échanges, j'ai eu l'impression de parler avec vous d'égal à égal, de chercheur à chercheur. J'ai pu donner mon avis librement, sans peur de l'erreur ou du jugement, j'ai pu décider de mes projets de recherche, qu'ils aboutissent ou pas, et j'ai toujours pu parler avec vous ouvertement des problèmes que je rencontrais. Pour toutes ces raisons, je profite de cette occasion pour vous dire merci, pour vos connaissances et votre savoir-faire, et aussi pour votre gentillesse et votre humanité. Je vous souhaite à tous les deux le meilleur pour la suite, et à toi Thomas plein de bonheur dans ta future vie de papa.

Maintenant, j'aimerais remercier l'équipe, car les trois ans que j'ai passés ici n'auraient pas été les mêmes sans vous. Au fil des joyeuses discussions, des déjeuners et des pauses, au fil des échanges scientifiques et des collaborations, au fil des sprints de code, des séminaires d'équipe et des conférences, et au fil des foots et des afterworks à l'ambiance chaleureuse et souvent très festive, j'ai eu l'occasion de mieux connaître chacun d'entre vous. Durant tous ces moments qu'on a passés ensemble, on a découvert la recherche, on a créé des liens, on a partagé nos opinions et nos expériences, parfois nos joies et nos peines, et on s'est raconté nos vies et nos histoires. Je ressors de ces trois ans de thèse avec des souvenirs que je garderai toute ma vie. Je tiens à vous remercier pour tous les moments que j'ai passés avec vous, j'espère que nous aurons l'occasion de nous revoir régulièrement, et dans tous les cas je vous souhaite à tous de la réussite et du bonheur dans la suite de vos parcours en recherche, et dans la suite de vos vies.

Durant mon parcours académique qui se conclut aujourd'hui, j'ai eu la chance de rencontrer

plein de gens formidables avec qui j'ai tant appris et partagé, et que je remercie. En ce qui concerne mes années de thèse, merci à Clément Bonet pour tous nos échanges et nos discussions scientifiques, avec en prime la publication d'un article dont je suis très fier et sur lequel j'ai beaucoup aimé travailler avec toi, et merci à Louis Charlot pour toutes les heures que tu as passées à m'écouter, à me conseiller et à me soutenir, et pour ton amitié.

Enfin, je tiens à remercier mes proches et en particulier ma sœur Anna et mes parents May et Francis, pour votre soutien inconditionnel au quotidien et pour votre aide dans les périodes difficiles. Je ne serais jamais arrivé là où je suis sans vous, j'en suis conscient, et je profite de cette occasion pour vous exprimer ma gratitude et mon affection. Merci beaucoup.

Contents

Notations

$\mathbb{N}$	Set of integers
$\mathbb{R}$	Set of real numbers
$\mathbb{R}^n$	Set of real valued vectors of dimension n
$\mathbb{R}^{n\times m}$	Set of real valued matrices of dimension $n \times m$
$\mathbb{1}_E$	0-1 indicator function of a set E
$A^\dagger$	Pseudo-inverse of $A \in \mathbb{R}^{n\times m}$
$A^\top$	Transpose of $A \in \mathbb{R}^{n\times m}$
$\mathrm{Tr}(A)$	Trace of A
$\ker A$	Kernel space of $A \in \mathbb{R}^{n\times m}$
$\odot$	Hadamard product (element-wise product)
$*$	Convolution product
$\otimes$	Product measure
$E^\perp$	Orthogonal to the vector space E
$\langle u, v\rangle_E$	Inner product between $u \in E$ and $v \in E$
$\|\cdot\|_0$	Number of non-zero coordinates (ℓ_0 pseudo-norm)
$\|\cdot\|_p$	Euclidean ℓ_p norm
$\|\cdot\|_\infty$	Euclidean ℓ_∞ norm
$\|\cdot\|_{\mathrm{TV}}$	Total Variation norm
∂_f	Sub-gradient of f
∇f	Gradient of f
sgn	Sign operator
prox_f	Proximal operator of f
f^*	Fenchel transform of f
ι_C	0-∞ indicator function of C
$\mathcal{P}_p(E)$	Set of probability measures with moment p defined on Borel sets of E
$\mathcal{N}(\mu, \Sigma)$	Gaussian distribution of mean μ and covariance matrix Σ
$T_\#$	Push-forward operator for the measurable map T
$S_d(\mathbb{R})$	Set of real valued symmetric matrices of dimension d
$S_d^{++}(\mathbb{R})$	Set of real valued symmetric positive definite matrices of dimension d
$\mathcal{O}_d$	Set of orthogonal matrices of dimension d

Motivation

Magnetoencephalography and electroencephalography (M/EEG) are non invasive neuroimaging modalities that record the electrical activity of the brain. Concretely, a few dozen or hundred sensors are placed around the head of a subject, and the machine records the intensity of the electric or magnetic field at high temporal resolution, of the order of 1000Hz. This makes it possible to follow the temporal and spatial evolution of the electrical currents flowing in the dendrites of neurons during synaptic transmission. The resulting data are made up of as many time series as there are sensors and can contain a large volume of samples, varying with the duration of the recording. The purpose of this process is to provide information about the cognitive process or biological state of a subject. There are many ways to extract useful information from this data, depending on the downstream task. Each one of them comes with specific challenges.

First challenge: efficient pattern extraction from M/EEG data. A method to automate the analysis of M/EEG signals consists in extracting cognitive patterns that correspond to physiological activities, such as recurrent artifacts like heartbeats or eye blinks, or external auditory or visual stimuli that are presented to a subject. As a matter of fact, the identification of prototypical waveforms and their occurrences in the data allows to design an event-based representation of the temporal dynamic of the signal, and link the stimuli to the evoked responses they trigger. This representation makes it possible to develop statistical tools to determine the latency of the triggering process, as well as its intensity.

One way to achieve this is to leverage Convolutional Dictionary Learning, which is particularly effective for pattern learning tasks on time series or images. However, optimization methods for Convolutional Dictionary Learning struggle to scale to the large amounts of data considered in real-world applications such as MEG data analysis. In particular, these methods work well on small recordings from one specific subject, but are hardly applicable to cohorts of tens or hundreds of individuals. Thus, providing new tools to analyze M/EEG recordings at the population level presents a major bottleneck: we must first improve the computational efficiency of Convolutional Dictionary Learning. To do so, we have to find faster algorithms or make acceptable concessions that preserve the quality of the final result.

In the following, we study to what extent the usage of automatic differentiation and unrolled algorithms – a special kind of neural network based on iterative optimization schemes – allows to speed up Dictionary Learning for large data. We also provide tools to make Convolutional Dictionary Learning faster in practice, as well as perspectives of applications in MEG data analysis.

Second challenge: prior learning for linear inverse problems. Measuring the electrical activity at the surface of the head through M/EEG comes with a loss of potentially critical information concerning the localization of the electrical sources inside the brain. Indeed, each sensor records the signal emitted by each source with intensity depending on the distance between them and the physical properties of the brain tissues the electromagnetic wave has to go through. Due to the linearity of Maxwell's equations, the observed signal is obtained as a linear transformation of the sources, corrupted by measurement noise. Combining Finite or Border Elements Methods to MRI data and knowledge of the brain tissues allows to compute a numerical approximation of this linear transformation. Given that we know both the observations and the linear transformation, we are interested in recovering the sources that generated these observations. This is called the source localization problem, and it is a specific type of linear inverse problem.

As the dimension of the measurements – provided by a few hundreds sensors – is usually much smaller than the dimension of the sources – a few thousand voxels in the brain – this inverse problem is ill-posed, meaning that several solutions could be correct given a set of observations. The uncertainty on the measurements, corrupted by noise, increases the number of potential solutions. Therefore, practitioners rely on prior knowledge of the data to select the most plausible solution among all possible ones. In some cases, efficient and analytic transforms are available and produce satisfying results, such as wavelets for images or Gaborlets for audio signals. However, the complexity and the variability of the signals often make it hard to rely on *ad hoc* priors or dictionaries, especially in the unsupervised scenario of M/EEG where no ground truth data are available. Thus, a second challenge arises: we are looking for an efficient and robust way of summarizing the structure of the data in order to solve a linear inverse problem without the need of expert knowledge.

Here, we put the focus on the possible usage of unrolling as a prior learning method for inverse problems resolution. More specifically, we demonstrate practical limitations of prior learning in the unsupervised setting, and compare several kinds of unrolled dictionary learning algorithms for unsupervised and self-supervised inverse problems.

Third challenge: prediction from M/EEG data. M/EEG data may contain a lot of information on cognitive processes or on the biological state of a given subject. To access this information, it is of great interest to design prediction tools that are able to deal with this specific type of data. Applications of M/EEG include Brain Computer Interface – which consists of establishing a communication interface between the brain and an external device – brain-age prediction – which can help characterize biological aging and disease severity – or dementia screening – where we want to detect whether a subject shows warning signs of a neurodegenerative disease.

A common practice in statistical learning on M/EEG data is to consider that the target

is a function of the power of the sources. As solving the inverse problem to recover these sources is costly and uncertain, we are faced with a third challenge: we need to design methods that are able to perform accurate predictions directly from the measurements.

In this work, we propose a new way to process M/EEG signals based on covariance matrices from the measurements. We leverage tools from optimal transport and Riemannian geometry to design a new distance on distributions of Symmetric Positive Definite matrices and evaluate its efficiency on several prediction tasks in M/EEG.

Organization of this work. The thesis is divided into two parts. The first one studies unrolled dictionary learning and its usage in inverse problems. The second one focuses on M/EEG applications.

Publications

Benoît Malézieux, Thomas Moreau, and Matthieu Kowalski. Understanding approximate and unrolled dictionary learning for pattern recovery. *International Conference on Learning Representations*, 2022a

Benoît Malézieux, Thomas Moreau, and Matthieu Kowalski. Apprentissage de dictionnaire par différentiation automatique pour la résolution de problèmes inverses. *GRETSI*, 2022b

Clément Bonet, Benoît Malézieux, Alain Rakotomamonjy, Lucas Drumetz, Thomas Moreau, Matthieu Kowalski, and Nicolas Courty. Sliced-wasserstein on symmetric positive definite matrices for m/eeg signals. *International Conference on Machine Learning*, 2023a

Florent Michel, Benoît Malézieux, Thomas Moreau, and Matthieu Kowalski. L'entropie comme mesure de difficulté des problèmes inverses. *GRETSI*, 2023

Part I

Understanding unrolled dictionary learning and its usage in inverse problems

A common problem encountered in observational sciences is the design of a procedure that recovers physical quantities of interest from a set of measurements. A good knowledge of the law of physics and of the measurement devices allows one to model the process that transforms the input into the output, which is called the *forward model*. Thus, in the case where the forward model is available, it is straightforward to calculate the set of measurements that would be observed for a set of input signals. However, the practitioner is often interested in the reverse procedure: given a forward model and a set of measurements, what is the set of signals that caused those measurements ? This problem is called an *inverse problem*, and is found in a wide variety of applications.

- **Image processing**. A camera is a good example of machine designed to capture information about our surrounding environment. What it does is to analyze the number and the wavelength of photons emitted by a source to create an image that matches what we see in our daily life. However, as every physical system, the camera makes some mistakes and often produces degraded results which have to be processed to recover the original one. For example, the image could be noisy, blurred or lacking some pixels. Given the observed image and the knowledge of the acquisition process, one would like to recover the image as the human eye would have seen it. Examples of processing techniques can be found in Burger and Burge [2016].

- **Neuroimaging**. It is not always possible to get an exact representation of the input signal. In image processing, we expect the image to look like what we see. But in other areas like neuroimaging, the quantities of interest are hidden, for example in the brain, and it is only possible to measure their effects outside the head. In MRI and M/EEG, the device records electric and magnetic field intensities to get an idea on the subject's brain activity. The laws of physics and Maxwell's equations provide a framework to compute the effects when the causes are known, but the other way around – the inverse problem – is trickier to deal with. Indeed, the number of areas of interest in the brain is very large compared to the number of sensors that are placed over the head. Thus, the resolution generally relies on expert knowledge on the signal [Gramfort et al., 2012, Costa et al., 2017].

- **Audio**. Given a setup where a few microphones record a conversation or a song, several applications involve separating the sound sources – meaning to isolate each voice or instrument. A microphone outputs a mixture of the sources, that can be modeled as a linear transformation. If this mixture is known, the source separation problem is a linear inverse problem for which the practitioner knows the forward model. Otherwise, the problem is called blind source separation and requires other tools, like the ones presented in Ozerov and Févotte [2009].

- **Astrophysics**. Solving an inverse problems means retrieving unknown causes of known effects. This scenario happens in plenty of contexts in physics. For instance, it is of interest to compute the gravitational field in some areas of space to study the celestial bodies or matter they contain. In this case, the effects are known because they can be observed with appropriate instruments, and the goal of the problem is to compute the operator which models the physical phenomenon. More details on inverse problems in astrophysics can be found in Starck [2016].

Mathematically speaking, the problem can be modeled as follows. Let $x \in \mathbb{R}^n$ be a real valued signal of dimension n, and $y \in \mathbb{R}^m$ be a real valued measurement of dimension m. We denote $\mathcal{A}$ the forward model which allows to compute y from x. The additive noise modeling the degradation of the signal through the measurement device is denoted by the real valued random vector $b \in \mathbb{R}^m$. Then we have

$$y = \mathcal{A}(x) + b \ . \tag{1}$$

In the case where $\mathcal{A}$ is a linear operator, we use the corresponding matrix $A \in \mathbb{R}^{m \times n}$, leading to

$$y = Ax + b \ . \tag{2}$$

In the following, we will only consider linear inverse problems with Gaussian noise, *i.e.* $b \sim \mathcal{N}(0, \sigma^2 I_m)$ where $\sigma > 0$ and $I_m \in \mathbb{R}^{m \times m}$ is the identity matrix of dimension m. It may seem very restrictive to constrain $\mathcal{A}$ to be linear. However, this is realistic for a lot of applications, including imaging or neuroimaging. Regarding the noise, b is not always normally distributed, but this approximation makes the computations easier and gives acceptable results in practice in a wide variety of applications. We now go into the details of classical and modern resolution methods for linear inverse problems.

Chapter 1

Background on sparse representations and optimization for inverse problems

When the output is noiseless, *i.e.* $b = 0$, the inverse problem reduces to a matrix inversion problem. If the dimension n of the input and the dimension m of the output are equal, and if the matrix A is invertible, then the solution is unique and thus well defined, and solving the linear system provides an exact solution. Otherwise, the problem is more complex to deal with.

In our main case of interest, the dimension of the observation is smaller than the dimension of the signal, *i.e.* $m < n$. This scenario is called under-determined and is present in various areas of signal processing, as in M/EEG where there is a 1 to 100 ratio between m and n. A first attempt to solve the problem may involve finding the Ordinary Least Square (OLS) solution. Denoting $\|\cdot\|_2$ the euclidean ℓ_2 norm, we wish to solve

$$\min_{x \in \mathbb{R}^n} \|y - Ax\|_2^2 \ . \tag{1.1}$$

The least square solution x_{LS} is the orthogonal projection of y onto $(\ker A)^\perp$. Let $x_0 \in \ker A$, then $x_0 + x_{LS}$ is a solution. Thus, there are an infinite number of solutions and the problem is said to be *ill-posed*. Moreover, the noiseless case is unrealistic in signal processing, and b can't be considered equal to 0. In the case of Gaussian noise, the least square solution performs poorly because the noise level is not taken into account in the model. Indeed, $\|y - Ax\|_2^2$ has to be close to $m\sigma^2$ in order to respect the statistics of the measurement.

This simple example shows that trying to solve an under-determined inverse problem without adding any information to the model is unrealistic. Thus, some prior knowledge has to be integrated in order to select a plausible solution.

1.1 . Sparse representations and optimization for signal reconstruction

In order to solve the issues noticed in OLS, one idea is to leverage the information we have about the signal to avoid selecting bad solutions. This information is called a prior and it can take many forms. One simple example is the usage of a regularization term that can be added to the data-fitting term in Equation 1.1. A function $\mathcal{R}$ is designed by the practitioner depending on the properties of the signal he wants to preserve, and is then integrated to the model. It is associated with a hyper-parameter $\lambda > 0$ to adapt the penalization to the data and the noise level. The optimization problem we would like to solve becomes

$$\min_{x \in \mathbb{R}^n} \frac{1}{2}\|y - Ax\|_2^2 + \lambda \mathcal{R}(x) \ . \tag{1.2}$$

As an example, let's consider the well-known Ridge or Tikhonov regularization [Ito and Jin, 2014], where $\mathcal{R}(x) = \|x\|_2^2$. This time, the problem is strictly convex and admits a unique solution, defined as $x_{\text{Ridge}} = (A^\top A + \lambda I_n)^{-1} A^\top y$ where $A^\top$ is the transpose of A, and λ can be tuned to fit the statistics of the noise. Ridge regularization provides a way to penalize vectors with high ℓ_2 norms.

Many other types of regularization exist and the choice of $\mathcal{R}$ is crucial in order to get an acceptable solution for a given problem. We now focus on the design of a suitable $\mathcal{R}$ by leveraging one of the concepts that has had the most significant impact on the field of inverse problems in the last thirty years: sparsity.

Sparsity and the Lasso

A vector or matrix is called sparse if most of its entries are equal to zero. The main interest of sparsity is that it makes the recovery process easier because it is not necessary to take as many measurements as components in the signal, as studied in compressed sensing theory [Foucart and Rauhut, 2013a]. In our framework where $m < n$, it seems relevant to take advantage of sparsity to build a penalization.

Let's first focus on sparsity-inducing regularizations $\mathcal{R}$, *i.e.* functions that prioritize signals with a significant proportion of coordinates equal to 0. The most straightforward example is the ℓ_0 pseudo-norm [Elad, 2010a], which is defined as

$$\|x\|_0 = |\{i, \ x_i \neq 0\}| \ . \tag{1.3}$$

The ℓ_0 pseudo-norm gives the number of non-zero coefficients in a signal. A way to force most of them to be equal to zero is to constrain $\|x\|_0$ to stay under an upper bound $k_0 > 0$. The optimization problem in Equation 1.2 then becomes

$$\min_{x \in \mathbb{R}^n} \|Ax - y\|_2^2 \quad \text{s.t.} \quad \|x\|_0 \leq k_0 \ . \tag{1.4}$$

One of the main challenge of the optimization problem in Equation 1.4 is that it is a non-convex and NP-hard problem. There exist several heuristics and greedy algorithms to solve it, like matching pursuit algorithms [Mallat and Zhang, 1994], and some of them are detailed in Elad [2010a]. In order to benefit from convex optimization tools and make the problem easier to solve, one idea is to replace Equation 1.4 by a convex relaxation which promotes sparsity. Natural candidates are ℓ_p norms, which are defined as

$$\|x\|_p = \Big(\sum_{i=1}^{n} x_i^p\Big)^{\frac{1}{p}} \ . \tag{1.5}$$

Taking $\mathcal{R}(x) = \|x\|_p$, the optimization problem in Equation 1.2 becomes convex when $p \geq 1$, and leads to sparse solutions when $p \leq 1$. This observation led to the introduction of the Least Absolute Shrinkage and Selection Operator (Lasso) problem in Tibshirani [1996], a convex optimization problem of the form

$$\min_{x \in \mathbb{R}^n} \frac{1}{2}\|Ax - y\|_2^2 + \lambda \|x\|_1 \ . \tag{1.6}$$

Properties of the Lasso.

As the cost function is convex and bounded below, the set of solutions is convex and non empty. Therefore, it contains either a unique or an infinite number of solutions. Moreover, if the entries of A are independent samples drawn from a continuous probability distribution, then the solution is unique with probability 1 [Tibshirani and Ryan, 2012].

An optimal solution should verify the KKT conditions, *i.e.*

$$0 \in A^T(Ax - y) + \lambda \partial_{\|\cdot\|_1}(x) \ , \tag{1.7}$$

where $\partial_{\|\cdot\|_1}$ is the sub-gradient of the l_1 norm in dimension 1, defined coordinate-wise as

$$\partial_{\|\cdot\|_1}(x)_i = \begin{cases} \operatorname{sgn}(x_i) & \text{if } x_i \neq 0 \\ [-1, 1] & \text{if } x_i = 0 \end{cases} \ . \tag{1.8}$$

When $\lambda \geq \|A^\top y\|_\infty$, 0 satisfies the KKT conditions and it is therefore an optimal solution. Moreover, the value of λ determines the number of zero coordinates in x. Indeed, if $\lambda \geq |(A^\top y)_i|$, then $x_i = 0$. The set of indexes of non-zero coordinates of a solution is called the *support*.

The dual of the Lasso also provides information on the solution. Denoting $r = Ax - y$ the residual, and $\theta \in \mathbb{R}^m$ the dual variable, the problem can be written as

$$\max_{\theta} \min_{r,x} \frac{1}{2}\|r\|_2^2 + \lambda \|x\|_1 + \theta^\top (r - Ax + y) \ . \tag{1.9}$$

Minimizing over r leads to

$$\min_{r} \frac{1}{2}\|r\|_2^2 + \theta^\top r = -\frac{1}{2}\|\theta\|_2^2 \ , \tag{1.10}$$

with $r = \theta$, and minimizing over x leads to

$$\min_{x} \lambda \|x\|_1 - \theta^\top A x = -\iota_{\mathcal{F}_\lambda(A)}(\theta) \ , \tag{1.11}$$

where ι_E is the $0 - \infty$ indicator function of the set E, and $\mathcal{F}_\lambda(A) = \{\theta \mid \|A^T\theta\|_\infty \leq \lambda\}$. Hence the problem becomes

$$\max_{\theta} -\frac{1}{2}\|\theta\|_2^2 + \theta^\top y - \iota_{\mathcal{F}_\lambda(A)}(\theta) \tag{1.12}$$

$$= \max_{\theta} \frac{1}{2}(\|y\|_2^2 - \|y - \theta\|_2^2) - \iota_{\mathcal{F}_\lambda(A)}(\theta) \ . \tag{1.13}$$

Thus, the maximum is reached when θ minimizes $\|y - \theta\|_2$ over the convex set $\mathcal{F}_\lambda(A)$, and the residual r is equal to $\operatorname{proj}_{\mathcal{F}_\lambda(A)}(y)$, *i.e.* the projection of y onto the convex set $\mathcal{F}_\lambda(A)$.

Algorithm 1 FISTA

y, A, λ, T
$x_0 = 0$, $t = 1$, $\alpha_0 = 1$
Compute the Lipschitz constant L of $A^\top A$
while $t \leq T$ **do**
  $v_{t+1} \leftarrow \mathrm{ST}_{\frac{\lambda}{L}}\big(x_t - \frac{1}{L}A^\top(Ax_t - y)\big)$
  $\alpha_{t+1} \leftarrow \frac{1+\sqrt{1+4\alpha_t^2}}{2}$
  $x_{t+1} \leftarrow v_{t+1} + (\frac{\alpha_t - 1}{\alpha_{t+1}})(v_{t+1} - v_t)$
  $t \leftarrow t + 1$
end while

More information about the Lasso can be found in Hastie et al. [2015].

Algorithms.

As opposed to Ridge regression, the Lasso does not have a closed-form solution. Thus, iterative algorithms are generally used to find a precise enough approximation of a solution. Introduced in Daubechies et al. [2004], the Iterative Shrinkage Thresholding Algorithm (ISTA) produces a sequence $(x_k)_{1 \leq k \leq T}$, such that at each iteration $1 \leq t \leq T - 1$

$$x_{t+1} = \mathrm{ST}_{\tau\lambda}\big(x_t - \tau A^\top(Ax_t - y)\big) \ , \tag{1.14}$$

where τ is a step-size and where ST is the Soft-Thresholding operator defined for all index $1 \leq i \leq n$ as

$$\mathrm{ST}_\gamma(x)_i = \mathrm{sgn}(x_i)\max(0, |x_i| - \gamma) \ . \tag{1.15}$$

With an appropriate choice of τ, ISTA converges toward a fixed point of the function $x \mapsto \mathrm{ST}_{\tau\lambda}\big(x - \tau A^\top(Ax - y)\big)$ at rate $\mathcal{O}(\frac{1}{t})$, and the set of fixed points corresponds to the set of solutions of the Lasso in Equation 1.6. This happens for $0 < \tau < \frac{2}{L}$, L being the highest eigenvalue of $A^\top A$. See Tseng [2010] for details.

There exist many algorithms designed to solve the Lasso more efficiently. One of the most well-known example is called Fast ISTA (FISTA) [Beck and Teboulle, 2009], described in Algorithm 1, and is an adaptation of gradient descent with momentum. It has a convergence rate of $\mathcal{O}(\frac{1}{t^2})$. Variations have been developed, for instance in Chambolle and Dossal [2014] where the authors study other possible linear combinations between the iterates, taking $\alpha_t = \frac{t+a-1}{a}$ with $a > 2$. Other algorithms to solve the Lasso are presented in Hastie et al. [2015].

Sparse representations and splitting algorithms for reconstruction

As is, the Lasso may not seem appropriate in the context of signal processing. Indeed, natural signals are rarely sparse in time or in space. Using the examples previously mentioned, images have numerous non-zero pixels, and the same is true for audio or M/EEG recordings. However, looking at the signal the right way makes things much easier.

As mentioned in Foucart and Rauhut [2013a], many real-world signals can be approximated by sparse components if the representation is appropriately chosen. For instance, let's

consider a family of signals generated as linear combinations of elements from a set of basic bricks, or *atoms*. The assumption that one signal taken from this family is made of a few bricks only does not appear too restrictive. For instance, in M/EEG, one chunk of signal may correspond to a heartbeat, another to eye-blinks and another to a visual stimulation. Thus, the signal can be considered as sparse when decomposed on that "basis". Taking another example, let's consider a piece-wise constant signal. It can be fully recovered by knowing the value at the origin, and the location and intensity of each discontinuity, which are sparse. In a lot of applications, looking at the signal the right way allows simple and sparse representations of the signal to be found.

Two challenges emerge from this approach. First, one needs to find either the family of atoms that generated the signals of interest, or a transform which gives access to a representation of the signal in a well-chosen space. This can be done either manually from expert knowledge, or automatically by learning it from the data. Second, one needs to recover the linear combination of atoms that generated each measurement, with the key assumption that this representation is sparse. This procedure is known as *sparse coding* [Elad, 2010a].

Analysis and Synthesis.

Mathematically speaking, two formulations of the problem are widely used: Analysis and Synthesis [Elad et al., 2007].

- Analysis. A classical approach considers that the signal to reconstruct x can be represented as a sparse vector in an unknown space. A first way to impose such prior is to assume that there exists a forward transform $\Gamma \in \mathbb{R}^{L \times n}$, which makes the signal sparse. This formulation is called Analysis, and comes with the following optimization problem

$$\min_{x \in \mathbb{R}^{n \times T}} \frac{1}{2} \|Ax - y\|_2^2 + \lambda \|\Gamma^\top x\|_1 \ . \tag{1.16}$$

  Here, the link with the initial regularized problem in Equation 1.2 is straightforward as $\mathcal{R}(x) = \|\Gamma^\top x\|_1$.

- Synthesis. Another possibility is to assume that the signal can be decomposed into a sparse representation in a redundant basis of patterns or atoms. In other words, the goal is to recover sparse codes $z \in \mathbb{R}^{L \times T}$ from a dictionary $D \in \mathbb{R}^{n \times L}$ and noisy measurements y. This formulation is called Synthesis, as the signal is synthesized from the dictionary, *i.e.* $x = Dz$, and it consists in solving the Lasso-based optimization problem given by

$$\min_{z \in \mathbb{R}^{L \times T}} \frac{1}{2} \|ADz - y\|_2^2 + \lambda \|z\|_1 \ . \tag{1.17}$$

A detailed comparison between the two formulations is provided in Elad et al. [2007]. The simplest case appears when D or $\Gamma^\top$ are invertible. Then the two problems are equivalent by doing the change of variable $Dz = x$ or $\Gamma^\top x = z$. It is quite usual to take $L \geq n$, meaning that the length of the sparse code is larger than the length of the signal and the

measurement, in order to store more information in the dictionary. Then, the dictionary is called redundant because the columns don't form a basis, as their rank is smaller than n. In this case, there is no straightforward relation between Equation 1.16 and Equation 1.17.

Focus on optimization: splitting algorithms

While solving Synthesis simply reduces to solving the Lasso, Analysis requires other tools from convex optimization to solve minimization problems involving composite convex functions. Details and proofs can be found in Boyd and Vandenberghe [2004], Bauschke et al. [2011] and Parikh et al. [2014].

Let $f : \mathbb{R}^n \to]-\infty, \infty]$ be a closed proper convex function, meaning that the epigraph

$$\text{epi} f = \{(x, \lambda) \in \mathcal{X} \times \mathbb{R}, f(x) \leq t\} \tag{1.18}$$

is a nonempty closed convex set. Then the *proximal operator* of f $\text{prox}_f : \mathbb{R}^n \to \mathbb{R}^n$ is defined as

$$\text{prox}_f(y) = \underset{x}{\text{argmin}}\, f(x) + \frac{1}{2}\|x - y\|_2^2 \ . \tag{1.19}$$

This operator is well-defined for all $y \in \mathbb{R}^n$ as f is convex and not everywhere infinite, and $\|\cdot - y\|_2^2$ is strongly convex. Proposition 1.1.1 makes a connection between proximal operators and fixed point theory, and states that the set of fixed points of prox_f is the set of minimizers of f.

PROPOSITION 1.1.1. *$x^* \in \mathbb{R}^n$ minimizes f if and only if $x^* = \text{prox}_f(x^*)$.*

Thus, minimizing f reduces to solve the fixed point equation $x = \text{prox}_f(x)$. Solutions of fixed point equations of the form $x = T(x)$ where T is a contraction, *i.e.* $\|T\| < 1$, are generally found with iterative procedures generating sequences $(x_t)_{t\in\mathbb{N}}$ such that $x_{t+1} = T(x_t)$. Proximal operators are non expansive, meaning that their norm is inferior or equal to 1, but they are not contractions. Therefore, convergence is ensured by other properties. Proposition 1.1.2 shows how to build converging fixed point iterations from non expansive operators.

PROPOSITION 1.1.2. *Let N be a non expansive operator and $\alpha \in (0, 1)$. The α-averaged operator $T = \alpha N + (1 - \alpha)I$ has the same fixed points as N and the sequence $x_{t+1} = T(x_t)$ converges toward a fixed point of N.*

As stated in Proposition 1.1.3, proximal operators are actually α-averaged operators.

PROPOSITION 1.1.3. prox_f *is firmly non expansive,* i.e.

$$\|\text{prox}_f(x) - \text{prox}_f(y)\| \leq (x - y)^\top (\text{prox}_f(x) - \text{prox}_f(y)) \ .$$

Moreover, firmly non expansive operators are $\frac{1}{2}$-averaged.

Thus, the fixed point iterates $x_{t+1} = \text{prox}_f(x_t)$ will converge toward a minimizer of f.

Let's now focus on composite functions of the form $F = f + g$, where $f : \mathbb{R}^n \to \mathbb{R}$ and $g : \mathbb{R}^n \to]-\infty, \infty]$ are closed proper convex functions, and f is differentiable with L-

Lipschitz gradient. Minimizers x^* of F satisfy the Fermat condition

$$0 \in \nabla f(x^*) + \partial_g(x^*) \ . \tag{1.20}$$

Let $\tau > 0$, then

$$0 \in \tau\nabla f(x^*) + \tau\partial_g(x^*) \tag{1.21}$$
$$0 \in \tau\nabla f(x^*) - x^* + x^* + \tau\partial_g(x^*) \tag{1.22}$$
$$x^* = (I + \tau\partial_g)^{-1}(I - \tau\nabla f)(x^*) \ . \tag{1.23}$$

It turns out that the proximal operator $\text{prox}_{\tau g}$ is the *resolvent* of the sub-differential ∂_g with parameter τ, as stated in Proposition 1.1.4.

PROPOSITION 1.1.4. *Let $\tau > 0$. Then*

$$\text{prox}_{\tau g} = (I + \tau\partial_g)^{-1} \ .$$

Therefore, the Fermat condition reduces to

$$x^* = \text{prox}_g \circ (I - \tau\nabla f)(x^*) \ . \tag{1.24}$$

As soon as $\tau \leq \frac{1}{L}$, where L is the Lipschitz constant of ∇f, the *forward-backward operator* $\text{prox}_g \circ (I - \tau\nabla f)$ is α-averaged and the fixed point iterates $x_{t+1} = \text{prox}_g \circ (I - \tau\nabla f)(x_t)$ converge toward a minimizer of $F = f + g$.

The proximal operator provides a simple optimization procedure to get a solution of $\min_x F(x)$. However, it is not always straightforward to compute. One example where this is feasible is the Lasso, where $g(x) = \lambda\|x\|_1$. Indeed, $\text{argmin}_x \frac{1}{2}\|x - y\|_2^2 + \lambda\|x\|_1$ can be obtained with the KKT conditions coordinate-wise

$$0 \in x_i - y_i + \lambda\partial_{|\cdot|}(x_i) \tag{1.25}$$
$$x_i = \begin{cases} 0 & \text{if } |y_i| - \lambda \leq 0 \\ \text{sgn}(y_i)(|y_i| - \lambda) & \text{otherwise} \end{cases} \ , \tag{1.26}$$

and we recognize the Soft-Thresholding operator. Thus, the ISTA is simply the forward-backward splitting algorithm applied to the ℓ_1 norm.

The Analysis formulation implies minimizing composite functions of the form $F = f + g \circ \Gamma^\top$, where Γ is a linear operator. Besides when $\Gamma^\top$ is orthogonal, *i.e.* $\Gamma^\top\Gamma = \Gamma\Gamma^\top = I$, there is no simple formula to compute $\text{prox}_{g\circ\Gamma^\top}$ given prox_g. Thus, these problems are generally solved via *primal-dual* algorithms, which alternate minimization steps in the primal domain and maximization steps in the dual domain.

- PDHG.

  The Fenchel transform – also called convex conjugate – of a function f taking values on the extended real number line is defined as follows

$$f^*(y) = \sup_x \left(y^\top x - f(x)\right) \ . \tag{1.27}$$

Algorithm 2 PDHG

T, σ, τ
$x_0 = 0$, $y_0 = 0$, $t = 1$
while $t \leq T$ **do**
  $x_{t+1} \leftarrow x_t - \tau(\nabla f(x_t) + \Gamma y_t)$
  $y_{t+1} \leftarrow \text{prox}_{\sigma g^*}(y_t + \sigma\Gamma^\top(2x_{t+1} - x_t))$
  $t \leftarrow t + 1$
end while

f^* is convex, and $f^{**} = f$ whenever f is convex and proper. Thus

$$\min_x f(x) + g(\Gamma^\top x) = \min_x f(x) + g^{**}(\Gamma^\top x) \tag{1.28}$$

$$= \min_x \max_y f(x) + (\Gamma^\top x)^\top y - g^*(y) \ . \tag{1.29}$$

The most basic algorithm finds a saddle point by alternating a gradient descent over x with step size $\tau > 0$ and a proximal ascent over y with step size $\sigma > 0$, as follows

$$x_{t+1} = x_t - \tau(\nabla f(x_t) + \Gamma y_t) \tag{1.30}$$

$$y_{t+1} = \text{prox}_{\sigma g^*}(y_t + \sigma\Gamma^\top x_{t+1}) \ , \tag{1.31}$$

In Chambolle and Pock [2011], the authors update y_{t+1} differently by replacing x_{t+1} with $2x_{t+1} - x_t$ to obtain a converging algorithm when $\tau > 0$ and $\sigma > 0$ are chosen such that $\tau\sigma\|\Gamma^\top\|^2 \leq 1$, leading to the Primal-Dual Hybrid Gradient (PDHG) algorithm, described in Algorithm 2.

The Fenchel transform g^* and its proximal operator are sometimes easier to compute than $\text{prox}_{g \circ A}$. For instance, when $g = \lambda\|\cdot\|_1$, then $g^* = \iota_{\mathcal{B}(0,\lambda)}$ is the indicator function of the ball of center 0 and radius λ for the ℓ_∞ norm, *i.e.*

$$g^*(x) = \begin{cases} 0 & \text{if } \|x\|_\infty \leq \lambda \\ \infty & \text{otherwise} \end{cases} \ . \tag{1.32}$$

Therefore, its proximal operator prox_{g^*} is simply the projection on $\mathcal{B}(0, \lambda)$.

- ADMM.

  The Alternating Direction Method of Multipliers consists of replacing $\Gamma^\top x$ by another variable y and adding a Lagrangian term to the cost function in order to preserve the constraint $y = \Gamma^\top x$. The problem becomes

$$\min_{x,y} \max_z f(x) + g(y) + z^\top(\Gamma^\top x - y) + \frac{\gamma}{2}\|\Gamma^\top x - y\|_2^2 \ , \tag{1.33}$$

  where $\gamma > 0$ is a hyper-parameter. As in PDHG, it is possible to alternate between a minimization over x and y, and a gradient ascent step over the Lagrange multiplier z, as described in Algorithm 3. There exist strong links between PDHG, ADMM and another algorithm called Douglas-Rachford splitting that won't be described here.

Variants and accelerations, as well as more in-depth theoretical foundations, can be found in Condat et al. [2019].

Algorithm 3 ADMM

T, γ
$x_0 = 0$, $y_0 = 0$, $z_0 = 0$, $t = 1$
while $t \leq T$ **do**
  $x_{t+1} \leftarrow \text{argmin}_x \, f(x) + z_t^\top(\Gamma^\top x) + \frac{\gamma}{2}\|\Gamma^\top x - y_t\|_2^2$
  $y_{t+1} \leftarrow \text{argmin}_y \, g(y) - z_t^\top y + \frac{\gamma}{2}\|x_{t+1} - y\|_2^2$
  $z_{t+1} \leftarrow z_t + \gamma(\Gamma^\top x_{t+1} - y_{t+1})$
  $t \leftarrow t + 1$
end while

1.2 . Dictionary Learning

While there exist efficient and analytic transforms that produce satisfying results on specific data, such as wavelets for images or Gaborlets for audio signals [Mallat, 2008], the signals complexity and variability often make it hard to rely on *ad hoc* priors or dictionaries. Therefore, it is of interest to learn sparse representations from ground truth or noisy data in a supervised setting.

Dictionary Learning [Olshausen and Field, 1997, Aharon et al., 2006, Mairal et al., 2009] is a common way to encode data-driven prior knowledge on signals. A typical example is pattern learning, which provides insightful information on the data in various biomedical applications. This includes the study of magnetoencephalography (MEG) recordings, where one aims to analyze the electrical activity in the brain from measurements of the magnetic field around the scalp of the patient [Dupré la Tour et al., 2018]. One may also mention neural oscillations study in the local field potential [Cole and Voytek, 2017] or QRS complex detection in electrocardiograms [Xiang et al., 2018], among others. This problem is typically seen as a factorization problem where the signal is assumed to be the product of a redundant basis of patterns or atoms – the dictionary – and of a sparse representation vector – the sparse codes. There are multiple ways of finding such decomposition, and this work focuses on Lasso-based Dictionary Learning.

Lasso-based Dictionary Learning.

Let $X \in \mathbb{R}^{n \times T}$ be a data-set of noisy signals. The goal is to recover a sparse code $Z \in \mathbb{R}^{L \times T}$ and a dictionary $D \in \mathbb{R}^{n \times L}$ from the signals X which are supposedly obtained as the linear transformation DZ, corrupted with noise $B \in \mathbb{R}^{n \times T}$: $X = DZ + B$. Sparsity-based optimization problems related to Dictionary Learning generally rely on the usage of the ℓ_0 or ℓ_1 regularizations. Here, we study Lasso-based Dictionary Learning where the dictionary D is learned in a convex set of constraints $\mathcal{C}$ by solving

$$\min_{Z \in \mathbb{R}^{L \times T}, D \in \mathcal{C}} F(Z, D) \triangleq \frac{1}{2}\|DZ - X\|_2^2 + \lambda\|Z\|_1 \ . \tag{1.34}$$

The signal estimate $\hat{X} = \hat{D}\hat{Z}$ is invariant to several transformations of $(\hat{Z}, \hat{D}) \in \mathbb{R}^{L \times T} \times \mathcal{C}$.

- Scale invariant. Let $\alpha > 0$, and $(\hat{Z}, \hat{D}) \in \mathbb{R}^{L \times T} \times \mathcal{C}$. Then $\hat{X} = \hat{D}\hat{Z} = \alpha\hat{D}\frac{1}{\alpha}\hat{Z}$. Therefore, $(\hat{Z}, \hat{D})$ and $(\frac{1}{\alpha}\hat{Z}, \alpha\hat{D})$ are equivalent representations of the signal. However, if $\alpha > 1$, then $F(\hat{Z}, \hat{D}) < F(\frac{1}{\alpha}\hat{Z}, \alpha\hat{D})$ and the optimization problem in Equation 1.34

will prefer solutions with smaller ℓ_1 norm, and thus α as big as possible. To alleviate this issue, the constraint set $\mathcal{C}$ generally includes a normalization criterion, where each atom of D is constrained to belong to a ball of fixed radius.

- Permutation invariant. Let $P \in \mathbb{R}^{L \times L}$ be a permutation matrix, *i.e.* the identity matrix I_L with permuted columns, and $(\hat{Z}, \hat{D}) \in \mathbb{R}^{L \times T} \times \mathcal{C}$. Then $\hat{X} = \hat{D}\hat{Z} = \hat{D}P^\top P\hat{Z}$. Therefore, $(\hat{Z}, \hat{D})$ and $(P\hat{Z}, \hat{D}P^\top)$ are equivalent representations of the signal, and $F(\hat{Z}, \hat{D}) = F(P\hat{Z}, \hat{D}P^\top)$.
- Sign invariant. Let $S \in \mathbb{R}^{L \times L}$ be a sign change matrix, *i.e.* a diagonal matrix with coefficients 1 and -1, and $(\hat{Z}, \hat{D}) \in \mathbb{R}^{L \times T} \times \mathcal{C}$. Then $\hat{X} = \hat{D}\hat{Z} = \hat{D}SS\hat{Z}$. Therefore, $(\hat{Z}, \hat{D})$ and $(S\hat{Z}, \hat{D}S)$ are equivalent representations of the signal, and $F(\hat{Z}, \hat{D}) = F(S\hat{Z}, \hat{D}S)$.

All this shows that the optimization problem in Equation 1.34 is highly non-convex. Indeed, for each solution (Z, D), there are at least $2^L L! - 1$ other solutions given by permutations and change of sign of coordinates of Z and columns of D.

Besides normalization, other sets of constraints have been studied. For instance, $\mathcal{C}$ may constrain the atoms to belong to normalized convolutional kernels to perform Convolutional Dictionary Learning [Grosse et al., 2007], or to be orthonormal, *i.e.* belong to $\{U \mid UU^\top = I, \forall i \in I \ \|U_i\|_2 = 1\}$ [Yaghoobi et al., 2013]. Specific applications require physics-informed constraints, like rank one convolutional dictionaries used in Dupré la Tour et al. [2018] to extract meaningful patterns from M/EEG signals.

Algorithms and properties.

Finding a proper decomposition requires to solve a non-convex problem. What makes things simpler is that this problem is also bi-convex, meaning that it is convex in each variable Z or D when the other one is fixed and when the set of constraint $\mathcal{C}$ is convex as well. In particular, successful algorithms often exploit this and alternate between minimization over Z with fixed D, and minimization over D with fixed Z. This framework is called *Alternating Minimization*.

The most simple example is the Method of Optimal Direction (MOD). It consists of minimizing F over Z with a sparse coding algorithm like FISTA, and then performing a projected gradient descent over D. This produces a sequence of tuples $(Z_t, D_t)_{1 \leq t \leq T}$ as follows

$$Z_{t+1} = \underset{Z}{\operatorname{argmin}} \frac{1}{2}\|X - D_t Z\|_2^2 + \lambda \|Z\|_1 \tag{1.35}$$

$$D_{t+1} = \operatorname{proj}_{\mathcal{C}}(D_t - \tau_t (D_t Z_{t+1} - X) Z_{t+1}^\top) \ , \tag{1.36}$$

where $(\tau_t)_{1 \leq t \leq T}$ are step sizes. The initialization of the dictionary plays a key role in the performance of alternating minimization algorithms, as the problem is non convex. Usual methods include random initialization, or initialization from chunks of signals. The choice of λ also impacts the performance, and is quite difficult when the problem is unsupervised.

Other Dictionary Learning algorithms have been developed in the past years. Three major

examples are K-SVD [Aharon et al., 2006], Online Dictionary Learning [Mairal et al., 2009], and Proximal Alternating Linearized Minimization (PALM) [Bolte et al., 2013].

- K-SVD.

  K-SVD [Aharon et al., 2006] is a widely used heuristic to find a good solution to the ℓ_0 dictionary learning optimization problem

  $$\min_{D,X} \|X - DZ\|_2^2 \quad \text{s.t.} \quad \forall i \ \|Z_i\|_0 \leq T_0 \ , \tag{1.37}$$

  where $T_0 > 0$ is the maximal number of non-zero coordinates in the sparse codes. This can be seen as an extension of the k-means algorithm, where each data point belongs to several clusters instead of one. K-SVD is an alternate procedure, and consists of applying a sparse coding algorithm like Orthogonal Matching Pursuit [Pati et al., 1993] to find an appropriate solution X of the ℓ_0 problem when D is fixed, and then updating each column of the dictionary as follows. For each column d_k, we have

  $$\|X - DZ\|_F^2 = \|\big(X - \sum_{j \neq k} d_j z_j^\top\big) - d_k z_k^\top\|_F^2 \tag{1.38}$$
  $$= \|E_k - d_k z_k^\top\|_F^2 \tag{1.39}$$
  $$= \|E_k \Omega_k - d_k z_k^\top \Omega_k\|_F^2 \ , \tag{1.40}$$

  where Ω_k discards the zero entries in the row vector $z_k^\top$, and $E_k = X - \sum_{j \neq k} d_j z_j^\top$. Denoting $E_k \Omega_k = U \Lambda V^\top$ the SVD decomposition of $E_k \Omega_k$, the optimal new column d_k is the first column of U – with highest eigenvalue – and the new $z_k^\top \Omega_k$ is the first column of $\Lambda_{1,1} V$.

- Online Dictionary Learning.

  Dictionary learning methods often suffer from large computation time when dealing with large data-sets. Indeed, at each iteration, Alternating Minimization methods for the ℓ_1 problem require solving the Lasso for all data points, while K-SVD computes expensive decompositions to update the columns of the dictionary. Online dictionary learning [Mairal et al., 2009] alleviates this issue by processing the data sequentially to minimize a quadratic surrogate function of the empirical cost, while converging to a stationary point of the main objective function.

- PALM.

  PALM [Bolte et al., 2013] is a simple and yet efficient method which solves bi-variate optimization problems by alternating steps of proximal gradient descent. In the case of Dictionary Learning, it performs a proximal gradient descent step over Z and a proximal gradient descent step over D, i.e.

  $$Z_{t+1} = \mathrm{ST}_{\lambda \sigma_t}\big(Z_t - \sigma_t D_t^\top (D_t Z_t - X)\big) \tag{1.41}$$
  $$D_{t+1} = \mathrm{proj}_{\mathcal{C}}\left(D_t - \tau_t (D_t Z_{t+1} - X) Z_{t+1}^\top\right) \ . \tag{1.42}$$

  With an appropriate choice of step sizes $(\sigma_t, \tau_t)_{t \in \mathbb{N}}$, this produces a converging sequence $(Z_t, D_t)_{t \in \mathbb{N}}$ which tends to a local minimum of Equation 1.34.

The literature on Dictionary Learning provides various theoretical results on local minima, local recovery or convergence of optimization algorithms. Several works analyze Alternating Minimization, as done in Arora et al. [2015], Agarwal et al. [2016], Chatterji and Bartlett [2017], and show that if the initial dictionary is taken sufficiently close to the true solution up to permutation and sign invariant, then the algorithm will converge and recover the dictionary which generated the data up to a small error. The main assumption is that the support of the sparse code estimate is equal or contained in the true support at each iteration, which allows to compute lower bounds on the correlation between the gradient and the optimal direction. Then this lower bound is used to prove that the iterates of the algorithm converge to a proper solution with results based on convex optimization. Other works have focused on local and global optima in several Dictionary Learning frameworks. Some of them treat the case of ℓ_1-norm minimization and show that with sufficiently incoherent bases and sparse signals, local identifiability is guaranteed with high probability [Gribonval and Schnass, 2010, Gribonval et al., 2015]. Besides the ℓ_1 criterion, global optimality of the true dictionary have been studied in the context of square and invertible matrices [Sun et al., 2016], separable dictionary learning [Schwab et al., 2019] or in more general factorization contexts [Haeffele and Vidal, 2015].

Convolutional Dictionary Learning.

In order to adapt Dictionary Learning to shift invariant signals like time series or images, it is possible to replace the dictionary D by a set of convolutional filters $(d_k)_{1\leq k\leq L}$. This variant, called Convolutional Dictionary Learning [Grosse et al., 2007], involves solving an optimization problem of the form

$$\min_{z_k, d_k \in \mathcal{C}'} \frac{1}{2}\|\sum_{k=1}^{L} d_k * z_k - y\|_2^2 + \lambda \sum_{k=1}^{L} \|z_k\|_1 \ , \tag{1.43}$$

where $\mathcal{C}'$ is the set of filters with unit norm. Algorithms that find solutions to Equation 1.43 are essentially built on methods previously mentioned, including Alternating Minimization, and may include the usage of Fourier transforms [Wohlberg, 2015]. The main downside is that convolutions $d_k * z_k$ are costly to compute when the size of the signal gets large.

Learning dictionaries with Analysis.

While most Dictionary Learning methods focus on the Synthesis formulation in Equation 1.34, several works consider the Analysis counterpart [Peyré and Fadili, 2011, Yaghoobi et al., 2013, Li et al., 2017]. In other words, the problem to solve becomes

$$\min_{\hat{X}\in\mathbb{R}^{n\times T}, \Gamma\in\mathcal{C}_{\mathcal{A}}} F_A(X,\Gamma) \triangleq \frac{1}{2}\|\hat{X} - X\|_2^2 + \lambda\|\Gamma^\top \hat{X}\|_1 \ , \tag{1.44}$$

where $\mathcal{C}_{\mathcal{A}}$ is a set of constraint. The most classical constraint set chosen for $\mathcal{C}_A$ is the Unit Norm, where each atom is normalized, as in Synthesis. In Yaghoobi et al. [2013], the authors notice that it is insufficient to guarantee that the learned dictionary is nontrivial with Analysis. As a matter of fact, rank one dictionaries can be shown to be optimal under Unit Norm. Therefore, they propose to use the Unit Norm Tight Frame constraint $\{U \text{ s.t. } UU^\top = I, \forall i \in I \|U_i\|_2 = 1\}$ which is a subset of the Stiefel manifold corresponding to orthonormal k-frames with normalized atoms. Finally, $\mathcal{C}_A$ can also be chosen as normalized convolutional kernels. This last constraint corresponds to learning finite difference

schemes for Analysis – as done in Chambolle and Pock [2020] and Kobler et al. [2020] for Total Variation.

As for Synthesis, the optimization problem in Equation 1.44 is bi-convex with appropriate constraints $\mathcal{C}_A$. However, the regularization term $\|\Gamma^\top \hat{X}\|_1$ being non-differentiable, the minimization over Γ is generally done by sub-gradient descent [Yaghoobi et al., 2013], *i.e.*

$$X_{t+1} = \underset{\hat{X}}{\operatorname{argmin}} \frac{1}{2}\|\hat{X} - X\|_2^2 + \lambda\|\Gamma^\top \hat{X}\|_1 \tag{1.45}$$

$$\Gamma_{t+1} = \operatorname{proj}_{\mathcal{C}_A}\left(\Gamma_t - \tau_t u\right) \text{ s.t. } u \in \partial_{\|.^\top X_{t+1}\|}(\Gamma_t) \ . \tag{1.46}$$

Usage in inverse problems.

There are three main learning settings in inverse problems:

- Supervised.

  The practitioner has access to ground truth data and to a fixed operator A, which allows him to build a training data-set (Y, X) where $Y = AX + B$ is the observations, and to fit a model to predict X from Y. Then, this model can be applied to other data from the same distribution given that the measurement process does not vary.

- Self-supervised.

  The practitioner has access to ground truth or noisy data, but the operator A changes depending on the problem. For instance, this happens in deblurring where the shape or intensity of the blur A are not fixed and depend on the device or acquisition process. Thus, the model learned from the data has to be sufficiently modular to allow to integrate information on the measurement operator A that are not available during training.

- Unsupervised.

  The practitioner only has access to observations $Y = AX + B$, possibly acquired through different perspectives, i.e. with different operators A.

As demonstrated in Mairal et al. [2009], the most straightforward way to apply Dictionary Learning to inverse problems resolution is to learn the dictionary from ground truth or noisy data X_{train}, i.e. solving Equation 1.34 to obtain $\hat{D}$ or Equation 1.44 to obtain $\hat{\Gamma}$, and then to use this dictionary to find an estimate $\hat{X}$ from noisy observations $Y = AX_{\text{test}} + B$ with Synthesis

$$\hat{X} = \hat{D}\big(\underset{Z}{\operatorname{argmin}} \frac{1}{2}\|A\hat{D}Z - Y\|_2^2 + \lambda\|Z\|_1\big) \ , \tag{1.47}$$

or Analysis

$$\hat{X} = \underset{X}{\operatorname{argmin}} \frac{1}{2}\|AX - Y\|_2^2 + \lambda\|\hat{\Gamma}^\top X\|_1 \ . \tag{1.48}$$

This methodology solves the self-supervised setting, because the dictionary may be used with all kinds of operators, given that the data distribution does not change. It is also

possible to replace $\hat{D}$ and $\hat{\Gamma}$ by non linear operators learned from the data, as done in Oberlin and Verm [2021] with deep generative models.

Even though Dictionary Learning has mainly been studied and used in noiseless or noisy scenarios, a few works have demonstrated that it is possible to learn dictionaries from incomplete data in an unsupervised setting, especially in the context of missing values or inpainting in imaging [Szabó et al., 2011, Studer and Baraniuk, 2012, Naumova and Schnass, 2017], or in multiview compressive sensing [Anaraki and Hughes, 2013, Pourkamali-Anaraki et al., 2015, Chang et al., 2019]. The Dictionary Learning problem in Equation 1.34 becomes

$$\min_{Z\in\mathbb{R}^{L\times T}, D\in\mathcal{C}} \frac{1}{2}\|ADZ - Y\|_2^2 + \lambda\|Z\|_1 \tag{1.49}$$

with one measurement operator A, and

$$\min_{Z_i\in\mathbb{R}^{L\times T}, D\in\mathcal{C}} \sum_{i=1}^{N_m} \frac{1}{2}\|A_iDZ_i - Y_i\|_2^2 + \lambda\|Z_i\|_1 \tag{1.50}$$

with multiple measurement operators $(A_i)_{1\leq i\leq N_m}$. Another line of work studied online factorization of large matrices by aggregating partial information randomly selected from the data at each iteration [Mensch et al., 2016, 2017]. This is equivalent to learning a dictionary from incomplete data, except that one sample can be looked at multiple times from different angles, which is hardly possible in an inverse problem context.

One can also mention applications to Blind Source Separation where signals from different sources are mixed before measurement, and where sparse coding is useful to recover an appropriate factorization [Gribonval and Lesage, 2006].

1.3 . Learning to optimize with Unrolling

Optimization procedures are often cumbersome when the size of the data increases. This is particularly true in inverse problems where resolution methods are often based on iterative algorithms that need hundreds or thousands of iterations to converge and produce an accurate solution. Thus, with the success of automatic differentiation and Deep Learning, a new paradigm called *Learning to Optimize* [Chen et al., 2022] intends to develop new optimization methods trained with back-propagation to reduce this computational cost, and scale up to large data-sets. More concretely, this paradigm generally involves a model trained to solve an optimization problem or a surrogate problem on a training data set, supposedly more efficiently than standard iterative methods. Then, it can be applied as is or integrated in a more complex system to process new data without the need of classical optimization algorithms. In this work, we mainly deal with one framework based on Learning to Optimize, called *Unrolling* [Gregor and LeCun, 2010].

Unrolling

Many reconstruction and optimization methods are based on iterative algorithms. The idea of Unrolling was to unroll/unfold the iterations of these algorithms and learn the weights as if they were parameters of a neural network. As this work mainly focuses on sparse representations and optimization problems based on the ℓ_1 norm, we will present

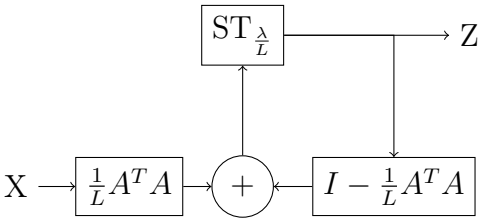

Figure 1.1: ISTA as a RNN

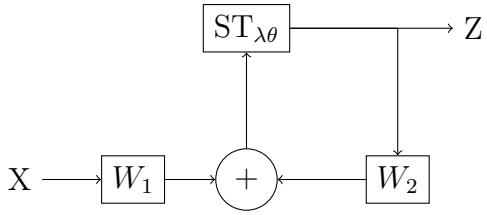

Figure 1.2: ISTA with parameters

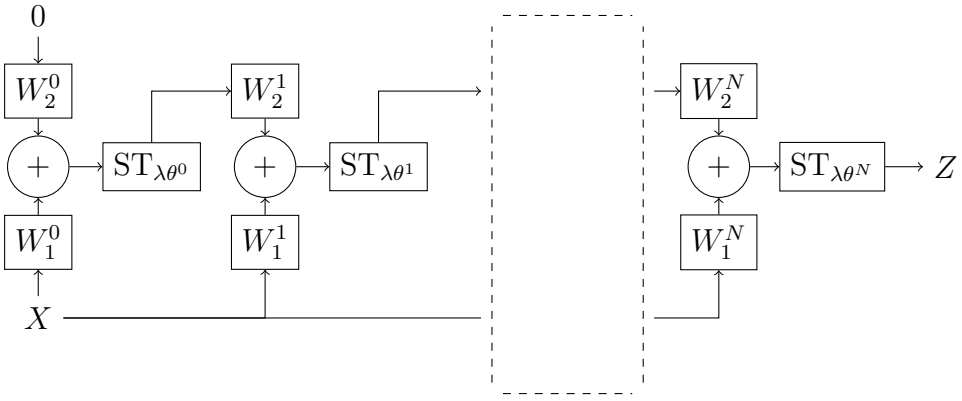

Figure 1.3: Learned ISTA (LISTA)

the concept of Unrolling with the example of the Lasso, and Learned ISTA [Gregor and LeCun, 2010].

Learned ISTA.

FISTA and coordinate descent are well-performing on small problems. However, they don't scale to larger ones. Thus, in order to improve the computation time and the convergence rate during the first iterations, an idea is to use Deep Learning to learn a better way to perform proximal gradient descent. The intuition is that the descent directions and steps may be fitted to a given data distribution.

This observation motivated the work of Gregor and LeCun [2010], in which the authors introduce Learned ISTA (LISTA). As a matter of fact, ISTA can be seen as a recurrent neural network, as represented in Figure 1.1. Denoting $W_1 = \frac{1}{L}A^TA$, $W_2 = I - \frac{1}{L}A^TA$ and $\theta = \frac{1}{L}$, it is possible to rewrite the network and see W_1, W_2, θ as parameters, resulting in the RNN in Figure 1.2.

Then, N iterations can be unfolded to obtain LISTA [Gregor and LeCun, 2010], as shown in Figure 1.3. In LISTA, the parameters $(W_1^i, W_2^i, \theta^i)_{1 \leq i \leq N}$ at iteration i are learned to minimize a loss, either in a supervised setting as in the original paper, or in an unsupervised setting to solve the Lasso without any ground truth as in Ablin et al. [2019]. The weights are generally initialized with ISTA, *i.e.* $W_1^i = \frac{1}{L}A^TA$, $W_2^i = I - \frac{1}{L}A^TA$, $\theta^i = \frac{1}{L}$, and optimized by gradient descent with the help of automatic differentiation.

Properties.

A large amount of works provide theoretical explanations on the behavior of LISTA, analyze its properties and derive strategies to learn with less parameters. The review in Chen et al. [2022] provides a complete overview of the field. As an example, Moreau and Bruna [2017] presents LISTA as a matrix factorization method whose goal is to improve the directions and the step sizes of the gradient descent. In Chen et al. [2018], the asymptotic properties of the weights are analyzed in order to detect a convergence pattern. As a matter of fact, $||W_1 - (I - W_2)||$ converges toward 0 when the layer position goes to infinity. Notice that, by definition, the relation $W_1 - (I - W_2) = 0$ is verified in ISTA. Therefore, the authors propose a new network based on that property where the parameters are coupled, meaning that $W_1^i - (I - W_2^i) = 0$ for each layer. This trick makes it possible to reduce the number of parameters in the network. In Liu and Chen [2019], the authors extend this idea and show that an analytic formula for the weights allows to learn only the steps sizes and the thresholds, while retaining the properties of LISTA.

Another interesting observation is that the thresholds are bigger at the beginning than at the end of the algorithm. Big thresholds make the zero coefficients converge faster. After a few iterations, the threshold level goes back to normal. The asymptotic behavior of the weights and the thresholds is explained in Ablin et al. [2019]. In particular, the authors prove the convergence of LISTA toward ISTA when the number of iterations becomes large. They also propose to learn only the step sizes in ISTA.

Usage for reconstruction and inverse problems.

As explained before, inverse problems are commonly solved by finding a solution to an optimization problem of the form

$$\min_{x \in \mathbb{R}^n} \frac{1}{2}\|y - Ax\|_2^2 + \lambda \mathcal{R}(x) \ . \tag{1.51}$$

This is generally achieved with the usage of iterative procedures, like forward backward splitting algorithms. Unrolling provides a way to fit the regularization to the data by learning the parameters of the algorithm. More precisely, given a dataset of N_d clean signals $(x_i)_{1 \leq i \leq N_d}$ and related measurements $(y_i)_{1 \leq i \leq N_d}$, a network $f_{N,\theta}$ built from N iterations of an iterative algorithm is trained to minimize

$$\mathbb{E}_{x,y}[\|f_{N,\theta}(y) - x\|^2] \tag{1.52}$$

over θ. This technique has been used to solve a broad range of inverse problems like computed tomography [Hammernik et al., 2017], image restoration [Lecouat et al., 2020, Bertocchi et al., 2020, Kobler et al., 2020, Jiu and Pustelnik, 2021], medical imaging [Kofler et al., 2020, Ramzi et al., 2020, Li et al., 2021] and phase retrieval [Vial et al., 2022].

Usage for dictionary learning.

Classical Dictionary Learning methods solve (2.1) through Alternating Minimization (AM) [Mairal et al., 2009, Peyré and Fadili, 2011]. It consists of minimizing the cost function over Z with a fixed dictionary D and then performing gradient descent to optimize the dictionary with a fixed Z. While AM provides a simple strategy to perform Dictionary Learning, it can be inefficient on large-scale data sets due to the need to precisely resolve the inner problems. Over the past years, many studies have proposed to use algorithm

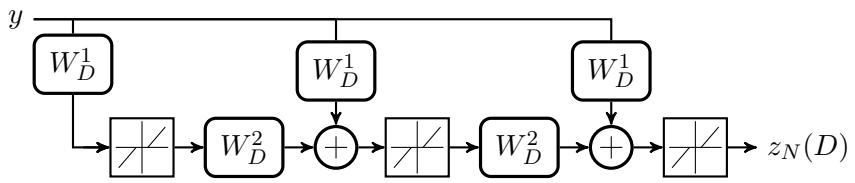

Figure 1.4: Illustration of LISTA for Dictionary Learning with initialization $Z_0 = 0$ for $N = 3$. $W_D^1 = \frac{1}{L}D^\top D$, $W_D^2 = (I - \frac{1}{L}D^\top D)$, where $L = \|D\|^2$. The result $Z_N(D)$ output by the network is an approximation of the solution of the LASSO.

unrolling, either for Analysis [Chambolle and Pock, 2020, Lecouat et al., 2020] or Synthesis [Tolooshams et al., 2020, Scetbon et al., 2021], to overcome that issue. For such unrolled algorithms, the weights W^1 and W^2 can be re-parameterized as functions of D – as illustrated in Figure 1.4 – such that the output $Z_N(D)$ of the network matches the result of N iterations of ISTA, *i.e.*

$$W_D^1 = \frac{1}{L}D^\top \quad \text{and} \quad W_D^2 = (I - \frac{1}{L}D^\top D), \quad \text{where} \quad L = \|D\|^2 \ . \tag{1.53}$$

Then, the dictionary can be learned by minimizing the loss over D with back-propagation. This approach is commonly referred to as *Deep Dictionary Learning* (DDL).

Variants of DDL with different kinds of regularization have been proposed in the literature. Scetbon et al. [2021] proposes to unroll sparse coding to learn a dictionary and the regularization hyperparameter from the data. In Tolooshams et al. [2020], the authors propose an unrolled algorithm based on expectation maximization to learn convolutional filters. Other works focus on image processing based on metric learning [Tang et al., 2022] and classification tasks with scattering [Zarka et al., 2019], among others.

Networks adapted to learn Analysis dictionaries have also been studied [Chambolle and Pock, 2020, Jiu and Pustelnik, 2021]. Indeed, the corresponding optimization problem can be solved with a primal-dual algorithm like Condat-Vu [Condat, 2013, Vu, 2013], which consists of a primal descent with step size τ and a dual ascent with step size σ. N iterations of Condat-Vu can be unrolled to obtain $x_N(\Gamma)$ on the same principle as LISTA and DDL in Synthesis [Jiu and Pustelnik, 2021]. In Lecouat et al. [2020], the authors leverage a smooth regularization of Analysis based on the Moreau envelope and use this as a basis to build their unrolled architecture and perform image reconstruction.

Contributions on Unrolling for Dictionary Learning and inverse problems

In the first part of this work, we focus on two questions related to the usage of unrolling for dictionary learning and inverse problems. The first one is: *does Unrolling make Dictionary Learning more efficient ?* As explained above, many authors have proposed to unroll sparse coding algorithms and learn dictionaries with back-propagation in order to improve either the solution or the convergence speed. In chapter 2, we analyze the efficiency and limitations of Unrolling in this context.

The second question we address is: *to what extent can unrolled Dictionary Learning be*

used to solve inverse problems ? In chapter 3, we provide an empirical study of prior learning and Unrolling in unsupervised inverse problems. Then, we study to what extent unrolled networks built from Analysis and Synthesis can be used as pre-trained denoisers in a self-supervised setting in chapter 4.

Chapter 2

Efficiency and limitations of Unrolling for Dictionary Learning

The content of this chapter was published in:

Benoît Malézieux, Thomas Moreau, and Matthieu Kowalski. Understanding approximate and unrolled dictionary learning for pattern recovery. *International Conference on Learning Representations*, 2022a

While Deep Dictionary Learning methods have proven to be efficient on a large amount of tasks in signal processing, the reasons why are still unclear.

From now on and for the sake of simplicity, we will focus on Synthesis and adopt the vector notation z for sparse codes Z. Dictionary Learning can be written as a bi-level optimization problem to minimize the cost function F of the Lasso with respect to the dictionary only, as mentioned in Mairal et al. [2009],

$$\min_{D \in \mathcal{C}} G(D) \triangleq F(z^*(D), D) \quad \text{with} \quad z^*(D) = \operatorname*{argmin}_{z \in \mathbb{R}^L} F(z, D) \ . \tag{2.1}$$

Computing the data representation $z^*(D)$ is often referred to as the inner problem, while the global minimization is the outer problem.

Bi-level optimization problems involving the Lasso have been studied in Bertrand et al. [2020] for hyper-parameter selection. The authors show that it is possible to find an optimal hyper-parameter λ for a given data-set by gradient descent. This involves solving the inner problem, *i.e.* the Lasso, and then computing the weak Jacobian of the outer loss over the hyper-parameter in order to get the value of the gradient. As the objective is smooth on the support of the solution, the loss becomes differentiable when the estimate is precise enough.

The quality of Jacobians and gradients rendered by automatic differentiation for smooth

bi-level problems have been studied in Ablin et al. [2020]. The idea is that instead of computing a costly exact solution $z^*(\theta)$ to optimize a loss function $f(\theta, z^*(\theta))$ with respect to θ, the estimate is replaced by an approximation $z_N(\theta)$ obtained through N iterations of a numerical optimization scheme. Then the gradient $\nabla_\theta f$ can be found either by automatic differentiation of $f(\theta, z_N(\theta))$ or by Alternating Minimization with an analytic formula where z_N acts as z^*. The authors show that the estimate with automatic differentiation converge twice as fast as the standard estimate with Alternating Minimization in the case of smooth loss.

Both papers have inspired the work of Tolooshams and Ba [2021], where the authors focus on unrolled Dictionary Learning and study its behavior when the support of $z^*(D)$ has been reached by $z_N(D)$, which makes the loss smooth.

However, computational limitations often prevent from reaching the support when unrolling sparse coding algorithms. As a matter of fact, back-propagation does not scale to hundreds or thousands of iterations, which are necessary to converge when starting from a random initialization, and support estimation can be difficult in a broad range of situations. Thus, this raises the question of the behavior of Unrolling out of the support, which we now address.

Contributions of chapter 2.

In this section, we study the efficiency and limitations of Unrolling for Dictionary Learning. As many works have focused on smooth losses [Bertrand et al., 2020, Ablin et al., 2020, Tolooshams and Ba, 2021], we study the instability of non-smooth bi-level optimization and unrolled sparse coding out of the support, which is of major interest in practice with a small number of layers. We provide details on Unrolling in Lasso-based Dictionary Learning in section 2.1, and we analyze the convergence of the Jacobian computed with automatic differentiation in section 2.2 to find out that its stability is guaranteed on the support of the sparse codes only. De facto, approximation errors in its estimation make Unrolling inefficient after a few dozen iterations. We also provide a lower bound of these errors for an ill-conditioned example and highlight that they can be arbitrarily large depending on the data and the optimization path. Finally, we give practical details on the usage of unrolled Dictionary Learning in section 2.3.

2.1 . Bi-level optimization and gradient estimation in Dictionary Learning

As previously mentioned, $z^*(D)$ does not have a closed-form expression, and G cannot be computed directly. A solution is to replace the inner problem $z^*(D)$ by an approximation $z_N(D)$ obtained through N iterations of a numerical optimization algorithm or its unrolled version. This reduces the problem to minimizing $G_N(D) \triangleq F(z_N(D), D)$. The first question is how sub-optimal global solutions of G_N are compared to the ones of G. Proposition 2.1.1 shows that the global minima of G_N converge as fast as the numerical approximation z_N in function value.

PROPOSITION 2.1.1. *Let $D^* = \operatorname{argmin}_D G(D)$ and $D_N^* = \operatorname{argmin}_D G_N(D)$, where N is the number of unrolled iterations. We denote by $K(D^*)$ a constant depending on D^*, and by $C(N)$ the convergence speed of the algorithm, which approximates the inner problem solution. We have*

$$G_N(D_N^*) - G(D^*) \leq K(D^*)C(N) \ .$$

Proof 2.1.1

Let $G(D) \triangleq F(z^*(D), D)$ and $G_N(D) \triangleq F(z_N(D), D)$ where $z^*(D) = \operatorname{argmin}_{z \in \mathbb{R}^L} F(z, D)$ and $z_N(D) = \text{FISTA}(D, N)$. Let $D^* = \operatorname{argmin}_D G(D)$ and $D_N^* = \operatorname{argmin}_D G_N(D)$. We have

$$\begin{aligned} G_N(D_N^*) - G(D^*) &= G_N(D_N^*) - G_N(D^*) + G_N(D^*) - G(D^*) && (2.2)\\ &= F(z_N(D_N), D_N) - F(z_N(D^*), D^*) && (2.3)\\ &\quad + F(z_N(D^*), D^*) - F(z(D^*), D^*) && (2.4) \end{aligned}$$

By definition of D_N^*

$$F(z_N(D_N^*), D_N^*) - F(z_N(D^*), D^*) \leq 0 \tag{2.5}$$

The convergence rate of FISTA in function value for a fixed dictionary D is

$$F(z_N(D), D) - F(z_N(D), D) \leq \frac{K(D)}{N^2} \tag{2.6}$$

where $K(D)$ is a constant depending on D. Therefore

$$F(z_N(D^*), D^*) - F(z(D^*), D^*) \leq \frac{K(D^*)}{N^2} \tag{2.7}$$

Hence

$$G_N(D_N^*) - G(D^*) \leq \frac{K(D^*)}{N^2} \tag{2.8}$$

Proposition 2.1.1 implies that when z_N is computed with FISTA [Beck and Teboulle, 2009], the function value for global minima of G_N converges with speed $C(N) = \frac{1}{N^2}$ toward the value of the global minima of F. Therefore, solving the inner problem approximately leads to suitable solutions for (2.1), given that the optimization procedure is efficient enough to find a proper minimum of G_N. As the computational cost of z_N increases with N, the choice of N results in a trade-off between the precision of the solution and the computational efficiency, which is critical for processing large data sets.

Moreover, learning the dictionary and computing the sparse codes are two different tasks. The loss G_N takes into account the dictionary and the corresponding approximation $z_N(D)$ to evaluate the quality of the solution. However, the dictionary evaluation should reflect its ability to generate the same signals as the ground truth data and not consider an approximate sparse code that can be recomputed afterward. Therefore, we should distinguish the ability of the algorithm to recover a good dictionary from its ability to learn the dictionary and the sparse codes simultaneously. In this work, we compare the atoms using their correlation and denote as C the cost matrix whose entry i, j compare the atom i of the first dictionary and j of the second. We define a sign and permutation invariant metric

$S(C) = \max_{\sigma \in \mathfrak{S}_L} \frac{1}{L} \sum_{i=1}^{L} |C_{\sigma(i),i}|$, where $\mathfrak{S}_L$ is the group of permutations of $[1, L]$. This metric corresponds to the best linear sum assignment on the cost matrix C, which can be computed with the Hungarian algorithm. Note that doing so has several limitations and that evaluating the dictionary is still an open problem. The practical study in section 2.3 provides details on the performance of unrolled Dictionary Learning with respect to this metric.

Dictionary Learning is a non-convex problem, meaning that good or poor local minima of G_N may be reached depending on the initialization, the optimization path, and the structure of the problem. Therefore, a gradient descent on G_N cannot guarantee an adequate minimizer of G. While complete theoretical analysis of these problems is arduous, we propose to study the correlation between the gradient obtained with G_N and the actual gradient of G as a way to ensure that the optimization dynamics are similar. Once $z^*(D)$ is known, Danskin [1967, Thm 1] states that $g^*(D) = \nabla G(D)$ is equal to $\nabla_2 F(z^*(D), D)$, where ∇_2 indicates that the gradient is computed relatively to the second variable in F. Even though the inner problem is non-smooth, this result holds as long as the solution $z^*(D)$ is unique. In the following, we will assume that $D^\top D$ is invertible on the support of $z^*(D)$, which implies the uniqueness of $z^*(D)$. This occurs with probability one if D is sampled from a continuous distribution [Tibshirani, 2013]. Alternating Minimization (AM) and Deep Dictionary Learning (DDL) differ in how they estimate the gradient of G. AM relies on the analytical formula of g^* and uses an approximation z_N of z^*, leading to the approximate gradient $g_N^1(D) = \nabla_2 F(z_N(D), D)$. We evaluate how well g_N^1 approximates g^* in Proposition 2.1.2.

PROPOSITION 2.1.2. *Let $D \in \mathbb{R}^{n \times L}$, and $z_0(D) = 0$. Then, there exists a constant $L > 0$, depending on D, such that for every number of iterations N*

$$\|g_N^1 - g^*\| \leq L \|z_N(D) - z^*(D)\| \ .$$

Proof 2.1.2

We have

$$F(z, D) = \frac{1}{2}\|Dz - y\|_2^2 + \lambda \|z\|_1 \tag{2.9}$$

$$\nabla_2 F(z, D) = (Dz - y)z^\top \tag{2.10}$$

$z_0(D) = 0$ and the iterates $(z_N(D))_{N \in \mathbb{N}}$ converge toward $z^*(D)$. Hence, they are contained in a closed ball around $z^*(D)$ that depends on D. As $\nabla_2 F(\cdot, D)$ is continuously differentiable, it is locally Lipschitz on this closed ball, and there exists a constant $L(D)$ depending on D such that

$$\|g_N^1 - g^*\| = \|\nabla_2 F(z_N(D), D) - \nabla_2 F(z^*(D), D)\| \tag{2.11}$$

$$\leq L(D) \|z_N(D) - z^*(D)\| \tag{2.12}$$

Proposition 2.1.2 shows that g_N^1 converges as fast as the iterates of ISTA converge. DDL computes the gradient automatically through $z_N(D)$. Unlike AM, this directly minimizes

the loss $G_N(D)$. Automatic differentiation yields a sub-gradient $g_N^2(D)$ such that

$$g_N^2(D) \in \nabla_2 F(z_N(D), D) + J_N^+\Big(\partial_1 F(z_N(D), D)\Big) \ , \tag{2.13}$$

where $J_N : \mathbb{R}^{n\times L} \to \mathbb{R}^L$ is the weak Jacobian of $z_N(D)$ with respect to D and J_N^+ denotes its adjoint. The product between J_N^+ and $\partial_1 F(z_N(D), D)$ is computed via automatic differentiation. As for AM, we characterize the convergence of g_N^2 toward g^* in Proposition 2.1.3.

Proposition 2.1.3. *Let $D \in \mathbb{R}^{n\times L}$. Let S^* be the support of $z^*(D)$, S_N be the support of z_N and $\widetilde{S}_N = S_N \cup S^*$. Let $f(z, D) = \frac{1}{2}\|Dz - y\|_2^2$. Let $R(J, \widetilde{S}) = J^+\big(\nabla_{1,1}^2 f(z^*, D) \odot \mathbb{1}_{\widetilde{S}}\big) + \nabla_{2,1}^2 f(z^*, D) \odot \mathbb{1}_{\widetilde{S}}$. Then there exists a constant $L > 0$ and a sub-sequence of (F)ISTA iterates $z_{\phi(N)}$ such that for all $N \in \mathbb{N}$:*

$$\exists\ g_{\phi(N)}^2 \in \nabla_2 f(z_{\phi(N)}, D) + J_{\phi(N)}^+\Big(\nabla_1 f(z_{\phi(N)}, D) + \lambda\partial_{\|\cdot\|_1}(z_{\phi(N)})\Big) \ s.t. :$$
$$\|g_{\phi(N)}^2 - g^*\| \leq \|R(J_{\phi(N)}, \widetilde{S}_{\phi(N)})\|\|z_{\phi(N)} - z^*\| + \frac{L}{2}\|z_{\phi(N)} - z^*\|^2 \ .$$

This sub-sequence $z_{\phi(N)}$ corresponds to iterates on the support of z^*.

Proof 2.1.3

We have

$$g_N^2(D) \in \nabla_2 f(z_N(D), D) + J_N^+\ \big(\nabla_1 f(z_N(D), D) + \lambda\partial_{\|\cdot\|_1}(z_N)\big) \tag{2.14}$$

We adapt equation (6) in Ablin et al. [2020]

$$g_N^2 = g^* + R(J_N, \widetilde{S_N})(z_N - z^*) + R_N^{D,z} + J_N^+ R_N^{z,z} \tag{2.15}$$

where

$$R(J, \widetilde{S}) = J^+\big(\nabla_{1,1}^2 f(z^*, D) \odot \mathbb{1}_{\widetilde{S}}\big) + \nabla_{2,1}^2 f(z^*, D) \odot \mathbb{1}_{\widetilde{S}} \tag{2.16}$$
$$R_N^{D,z} = \nabla_2 f(z_N, D) - \nabla_2 f(z^*, D) - \nabla_{2,1}^2 f(z^*, D)(z_N - z^*) \tag{2.17}$$
$$R_N^{z,z} \in \nabla_1 f(z_N, D) + \lambda\partial_{\|\cdot\|_1}(z_N) - \nabla_{1,1}^2 f(z^*, D)(z_N - z^*) \tag{2.18}$$

As z_N and z^* are on $\widetilde{S_N}$

$$\nabla_{2,1}^2 f(z^*, D)(z_N - z^*) = \Big(\nabla_{2,1}^2 f(z^*, D) \odot \mathbb{1}_{\widetilde{S_N}}\Big)(z_N - z^*) \tag{2.19}$$
$$J^+\big(\nabla_{1,1}^2 f(z^*, D)(z_N - z^*)\big) = J^+\Big(\nabla_{1,1}^2 f(z^*, D) \odot \mathbb{1}_{\widetilde{S_N}}(z_N - z^*)\Big) \tag{2.20}$$

As stated in Proposition 2.1.2, $\nabla_2 f(\cdot, D)$ is locally Lipschitz, and $R_N^{D,z}$ is the Taylor rest of $\nabla_2 f(\cdot, D)$. Therefore, there exists a constant $L_{D,z}$ such that

$$\forall N \in \mathbb{N}, \|R_N^{D,z}\| \leq \frac{L_{D,z}}{2}\|z_N(D) - z^*(D)\|^2 \tag{2.21}$$

We know that $0 \in \nabla_1 f(z^*, D) + \lambda\partial_{\|\cdot\|_1}(z^*)$. In other words, $\exists u^* \in \lambda\partial_{\|\cdot\|_1}(z^*)$ s.t. $\nabla_1 f(z^*, D) + u^* = 0$. Therefore we have:

$$R_N^{z,z} \in \nabla_1 f(z_N, D) - \nabla_1 f(z^*, D) - \nabla_{1,1}^2 f(z^*, x)(z_N - z^*) + \lambda\partial\|z_N\|_1 - u^* \tag{2.22}$$

Let $L_{z,z}$ be the Lipschitz constant of $\nabla_1 f(\cdot, D)$. (F)ISTA outputs a sequence such that there exists a sub-sequence $(z_{\phi(N)})_{N\in\mathbb{N}}$ which has the same support as z^*. For this sub-sequence, $u^* \in \lambda\partial_{\|\cdot\|_1}(z_{\phi(N)})$. Therefore, there exists $R_{\phi(N)}^{z,z}$ such that

1. $R^{z,z}_{\phi(N)} \in \nabla_1 f(z_{\phi(N)}, D) + \lambda \partial_{\|\cdot\|_1}(z_{\phi(N)}) - \nabla^2_{1,1} f(z^*, x)(z_{\phi(N)} - z^*)$

2. $\|R^{z,z}_{\phi(N)}\| \leq \frac{L_{z,z}}{2}\|z_{\phi(N)} - z^*\|^2$

For this sub-sequence, we can adapt Proposition 2 from Ablin et al. [2020]. Let $L = L_{D,z} + L_{z,z}$*, we have*

$$\exists\ g^2_{\phi(N)} \in \nabla_2 f(z_{\phi(N)}, D) + J_{\phi(N)}\big(\nabla_1 f(z_{\phi(N)}, D) + \lambda \partial \|z_{\phi(N)}\|_1\big),\ s.t.: \tag{2.23}$$

$$\|g^2_{\phi(N)} - g^*\| \leq \|R(J_{\phi(N)}, \widetilde{S_{\phi(N)}})\|\|z_{\phi(N)} - z^*\| + \frac{L}{2}\|z_{\phi(N)} - z^*\|^2 \tag{2.24}$$

Proposition 2.1.3 shows that g^2_N may converge faster than g^1_N once the support is reached. Ablin et al. [2020] and Tolooshams and Ba [2021] have studied the behavior of strongly convex functions, as it is the case on the support, and found similar results. This allowed Tolooshams and Ba [2021] to focus on support identification and show that automatic differentiation leads to a better gradient estimation in Dictionary Learning on the support under minor assumptions.

However, we are also interested in characterizing the behavior outside of the support. Besides the computation of the sub-differential – in practice, automatic differentiation uses the sign operator as a sub-gradient of the ℓ_1 norm – the convergence behavior of g^2_N is driven by $R(J_N, \widetilde{S_N})$ and thus by the weak Jacobian computed via back-propagation. Therefore, it is of high interest to study the behavior of this Jacobian. We first compute a closed-form expression of the weak Jacobian of $z^*(D)$ and $z_N(D)$. We then show that $R(J_N, \widetilde{S_N}) \leq L\|J_N - J^*\|$ and we analyze the convergence of J_N toward J^*.

2.2 . Study of the Jacobian.

The computation of the Jacobian can be done by differentiating through ISTA. In Theorem 2.2.1, we show that J_{N+1} depends on J_N and the past iterate z_N, and converges toward a fixed point. This formula can be used to compute the Jacobian during the forward pass, avoiding the computational cost of back-propagation and saving memory.

THEOREM 2.2.1. *At iteration $N+1$ of ISTA, the weak Jacobian of z_{N+1} relatively to D_l, where D_l is the l-th row of D, is given by induction:*

$$\frac{\partial(z_{N+1})}{\partial D_l} = \mathbb{1}_{|z_{N+1}|>0} \odot \left(\frac{\partial(z_N)}{\partial D_l} - \frac{1}{L}\left(D_l z_N^\top + (D_l^\top z_N - y_l) I_n + D^\top D \frac{\partial(z_N)}{\partial D_l}\right)\right) .$$

$\frac{\partial(z_N)}{\partial D_l}$ will be denoted by J^N_l. It converges toward the weak Jacobian J^*_l of z^* relatively to D_l, whose values are

$$J^*_{l\,S^*} = -(D^\top_{:,S^*} D_{:,S^*})^{-1}(D_l z^{*\top} + (D_l^\top z^* - y_l) I_n)_{S^*} ,$$

on the support S^* of z^*, and 0 elsewhere. Moreover, $R(J^*, S^*) = 0$.

Proof 2.2.1

We start by recalling a Lemma from Deledalle et al. [2014].

Lemma. *The soft-thresholding ST_μ defined by $ST_\mu(z) = sgn(z) \odot (|z| - \mu)_+$ is weakly differentiable with weak derivative $\frac{dST_\mu(z)}{dz} = \mathbb{1}_{|z|>\mu}$.*

Coordinate-wise, ISTA corresponds to the following equality:

$$z_{N+1} = ST_\mu((I - \frac{1}{L}D^\top D)z_N + \frac{1}{L}D^\top y) \tag{2.25}$$

$$(z_{N+1})_i = ST_\mu((z_N)_i - \frac{1}{L}\sum_{p=1}^{m}(\sum_{j=1}^{n} D_{ji}D_{jp})(z_N)_p + \frac{1}{L}\sum_{j=1}^{n} D_{ji}y_j) \tag{2.26}$$

The Jacobian is computed coordinate wise with the chain rule and the Lemma:

$$\frac{\partial(z_{N+1})_i}{\partial D_{lk}} = \mathbb{1}_{|(z_{N+1})_i|>0} \cdot (\frac{\partial(z_N)_i}{\partial D_{lk}} - \frac{1}{L}\frac{\partial}{\partial D_{lk}}(\sum_{p=1}^{m}(\sum_{j=1}^{n} D_{ji}D_{jp})(z_N)_p) + \frac{1}{L}\frac{\partial}{\partial D_{lk}}\sum_{j=1}^{n} D_{ji}y_j)) \tag{2.27}$$

Last term:

$$\frac{\partial}{\partial D_{lk}}\sum_{j=1}^{n} D_{ji}y_j = \delta_{ik}y_l \tag{2.28}$$

Second term:

$$\frac{\partial}{\partial D_{lk}}\sum_{p=1}^{m}\sum_{j=1}^{n} D_{ji}D_{jp}(z_N)_p = \sum_{p=1}^{m}\sum_{j=1}^{n} D_{ji}D_{jp}\frac{\partial(z_N)_p}{\partial D_{lk}} \; + \; \sum_{p=1}^{m}\sum_{j=1}^{n}\frac{\partial D_{ji}D_{jp}}{\partial D_{lk}}(z_N)_p \tag{2.29}$$

$$\frac{\partial D_{ji}D_{jp}}{\partial D_{lk}} = \begin{cases} 2D_{lk} & \text{if } j = l \text{ and } i = p = k \\ D_{lp} & \text{if } j = l \text{ and } i = k \text{ and } p \neq k \\ D_{li} & \text{if } j = l \text{ and } i \neq k \text{ and } p = k \\ 0 & \text{else} \end{cases} \tag{2.30}$$

Therefore:

$$\sum_{p=1}^{m}\sum_{j=1}^{n}\frac{\partial D_{ji}D_{jp}}{\partial D_{lk}}(z_N)_p = \sum_{p=1}^{m}(2D_{lk}\delta_{ip}\delta_{ik} + D_{li}\delta_{pk}\mathbb{1}_{i\neq k} + D_{lp}\delta_{ik}\mathbb{1}_{k\neq p})(z_N)_p \tag{2.31}$$

$$= 2D_{lk}(z_N)_k\delta_{ik} + D_{li}(z_N)_k\mathbb{1}_{i\neq k} + \sum_{\substack{p=1 \\ p\neq k}}^{m} D_{lp}(z_N)_p\delta_{ik} \tag{2.32}$$

$$= D_{li}(z_N)_k + \delta_{ik}\sum_{p=1}^{m} D_{l_p}(z_N)_p \tag{2.33}$$

Hence:

$$\frac{\partial(z_{N+1})_i}{\partial D_{lk}} = \mathbb{1}_{|(z_{N+1})_i|>0} \cdot \Big(\frac{\partial(z_N)_i}{\partial D_{lk}} - \frac{1}{L}(D_{li}(z_N)_k + \tag{2.34}$$
$$\delta_{ik}(\sum_{p=1}^{m} D_{lp}(z_N)_p) + \sum_{p=1}^{m}\sum_{j=1}^{n}\frac{\partial(z_N)_p}{\partial D_{lk}}D_{ji}D_{jp} - \delta_{ik}y_l)\Big)$$

This leads to the following vector formulation:

$$\frac{\partial(z_{N+1})}{\partial D_l} = \mathbb{1}_{|z_{N+1}|>0} \; \odot \; \left(\frac{\partial(z_N)}{\partial D_l} - \frac{1}{L}\left(D_l z_N^\top + (D_l^\top z_N - y_l)I_m + D^\top D\frac{\partial(z_N)}{\partial D_l}\right)\right) \tag{2.35}$$

On the support of z^*, denoted by S^*, this quantity converges toward the fixed point:

$$J_l^* = -(D_{:,S^*}^\top D_{:,S^*})^{-1}(D_l z^{*\top} + (D_l^\top z^* - y_l) I_m)_{S^*} \tag{2.36}$$

Elsewhere, J_l^* is equal to 0. To prove that $R(J^*, S^*) = 0$, we use the expression given by (2.35)

$$J^* = \mathbb{1}_{S^*} \odot \left(J^* - \frac{1}{L} \left(\nabla^2_{2,1} f(z^*, D_l)^\top + \nabla^2_{1,1} f(z^*, D)^\top J^* \right) \right) \tag{2.37}$$

$$J^* - \mathbb{1}_{S^*} \odot J^* = \frac{1}{L} \mathbb{1}_{S^*} \odot \nabla^2_{2,1} f(z^*, D_l)^\top + \mathbb{1}_{S^*} \odot \nabla^2_{1,1} f(z^*, D)^\top J^* \tag{2.38}$$

$$0 = J^{*+} \left(\nabla^2_{1,1} f(z^*, D) \odot \mathbb{1}_{S^*} \right) + \nabla^2_{2,1} f(z^*, D) \odot \mathbb{1}_{S^*} \tag{2.39}$$

$$0 = R(J^*, S^*) \tag{2.40}$$

This result is similar to Bertrand et al. [2020] where the Jacobian of z is computed over λ to perform hyper-parameter optimization in Lasso-type models. Using $R(J^*, S^*) = 0$, we can write

$$\|R(J_N, \widetilde{S}_N)\| \leq \|R(J_N, \widetilde{S}_N) - R(J^*, S^*)\| \leq L \|J_N - J^*\| \ , \tag{2.41}$$

as $\|\nabla^2_{1,1} f(z^*, D)\|_2 = L$. If the back-propagation were to output an accurate estimate J_N of the weak Jacobian J^*, $\|R(J_N, \widetilde{S_N})\|$ would be 0, and the convergence rate of g_N^2 could be twice as fast as the one of g_N^1.

Convergence analysis

We now analyze the convergence of J_N toward J^* to quantify this. In Proposition 2.2.2, we compute an upper bound of $\|J_l^N - J_l^*\|$ with possible usage of truncated back-propagation [Shaban et al., 2019]. Truncated back-propagation of depth K corresponds to an initial estimate of the Jacobian $J_{N-K} = 0$ and iterates the induction in Theorem 2.2.1.

PROPOSITION 2.2.2. *Let N be the number of iterations and K be the back-propagation depth. We assume that $\forall n \geq N - K$, $S^* \subset S_n$. Let $\bar{E}_N = S_n \setminus S^*$, let L be the largest eigenvalue of $D_{:,S^*}^\top D_{:,S^*}$, and let μ_n be the smallest eigenvalue of $D_{:,S_n}^\top D_{:,S_{n-1}}$. Let $B_n = \|P_{\bar{E}_n} - D_{:,\bar{E}_n}^\top D_{:,S^*}^{\dagger\top} P_{S^*}\|$, where P_S is the projection on $\mathbb{R}^S$ and $D^\dagger$ is the pseudo-inverse of D. We have*

$$\|J_l^N - J_l^*\| \leq \prod_{k=1}^{K} \left(1 - \frac{\mu_{N-k}}{L}\right) \|J_l^*\| + \frac{2}{L} \|D_l\| \sum_{k=0}^{K-1} \prod_{i=1}^{k} (1 - \frac{\mu_{N-i}}{L}) \left(\|z_l^{N-k} - z_l^*\| + B_{N-k} \|z_l^*\| \right) \ .$$

Proof 2.2.2

We denote by G the matrix $(I - \frac{1}{L} D^\top D)$. For z_N with support S_N and $z*$ with support S^*, we have with the induction in Theorem 2.2.1

$$J_{l,S_N}^N = \left(G J_l^{N-1} + u_l^{N-1} \right)_{S_N} \tag{2.42}$$

$$J_{l,S^*}^* = \left(G J_l^* + u_l^* \right)_{S^*} \tag{2.43}$$

where $u_l^N = -\frac{1}{L} \left(D_l z_N^\top + (D_l^\top z_N - y_l) I \right)$ and the other terms on $\bar{S}_N$ and $\bar{S}^*$ are 0.
We can thus decompose their difference as the sum of two terms, one on the support S^*

and one on this complement $\bar{E}_N = S_N \setminus S^*$

$$J_l^* - J_l^N = (J_l^* - J_l^N)_{S^*} + (J_l^* - J_l^N)_{\bar{E}_N} \ .$$

Recall that we assume $S^* \subset S_N$*. Let's study the terms separately on* S^* *and* $\bar{E}_N = S_N \setminus S^*$*. These two terms can be decompose again to constitute a double recursion system,*

$$\begin{aligned}(J_l^N - J_l^*)_{S^*} &= G_{S^*}(J_l^{N-1} - J_l^*) + (u_l^{N-1} - u_l^*)_{S^*} && (2.44)\\ &= G_{S^*,S^*}(J_l^{N-1} - J_l^*)_{S^*} + G_{S^*,\bar{E}_{N-1}}(J_l^{N-1} - J^*)_{\bar{E}_{N-1}} + (u_l^{N-1} - u_l^*)_{S^*} \ , && (2.45)\end{aligned}$$

$$\begin{aligned}(J_l^N - J_l^*)_{\bar{E}_N} &= (J_l^N)_{\bar{E}_N} = G_{\bar{E}_N}(J_l^{N-1} - J_l^*) + G_{\bar{E}_N,S^*}J_l^* + (u_l^{N-1})_{\bar{E}_N} && (2.46)\\ &= G_{\bar{E}_N,S^*}(J_l^{N-1} - J_l^*)_{S^*} + G_{\bar{E}_N,\bar{E}_{N-1}}(J_l^{N-1} - J_l^*)_{\bar{E}_{N-1}} && (2.47)\\ &\quad + (u_l^{N-1} - u_l^*)_{\bar{E}_N} + \Big((u_l^*)_{\bar{E}_N} - D^\top_{:,\bar{E}_N}D_{:,S^*}(D^\top_{:,S^*}D_{:,S^*})^{-1}(u_l^*)_{S^*}\Big) \ .\end{aligned}$$

We define as $\mathcal{P}_{S_N,\bar{E}_N}$ *the operator which projects a vector from* $\bar{E}_N$ *on* $(S_N, \bar{E}_N)$ *with zeros on* S_N*. As* $S^* \cup \bar{E}_N = S_N$*, we get by combining these two expressions,*

$$\begin{aligned}(J_l^N - J_l^*)_{S_N} =& G_{S_N,S_{N-1}}(J_l^{N-1} - J_l^*)_{S_{N-1}} + (u_l^{N-1} - u_l^*)_{S_N} && (2.48)\\ &+ \mathcal{P}_{S_N,\bar{E}_N}\Big((u_l^*)_{\bar{E}_N} - D^\top_{:,\bar{E}_N}D_{:,S^*}(D^\top_{:,S^*}D_{:,S^*})^{-1}(u_l^*)_{S^*}\Big)\end{aligned}$$

Taking the norm yields to the following inequality,

$$\begin{aligned}\|J_l^N - J_l^*\| \leq& \|G_{S_N,S_{N-1}}\|\|J_l^{N-1} - J_l^*\| + \|u_l^{N-1} - u_l^*\| && (2.49)\\ &+ \|(u_l^*)_{\bar{E}_N} - D^\top_{:,\bar{E}_N}D_{:,S^*}(D^\top_{:,S^*}D_{:,S^*})^{-1}(u_l^*)_{S^*}\| \ .\end{aligned}$$

Denoting by μ_N *the smallest eigenvalue of* $D^\top_{:,S_N}D_{:,S_{N-1}}$*, then* $\|G_{S_N,S_{N-1}}\| = (1 - \frac{\mu_N}{L})$ *and we get that*

$$\begin{aligned}\|J_l^N - J_l^*\| \leq& \prod_{k=1}^{K}(1 - \frac{\mu_{N-k}}{L})\|J_l^{N-K} - J_l^*\| && (2.50)\\ &+ \sum_{k=0}^{K-1}\prod_{i=1}^{k}(1 - \frac{\mu_{N-i}}{L})\Big(\|u_l^{N-k} - u_l^*\| + \|(u_l^*)_{\bar{E}_{N-k}} - D^\top_{:,\bar{E}_{N-k}}D^{\dagger\top}_{:,S^*}(u_l^*)_{S^*}\|\Big) \ .\end{aligned}$$

The back-propagation is initialized as $J_l^{N-K} = 0$*. Therefore* $\|J_l^{N-K} - J_l^*\| = \|J_l^*\|$*. Moreover* $\|u_l^{N-k} - u_l^*\| \leq \frac{2}{L}\|D_l\|\|z_l^{N-k} - z_l^*\|$*. Finally,* $\|(u_l^*)_{\bar{E}_{N-k}} - D^\top_{:,\bar{E}_{N-k}}D^{\dagger\top}_{:,S^*}(u_l^*)_{S^*}\|$ *can be rewritten with projection matrices* $P_{\bar{E}_{N-k}}$ *and* $P_{\bar{S}^*}$ *to obtain*

$$\begin{aligned}\|(u_l^*)_{\bar{E}_{N-k}} - D^\top_{:,\bar{E}_{N-k}}D^{\dagger\top}_{:,S^*}(u_l^*)_{S^*}\| \leq& \|P_{\bar{E}_{N-k}}u_l^* - D^\top_{:,\bar{E}_{N-k}}D^{\dagger\top}_{:,S^*}P_{S^*}u_l^*\| && (2.51)\\ \leq& \|P_{\bar{E}_{N-k}} - D^\top_{:,\bar{E}_{N-k}}D^{\dagger\top}_{:,S^*}P_{S^*}\|\|u_l^*\| && (2.52)\\ \leq& \|P_{\bar{E}_{N-k}} - D^\top_{:,\bar{E}_{N-k}}D^{\dagger\top}_{:,S^*}P_{S^*}\|\frac{2}{L}\|D_l\|\|z_l^*\| \ . && (2.53)\end{aligned}$$

Let $B_{N-k} = \|P_{\bar{E}_{N-k}} - D^\top_{:,\bar{E}_{N-k}}D^{\dagger\top}_{:,S^*}P_{S^*}\|$*. We have*

$$\|J_l^N - J_l^*\| \leq \prod_{k=1}^{K}(1 - \frac{\mu_{N-k}}{L})\|J_l^*\| + \frac{2}{L}\|D_l\|\sum_{k=0}^{K-1}\prod_{i=1}^{k}(1 - \frac{\mu_{N-i}}{L})\Big(\|z_l^{N-k} - z_l^*\| + B_{N-k}\|z_l^*\|\Big) \ . \quad (2.54)$$

We now suppose that the support is reached at iteration $N-s$, with $s \geq K$. Therefore, $\forall n \in [N-s, N]$ $S_n = S^*$. Let $\Delta_n = F(z_n, D) - F(z^*, D) + \frac{L}{2}\|z_n - z^*\|$. On the support, F is a μ-strongly convex function and the convergence rate of (z_N) is

$$\|z^* - z_N\| \leq \big(1 - \frac{\mu}{L}\big)^s \frac{2\Delta_{N-s}}{L} \tag{2.55}$$

Thus, we obtain

$$\begin{aligned}
\|J_l^N - J_l^*\| &\leq \prod_{k=1}^{K}(1 - \frac{\mu_{N-k}}{L})\|J_l^*\| && (2.56)\\
&\quad + \frac{2}{L}\|D_l\| \sum_{k=0}^{K-1}\prod_{i=1}^{k}(1 - \frac{\mu_{N-i}}{L})\Big(\|z_l^{N-k} - z_l^*\| + B_{N-k}\|u_l^*\|\Big)\\
&\leq \prod_{k=1}^{K}(1 - \frac{\mu_{N-k}}{L})\|J_l^*\| && (2.57)\\
&\quad + \frac{2}{L}\|D_l\| \sum_{k=0}^{s-1}(1 - \frac{\mu}{L})^k\Big(\|z_l^{N-k} - z_l^*\|\Big)\\
&\quad + \frac{2}{L}\|D_l\|(1 - \frac{\mu}{L})^s \sum_{k=s-1}^{K-1}\prod_{i=s-1}^{k}(1 - \frac{\mu_{N-i}}{L})\Big(\|z_l^{N-k} - z_l^*\| + B_{N-k}\|(u_l^*)\|\Big)\\
&\leq \prod_{k=1}^{K}(1 - \frac{\mu_{N-k}}{L})\|J_l^*\| && (2.58)\\
&\quad + \frac{2}{L}\|D_l\| \sum_{k=0}^{s-1}(1 - \frac{\mu}{L})^k\big(1 - \frac{\mu}{L}\big)^{s-1-k}\frac{2\Delta_{N-s}}{L}\\
&\quad + \frac{2}{L}\|D_l\|(1 - \frac{\mu}{L})^s \sum_{k=s-1}^{K-1}\prod_{i=s-1}^{k}(1 - \frac{\mu_{N-i}}{L})\Big(\|z_l^{N-k} - z_l^*\| + B_{N-k}\|(u_l^*)\|\Big)\\
&\leq \prod_{k=1}^{K}(1 - \frac{\mu_{N-k}}{L})\|J_l^*\| && (2.59)\\
&\quad + \|D_l\|(1 - \frac{\mu}{L})^{s-1} s\frac{4\Delta_{N-s}}{L^2}\\
&\quad + \frac{2}{L}\|D_l\|(1 - \frac{\mu}{L})^s \sum_{k=s-1}^{K-1}\prod_{i=s-1}^{k}(1 - \frac{\mu_{N-i}}{L})\Big(\|z_l^{N-k} - z_l^*\| + B_{N-k}\|(u_l^*)\|\Big)\\
& && (2.60)
\end{aligned}$$

Proposition 2.2.2 shows that the error $\|J_N - J^*\|$ is upper bounded by a sum of two terms $C_N^1 + C_N^2$, where C_N^1 converges toward 0 and C_N^2 increases before the iterates reach the support. This upper bound reveals multiple stages in the Jacobian estimation. First, one can see that if all iterates used for the back-propagation lie on the support S^*, the Jacobian estimate has a quasi-linear convergence, as shown in Corollary 2.2.3.

COROLLARY 2.2.3. *Let $\mu > 0$ be the smallest eigenvalue of $D_{:,S^*}^\top D_{:,S^*}$. Let $K \leq N$ be the back-propagation depth and let $\Delta_N = F(z_N, D) - F(z^*, D) + \frac{L}{2}\|z_N - z^*\|$. Suppose that $\forall n \in [N-K, N]$; $S_n \subset S^*$. Then, we have*

$$\|J_l^* - J_l^N\| \leq \left(1 - \frac{\mu}{L}\right)^K \|J_l^*\| + K\left(1 - \frac{\mu}{L}\right)^{K-1} \|D_l\| \frac{4\Delta_{N-K}}{L^2} \quad .$$

Proof 2.2.3

The term $\frac{2}{L}\|D_l\|(1-\frac{\mu}{L})^s \sum_{k=s-1}^{K-1} \prod_{i=s-1}^{k}(1-\frac{\mu_{N-i}}{L})\Big(\|z_l^{N-k} - z_l^*\| + B_{N-k}\|(u_l^*)\|\Big)$ vanishes when the algorithm is initialized on the support. Otherwise, it goes to 0 as $s, K \to N$ and $N \to \infty$ because $\forall n > N - s,\ \mu_n = \mu < 1$.

Once the support is reached, ISTA also converges with the same linear rate $(1 - \frac{\mu}{L})$. Thus the gradient estimate g_N^2 converges almost twice as fast as g_N^1 in the best case – with optimal sub-gradient – as $\mathcal{O}(K(1-\frac{\mu}{L})^{2K})$. This is similar to Ablin et al. [2020, Proposition.5] and Tolooshams and Ba [2021].

Second, Proposition 2.2.2 shows that $\|J_l^* - J_l^N\|$ may increase when the support is not well-estimated, leading to a deterioration of the gradient estimate. This is due to an accumulation of errors materialized by the sum in the right-hand side of the inequality, as the term $B_N\|z^*\|$ may not vanish to 0 as long as $S_N \not\subset S^*$. Interestingly, once the support is reached at iteration $S < N$, the errors converge linearly toward 0, and we recover the fast estimation of g^* with g^2. This result suggests that unrolling too many iterations has a negative impact on the gradient estimation. Therefore, DDL should either be used with a low number of steps or truncated back-propagation to ensure stability.

Lower bound for an ill-conditioned example

The upper bound of $\|J_N - J^*\|$ we provided is helpful to highlight the ambiguous behavior of the Jacobian estimate before the support in the general case. However, this behavior may change depending on the dictionary and the data. In the following, we compute a lower bound of $\|J_N - J^*\|$ in a simple case where the iterates $(z^N)_{N\in\mathbb{N}}$ reach arbitrarily slowly the support of z^*, and demonstrate that the lower bound is representative of what happens in practice in certain cases.

First, we define a dictionary D of dimension 2×3 with the following triplet of correlated atoms

$$d_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \qquad d_2 = \begin{bmatrix} 1-\epsilon \\ \sqrt{\epsilon(2-\epsilon)} \end{bmatrix} \qquad d_3 = \begin{bmatrix} 1-\epsilon \\ -\sqrt{\epsilon(2-\epsilon)} \end{bmatrix}$$

and denote $D = [d_1, d_2, d_3]$. The gram matrix $A = D^\top D$ for this dictionary is

$$A = \begin{bmatrix} 1 & 1-\epsilon & 1-\epsilon \\ 1-\epsilon & 1 & 1-4\epsilon+2\epsilon^2 \\ 1-\epsilon & 1-4\epsilon+2\epsilon^2 & 1 \end{bmatrix} \quad .$$

This matrix is rank 2, with eigenvalues $L = \lambda_1 = 3 - 4\epsilon + 2\epsilon^2$, $\lambda_2 = 2(2\epsilon - \epsilon^2)$ and $\lambda_3 = 0$

and associated eigenvectors

$$v_1 = \begin{bmatrix} \frac{1}{1-\epsilon} \\ 1 \\ 1 \end{bmatrix} \quad , \qquad v_2 = \begin{bmatrix} 0 \\ -1 \\ 1 \end{bmatrix} \qquad \text{and} \quad v_3 = \begin{bmatrix} -2(1-\epsilon) \\ 1 \\ 1 \end{bmatrix} \quad .$$

We consider the input point $y = d_1$. Then KKT conditions lead to $z^* = \begin{bmatrix} 1-\lambda \\ 0 \\ 0 \end{bmatrix}$. If we suppose that epsilon is small enough, we have

$$z^0 = \frac{D^\top y}{L} - \frac{\lambda}{L} = \frac{1}{L} \begin{bmatrix} 1-\lambda \\ 1-\lambda-\epsilon \\ 1-\lambda-\epsilon \end{bmatrix} = \alpha_1 v_1 + \alpha_3 v_3 \quad ,$$

with $\alpha_1 = \frac{1}{L\|v_1\|_2^2}(\frac{1-\lambda}{1-\epsilon} + 2(1-\lambda-\epsilon))$ and $\alpha_3 = \frac{1}{L\|v_3\|_2^2}(-2(1-\lambda)(1-\epsilon) + 2(1-\lambda-\epsilon)) = \frac{-2\epsilon(2-\lambda)}{L\|v_3\|_2^2}$ Here ϵ needs to be smaller than $1-\lambda$, so the soft thresholding is simply the operation $x \to x - \lambda$ as all coordinates are positive. The dynamic of the forward operation is the one of ISTA. As long as no coordinates of z^{t+1} becomes negative, we have:

$$z^{t+1} = (Id - \frac{A}{L})z^t + \underbrace{\frac{D^\top y}{L} - \frac{\lambda}{L}}_{z_0} \quad .$$

We denote $z^t = \alpha_1^t v_1 + \alpha_2^t v_2 + \alpha_3^t v_3$. Decomposing the dynamic in the eigenspaces of A, we get

$$\alpha_1^t = \alpha_1 \quad , \quad \alpha_2^t = 0 \quad \text{and} \quad \alpha_3^t = \alpha_3^{t-1} + \alpha_3 = (t+1)\alpha_3 \tag{2.61}$$

Note that this dynamic is true until one coefficient of z^t becomes negative. With this dynamic, the first coordinate of z^t is $\frac{\alpha_1}{1-\epsilon} - 2(t+1)(1-\epsilon)\alpha_3$. This is always positive, as α_3 is negative. For the coordinate 2 and 3 of z^t, we have $c^t = \alpha_1 + (t+1)\alpha_3$. This is positive as long as

$$t < \frac{-\alpha_1}{\alpha_3} - 1 = \frac{\|v_3\|}{\|v_1\|} \frac{2(1-\lambda-\epsilon) + \frac{1-\lambda}{1-\epsilon}}{2\epsilon(2-\lambda)} - 1 = \mathcal{O}(\frac{1}{\epsilon})$$

Thus, we have a simple dynamic out of the support for a very long amount of time. This will cause an error accumulation for the Jacobian, as shown in Proposition 2.2.4.

PROPOSITION 2.2.4. *Let $D \in \mathbb{R}^{2\times 3}$ be defined as $[d_1, d_2, d_3]$. Let $y = d_1$, $\epsilon < 1$ and $\lambda < 1$. Then a lower bound of $\|J_t - J^*\|_2$ before reaching the support is*

$$\frac{1}{\|v_3\|_2^2} |4(1-\epsilon)^2(2\lambda - 1) + \frac{t}{L}(\frac{\alpha_1}{1-\epsilon} + 2(1-\epsilon)\alpha_1 - 1)| \leq \|J_t - J^*\|_2 \quad .$$

The error increases linearly with the number of iterations outside of the support.

Proof 2.2.4

Given that z^t has strictly positive coefficients when t is small enough, the induction on the Jacobian reduces to

$$J_l^{t+1} = (I - \frac{1}{L}A)J_l^t - \frac{1}{L}(D_l z^{t\top} + (D_l^\top z^t - y_l)I) \tag{2.62}$$

The limit is

$$J_l^* = -(D_{:,S^*}^\top D_{:,S^*})^{-1}(D_l z^{*\top} + (D_l^\top z^* - y_l)I_m)_{S^*} \tag{2.63}$$

on the support S^* and 0 otherwise.
Let's consider the case $l = 1$ (first row). We denote $P = (v_1, v_2, v_3)$ and $\Lambda = \mathrm{diag}(\lambda_1, \lambda_2, \lambda_3)$. During the first iterations, when t is small enough, we have

$$J_1^{t+1} = (I - \frac{1}{L}A)J_1^t - \frac{1}{L}(D_1 z^{t\top} + (D_1^\top z^t - y_1)I) \tag{2.64}$$

$$= (I - \frac{1}{L}A)J_1^t - \frac{1}{L}M_t \tag{2.65}$$

$$P^{-1}J_1^{t+1}P = \Lambda P^{-1}J_1^t P - \frac{1}{L}P^{-1}M_t P \tag{2.66}$$

Thus

$$J_1^0 = 0 \tag{2.67}$$

$$J_1^T = -\frac{1}{L}\sum_{t=0}^{T-1}(I - \frac{1}{L}A)^{T-1-t}M_t \quad T \geq 1 \tag{2.68}$$

$$= -\frac{1}{L}\sum_{t=0}^{T-1} P\Lambda^{T-1-t}P^{-1}M_t \tag{2.69}$$

$$P^{-1}J_1^T P = -\frac{1}{L}\sum_{t=0}^{T-1}\Lambda^{T-1-t}P^{-1}M_t P \tag{2.70}$$

with

$$P = \begin{bmatrix} \frac{1}{1-\epsilon} & 0 & -2(1-\epsilon) \\ 1 & -1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{2.71}$$

$$P^{-1} = diag(C_1, C_2, C_3) \times P^\top = \begin{bmatrix} \frac{C_1}{1-\epsilon} & C_1 & C_1 \\ 0 & -C_2 & C_2 \\ -2(1-\epsilon)C_3 & C_3 & C_3 \end{bmatrix} \tag{2.72}$$

$$\Lambda = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 - \frac{2\epsilon(2-\epsilon)}{L} & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.73}$$

$$\tag{2.74}$$

where $C_1 = \frac{1}{\|v_1\|_2^2}$, $C_2 = \frac{1}{\|v_2\|_2^2}$, $C_3 = \frac{1}{\|v_3\|_2^2}$. When t is big enough, the dynamic changes and comes back to what was studied before. Thus, after convergence, the limit of the Jacobian is

$$J_1^* = \begin{bmatrix} 2\lambda - 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{2.75}$$

$$P^{-1}J_1^* P = diag(C_1, C_2, C_3) \times \begin{bmatrix} \frac{2\lambda-1}{1-\epsilon} & 0 & -2(1-\epsilon)(2\lambda-1) \\ 0 & 0 & 0 \\ -2(2\lambda-1) & 0 & 4(1-\epsilon)^2(2\lambda-1) \end{bmatrix} \tag{2.76}$$

$$\tag{2.77}$$

Let's compute $P^{-1}M_tP$. Recall that $z_2 = z_3$.

$$P^{-1}M_tP = diag(C_1, C_2, C_3)(P^\top D_1 {z^t}^\top P + (D_1^\top z^t - y_1)I) \tag{2.78}$$

$$= C \begin{bmatrix} \frac{2-\epsilon}{1-\epsilon}(z_1 + 2(1-\epsilon)z_2) - 1 & 0 & 2(z_2 - (1-\epsilon)z_1) \\ 0 & z_1 + 2(1-\epsilon)z_2 - 1 & 0 \\ 0 & 0 & z_1 + 2(1-\epsilon)z_2 - 1 \end{bmatrix} \tag{2.79}$$

$$\tag{2.80}$$

When $t < (T-1)$, multiplying by Λ^{T-1-t} leads to

$$\Lambda^{T-1-t}P^{-1}M_tP = \begin{bmatrix} 0 & 0 & 0 \\ 0 & C_2(1 - \frac{2\epsilon(2-\epsilon)}{L})^{T-1-t}(z_1 + 2(1-\epsilon)z_2 - 1) & 0 \\ 0 & 0 & C_3(z_1 + 2(1-\epsilon)z_2 - 1) \end{bmatrix} \tag{2.81}$$

$$= \begin{bmatrix} 0 & 0 & 0 \\ 0 & C_2(1 - \frac{2\epsilon(2-\epsilon)}{L})^{T-1-t}(\frac{\alpha_1}{1-\epsilon} + 2(1-\epsilon)\alpha_1 - 1) & 0 \\ 0 & 0 & C_3(\frac{\alpha_1}{1-\epsilon} + 2(1-\epsilon)\alpha_1 - 1) \end{bmatrix} \tag{2.82}$$

$$\tag{2.83}$$

Looking at the dynamic of the bottom right corner coordinate gives a lower bound of $\|J_t - J^*\|$, as we have

$$|C_3||4(1-\epsilon)^2(2\lambda - 1) + \frac{t}{L}(\frac{\alpha_1}{1-\epsilon} + 2(1-\epsilon)\alpha_1 - 1)| \leq \|J_t - J^*\| \ . \tag{2.84}$$

Before reaching the support, the error between the Jacobian estimate and the true Jacobian increases linearly. For ϵ and λ small enough, the error can become very large. This shows that when the iterates are far from the support, and the optimization path is too long, Unrolling will fail on ill-conditioned examples like the one we presented.

Numerical illustrations

We now illustrate these theoretical results depending on the number N of unrolled iterations. Figure 2.1 confirms the linear convergence of J_l^N once the support is reached. However, the convergence might be unstable when the number of iterations grows, leading to exploding gradient, as illustrated in the second case. When this happens, using a small number of iterations or truncated back-propagation becomes necessary to prevent accumulating errors. It is also interesting to look at the proportion of unstable Jacobians (see Figure 2.2). We recover behaviors observed in the first and second cases in Figure 2.1. 40% samples suffer from numerical instabilities in this example. Interestingly, ill-conditioned patterns of convergence look like what we obtain with our toy example in Figure 2.2. This suggests that bad conditioning may be a common problem even in very simple contexts and may negatively impact the gradient estimation outside of the support.

We will now focus on the gradient estimation. We display the convergence behavior of

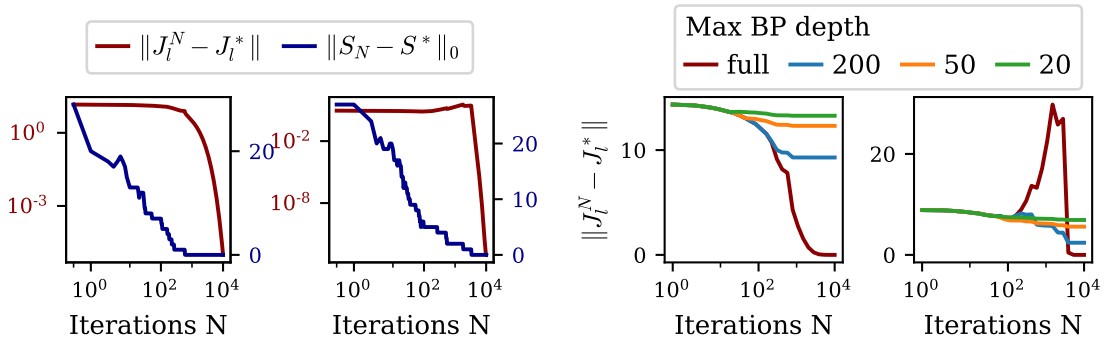

Figure 2.1: Average convergence of J_l^N toward J_l^* for two samples from the same data set. We generate a normalized random Gaussian dictionary D of dimension 30×50, and sparse codes z from a Bernoulli Gaussian distribution of sparsity 0.3 and $\sigma^2 = 1$. The signal to process is $X = DZ + B$ where B is an additive Gaussian noise with $\sigma^2_{noise} = 0.1$. The Jacobians are computed for a random perturbation $D + B_D$ of D where B_D is a Gaussian noise of scale $0.5\sigma_D^2$. J_l^N corresponds to the approximate Jacobian with N iterations of ISTA with $\lambda = 0.1$. J_l^* corresponds the true Jacobian computed with sparse codes obtained after 10^4 iterations of ISTA with $\lambda = 0.1$. $\|J_l^* - J_l^N\|$ converges linearly on the support in both cases. However, for sample 2, full back-propagation makes the convergence unstable, and truncated back-propagation improves its behavior, as described in Proposition 2.2.2. The proportion of stable and unstable samples in this particular example is displayed in Figure 2.2.

the gradients estimated by AM and by DDL with different back-propagation depths (20, 50, full) for simulated data and images in Figure 2.3. We unroll FISTA instead of ISTA to make the convergence faster. We observed similar behaviors for both algorithms in early iterations but using ISTA required too much memory to reach full convergence. As we optimize using a line search algorithm, we are mainly interested in the ability of the estimate to provide an adequate descent direction. Therefore, we display the convergence in angle defined as the cosine similarity $\langle g, g^* \rangle = \frac{Tr(g^T g^*)}{\|g\| \|g^*\|}$. The angle provides a good metric to assert that the two gradients are correlated and thus will lead to similar optimization paths. We compare g_N^1 and g_N^2 with the relative difference of their angles with g^*, defined as $\frac{\langle g_N^2, g^* \rangle - \langle g_N^1, g^* \rangle}{1 - \langle g_N^1, g^* \rangle}$. When its value is positive, DDL provides the best descent direction. Generally, when the back-propagation goes too deep, the performance of g_N^2 decreases compared to g_N^1, and we observe large numerical instabilities. This behavior is coherent with the Jacobian convergence patterns studied in Proposition 2.2.2. Once on the support, g_N^2 reaches back the performance of g_N^1 as anticipated. In the case of a real image, Unrolling beats AM by up to 20% in terms of gradient direction estimation when the number of iterations does not exceed 50, especially with a small back-propagation depth. This highlights that the principal interest of unrolled algorithms is to use them with a small number of layers – i.e., a small number of iterations.

2.3 . Unrolled Dictionary Learning in practice

This section introduces practical guidelines on Lasso-based approximate dictionary learning with unit norm constraint, and we provide empirical justifications for its ability to recover

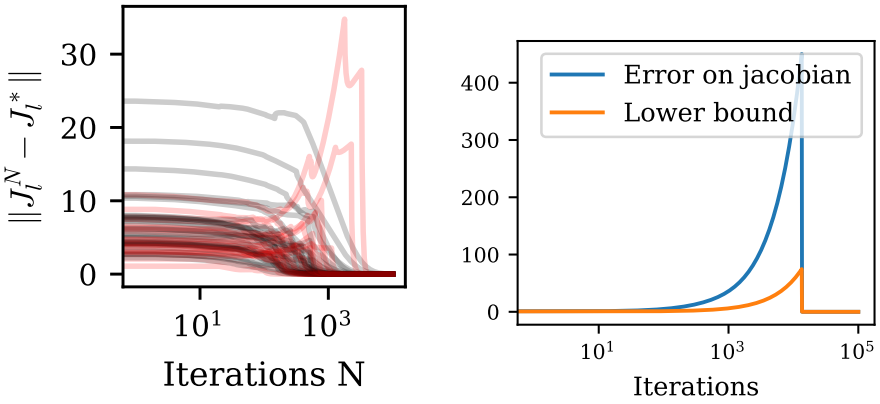

Figure 2.2: (Left) *Average convergence of J_l^N toward J_l^* for 50 samples. The data generation process is the same as in Figure 2.1. In this example, 40% of the Jacobians are unstable (red curves).* (Right) *Comparison between $\|J_N - J^*\|_2$ and the lower bound computed in Proposition 2.2.4. As expected, the error increases linearly before reaching the support in our example.*

the dictionary. We optimize with projected gradient descent and a line search to compute high-quality step sizes. The computations have been performed on a GPU NVIDIA Tesla V100-DGXS 32GB using `PyTorch` [Paszke et al., 2019].

Improvement of precision.

As stated before, a low number of iterations allows for efficient and stable computations, making the sparse code less precise. One can learn the step sizes of (F)ISTA to speed up convergence and compensate for imprecise representations, as proposed by Ablin et al. [2019] for LISTA. To avoid poor results due to large degrees of freedom in unsupervised learning, we propose a method in two steps to refine the initialization of the dictionary before relaxing the constraints on the steps sizes:

1. We learn the dictionary with fixed step sizes equal to $\frac{1}{L}$ where $L = \|D\|^2$, given by convergence conditions. Lipschitz constants or upper bounds are computed at each gradient step with norms, or the FFT for convolutions, outside the network graph's scope.

2. Then, once convergence is reached, we jointly learn the step sizes and the dictionary. Both are still updated using gradient descent with line search to ensure stable optimization.

LISTA-like algorithms with no ground truth generally aim to improve the speed of sparse coding when high precision is not required. When it is the case, the final sparse codes can be computed separately with FISTA [Beck and Teboulle, 2009] or coordinate descent [Wu et al., 2008] to improve the quality of the representation.

Unrolling v. AM.

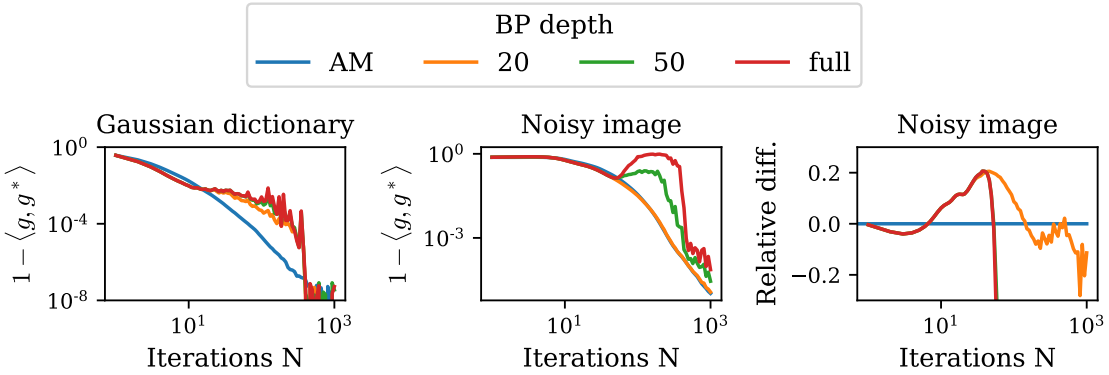

*Figure 2.3: (*left*) Gradient convergence in angle for 1000 synthetic samples. We generate a normalized random Gaussian dictionary D of dimension 30×50, and 1000 sparse codes z from a Bernoulli Gaussian distribution of sparsity 0.3 and $\sigma^2 = 1$. The signal to process is $y = Dz + b$ where b is an additive Gaussian noise with $\sigma^2_{noise} = 0.1$. The gradients are computed for a random perturbation $D + b_D$ of D where b_D is a Gaussian noise of scale $0.5\sigma^2_D$. (*center*) Gradient convergence in angle for patches from a noisy image. A 128×128 black-and-white image is degraded by a Gaussian noise with $\sigma^2_{noise} = 0.1$ and normalized. We processed 1000 patches of dimension 10×10 from the image and computed the gradients for a dictionary composed of 128 random patches. (*right*) Relative difference between angles from DDL and AM. g_N corresponds to the gradient for N iterations of FISTA with $\lambda = 0.1$. g^* corresponds to the true gradient computed with a sparse code obtained after 10^4 iterations of FISTA. Convergence is faster with DDL in early iterations and becomes unstable with too many steps.*

In Figure 2.4, we show the number of gradient steps before reaching convergence, the behavior of the loss F_N, and the recovery score defined at the beginning of the section for synthetic data generated by a Gaussian dictionary. As a reminder, $S(C) = \max_{\sigma \in \mathfrak{S}_L} \frac{1}{L} \sum_{i=1}^{L} |C_{\sigma(i),i}|$ where C is the correlation matrix between the columns of the true dictionary and the estimate. The number of iterations corresponds to N in the estimate $z_N(D)$. First, DDL leads to fewer gradient steps than AM in the first iterations. This suggests that automatic differentiation better estimates the directions of the gradients for small depths. However, computing the gradient requires back-propagating through the algorithm, and DDL takes 1.5 times longer to perform one gradient step than AM on average for the same number of iterations N. When looking at the loss and the recovery score, we notice that there is no gain in the usage of DDL for the minimization of F_N without learning the step sizes, but there is an increase in performance concerning the recovery score. DDL better estimates the dictionary for small depths, inferior to 50. When unrolling more iterations, AM performs as well as DDL on the approximate problem and is faster.

Approximate DL.

Figure 2.4 also shows that high-quality dictionaries are obtained before the convergence of F_N, either with AM or DDL. 40 iterations are sufficient to reach a reasonable solution concerning the recovery score, even though the loss is still very far from the optimum at this stage. This suggests that computing optimal sparse codes at each gradient step is unnecessary to recover the dictionary. To illustrate that, we display in Figure 2.4 the PSNR

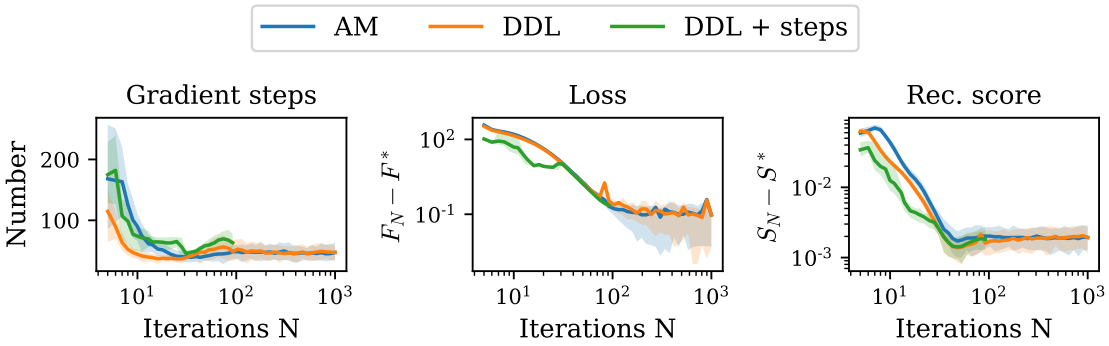

*Figure 2.4: (*left*) Number of gradient steps performed by the line search before convergence, (*center*) distance to the optimal loss, and (*right*) distance to the optimal dictionary recovery score depending on the number of unrolled iterations. We generate a normalized random Gaussian dictionary D of dimension 30×50 and sparse codes z from a Bernoulli Gaussian distribution of sparsity 0.3 and $\sigma^2 = 1$. The signal to process is $y = Dz + b$ where b is an additive Gaussian noise with $\sigma^2_{noise} = 0.1$. The initial dictionary is a random perturbation $D + b_D$ of D where b_D is a Gaussian noise of scale $0.5\sigma^2_D$. N corresponds to the number of unrolled iterations of FISTA. F^* is the value of the loss for 10^3 iterations minus 10^{-3}. S^* is the score obtained after 10^3 iterations plus 10^{-3}. The optimization is done with $\lambda = 0.1$. We compare the number of gradient steps (*left*), the loss values (*center*), and the recovery scores (*right*) for 50 different dictionaries. Due to memory and optimization time issues, DDL with step sizes learning is evaluated on 100 iterations only. We display the mean and the 10% and 90% quantiles over 50 random experiments. DDL needs fewer gradient steps to converge in early iterations and Unrolling obtains high recovery scores with only a few dozen iterations.*

of a noisy image reconstruction depending on the number of iterations, compared to full AM dictionary learning with 1000 iterations. As for synthetic data, optimal performance is reached very fast. In this particular case, the model converges after 80 seconds with approximate DL unrolled for 20 iterations of FISTA compared to 600 seconds in the case of standard DL. Note that the speed rate highly depends on the value of λ. Higher values of λ tend to make FISTA converge faster, and Unrolling becomes unnecessary in this case. On the contrary, Unrolling is more efficient than AM for lower values of λ.

Loss landscape.

The ability of gradient descent to find adequate local minima strongly depends on the structure of the problem. To quantify this, we evaluate the variation of PSNR depending on the Signal to Noise Ratio (SNR) – defined as $10 \log_{10}\left(\frac{\sigma^2}{\sigma_b^2}\right)$ where σ_b^2 is the variance of the noise – for 50 random initializations in the context of convolutional dictionary learning on a task of image denoising, with 20 unrolled iterations. We also compare in Figure 2.4 (*right*) the shapes of local minima in 1D by computing the values of the loss along a line between two local minima. This visualization confirms that (approximate) dictionary learning locally behaves like a convex function with similar local minima.

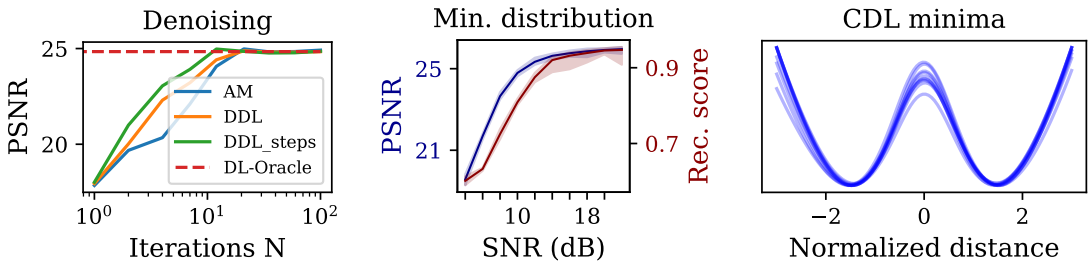

*Figure 2.5: We consider a normalized image degraded by Gaussian noise. (*left*) PSNR depending on the number of unrolled iterations for* $\sigma^2_{noise} = 0.1$*. DL stands for full AM dictionary learning. There is no need to unroll too many iterations to obtain satisfying results. (*center*) PSNR and average recovery score between dictionaries depending on the SNR for 50 random initializations in CDL. (*right*) 10 loss landscapes in 1D for* $\sigma^2_{noise} = 0.1$*. DDL is robust to random initialization when there is not too much noise.*

Conclusion

Dictionary learning is an efficient technique to learn patterns in a signal but is challenging to apply to large real-world problems. This work showed that approximate dictionary learning, which consists in replacing the optimal solution of the Lasso with a time-efficient approximation, offers a valuable trade-off between computational cost and quality of the solution compared to complete Alternating Minimization. This work provided a theoretical study of the asymptotic behavior of unrolling in approximate dictionary learning. In particular, we showed that numerical instabilities make the usage of unrolled Dictionary Learning inefficient with too many layers or iterations.

Chapter 3

Unrolled Dictionary Learning for unsupervised inverse problems

This work was carried out with the help of Florent Michel. The content of this chapter was partly published in:

Benoît Malézieux, Thomas Moreau, and Matthieu Kowalski. Apprentissage de dictionnaire par différentiation automatique pour la résolution de problèmes inverses. *GRETSI*, 2022b

While data-driven methods have been extensively studied in the context of supervised inverse problems, recent works have focused on unsupervised scenarios and provided new algorithms to learn from corrupted data only [Lehtinen et al., 2018, Bora et al., 2018, Liu et al., 2020]. Chen et al. [2021] and Tachella et al. [2022] demonstrate that a necessary condition to learn extensive priors from degraded signals is either to measure them with multiple operators which span the whole space, or to introduce weak prior knowledge such as group structures and equivariance in the model when only one operator is available. Other works based on Deep Learning have leveraged successful architectures to recover images without access to any ground truth data. In particular, Deep Image Prior shows that CNNs contain enough prior information to recover an image in several inverse problems, such as denoising or inpainting [Ulyanov et al., 2018].

Contributions of chapter 3.

We demonstrate practical limitations of prior learning methods for unsupervised inverse problems. We first provide an analysis of dictionary learning when the data is measured with a single or multiple operators. As mentioned by Tachella et al. [2022], "seeing the whole space" is a necessary condition to learn a good prior from the data, as nothing can be recovered in the kernel of the operator A. However, we point out that this is not sufficient in the case of dictionary learning. Indeed, the problem is made harder by the measurement operators, and is sometimes unfeasible even with access to the whole space.

Then we study the practical behavior of methods heavily relying on convolutions in cases where they work well (inpainting) and in cases where they fail because the prior is too weak (deblurring), and provide experiments complementary to the theoretical study of Tachella et al. [2022]. We present three examples, namely Convolutional Dictionary Learning, Deep Image Prior, and Plug and Play, and train the prior "as is" in the range space without relying on any data augmentation technique or equivariance. Finally, we compare unrolled networks based on Analysis and Synthesis. We evaluate their performance on several inverse problems without access to any ground truth data for three classes of dictionaries in terms of recovery of the dictionary and reconstruction of the signal.

3.1 . The main bottleneck of prior learning in inverse problems.

For inverse problems, the dimension of the measurements m is often smaller than the dimension of the signal n. This dimension reduction implies that information on the signal contained in the null space of $A \in \mathbb{R}^{m\times n}$ is lost during the observation process, and needs to be reconstructed from the observed signal. We first aim to study the impact of this degradation on constraint-free prior learning through the lens of dictionary learning.

Dictionary learning with a single measurement operator. Dictionary learning assumes that the signal can be decomposed into a sparse representation in a redundant basis of patterns. Taking the example of Lasso-based dictionary learning, recovering X would require solving a problem of the form

$$\min_{Z\in\mathbb{R}^{L\times N}, D\in\mathcal{C}} \frac{1}{2}\|ADZ - Y\|_2^2 + \lambda\|Z\|_1 \ . \tag{3.1}$$

We first aim to see the impact of A on the algorithm ability to recover a proper dictionary. In Proposition 3.1.1, we focus on inpainting where the measurement operator is a binary mask or equivalently a diagonal matrix with m non-zeros elements.

PROPOSITION 3.1.1. *Let $A = diag(\lambda_1, \cdots, \lambda_n) \in \mathbb{R}^{n\times n}$ be a diagonal measurement matrix where $m < n$, $\lambda_1 \geq \cdots \geq \lambda_m > 0$ and $\lambda_{m+1} = \cdots = \lambda_n = 0$. Let $D_0 \in \mathbb{R}^{n\times L}$ and D' be such that*

$$D' = \begin{pmatrix} \frac{\|D_{0,j}\|}{\|D_{0,j,m}\|} D_{0,j,m} \\ 0_{n-m} \end{pmatrix}_{1\leq j\leq L}, \text{ where } D_0 = \begin{pmatrix} D_{0,m} \\ D_{0,n-m} \end{pmatrix}$$

Then

$$\begin{aligned} &\min_Z \frac{1}{2}\|AD'Z - Y\|_2^2 + \lambda\|Z\|_1 \\ &\leq \min_Z \frac{1}{2}\|AD_0Z - Y\|_2^2 + \lambda\|Z\|_1 \ . \end{aligned}$$

Proof 3.1.1

Let $Z_0 \in \operatorname{argmin}_Z \frac{1}{2}\|AD_0Z - Y\|_2^2 + \lambda\|Z\|_1$. Let $Z'_j = \frac{\|D_{0,j,m}\|}{\|D_{0,j}\|} Z_{0j}$. Then

$$\|AD'Z' - Y\|_2 = \|AD'_0Z'_0 - Y\|_2 \tag{3.2}$$

$$\|Z'\|_1 \leq \|Z_0\|_1 \tag{3.3}$$

The result follows.

In this simple case, our proposition shows that the optimal dictionary must be 0 in the null space of A. The core idea behind the proof is that due to invariances, the optimal solution for dictionary learning is contained in an equivalence class $\{PSD' + V\}$ where P is a permutation matrix, S is a scaling matrix, D' is a matrix of rank m and V is a matrix of rank $n - m$ such that $PSD' \in \ker(A)^\perp$ and $V \in \ker(A)$. Given a dictionary $PSD' + V$ in this equivalence class, the dictionary PSD' is always a better minimizer after proper rescaling. Therefore, the solver puts to 0 all directions from which A loses the information to maximize the input from the others. Proposition 3.1.2 generalizes Proposition 3.1.1 to the case of rectangular matrices.

PROPOSITION 3.1.2. *Let $A \in \mathbb{R}^{m\times n}$ be a measurement matrix where $m < n$, and let $Y \in \mathbb{R}^{m\times N}$ be the observed data. If a dictionary $D \in \mathbb{R}^{n\times L}$ minimizes $\min_{Z\in\mathbb{R}^{L\times N}, D\in\mathcal{C}} \frac{1}{2}\|ADZ - Y\|_2^2 + \lambda\|Z\|_1$, then $D \in \ker(A)^\perp$.*

Proof 3.1.2

Let $A \in \mathbb{R}^{m\times n}$, $Y \in \mathbb{R}^{m\times T}$. We aim to solve

$$\min_{D\in\mathcal{C},Z} \frac{1}{2}\|ADZ - Y\|_2^2 + \lambda\|Z\|_1 \tag{3.4}$$

Performing a SVD on A leads to

$$A = U\Lambda V^* \text{ s.t. } U \in \mathbb{R}^{m\times m}, V \in \mathbb{R}^{n\times n} \text{ and } UU^* = I_m, VV^* = I_n \tag{3.5}$$

$$\Lambda = \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & \lambda_m & \cdots & 0 \end{bmatrix} \tag{3.6}$$

Then,

$$\min_{D\in\mathcal{C},Z} \frac{1}{2}\|ADZ - Y\|_2^2 + \lambda\|Z\|_1 \tag{3.7}$$

$$= \min_{D\in\mathcal{C},Z} \frac{1}{2}\|U\Lambda V^*DZ - Y\|_2^2 + \lambda\|Z\|_1 \tag{3.8}$$

$$= \min_{D\in\mathcal{C},Z} \frac{1}{2}\|\Lambda V^*DZ - U^*Y\|_2^2 + \gamma\|Z\|_1 \tag{3.9}$$

$$= \min_{\tilde{D}\in\mathcal{C},Z,D=V\tilde{D},\tilde{Y}=U^*Y} \frac{1}{2}\|\Lambda\tilde{D}Z - \tilde{Y}\|_2^2 + \gamma\|Z\|_1 \tag{3.10}$$

Adding zeros to Λ to make it square, and adding zeros at the end of the measurement vector U^*Y to respect dimensions, the problem reduces to

$$\begin{aligned} &\min_{D\in\mathcal{C},Z} \frac{1}{2}\|\Lambda\tilde{D}Z - \tilde{Y}\|_2^2 + \lambda\|Z\|_1 \\ &\text{s.t. } \Lambda = diag(\lambda_1, \cdots, \lambda_m, 0, \cdots, 0), \tilde{Y} = \begin{pmatrix} U^*Y \\ 0_{n-m} \end{pmatrix} . \end{aligned} \tag{3.11}$$

Then, Proposition 3.1.1 applies and an optimal dictionary is contained in $\ker(A)^\perp$.

Similarly to what happens in inpainting, nothing can be recovered in the null space of A. Thus, we can only expect to learn a dictionary of rank m with a single measurement matrix. This result relates to the one from Tachella et al. [2022] which says that the signals cannot be recovered where there is no information.

Dimension reduction makes dictionary learning harder in the range space. Even in the range space of the signal, a good dictionary cannot always be learned reliably. Guarantees of identifiability of the dictionary or local recovery are strongly based on the ability of the sparse coding algorithm to recover an accurate estimation of Z Arora et al. [2015], Gribonval et al. [2015], Chatterji and Bartlett [2017]. As the dimension of the measurement m becomes smaller than the dimension of the signal n, these conditions are not valid anymore. As an example, if D is a Gaussian random dictionary, the theory of compressed sensing states that $n \geq 2s\ln(\frac{L}{s})$ where s is the sparsity of Z is a sufficient condition to be able to recover Z with high probability Foucart and Rauhut [2013b]. When the dictionary is degraded by a matrix A, this constraint becomes $m \geq 2s\ln(\frac{L}{s})$ and the sparsity level s has to decrease by a ratio close to $\frac{m}{n}$ to compensate for the loss of information. This implies that recovering the part of the dictionary not contained in the null space of A also becomes harder with the corruption of the data.

Figure 3.1 shows the recovery score for data generated with a 100×100 random Gaussian dictionary, depending on the size of the measurements and on the sparsity of a Bernoulli Gaussian signal, after degradation by a single compressed sensing operator. We compare it to the perfect score that we can achieve in the range space of the operator. We evaluate the quality of the dictionary, based on the Pearson correlation of their columns. To make the metric sign and permutation invariant, we use a best linear sum assignment $S(C) = \max_{\sigma\in\mathfrak{S}_n} \frac{1}{n}\sum_{i=1}^{n} |C_{\sigma(i),i}|$, where $\mathfrak{S}_n$ is the group of permutations of $[1, n]$ and C is the cost matrix whose entry i, j compares the atom i of the first dictionary and j of the second. It is equal to 1 when the dictionary is perfectly recovered. The recovery score drops when the dimension m decreases and small values of m require a high sparsity level to recover the dictionary in the range of A.

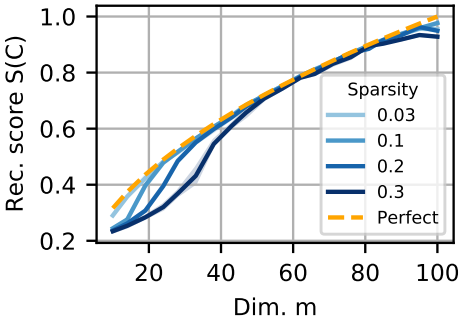

Figure 3.1: Recovery score for Gaussian dictionaries 100×100, after degradation by a single compressed sensing operator $m \times 100$. When the dimension m decreases, the part of the dictionary not contained in the null space can be recovered only with sparse signals.

Seeing the data through multiple operators. Even though it is not possible to recover the whole dictionary from a single measurement operator, the situation changes when the measurement matrix is sample dependent. Indeed, several operators may span different parts of the signal space and make it possible to recover the missing part of the signal. In this section, we focus on cases where the data are observed through a set of N_m measurement matrices $(A_i)_{1\leq i\leq N_m}$, and consider the task of learning a dictionary with the

associated lasso-based optimization problem

$$\begin{aligned} &\min_{D\in\mathcal{C}} F(Z_A(D), D) \\ &\triangleq \sum_{i=1}^{N_m} \frac{1}{2}\|A_i D Z_{A_i}(D) - Y_i\|_2^2 + \lambda_i \|Z_{A_i}(D)\|_1 \ , \\ &\text{with} \quad Z_{A_i}(D) = \operatorname*{argmin}_Z \frac{1}{2}\|A_i D Z - Y_i\|_2^2 + \lambda_i \|Z\|_1 \ . \end{aligned} \tag{3.12}$$

Here $Z_A(D) = (Z_{A_1}(D), \dots Z_{A_{N_m}}(D))$ denotes the sparse codes related to each operator. This problem is non-convex and is usually solved through gradient descent, in order to find a local minimum. In the following, we study the cases when the local minima of Equation 3.12 are also local minima for the problem without observation operators and provide an empirical analysis in different scenarios. With multiple measurement operators, the gradient of Equation 3.12 is given by

$$\nabla_D F(Z_A(D), D) = \sum_{i=1}^{N_m} A_i^\top (A_i D Z_{A_i}(D) - Y_i) Z_{A_i}(D)^\top \ . \tag{3.13}$$

The main difficulty in studying this quantity is that the sparse codes estimate $Z_{A_i}(D)$ depends on D and A_i. Each operator provides measurements from a limited number of samples in the data-set, and the sparse codes are different with and without A. Thus, we consider the simplest case where $Z = Z_{A_i}(D)$ is the same for all A_i, including $A = I$. This is an easier problem than the general formulation in Equation 3.12. If this is not feasible, then Equation 3.12 is not feasible. In this case, we have

$$\nabla_D F(Z_A(D), D) = (\sum_i A_i^\top A_i) \nabla_D F(Z_I(D), D) \ . \tag{3.14}$$

KKT conditions imply that the gradient $\nabla_D F(Z_A(D), D)$ must vanish at local minima. Whenever $\sum_i A_i^\top A_i$ is injective, $\nabla_D F(Z_I(D), D)$ vanishes if and only if $\nabla_D F(Z_A(D), D)$ vanishes. Thus, local minima of Equation 3.12 are also local minima for the original problem where $A_i = I$. This means that when $\sum_i A_i^\top A_i$ spans the entire space, the dictionary from the original problem can be recovered. Otherwise local minima of $F(Z_A(D), D)$ are not necessarily local minima of $F(Z_I(D), D)$. This case boils down to the case previously studied, as $\sum_i A_i^\top A_i$ is full rank whenever the rank of the matrix obtained by stacking the operators $(A_1^\top, \cdots, A_{N_m}^\top)$ is equal to n. The message from these results is essentially the same as the one from Chen et al. [2021]: a necessary condition for recovery is that the operators span the whole space. It is however important to note that this is only a necessary condition to recover the dictionary, as sparse coding guarantees may not be met when the dimension m is too small. We now present two examples of inverse problems with multiple operators to illustrate what happens in practice.

- Compressed sensing (CS). When all A_i are random Gaussian matrices, $(A_1^\top, \cdots, A_{N_m}^\top)$ is also a random Gaussian matrix of dimension $n \times N_m m$. Therefore, it is of rank n with probability 1 if $N_m \geq \lfloor \frac{n}{m} \rfloor + 1$. Figure 3.3 illustrates that it is indeed a necessary condition to recover D, but it is not sufficient when m is too small, because sparse coding becomes inefficient. Multiview compressive dictionary learning has also been studied in Anaraki and Hughes [2013], Pourkamali-Anaraki et al. [2015], Chang et al. [2019].

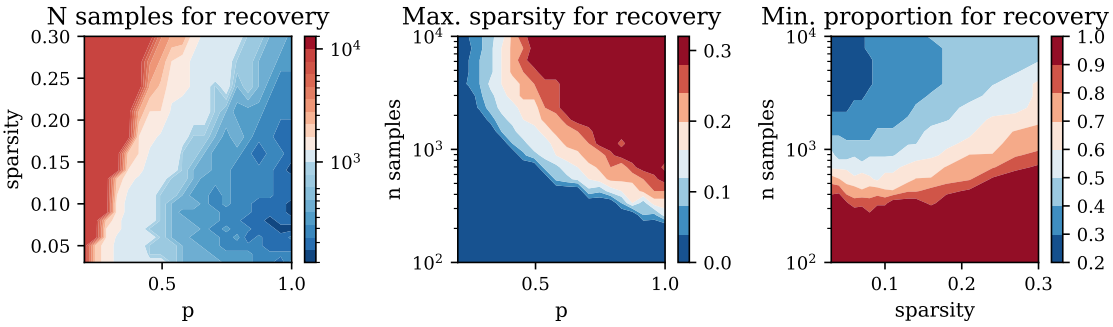

Figure 3.2: We generate the data from a Gaussian dictionary of size 100×100 *and Bernoulli Gaussian sparse codes of sparsity* s *(average rate of non zero coordinates). Then, we degrade the data with a binary mask of variable rates of available coordinates* p*. We learn a dictionary of size* 100×100 *over several values of* λ*. We show the ability of the algorithm to recover the dictionary, depending on* p*, the number of training samples and the level of sparsity in the data. Dictionary recovery is defined as obtaining a recovery score of at least 0.95. We display the results from three different perspectives: (Left) Minimal number of samples necessary to recover the dictionary depending on sparsity and rate of available coordinates in the data. A number of samples larger than* 10^4 *means no recovery possible. (Center) Maximal level of sparsity* s *(maximal proportion of non zero coordinates) to recover the dictionary depending on the number of samples and the rate of available coordinates. A level equal to 0 means no recovery possible. (Right) Minimal rate of available coordinates for recovery depending on the number of samples and the level of sparsity. A level equal to 1 means no recovery possible. These figures show that there is a hard limit to what can be learned depending on the proportion of missing values and sparsity, regardless the number of training samples. Having access to the whole signal space is not a sufficient condition to recover the dictionary.*

- Inpainting. All A_i are binary masks with coefficients following Bernoulli distributions of parameters $p_1, \cdots, p_n$, i.e. $A_i = \text{diag}(a_i^1, \cdots, a_i^n)$ where each a_i^j is equal to 1 with probability p_j. The rank of $(A_1^\top, \cdots, A_{N_m}^\top)$ is equal to n if for each coordinate j there exists an index i such that $a_i^j = 1$. This happens with probability $\prod_j (1-(1-p_j)^{N_m})$. Figure 3.2 shows that similar to CS, this is a necessary but insufficient condition to recover a proper dictionary. Even when the number of samples compensates for missing values, the sparsity of the data plays a great role in the ability of the algorithm to recover the proper dictionary after heavy dimension reduction. To illustrate what happens on real data, we consider the example of image inpainting. Let $A \in \{0,1\}^{h \times w}$ be a binary mask used to observe an image $X \in [0,1]^{h \times w}$ and $Y = A \odot X$ be the observed image. While the operator is unique when we consider the whole image at once, learning a dictionary from patches of size n from the image is equivalent to learning a dictionary with multiple operators in Equation 3.12. Denoting $A_{ij} = \text{diag}(A_{ni:n(i+1),nj:n(j+1)})$ and $Y_{ij} = \text{vect}(Y_{ni:n(i+1),nj:n(j+1)})$ the i,j-patch, patch-based dictionary learning solves

$$\min_{Z_{ij}, D \in \mathcal{C}} \sum_{i,j} \frac{1}{2} \|A_{ij} D Z_{ij} - Y_{ij}\|_2^2 + \lambda \|Z_{ij}\|_1 \tag{3.15}$$

The dictionary should be recovered if the image is large enough and if there are not too many masked pixels. In Figure 3.3, we show the PSNR (Peak Signal to Noise

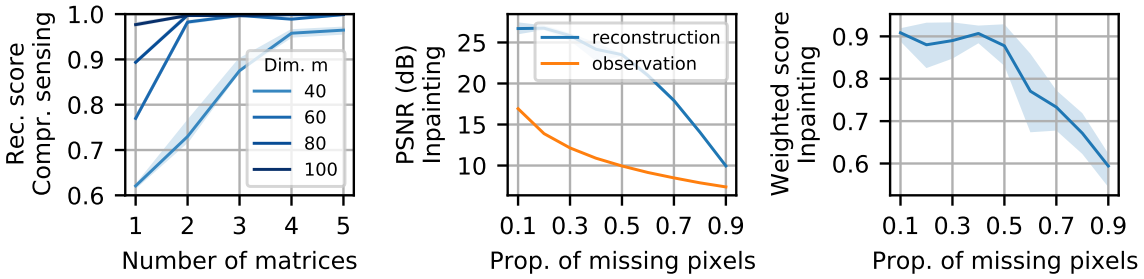

*Figure 3.3: (**left**) Recovery score for Gaussian dictionaries 100×100, after degradation by N_m compressed sensing operators $m \times 100$. $N_m \geq \lfloor \frac{n}{m} \rfloor + 1$ is necessary, but not sufficient to recover the dictionary when m is too small. In the case of inpainting, PSNR (**center**) and weighted recovery score (**right**) depend on the proportion of missing values in dictionary learning on patches from a natural image. When the dimension of the measurement space is large enough, the algorithm recovers the image and the supervised dictionary successfully.*

ratio) and the recovery score depending on the proportion of missing values in an image of resolution 128×128, from which we extract patches of size 10×10. To take into account that some atoms might not be as relevant as others, the score is re-weighted by the sum of corresponding activations in the sparse codes Z. The recovery score drops when the proportion of missing values is larger than 50%. Otherwise, the image is successfully recovered even when the dictionary is learned from the degraded observation. This is why dictionary learning led to good results in unsupervised inpainting in the literature Szabó et al. [2011], Studer and Baraniuk [2012], Naumova and Schnass [2017].

We have demonstrated that dictionary learning won't operate in the null space of the measurement matrix. However, the usage of multiple operators allows for mitigating that issue, the whole signal space being seen through different matrices A_i. Our experiments with synthetic and real data also show that this is only a necessary condition to learn a good dictionary. In some cases, the sparse codes cannot be recovered as the information is too degraded. Reducing the dimension of the observations could then be a hard limit to dictionary learning, and theoretical results on the convergence of classical optimization methods such as Alternating Minimization would be of great interest to ensure the identifiability of the dictionary with multiple operators. In the following, we show that well chosen weak prior knowledge can lift the problem and allow to recover the information from the kernel space of a single operator through the example of convolutions in imaging.

3.2 . Weak prior knowledge through convolutions

The usage of convolutions in Deep Learning LeCun et al. [1998] has encountered tremendous success in a broad range of tasks from image classification to reconstruction. Convolutions and convolutional neural networks are efficient to analyze translation invariant data while reducing the number of parameters to be learned. In this section, we provide elements to understand the efficiency and the limitations of convolutions used as weak prior knowledge for unsupervised image reconstruction through the study of three methods based on prior learning: Convolutional Dictionary Learning Grosse et al. [2007], Plug

and Play Chan et al. [2016] and Deep Image Prior Ulyanov et al. [2018]. All computations have been performed on a GPU NVIDIA Tesla V100-DGXS 32GB using `PyTorch` Paszke et al. [2019].

Convolutional dictionary learning (CDL) consists in learning kernels of relatively small dimensions from a signal Y. Lasso-based CDL solves a problem of the form

$$\min_{z_k, d_k \in \mathcal{C}} \frac{1}{2} \|A \sum_k d_k * z_k - Y\|_2^2 + \lambda \sum_k \|z_k\|_1 \ . \tag{3.16}$$

Deep Image Prior (DIP) takes advantage of CNN architectures to project the observed image into a well-suited range space by drawing a random code vector z in the latent space and optimizing the parameters of the network f as follows

$$\min_\theta \|Y - A f_\theta(z)\|_2^2 \ . \tag{3.17}$$

Plug and Play (PnP) is an iterative algorithm inspired from proximal gradient descent, which recovers images from an observation Y with steps of the form

$$X_{n+1} = f_\theta \left(X_n - \tau A^*(AX_n - Y) \right) \quad \forall n \geq 1 \ , \tag{3.18}$$

where $X_0 = 0$, τ is a step size and f_θ is an image denoiser. CDL and DIP can be applied to a single observation without training on a data set, the prior being learned directly on one piece of degraded data without needing other information. In contrast, PnP usually resort to a deep denoiser generally pre-trained on a clean database. As we focus on the unsupervised setting, we adapt PnP by training the denoiser on degraded data instead. In this case, we consider that we have access to a dataset $(Y_i)_{1 \leq i \leq N}$ where each $Y_i = AX_i + \epsilon_i$ is an observation of an original image X_i degraded by the same operator A and a gaussian noise ϵ_i. We artificially generate noisy images $(Y_i')_{1 \leq i \leq N}$ from our data-set of observations $Y_i' = Y_i + \epsilon_i'$, and we train a DnCNN Zhang et al. [2017] to recover Y_i from Y_i' in the range space of A by minimizing

$$\min_\theta \frac{1}{N} \sum_i \|A(f_\theta(Y_i') - Y_i)\|_2^2 \ . \tag{3.19}$$

The idea is to check in which case the architecture can compensate for the lack of information in the kernel of A by learning from the information in the range space of A. To point out the limits of these prior learning algorithms, we will compare them to two reconstruction methods based on Total Variation (TV) Chambolle et al. [2010] and sparse wavelets Mallat [2008].

The purpose is to highlight the hard limits of unsupervised methods in various contexts. Therefore, we evaluate the performance reached by each algorithm over oracle hyper-parameters, namely hyper-parameters leading to the best performances. While evaluating hyper-parameter sensitivity is necessary when comparing different methods, it is orthogonal to our study, which considers the difference between supervised and unsupervised training of similar methods.

Why convolutions are likely to work on tasks like inpainting.

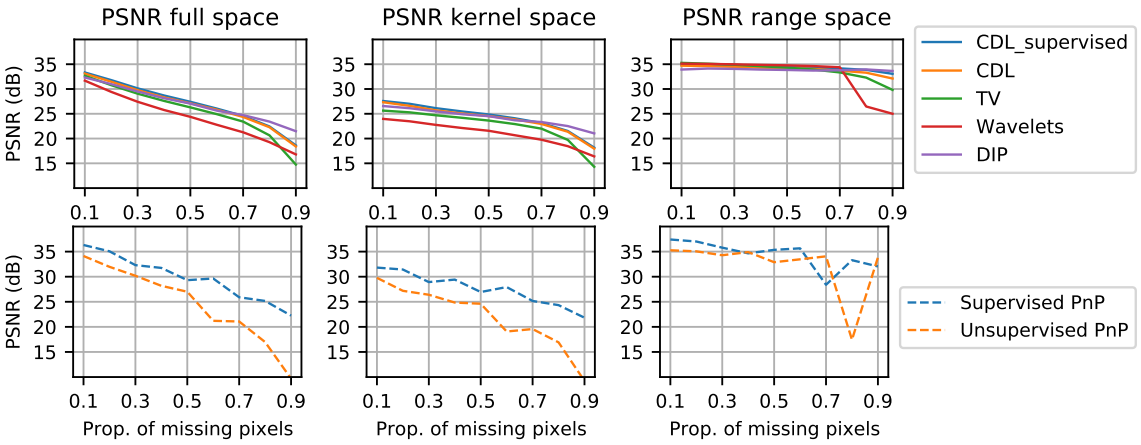

*Figure 3.4: PSNR depending on the proportion of available pixels in the full space, kernel space, and range space of the masking operator for reconstruction methods based on CDL, DIP, TV, and wavelets for a 256 × 256 grey-level image (**top**), and for PnP based reconstruction on 160 × 160 grey-level images (**bottom**) with a SNR of 20db. Unsupervised prior learning methods work better than hand-crafted methods even with a lot of lacking information and can recover missing information in the kernel space. Moreover, they perform close to supervised methods (CDL supervised and PnP supervised) when the ratio of missing pixels is not too high.*

Works on prior learning in unsupervised inverse problems often evaluate the performance of the methods they propose on an inpainting task Ulyanov et al. [2018], Chen et al. [2021] and achieve very good performance compared to supervised learning techniques. Here, we provide elements to understand why this problem, in particular, is feasible with the help of convolutional dictionaries or neural networks without access to ground truth data.

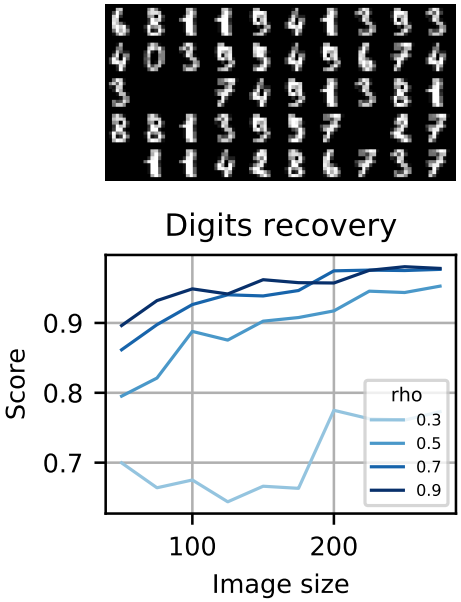

Figure 3.5: The recovery score of convolutional dictionaries depends on the image size and the rate of available pixels ρ. Increasing the size improves the quality at high enough rates.

Learning convolutional dictionaries from incomplete data. To understand what happens in inpainting, let's consider a simple one-dimensional signal example. Let X_t be a wide sense stationary (WSS) random process, and let A_t be an i.i.d Bernoulli process of mean ρ. The observed signal $Y_t = A_t X_t$ is also a WSS random process and its auto-correlation function $R_Y(\tau)$ is

$$R_Y(\tau) = \mathbb{E}[A_t X_t A_{t+\tau} X_{t+\tau}] = R_X(\tau)\mathbb{E}[A_t A_{t+\tau}] \tag{3.20}$$

$$= \rho^2 R_X(\tau)\mathbb{1}_{\tau\neq 0} + \rho R_X(\tau)\mathbb{1}_{\tau=0} \ . \tag{3.21}$$

Then, the Wiener-Khintchine theorem assures that the power spectral density of X and Y are proportional. This shows that with sufficient samples in the signal, the masking process won't affect the spectrum of the original signal X, and translation invariant priors can take advantage of the information from all

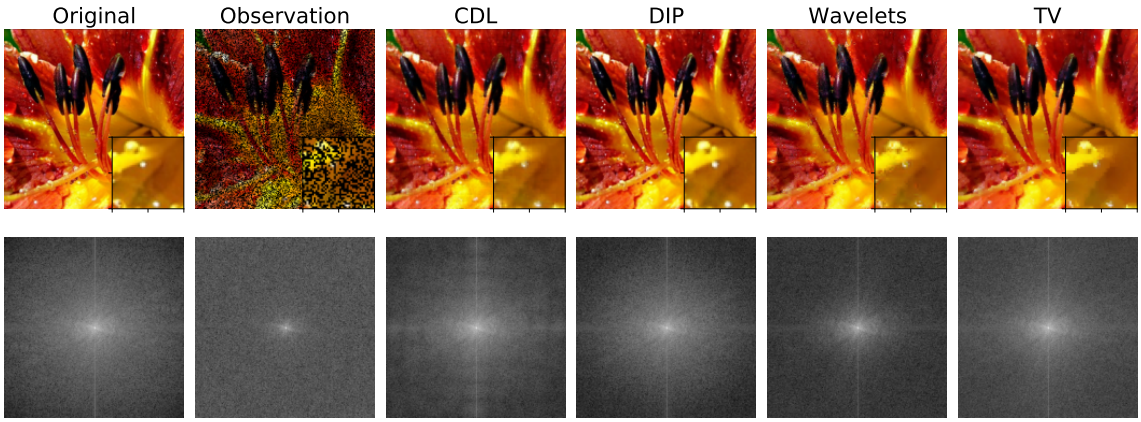

Figure 3.6: Reconstruction and PSD of a 256 × 256 RGB image with 50% missing pixels in a noiseless scenario. PSNR: CDL 34.8dB, DIP 34.1dB, Wavelets 34.7dB, TV 34.3dB. The PSD reveals the presence of ringing artifacts in the reconstruction by CDL. Otherwise, unsupervised algorithms recover the whole spectrum of the original image and do as well as hand-crafted methods.

frequencies.

We illustrate the practical implication of this observation on the ability of CDL to recover 10 digits from an image, depending on the size of the image and the rate of available pixels ρ in Figure 3.5. As expected, the performance increases with the size when ρ is not too low. It is essential to note that having access to all frequencies is only a necessary condition to learn a good dictionary, as sparse coding assumptions are not met when there are too many missing values. Of course, these results do not stand for non-stationary signals.

Unsupervised reconstruction. Similar effects can be observed for reconstruction. Natural images are stable enough to allow convolution-based algorithms to learn from all frequencies that are present in the signal. Figure 3.4 presents an example where a single natural image is degraded by a random binary mask and gaussian noise and reports the PSNR of the reconstruction in the mask kernel space and range space for CDL, DIP, and methods based on TV and sparse wavelets for different rates of missing pixels with a SNR of 20dB. Supervised means the algorithm learns a prior on the clean signal and uses it for reconstruction after degradation, whereas unsupervised means that the prior is learned directly on the observation. The experiments highlight that unsupervised methods work as well as supervised CDL and are better than hand-crafted priors in the kernel of A when the noise level is not too high (SNR $\geq$ 20). They succeed in learning in the range space of A and generalizing in the kernel space. The PSNR drops in favor of TV and wavelets when the noise increases, as it becomes more challenging to learn the structure of the signal. Figure 3.6 provides a visual example in the noiseless case. Unsupervised algorithms successfully recover the original image after degradation by a binary mask with 50% pixels missing. The PSD shows that low and high frequencies are retrieved, despite ringing artifacts in the case of CDL.

In the case of Plug-and-Play, we train a DnCNN to recover noisy images from the dataset

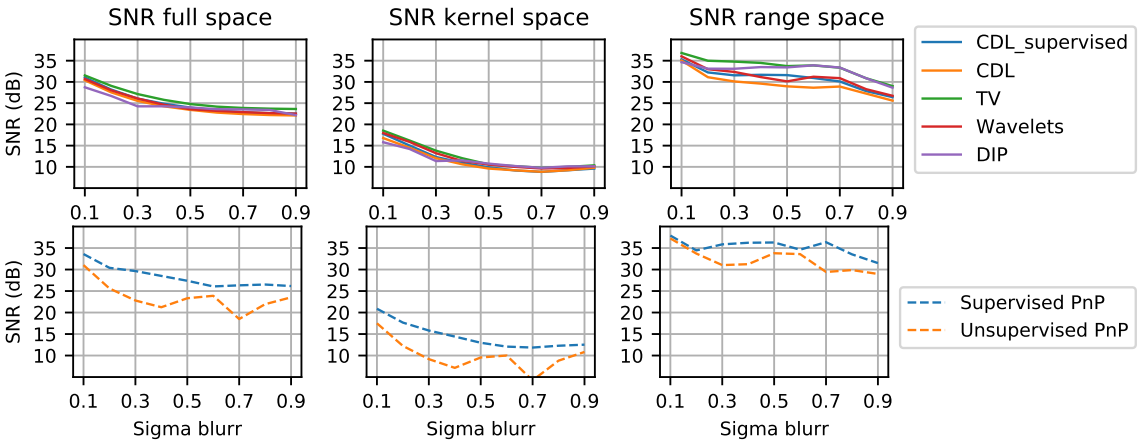

*Figure 3.7: SNR depending on the size of the blur in the full space, kernel space, and range space of the blur operator for reconstruction methods based on CDL, DIP, TV, and wavelets for a 256 × 256 grey-level image (**top**), and for PnP based reconstruction on 160 × 160 grey-level images (**bottom**) with a SNR of 20db. This time, unsupervised prior learning methods fail to recover information in the kernel space. More surprisingly, supervised PnP and CDL also struggle in the kernel space.*

Imagenette[1] and plug it into an iterative reconstruction algorithm. In the unsupervised case, the denoiser learns how to recover degraded images, as explained above. The results are shown in Figure 3.4 for SNR=20dB. When the noise is low, i.e., SNR ≥ 20, and when the rate of missing pixels stays below 50%, unsupervised and supervised PnP leads to similar performance levels in terms of PSNR. As for the single image example, unsupervised PnP can generalize what is learned in the range space of A to the kernel space and performs closely to its supervised counterpart as long as the rate of masked pixels is not too large.

The pitfall of convolutions in deblurring.

Convolutions work well when all frequencies are preserved, as shown for inpainting. However, several inverse problems involve recovering a signal with missing frequencies. In super-resolution, all odd frequencies lack in the signal. In deblurring, the signal is observed after degradation by a low-pass filter. Unsupervised prior learning becomes troublesome in these cases, as mentioned in Tachella et al. [2022]. We will focus on the example of deblurring in the following.

The CDL problem can be re-written in terms of Fourier transforms with the Parseval equality

$$\min_{z_k, d_k} \frac{1}{2} \|\hat{A} \sum_k \hat{d}_k \hat{z}_k - \hat{Y}\|_2^2 + \lambda \sum_k \|z_k\|_1 \ . \tag{3.22}$$

As the spectrum $\hat{A}$ is low-pass, nothing is observed in high frequencies. Thus, optimal dictionaries contain atoms $(d_k)_k$ with high frequencies set to 0, for the same reason as pointed out in Proposition 3.1.1.

For a blurred image, Figure 3.8 displays its reconstructions and their PSD for various methods and Figure 3.7 their performances for various blur sizes in the kernel and range

[1]The data are available at https://github.com/fastai/imagenette

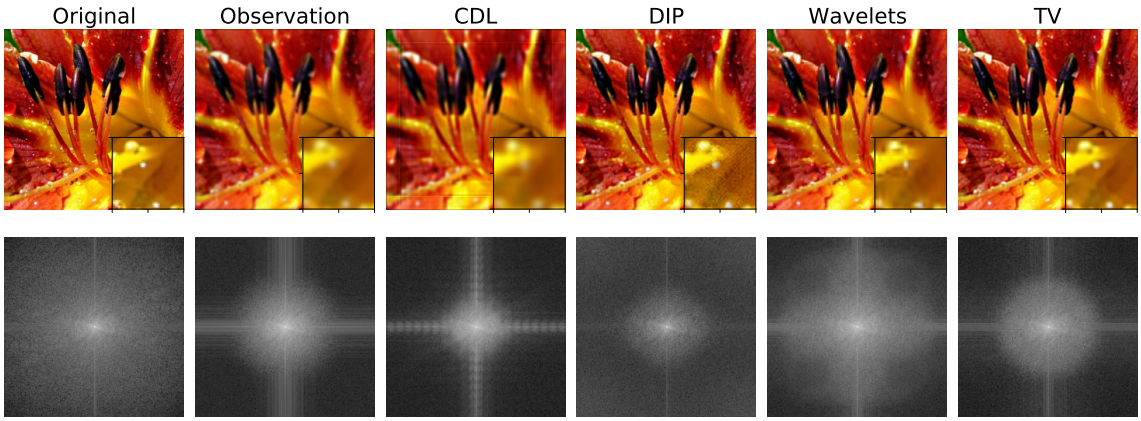

Figure 3.8: In a noiseless scenario, reconstruction and PSD of a 256 × 256 RGB image blurred by a gaussian kernel. PSNR: CDL 31.9dB, DIP 31.7dB, Wavelets 34.6dB, TV 32.7dB. The PSD clearly shows that nothing is learned in high frequencies compared to what was obtained in inpainting.

spaces. These results show that neither CDL nor DIP can recover information outside the span of the blur, i.e., in high frequencies. While CDL puts all high frequencies to 0, DIP adds noise. The same phenomenon appears with PnP: there is a performance gap between supervised and unsupervised learning in the kernel, and generalization from the range space to the kernel space is impossible.

3.3 . Unrolled dictionary learning in unsupervised inverse problems

In section 2.1, the study on gradient estimation in unrolled dictionary learning highlighted that Unrolling may only be helpful in early iterations. However, the performances critically depend on the optimization problem and evaluating the differences between Analysis and Synthesis in the framework of unrolled algorithms is of interest. Here, we empirically compare both formulations for dictionary learning in inverse problems without access to ground truth data regarding dictionary recovery, reconstruction quality, and objective function behavior. We aim to study whether Unrolling allows to learn a dictionary and recover the signal with either Analysis or Synthesis, within the limits described above.

Unrolling Analysis.

As opposed to Synthesis, Analysis supposes the existence of a transform $\Gamma^\top$, which changes the signal into a sparse representation. The optimization problem corresponding to Analysis is

$$\min_{X\in\mathbb{R}^{n\times T},\Gamma\in\mathcal{C}_A} F_A(X,\Gamma) \triangleq \frac{1}{2}\|AX-Y\|_2^2+\lambda\|\Gamma^\top X\|_1 \ , \tag{3.23}$$

where $\mathcal{C}_\mathcal{A}$ is the set of constraint and λ is a regularization hyperparameter. As a reminder, Synthesis consists of learning D in a set of constraints $\mathcal{C}_S$ by solving

$$\min_{Z\in\mathbb{R}^{L\times T},D\in\mathcal{C}_S} F_S(X,D) \triangleq \frac{1}{2}\|ADZ-Y\|_2^2+\lambda\|Z\|_1 \ . \tag{3.24}$$

Moreover, all the results on gradient estimation given in section 2.1 remain valid in the case of inverse problems where $A \neq I$. As a matter of fact, Corollary 3.3.1 emphasizes that

there is no difference between the convergence behavior of the Jacobian with and without A.

COROLLARY 3.3.1. *Let D_l be row l of D. At iteration N of ISTA, the Jacobian of $z_{N+1}(AD)$ with respect to D_l can be written as a function of $(J^i_{N+1}(AD))_{1\leq i\leq m}$ where $J^i_{N+1}(AD)$ is the Jacobian of row i of AD computed in Theorem 2.2.1:*

$$\frac{\partial(z_{N+1}(AD))}{\partial D_l} = \sum_{i=1}^{m} A_{i,l} J^i_{N+1}(AD)$$

The Jacobian of $z_{N+1}(AD)$ converges at the same speed as $(J^i_{N+1}(AD))_{1\leq i\leq m}$.

Proof 3.3.1

$z_{N+1}(AD)$ can be written as a function of all rows of AD:

$$z_{N+1}(AD) = z_{N+1}((AD)_1, \dots, (AD)_L) \tag{3.25}$$

Then the chain rule can be applied as follows:

$$\frac{\partial z_{N+1}(AD)}{\partial D_l} = \sum_{i=1}^{L} \frac{\partial (AD)_i}{\partial D_l} \frac{\partial z_{N+1}((AD)_1, \dots, (AD)_L)}{\partial x_i} \tag{3.26}$$

$$= \sum_{i=1}^{L} A_{i,l} \frac{\partial z_{N+1}((AD)_1, \dots, (AD)_L)}{\partial x_i} \tag{3.27}$$

$$= \sum_{i=1}^{L} A_{i,l} J^i_{N+1}(AD) \tag{3.28}$$

Analysis can be written as a bi-level optimization problem on the same principle as Synthesis in section 2.1

$$\min_{\Gamma\in\mathcal{C}_A} G_A(\Gamma) \triangleq F_A(X^*(\Gamma), \Gamma) \quad \text{with} \quad X^*(\Gamma) = \operatorname*{argmin}_{X\in\mathbb{R}^{n\times T}} F_A(X, \Gamma) \ . \tag{3.29}$$

Several algorithms have been proposed in the literature to solve the inner problem, i.e., to compute $X^*(\Gamma)$. In particular, Lasso-based Analysis is a particular case of a more general problem

$$\min_x f(Ax) + g(\Gamma^\top x) \ , \tag{3.30}$$

with f convex differentiable and g convex non smooth. Equation 3.30 has been largely studied in the literature in the past years, and several algorithms have been proposed to solve it efficiently [Condat et al., 2019]. In this work, we chose to unroll the algorithm referred to as Condat-Vu algorithm [Condat, 2013, Vu, 2013], which is a primal-dual splitting algorithm. We describe it in Algorithm 4. As for Synthesis, the idea is to replace $X^*(\Gamma)$ by an approximation $X_N(\Gamma)$ given by N iterations of Condat-Vu, then compute the gradient of $F_A(X_N(\Gamma), \Gamma)$ with respect to Γ with automatic differentiation and minimize the loss by gradient descent. We will refer to this architecture as *Deep Primal-Dual Prior Learning* (DPDPL). This network is similar to the one proposed in Jiu and Pustelnik [2021] for supervised image reconstruction.

Algorithm 4 Condat-Vu

y, A, Γ, λ, N
(τ, σ) s.t. $\frac{1}{\tau} - \sigma\|\Gamma^\top\|^2 \geq \frac{\|A\|^2}{2}$
$p_0 = 0$, $d_0 = 0$, $n = 0$
while $n < N$ **do**
    $p_{n+1} \leftarrow p_N - \tau A^\top(Ap_N - y) - \tau\Gamma d_N$
    $d_{n+1} \leftarrow \text{prox}_{\sigma\lambda(\|\cdot\|_1^*)}(d_N + \sigma\Gamma^\top(2p_{n+1} - p_N))$
    $n \leftarrow n + 1$
end while

Optimization and steps sizes learning.

We aim to compare the ability of Analysis and Synthesis to learn Γ and D, respectively. We focus on three constraints: Unit Norm, Convolutions, and Unit Norm Tight Frame. For **UN** and **Conv+UN**, the optimization is performed with projected gradient descent, and matrix multiplications are replaced with convolutions in `PyTorch` when necessary. For **UNTF**, the new point on the manifold is computed with a Cayley transform [Wen and Yin, 2013, Li et al., 2017]. We rely on full-batch gradient descent and line search for the sake of precision and robustness of our experiments, as done in [Ablin et al., 2019].

As stated before, we learn the dictionary and the step sizes in two steps to compensate for the imprecise resolution of the inner problem.

1. We learn the dictionary with fixed step sizes given by convergence conditions. For Synthesis, the step size is equal to $\frac{1}{L}$ where $L = \|AD\|^2$. For Analysis, we take $\sigma = 1$ and $\tau = \frac{1}{\frac{1}{2}\|A\|^2 + \sigma\|\Gamma^\top\|^2}$. Lipschitz constants are computed at each gradient step outside the network graph's scope.

2. Then, once convergence is reached, we jointly learn the step sizes and the dictionary. Both are still updated using gradient descent with line search to ensure stable optimization. This improves the convergence of the networks toward $Z_N^*(D)$ or $X_N^*(\Gamma)$.

Reconstruction on real data.

To illustrate the performance of unrolled algorithms for dictionary learning, we consider an unsupervised inpainting task. The original image is degraded with additive Gaussian noise to get a SNR of 10dB, and then a fraction of its pixels are randomly removed. We learn dictionaries and recover the image with unrolled algorithms using 20 iterations of FISTA or Condat-Vu. For UN and UNTF, the image is decomposed into patches, while convolutions can be applied directly to the whole image. Figure 3.9 displays the best results for each method. Synthesis-based methods achieve better image reconstruction compared to their Analysis counterpart. Convolutions improve the performances for Analysis, especially with small kernels – size 4 in that case, compared to 10 for Synthesis – emphasizing the importance of the selected constraints set. These results also demonstrate the ability of Synthesis to recover a good dictionary in the case of compressed or lacking information. Finally, Synthesis leads to more realistic reconstructions than TV-based methods. Fig-

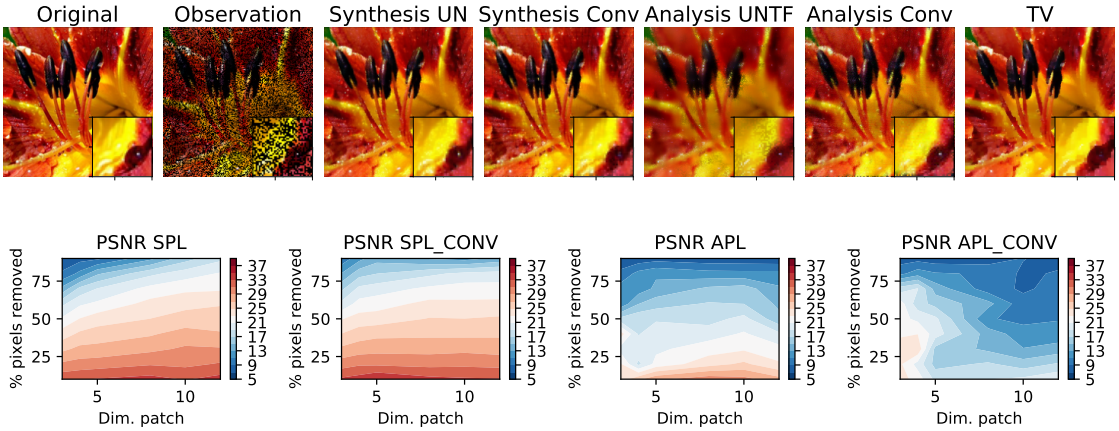

Figure 3.9: (top) *Inpainting on a color image of size* 200×200*, with 50% lacking pixels. Each RGB channel is processed separately. PSNR: Synthesis UN 33.4dB; Synthesis Conv 33.3dB; Analysis UNTF 29.6dB; Analysis Conv 30.8dB; TV 32.7 dB.* (bottom) *Shows the maximal PSNR obtained for an inpainting task on a grayscale image of size* 128×128*, depending on the dimension of the patches and the percentage of pixels removed. Several values of* λ *and the number of atoms are evaluated for several runs. 20 iterations are unrolled in this example. Synthesis and Analysis with UNTF constraint perform better with large patches, while Analysis with convolutions is better conditioned with small kernels. Synthesis performs better in general, even with a significant rate of lacking pixels.*

ure 3.9 also provides additional results for a gray-level image with different rates of lacking pixels and several patch dimensions. Analysis with convolutions seems to only work with small kernels, whereas Synthesis leads to good performance with larger patches.

The ability of gradient descent to find adequate local minima strongly depends on the structure of the problem. To quantify this, we evaluate the variation of PSNR depending on the percentage of lacking pixels for 50 random initializations in the context of convolutional dictionary learning. Figure 3.10 shows that Synthesis is more robust to random initialization, and almost all local minima are similar in terms of reconstruction quality. On the contrary, Analysis suffers from a significant number of poor local minima, and the reconstruction quality highly depends on the initialization. We propose to study the loss landscape for Analysis and Synthesis with the help of visualization techniques presented in Li et al. [2018]. The 3D landscape is displayed in Figure 3.10 using the Python library K3D-Jupyter[2]. We also compare the shapes of local minima in 1D by computing the normalized values of the loss along a line between two local minima. These representations of the loss landscape clearly show that Synthesis is much smoother than its Analysis counterpart and the different minima have similar performances in this case. Synthesis locally behaves like a convex function, while Analysis landscape is much steeper with poor local minima.

Dictionary recovery on synthetic data.

We now evaluate the dictionary recovery performance of unrolled algorithms on synthetic

[2]Package available at https://github.com/K3D-tools/K3D-jupyter

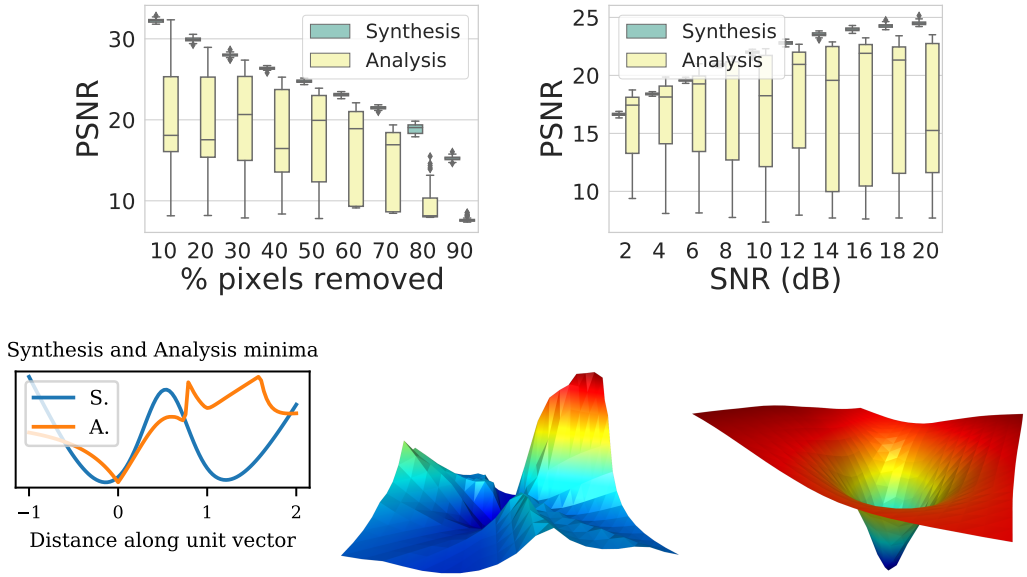

Figure 3.10: (top) *PSNR depending on the rate of lacking pixels without noise* (left) *and depending on the SNR (dB) with 50% pixels removed for 50 random initializations with convolutions on a grayscale image of size 128 ×.* (bottom) *Loss landscapes in 1D and 2D. Analysis* (center) *is poorly conditioned compared to Synthesis* (right) *and suffers from bad local minima due to its high sensitivity to initialization.*

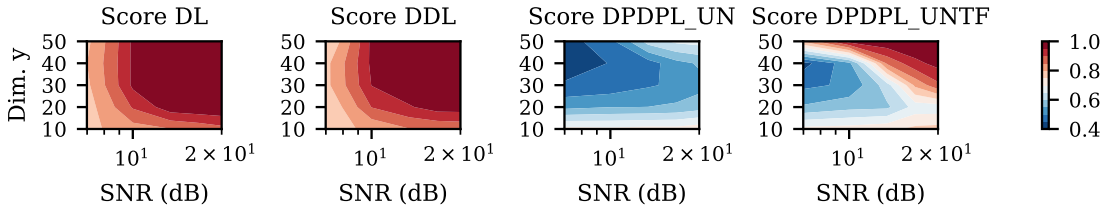

Figure 3.11: Dictionary recovery in the context of denoising depends on the SNR and the dimension of y. DPDPL performs well only with the UNTF constraint when the dictionary is almost square.

data. For Synthesis, a sparse code z is drawn from a Bernoulli Gaussian distribution of mean 0, variance σ^2, and Bernoulli parameter p. The signal is computed as $D_{ref}z$. For Analysis, we adopt the data generation process proposed in Elad et al. [2007]. The sparsity s of the signal is drawn from a Bernoulli distribution of parameter p, and $L - s$ rows of $\Gamma_{ref}^{\top}$ are chosen such that the corresponding sub-matrix is full-rank.

The generated signal is $u - (\Gamma_{ref}^{\top})^{\dagger}\Gamma_{ref}u$ with $u \sim \mathcal{N}(0, \sigma^2)$.

- Denoising. The data are generated with a UNTF dictionary of 50 atoms. We compare the maximal score for several values of λ and depth (between 5 and 50 iterations) depending on the Signal to Noise Ratio (SNR) and the dimension of the observation. The SNR is defined as $10\log_{10}(\frac{\sigma^2}{\sigma_b^2})$ where σ_b^2 is the variance of the noise. We assume that the number of atoms is known, and the results are provided in Figure 3.11. DDL behaves like standard Dictionary Learning, implemented in scikit-learn [Pedregosa et al., 2012]. The experiment confirms that UN does not work. Moreover, UNTF

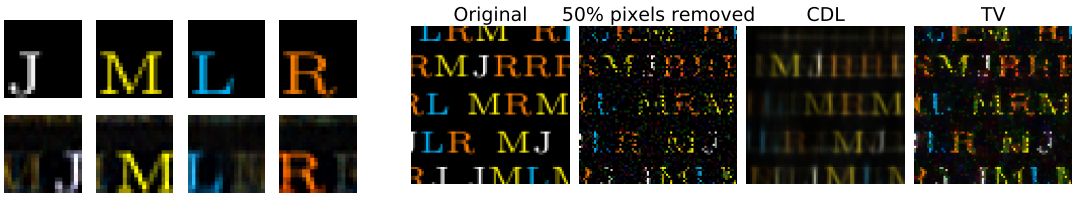

Figure 3.12: Inpainting task on a noisy image generated using seven letters {J, M, L, R} with 50% of pixels removed. DDL recovers a dictionary (left) and the image (right).

works only when the measurement dimension is close to the dimension of the signal and for a large SNR.

- Compressed sensing. The data are generated with an orthogonal dictionary of size 50 in a compressed sensing scenario. The results are presented in Table 3.1. DDL outperforms DPDPL, especially when the size of the measurements decreases. Analysis seems poorly conditioned compared to Synthesis, and the UNTF constraint is not sufficient to recover the orthogonal prior. Compared to other regularization techniques in inverse problems, the main interest of dictionary learning is to provide meaningful insights into the data. This can be achieved with the method studied in this paper, but the results depend on the structure of the data.

Convolutional dictionary learning with Synthesis.

We generate a text image from four letters {J, M, L, R}, and degrade it by removing pixels and adding noise. We compare the quality of reconstruction of TV and DDL in Figure 3.12, for a dictionary of 30 atoms of size 20×20. TV cannot adapt to the structure of the data, while DDL with convolutions learns a dictionary of patterns from the measurements and recovers the image simultaneously. The quality of the recovery highly depends on the amount of available information. Figure 3.13 (left) shows the average recovery score of CDL with initialization from patches of the image. The dictionary can be well-recovered for a large fraction of missing pixels, given that the SNR is reasonable.

Convolutional dictionary learning with Analysis.

Analysis is not adapted to recover patterns but can be applied to signals generated from recurrent relations such as Partial Differential Equations (PDEs) as done in auto-regressive models or textures. For instance, Analysis has been used in Kitić et al. [2015] to regularize inverse problems involving physical signals with PDEs. In fact, a discrete scheme from an ODE is equivalent to a convolution. Let's consider the function $x : t \to A\sin(\omega t)$. It satisfies $\ddot{x} + \omega^2 x = 0$, and the associated Euler discretization is $x_{n+1} + (dt^2\omega^2 - 2)x_n + x_{n-1} = 0$,

Table 3.1: Orthogonal dictionary recovery for compressed sensing with SNR = 20dB

Algo	dim. y = 50	dim. y = 40	dim. y = 30	dim. y = 20	dim. y = 10
DDL	0.98	0.89	0.76	0.61	0.36
DPDPL	0.96	0.78	0.57	0.43	0.37

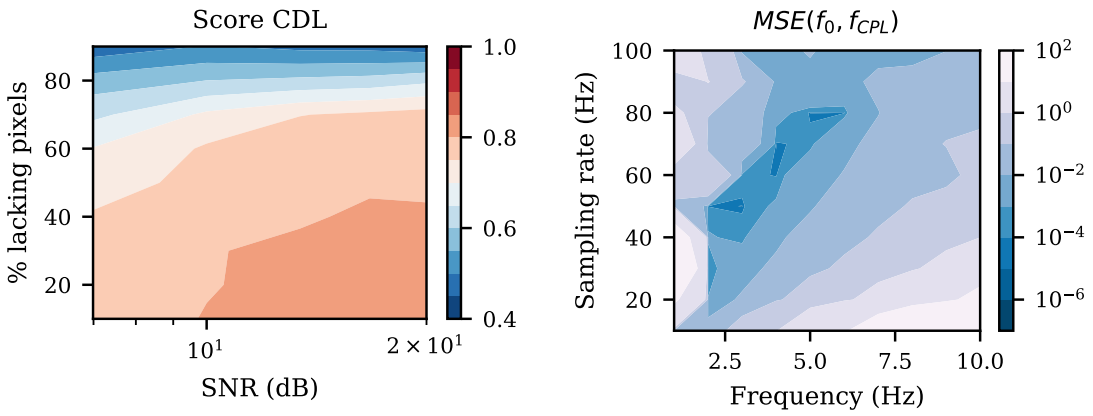

Figure 3.13: Average recovery score in Deep CDL depending VS SNR (dB) and rate of lacking pixels (%) (left). $MSE(f_0, f_{CPL})$ VS sampling rate (Hz) and signal frequency (Hz) (right).

where dt is the step size. In Figure 3.13, we measure the ability of convolutional dictionary learning with Analysis to recover a filter close to $[1, dt^2\omega^2 - 2, 1]$, which corresponds to this discretization, by estimating the frequency $f_{CPL} = \frac{\omega}{2\pi}$ given noisy data. The algorithm is successful when the sampling rate is high enough. However, the question of kernel identifiability arises: while the algorithm denoises the signal, it fails to recover meaningful kernels in 2D without additional constraints.

Conclusion

Prior knowledge of the signal plays a key role in inverse problems resolution, especially without access to ground truth data. In the context of unsupervised inverse problems, this work showed that prior learning techniques, as Dictionary Learning, achieve good results only with multiple operators or appropriate constraints in the model. When the operator is too ill-conditioned, which is typically the case in deblurring, the prior knowledge should compensate for the lack of information, and exclusively relying on convolutions is not enough.

Moreover, even though unrolled algorithms perform well on various kinds of inverse problems with a small number of layers or iterations, their behavior highly depends on the structure of the optimization problem they solve.

Chapter 4

A study of Plug and Play with Unrolling for inverse problems

This work was carried out in collaboration with Audrey Repetti, with the help of Mathieu Dagréou.

A common approach to solve linear inverse problems is to define the solution $X^\star$ to be a minimizer of a penalized least squares objective, i.e., to find

$$X^\star \in \operatorname*{argmin}_{X \in \mathbb{R}^N} \frac{1}{2}\|AX - Y\|^2 + p(X), \tag{4.1}$$

where $p\colon \mathbb{R}^N \to (-\infty, +\infty]$ is a convex, lower semi-continuous, proper function, corresponding to a penalization term, incorporating prior information on the target image. In particular, methods to efficiently summarize the observations structure by leveraging sparsity have been extensively studied [Elad, 2010b]. In this case, the prior p may be taken as a regularization term which promotes sparsity of the signal in a well-chosen representation space. For instance, this function may involve Fourier or Wavelet transforms [Mallat, 2008] or regularizations based on Total Variation [Chambolle et al., 2010]. However, modern resolution approaches rely on prior learning, *i.e.* learning the structure of the signal from a clean dataset in a supervised setting. One successful way to do so is to rely on Plug-and-Play algorithms.

A standard method to solve Equation 4.1 is to use a forward-backward (FB) algorithm – also known as proximal gradient descent [Combettes and Wajs, 2005]. This scheme alternates at each iteration between a gradient step on the differentiable least squares function, and a proximal step on the non-smooth function p. This algorithm reads

$$\begin{aligned} &\text{for } k = 0, 1, \dots \\ &\quad \lfloor\ X_{k+1} = \operatorname{prox}_{\tau_k p}\big(X_k - \tau_k A^*(AX_k - Y)\big), \end{aligned} \tag{4.2}$$

where, for every $k \in \mathbb{N}$, $\tau_k > 0$ is a step size parameter, A^* denotes the adjoint operator of

A, and prox_p denotes the proximity operator of p defined as

$$(\forall V \in \mathbb{R}^N) \quad \text{prox}_p(V) = \underset{X \in \mathbb{R}^N}{\text{argmin}}\, p(X) + \frac{1}{2}\|X - V\|^2. \tag{4.3}$$

An important comment is that that the proximity operator of p can be interpreted as a maximum *a posteriori* estimate for a Gaussian denoising problem. Leveraging this fact, multiple works in the past decades have proposed to replace proximity operators related to penalization terms by more powerful denoisers, leading to plug-and-play (PnP) techniques [Brifman et al., 2016]. Among these works, we can distinguish PnP involving hand-crafted priors (e.g., BM3D [Dabov et al., 2009]), and learned priors such as neural networks [Chan et al., 2016, Romano et al., 2017, Rick Chang et al., 2017] denoisers. In this context, the PnP version of Equation 4.2, dubbed FB-PnP, can be rewritten as

$$\begin{array}{l} \text{for } k = 0, 1, \ldots \\ \quad \lfloor\ X_{k+1} = f_\theta\big(X_k - \tau_k A^*(AX_k - Y)\big), \end{array} \tag{4.4}$$

where $f_\theta \colon \mathbb{R}^N \to \mathbb{R}^N$ is a *good* denoiser. The main challenge of building f_θ is to find a trade-off between convergence of the algorithm, to guarantee that it produces a converging sequence of estimates $(X_k)_{k\in\mathbb{N}}$, and reconstruction performance. Thus, finding guarantees on denoisers is a central problem in the literature [Ryu et al., 2019].

Analysis and Synthesis dictionaries have been widely used in signal processing and inverse problems in the past decades due to their ability to provide simple representations of a family of signals. Recently, it has been proposed in Repetti et al. [2022] to define the operator f_θ in Equation 4.4 as an unrolled proximal algorithm designed to solve the Analysis formulation, *i.e.* Equation 1.16. This technique has been shown to perform at least as well as FB-PnP algorithms involving advanced denoising networks, while reducing drastically the number of learned parameters. For instance, compared to classical DnCNN architectures Zhang et al. [2017], the proposed unfolded proximal network has 20 times less learnable parameters, while comparing to DRU-Net architectures Zhang et al. [2021], it has 400 times less parameters. In terms of reconstruction quality, the FB-PnP algorithm with unfolded proximal network outperformed the FB-PnP with DnCNN for deconvolution problems, and performed as well as using a DRU-Net.

The approach adopted in Repetti et al. [2022] aims to build a denoiser f_θ such that $f_\theta(V) \approx X^\star$, where $X^\star \in \mathbb{R}^N$ is an estimate of an original unknown image $\overline{X} \in \mathbb{R}^N$ from a noisy observation $V = \overline{X} + \upsilon W$, for $\upsilon > 0$ and $W \in \mathbb{R}^N$ a realization of an i.i.d. standard normal random variable. As explained in the previous section, a standard approach to obtain such an estimate is to rely on the proximity operator as defined in Equation 4.3. Hence, we can define

$$f_\theta(V) \approx \text{prox}_p(V) \ , \tag{4.5}$$

where p is an Analysis sparsity prior, i.e.

$$(\forall X \in \mathbb{R}^N) \quad p(X) = \lambda g(\Gamma^* X) \ , \tag{4.6}$$

where $\lambda > 0$ is a regularization parameter, $\Gamma^* \colon \mathbb{R}^N \to \mathbb{R}^S$ a sparsifying linear operator, and $g \colon \mathbb{R}^S \to (-\infty, +\infty]$ is a convex, lower semi-continuous, proper function (e.g., an ℓ_1 norm). Let V be an input image to denoise. Repetti et al. [2022] propose a denoiser f_θ

which is built by unrolling N iterations of a dual FB algorithm [Combettes and Pesquet, 2011], as follows

$$\begin{aligned} &\text{for } \ell = 0, \dots, L-1, \\ &\quad U_{\ell+1} = (I - \operatorname{prox}_{\lambda\sigma_\ell^{-1} g})\big(U_\ell + \Gamma_{\ell,\theta}{}^*(V - \sigma_\ell \Gamma_{\ell,\theta} U_\ell)\big) \\ &f_{L,\theta}(V) = V - \sigma_L \Gamma_{L,\theta} U_L \ , \end{aligned} \tag{4.7}$$

where for every $l \in \mathbb{N}$, $\sigma_l > 0$ is a step size parameter. The main interest of this approach lies in the convergence properties of the denoiser. As a matter of fact, when the $(\Gamma_{\ell,\theta})_{1\leq\ell\leq L}$ are the same for every ℓ and L tends to infinity, the left and right terms in Equation 4.5 are equal. Indeed, we can deduce the following convergence result from Combettes and Pesquet [2011].

Let $V \in \mathbb{R}^N$, and $L \in \mathbb{N}^*$. Let $(U_\ell)_{1\leq\ell\leq L}$ be generated by Equation 4.7 where, for every $\ell \in \{1, \dots, L\}$, $\Gamma_{\ell,\theta} = \Gamma_\theta \colon \mathbb{R}^S \to \mathbb{R}^N$ is a linear operator, and $\sigma_\ell \in \big(0, 2/\|\Gamma_\theta\|^2\big)$. Then

$$\operatorname{prox}_p(V) = \lim_{L\to\infty} f_{L,\theta}(V) = \lim_{L\to\infty} V - \sigma\Gamma_\theta U_L. \tag{4.8}$$

Thus, the Plug and Play algorithm converges toward a fixed point.

Unrolled Sparse Coding as a denoiser in Plug-and-Play. The unfolded network described above is built on an iterative algorithm which solves the convex minimization problem

$$\min_{X\in\mathbb{R}^N} \frac{1}{2}\|X - V\|^2 + \lambda g(\Gamma^* X) \ , \tag{4.9}$$

where $V \in \mathbb{R}^N$ is a noisy observation of an original unknown signal $\overline{X} \in \mathbb{R}^N$. The problem described in Equation 4.9 corresponds to the Analysis formulation, and involves the usage of a linear operator Γ, also known as a (sparsifying) dictionary. Such a prior assumes that there exists a forward transform $\Gamma \colon \mathbb{R}^S \to \mathbb{R}^N$, which makes the signal sparse. In the case where the dictionary Γ does not change over the layers of the unfolded network, the denoising operator f_θ corresponds to an iterative algorithm which approximates the proximal operator of $p(X) = \lambda g(\Gamma^* X)$.

On the same principle, it is possible to replace the denoiser built from the Analysis formulation by a denoiser built from the Synthesis formulation. This time, the unfolded network is inspired from a proximal gradient descent that solves

$$\min_{Z\in\mathbb{R}^S} \frac{1}{2}\|DZ - V\|_2^2 + \lambda g(Z) \ . \tag{4.10}$$

In other words, the goal is to recover sparse codes $Z \in \mathbb{R}^S$ from a dictionary $D \in \mathbb{R}^{N\times S}$. Then, if Z^* is a solution to the problem described in Equation 4.10, $X^* = DZ^*$ corresponds to an estimate of $\overline{X}$ given V. As before, the denoiser f_θ is parameterized by a dictionary D, and solves the denoising problem with a few iterations of an iterative algorithm.

Taking f_θ as an unrolled sparse coding algorithm (Synthesis), or a dual forward backward algorithm (Analysis), it is possible to learn either one or several dictionaries in order to

denoise the data in a supervised setting. Concretely, given a set of clean data $(X_i)_{1\leq i\leq N_X}$, the unrolled network f_θ is trained by solving the minimization problem

$$\min_\theta \frac{1}{N_X}\sum_{i=1}^{N_X} \mathcal{L}(f_\theta(X_i+\epsilon_i), X_i) \ , \tag{4.11}$$

where $(\epsilon_i)_{1\leq i\leq N_X}$ is a set of noises, and $\mathcal{L}$ is a loss function. In this case θ corresponds to the dictionary or dictionaries to be learned. Then, the unrolled network can be plugged in an iterative algorithm to solve an inverse problem, as done by Repetti et al. [2022].

Contributions of chapter 4. Besides reconstruction performance, the interest of Plug-and-Play algorithms is to find mathematical guarantees on their convergence behaviors and on the set of solutions. In particular, the usage of approximate proximal operators and unrolled algorithms in Repetti et al. [2022] is motivated by the fact that we can explain why the algorithm converges – as described in Equation 4.8 – and why this particular kind of architectures works well in practice on natural signals, by making use of the literature on Dictionary Learning and proximal algorithms.

Analysis and Synthesis Dictionary Learning and sparse coding have been extensively studied in the last 20 years, from optimization algorithms to reconstruction properties. In this paper, we develop ideas to bridge the gap between unrolled denoisers in Plug and Play and the literature on sparse coding for inverse problems.

More specifically, we first study to what extent Dictionary Learning and sparse coding, being Synthesis or Analysis, are well-suited as denoising techniques in Plug-and-Play. Then, we show that there is a strong link between modern Plug-and-Play algorithms leveraging unfolded networks and classical optimization approaches based on Dictionary Learning. Finally, we present numerical results highlighting the good trade-off between convergence guarantees and reconstruction performance.

4.1 . Why denoisers based on Dictionary Learning are adapted to Plug-and-Play

Plug-and-Play algorithms were designed from the observation that iterative optimization schemes involving proximal gradient descent perform well in inverse problem resolution. Let f_θ be a denoising operator, the related PnP algorithm reads

$$\begin{array}{l} \text{for } k = 0, 1, \dots \\ \quad \lfloor \ X_{k+1} = f_\theta\big(X_k - \tau_k A^*(AX_k - Y)\big) \ . \end{array} \tag{4.12}$$

Classical resolution methods rely on proximal operators of well-chosen regularization functions p and on proximal gradient descent – also called forward backward algorithms. In particular, proximal operators of functions involving sparsifying regularizations have been widely used in signal processing, for instance with Total Variation or Analysis dictionaries combined to the ℓ_1 norm. These methods are related to Plug-and-Play algorithms in the sense that they correspond to taking $f_\theta = \text{prox}_p$, where p is the penalization contained in the loss. In this specific case where the denoiser f_θ is not learned from data but built from

expert knowledge on the signal, PnP just reduces to a proximal gradient descent, *i.e.*

$$\begin{aligned} &\text{for } k = 0, 1, \dots \\ &\quad \lfloor \; X_{k+1} = \text{prox}_p \left(X_k - \tau_k A^*(AX_k - Y) \right) \; . \end{aligned} \tag{4.13}$$

PnP algorithms output converging sequences $(X_k)_{k \in \mathbb{N}}$ if the denoiser f_θ is firmly non expansive, *i.e.* when for all $x, y \in \mathbb{R}^N$

$$\|f_\theta(x) - f_\theta(y)\| \leq \langle (x - y), (f_\theta(x) - f_\theta(y)) \rangle \; , \tag{4.14}$$

as a consequence of fixed point theory [Bauschke et al., 2011]. Thus, denoisers that verify this property lead to convergence of the algorithm. As proximal operators are known to be firmly non expansive, they happen to be operators of choice for obtaining good results.

Analysis. Going back to Analysis Dictionary Learning, solving the sparse coding optimization problem described is equivalent to computing the proximal operator of the regularization function $p(X) = \lambda g(\Gamma^* X) = g_\lambda(\Gamma^* X)$. Thus, for every dictionary Γ, including ones that are learned from data, the denoising operator that consists of finding the minimum of the Analysis optimization problem is a proximal operator.

PROPOSITION 4.1.1. *Let $\Gamma \in \mathbb{R}^{N \times M}$, and let f_Γ be the Analysis denoising operator parameterized by the dictionary Γ. Then f_Γ^A is a proximal operator,* i.e.

$$\begin{aligned} f_\Gamma^A(V) &= \underset{X \in \mathbb{R}^N}{\text{argmin}} \frac{1}{2}\|X - V\|^2 + g_\lambda(\Gamma^* X) \\ &= \text{prox}_{g_\lambda(\Gamma^* \cdot)}(V) \; , \end{aligned}$$

and the related PnP algorithm

$$\begin{aligned} &\textit{for } k = 0, 1, \dots \\ &\quad \lfloor \; X_{k+1} = \text{prox}_{g_\lambda(\Gamma^* \cdot)} \left(X_k - \tau_k A^*(AX_k - Y) \right) \end{aligned}$$

produces a converging sequence.

Proof 4.1.1

Immediate from the definition of the prox.

The result from Proposition 4.1.1, which is a result of the study of proximal operators [Bauschke et al., 2011], shows that once we have access to a proper Analysis dictionary Γ, then the denoising procedure based on the Analysis formulation is a good candidate to be plugged in PnP algorithms. We now present a new result which establishes that the same is true for Synthesis.

Synthesis. As opposed to its Analysis counterpart, Synthesis sparse coding can't be directly expressed as a proximal operator, but we show here that it is. Given a dictionary $D \in \mathbb{R}^{M \times N}$, the idea of Synthesis is first to compute a decomposition – also called sparse codes – V^* in the redundant basis induced by D, and then recover the signal with the product $X^* = DV^*$. To prove our results, we rely on the notion of Fenchel convex conjugate. We recall its definition in what follows.

DEFINITION. *Let $\mathbb{E}$ be a euclidean vector space with inner product denoted $\langle \cdot, \cdot \rangle$. Let $f : \mathbb{E} \mapsto \mathbb{R} \cup \{-\infty, \infty\}$ be a function defined on $\mathbb{E}$ and taking values on the extended real line. The convex conjugate (or Fenchel conjugate) denoted f^* is defined as*

$$f^*(y) = \sup_{x \in \mathbb{E}} \left(\langle y, x \rangle - f(x) \right) \ .$$

Then we show in Proposition 4.1.2 that the denoising procedure based on Synthesis actually corresponds to a proximal operator.

PROPOSITION 4.1.2. *Let $D \in \mathbb{R}^{M \times N}$, and let f_D^S be the Synthesis denoising operator parmeterized by the dictionary D. Then f_D^S is a proximal operator,* i.e.

$$\begin{aligned} f_D^S(V) &= D \operatorname*{argmin}_{Z \in \mathbb{R}^S} \frac{1}{2} \|DZ - V\|^2 + g_\lambda(Z) \\ &= \operatorname{prox}_{(g_\lambda^* \circ D^*)^*}(V) \ , \end{aligned}$$

and the related PnP algorithm

for $k = 0, 1, \ldots$
$\quad X_{k+1} = \operatorname{prox}_{(g_\lambda^* \circ D^*)^*} \left(X_k - \tau_k A^*(AX_k - Y) \right)$

produces a converging sequence.

Proof 4.1.2

Let $V^* = DZ^*$ where

$$Z^* = \operatorname*{argmin}_{Z \in \mathbb{R}^S} \frac{1}{2} \|V - DZ\|_2^2 + \lambda g(Z) \ . \tag{4.15}$$

The associated dual problem to Equation 4.15 reads (Fenchel Rockafellar)

$$U^* = \operatorname*{argmin}_{U \in \mathbb{R}^N} \frac{1}{2} \|V - U\|_2^2 + (\lambda g)^*(D^* U) = \operatorname{prox}_{(\lambda g)^* \circ D^*}(V) \ . \tag{4.16}$$

In addition we have [Bauschke et al., 2011][Theorem 19.1]

$$-U^* \in \partial_{\frac{1}{2}\|V - \cdot\|_2^2}(DZ^*) = \{DZ^* - V\} \ . \tag{4.17}$$

Therefore, Moreau's identity leads to

$$V^* = DZ^* = (I - \operatorname{prox}_{(\lambda g)^* \circ D^*})(V) = \operatorname{prox}_{((\lambda g)^* \circ D^*)^*}(V) \ . \tag{4.18}$$

The convergence of the sequence $(X_k)_{k \in \mathbb{N}}$ is then obtained using results from Bauschke et al. [2011].

Proposition 4.1.2 shows that the process combining Synthesis sparse coding to compute Z and the signal reconstruction obtained with the product DZ actually corresponds to the proximal operator of the function $(g_\lambda^* \circ D^*)^*$. Thus, plugging it into a PnP algorithm for reconstruction produces a converging sequence. As an example, let's consider the case of the Lasso, *i.e.* $g_\lambda = \lambda \|\cdot\|_1$. We have $(g_\lambda^* \circ D^*)^* = \iota^*_{\mathcal{F}_\lambda(D)}$ where $\mathcal{F}_\lambda(D) = \{x \mid \|D^* x\|_\infty \leq \lambda\}$ is a convex polytope. Note that this result is only true when considering the whole process,

as Synthesis sparse coding alone does not correspond to a proximal operator.

For now, we highlighted that both Analysis and Synthesis denoisers are actually proximal operators and that this is why they are well-suited to be used in a Plug and Play algorithm. The main challenge is that the evaluation of these particular proximal operators – for which we do not have access to a closed-form – requires iterative optimization schemes that may take a long time to converge. This is why a recent line of work is turning to unfolded networks, for which the denoising procedure only takes a few iterations. Thus, the question we address in the following is: to what extent are PnP algorithms with unfolded networks based on Dictionary Learning linked to classical methods for inverse problems resolution ?

4.2 . Unrolled dictionary learning for Plug and Play algorithms

In this section, we propose to study the behavior of unrolled algorithms that solve Equation 4.9 and Equation 4.10 when used as denoisers plugged in the PnP-FB algorithm. Let $\overline{X} \in \mathbb{R}^N$ be an original unknown image, and $V = \overline{X} + \upsilon W$ be a noisy observation of it, where $\upsilon > 0$ and $W \in \mathbb{R}^N$ is a realization of an i.i.d. standard normal random variable. We aim to build a denoiser f_θ such that $f_\theta(V) \approx X^\star$, where $X^\star \in \mathbb{R}^N$ is an estimate of $\overline{X}$. We will do so based either on the algorithm that solves the Analysis formulation from Equation 4.9 or on the one that solves the Synthesis formulation from Equation 4.10.

Analysis. We first focus on Analysis, and build an operator $f^A_{L,\lambda}$ corresponding to an unfolded iterative scheme used to approximate the proximal operator prox_p where $p = \lambda g \circ \Gamma^*$. In our notation, we drop index θ representing the learned parameters – the dictionary Γ in our case – and instead we make $f^A_{L,\lambda}$ dependent on the regularization parameter λ. As previously mentioned, such a denoiser can be built by unrolling a fixed number $L \in \mathbb{N}^*$ of dual-FB iterations

$$
\begin{aligned}
&\text{let } U_0 \in \mathbb{R}^S, \\
&\text{for } \ell = 0, \dots, L-1, \\
&\quad U_{\ell+1} = \mathrm{prox}_{\sigma(\lambda g)^*}\left(U_\ell + \sigma\Gamma^*(V - \Gamma U_\ell)\right) \\
&f^A_{L,\lambda}(V, U_0) = V - \Gamma U_L,
\end{aligned}
\tag{4.19}
$$

where $\sigma > 0$.

In the following proposition, we show that the operator $f^A_{L,\lambda}$ defined in Equation 4.19 can be formulated as a feed-forward network.

PROPOSITION 4.2.1. *Let $L \in \mathbb{N}^*$, $\sigma > 0$, and*

$$\widetilde{f}^A_{L,\lambda} \colon \mathbb{R}^S \times \mathbb{R}^N \to \mathbb{R}^S \times \mathbb{R}^N$$

be the operator defined as L compositions of operator

$$\begin{aligned} A_\lambda \colon \mathbb{R}^S \times \mathbb{R}^N &\to \mathbb{R}^S \times \mathbb{R}^N \\ (U, V) &\mapsto \operatorname{prox}_{\sigma(\lambda g)^*}\Big(\big(I - \sigma\Gamma^*\Gamma\big)U + \sigma\Gamma^* V\Big). \end{aligned}$$

Let $(V, U_0) \in \mathbb{R}^S \times \mathbb{R}^N$. Then $f^A_{L,\lambda}(V, U_0) = V - \Gamma \widetilde{f}^A_{L,\lambda}(V, U_0)$, where $f^A_{L,\lambda}$ is defined in Equation 4.19.

Proof 4.2.1

Directly obtained by reformulation of Equation 4.19.

Note that that this operator converges toward the proximal map $\operatorname{prox}_{\lambda g}$, as described in Equation 4.8. In particular, we can deduce the following result regarding the convergence of this algorithm toward a solution to the Analysis formulation.

COROLLARY 4.2.2. *Let $(U_\ell)_{1 \le \ell \le L}$ be generated by Equation 4.19, for $L \in \mathbb{N}^*$ and $\sigma \in \big(0, 2/\|\Gamma\|^2\big)$. Then, $f^A_{L,\lambda}(V, U_0)$ converges to a solution to*

$$\operatorname*{argmin}_{X \in \mathbb{R}^N} \frac{1}{2}\|X - V\|^2 + g_\lambda(\Gamma^* X)$$

when $L \to +\infty$, for any choice of $\Gamma \colon \mathbb{R}^S \to \mathbb{R}^N$, $\lambda > 0$, and $(V, U_0) \in \mathbb{R}^N \times \mathbb{R}^S$.

Proof 4.2.2

Direct consequence of the result described in Equation 4.8.

When plugging $f^A_{L,\lambda}$ into algorithm Equation 4.4, this leads to

$$\begin{aligned} &\text{for } k = 0, 1, \dots \\ &\quad \left\lfloor \begin{aligned} V_k &= X_k - \tau A^*(AX_k - Y), \\ U_{k+1} &= \widetilde{f}^A_{L,\lambda\tau}(U_k, V_k), \\ X_{k+1} &= V_k - \Gamma U_{k+1}, \end{aligned} \right. \end{aligned} \tag{4.20}$$

where $\tau > 0$ is the stepsize of the FB algorithm. Using results from Combettes and Wajs [2005], we can show the following convergence result.

THEOREM 4.2.3. *Let $(X_k)_{k \in \mathbb{N}}$ be generated by the algorithm in Equation 4.20, where $\tau \in \big(0, 2/\|A\|^2\big)$. Then $(X_k)_{k \in \mathbb{N}}$ converges to a solution of*

$$\operatorname*{argmin}_{X \in \mathbb{R}^N} \frac{1}{2}\|Y - AX\|^2 + g_\lambda(\Gamma^* X)$$

when using $f^A_{\infty,\lambda} = \lim_{L \to \infty} f^A_{L,\lambda}$.

Proof 4.2.3

When $L \to \infty$, the algorithm in Equation 4.20 reduces to a proximal gradient descent.

These two results only hold when $L \to \infty$, i.e. if $f^A_{L,\lambda}$ has an infinite number of layers. This corresponds to the case where we have access to the true value of the proximal operator related to Analysis, as explained in the previous section. In the context of deep dictionary learning however, the number of layers is finite, and the network often comprises only a few dozens of layers. We propose to investigate to what extent this impacts the convergence and the nature of the final solution. Note that in Equation 4.20, the unrolled network benefits from a warm restart on variable U_k. In Proposition 4.2.4, we show that unrolling combined to warm restart of the dual variable enables Equation 4.20 to converge to a fix point when $L = 1$.

PROPOSITION 4.2.4. *When $L = 1$, the algorithm described in Equation 4.20 converges to a solution to*

$$\operatorname*{argmin}_{X \in \mathbb{R}^N} \frac{1}{2}\|Y - AX\|^2 + g_\lambda(\Gamma^* X) \ ,$$

given that $\sigma < \frac{2}{\|\Gamma\|^2}$ and $\tau < \frac{1}{\|A\|^2}$. More specifically, it produces a sequence $(X_k, U_k)_{k\in\mathbb{N}}$ that correspond to the sequence output by the algorithm described in Loris and Verhoeven [2011][Section 6 (40)], i.e.

$$\begin{aligned}
\tilde{U}_k &= \operatorname{prox}_{(\lambda g)^* \frac{\sigma}{\tau}} \left(\tilde{U}_{k-1} + \frac{\sigma}{\tau}\Gamma^*(X_{k-1} - \tau A^*(AX_{k-1} - Y) - \tau\Gamma\tilde{U}_{k-1})\right) \\
X_k &= X_{k-1} - \tau A^*(AX_{k-1} - Y) - \tau\Gamma\tilde{U}_k \ ,
\end{aligned}$$

up to rescaling $\tilde{U}_k = \frac{1}{\tau}U_k$.

Proof 4.2.4

We have

$$\begin{aligned}
X_k &= V_{k-1} - \Gamma U_k && (4.21)\\
&= X_{k-1} - \tau A^*(AX_{k-1} - Y) - \Gamma U_k && (4.22)
\end{aligned}$$

and

$$\begin{aligned}
U_k &= \operatorname{prox}_{\sigma(\lambda\tau g)^*} \left(U_{k-1} - \sigma\Gamma^*\Gamma U_{k-1} + \sigma\Gamma^* V_{k-1}\right) && (4.23)\\
&= \operatorname{prox}_{\sigma(\lambda\tau g)^*} \left(U_{k-1} - \sigma\Gamma^*\Gamma U_{k-1} + \sigma\Gamma^*(X_{k-1} - \tau A^*(AX_{k-1} - Y))\right) && (4.24)\\
&= \operatorname{prox}_{\sigma(\lambda\tau g)^*} \left(U_{k-1} + \sigma\Gamma^*(X_{k-1} - \tau A^*(AX_{k-1} - Y) - \Gamma U_{k-1})\right) && (4.25)
\end{aligned}$$

Thus the optimization scheme becomes

$$\begin{aligned}
U_k &= \operatorname{prox}_{\sigma(\lambda\tau g)^*} \left(U_{k-1} + \sigma\Gamma^*(X_{k-1} - \tau A^*(AX_{k-1} - Y) - \Gamma U_{k-1})\right) \\
X_k &= X_{k-1} - \tau A^*(AX_{k-1} - Y) - \Gamma U_k
\end{aligned} \qquad (4.26)$$

Moreover, Moreau's equality applied to a l.s.c function ϕ leads to [Combettes and Wajs, 2005][Section 2.5]

$$x = \operatorname{prox}_{\gamma\phi}(x) + \gamma \operatorname{prox}_{\frac{\phi^*}{\gamma}}(\frac{x}{\gamma}) \ , \qquad (4.27)$$

where $\gamma > 0$. Applying this result to $\phi = (\lambda\tau g)^*$ and $\gamma = \sigma$ leads to

$$\begin{aligned}
x &= \operatorname{prox}_{\sigma(\lambda\tau g)^*}(x) + \sigma \operatorname{prox}_{\frac{\lambda\tau g}{\sigma}}(\frac{x}{\sigma}) && (4.28)\\
\frac{x}{\sigma} &= \frac{1}{\sigma}\operatorname{prox}_{\sigma(\lambda\tau g)^*}(x) + \operatorname{prox}_{\frac{\lambda\tau g}{\sigma}}(\frac{x}{\sigma}) && (4.29)
\end{aligned}$$

On the same principle, applying this result to $\phi = \lambda g$ and $\gamma = \frac{\tau}{\sigma}$ leads to

$$x = \operatorname{prox}_{\frac{\tau}{\sigma}\lambda g}(x) + \frac{\tau}{\sigma}\operatorname{prox}_{(\lambda g)^* \frac{\sigma}{\tau}}(x\frac{\sigma}{\tau}) \tag{4.30}$$

$$\frac{x}{\sigma} = \operatorname{prox}_{\frac{\tau}{\sigma}\lambda g}(\frac{x}{\sigma}) + \frac{\tau}{\sigma}\operatorname{prox}_{(\lambda g)^* \frac{\sigma}{\tau}}(x\frac{1}{\tau}) \tag{4.31}$$

By combining these two equalities, we get that

$$\operatorname{prox}_{\frac{\tau}{\sigma}\lambda g}(\frac{x}{\sigma}) + \frac{\tau}{\sigma}\operatorname{prox}_{(\lambda g)^* \frac{\sigma}{\tau}}(x\frac{1}{\tau}) = \frac{1}{\sigma}\operatorname{prox}_{\sigma(\lambda\tau g)^*}(x) + \operatorname{prox}_{\frac{\lambda\tau g}{\sigma}}(\frac{x}{\sigma}) \tag{4.32}$$

$$\tau\operatorname{prox}_{(\lambda g)^* \frac{\sigma}{\tau}}(x\frac{1}{\tau}) = \operatorname{prox}_{\sigma(\lambda\tau g)^*}(x) \tag{4.33}$$

Therefore, the optimization scheme leading to U_k can be formulated as follows

$$U_k = \tau\operatorname{prox}_{(\lambda g)^* \frac{\sigma}{\tau}}(\frac{1}{\tau}U_{k-1} + \frac{\sigma}{\tau}\Gamma^*(X_{k-1} - \tau A^*(AX_{k-1} - Y) - \Gamma U_{k-1})) \tag{4.34}$$

and with the change of variable $\tilde{U}_k = \frac{1}{\tau}U_k$

$$\begin{aligned} \tilde{U}_k &= \operatorname{prox}_{(\lambda g)^* \frac{\sigma}{\tau}}\left(\tilde{U}_{k-1} + \frac{\sigma}{\tau}\Gamma^*(X_{k-1} - \tau A^*(AX_{k-1} - Y) - \tau\Gamma\tilde{U}_{k-1})\right) \\ X_k &= X_{k-1} - \tau A^*(AX_{k-1} - Y) - \tau\Gamma\tilde{U}_k \end{aligned} \tag{4.35}$$

which corresponds to the algorithm described in Loris and Verhoeven [2011][Section 6 (40)], and produces a sequence $(X_k, U_k)_{k\in\mathbb{N}}$ that converges to a solution of

$$\operatorname*{argmin}_{X\in\mathbb{R}^N} \frac{1}{2}\|Y - AX\|^2 + g_\lambda(\Gamma^* X)$$

with $\sigma < \frac{2}{\|\Gamma\|^2}$ and $\tau < \frac{1}{\|A\|^2}$.

Proposition 4.2.4 shows that taking $L = 1$ is equivalent to using a primal-dual algorithm, namely the Loris-Verhoeven algorithm introduced by Loris and Verhoeven [2011], and actually produces the same sequence of iterates. Moreover, the conditions on τ and σ are disjoint, as opposed to other primal dual algorithms like Chambolle-Pock [Chambolle and Pock, 2011] or Condat-Vu [Condat, 2013, Vu, 2013]. This fits well to a context where the denoiser is trained independently from the measurement operator A.

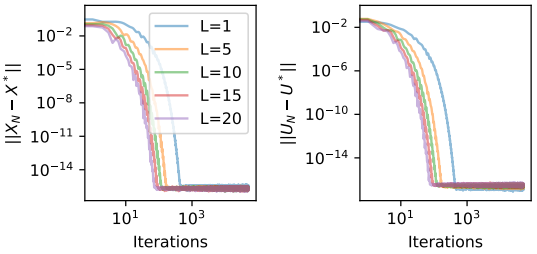

Figure 4.1: Convergence of primal and dual variables for different values of L. The number of layers does not change anything to the result of the algorithm when the dictionary is fixed.

We conjecture that taking $1 < L < \infty$ does not impact the convergence of the algorithm and leads to the same result. As demonstrated in Figure 4.1, taking different values of L does not change the nature of the solution with a fixed dictionary. Of course, training a network with more or less iterations may lead to different dictionaries. In the following,

we propose a proof of convergence in the case where $1 < L < \infty$ for a more restrictive scenario where the regularization function g is replaced by its Moreau envelope, making it smooth.

Synthesis. Now, we aim to build an operator $f^S_{L,\lambda}$ based on the Synthesis formulation, given in Equation 4.10. First, we note that if g is proximable, then Equation 4.10 can be solved with FB iterations. Hence, a denoiser can be built by unrolling a fixed number $L \in \mathbb{N}^*$ of FB iterations as follows

$$\begin{aligned} &\text{let } Z_0 \in \mathbb{R}^S, \\ &\text{for } \ell = 0, \dots, L-1, \\ &\quad Z_{\ell+1} = \operatorname{prox}_{\lambda\sigma g}\left(Z_\ell - \sigma D^*(DZ_\ell - V)\right) \\ &f^S_{L,\lambda}(V, Z_0) = DZ_L, \end{aligned} \tag{4.36}$$

where $\sigma > 0$.

In the following proposition, we show that operator $f^S_{L,\lambda}$ defined in Equation 4.36 can be formulated as a feedforward network.

PROPOSITION 4.2.5. *Let $L \in \mathbb{N}^*$, $\sigma > 0$, and*

$$\widetilde{f}^S_{L,\lambda} \colon \mathbb{R}^S \times \mathbb{R}^N \to \mathbb{R}^S \times \mathbb{R}^N$$

be the operator defined as L compositions of operator

$$\begin{aligned} B_\lambda \colon \mathbb{R}^S \times \mathbb{R}^N &\to \mathbb{R}^S \times \mathbb{R}^N \\ (V, Z) &\mapsto \operatorname{prox}_{\lambda\sigma g}\Big(\big(I - \sigma D^* D\big) Z_\ell + \sigma D^* V\Big). \end{aligned}$$

Let $(V, Z_0) \in \mathbb{R}^S \times \mathbb{R}^N$. Then $f^S_{L,\lambda}(V, Z_0) = D\widetilde{f}^S_{L,\lambda}(V, Z_0)$, where $f^S_{L,\lambda}$ is defined in Equation 4.36.

Proof 4.2.5

Directly obtained by reformulation of Equation 4.36.

First note that the operator A_λ defined for Analysis and the operator B_λ defined for Synthesis have the same structure, except that A uses the Fenchel conjugate of g in the prox, while B uses g. As for Analysis, we can deduce a convergence result when L tends to infinity.

COROLLARY 4.2.6. *Let $(Z_\ell)_{1 \le \ell \le L}$ be generated by Equation 4.36, for $L \in \mathbb{N}^*$ and $\sigma \in (0, 1/\|D\|^2)$. Then, $f^S_{L,\lambda}(V, Z_0)$ converges to a solution to the synthesis problem*

$$D \operatorname*{argmin}_{Z \in \mathbb{R}^S} \frac{1}{2}\|DZ - V\|^2 + g_\lambda(Z) \ ,$$

for any choice of $D : \mathbb{R}^S \to \mathbb{R}^N$, $\lambda > 0$, and $(V, Z_0) \in \mathbb{R}^N \times \mathbb{R}^S$.

Proof 4.2.6

Consequence of convergence properties of Forward Backward algorithms.

When plugging $f^S_{L,\lambda}$ into algorithm Equation 4.4, this leads to

$$\begin{array}{l} \text{for } k = 0, 1, \dots \\ \left\lfloor \begin{array}{l} V_k = X_k - \tau A^*(AX_k - Y), \\ Z_{k+1} = \widetilde{f}^S_{L,\lambda\tau}(Z_k, V_k), \\ X_{k+1} = DZ_{k+1}, \end{array} \right. \end{array} \tag{4.37}$$

where $\tau > 0$ is the stepsize of the FB algorithm.

As we showed in Proposition 4.1.2, when $L \to \infty$, $f^S_{L,\lambda}$ – which corresponds to denoising with sparse coding and a Dictionary D – is actually a proximal operator. Indeed, when $L \to \infty$, the signal recovery process based on Dictionary Learning corresponds to the proximal operator of the conjugate of $g^*_\lambda \circ D^*$. Thus, as for Analysis and according to Combettes and Wajs [2005], we can deduce the following convergence result for Plug and Pay based on Synthesis in an inverse problem context with measurement operator A.

THEOREM 4.2.7. *Let $(X_k)_{k\in\mathbb{N}}$ be generated by the algorithm in Equation 4.37, where $\tau \in \left(0, 2/\|A\|^2\right)$. Then $(X_k)_{k\in\mathbb{N}}$ converges to a solution to*

$$\operatorname*{argmin}_{X\in\mathbb{R}^N} \frac{1}{2}\|AX - Y\|^2 + (g^*_\lambda \circ D^*)^*(X) \ ,$$

when using $f^S_{\infty,\lambda} = \lim_{L\to\infty} f^S_{L,\lambda}$.

Proof 4.2.7

We have that $\lim_{L\to\infty} f^S_{L,\lambda} = \operatorname{prox}_{(g^*_\lambda \circ D^*)^*}$. Thus, the algorithm in Equation 4.37 reduces to a proximal gradient descent.

This result emphasizes that it is actually possible to write the Synthesis optimization problem that solves the inverse problem with the Analysis formulation, replacing $p(X) = \lambda g(\Gamma^* X)$ by $p(X) = (g^*_\lambda \circ D^*)^*(X)$. Once again, this behavior only holds when $L \to \infty$, i.e., if $f^S_{L,\lambda}$ has an infinite number of layers, and we now study what happens in the case $L < \infty$. First, when the dictionary has been learned, the reconstruction problem is classically solved by minimizing $\frac{1}{2}\|Y - ADZ\|_2^2 + \lambda g(Z)$ over Z. This problem can be solved with FISTA or any other sparse coding algorithm. For instance, the sparse codes Z can be obtained by proximal gradient descent algorithm, as follows

$$\forall t \in \mathbb{N} \quad Z_{t+1} = \operatorname{prox}_{\sigma\lambda g}\left(Z_t - \sigma D^* A^*(ADZ_t - Y))\right) \ , \tag{4.38}$$

and the signal reconstruction is then obtained with $X = DZ$. As demonstrated in Proposition 4.1.2, the Plug and Play counterpart introduced in this work recovers the signal by solving a problem of the form

$$\operatorname*{argmin}_{X} \frac{1}{2}\|Y - AX\|_2^2 + (g^*_\lambda \circ D^*)^*(X) \ . \tag{4.39}$$

In Proposition 4.2.8, we show that these methods solve equivalent problems.

PROPOSITION 4.2.8. *The two optimization problems*

$$(i) \quad D\big(\operatorname*{argmin}_{Z\in\mathbb{R}^S} \frac{1}{2}\|Y - ADZ\|_2^2 + \lambda g(Z)\big)$$

$$(ii) \quad \operatorname*{argmin}_{X\in\mathbb{R}^N} \frac{1}{2}\|Y - AX\|_2^2 + (g_\lambda^* \circ D^*)^*(X)$$

are equivalent.

Proof 4.2.8

Let $f = \frac{1}{2}\|A\cdot - Y\|_2^2$, then problem (i) finds a minimizer $Z^\star$ of

$$f(DZ) + \lambda g(Z) \ , \tag{4.40}$$

and reconstruct the signal with $DZ^\star$. By strong duality [Bauschke et al., 2011][Proposition 15.22], we have that

$$\inf_Z f(DZ) + \lambda g(Z) = \sup_V \big(- (\lambda g)^*(D^*V) - f^*(-V)\big) \tag{4.41}$$

$$= \sup_V \big(- ((\lambda g)^* \circ D^*)(V) - \sup_X \langle -V, X\rangle - \frac{1}{2}\|AX - Y\|_2^2\big) \tag{4.42}$$

$$= \sup_V \big(- ((\lambda g)^* \circ D^*)(V) + \inf_X \langle V, X\rangle + \frac{1}{2}\|AX - Y\|_2^2\big) \tag{4.43}$$

$$= \inf_X \big(\frac{1}{2}\|AX - Y\|_2^2 + \sup_V \langle V, X\rangle - ((\lambda g)^* \circ D^*)(V)\big) \tag{4.44}$$

$$= \inf_X \frac{1}{2}\|AX - Y\|_2^2 + ((\lambda g)^* \circ D^*)^*(X) \tag{4.45}$$

This shows that the values of the two problems minima are the same. Let's now show that when $X^\star$ is a minimizer of problem (ii), then $X^\star = DZ^\star$ where $Z^\star$ is a solution of problem (i), and when $Z^\star$ is a solution of problem (i), then $DZ^\star$ is a solution of problem (ii). Observe that if we take a minimizer $Z^\star$ of problem (i), then we have

$$\frac{1}{2}\|ADZ^\star - Y\|_2^2 + ((\lambda g)^* \circ D^*)^*(DZ^\star) = \frac{1}{2}\|ADZ^\star - Y\|_2^2 + \sup_V \langle V, DZ^\star\rangle - (\lambda g)^*(D^*V) \tag{4.46}$$

$$= \frac{1}{2}\|ADZ^\star - Y\|_2^2 + \sup_V \langle D^*V, Z^\star\rangle - (\lambda g)^*(D^*V) \tag{4.47}$$

$$\leq \frac{1}{2}\|ADZ^\star - Y\|_2^2 + \sup_V \langle V, Z^\star\rangle - (\lambda g)^*(V) \tag{4.48}$$

$$\leq \frac{1}{2}\|ADZ^\star - Y\|_2^2 + \lambda g(Z^\star) \tag{4.49}$$

$$\leq \frac{1}{2}\|AX^\star - Y\|_2^2 + ((\lambda g)^* \circ D^*)^*(X^\star) \tag{4.50}$$

where Equation 4.48 results from the fact that $\{D^*V; V \in \mathbb{R}^N\} \subset \mathbb{R}^N$, and Equation 4.50 results from the equality in Equation 4.45. Thus, if $Z^\star$ is a solution of problem (i), then $DZ^\star$ is a solution of problem (ii).

Let us now consider the reverse implication, that if $X^\star$ is a solution of (ii), then there exists $Z^\star$ a solution of (i) such that $X^\star = DZ^\star$. If we assume that the component of $X^\star$ contained in $\mathrm{Im}(D)^\perp = \ker D^*$ is not 0, then

$$\sup_V \langle V, X^\star\rangle - ((\lambda g)^* \circ D^*)(V) \geq \sup_{V \in \ker D^*} \langle V, X^\star\rangle - (\lambda g)^*(0) \tag{4.51}$$

$$\geq \infty \ . \tag{4.52}$$

As the value of Equation 4.45 is finite, this would violate the optimality of $X^\star$, thus $X^\star$ is contained in the image of D, and there exists $Z_X^\star$ such that $X^\star = DZ_X^\star$. Moreover, for bounded linear operators D and convex functions g, Bauschke et al. [2011][Proposition 13.21] show

$$(Dg)^* = g^* \circ D^* \ , \tag{4.53}$$

where $Dg(X) = \inf\{g(Z) \mid DZ = X\}$. Thus, $((\lambda g)^* \circ D^*)^* = (D(\lambda g))^{**} = D(\lambda g)$ because $D(\lambda g)$ is convex, and we have

$$\frac{1}{2}\|AX^\star - Y\|_2^2 + ((\lambda g)^* \circ D^*)^*(X^\star) = \frac{1}{2}\|AX^\star - Y\|_2^2 + \inf_Z\{\lambda g(Z) \mid DZ = X^\star\} \tag{4.54}$$

$$= \min_{Z, X^\star = DZ} \frac{1}{2}\|ADZ - Y\|_2^2 + \lambda g(Z) \ . \tag{4.55}$$

Taking $Z_X^\star$ a solution of Equation 4.55, we have

$$\frac{1}{2}\|ADZ_X^\star - Y\|_2^2 + \lambda g(Z_X^\star) = \frac{1}{2}\|AX^\star - Y\|_2^2 + ((\lambda g)^* \circ D^*)^*(X^\star) \tag{4.56}$$

$$= \frac{1}{2}\|ADZ^\star - Y\|_2^2 + \lambda g(Z^\star) \ . \tag{4.57}$$

This shows that $Z_X^\star$ is a minimizer of problem (i) and this conclude the proof.

In particular, when $g = \lambda\|\cdot\|_1$ and denoting $\mathcal{F}_\lambda(D) = \{x \mid \|D^*x\|_\infty \leq \lambda\}$,

$$\text{(i)} \quad D\big(\operatorname*{argmin}_Z \frac{1}{2}\|Y - ADZ\|_2^2 + \lambda\|Z\|_1\big) \tag{4.58}$$

$$\text{(ii)} \quad \operatorname*{argmin}_X \frac{1}{2}\|Y - AX\|_2^2 + \iota^*_{\mathcal{F}_\lambda(D)}(X) \tag{4.59}$$

are equivalent.

Proposition 4.2.8 shows that the classical resolution method based on sparse coding algorithms with the operator AD and the optimization method based on the computation of the prox operator described in Proposition 4.1.2 are equivalent. The following proposition shows that taking $L = 1$ leads to the same solution as the one obtained with $L = \infty$ in Theorem 4.2.7. This result follows from the fact that solving problem (i) in Proposition 4.2.8 is equivalent to using the algorithm described in Equation 4.37 with $f_{1,\lambda}^S$.

PROPOSITION 4.2.9. *Solving (i) in Proposition 4.2.8 is equivalent to a Plug and Play approach with $f_{1,\tau\lambda}^S$ or with the quadratic approximation defined as*

$$\hat{f}_{\tau\lambda}^S(Z_0, V) = D \operatorname*{argmin}_V \big(\frac{1}{2}\|V - DZ_0\|_2^2 + (Z - Z_0)^T(D^*(DZ_0 - V)) + \frac{1}{2\sigma}\|Z - Z_0\|_2^2 + \lambda\tau g(Z)\big) \ .$$

Proof 4.2.9

Let $\hat{f}^S_{\tau\lambda}$ *be defined as*

$$\hat{f}^S_{\tau\lambda}(Z_0, V) = D \operatorname*{argmin}_Z \big(\frac{1}{2}\|X - DZ_0\|_2^2 + (Z - Z_0)^T(D^*(DZ_0 - V)) + \frac{1}{2\sigma}\|Z - Z_0\|_2^2 + \lambda\tau g(Z)\big) \tag{4.60}$$

$$= D \operatorname{prox}_{\lambda\sigma g}\left(Z_0 - \sigma D^*(DZ_0 - V)\right) \ . \tag{4.61}$$

This is equivalent to take $L = 1$ *in the unrolled Synthesis algorithm,* i.e. $f^S_{1,\tau\lambda} = \hat{f}^S_{\tau\lambda}$. *Then, the PnP algorithm reduces to*

$$Z_{t+1} = \operatorname{prox}_{\lambda\sigma\tau g}\left(Z_t - \sigma D^*(DZ_t - (X_t - \tau A^*(AX_t - Y)))\right) \tag{4.62}$$

$$= \operatorname{prox}_{\lambda\sigma\tau g}\left((I - \sigma D^*D)Z_t + \sigma D^*(X_t - \tau A^*(AX_t - Y))\right) \tag{4.63}$$

$$X_{t+1} = DZ_{t+1} \ . \tag{4.64}$$

As $X_t = DZ_t$, *we have*

$$Z_{t+1} = \operatorname{prox}_{\lambda\sigma\tau g}\left((I - \sigma D^*D)Z_t + \sigma D^*(X_t - \tau A^*(AX_t - Y))\right) \tag{4.65}$$

$$= \operatorname{prox}_{\lambda\sigma\tau g}\left((I - \sigma D^*D)Z_t + \sigma D^*DZ_t - \sigma\tau D^*A^*(ADZ_t - Y))\right) \tag{4.66}$$

$$= \operatorname{prox}_{\lambda\sigma\tau g}\left(Z_t - \sigma\tau D^*A^*(ADZ_t - Y))\right) \ . \tag{4.67}$$

Thus, this is equivalent to the proximal gradient descent algorithm for (i) in Proposition 4.2.8.

As in the case of Analysis, Proposition 4.2.9 shows that unrolling one iteration is equivalent to solving the problem with a classical sparse coding algorithm. Moreover, we conjecture that taking $1 < L < \infty$ leads to the same result, as experimentally illustrated in Figure 4.2.

These results highlight that unfolded networks based on Analysis and Synthesis combined to Plug-and-Play with warm restart actually correspond to well-known formulations of the reconstruction problem in border cases, *i.e.* $L = 1$ and $L \to \infty$, and lead to the same results as classical optimization methods.

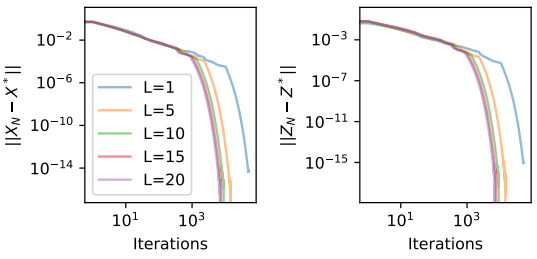

Figure 4.2: Convergence of signal and sparse code variables for different values of L in Synthesis. The number of layers does not change the convergence to the optimal solution X^* when the dictionary is fixed/

Bi-level optimization for approximate prox. When using Unrolling in Plug and Play, the success of the algorithm relies on the convergence of the optimization scheme with finite number L of layers/iterations. In particular, we proved in previous section that the special case $L = 1$ allows to recover the same solution as for $L \to \infty$. Here, we study to what extent this result generalizes to other values of L, i.e. $1 < L < \infty$. We focus on

the following regularized version of the problem

$$\min_{x,u\in\mathbb{R}^n} F(x,u) \triangleq f(x) + g(u) + \frac{1}{2\lambda}\|x-u\|_2^2 \ , \tag{4.68}$$

where $g(x)$ has been replaced by its (λ-)Moreau enveloppe $\min_u g(u) + \frac{1}{2\lambda}\|x-u\|_2^2$. This is equivalent to the problem

$$\min_x h(x) \triangleq F(x,u(x)) \quad \text{s.t.} \quad u(x) = \operatorname*{argmin}_u G(x,u) \tag{4.69}$$

where $G(x,u) = g(u) + \frac{1}{2\lambda}\|x-u\|_2^2$. In other words, $u(x) = \mathrm{prox}_{g_\lambda}(x)$, and this problem is well-defined as this is strongly convex.

In practice, we often don't have access to prox_g. Hence, it is approximated with an iterative algorithm $u^{t+1} = \mathcal{A}(u^t, x)$ which produces a sequence $(u^t(x))_{t\in\mathbb{N}}$ that converges toward $u(x)$ when $t \to \infty$. Here, we study the behavior of the bi-level optimization scheme

$$\begin{aligned} x^{t+1} &= x^t - \rho\nabla_x F(x,u^t) \\ u^{t+1} &= \mathcal{A}^L(u^t, x^{t+1}) \ . \end{aligned} \tag{4.70}$$

Note that an important point here is that we restart the prox algorithm with the output of the previous operator. In the following, we denote $u^{t+1}(x) = \mathcal{A}^L(u^t, x)$, and we make the hypothesis that

$$\forall x \in \mathbb{R}^n, \ \forall t \in \mathbb{N}, \ \|\mathcal{A}^L(u^t(x),x) - u(x)\|^2 \leq \tau(L)\|u^t(x) - u(x)\|^2 \ . \tag{4.71}$$

PROPOSITION 4.2.10. *If L is chosen such that $\tau(L) < \frac{1}{2}$, there exists $\rho > 0$ such that the algorithm described in Equation 4.70 converges to a solution of Equation 4.68.*

Proof 4.2.10

We have from the inequality $\|a+b\|^2 \leq 2\|a-c\|^2 + 2\|c-b\|^2$ that

$$\begin{aligned} \|u^{t+1} - u(x^{t+1})\|^2 &= \|\mathcal{A}^L(u^t(x^t), x^{t+1}) - u(x^{t+1})\|^2 && (4.72)\\ &\leq \tau(L)\|u^t(x^t) - u(x^{t+1})\|^2 && (4.73)\\ &\leq 2\tau(L)\big(\|u^t - u(x^t)\|^2 + \|u(x^t) - u(x^{t+1})\|^2\big) && (4.74) \end{aligned}$$

First, we use the fact that the prox is non-expansive and the gradient descent step $x^{t+1} = x^t - \rho\nabla_x F(x^t,u^t)$ to show that

$$\begin{aligned} \|u(x^t) - u(x^{t+1})\|^2 &\leq \|x^t - x^{t+1}\|^2 && (4.75)\\ &\leq \rho^2\|\nabla_x F(x^t,u^t)\|^2 && (4.76) \end{aligned}$$

Moreover,

$$h(x^{t+1}) = h(x^t - \rho \nabla_x F(x^t, u^t)) \tag{4.77}$$
$$\leq h(x^t) - \rho \langle \nabla_x F(x^t, u^t), \nabla h(x^t) \rangle + \frac{L_h}{2} \rho^2 \|\nabla_x F(x^t, u^t)\|^2 \tag{4.78}$$
$$\leq h(x^t) - \frac{\rho}{2} \|\nabla_x F(x^t, u^t)\|^2 - \frac{\rho}{2} \|\nabla h(x^t)\|^2 + \frac{\rho}{2} \|\nabla_x F(x^t, u^t) - \nabla h(x^t)\|^2 + \frac{L_h}{2} \rho^2 \|\nabla_x F(x^t, u^t)\|^2 \tag{4.79}$$
$$\leq h(x^t) - \frac{\rho}{2} \|\nabla_x h(x^t)\|^2 - \frac{\rho}{2} (1 - L_h \rho) \|\nabla_x F(x^t, u^t)\|^2 + \frac{\rho}{2} \|\nabla_x F(x^t, u^t) - \nabla h(x^t)\|^2 \tag{4.80}$$

and

$$\nabla_x F(x^t, u^t) - \nabla h(x^t) = \nabla f(x^t) + \frac{1}{\lambda}(x^t - u^t) - \nabla f(x^t) - \frac{1}{\lambda}(x^t - u(x^t)) \tag{4.81}$$
$$= \frac{1}{\lambda}(u(x^t) - u^t) \tag{4.82}$$

thus

$$h(x^{t+1}) \leq h(x^t) - \frac{\rho}{2} \|\nabla_x h(x^t)\|^2 - \frac{\rho}{2} (1 - L_h \rho) \|\nabla_x F(x^t, u^t)\|^2 + \frac{\rho}{2\lambda^2} \|u^t - u(x^t)\|^2 \tag{4.83}$$

We define

$$\mathcal{L}^t = h(x^t) + \phi \|u^t - u(x^t)\|^2 \ . \tag{4.84}$$

Then

$$\mathcal{L}^{t+1} - \mathcal{L}^t = h(x^{t+1}) - h(x^t) + \phi \|u^{t+1} - u(x^{t+1})\|^2 - \phi \|u^t - u(x^t)\|^2 \tag{4.85}$$
$$\leq \big(- \frac{\rho}{2} \|\nabla_x h(x^t)\|^2 - \frac{\rho}{2} (1 - L_h \rho) \|\nabla_x F(x^t, u^t)\|^2 + \frac{\rho}{2\lambda^2} \|u^t - u(x^t)\|^2 \big) + \phi \|u^{t+1} - u(x^{t+1})\|^2 - \phi \|u^t - u(x^t)\|^2 \tag{4.86}$$
$$\leq \big(- \frac{\rho}{2} \|\nabla_x h(x^t)\|^2 - \frac{\rho}{2} (1 - L_h \rho) \|\nabla_x F(x^t, u^t)\|^2 + \frac{\rho}{2\lambda^2} \|u^t - u(x^t)\|^2 \big) - \phi (1 - 2\tau(L)) \|u^t - u(x^t)\|^2 + 2\phi \rho^2 \tau(L) \|\nabla_x F(x^t, u^t)\|^2 \tag{4.87}$$
$$\leq - \frac{\rho}{2} \|\nabla_x h(x^t)\|^2 - \frac{\rho}{2} (1 - L_h \rho - 4\rho\phi\tau(L)) \|\nabla_x F(x^t, u^t)\|^2 - (\phi - 2\phi\tau(L) - \frac{\rho}{2\lambda^2}) \|u^t - u(x^t)\|^2 \big) \tag{4.88}$$

If ρ, ϕ verify

$$\begin{aligned} 1 - L_h \rho - 4\phi\rho\tau(L) &\geq 0 \\ \frac{\rho}{2\lambda^2} + 2\phi\tau(L) &\leq \phi \end{aligned} \tag{4.89}$$

and L sufficiently large so that $\tau(L) \leq \frac{1}{2}$, then

$$\mathcal{L}^{t+1} - \mathcal{L}^t \leq -\frac{\rho}{2}\|\nabla_x h(x^t)\|^2 \tag{4.90}$$

$$\text{i.e.} \quad (\mathcal{L}^T - \mathcal{L}^0) \leq -\frac{\rho}{2}\sum_{i=0}^{T-1}\|\nabla h(x^i)\|^2 \tag{4.91}$$

$$0 \leq \sum_{i=0}^{T-1}\|\nabla h(x^i)\|^2 \leq \frac{2\mathcal{L}^0}{\rho} \quad \forall T \in \mathbb{N} \tag{4.92}$$

In this case, $(\|\nabla h(x^t)\|)_{t\in\mathbb{N}}$ converges toward 0. As h is strictly convex with continuous gradient, the gradient vanishes at the minimum and $(x^t)_{t\in\mathbb{N}}$ converges toward the minimum of h.

Proposition 4.2.10 shows that when L is sufficiently large to get $\tau(L) < \frac{1}{2}$, then the optimization scheme described in Equation 4.70 converges to a solution of the regularized problem in Equation 4.68 for well-chosen values of ρ. In other words, taking the inexact proximal operator and combining the PnP scheme to warm restart allows to solve smoothed problems. We leave open the question of whether this result can be generalized to non-smooth regularization functions, as we previously conjectured.

Moreover, it is not straightforward to estimate values of L for which the main assumption $\|\mathcal{A}^L(u^t(x), x) - u(x)\|^2 < \frac{1}{2}\|u^t(x) - u(x)\|^2$ is true for all $x \in \mathbb{R}^n$ and for all $t \in \mathbb{N}$. As an example, if the algorithm that estimates the prox produces a sequence $(u^t)_{t\in\mathbb{N}}$ such that $G(x, u^{t+L}) - G(x, u^*) \leq \frac{C}{L}\|u^t - u^*\|_2^2$ where C is a constant that does not depend on x, then we have $\|u^{t+L} - u^*\|_2^2 \leq \frac{2\lambda C}{L}\|u^t - u^*\|_2^2$ by $(\frac{1}{\lambda})$-strong convexity of $G(x, \cdot)$. In this case, L should be superior to $4\lambda C$ to satisfy the hypothesis.

Reformulation of Analysis and Synthesis in the case of the Lasso. In the special case of the Lasso, it is possible to reformulate Analysis and Synthesis PnP methods to better show the similarities between then. Proposition 4.2.11 proposes a reformulation of these two problems.

PROPOSITION 4.2.11. *Let $\mathcal{F}_\lambda^A(\Gamma) = \{X \mid \inf\{\|Z\|_\infty \mid \Gamma Z = X\} \leq \lambda\}$ and $\mathcal{F}_\lambda^S(D) = \{X \mid \|D^*X\|_\infty \leq \lambda\}$. Analysis and Synthesis can be written as*

$$\textit{(Synthesis)} \quad \operatorname*{argmin}_X \max_{X_0 \in \mathcal{F}_\lambda^S(D)} \frac{1}{2}\|Y - AX\|_2^2 + \langle X, X_0\rangle$$

$$\textit{(Analysis)} \quad \operatorname*{argmin}_X \max_{X_0 \in \mathcal{F}_\lambda^A(\Gamma)} \frac{1}{2}\|Y - AX\|_2^2 + \langle X, X_0\rangle \ ,$$

or equivalently, denoting $\mathcal{B}(0, \lambda) = \{X \mid \|X\|_\infty \leq \lambda\}$,

$$\textit{(Synthesis)} \quad \operatorname*{argmin}_X \frac{1}{2}\|AX - Y\|_2^2 + (\inf\{\iota_{\mathcal{B}(0,\lambda)}(Z) \mid Z = D^*X\})^*$$

$$\textit{(Analysis)} \quad \operatorname*{argmin}_X \frac{1}{2}\|AX - Y\|_2^2 + (\inf\{\iota_{\mathcal{B}(0,\lambda)}(U) \mid \Gamma U = X\})^* \ .$$

Proof 4.2.11

Analysis solves

$$\min_X \frac{1}{2}\|AX - Y\|_2^2 + \lambda\|\Gamma^* X\|_1 \ . \tag{4.93}$$

Moreover, we have that for bounded linear operators L and convex functions f

$$(Lf)^* = f^* L^* \ , \tag{4.94}$$

where $Lf(y) = \inf\{f(x) \mid Lx = y\}$. Thus, as the conjugate of the ℓ_1 norm is the indicator of the ℓ_∞ unit ball, we have

$$\lambda\|\Gamma^* X\|_1 = \iota^*_{\mathcal{B}(0,\lambda)}(\Gamma^* X) \tag{4.95}$$

$$= (\inf\{\iota_{\mathcal{B}(0,\lambda)}(U) \mid \Gamma U = X\})^* \ . \tag{4.96}$$

Comparing with Synthesis, we get

$$\textit{(Synthesis)} \qquad \min_X \frac{1}{2}\|AX - Y\|_2^2 + (\iota_{\mathcal{B}(0,\lambda)}(D^* X))^* \tag{4.97}$$

$$= \min_X \frac{1}{2}\|AX - Y\|_2^2 + (\inf\{\iota_{\mathcal{B}(0,\lambda)}(Z) \mid Z = D^* X\})^* \tag{4.98}$$

$$\textit{(Analysis)} \qquad \min_X \frac{1}{2}\|AX - Y\|_2^2 + (\inf\{\iota_{\mathcal{B}(0,\lambda)}(U) \mid \Gamma U = X\})^* \tag{4.99}$$

Let $\mathcal{F}^A_\lambda(\Gamma) = \left\{X \mid \inf\{\|Z\|_\infty \mid \Gamma Z = X\} \leq \lambda\right\}$ and $\mathcal{F}^S_\lambda(D) = \{X \mid \|D^* X\|_\infty \leq \lambda\}$. Using the definition of the convex conjugate, i.e. $f^*(x) = \max_y \langle x, y\rangle - f(y)$, Analysis and Synthesis can be written as

$$\textit{(Synthesis)} \qquad \operatorname*{argmin}_X \max_{X_0 \in \mathcal{F}^S_\lambda(D)} \frac{1}{2}\|Y - AX\|_2^2 + \langle X, X_0\rangle \tag{4.100}$$

$$\textit{(Analysis)} \qquad \operatorname*{argmin}_X \max_{X_0 \in \mathcal{F}^A_\lambda(\Gamma)} \frac{1}{2}\|Y - AX\|_2^2 + \langle X, X_0\rangle \tag{4.101}$$

This result shows that in the context of inverse problems, Analysis and Synthesis both correspond to the same min-max optimization problem with different convex constraints on the second variable X_0. In the case of Synthesis, the convex set is

$$\mathcal{F}^S_\lambda(D) = \{X \mid \|D^* X\|_\infty \leq \lambda\} \ , \tag{4.102}$$

while for Analysis the convex set is

$$\mathcal{F}^A_\lambda(\Gamma) = \left\{X \mid \inf\{\|V\|_\infty \mid \Gamma V = X\} \leq \lambda\right\} \ . \tag{4.103}$$

Using these constraints, both problems can be reformulated as least squares with regularization functions that can be expressed as the convex conjugate of an infimum that depends on X. Moreover, denoting $\mathcal{C}$ a closed convex set and assuming $X \notin \mathcal{C}$, we have

$$\operatorname*{argmax}_{X_0 \in \mathcal{C}} \langle X, X_0\rangle = \mathrm{proj}_{\mathcal{C}}(X) \ . \tag{4.104}$$

Thus, the fixed point equation of gradient descent over X and maximization over X_0 leads

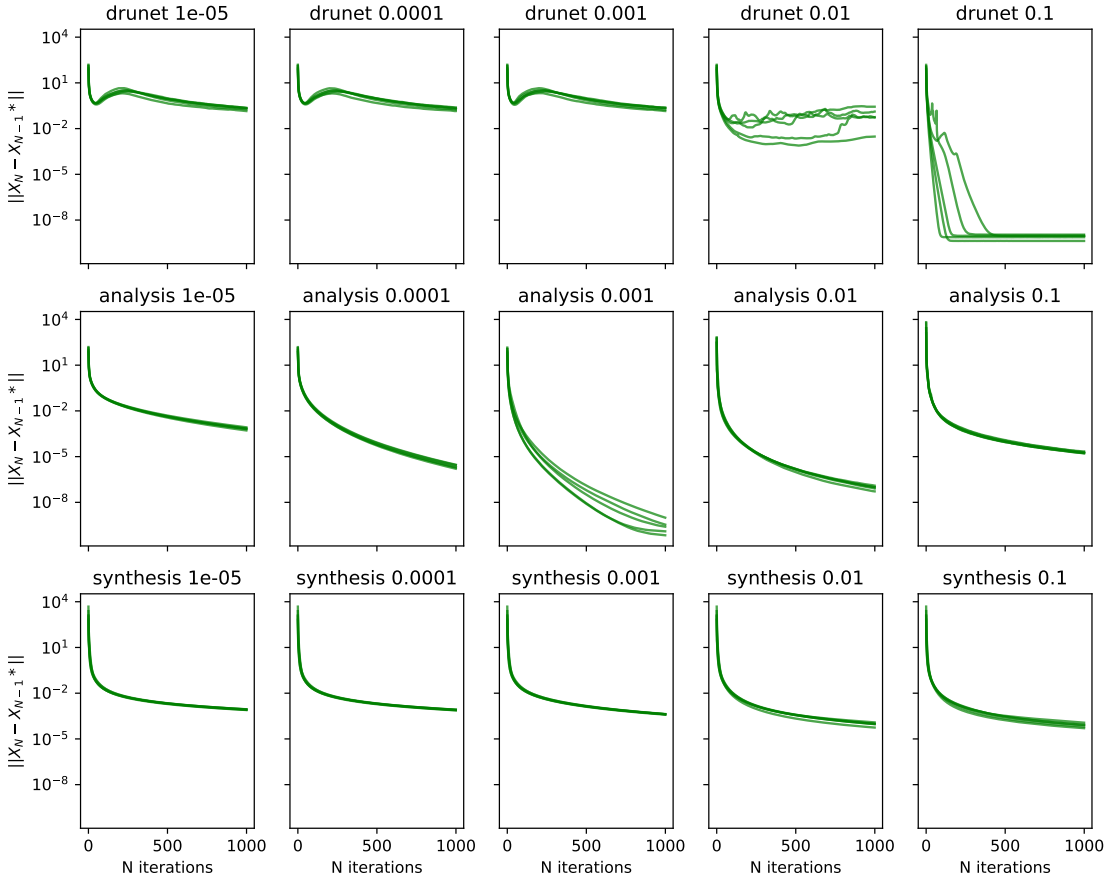

Figure 4.3: Convergence depending on the number of iterations. Analysis and Synthesis bases Plug-and-Play are proavbly stable, while DRU-Net may diverge for some regularization parameters.

to

$$X^* = (I - \tau A^* A)X^* + \tau(A^* Y + \text{proj}_{\mathcal{C}}(X^*)) \tag{4.105}$$

$$(A^* A + \text{proj}_{\mathcal{C}})(X^*) = A^* Y \ . \tag{4.106}$$

As expected, when $\lambda \to 0$, $\text{proj}_{\mathcal{C}}(X^*) \to 0$ and we recover the OLS solution $X^* = (A^* A)^{-1} A^* Y$.

Numerical results

We now present numerical illustrations of the behavior of unrolled dictionary learning compared to classical Plug-and-Play algorithms based on neural networks [Zhang et al., 2021]. Both Analysis and Synthesis based networks are trained with a fixed number of layers – 20 in our examples – on a train dataset from the BSD S500 image bank. Since we studied what happens when the dictionary is the same for each iteration, every layer shares the same parameters to reflect this. Moreover, the parameters are made of 50 convolutional filters of dimension 5×5. Then, images from the test dataset are blurred with a known kernel and reconstructed by plugging the network into a reconstruction algorithm with warm restart between iterations, as described in previous sections.

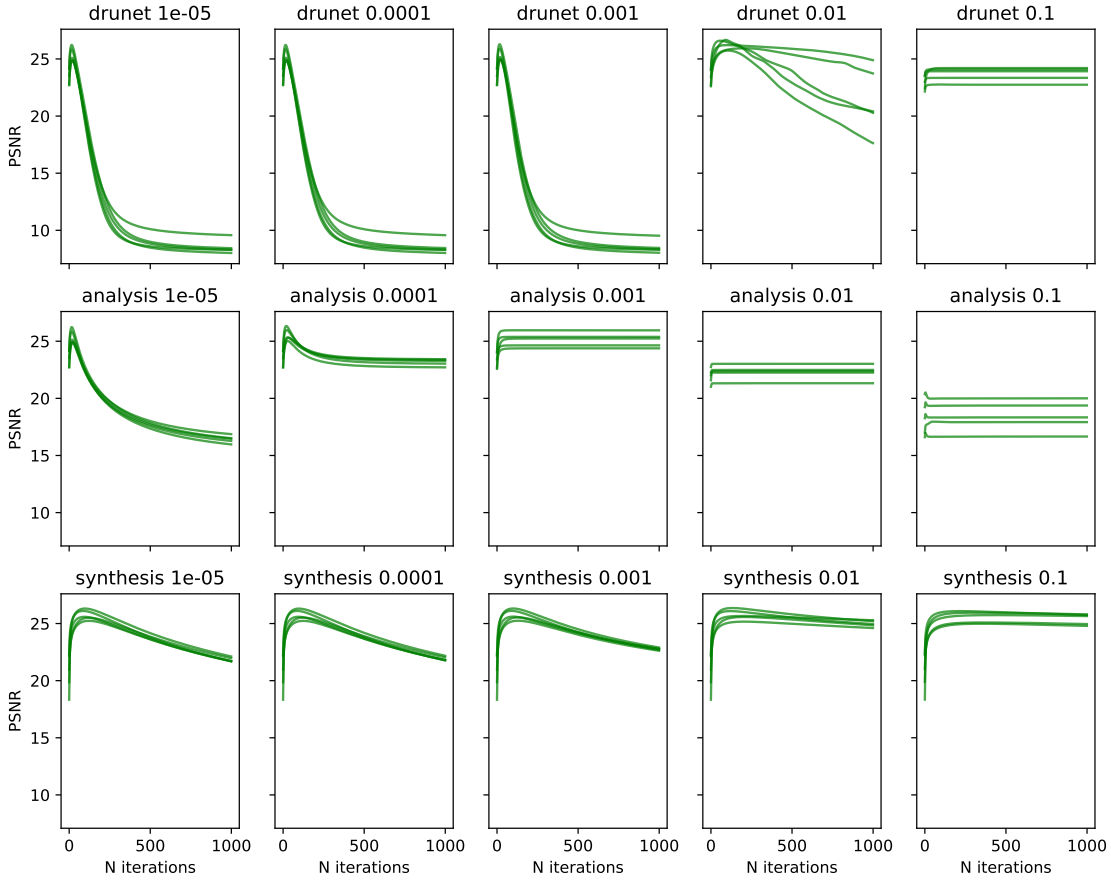

Figure 4.4: PSNR depending on the number of iterations. Analysis and Synthesis are globally more stable than DRU-Net when used as denoisers in Plug-and-Play.

We first compare convergence behaviors between Analysis, Synthesis and DRU-Net in Figure 4.3. The Plug-and-Play algorithm is run with different values of regularization, from 10^{-5} to 10^{-1}, and we display $\|X_N - X_{N-1}\|$ for different values of N on 5 different images from the test dataset. Plug-and-Play based on Synthesis and Analysis converges for each value of the regularization parameter, even though the speed of convergence may vary. On the other hand, Plug-and-Play based on DRU-Net only converges for the largest regularization value.

Then, we compare the reconstruction performance in Figure 4.4. This time, we display the PSNR depending on the regularization parameter, from 10^{-5} to 10^{-1}, and the number of iterations. The results are provided for 5 different images from the test dataset. This figure shows that the choice of hyper-parameter has a strong impact on the performance on the method based on DRU-Net and Analysis, while Synthesis is more robust to the choice of regularization. In particular, DRU-Net and Analysis reach the highest level of PSNR in early iterations for small parameters, and diverge to poor solutions without early stopping.

We propose five visual examples of the behavior of Analysis, Synthesis and DRU-Net after 1000 iterations of PnP in Figure 4.5. The choice of hyper-parameters is the result of a trade-

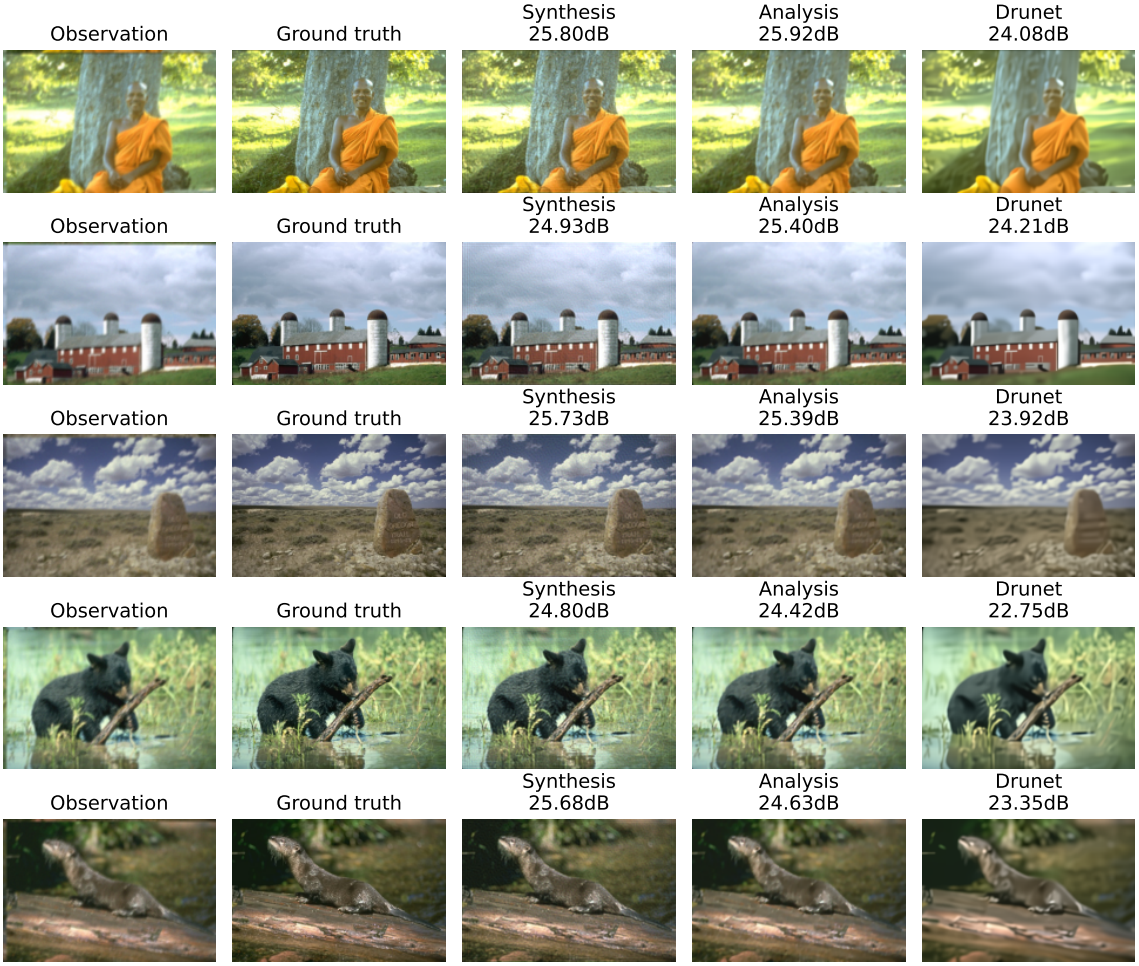

Figure 4.5: Visual results for deblurring on 5 images from the BSD S500 dataset.

off between the quality of the image and the convergence behavior, from results provided in previous experiments. In particular, the values of regularization parameters are 0.1 for DRU-Net, 0.001 for Analysis and 0.1 for Synthesis. Visually, Synthesis and Analysis work well compared to what we get with other methods based on PnP, even though Synthesis suffers from artefacts on textures. Moreover, the performance displayed for DRU-Net is obtained after convergence while it gives the best results with early-stopping – which is often used in practice.

Conclusion

We have proposed a study of unrolled Dictionary Learning in the context of Plug-and-Play by replacing the denoiser with a neural network inspired from a sparse coding algorithm, and parameterized with a dictionary. First, when the network is built from the Analysis formulation, it can be shown that the Plug-and-Play algorithm converges for an infinite number of layers and for one layer. Interestingly, both cases lead to the same final result, and the one layer algorithm is actually equivalent to a primal dual algorithm. Then, regarding the Synthesis formulation, we demonstrated that Dictionary based reconstructions are proximal operators and showed that Plug-and-Play combined to unrolling converges with an infinite number of layers and with one layer, as in the case of Analysis. In par-

ticular, the study of reconstruction instead of sparse codes in Synthesis offers a different perspective from what is usually found on the process, as this allows to see it as a proximal operator. Finally, we showed that this result extends to other numbers of layers both experimentally and theoretically in a particular case where the regularization function is replaced by its Moreau envelope, making it smooth. Future works could include proving that convergence is also guaranteed in the general case without the Moreau envelope, and studying what happens when the dictionary is different on each layer.

Part II

Insights on M/EEG signal processing

Electrophysiology refers to the branch of physiology that studies electrical phenomena occurring in organic tissues, like muscle fibers or neurons. In particular, the electrical activity of neurons is induced by the emission of post-synaptic potentials consecutive to ion exchanges at the synapse level. Then, the potentials of certain groups of active neurons sum up and induce an electro-magnetic field that becomes measurable at a macroscopic scale [Gramfort, 2009].

Magnetoencephalography and electroencephalography (M/EEG) are non-invasive techniques for recording the electrical activity of the brain [Hämäläinen et al., 1993]. Electroencephalography captures differences of potential all over the scalp, while Magnetoencephalography measures the density of the magnetic flux that go through the head. In both cases, the data consist of multivariate time series output by sensors placed around the head, which capture the intensity of the electric or magnetic field with very high temporal resolution. Even though the spatial resolution is rather poor due to the low number of sensors – which varies from a few dozens to around 200-300 – the high temporal sampling frequency makes these methods attractive for neuroscientists. As a matter of fact, those measurements provide information on cognitive processes as well as on the biological state of a subject that would not be available with other neuroimaging methods like MRI or fMRI that are limited by the time scales of hemodynamic responses.

Statistical analysis and learning from neural recordings is becoming more and more important in modern neuroscience. This motivates the development of computational tools able to deal with this type of signal. In the following, we will focus on two challenges of M/EEG signal processing: characteristic waveforms detection and prediction for brain-age regression and BCI.

Contributions on M/EEG signal processing

First, the quantitative analysis of M/EEG signals often boils down to the identification and study of temporal patterns such as evoked responses. As it is of interest to know to what extent these patterns are linked to a specific stimulus, event-based pattern learning methods are being developed [Dupré la Tour et al., 2018, Allain et al., 2021]. However, these methods do not scale well to large data-sets consisting of more than a few subjects. The contribution presented in chapter 5 is an efficient implementation of an MEG pattern learning algorithm at the population level.

Second, recent advances in modeling brain signals have highlighted that Riemannian geometry allows to effectively summarize M/EEG multivariate time series. Thus, many algorithms take advantage of covariance matrices and the underlying manifold to predict quantities of interest from the signal [Barachant et al., 2013, Sabbagh et al., 2019]. The contribution presented in chapter 6 is the development and study of a distance between distributions of symmetric positive definite matrices linked to covariance matrices. In particular, we use this distance to design Machine Learning methods for prediction and domain adaptation from M/EEG data.

Chapter 5

WinCDL: Fast Convolutional Dictionary Learning for M/EEG signals

This work was carried out in collaboration with Cédric Allain.

Neurophysiological signals recorded by M/EEG are composed of specific temporal patterns such as evoked responses induced by stimulations, transient bursts of neural oscillations, or artefacts that correspond, for instance, to eye blinks or heartbeats. The identification of these patterns plays a key role in the quantitative analysis of the signal, and an increasingly large line of work provides methods to automatically detect the waveforms and their occurrences in the data [Cole and Voytek, 2017, Dupré la Tour et al., 2018, Allain et al., 2021, Power et al., 2023].

In particular, Convolutional Dictionary Learning [Grosse et al., 2007] makes it possible to learn cognitive patterns corresponding to physiological activities by decomposing neural signals as combinations of spatio-temporal atoms that are time-invariant. As the electromagnetic waves propagate through the brain at the speed of light, every sensor measures the same waveform simultaneously but not at the same intensity. Thus, Dupré la Tour et al. [2018] propose to rely on multivariate convolutional sparse coding (CSC) with rank-1 constraint to leverage this physical property and learn prototypical patterns.

Fast numerical solvers for Dictionary Learning on large data have been successful on a wide range of tasks [Mairal et al., 2009, Wohlberg, 2015, Mensch et al., 2016]. Thus, efforts have been made to create new tools that allow to factorize the signal in such a way at the subject level in M/EEG [Jas et al., 2017, Dupré la Tour et al., 2018, Moreau and Gramfort, 2020], but they still require too many computations to scale to larger data-sets. This makes it very hard to develop analytical and statistical tools at the population level based on Dictionary Learning.

Contributions of chapter 5. We study stochastic sub-windowing in CDL, which consists of performing the sparse coding on sub-frames of signals to compute the gradient and demonstrate its practical efficiency on M/EEG signals. To do so, we introduce a novel

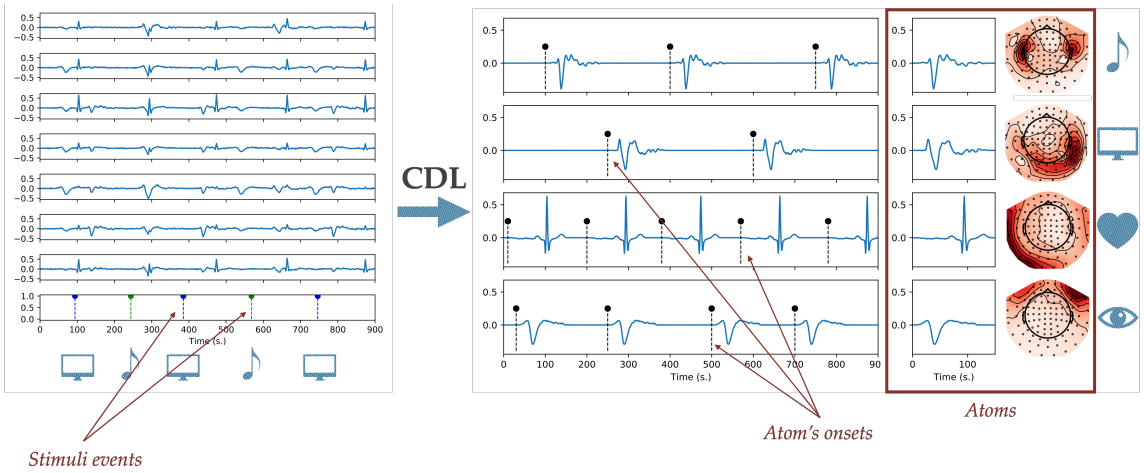

Figure 5.1: Rank-1 Convolutional Dictionary Learning learns temporal and spatial patterns corresponding to stimuli events and artefacts from the M/EEG signal, as well as the occurences of the patterns in the signal. Figure used with the courtesy of Cédric Allain.

implementation of CDL for large signals based on stochastic sub-windowing, approximate dictionary learning and `Pytorch` [Paszke et al., 2019]. We provide a benchmark of CDL algorithms and libraries on simulated data and MEG signals with the help of `Benchopt` [Moreau et al., 2022], as well as a theoretical study of the behavior of stochastic sub-windowing in Dictionary Learning.

5.1 . Background on Rank-1 Convolutional Dictionary Learning for pattern extraction in M/EEG

M/EEG sensors record time series at specific locations on the head. The physical process that links the electrical sources to the data, based on Maxwell's equations, shows that each sensor measures the same waveform at the same time. However, due to differences in tissue conductivity and to the distance between the electrical sources and the sensors, the intensity of the magnetic or electric field changes depending on the location.

In order to take these properties into account and restrict the set of solutions to the most realistic ones, Dupré la Tour et al. [2018] propose to rely on multivariate convolutional sparse coding with rank-1 constraint, as illustrated in Figure 5.1. In this case, space and time patterns are disjoint in each atom: $d_k = u_k v_k^T$ where u gathers the spatial activations on each channel and v corresponds to the temporal pattern. This leads to the model

$$\min_{z_k \in \mathbb{R}^T, u_k \in \mathbb{R}^S, v_k \in \mathbb{R}^t} \frac{1}{2} \| \sum_{k=1}^{n} (u_k v_k^\top) * z_k - y \|_2^2 + \lambda \sum_{k=1}^{n} \|z_k\|_1 \ , \tag{5.1}$$

where y is the signal, n is the number of atoms, T is the total recording time, t is the kernel size, and S is the number of sensors. For a dictionary $D = (d_k)_{1 \leq k \leq N}$ and sparse codes $Z = (z_k)_{1 \leq k \leq N}$, we define the cost function

$$F(Z, D, x) = \frac{1}{2} \| \sum_{k=1}^{N} d_k * z_k - y \|_2^2 + \lambda \sum_{k=1}^{N} \|z_k\|_1 \ . \tag{5.2}$$

Convolutional Dictionary learning can be written as a bi-level optimization problem to minimize the cost function with respect to the dictionary only, as mentioned in Mairal et al. [2009], by solving

$$\begin{aligned} &\min_{D \in \mathcal{C}} \; G(D, y) = F(D, Z^*(D), y) \\ &\text{with } Z^*(D) = \operatorname*{argmin}_{Z} F(D, Z, y) \;\; , \end{aligned} \tag{5.3}$$

with $\mathcal{C} = \{(d_k) \in \mathbb{R}^{c \times t}, \|d_k\|_2 \leq 1, d_k = u_k v_k^T\}$.

Classical dictionary learning methods solve this bi-convex optimization problem through *Alternating Minimization* (AM) [Mairal et al., 2009]. It consists in alterning on two steps. The first one involves minimizing the cost function F over Z with a fixed dictionary D, generally with the help of sparse coding algorithms like (F)ISTA [Daubechies et al., 2004, Beck and Teboulle, 2009], coordinate descent [Wu et al., 2008, Moreau et al., 2018], or ADMM [Bristow et al., 2013, Wohlberg, 2015]. Then, the second step consists either of computing the gradient $\nabla_1 F(D, Z, y)$, where ∇_1 indicates that the gradient is computed relatively to the first variable in F, to perform one or several steps of projected gradient descent on the dictionary, or of directly finding the optimal D by solving a least squares problem and thus computing a pseudo-inverse.

In the following, we will focus on gradient descent on D. Once $Z^*(D)$ is known, Danskin [1967, Thm 1] states that the gradient $g^* = \nabla G(D, y)$ is equal to $\nabla_1 F(D, Z^*(D), y)$. Even though the inner problem is non-smooth, this result holds as long as the solution $Z^*(D)$ is unique. Denoting by $D^\top$ the transpose operator of D, we will assume that $D^\top D$ is invertible on the support of $Z^*(D)$ in the following. This implies the uniqueness of $Z^*(D)$.

While CDL users generally want to get both D and Z, it is of interest to efficiently optimize Equation 5.3 in order to get a solution D^* as fast as possible, and then compute an optimal $Z^*(D^*)$. Indeed, the cumbersome part of the calculation is generally the sparse coding step, especially for large signals for which convolutions are expensive. In order to do that, several strategies are available. One is to compute approximate gradients, approach popularized by the usage of unrolled algorithms and automatic differentiation [Gregor and LeCun, 2010, Scetbon et al., 2021, Tolooshams and Ba, 2021] in the context of Dictionary Learning, that was studied in chapter 2. Another one is to adapt stochastic gradient descent to Dictionary Learning for large images or time series [Mensch et al., 2016].

5.2 . WinCDL

We here introduce a novel algorithm – called WinCDL – to efficiently solve Equation 5.1. The idea of WinCDL is to combine the empirical observations from the work on Unrolled Dictionary Learning in chapter 2 – *i.e.* a small number of iterations is sufficient to get a good estimate of the gradient – to stochastic sub-windowing of the signal where random sub-windows are sampled from the recording to get a stochastic estimate of the gradient. The aim is to reduce the computational costs associated with sparse coding. Another issue that comes up is that the choice of gradient steps is critical to the optimization process in dictionary learning, and SGD methods based on simple heuristics like rate decay are difficult to tune in this context. Thus, we propose to leverage an optimization scheme

Algorithm 5 Pseudo-code for WinCDL

Set N the number of iterations
Set $(U_{\alpha,t})_{t\in\mathbb{N}}$ a sequence of maximal step sizes decreasing to 0
Set $\mathcal{C}$ a set of constraints for (u, v)
for $1 \leq t \leq N$ **do**
  Sample index i and window $y[i, i + W]$ in the dataset
  Compute gradient $\nabla\mathcal{L}_i(u_t, v_t)$ of u_t and v_t with approximate sparse coding
  Compute best step size α_t with line search and starting point $U_{\alpha,t}$
  $(u_{t+1}, v_{t+1}) \leftarrow P_{\mathcal{C}}\big((u_{t+1}, v_{t+1}) - \alpha_t\nabla\mathcal{L}_i(u_t, v_t)\big)$
end for

introduced in Vaswani et al. [2019], which consists of performing a stochastic line search. The algorithm computes a good step size at each epoch, after which a heuristic decreases the maximal step. The use of a line search is possible due to the efficient computation of the loss with the approximate sparse codes $Z_N(D)$ when N is taken sufficiently small. The main steps of the algorithm are listed in Algorithm 5, and we now describe each step in details.

Approximate sparse coding.

As illustrated in chapter 2, optimizing over D does not necessarily require precise sparse coding at each gradient descent step. Here, we build our algorithm from a fixed number of FISTA iterations in the same spirit as Unrolling, but without leveraging back-propagation. Concretely, we thus have an approximation $Z_N(D)$ of $Z^*(D)$, where $Z_N(D)$ is given by N iterations of FISTA starting from 0, *i.e.*

$$Z_0 = 0 \text{ and } \alpha_0 = 1 \tag{5.4}$$

$$V_{t+1} = \mathrm{ST}_{\frac{\lambda}{L}}\big(Z_t - \frac{1}{L}D^\top * (D * Z_t - y)\big) \tag{5.5}$$

$$\alpha_{t+1} = \frac{1 + \sqrt{1 + 4\alpha_t^2}}{2} \tag{5.6}$$

$$Z_{t+1} = V_{t+1} + \frac{\alpha_t - 1}{\alpha_{t+1}}(V_{t+1} - V_t) \ , \tag{5.7}$$

where L is the Lipschitz constant of $D^\top * D$, and where $D * Z = \sum_{k=1}^n d_k * z_k$. In practice, we observed that choosing N between 20 and 30 is enough to get accurate results on MEG data.

Stochastic sub-windowing.

Given approximate sparse coding, the loss that we want to minimize is

$$\mathcal{L}(u, v) = \frac{1}{2}\|\sum_{k=1}^n (u_k v_k^\top) * z_{N,k}(u_k v_k^\top) - y\|_2^2 \ . \tag{5.8}$$

For datasets that contain large time series, as is the case in M/EEG, convolutions are expensive and the computational cost of FISTA increases. Thus, instead of processing all the signal at each iteration, we propose to sample smaller chunks of data from the

recordings when evaluating the loss. This makes the descent algorithm stochastic, but allows to dramatically reduce the computational cost. The sampling procedure consists in choosing a random chunk of signal $y[i : i + W]$ at each iteration, where W is a hyper-parameter corresponding to the length of the window and where i is an index in $[0, T - W]$. Then, we compute the gradient of the corresponding loss

$$\mathcal{L}_i(u, v) = \frac{1}{2} \| \sum_{k=1}^{n} (u_k v_k^\top) * z_{N,k,i}(u_k v_k^\top) - y[i : i + W] \|_2^2 \tag{5.9}$$

with respect to u and v assuming that the sparse codes are constant terms. This corresponds to usual Alternating Minimization, but on random chunks from the original signal y. We show in section 5.3 that this gives an unbiased estimate of the gradient in specific cases.

Stochastic line search.

Standard stochastic gradient descent algorithms fail to produce satisfying results on our problem because of the difficulty to tune the step size. Thus, we leverage a method called Stochastic line search, introduced in Vaswani et al. [2019], that extends the line search to the case where the gradient is stochastic. Indeed, line search algorithms are very helpful in situations where the step size of the gradient descent is hard to tune.

In usual gradient descent, the parameters $\theta_t = (u_t, v_t)$ are updated at each gradient step t with the help of $\nabla_\theta \mathcal{L}(\theta_t)$ and a step size parameter α_t, as follows

$$\theta_{t+1} = \theta_t - \alpha_t \nabla_\theta \mathcal{L}(\theta_t) \ . \tag{5.10}$$

The line search algorithm consists of finding the largest possible step size α_t starting from an upper bound $\bar{\alpha}$, by evaluating the loss $\mathcal{L}(\theta_t - \rho^n \bar{\alpha} \nabla_\theta \mathcal{L}(\theta_t))$ for $0 < \rho < 1$ and $n \in \mathbb{N}$ until the loss satisfies a condition of the form $\mathcal{L}(\theta_t - \rho^n \bar{\alpha} \nabla_\theta \mathcal{L}(\theta_t)) < \mathcal{L}(\theta_t) - c$ where c can either be constant or depend on the problem, or until $\rho^n < \epsilon$ where $\epsilon > 0$ is a stopping criterion. Then, $\alpha_t = \rho^{n_f}$ where n_f is the final number of iterations.

The idea of the Stochastic line search is to extend this principle in a scenario where we only have access to an estimate of the true gradient. In this case, naively applying a line search with each sample of gradient would lead to a non converging sequence of $(\theta_t)_{t \in \mathbb{N}}$, because the algorithm restarts the process at the upper bound $\bar{\alpha}$ for each new window of signal. Instead, the Stochastic extension uses a decreasing sequence of upper bounds $(\bar{\alpha}_i)_{i \in \mathbb{N}}$ that should converge to 0, with a well-chosen heuristic. For instance, our implementation is based on a sequence obtained through cosine annealing, starting from an initial upper bound that is a hyper-parameter of the algorithm. Thus, for each random sample i, once a gradient has been computed, the parameters are updated by evaluating $\mathcal{L}_i(\theta_t - \rho^n \bar{\alpha}_i \nabla_\theta \mathcal{L}_i(\theta_t))$ for increasing values of n, until the stopping criterion is reached.

Line search gradient descent is known to be time-consuming because it is necessary to compute the new loss for each potential choice of parameters. In the case of Dictionary Learning, this is usually a major issue, because the computation of the loss involves a sparse coding procedure. Our implementation replaces this expensive step by N iterations of proximal gradient descent that are fast to compute on GPU, which makes it possible to rely on a line search algorithm.

5.3 . Stochastic sub-windowing

Stochastic gradient descent traditionally consists of sampling i.i.d points from a data-set in order to get an estimate of the gradient. In the case of CDL, the problem comes from the fact that computing the gradient for a single point of the data-set is computationally expensive, for example when the point is a large time series or image. Thus, a simple idea is to reduce the computation to a sub-window of signal, for sparse coding to run in an acceptable time.

Let's take a one dimensional example. Denoting by $y \in \mathbb{R}^T$ a univariate time series, we can sample a window of size W from y and then estimate the gradient on this sub-window. In other words, we first sample an index i uniformly in $[0, T-W]$ and compute the sparse code estimator corresponding to $y[i : i+W]$. We define

$$\begin{aligned} S &= [1, T] \\ S_{W,i} &= [i, i+W-1] \\ S_{W,i,L} &= [i-L+1, i+W+L-2] \\ \partial S_{W,i,L} &= [i-(L-1), i] \cup [i+W-1, i+W-1+(L-1)] \ . \end{aligned} \tag{5.11}$$

Note that $S_{W,i} \cup \partial S_{W,i,L} = S_{W,i,L}$. We also define the restriction of Z onto $S_{W,i}$ (resp. $S_{W,i,L}$) by $Z_{S_{W,i}} \coloneqq Z[i, i+W-L] \in \mathbb{R}^{W-L+1}$ (resp. $Z_{S_{W,i,L}} \coloneqq Z[i-L+1, i+W-1] \in \mathbb{R}^{W+L-1}$).

To avoid border effects in the gradient estimation, it is possible to compute the sparse codes over the whole window W and to use only its restriction over $S_{W,i,L}$ where $L \in \mathbb{N}$ is the width of the buffer zone [Moreau and Gramfort, 2020]. In the following, we will make the assumption that the sparse code estimator obtained as explained above is equal to the original window $Z_{S_{W,i,L}}$. In that case, the gradient estimate is given by

$$\hat{g}^{W,i} = \nabla_1 F(D, Z_{S_{W,i,L}}, y[i : i+W]) \ , \tag{5.12}$$

and Proposition 5.3.1 shows that this is an unbiased estimator of the true gradient g^*.

PROPOSITION 5.3.1. *Under the assumption that for each window i, we have access to the correct value of $Z_{S_{W,i,L}}$ on the interval $S_{W,i,L}$, then*

$$\mathbb{E}_i \left[\hat{g}^{W,i}\right] = g^* \ .$$

Proof 5.3.1

The true gradient is

$$g^* = Z^- * X - Z^- * Z * D = \psi - \phi * D \in \mathbb{R}^L \tag{5.13}$$

where Z^- is obtained by reversal of the temporal dimension, i.e., *$Z^-[t] = Z[T+1-t]$, and $\forall s \in [1, L]$, $\psi[s] = \sum_{\tau=1}^{T-L+1} z[\tau] X[s+\tau-1]$ and $\phi[s] = \sum_{\tau=1}^{T-L+1} z[\tau] z[s+\tau-1]$. The estimated gradient on window $S_{W,i} = [i, i+W-1], 1 \leq i \leq T-W$ is*

$$\hat{g}^{W,i} = Z^-_{S_{W,i}} * X_{S_{W,i}} - Z^-_{S_{W,i}} * Z_{S_{W,i}} * D \tag{5.14}$$

$$= \psi^{W,i} - \phi^{W,i} * D \in \mathbb{R}^L \tag{5.15}$$

where $\psi^{W,i}[s] = \sum_{\tau=i}^{i+W} z[\tau+s]x[\tau]$ *for* $s \in [0, L-1]$ *and* $\phi^{W,i}[s] = \sum_{\tau=i}^{i+W} z[\tau+s]z[\tau]$ *for* $s \in [-L+1, L-1]$. *Here, we make the assumption that for each window* i, *we have access to the correct value of* z *on the interval* $S_{W,i,L}$ *(we add the border of size* L $\partial S_{W,i,L}$*).*
For $s \in [0, L-1]$, *we have that the value of* $\psi[s]$ *for the full signal is:*

$$\psi[s] = \sum_{\tau=0}^{T-L+1} z[\tau+s]x[\tau] \tag{5.16}$$

$$= \sum_{i=-W+1}^{T-L+1} \frac{1}{W} \sum_{\tau=i}^{i+W} z[\tau+s]x[\tau] \tag{5.17}$$

$$= \frac{1}{W} \sum_{i=-W+1}^{T-L+1} \psi^{W,i}[s] \tag{5.18}$$

$$\tag{5.19}$$

where the second line derives from the fact that each coefficient is seen through W *windows. Here, to avoid border effect, we consider we can take windows on the extended interval* $[-W+1, T-L+1+W]$, *for instance with zero padding.*
Thus, we can see that

$$\mathbb{E}_i\left[\psi^{W,i}[s]\right] = \frac{1}{T-L+W} \sum_{i=-W+1}^{T-L+1} \psi^{W,i}[s] = \frac{W}{T-L+W}\psi[s] \tag{5.20}$$

Similarly, we can show that $\mathbb{E}_i\left[\phi^{W,i}[s]\right] = \frac{W}{T-L+W}\phi[s]$ *for all* $s \in [-L+1, L-1]$.
Thus, by linearity of the convolution and the expectation, we get:

$$\mathbb{E}_i\left[\hat{g}^{W,i}\right] = \mathbb{E}_i\left[\psi^{W,i}\right] - \mathbb{E}_i\left[\phi^{W,i}\right] * D = \psi - \phi * D = g^* \tag{5.21}$$

Proposition 5.3.1 shows that under the assumption that the sparse code estimator is accurate on the sub-window, then we have access to an unbiased estimator and can apply stochastic gradient descent on our problem. This hypothesis may seem very restrictive, and is verified in practice only for large values of L when the sparsity of Z is high enough. In other words, patterns in the signal have to be well separated in order to avoid propagating border effects. However, we observed that taking sufficiently large windows W made it possible to reduce the bias that the error on the borders of $Z_{S_{W,i,L}}$ could introduce. The study of the impact of sparse code estimation errors on the gradient estimate is left for future work.

Numerical Results

In Figure 5.2, we provide a comparison of performance between WinCDL and two methods for rank-1 CDL implemented in the `Python` package `alphacsc`. The cost value is taken as $G(D, Y) = F(D, Z^*(D), Y)$, where the unknown to optimize is the dictionary. Thus, we do not take into account the ability of the algorithm to compute D and the sparse codes Z at the same time, but we evaluate the dictionary D by computing F with the exact sparse codes $Z^*(D)$. The figure shows that the usage of line search and sub-windowing allows

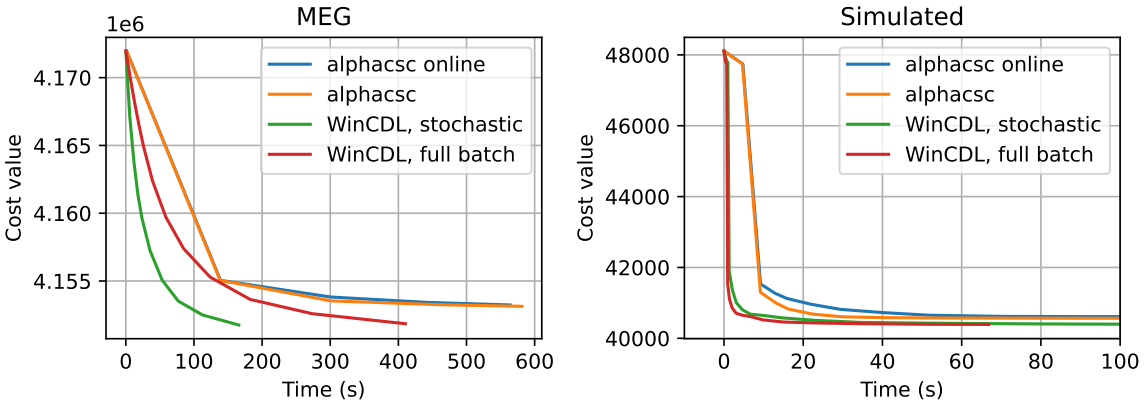

Figure 5.2: Cost value as a function of the time for algorithms from ***`alphacsc`*** *and WinCDL. WinCDL linesearch and sub-windowing allows to speed up the Dictionary Learning process.*

to speed up the Dictionary Learning process, both on simulated data where multivariate time series are generated as linear combination of cosine, and on MEG data.

The MEG data-set contains recordings of a subject submitted to audio and visual stimuli. The subject was also asked to press a button as fast as possible when shown a smiley face. The experiment lasts about 5 minutes. Figure 5.3 presents 30 atoms learned from this recording with WinCDL. We plot the spatial and temporal representations of each pattern, which may correspond either to artefacts or to evoked responses. The results are coherent with what is generally obtained with `alphacsc` on the same task with equivalent parameters, as shown in Figure 5.4.

Conclusion

We proposed a new implementation for Convolutional Dictionary Learning on large data for M/EEG recordings based on stochastic sub-windowing and a stochastic line search. The implementation allows to learn the dictionary faster, without the need of precise optimization of the sparse codes at each gradient descent iteration. In practice, we obtain similar results as reference libraries regarding the temporal waveforms and spatial patterns in the atoms, but the method can be applied to larger datasets, including at the population level. Thus, this work might be seen as a first step towards the design of efficient tools based on dictionary learning and sparse coding for population studies in neuroscience.

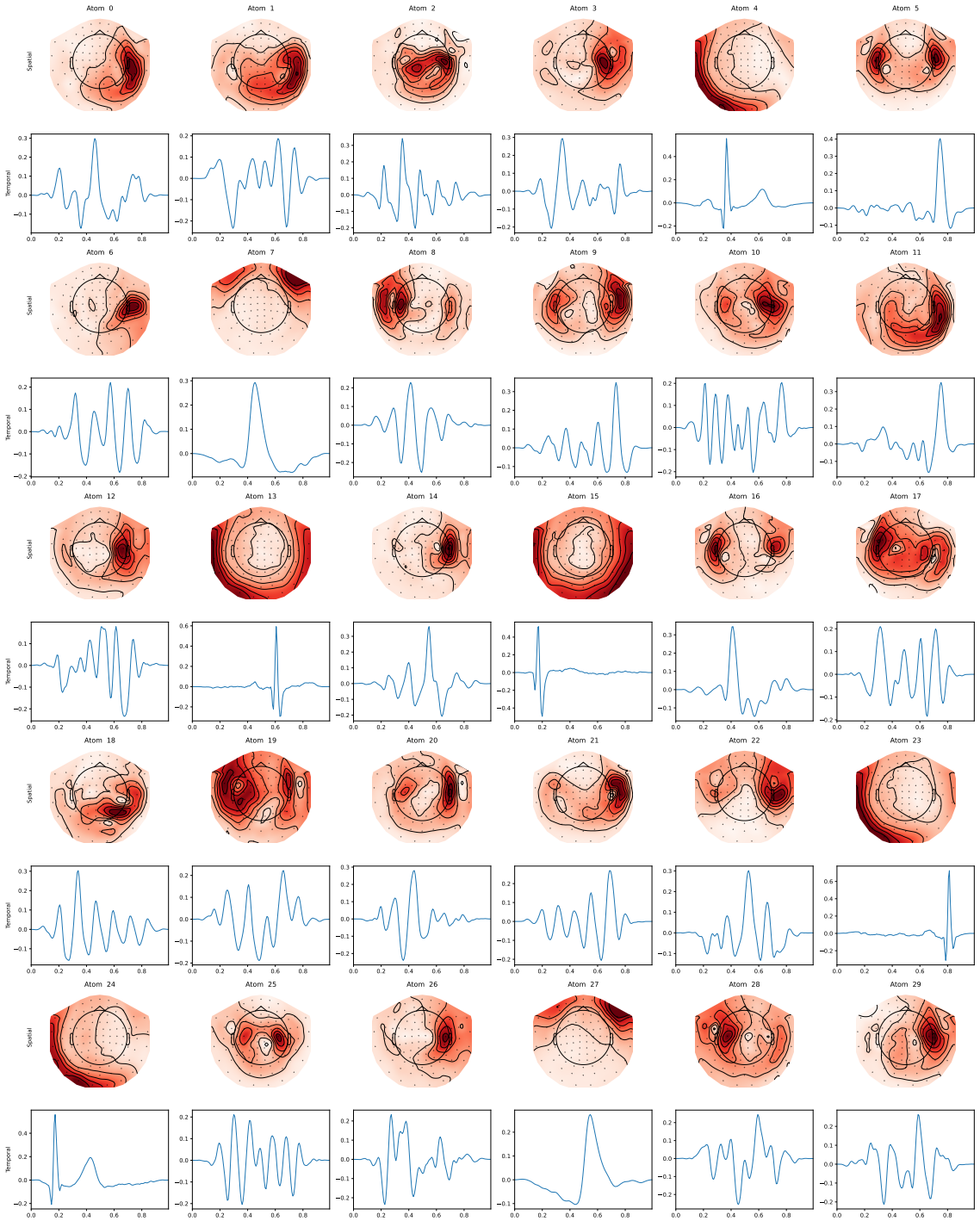

Figure 5.3: 30 atoms learned by WinCDL from a MEG recording.

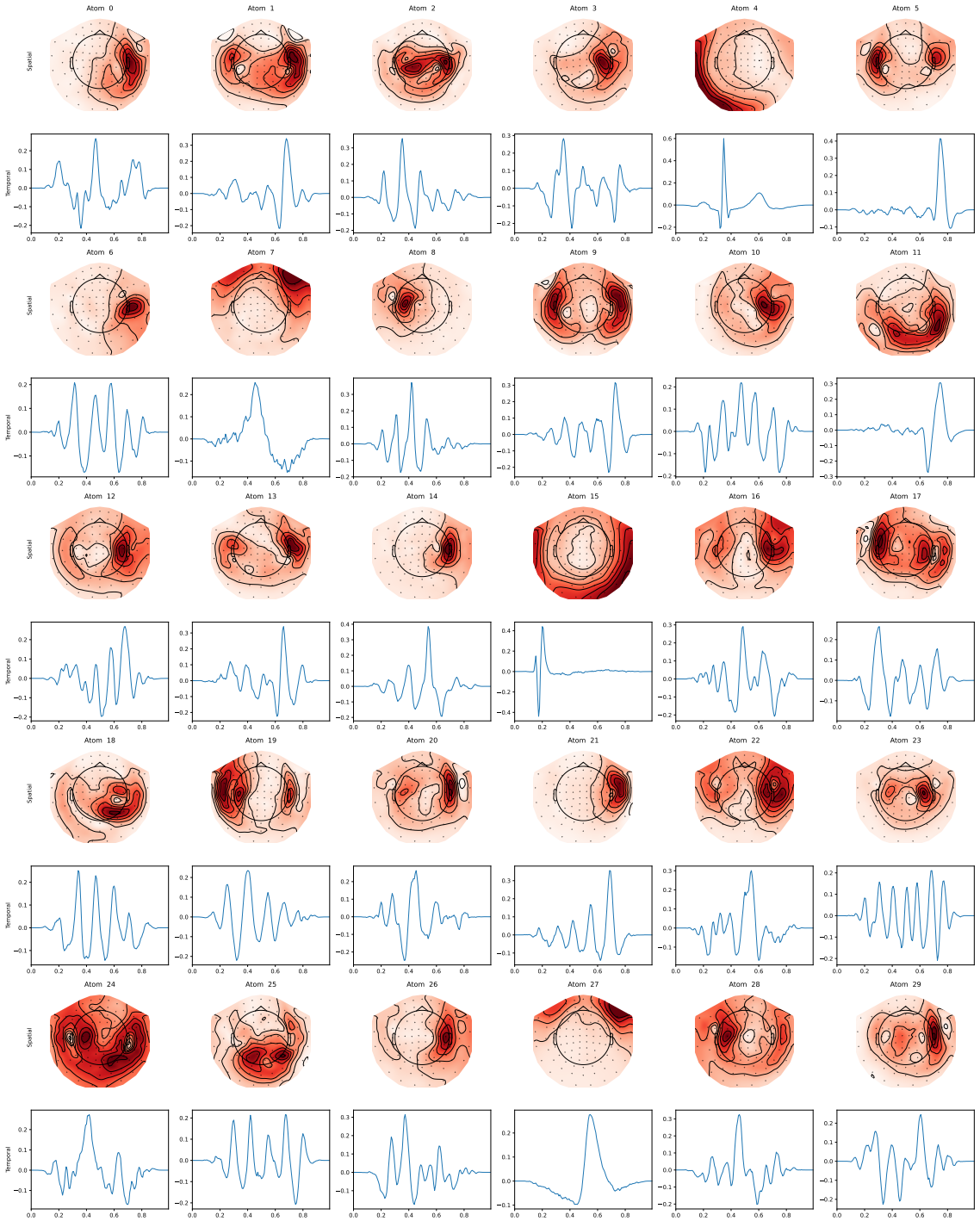

Figure 5.4: 30 atoms learned by **`alphacsc`** *from a MEG recording.*

Chapter 6

Sliced-Wasserstein on Symmetric Positive Definite Matrices for M/EEG signals

This work was carried out in collaboration with Clément Bonet, with the help of Alain Rakotomamonjy, Lucas Drumetz, and Nicolas Courty. The content of this chapter was published in:

Clément Bonet, Benoît Malézieux, Alain Rakotomamonjy, Lucas Drumetz, Thomas Moreau, Matthieu Kowalski, and Nicolas Courty. Sliced-wasserstein on symmetric positive definite matrices for m/eeg signals. *International Conference on Machine Learning*, 2023a

Successful machine learning (ML) techniques that deal with M/EEG data often rely on covariance matrices estimated from band-passed filtered signals in several frequency bands [Blankertz et al., 2007]. The main difficulty that arises when processing such covariance matrices is that the set of symmetric positive definite (SPD) matrices is not a linear space, but a Riemannian manifold [Bhatia, 2009, Bridson and Haefliger, 2013]. Therefore, specific algorithms have to be designed to take into account the non Euclidean structure of the data. The usage of Riemannian geometry on SPD matrices has become increasingly popular in the ML community [Huang and Van Gool, 2017, Chevallier et al., 2017, Ilea et al., 2018, Brooks et al., 2019]. In particular, these tools have proven to be very effective on prediction tasks with M/EEG data in Brain Computer Interface (BCI) applications [Barachant et al., 2011, 2013, Gaur et al., 2018] or more recently in brain-age prediction [Sabbagh et al., 2019, 2020, Engemann et al., 2022]. As covariance matrices sets from M/EEG data are often modeled as samples from a probability distribution – for instance in domain adaptation for BCI [Yair et al., 2019] – it is of great interest to develop efficient tools that work directly on those distributions.

Optimal transport (OT) [Villani, 2009, Peyré et al., 2019] provides a powerful theoretical framework and computational toolbox to compare probability distributions while respect-

ing the geometry of the underlying space. It is well defined on Riemannian manifolds [McCann, 2001, Cui et al., 2019, Alvarez-Melis et al., 2020] and in particular on the space of SPD matrices that is considered in M/EEG learning tasks [Brigant and Puechmorel, 2018, Yair et al., 2019, Ju and Guan, 2022]. The original OT problem defines the Wasserstein distance which has a super cubic complexity *w.r.t* samples. To alleviate the computational burden, different alternatives were proposed such as adding an entropic regularization [Cuturi, 2013] or computing the distance between mini-batches [Fatras et al., 2020]. Another popular alternative is the Sliced-Wasserstein distance (SW) [Rabin et al., 2011] which computes the average of the Wasserstein distance between one-dimensional projections. SW has recently received a lot of attention as it significantly reduces the computational burden while preserving topological properties of Wasserstein [Bonnotte, 2013, Nadjahi et al., 2020, Bayraktar and Guo, 2021]. Moreover, Kolouri et al. [2016], Meunier et al. [2022] have shown that, as opposed to Wasserstein, SW allows to properly extend kernel methods to data-sets of distributions with very efficient computation of the kernel matrix. This opens the way to new regression and classification methods. However, the initial construction of SW is restricted to Euclidean spaces. Thus, a new line of work focuses on its extension to specific manifolds [Rustamov and Majumdar, 2020, Bonet et al., 2022, 2023b].

Contributions of chapter 6.

In order to benefit from the advantages of SW in the context of M/EEG, we propose an SW distance on the manifold of SPD matrices and evaluate its efficiency on two prediction tasks.

- We introduce an SW discrepancy between measures of symmetric positive definite matrices (SPDSW), and provide a well-founded numerical approximation.
- We derive theoretical results, including topological, statistical, and computational properties. In particular, we prove that SPDSW is a distance topologically equivalent to the Wasserstein distance in this context.
- We extend the distribution regression with SW kernels to the case of SPD matrices, apply it to brain-age regression with MEG data, and show that it performs better than other methods based on Riemannian geometry.
- We show that SPDSW is an efficient surrogate to the Wasserstein distance in domain adaptation for BCI.

6.1 . Sliced-Wasserstein on SPD matrices

In this section, we introduce an SW discrepancy on SPD matrices and provide a theoretical analysis of its properties and behavior.

Euclidean Sliced-Wasserstein distance.

For $\mu, \nu \in \mathcal{P}_p(\mathbb{R}^d)$ two measures with finite moments of order $p \geq 1$, the Wasserstein distance is defined as

$$W_p^p(\mu, \nu) = \inf_{\gamma \in \Pi(\mu,\nu)} \int \|x - y\|_2^p \, \mathrm{d}\gamma(x, y) \ , \tag{6.1}$$

where $\Pi(\mu,\nu) = \{\gamma \in \mathcal{P}(\mathbb{R}^d \times \mathbb{R}^d),\ \pi^1_\#\gamma = \mu,\ \pi^2_\#\gamma = \nu\}$ denotes the set of couplings between μ and ν, π^1 and π^2 the projections on the first and second coordinate and $\#$ is the push-forward operator, defined as a mapping on all borelian $A \subset \mathbb{R}^d$, such that $T_\#\mu(A) = \mu(T^{-1}(A))$. For practical ML applications, this distance is computed between two empirical distributions with n samples and the main bottleneck consists in solving the linear program (6.1). Its computational complexity is $O(n^3 \log n)$ [Pele and Werman, 2009] which is expensive for large scale applications.

While computing (6.1) is costly in general, it can be computed efficiently for problems where $d = 1$, as it admits the following closed-form [Peyré et al., 2019, Remark 2.30]

$$W_p^p(\mu,\nu) = \int_0^1 |F_\mu^{-1}(u) - F_\nu^{-1}(u)|^p \, \mathrm{d}u \ , \tag{6.2}$$

where F_μ^{-1} and F_ν^{-1} are the quantile functions of μ and ν. By computing order statistics, this can be approximated from samples in $O(n \log n)$.

This observation motivated the construction of the SW distance [Rabin et al., 2011, Bonneel et al., 2015] which is defined as the average of the Wasserstein distance between one dimensional projections of the measures in all directions, *i.e.* for $\mu, \nu \in \mathcal{P}_p(\mathbb{R}^d)$,

$$\mathrm{SW}_p^p(\mu,\nu) = \int_{S^{d-1}} W_p^p(t^\theta_\#\mu, t^\theta_\#\nu) \, \mathrm{d}\lambda(\theta) \ , \tag{6.3}$$

where λ is the uniform distribution on the sphere $S^{d-1} = \{\theta \in \mathbb{R}^d,\ \|\theta\|_2 = 1\}$ and t^θ is the coordinate of the projection on the line $\mathrm{span}(\theta)$, *i.e.* $t^\theta(x) = \langle x, \theta\rangle$ for $x \in \mathbb{R}^d$, $\theta \in S^{d-1}$. This distance has many advantages, motivating its use in place of the Wasserstein distance. First, it can be approximated in $O\big(Ln(d + \log n)\big)$ with L projections and a Monte-Carlo method. Moreover, it is topologically equivalent to the Wasserstein distance as it also metrizes the weak convergence [Nadjahi et al., 2019], and its sample complexity is independent of the dimension [Nadjahi et al., 2020] as opposed to Wasserstein. Finally, it is a Hilbertian metric and it can be used to define kernels over probability distributions [Kolouri et al., 2016, Carriere et al., 2017, Meunier et al., 2022]. This is particularly interesting for regression or classification over data-sets of distributions, as we will see for brain-age prediction.

Background on SPD matrices.

Let $S_d(\mathbb{R})$ be the set of symmetric matrices of $\mathbb{R}^{d\times d}$, and $S_d^{++}(\mathbb{R})$ be the set of SPD matrices of $\mathbb{R}^{d\times d}$, *i.e.* matrices $M \in S_d(\mathbb{R})$ satisfying

$$\forall x \in \mathbb{R}^d \setminus \{0\},\ x^T M x > 0 \ . \tag{6.4}$$

$S_d^{++}(\mathbb{R})$ is a Riemannian manifold [Bhatia, 2009], meaning that it behaves locally as a linear space, called a tangent space. Each point $M \in S_d^{++}(\mathbb{R})$ defines a tangent space $\mathcal{T}_M$, which can be given an inner product $\langle\cdot,\cdot\rangle_M : \mathcal{T}_M \times \mathcal{T}_M \to \mathbb{R}$, and thus a norm. The choice of this inner-product induces different geometry on the manifold. One example is the geometric and Affine-Invariant metric [Pennec et al., 2006], where the inner product is defined as

$$\begin{aligned} &\forall M \in S_d^{++}(\mathbb{R}),\ A, B \in \mathcal{T}_M, \\ &\langle A, B\rangle_M = \mathrm{Tr}(M^{-1}AM^{-1}B) \ . \end{aligned} \tag{6.5}$$

Denoting by Tr the Trace operator, the corresponding geodesic distance $d_{AI}(\cdot,\cdot)$ is given by

$$\forall X, Y \in S_d^{++}(\mathbb{R}), d_{AI}(X,Y) = \sqrt{\mathrm{Tr}(\log(X^{-1}Y)^2)} \ . \tag{6.6}$$

Another example is the Log-Euclidean metric [Arsigny et al., 2005, 2006] for which,

$$\begin{aligned} &\forall M \in S_d^{++}(\mathbb{R}),\ A, B \in \mathcal{T}_M, \\ &\langle A, B\rangle_M = \langle D_M \log A, D_M \log B\rangle \ , \end{aligned} \tag{6.7}$$

with log the matrix logarithm and $D_M \log A$ the directional derivative of the log at M along A [Huang et al., 2015]. This definition provides another geodesic distance [Arsigny et al., 2006]

$$\forall X, Y \in S_d^{++}(\mathbb{R}),\ d_{LE}(X,Y) = \| \log X - \log Y \|_F \ , \tag{6.8}$$

which is simply a Euclidean distance in $S_d(\mathbb{R})$ in this case. We will use the Log-Euclidean metric in the following, as it is simpler and faster to compute while being a good first order approximation of the Affine-Invariant metric [Arsigny et al., 2005, Pennec, 2020]. In this case, the geodesic between $X, Y \in S_d^{++}(\mathbb{R})$ is $t \in \mathbb{R} \mapsto \exp((1-t)\log X + t \log Y)$. log is a diffeomorphism from $S_d^{++}(\mathbb{R})$ to $S_d(\mathbb{R})$, whose inverse is exp. Thus, the geodesic line going through $A \in S_d(\mathbb{R})$ and the origin of $S_d(\mathbb{R})$ is $\mathcal{G}_A = \{\exp(tA),\ t \in \mathbb{R}\}$. To span all such geodesics, we can restrict to A with unit Frobenius norm, i.e. $\|A\|_F = 1$.

Construction of SPDSW**.**

On a Euclidean space, the SW distance is defined by averaging the Wasserstein distance between the distributions projected over all possible straight lines passing through the origin. As $S_d^{++}(\mathbb{R})$ with Log-Euclidean metric is a geodesically complete Riemannian manifold, *i.e.* there exists a geodesic curve between each couple of points and each geodesic curve can be extended to $\mathbb{R}$, a natural generalization of SW on this space can be obtained by averaging the Wasserstein distance between distributions projected over all geodesics passing through the origin I_d.

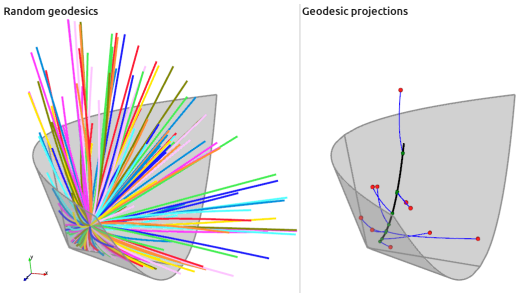

*Figure 6.1: (***Left***) Random geodesics drawn in $S_2^{++}(\mathbb{R})$. (***Right***) Projections (green points) of covariance matrices (depicted as red points) over one geodesic (in black) passing through I_2 along the Log-Euclidean geodesics (blue lines).*

To construct SPDSW, we need several ingredients. First, it is required to find the projection onto a geodesic $\mathcal{G}_A$ passing through I_d where $A \in S_d(\mathbb{R})$. Such projection $P^{\mathcal{G}_A}$ can be obtained as follows

$$\forall M \in S_d^{++}(\mathbb{R}),\ P^{\mathcal{G}_A}(M) = \underset{X \in \mathcal{G}_A}{\mathrm{argmin}}\ d_{LE}(X, M) \ , \tag{6.9}$$

and we provide the closed-form in Proposition 6.1.1.

Proposition 6.1.1. *Let $A \in S_d(\mathbb{R})$ with $\|A\|_F = 1$, and let $\mathcal{G}_A$ be the associated geodesic line . Then, for any $M \in S_d^{++}(\mathbb{R})$, the geodesic projection on $\mathcal{G}_A$ is*

$$P^{\mathcal{G}_A}(M) = \exp\left(\mathrm{Tr}(A \log M) A\right) \ .$$

Proof 6.1.1

Let $M \in S_d^{++}(\mathbb{R})$. We want to solve

$$P^{\mathcal{G}_A}(M) = \underset{X \in \mathcal{G}_A}{\operatorname{argmin}} \; d_{LE}(X, M)^2. \tag{6.10}$$

In the case of the Log-Euclidean metric, $\mathcal{G}_A = \{\exp(tA),\ t \in \mathbb{R}\}$. We have

$$\begin{aligned} d_{LE}(\exp(tA), M)^2 &= \| \log \exp(tA) - \log M \|_F^2 \\ &= \| tA - \log M \|_F^2 \\ &= t^2 \mathrm{Tr}(A^2) + \mathrm{Tr}(\log(M)^2) - 2t\mathrm{Tr}(A \log M) \\ &= g(t) \ . \end{aligned} \tag{6.11}$$

Hence

$$g'(t) = 0 \iff t = \frac{\mathrm{Tr}(A \log M)}{\mathrm{Tr}(A^2)} \ . \tag{6.12}$$

Therefore

$$P^{\mathcal{G}_A}(M) = \exp\left(\frac{\mathrm{Tr}(A \log M)}{\mathrm{Tr}(A^2)} A \right) = \exp\left(\mathrm{Tr}(A \log M) A \right) \ , \tag{6.13}$$

since $\|A\|_F^2 = \mathrm{Tr}(A^2) = 1$.

Then, the coordinate of the projection on $\mathcal{G}_A$ can be obtained by giving an orientation to $\mathcal{G}_A$ and computing the distance between $P^{\mathcal{G}_A}(M)$ and the origin I_d, as follows

$$t^A(M) = \mathrm{sign}(\langle \log M, A \rangle_F) d_{LE}(P^{\mathcal{G}}(M), I_d) \ . \tag{6.14}$$

The closed-form expression is given by Proposition 6.1.2.

PROPOSITION 6.1.2. *Let $A \in S_d(\mathbb{R})$ with $\|A\|_F = 1$, and let $\mathcal{G}_A$ be the associated geodesic line. Then, for any $M \in S_d^{++}(\mathbb{R})$, the geodesic coordinate on $\mathcal{G}_A$ is*

$$t^A(M) = \langle A, \log M \rangle_F = \mathrm{Tr}(A \log M) \ .$$

Proof 6.1.2

First, we give an orientation to the geodesic. This can be done by taking the sign of the inner product between $\log(P^{\mathcal{G}_A}(M))$ and A.

$$\begin{aligned} t^A(M) &= \mathrm{sign}(\langle A, \log(P^{\mathcal{G}_A}(M)) \rangle_F) d\big(P^A(M), I\big) && (6.15) \\ &= \mathrm{sign}(\langle A, \log(P^{\mathcal{G}_A}(M)) \rangle_F) d\left(\exp\left(\mathrm{Tr}(A \log M) A\right), I\right) && (6.16) \\ &= \mathrm{sign}(\langle A, \langle A, \log M \rangle_F A \rangle_F) \| \langle A \log M \rangle_F A - \log I \|_F && (6.17) \\ &= \mathrm{sign}(\langle A, \log M \rangle_F) |\langle A, \log M \rangle_F| && (6.18) \\ &= \langle A, \log M \rangle_F && (6.19) \\ &= \mathrm{Tr}(A \log M) \ . && (6.20) \end{aligned}$$

These two properties give a closed-form expression for the Riemannian equivalent of one-dimensional projection in a Euclidean space. In Figure 6.1, we illustrate the projections by matrices $M \in S_2^{++}(\mathbb{R})$ embedded as vectors $(m_{11}, m_{22}, m_{12}) \in \mathbb{R}^3$. $S_2^{++}(\mathbb{R})$ is an open

cone and we plot the projections of random SPD matrices on geodesics passing through I_2.

We are now ready to define an SW discrepancy on measures in $\mathcal{P}_p(S_d^{++}(\mathbb{R})) = \{\mu \in \mathcal{P}(S_d^{++}(\mathbb{R})),\ \int d_{LE}(X, M_0)^p \ \mathrm{d}\mu(X) < \infty,\ M_0 \in S_d^{++}(\mathbb{R})\}$.

DEFINITION. *Let λ_S be the uniform distribution on $\{A \in S_d(\mathbb{R}),\ \|A\|_F = 1\}$. Let $p \geq 1$ and $\mu, \nu \in \mathcal{P}_p(S_d^{++}(\mathbb{R}))$, then the* SPDSW *discrepancy is defined as*

$$\mathrm{SPDSW}_p^p(\mu, \nu) = \int_{S_d(\mathbb{R})} W_p^p(t^A_\# \mu, t^A_\# \nu) \ \mathrm{d}\lambda_S(A) \ .$$

As shown by the definition, being able to sample from λ_S is the cornerstone of the computation of SPDSW. In Lemma 6.1.3, we propose a practical way of uniformly sampling a symmetric matrix A. More specifically, we sample an orthogonal matrix P and a diagonal matrix D of unit norm and compute $A = PDP^T$ which is a symmetric matrix of unit norm. This is equivalent to sampling from λ_S as the measures are equal up to a normalization factor $d!$ which represents the number of possible permutations of the columns of P and D for which $PDP^T = A$.

LEMMA 6.1.3. *Let λ_O be the uniform distribution on $\mathcal{O}_d = \{P \in \mathbb{R}^{d\times d}, P^TP = PP^T = I\}$ (Haar distribution), and λ be the uniform distribution on $S^{d-1} = \{\theta \in \mathbb{R}^d, \|\theta\|_2 = 1\}$. Then $\lambda_S \in \mathcal{P}(S_d(\mathbb{R}))$, defined such that $\forall\ A = P\mathrm{diag}(\theta)P^T \in S_d(\mathbb{R})$, $\mathrm{d}\lambda_S(A) = d!\ \mathrm{d}\lambda_O(P)\mathrm{d}\lambda(\theta)$, is the uniform distribution on $\{A \in S_d(\mathbb{R}),\ \|A\|_F = 1\}$.*

Proof 6.1.3

A matrix in $S_d(\mathbb{R})$ has a unique decomposition $P\mathrm{diag}(\theta)P^T$ up to permutations of the columns of $P \in \mathcal{O}_d$ and coefficients of $\theta \in S^{d-1}$. Thus, there is a bijection between $\{A \in S_d(\mathbb{R}),\ \|A\|_F = 1\}$ and the set $S_{(O),S^{d-1}}$ of $d!$-tuple $\{(P_1, \theta_1), \dots, (P_{d!}, \theta_{d!}) \in (\mathcal{O}_d \times S^{d-1})^{d!}\}$ such that (P_i, θ_i) is a permutation of (P_j, θ_j). Therefore, the uniform distribution $\lambda_{S_{(O),S^{d-1}}}$ on $S_{(O),S^{d-1}}$, defined as $\mathrm{d}\lambda_{S_{(O),S^{d-1}}}((P_1, \theta_1), \dots, (P_{d!}, \theta_{d!})) = \sum_{i=1}^{n!} \mathrm{d}(\lambda_O \otimes \lambda)(P_i, \theta_i) = d! \cdot \mathrm{d}(\lambda_O \otimes \lambda)(P_1, \theta_1)$, allows to define a uniform distribution λ_S on $\{A \in S_d(\mathbb{R}),\ \|A\|_F = 1\}$. Let $A = P\mathrm{diag}\theta P^T$ with $(P, \theta) \in \mathcal{O}_d \times S^{d-1}$, then

$$\mathrm{d}\lambda_S(A) = d!\ \mathrm{d}(\lambda_O \otimes \lambda)(P, \theta) \ . \tag{6.21}$$

Then, the coordinate of the projection on the geodesic $\mathcal{G}_A$ is provided by $t^A(\cdot) = \mathrm{Tr}(A \log \cdot)$ defined in Proposition 6.1.2. The Wasserstein distance is easily computed using order statistics, and this leads to a natural extension of the SW distance in $S_d^{++}(\mathbb{R})$. There exists a strong link between SW on distributions in $\mathbb{R}^{d\times d}$ and SPDSW. Indeed, Proposition 6.1.4 shows that SPDSW is equal to a variant of SW where projection parameters are sampled from unit norm matrices in $S_d(\mathbb{R})$ instead of the unit sphere, and where the distributions are pushed forward by the log operator.

PROPOSITION 6.1.4. *Let $\tilde{\mu}, \tilde{\nu} \in \mathcal{P}_p(S_d(\mathbb{R}))$, and $\tilde{t}^A(B) = \mathrm{Tr}(A^T B)$ for $A, B \in S_d(\mathbb{R})$. We define*

$$\mathrm{SymSW}_p^p(\tilde{\mu}, \tilde{\nu}) = \int_{S_d(\mathbb{R})} W_p^p(\tilde{t}^A_{\#}\tilde{\mu}, \tilde{t}^A_{\#}\tilde{\nu}) \, \mathrm{d}\lambda_S(A) \ .$$

Then, for $\mu, \nu \in \mathcal{P}_p(S_d^{++}(\mathbb{R}))$,

$$\mathrm{SPDSW}_p^p(\mu, \nu) = \mathrm{SymSW}_p^p(\log_{\#}\mu, \log_{\#}\nu) \ .$$

Proof 6.1.4

Denoting $\tilde{t}^A(B) = \langle B, A \rangle_F$ for all $B \in S_d(\mathbb{R})$, we obtain using [Paty and Cuturi, 2019, Lemma 6]

$$W_p^p(\tilde{t^A}_{\#}\log_{\#}\mu, \tilde{t}^A_{\#}\log_{\#}\nu) = \inf_{\gamma \in \Pi(\mu,\nu)} \int_{S_d^{++}(\mathbb{R})\times S_d^{++}(\mathbb{R})} |\tilde{t}^A(\log(X)) - \tilde{t}^A(\log(Y))|^p \, \mathrm{d}\gamma(X,Y) \tag{6.22}$$

$$= \inf_{\gamma \in \Pi(\mu,\nu)} \int_{S_d^{++}(\mathbb{R})\times S_d^{++}(\mathbb{R})} |t^A(X) - t^A(Y)|^p \, \mathrm{d}\gamma(X,Y) \tag{6.23}$$

$$= W_p^p(t^A_{\#}\mu, t^A_{\#}\nu) \ , \tag{6.24}$$

since $\tilde{t}^A(\log X) = \langle A, \log X \rangle_F = t^A(X)$. Hence,

$$\mathrm{SymSW}_p^p(\log_{\#}\mu, \log_{\#}\nu) = \mathrm{SPDSW}_p^p(\mu, \nu) \ . \tag{6.25}$$

Thus, it seems natural to compare the results obtained with SPDSW to the Euclidean counterpart $\log \mathrm{SW} = \mathrm{SW}(\log_{\#}\cdot, \log_{\#}\cdot)$ where the distributions are made of projections in the log space and where the sampling is done with the uniform distribution on the sphere. The Wasserstein distance is also well defined on Riemannian manifolds, and in particular on the space of SPD matrices. Denoting d a geodesic distance on $S_d^{++}(\mathbb{R})$, we can define the corresponding Wasserstein distance between $\mu, \nu \in \mathcal{P}_p(S_d^{++}(\mathbb{R}))$ as

$$W_p^p(\mu, \nu) = \inf_{\gamma \in \Pi(\mu,\nu)} \int d(X,Y)^p \, \mathrm{d}\gamma(X,Y) \ . \tag{6.26}$$

In the following, we study properties of SPDSW and in particular, we show that it is a computationally efficient alternative to Wasserstein on $\mathcal{P}(S_d^{++}(\mathbb{R}))$ as it is topologically equivalent while having a better computational complexity and being better conditioned for regression of distributions.

6.2 . Properties of SPDSW

We now derive theoretical properties of SPDSW.

Topology.

Following usual arguments which are valid for any sliced divergence with any projection, we can show that SPDSW is a pseudo-distance. Here, $S_d^{++}(\mathbb{R})$ with the Log-Euclidean metric is of null sectional curvature [Arsigny et al., 2005, Xu, 2022] and we have access

to a diffeomorphism to a Euclidean space – the log operator. This allows us to show that SPDSW is a distance in Theorem 6.2.1.

THEOREM 6.2.1. *Let $p \geq 1$, then SPDSW_p is a finite distance on $\mathcal{P}_p(S_d^{++}(\mathbb{R}))$.*

Proof 6.2.1

Let $p \geq 1$, and $\mu, \nu \in \mathcal{P}_p(S_d^{++}(\mathbb{R}))$. First, let's check that $\mathrm{SPDSW}_p^p(\mu,\nu) < \infty$.
To see that, we will use on one hand Villani [2009, Definition 6.4] which states that on a Riemannian manifold $\mathcal{M}$, for any $x_0 \in \mathcal{M}$,

$$\forall x, y \in \mathcal{M},\ d(x,y)^p \leq 2^{p-1}\big(d(x,x^0)^p + d(x^0,y)^p\big). \tag{6.27}$$

Moreover, we will use that the projection t^A is equal (up to a sign) to the Busemann function which is 1-Lipschitz [Bridson and Haefliger, 2013, II. Proposition 8.22] and hence for any $A \in S_d(\mathbb{R})$ such that $\|A\|_F = 1$ and $X, Y \in S_d^{++}(\mathbb{R})$, $|t^A(X) - t^A(Y)| \leq d_{LE}(X,Y)$. Then, using Paty and Cuturi [2019, Lemma 6], we have, for any $\pi \in \Pi(\mu,\nu)$ and $X_0 \in S_d^{++}(\mathbb{R})$,

$$W_p^p(t^A_\#\mu, t^A_\#\nu) = \inf_{\gamma\in\Pi(\mu,\nu)} \int_{S_d^{++}(\mathbb{R})\times S_d^{++}(\mathbb{R})} |t^A(X) - t^A(Y)|^p \,\mathrm{d}\gamma(X,Y) \tag{6.28}$$

$$\leq \int_{S_d^{++}(\mathbb{R})\times S_d^{++}(\mathbb{R})} |t^A(X) - t^A(Y)|^p \,\mathrm{d}\pi(X,Y) \tag{6.29}$$

$$\leq 2^{p-1}\left(\int_{S_d^{++}(\mathbb{R})} |t^A(X) - t^A(X_0)|^p \,\mathrm{d}\mu(X) + \int_{S_d^{++}(\mathbb{R})} |t^A(X_0) - t^A(Y)|^p \,\mathrm{d}\nu(Y)\right) \tag{6.30}$$

$$\leq 2^{p-1}\left(\int_{S_d^{++}(\mathbb{R})} d_{LE}(X,X_0)^p \,\mathrm{d}\mu(X) + \int_{S_d^{++}(\mathbb{R})} d_{LE}(Y,X_0)^p \,\mathrm{d}\nu(Y)\right) \tag{6.31}$$

$$< \infty\ . \tag{6.32}$$

Let $p \geq 1$, then for all $\mu, \nu \in \mathcal{P}_p(S_d^{++}(\mathbb{R}))$, it is straightforward to see that $\mathrm{SPDSW}_p(\mu,\nu) \geq 0$, $\mathrm{SPDSW}_p(\mu,\nu) = \mathrm{SPDSW}_p(\nu,\mu)$. It is also easy to see that $\mu = \nu \implies \mathrm{SPDSW}_p(\mu,\nu) = 0$ using that W_p is a distance.
Now, we can also derive the triangular inequality using the triangular inequality for W_p and the Minkowski inequality, i.e. *$\forall \mu, \nu, \alpha \in \mathcal{P}_p(S_d^{++}(\mathbb{R}))$*

$$\mathrm{SPDSW}_p(\mu,\nu) = \Big(\int_{S_d(\mathbb{R})} W_p^p(t^A\#\mu, t^A\#\nu) \,\mathrm{d}\lambda_S(A)\Big)^{\frac{1}{p}} \tag{6.33}$$

$$\leq \Big(\int_{S_d(\mathbb{R})} \big(W_p(t^A_\#\mu, t^A_\#\alpha) + W_p(t^A_\#\alpha, t^A_\#\nu)\big)^p \,\mathrm{d}\lambda_S(A)\Big)^{\frac{1}{p}} \tag{6.34}$$

$$\leq \Big(\int_{S_d(\mathbb{R})} W_p^p(t^A\#\mu, t^A_\#\alpha) \,\mathrm{d}\lambda_S(A)\Big)^{\frac{1}{p}} \tag{6.35}$$

$$+ \Big(\int_{S_d(\mathbb{R})} W_p^p(t^A_\#\alpha, t^A_\#\nu) \,\mathrm{d}\lambda_S(A)\Big)^{\frac{1}{p}} \tag{6.36}$$

$$= \mathrm{SPDSW}_p(\mu,\alpha) + \mathrm{SPDSW}_p(\alpha,\nu). \tag{6.37}$$

Lastly, we can derive the indiscernible property. Let $\mu, \nu \in \mathcal{P}_p(S_d(\mathbb{R}))$ such that $\mathrm{SPDSW}_p(\mu, \nu) = 0$. *Then, as for all $A \in S_d(\mathbb{R})$, $W_p^p(t^A_\#\mu, t^A_\#\nu) \geq 0$, it implies that for λ_S-almost every A, $W_p^p(t^A_\#\mu, t^A_\#\nu) = 0$ which implies $t^A_\#\mu = t^A_\#\nu$ for λ_S-almost every A since W_p is a distance. By taking the Fourier transform, this implies that for all $s \in \mathbb{R}$, $\widehat{t^A_\#\mu}(s) = \widehat{t^A_\#\nu}(s)$. But, we have*

$$\begin{aligned}
\widehat{t^A_\#\mu}(s) &= \int_{\mathbb{R}} e^{-2i\pi ts} \, \mathrm{d}(t^A_\#\mu)(s) && (6.38)\\
&= \int_{S_d^{++}(\mathbb{R})} e^{-2i\pi t^A(M)s} \, \mathrm{d}\mu(M) && (6.39)\\
&= \int_{S_d^{++}(\mathbb{R})} e^{-2i\pi \langle sA, \log M\rangle_F} \, \mathrm{d}\mu(M) && (6.40)\\
&= \int_{S_d(\mathbb{R})} e^{-2i\pi \langle sA, S\rangle_F} \, \mathrm{d}(\log_\# \mu)(S) && (6.41)\\
&= \widehat{\log_\# \mu}(sA). && (6.42)
\end{aligned}$$

Hence, we get that $\mathrm{SPDSW}_p(\mu, \nu) = 0$ *implies that for λ_S-almost every A,*

$$\forall s \in \mathbb{R}, \ \widehat{\log_\# \mu}(sA) = \widehat{t^A_\#\mu}(s) = \widehat{t^A_\#\nu}(s) = \widehat{\log_\# \nu}(sA). \tag{6.43}$$

By injectivity of the Fourier transform on $S_d(\mathbb{R})$, we get $\log_\# \mu = \log_\# \nu$. Then, as log *is a diffeomorphism from $S_d^{++}(\mathbb{R})$ to $S_d(\mathbb{R})$, we have for all Borelian $M \subset S_d^{++}(\mathbb{R})$,*

$$\begin{aligned}
\mu(M) &= \int_{S_d^{++}(\mathbb{R})} \mathbb{1}_M(X) \, \mathrm{d}\mu(X) && (6.44)\\
&= \int_{S_d(\mathbb{R})} \mathbb{1}_M(\exp(S)) \, \mathrm{d}(\log_\# \mu)(S) && (6.45)\\
&= \int_{S_d(\mathbb{R})} \mathbb{1}_M(\exp(S)) \, \mathrm{d}(\log_\# \nu)(S) && (6.46)\\
&= \int_{S_d^{++}(\mathbb{R})} \mathbb{1}_M(Y) \, \mathrm{d}\nu(Y) && (6.47)\\
&= \nu(M). && (6.48)
\end{aligned}$$

Hence, we conclude that $\mu = \nu$ and that SPDSW_p *is a distance.*

In the case of the Affine-Invariant metric, the Riemannian manifold endowed with this metric has a non-positive and non-constant sectional curvature, and closed-forms of geodesics projections are not known to the best of our knowledge. We can however derive Busemann coordinates, which involve a costly additional projection. Moreover, whether or not it satisfies the indiscernible property remains an open question. Hence, we focus on SPDSW with Log-Euclidean metric.

An important property which justifies the use of the SW distance in place of the Wasserstein distance in the Euclidean case is that they both metrize the weak convergence [Bonnotte, 2013]. We show in Theorem 6.2.2 that this is also the case with SPDSW in $\mathcal{P}_p(S_d^{++}(\mathbb{R}))$.

THEOREM 6.2.2. *For $p \geq 1$, SPDSW_p metrizes the weak convergence,* i.e. *for $\mu \in \mathcal{P}_p(S_d^{++}(\mathbb{R}))$ and a sequence $(\mu_k)_k$ in $\mathcal{P}_p(S_d^{++}(\mathbb{R}))$, $\lim_{k\to\infty} \mathrm{SPDSW}_p(\mu_k, \mu) = 0$ if and only if $(\mu_k)_k$ converges weakly to μ.*

Proof 6.2.2

To prove Theorem 6.2.2, we will adapt the proof of Nadjahi et al. [2020] to our projection. First, we start by adapting Nadjahi et al. [2020, Lemma S1].

Lemma. *Let $(\mu_k)_k \in \mathcal{P}_p(S_d^{++}(\mathbb{R}))$ and $\mu \in \mathcal{P}_p(S_d^{++}(\mathbb{R}))$ such that $\lim_{k\to\infty} \mathrm{SPDSW}_1(\mu_k, \mu) = 0$. Then, there exists $\varphi : \mathbb{N} \to \mathbb{N}$ non decreasing such that $\mu_{\varphi(k)} \xrightarrow[k\to\infty]{\mathcal{L}} \mu$.*

Proof of the Lemma. *By Bogachev and Ruas [2007, Theorem 2.2.5],*

$$\lim_{k\to\infty} \int_{S^d(\mathbb{R})} W_1(t^A_\# \mu_k, t^A_\# \mu)\, \mathrm{d}\lambda_S(A) = 0 \tag{6.49}$$

implies that there exits a subsequence $(\mu_{\varphi(k)})_k$ such that for λ_S-almost every $A \in S_d(\mathbb{R})$,

$$W_1(t^A_\# \mu_{\varphi(k)}, t^A_\# \mu) \xrightarrow[k\to\infty]{} 0 \ . \tag{6.50}$$

As W_1 metrizes the weak convergence, this is equivalent to $t^A_\# \mu_{\varphi(k)} \xrightarrow[k\to\infty]{\mathcal{L}} t^A_\# \mu$. Then, by Levy's characterization theorem, this is equivalent to the pointwise convergence of the characterization function, i.e. *for all $t \in \mathbb{R}$, $\phi_{t^A_\# \mu_{\varphi(k)}}(t) \xrightarrow[k\to\infty]{} \phi_{t^A_\# \mu}(t)$. Moreover, we have for all $s \in \mathbb{R}$,*

$$\begin{aligned}
\phi_{t^A_\# \mu_{\varphi(k)}}(s) &= \int_{\mathbb{R}} e^{-its} \mathrm{d}(t^A_\# \mu_{\varphi(k)})(t) && (6.51)\\
&= \int_{S_d^{++}(\mathbb{R})} e^{-it^A(M)s}\, \mathrm{d}\mu_{\varphi(k)}(M) && (6.52)\\
&= \int_{S_d^{++}(\mathbb{R})} e^{-i\langle sA, \log M\rangle_F}\, \mathrm{d}\mu_{\varphi(k)}(M) && (6.53)\\
&= \int_{S_d(\mathbb{R})} e^{-i\langle sA, S\rangle_F}\, \mathrm{d}(\log_\# \mu_{\varphi(k)})(S) && (6.54)\\
&= \phi_{\log_\# \mu_{\varphi(k)}}(sA) && (6.55)\\
&\xrightarrow[k\to\infty]{} \phi_{\log_\# \mu}(sA) \ . && (6.56)
\end{aligned}$$

Then, working in $S_d(\mathbb{R})$ with the Frobenius norm, we can use the same proof of Nadjahi et al. [2020] by using a convolution with a gaussian kernel and show that it implies that $\log_\# \mu_{\varphi(k)} \xrightarrow[k\to\infty]{\mathcal{L}} \log_\# \mu$. Finally, let's show that it implies the weak convergence of $(\mu_{\varphi(k)})_k$ towards μ. Let $f \in C_b(S_d^{++}(\mathbb{R}))$, then

$$\begin{aligned}
\int_{S_d^{++}(\mathbb{R})} f\, \mathrm{d}\mu_{\varphi(k)} &= \int_{S_d(\mathbb{R})} f \circ \exp\, \mathrm{d}(\log_\# \mu_{\varphi(k)}) && (6.57)\\
&\xrightarrow[k\to\infty]{} \int_{S_d(\mathbb{R})} f \circ \exp\, \mathrm{d}(\log_\# \mu) && (6.58)\\
&= \int_{S_d^{++}(\mathbb{R})} f\, \mathrm{d}\mu \ . && (6.59)
\end{aligned}$$

Hence, we can conclude that $\mu_{\varphi(k)} \xrightarrow[k\to\infty]{\mathcal{L}} \mu$.

Main proof.
First, we suppose that $\mu_k \xrightarrow[k\to\infty]{\mathcal{L}} \mu$ *in* $\mathcal{P}_p(S_d^{++}(\mathbb{R}))$. *Then, by continuity, we have that for* λ_S *almost every* $A \in \mathcal{P}_p(S_d^{++}(\mathbb{R}))$, $t^A_\# \mu_k \xrightarrow[k\to\infty]{} t^A_\# \mu$. *Moreover, as the Wasserstein distance on* $\mathbb{R}$ *metrizes the weak convergence,* $W_p(t^A_\#\mu_k, t^A_\#\mu) \xrightarrow[k\to\infty]{} 0$. *Finally, as* W_p *is bounded and it converges for* λ_S*-almost every* A, *we have by the Lebesgue convergence dominated theorem that* $\mathrm{SPDSW}_p^p(\mu_k, \mu) \xrightarrow[k\to\infty]{} 0$.
On the other hand, suppose that $\mathrm{SPDSW}_p(\mu_k, \mu) \xrightarrow[k\to\infty]{} 0$. *We first adapt Lemma S1 of [Nadjahi et al., 2020] in the Lemma and observe that by the Hölder inequality,*

$$\mathrm{SPDSW}_1(\mu, \nu) \leq \mathrm{SPDSW}_p(\mu, \nu) \ , \tag{6.60}$$

and hence $\mathrm{SPDSW}_1(\mu_k, \mu) \xrightarrow[k\to\infty]{} 0$.
By the same contradiction argument as in Nadjahi et al. [2020], let's suppose that $(\mu_k)_k$ *does not converge to* μ. *Then, denoting* d_P *the Lévy-Prokhorov metric,* $\lim_{k\to\infty} d_P(\mu_k, \mu) \neq 0$. *Hence, there exists* $\epsilon > 0$ *and a subsequence* $(\mu_{\varphi(k)})_k$ *such that* $d_P(\mu_{\varphi(k)}, \mu) > \epsilon$ *for any* $k \in \mathbb{N}$. *Then, we have* $\lim_{k\to\infty} \mathrm{SPDSW}_1(\mu_{\varphi(k)}, \mu) = 0$. *Thus, by the Lemma, there exists a subsequence* $(\mu_{\psi(\varphi(k))})_k$ *such that* $\mu_{\psi(\varphi(k))} \xrightarrow[k\to\infty]{\mathcal{L}} \mu$ *which is equivalent to* $\lim_{k\to\infty} d_P(\mu_{\psi(\varphi(k))}, \mu) = 0$ *and contradicts the hypothesis.*
We conclude that $(\mu_k)_k$ *converges weakly to* μ.

Moreover, SPDSW_p and W_p – the p-Wasserstein distance with Log-Euclidean ground cost – are also weakly equivalent on compactly supported measures on $\mathcal{P}_p(S_d^{++}(\mathbb{R}))$, as demonstrated in Theorem 6.2.3.

THEOREM 6.2.3. *Let* $p \geq 1$, *let* $\mu, \nu \in \mathcal{P}_p(S_d^{++}(\mathbb{R}))$. *Then*

$$\mathrm{SPDSW}_p^p(\mu, \nu) \leq c_{d,p}^p W_p^p(\mu, \nu) \ ,$$

where $c_{d,p}^p = \frac{1}{d}\int \|\theta\|_p^p \ \mathrm{d}\lambda(\theta)$. *Let* $R > 0$ *and* $B(I, R) = \{A \in S_d^{++}(\mathbb{R}),\ d_{LE}(A, I_d) = \|\log A\|_F \leq R\}$ *be a closed ball. Then there exists a constant* $C_{d,p,R}$ *such that for all* $\mu, \nu \in \mathcal{P}_p(B(I, R))$,

$$W_p^p(\mu, \nu) \leq C_{d,p,R} \mathrm{SPDSW}_p(\mu, \nu)^{\frac{2}{d(d+1)+2}} \ .$$

Proof 6.2.3
See the paper for the proof.

The theorems above highlight that $\mathrm{SPDSW}_\mathrm{p}$ behaves similarly to W_p on $\mathcal{P}_p(S_d^{++}(\mathbb{R}))$. Thus, it is justified to use $\mathrm{SPDSW}_\mathrm{p}$ as a surrogate of Wasserstein and take advantage of the statistical and computational benefits that we present now.

Statistical properties.

In practice, we approximate SPDSW using the plug-in estimator [Niles-Weed and Rigollet,

2022, Manole et al., 2022], *i.e.* for $\mu, \nu \in \mathcal{P}_p(S_d^{++}(\mathbb{R}))$, we approximate $\text{SPDSW}_p^p(\mu, \nu)$ by $\text{SPDSW}_p^p(\hat{\mu}_n, \hat{\nu}_n)$ where $\hat{\mu}_n$ and $\hat{\nu}_n$ denote empirical distributions of μ and ν. Hence, we are interested in the speed of convergence towards $\text{SPDSW}_p^p(\mu, \nu)$, which we call the sample complexity. We derive the convergence rate for SPDSW in Proposition 6.2.4, relying on the proof of Nadjahi et al. [2020] and on the sample complexity of the Wasserstein distance [Fournier and Guillin, 2015]. The sample complexity we find does not depend on the dimension, which is an important property of sliced divergences [Nadjahi et al., 2020].

PROPOSITION 6.2.4. *Let $q > p \geq 1$, $\mu, \nu \in \mathcal{P}_p(S_d^{++}(\mathbb{R}))$, and $\hat{\mu}_n, \hat{\nu}_n$ the associated empirical measures. We define the moment of order q by $M_q(\mu) = \int \|X\|_F^q \, \mathrm{d}\mu(X)$, and $M_q(\mu, \nu) = M_q(\log_\# \mu)^{1/q} + M_q(\log_\# \nu)^{1/q}$. Then, there exists a constant $C_{p,q}$ depending only on p and q such that*

$$\mathbb{E}\big[|\text{SPDSW}_p(\hat{\mu}_n, \hat{\nu}_n) - \text{SPDSW}_p(\mu, \nu)|\big] \leq \alpha_{n,p,q} C_{p,q}^{1/p} M_q(\mu, \nu) \ ,$$

$$\text{where} \quad \alpha_{n,p,q} = \begin{cases} n^{-1/(2p)} & \text{if } q > 2p \\ n^{-1/(2p)} \log(n)^{1/p} & \text{if } q = 2p \\ n^{-(q-p)/(pq)} & \text{if } q \in (p, 2p) \ . \end{cases}$$

Proof 6.2.4

See the paper for the proof.

Proposition 6.2.4 assumes we can exactly compute the outer integral, which is not the case in practice, as it requires a Monte-Carlo approximation. In Proposition 6.2.5, we show that, L being the number of projections, the Monte-Carlo error is $O(\frac{1}{\sqrt{L}})$ for a fixed dimension d. This time, the dimension intervenes in $\text{Var}_{A \sim \lambda_S}\left[W_p^p(t_\#^A \mu, t_\#^A \nu)\right]$.

PROPOSITION 6.2.5. *Let $p \geq 1$, $\mu, \nu \in \mathcal{P}_p(S_d^{++}(\mathbb{R}))$. Then, the error made by the Monte Carlo estimate of SPDSW_p with L projections can be bounded as follows*

$$\mathbb{E}_A\left[|\widehat{\text{SPDSW}}_{p,L}^p(\mu, \nu) - \text{SPDSW}_p^p(\mu, \nu)|\right]^2 \leq \frac{1}{L} \text{Var}_{A \sim \lambda_S}\left[W_p^p(t_\#^A \mu, t_\#^A \nu)\right] \ ,$$

where $\widehat{\text{SPDSW}}_{p,L}^p(\mu, \nu) = \frac{1}{L} \sum_{i=1}^L W_p^p(t_\#^{A_i} \mu, t_\#^{A_i} \nu)$ with $(A_i)_{i=1}^L$ independent samples from λ_S.

Proof 6.2.5

See the paper for the proof.

Computational complexity and implementation.

Let $\mu, \nu \in \mathcal{P}_p(S_d^{++}(\mathbb{R}))$ and $(X_i)_{i=1}^n$ (resp. $(Y_j)_{j=1}^m$) samples from μ (resp. from ν). We approximate $\text{SPDSW}_p^p(\mu, \nu)$ by $\widehat{\text{SPDSW}}_{p,L}^p(\hat{\mu}_n, \hat{\nu}_m)$ where $\hat{\mu}_n = \frac{1}{n} \sum_{i=1}^n \delta_{X_i}$ and $\hat{\nu}_m = \frac{1}{m} \sum_{j=1}^m \delta_{Y_j}$. Sampling from λ_O requires drawing a matrix $Z \in \mathbb{R}^{d \times d}$ with i.i.d normally distributed coefficients, and then taking the QR factorization with positive entries on the

Algorithm 6 Computation of SPDSW

Input: $(X_i)_{i=1}^n \sim \mu$, $(Y_j)_{j=1}^m \sim \nu$, L the number of projections, p the order
for $\ell = 1$ **to** L **do**
  Draw $\theta \sim \mathrm{Unif}(S^{d-1}) = \lambda$
  Draw $P \sim \mathrm{Unif}(O_d(\mathbb{R})) = \lambda_O$
  $A = P\mathrm{diag}(\theta)P^T$
  $\forall i, j,\ \hat{X}_i^\ell = t^A(X_i),\ \hat{Y}_j^\ell = t^A(Y_j)$
  Compute $W_p^p(\frac{1}{n}\sum_{i=1}^n \delta_{\hat{X}_i^\ell}, \frac{1}{m}\sum_{j=1}^m \delta_{\hat{Y}_j^\ell})$
end for
Return $\frac{1}{L}\sum_{\ell=1}^L W_p^p(\frac{1}{n}\sum_{i=1}^n \delta_{\hat{X}_i^\ell}, \frac{1}{m}\sum_{j=1}^m \delta_{\hat{Y}_j^\ell})$

diagonal of R [Mezzadri, 2006], which needs $O(d^3)$ operations [Golub and Van Loan, 2013, Section 5.2]. Then, computing n matrix logarithms takes $O(nd^3)$ operations. Given L projections, the inner-products require $O(Lnd^2)$ operations, and the computation of the one-dimensional Wasserstein distances is done in $O(Ln\log n)$ operations. Therefore, the complexity of SPDSW is $O(Ln(\log n + d^2) + (L+n)d^3)$. The procedure is detailed in Algorithm 6. In practice, when it is required to call SPDSW several times in optimization procedures, the computational complexity can be reduced by drawing projections only once at the beginning.

Note that it is possible to draw symmetric matrices with complexity $O(d^2)$ by taking $A = \frac{Z+Z^T}{\|Z+Z^T\|_F}$. Although this is a great advantage from the point of view of computation time, we leave it as an open question to know whether this breaks the bounds in Theorem 6.2.3.

We illustrate the computational complexity *w.r.t* samples in Figure 6.2. We compare the runtime to the Wasserstein distance with Affine-Invariant (AIW) and Log-Euclidean (LEW) metrics, and to Sinkhorn algorithm (LES) which is a classical alternative to Wasserstein to reduce the computational cost. When enough samples are available, then computing the Wasserstein distance takes more time than computing the cost matrix, and SPDSW is fast to compute.

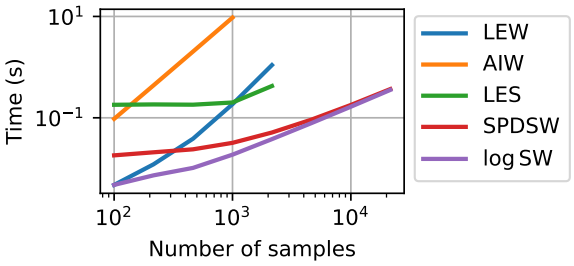

Figure 6.2: Runtime in log-log scale of SPDSW *and* log SW *(200 proj., d=20) compared to alternatives based on Wasserstein between Wishart samples. Sliced discrepancies can scale to larger distributions in* $S_d^{++}(\mathbb{R})$.

6.3 . From brain data to distributions in $S_d^{++}(\mathbb{R})$

M/EEG data consists of multivariate time series $X \in \mathbb{R}^{N_C \times T}$, with N_C channels, and T time samples. A widely adopted model assumes that the measurements X are linear combinations of N_S sources $S \in \mathbb{R}^{N_S \times T}$ degraded by noise $N \in \mathbb{R}^{N_C \times T}$. This leads to $X = AS + N$, where $A \in \mathbb{R}^{N_C \times N_S}$ is the forward linear operator [Hämäläinen et al., 1993]. A common practice in statistical learning on M/EEG data is to consider that the target is a function of the power of the sources, *i.e.* $\mathbb{E}[SS^T]$ [Blankertz et al., 2007, Dähne et al.,

2014, Sabbagh et al., 2019]. In particular, a broad range of methods rely on second-order statistics of the measurements, *i.e.* covariance matrices of the form $C = \frac{XX^T}{T}$, which are less costly and uncertain than solving the inverse problem to recover S before training the model. After proper rank reduction to turn the covariance estimates into SPD matrices [Harandi et al., 2017], and appropriate band-pass filtering to stick to specific physiological patterns Blankertz et al. [2007], Riemannian geometry becomes an appropriate tool to deal with such data.

In this section, we propose two applications of SPDSW to prediction tasks from M/EEG data. More specifically, we introduce a new method to perform brain-age regression, building on the work of Sabbagh et al. [2019] and Meunier et al. [2022], and another for domain adaptation in BCI.

Distributions regression for brain-age prediction.

Learning to predict brain age from population-level neuroimaging data-sets can help characterize biological aging and disease severity [Spiegelhalter, 2016, Cole and Franke, 2017, Cole et al., 2018]. Thus, this task has encountered more and more interest in the neuroscience community in recent years [Xifra-Porxas et al., 2021, Peng et al., 2021, Engemann et al., 2022]. In particular, Sabbagh et al. [2019] take advantage of Riemannian geometry for feature engineering and prediction with the following steps. First, one covariance estimate is computed per frequency band from each subject recording. Then these covariance matrices are projected onto a lower dimensional space to make them full rank, for instance with a PCA. Each newly obtained SPD matrix is projected onto the log space to obtain a feature after vectorization and aggregation among frequency bands. Finally, a Ridge regression model predicts brain age. This white-box method achieves state-of-the-art brain age prediction scores on MEG datasets like Cam-CAN [Taylor et al., 2017].

MEG recordings as distributions of covariance matrices. Instead of modeling each frequency band by a unique covariance matrix, we propose to use a distribution of covariance matrices estimated from small time frames. Concretely, given a time series $X \in \mathbb{R}^{N_C \times T}$ and a time-frame length $t < T$, a covariance matrix is estimated from each one of the $n = \lfloor \frac{T}{t} \rfloor$ chunks of signal available. This process models each subject by as many empirical distributions of covariance estimates $(C_i)_{i=1}^n$ as there are frequency bands. Then, all samples are projected on a lower dimensional space with a PCA, as done in Sabbagh et al. [2019]. Here, we study whether modeling a subject by such distributions provides additional information compared to feature engineering based on a unique covariance matrix. In order to perform brain age prediction from these distributions, we extend recent results on distribution regression with SW kernels [Kolouri et al., 2016, Meunier et al., 2022] to SPD matrices, and show that SPDSW performs well on this prediction task while being easy to implement.

SPDSW **kernels for distributions regression.** As shown in section 6.2, SPDSW is a well-defined distance on distributions in $S_d^{++}(\mathbb{R})$. The most straightforward way to build a kernel from this distance is to resort to well-known Gaussian kernels, *i.e.* $K(\mu, \nu) = e^{-\frac{1}{2\sigma^2}\mathrm{SPDSW}_2^2(\mu,\nu)}$.

However, this is not sufficient to make it a proper positive kernel. Indeed, we need SPDSW

to be a Hilbertian distance [Hein and Bousquet, 2005]. A pseudo-distance d on $\mathcal{X}$ is Hilbertian if there exists a Hilbert space $\mathcal{H}$ and a feature map $\Phi : \mathcal{X} \to \mathcal{H}$ such that $\forall x, y \in \mathcal{X}, d(x, y) = \|\Phi(x) - \Phi(y)\|_{\mathcal{H}}$. We now extend Meunier et al. [2022, Proposition 5] to the case of SPDSW in Proposition 6.3.1.

PROPOSITION 6.3.1. *Let m be the Lebesgue measure and let $\mathcal{H} = L^2([0,1] \times S_d(\mathbb{R}), m \otimes \lambda_S)$. We define Φ as*

$$\begin{aligned} \Phi : \mathcal{P}_p(S_d^{++}(\mathbb{R})) &\to \mathcal{H} \\ \mu &\mapsto \left((q, A) \mapsto F^{-1}_{t^A_\# \mu}(q)\right) \ , \end{aligned}$$

where $F^{-1}_{t^A_\# \mu}$ is the quantile function of $t^A_\# \mu$. Then, SPDSW_2 is Hilbertian and for all $\mu, \nu \in \mathcal{P}_p(S_d^{++}(\mathbb{R}))$,

$$\mathrm{SPDSW}_2^2(\mu, \nu) = \|\Phi(\mu) - \Phi(\nu)\|_{\mathcal{H}}^2 \ .$$

Proof 6.3.1

Let μ, ν be probability distributions on $S_d^{++}(\mathbb{R})$ with moments of order $p \geq 1$. Then

$$\begin{aligned} \mathrm{SPDSW}_2^2(\mu, \nu) &= \int_{S_d} \|F^{-1}_{t^A_\# \mu} - F^{-1}_{t^A_\# \nu}\|^2 \, \mathrm{d}\lambda_S(A) \\ &= \int_{S_d} \int_0^1 \left(F^{-1}_{t^A_\# \mu}(q) - F^{-1}_{t^A_\# \nu}(q)\right)^2 \mathrm{d}q \mathrm{d}\lambda_S(A) \\ &= \|\Phi(\mu) - \Phi(\nu)\|_{\mathcal{H}}^2 \ . \end{aligned}$$

Thus, SPDSW_2 is Hilbertian.

The proof is similar to the one of Meunier et al. [2022] for SW in Euclidean spaces and highlights two key results. The first one is that SPDSW extensions of Gaussian kernels are valid positive definite kernels, as opposed to what we would get with the Wasserstein distance [Meunier et al., 2022]. The second one is that we have access to an explicit and easy-to-compute feature map that preserves SPDSW, making it possible to avoid inefficient quadratic algorithms on empirical distributions from very large data. In practice, we rely on the finite-dimensional approximation of projected distributions quantile functions proposed in Meunier et al. [2022] to compute the kernels more efficiently with the ℓ_2-norm. Then, we leverage Kernel Ridge regression for prediction [Murphy, 2012]. Let $0 < q_1 < \cdots < q_M < 1$, and $(A_1, \ldots, A_L) \in S_d(\mathbb{R})^L$. The approximate feature map has a closed-form expression in the case of empirical distributions and is defined as

$$\hat{\Phi}(\mu) = \left(\frac{1}{\sqrt{ML}} F^{-1}_{t^{A_i}_\# \mu}(q_j)\right)_{1 \leq j \leq M, 1 \leq i \leq L} \ . \tag{6.61}$$

Regarding brain-age prediction, we model each couple of subject s and frequency band f as an empirical distribution $\mu_n^{s,f}$ of covariance estimates $(C_i)_{i=1}^n$. Hence, our data-set consists of the set of distributions in $S_d^{++}(\mathbb{R})$

$$\left(\mu_n^{s,f} = \frac{1}{n} \sum_{i=1}^n \delta_{C_i}\right)_{s,f} \ . \tag{6.62}$$

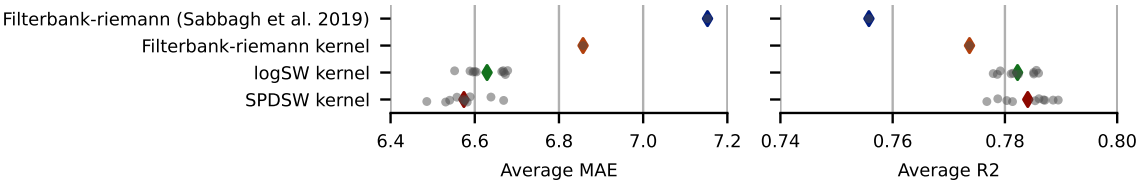

Figure 6.3: Average MAE and R^2 score for 10 random seeds on the Cam-CAN data-set with time-frames of 2s and 1000 projections. Kernel Ridge regression based on SW *kernels performs best.* SPDSW *and* log SW *are close to each other. Sampling from symmetric matrices offers a slight advantage but does not play a key role on performance. For information, Euclidean* SW *led to poor results on the task (MAE 9.7).*

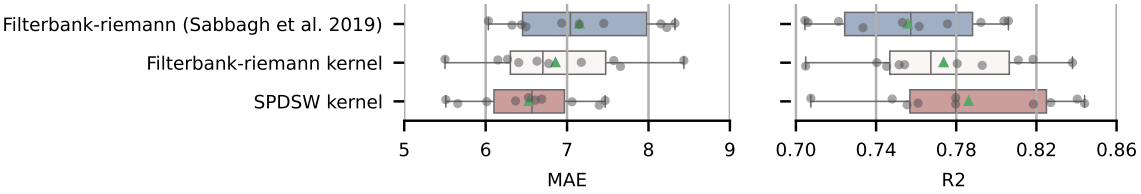

Figure 6.4: Results of 10-folds cross validation on the Cam-CAN data-set for one random seed. We display the Mean Absolute Error (MAE) and the R^2 coefficient. SPDSW*, with time-frames of 2s and 1000 projections, performs best. Note that Kernel Ridge regression based on the Log-Euclidean distance performs better than Ridge regression.*

First, we compute the associated features $(\hat{\Phi}(\mu_n^{s,f}))_{s,f}$ by loading the data and band-pass filtering the signal once per subject. Then, as we are interested in comparing each subject in specific frequency bands, we compute one approximate kernel matrix per frequency f, as follows

$$K_{i,j}^f = e^{-\frac{1}{2\sigma^2}\|\hat{\Phi}(\mu_n^{i,f}) - \hat{\Phi}(\mu_n^{j,f})\|_2^2} \quad . \tag{6.63}$$

Finally, the kernel matrix obtained as a sum over frequency bands, *i.e.* $K = \sum_f K^f$, is plugged into the Kernel Ridge regression of `scikit-learn` [Pedregosa et al., 2011].

Numerical results. We demonstrate the ability of our algorithm to perform well on brain-age prediction on the largest publicly available MEG data-set Cam-CAN Taylor et al. [2017], which contains recordings from 646 subjects at rest. We take advantage of the benchmark provided by Engemann et al. [2022] – available online[1] – to replicate the same pre-processing and prediction steps from the data, and thus produce a meaningful and fair comparison.

For each one of the seven frequency bands, we divide every subject time series into frames of fixed length. We estimate covariance matrices from each timeframe with OAS [Chen et al., 2010] and apply PCA for rank-reduction, as in Sabbagh et al. [2019], to obtain SPD matrices of size 53×53. This leads to distributions of 275 points per subject and per frequency band. In Sabbagh et al. [2019], the authors rely on Ridge regression on vectorized projections of SPD matrices on the tangent space. We also provide a comparison to Kernel Ridge regression based on a kernel with the Log-Euclidean metric, *i.e.* $K_{i,j}^{\log} = e^{-\frac{1}{2\sigma^2}\|\log C_i - \log C_j\|_F^2}$.

[1] https://github.com/meeg-ml-benchmarks/brain-age-benchmark-paper

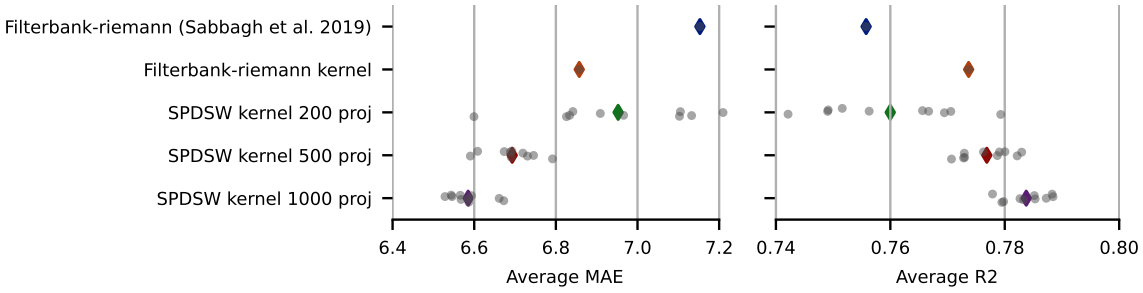

Figure 6.5: Average results for 10 random seeds with 200, 500 and 1000 projections for SPDSW *compared to average MAE and R^2 obtained with Ridge and Kernel Ridge regression on features from covariance estimates [Sabbagh et al., 2019]. With enough projections,* SPDSW *kernel does not suffer from variance and performs best.*

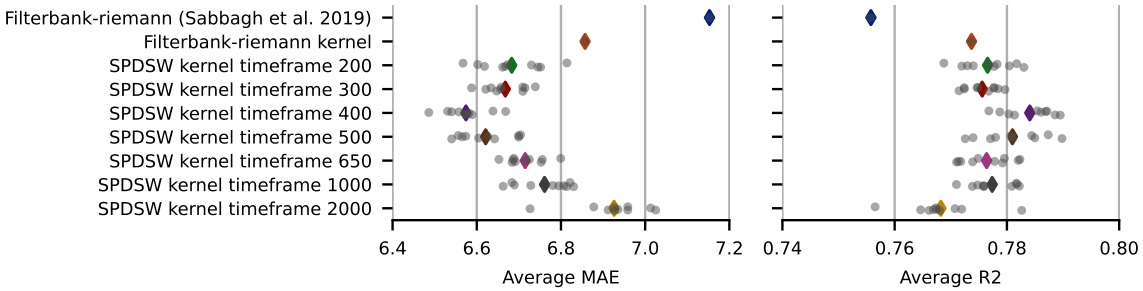

Figure 6.6: Average MAE and R^2 score on brain age regression with different time-frame lengths for 10 random seeds The performance depends on the time-frame length, and there is a trade-off to find between number of samples and noise in the samples.

Figure 6.3 shows that SPDSW (1000 projections, time-frames of 2s) performs best in average on 10-folds cross-validation for 10 random seeds, compared to the baseline with Ridge regression [Sabbagh et al., 2019] and to Kernel Ridge regression based on the Log-Euclidean metric, with identical pre-processing. We provide more details on scores for each fold on a single random seed in Figure 6.4. In particular, it seems that evaluating the distance between distributions of covariance estimates instead of just the average covariance brings more information to the model in this brain-age prediction task, and allows to improve the score. Also note that Log-Euclidean Kernel Ridge regression works better than the baseline method based on Ridge regression [Sabbagh et al., 2019]. Then, Figure 6.5 in the appendix shows that SPDSW does not suffer from variance with more than 500 projections in this use case with matrices of size 53×53. Finally, Figure 6.6 shows that there is a trade-off to find between smaller time-frames for more samples per distribution and larger time-frames for less noise in the covariance estimates and that this is an important hyper-parameter of the model.

Domain adaptation for BCI.

BCI consists of establishing a communication interface between the brain and an external device, in order to assist or repair sensory-motor functions [Daly and Wolpaw, 2008, Nicolas-Alonso and Gomez-Gil, 2012, Wolpaw, 2013]. The interface should be able to correctly interpret M/EEG signals and link them to actions that the subject would like to perform. One challenge of BCI is that ML methods are generally not robust to the change

Table 6.1: Accuracy and Runtime for Cross Session.

Subjects	Source	AISOTDA [Yair et al., 2019]	SPDSW	LogSW	LEW	LES	SPDSW	LogSW	LEW	LES
			Transformations in $S_d^{++}(\mathbb{R})$				Descent over particles			
1	82.21	80.90	84.70	84.48	84.34	84.70	85.20	85.20	77.94	82.92
3	79.85	87.86	85.57	84.10	85.71	86.08	87.11	86.37	82.42	81.47
7	72.20	82.29	81.01	76.32	81.23	81.23	81.81	81.73	79.06	73.29
8	79.34	83.25	83.54	81.03	82.29	83.03	84.13	83.32	80.07	85.02
9	75.76	80.25	77.35	77.88	77.65	77.65	80.30	79.02	76.14	70.45
Avg. acc.	77.87	82.93	82.43	80.76	82.24	82.54	83.71	83.12	79.13	78.63
Avg. time (s)	-	-	**4.34**	**4.32**	11.41	12.04	**3.68**	**3.67**	8.50	11.43

of data domain, which means that an algorithm trained on a particular subject will not be able to generalize to other subjects. Domain adaptation (DA) [Ben-David et al., 2006] offers a solution to this problem by taking into account the distributional shift between source and target domains. Classical DA techniques employed in BCI involve projecting target data on source data or vice versa, or learning a common embedding that erases the shift, sometimes with the help of optimal transport [Courty et al., 2016]. As Riemannian geometry works well on BCI [Barachant et al., 2013], DA tools have been developed for SPD matrices [Yair et al., 2019, Ju and Guan, 2022].

SPDSW for domain adaptation on SPD matrices. We study two training frameworks on data from $\mathcal{P}(S_d^{++}(\mathbb{R}))$. In the first case, a push forward operator f_θ is trained to change a distribution μ_S in the source domain into a distribution μ_T in the target domain by minimizing a loss of the form $L(\theta) = \mathcal{L}\big((f_\theta)_{\#}\mu_S, \mu_T\big)$, where $\mathcal{L}$ is a transport cost like Wasserstein on $\mathcal{P}(S_d^{++}(\mathbb{R}))$ or SPDSW. The model f_θ is a sequence of simple transformations in $S_d^{++}(\mathbb{R})$ [Rodrigues et al., 2018], *i.e.* $T_W(C) = W^T C W$ for $W \in S_d^{++}(\mathbb{R})$ (translations) or $W \in \mathrm{SO}_d$ (rotations), potentially combined to specific non-linearities [Huang and Van Gool, 2017]. The advantage of such models is that they provide a high level of structure with a small number of parameters.

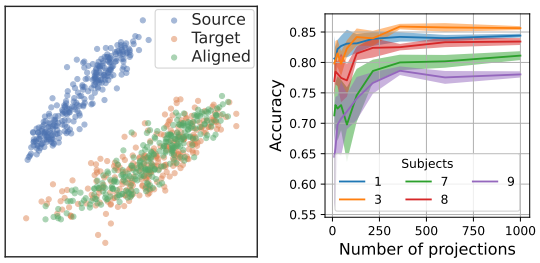

*Figure 6.7: (**Left**) PCA on BCI data before and after alignment. Minimizing* SPDSW *with enough projections allows aligning sources on targets. (**Right**) Accuracy* w.r.t *num. of projections for the cross-session task with transformations. Here, there is no need for too many projections to converge.*

In the second case, we directly align the source on the target by minimizing $\mathcal{L}$ with a Riemannian gradient descent directly over the particles [Boumal, 2020], *i.e.* by denoting $\mu_S((x_i)_{i=1}^{|X_S|}) = \frac{1}{|X_S|}\sum_{i=1}^{|X_S|}\delta_{x_i}$ with $X_S = \{x_i^S\}_i$ the samples of the source, we initialize at $(x_i^S)_{i=1}^{|X_S|}$ and minimize $L((x_i)_{i=1}^{|X_S|}) = \mathcal{L}\left(\mu_S((x_i)_{i=1}^{|X_S|}), \mu_T\right)$.

We use `Geoopt` [Kochurov et al., 2020] and `Pytorch` [Paszke et al., 2017] to optimize on manifolds. Then, an SVM is trained on the vectorized projections of X_S in the log space, *i.e.* from couples $(\mathrm{vect}(\log x_i^S), y_i)_{i=1}^{|X_S|}$, and we evaluate the model on the target distribution.

Table 6.2: Accuracy and Runtime for Cross Subject.

Subjects	Source	AISOTDA [Yair et al., 2019]	SPDSW	LogSW	LEW	LES	SPDSW	LogSW	LEW	LES
			Transformations in $S_d^{++}(\mathbb{R})$				Descent over particles			
1	42.09	62.94	61.91	60.50	62.89	63.64	62.56	61.91	62.84	63.25
3	35.62	71.01	66.40	66.53	66.34	66.30	65.74	64.96	60.27	62.29
7	39.52	63.98	60.42	57.29	60.89	60.43	60.97	58.49	53.18	59.52
8	42.90	66.06	61.09	60.19	61.29	62.14	60.95	60.00	61.68	61.77
9	29.94	59.18	53.31	50.63	54.79	54.89	58.72	54.91	58.22	64.90
Avg. acc.	38.01	64.43	60.63	59.03	61.24	61.48	61.79	60.05	59.24	62.55
Avg. time	-	-	**4.34**	**4.31**	11.76	11.21	**3.67**	**3.64**	9.54	10.32

Numerical results. In Table 6.1, we focus on cross-session classification for the BCI IV 2.a Competition dataset [Brunner et al., 2008] with 4 target classes and about 270 samples per subject and session. The cross-subject counterpart is detailed in Table 6.2. We compare accuracies and runtimes for several methods run on a GPU Tesla V100-DGXS-32GB. The distributions are aligned by minimizing different discrepancies, namely SPDSW, logSW, Log-Euclidean Wasserstein (LEW) and Sinkhorn (LES), computed with `POT` [Flamary et al., 2021]. Note that we did not tune hyper-parameters on each particular subject and discrepancy, but only used a grid search to train the SVM on the source dataset, and optimized each loss until convergence, *i.e.* without early stopping. We compare this approach to the naive one without DA, and to the barycentric OTDA [Courty et al., 2016] with Affine-Invariant metric reported from Yair et al. [2019]. Our results show that all discrepancies give equivalent accuracies. As expected, SPDSW has an advantage in terms of computation time compared to other transport losses. Moreover, transformations in $S_d^{++}(\mathbb{R})$ and descent over the particles work almost equally well in the case of SPDSW. We illustrate the alignment we obtain by minimizing SPDSW in Figure 6.7, with a PCA for visualization purposes. Additionally, Figure 6.7 shows that SPDSW does not need too many projections to reach optimal performance.

Conclusion

We proposed SPDSW, a discrepancy between distributions of SPD matrices with appealing properties such as being a distance and metrizing the weak convergence. Being a Hilbertian metric, it can be plugged as is into Kernel methods, as we demonstrate for brain age prediction from MEG data. Moreover, it is usable in loss functions dealing with distributions of SPD matrices, for instance in domain adaptation for BCI, with less computational complexity than its counterparts. Beyond M/EEG data, our discrepancy is of interest for any learning problem that involves distributions of SPD matrices, and we expect to see other applications of SPDSW in the future. One might also be interested in using other metrics on positive definite or semi-definite matrices such as the Bures-Wasserstein metric, with the additional challenges that this space is positively curved and not geodesically complete [Thanwerdas and Pennec, 2023].

Conclusion and perspectives

The starting point of the thesis was the study of Unrolling and automatic differentiation for Dictionary Learning, and their potential usage to solve inverse problems. At that time, as we were intrigued by the success of Unrolling in reconstruction, and also in gradient computations for smooth loss functions, we were wondering whether this technique may allow to speed up the computation of the dictionary, and why not, learn a better one. Then, the idea was to study how this technique could be used for prior learning in inverse problems, and apply it to M/EEG data.

However, the study on Unrolled Dictionary Learning summarized in chapter 2 highlighted the limitations of automatic differentiation for non smooth losses and non smooth algorithms. We now know that it does not allow to improve the results as we hoped at the beginning, and strangely enough, can even worsen them. But what we got instead was a new point of view of the Dictionary Learning problem that suits applications. Indeed, if the only variable that matters in optimization is the dictionary itself, then we do not have to care about precise or complete sparse coding in the learning process, and this can save a great amount of computation time. Then, if the sparse codes are needed, they can be computed once and for all at the end of the process when the dictionary is known, and we demonstrated the efficiency of this approach on MEG signals in chapter 5.

The second line of research that guided this thesis was the study of prior learning techniques for inverse problems resolution. As works focusing on supervised inverse problems predominate the literature, we first decided to deal with unsupervised inverse problems, which naturally arise in neuroscience and M/EEG. In chapter 3, we analyzed what was and was not possible to do through the lens of Dictionary Learning, and we ended up with the conclusion that expert knowledge on the signal and on the problem structure, conditioned by the measurement operator, can allow designing parameterized models able to learn all information necessary for reconstruction from observed data only. But the main condition is that prior knowledge on the data should not be correlated to the structure of the measurement operator, as is the case of shift-invariant signals in deblurring, and this makes it extremely difficult to learn relevant information in the kernel space of the operator. We also compared the behavior of Unrolled Dictionary Learning based on Analysis and Synthesis in practice on several inverse problems and with several constraints on the

Dictionary, and illustrated how the structure of the prior learning algorithm can impact the performance.

Then, we turned to self-supervised inverse problems and studied prior learning based on Unrolled Dictionary Learning to build Plug-and-Play estimators in chapter 4. The idea was to provide theoretical elements to help practitioners that use Unrolling in Plug-and-Play algorithms understand the similarities with classical inverse problem resolution methods based on Dictionary Learning. We were able to prove the stability of these approaches and their equivalence to primal dual and sparse coding optimization algorithms, and illustrate that they work well in imaging reconstruction problems.

Finally, we contributed to M/EEG signal processing by developing a method – explained in chapter 6 – to predict quantities of interest from M/EEG recordings and compare subjects between each other with distributions of covariance matrices. This allows making predictions without having to solve the source localization inverse problem.

Besides the contributions and answers provided in this thesis, our work has raised multiple questions that would deserve to be studied in more details in the future.

How to quantify the difficulty of an inverse problem ? The work on prior learning in unsupervised inverse problems presented in chapter 3 shows that there is an intrinsic limit to what we can expect to learn from observed data, and thus to the performance of reconstruction algorithms without expert knowledge in this context. While this may seem natural in an unsupervised scenario, it also applies to supervised scenarios where we have access to a training dataset of clean samples and to the measurement operator. As an example, it is possible to compute a lower bound of the optimal performance of reconstruction algorithms with the help of entropy estimators [Michel et al., 2023]. This quantity acts as a difficulty measure for an inverse problem defined as the combination of a data distribution, a measurement operator, and a noise distribution.

Thus, this metric might be used to identify areas of the signal domain from which all information is lost by the measurement operator, depending on the distribution of the data, so that specific expert knowledge can be inserted into the model to fill the gaps in prior learning. Another application would be to quantitatively assess the reconstruction performance of an algorithm relatively to the optimal value, and thus know to what extent it is worth spending time improving reconstruction on a particular problem and know in what part of signal domain the method succeeds or fails.

Is solving the inverse problem helpful for prediction ? In M/EEG, prediction tasks like brain-age prediction, detection of dementia or sleep stage classification rely on the output of the few hundreds of sensors that measure the electromagnetic activity around the head. Yet, the data are only degraded observations of the cerebral activity inside the head, observed through a physical model induced by Maxwell's equations which involves a heavy dimension reduction of the signal and which is subject-dependent. The question then arises as to whether finding the sources from the observations and the linear model, *i.e.* solving the source localization inverse problem, would improve performances in prediction tasks.

This question arises in multiple domains and is closely related to prediction from missing values [Le Morvan et al., 2021]. Indeed, the operator erases information in specific areas of the signal, and it would be of interest to know whether solving the related inverse problem might help improving prediction performance depending on the data distribution and the properties of the forward model. Moreover, turning the problem upside-down, prediction could serve as a performance metric for prior learning methods or reconstruction algorithms, and could even be used for hyper-parameter selection in unsupervised contexts. Taking the example of source localization in M/EEG, proxy problems like brain-age prediction are good candidates to assess whether a reconstruction algorithm works well or not.

Perspectives of Convolutional Dictionary Learning in M/EEG. The aim of speeding up CDL on M/EEG data was first and foremost to develop tools to process datasets at the population level. Yet, it is not straightforward to compare dictionaries between each other, or to learn a common dictionary from a population of subjects that takes into account variations that may appear in the signal, in order to draw interesting conclusions from the data. In addition, CDL could also be used in prediction from M/EEG data. Indeed, recent works have shown that Dictionary Learning could be used as an embedding method in self-supervised learning [Chen et al., 2023], and it would be of interest to extend this method to CDL in M/EEG and see whether the related embedding obtained from sparse coding are well-performing.

Perspectives of kernel methods and distribution regression in M/EEG. The work on SPDSW, and in particular its application in M/EEG for brain age prediction, opens up new possibilities to predict from M/EEG data. First, the idea of processing frequency bands in different kernels before summing them up can easily be extended with Multiple Kernel Learning [Gönen and Alpaydın, 2011], where each kernel would be weighted by a coefficient to be determined, $i.e\ K = \sum_f \beta_f K_f$. This would provide a framework to compare the contribution of each frequency band to the final result, and hopefully help improving the performance of the naive average. In addition, the idea of distribution regression can be developed further by studying to what extent seeing M/EEG data as distributions allows to model phenomena that a simple average can't model, and thus recover information that was lost with state of the art techniques. As an example, this could be the case for non stationary signals.

Bibliography

Pierre Ablin, Thomas Moreau, Mathurin Massias, and Alexandre Gramfort. Learning step sizes for unfolded sparse coding. In *Advances in Neural Information Processing Systems*, pages 13100–13110, 2019.

Pierre Ablin, Gabriel Peyré, and Thomas Moreau. Super-efficiency of automatic differentiation for functions defined as a minimum. In *Proceedings of the 37th International Conference on Machine Learning*, pages 32–41, 2020.

Alekh Agarwal, Animashree Anandkumar, Prateek Jain, and Praneeth Netrapalli. Learning sparsely used overcomplete dictionaries via alternating minimization. *SIAM Journal on Optimization*, 26(4):2775–2799, 2016.

Michal Aharon, Michael Elad, and Alfred Bruckstein. K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *Signal Processing, IEEE Transactions on*, 54:4311 – 4322, 2006. doi: 10.1109/TSP.2006.881199.

Cédric Allain, Alexandre Gramfort, and Thomas Moreau. Dripp: Driven point processes to model stimuli induced patterns in m/eeg signals. *arXiv preprint arXiv:2112.06652*, 2021.

David Alvarez-Melis, Youssef Mroueh, and Tommi Jaakkola. Unsupervised hierarchy matching with optimal transport over hyperbolic spaces. In *International Conference on Artificial Intelligence and Statistics*, pages 1606–1617. PMLR, 2020.

Farhad Pourkamali Anaraki and Shannon M Hughes. Compressive k-svd. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5469–5473. IEEE, 2013.

Sanjeev Arora, Rong Ge, Tengyu Ma, and Ankur Moitra. Simple, efficient, and neural algorithms for sparse coding. In *Conference on learning theory*, pages 113–149. PMLR, 2015.

Vincent Arsigny, Pierre Fillard, Xavier Pennec, and Nicholas Ayache. *Fast and Simple Computations on Tensors with Log-Euclidean Metrics.* PhD thesis, INRIA, 2005.

Vincent Arsigny, Pierre Fillard, Xavier Pennec, and Nicholas Ayache. Log-euclidean metrics for fast and simple calculus on diffusion tensors. *Magnetic Resonance in Medicine: An Official Journal of the International Society for Magnetic Resonance in Medicine*, 56(2): 411–421, 2006.

Alexandre Barachant, Stéphane Bonnet, Marco Congedo, and Christian Jutten. Multiclass brain–computer interface classification by riemannian geometry. *IEEE Transactions on Biomedical Engineering*, 59(4):920–928, 2011.

Alexandre Barachant, Stéphane Bonnet, Marco Congedo, and Christian Jutten. Classification of covariance matrices using a riemannian-based kernel for bci applications. *Neurocomputing*, 112:172–178, 2013.

Heinz H Bauschke, Patrick L Combettes, et al. *Convex analysis and monotone operator theory in Hilbert spaces*, volume 408. Springer, 2011.

Erhan Bayraktar and Gaoyue Guo. Strong equivalence between metrics of wasserstein type. *Electronic Communications in Probability*, 26:1–13, 2021.

Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Imaging Sciences*, 2:183–202, 2009.

Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. Analysis of representations for domain adaptation. In *Advances in Neural Information Processing Systems*, volume 19. MIT Press, 2006.

Carla Bertocchi, Emilie Chouzenoux, Marie-Caroline Corbineau, Jean-Christophe Pesquet, and Marco Prato. Deep unfolding of a proximal interior point method for image restoration. *Inverse Problems*, 36(3):034005, 2020.

Quentin Bertrand, Quentin Klopfenstein, Mathieu Blondel, Samuel Vaiter, Alexandre Gramfort, and Joseph Salmon. Implicit differentiation of lasso-type models for hyperparameter optimization. In *International Conference on Machine Learning*, pages 810–821. PMLR, 2020.

Rajendra Bhatia. Positive definite matrices. In *Positive Definite Matrices*. Princeton university press, 2009.

Benjamin Blankertz, Ryota Tomioka, Steven Lemm, Motoaki Kawanabe, and Klaus-Robert Muller. Optimizing spatial filters for robust eeg single-trial analysis. *IEEE Signal processing magazine*, 25(1):41–56, 2007.

Vladimir Igorevich Bogachev and Maria Aparecida Soares Ruas. *Measure theory*, volume 1. Springer, 2007.

Jérôme Bolte, Shoham Sabach, and Marc Teboulle. Proximal alternating linearized minimization for nonconvex and nonsmooth problems. *Mathematical Programming*, 146, 2013. doi: 10.1007/s10107-013-0701-9.

Clément Bonet, Laetitial Chapel, Lucas Drumetz, and Nicolas Courty. Hyperbolic sliced-wasserstein via geodesic and horospherical projections. *arXiv preprint arXiv:2211.10066*, 2022.

Clément Bonet, Benoît Malézieux, Alain Rakotomamonjy, Lucas Drumetz, Thomas Moreau, Matthieu Kowalski, and Nicolas Courty. Sliced-wasserstein on symmetric positive definite matrices for m/eeg signals. *International Conference on Machine Learning*, 2023a.

Clément Bonet, Paul Berg, Nicolas Courty, François Septier, Lucas Drumetz, and Minh-Tan Pham. Spherical sliced-wasserstein. In *International Conference on Learning Representations*, 2023b.

Nicolas Bonneel, Julien Rabin, Gabriel Peyré, and Hanspeter Pfister. Sliced and radon wasserstein barycenters of measures. *Journal of Mathematical Imaging and Vision*, 51 (1):22–45, 2015.

Nicolas Bonnotte. *Unidimensional and evolution methods for optimal transportation*. PhD thesis, Paris 11, 2013.

Ashish Bora, Eric Price, and Alexandros G Dimakis. Ambientgan: Generative models from lossy measurements. In *International conference on learning representations*, 2018.

Nicolas Boumal. An introduction to optimization on smooth manifolds. *Available online, May*, 3, 2020.

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. 2004. ISBN 0521833787. doi: 10.1017/CBO9780511804441.

Martin R Bridson and André Haefliger. *Metric spaces of non-positive curvature*, volume 319. Springer Science & Business Media, 2013.

Alon Brifman, Yaniv Romano, and Michael Elad. Turning a denoiser into a super-resolver using plug and play priors. In *2016 IEEE International Conference on Image Processing*, pages 1404–1408. IEEE, 2016.

Alice Le Brigant and Stéphane Puechmorel. Optimal riemannian quantization with an application to air traffic analysis. *arXiv preprint arXiv:1806.07605*, 2018.

Hilton Bristow, Anders Eriksson, and Simon Lucey. Fast convolutional sparse coding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 391–398, 2013.

Daniel Brooks, Olivier Schwander, Frederic Barbaresco, Jean-Yves Schneider, and Matthieu Cord. Riemannian batch normalization for spd neural networks. *Advances in Neural Information Processing Systems*, 32, 2019.

Clemens Brunner, Robert Leeb, Gernot Müller-Putz, Alois Schlögl, and Gert Pfurtscheller. Bci competition 2008–graz data set a. *Institute for Knowledge Discovery (Laboratory of Brain-Computer Interfaces), Graz University of Technology*, 16:1–6, 2008.

Wilhelm Burger and Mark J Burge. *Digital image processing: an algorithmic introduction using Java*. Springer, 2016.

Mathieu Carriere, Marco Cuturi, and Steve Oudot. Sliced wasserstein kernel for persistence diagrams. In *International conference on machine learning*, pages 664–673. PMLR, 2017.

Antonin Chambolle and Charles Dossal. On the convergence of the iterates of "fista". 2014.

Antonin Chambolle and Thomas Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40, 2011. doi: 10.1007/s10851-010-0251-1.

Antonin Chambolle and Thomas Pock. Learning consistent discretizations of the total variation. https://hal.archives-ouvertes.fr/hal-02982082/, 2020.

Antonin Chambolle, Vicent Caselles, Daniel Cremers, Matteo Novaga, and Thomas Pock. An introduction to total variation for image analysis. *Theoretical foundations and numerical methods for sparse recovery*, 9(263-340):227, 2010.

Stanley H Chan, Xiran Wang, and Omar A Elgendy. Plug-and-play admm for image restoration: Fixed-point convergence and applications. *IEEE Transactions on Computational Imaging*, 3(1):84–98, 2016.

Thomas Chang, Bahareh Tolooshams, and Demba Ba. Randnet: Deep learning with compressed measurements of images. In *2019 IEEE 29th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6. IEEE, 2019.

Niladri Chatterji and Peter L Bartlett. Alternating minimization for dictionary learning with random initialization. *Advances in Neural Information Processing Systems*, 30, 2017.

Dongdong Chen, Julián Tachella, and Mike E Davies. Equivariant imaging: Learning beyond the range space. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4379–4388, 2021.

Tianlong Chen, Xiaohan Chen, Wuyang Chen, Howard Heaton, Jialin Liu, Zhangyang Wang, and Wotao Yin. Learning to optimize: A primer and a benchmark. *Journal of Machine Learning Research*, 23(189):1–59, 2022.

Xiaohan Chen, Jialin Liu, Zhangyang Wang, and Wotao Yin. Theoretical linear convergence of unfolded ista and its practical weights and thresholds. *Advances in Neural Information Processing Systems*, 2018.

Yilun Chen, Ami Wiesel, Yonina C Eldar, and Alfred O Hero. Shrinkage algorithms for mmse covariance estimation. *IEEE Transactions on Signal Processing*, 58(10):5016–5029, 2010.

Yubei Chen, Zeyu Yun, Yi Ma, Bruno Olshausen, and Yann LeCun. Minimalistic unsupervised representation learning with the sparse manifold transform. In *The Eleventh International Conference on Learning Representations*, 2023.

Emmanuel Chevallier, Emmanuel Kalunga, and Jesus Angulo. Kernel density estimation on spaces of gaussian distributions and symmetric positive definite matrices. *SIAM Journal on Imaging Sciences*, 10(1):191–215, 2017.

James H Cole and Katja Franke. Predicting age using neuroimaging: innovative brain ageing biomarkers. *Trends in neurosciences*, 40(12):681–690, 2017.

James H Cole, Stuart J Ritchie, Mark E Bastin, Valdés Hernández, S Muñoz Maniega, Natalie Royle, Janie Corley, Alison Pattie, Sarah E Harris, Qian Zhang, et al. Brain age predicts mortality. *Molecular psychiatry*, 23(5):1385–1392, 2018.

Scott R Cole and Bradley Voytek. Brain oscillations and the importance of waveform shape. *Trends in cognitive sciences*, 21(2):137–149, 2017.

Patrick L Combettes and Jean-Christophe Pesquet. Proximal splitting methods in signal processing. *Fixed-point algorithms for inverse problems in science and engineering*, pages 185–212, 2011.

Patrick L Combettes and Valérie R Wajs. Signal recovery by proximal forward-backward splitting. *Multiscale modeling & simulation*, 4(4):1168–1200, 2005.

Laurent Condat. A primal–dual splitting method for convex optimization involving lipschitzian, proximable and linear composite terms. *Journal of Optimization Theory and Applications*, 158(2):460–479, 2013.

Laurent Condat, Daichi Kitahara, Andres Contreras, and Akira Hirabayashi. Proximal splitting algorithms: Overrelax them all! 2019.

Facundo Costa, Hadj Batatia, Thomas Oberlin, Carlos d'Giano, and Jean-Yves Tourneret. Bayesian eeg source localization using a structured sparsity prior. *NeuroImage*, 144: 142–152, 2017.

Nicolas Courty, Rémi Flamary, Devis Tuia, and Alain Rakotomamonjy. Optimal transport for domain adaptation. *IEEE transactions on pattern analysis and machine intelligence*, 39(9):1853–1865, 2016.

Li Cui, Xin Qi, Chengfeng Wen, Na Lei, Xinyuan Li, Min Zhang, and Xianfeng Gu. Spherical optimal transportation. *Computer-Aided Design*, 115:181–193, 2019.

Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26, 2013.

Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Bm3d image denoising with shape-adaptive principal component analysis. In *SPARS'09-Signal Processing with Adaptive Sparse Structured Representations*, 2009.

Sven Dähne, Frank C Meinecke, Stefan Haufe, Johannes Höhne, Michael Tangermann, Klaus-Robert Müller, and Vadim V Nikulin. Spoc: a novel framework for relating the amplitude of neuronal oscillations to behaviorally relevant parameters. *NeuroImage*, 86: 111–122, 2014.

Janis J Daly and Jonathan R Wolpaw. Brain–computer interfaces in neurological rehabilitation. *The Lancet Neurology*, 7(11):1032–1043, 2008.

John M. Danskin. *Theory of Max-Min and Its Application to Weapons Allocation Problems.* Springer Berlin Heidelberg, Berlin/Heidelberg, 1967.

Ingrid Daubechies, Michel Defrise, and Christine Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constrains. *Communications on Pure and Applied Mathematics*, 57, 2004. doi: 10.1002/cpa.20042.

Charles-Alban Deledalle, Samuel Vaiter, Jalal Fadili, and Gabriel Peyré. Stein unbiased gradient estimator of the risk (sugar) for multiple parameter selection. *SIAM Journal on Imaging Sciences*, 7(4):2448–2487, 2014.

Tom Dupré la Tour, Thomas Moreau, Mainak Jas, and Alexandre Gramfort. Multivariate convolutional sparse coding for electromagnetic brain signals. *Advances in Neural Information Processing Systems*, 31:3292–3302, 2018.

Michael Elad. *Sparse and Redundant Representations - From Theory to Applications in Signal and Image Processing.* 2010a. ISBN 978-1-4419-7010-7. doi: 10.1007/978-1-4419-7011-4.

Michael Elad. *Sparse and redundant representations: from theory to applications in signal and image processing.* Springer Science & Business Media, 2010b.

Michael Elad, Peyman Milanfar, and Ron Rubinstein. Analysis versus synthesis in signal priors. *Inverse Problems*, 23:947, 2007. doi: 10.1088/0266-5611/23/3/007.

Denis A Engemann, Apolline Mellot, Richard Höchenberger, Hubert Banville, David Sabbagh, Lukas Gemein, Tonio Ball, and Alexandre Gramfort. A reusable benchmark of brain-age prediction from m/eeg resting-state signals. *Neuroimage*, 262:119521, 2022.

Kilian Fatras, Younes Zine, Rémi Flamary, Rèmi Gribonval, and Nicolas Courty. Learning with minibatch wasserstein : asymptotic and gradient properties. In Silvia Chiappa and Roberto Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 2131–2141. PMLR, 26–28 Aug 2020.

Rémi Flamary, Nicolas Courty, Alexandre Gramfort, Mokhtar Z Alaya, Aurélie Boisbunon, Stanislas Chambon, Laetitia Chapel, Adrien Corenflos, Kilian Fatras, Nemo Fournier, et al. Pot: Python optimal transport. *J. Mach. Learn. Res.*, 22(78):1–8, 2021.

Simon Foucart and Holger Rauhut. *A mathematical introduction to compressive sensing.* 2013a. doi: 10.1007/978-0-8176-4948-7.

Simon Foucart and Holger Rauhut. An invitation to compressive sensing. In *A mathematical introduction to compressive sensing*, pages 1–39. Springer, 2013b.

Nicolas Fournier and Arnaud Guillin. On the rate of convergence in wasserstein distance of the empirical measure. *Probability Theory and Related Fields*, 162(3):707–738, 2015.

Pramod Gaur, Ram Bilas Pachori, Hui Wang, and Girijesh Prasad. A multi-class eeg-based bci classification using multivariate empirical mode decomposition based filtering and riemannian geometry. *Expert Systems with Applications*, 95:201–211, 2018.

Gene H Golub and Charles F Van Loan. *Matrix computations*. JHU press, 2013.

Mehmet Gönen and Ethem Alpaydın. Multiple kernel learning algorithms. *The Journal of Machine Learning Research*, 12:2211–2268, 2011.

Alexandre Gramfort. *Mapping, timing and tracking cortical activations with MEG and EEG: Methods and application to human vision*. PhD thesis, Ecole nationale supérieure des telecommunications-ENST, 2009.

Alexandre Gramfort, Matthieu Kowalski, and Matti Hämäläinen. Mixed-norm estimates for the m/eeg inverse problem using accelerated gradient methods. *Physics in medicine and biology*, 57:1937–61, 2012.

Karol Gregor and Yann LeCun. Learning fast approximations of sparse coding. *International conference on machine learning*, pages 399–406, 2010.

Rémi Gribonval and Sylvain Lesage. A survey of sparse component analysis for blind source separation: principles, perspectives, and new challenges. In *ESANN'06 proceedings-14th European Symposium on Artificial Neural Networks*, pages 323–330. d-side publi., 2006.

Rémi Gribonval and Karin Schnass. Dictionary identification—sparse matrix-factorization via \ell_1-minimization. *IEEE Transactions on Information Theory*, 56(7):3523–3539, 2010.

Rémi Gribonval, Rodolphe Jenatton, and Francis Bach. Sparse and spurious: dictionary learning with noise and outliers. *IEEE Transactions on Information Theory*, 61(11): 6298–6319, 2015.

Roger Grosse, Rajat Raina, Helen Kwong, and Andrew Y. Ng. Shift-Invariant Sparse Coding for Audio Classification. *Cortex*, 8:9, 2007.

Benjamin D Haeffele and René Vidal. Global optimality in tensor factorization, deep learning, and beyond. *arXiv preprint arXiv:1506.07540*, 2015.

Matti Hämäläinen, Riitta Hari, Risto J Ilmoniemi, Jukka Knuutila, and Olli V Lounasmaa. Magnetoencephalography—theory, instrumentation, and applications to noninvasive studies of the working human brain. *Reviews of modern Physics*, 65(2):413, 1993.

Kerstin Hammernik, Tobias Würfl, Thomas Pock, and Andreas Maier. A deep learning architecture for limited-angle computed tomography reconstruction. In *Bildverarbeitung für die Medizin 2017: Algorithmen-Systeme-Anwendungen. Proceedings des Workshops vom 12. bis 14. März 2017 in Heidelberg*, pages 92–97. Springer, 2017.

Mehrtash Harandi, Mathieu Salzmann, and Richard Hartley. Dimensionality reduction on spd manifolds: The emergence of geometry-aware methods. *IEEE transactions on pattern analysis and machine intelligence*, 40(1):48–62, 2017.

Hastie, Tibshirani, and Wainwright. *Statistical learning with sparsity: The lasso and generalizations*. 2015. doi: 10.1201/b18401.

Matthias Hein and Olivier Bousquet. Hilbertian metrics and positive definite kernels on probability measures. In *International Workshop on Artificial Intelligence and Statistics*, pages 136–143. PMLR, 2005.

Zhiwu Huang and Luc Van Gool. A riemannian network for spd matrix learning. In *Thirty-first AAAI conference on artificial intelligence*, 2017.

Zhiwu Huang, Ruiping Wang, Shiguang Shan, Xianqiu Li, and Xilin Chen. Log-euclidean metric learning on symmetric positive definite manifold with application to image set classification. In *International conference on machine learning*, pages 720–729. PMLR, 2015.

Ioana Ilea, Lionel Bombrun, Salem Said, and Yannick Berthoumieu. Covariance matrices encoding based on the log-euclidean and affine invariant riemannian metrics. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 393–402, 2018.

Kazufumi Ito and Bangti Jin. *Inverse problems: Tikhonov theory and algorithms*, volume 22. World Scientific, 2014.

Mainak Jas, Tom Dupré La Tour, Umut Şimşekli, and Alexandre Gramfort. Learning the morphology of brain signals using alpha-stable convolutional sparse coding. *Advances in Neural Information Processing Systems*, pages 1099–1108, 2017.

Mingyuan Jiu and Nelly Pustelnik. A deep primal-dual proximal network for image restoration. *IEEE Journal of Selected Topics in Signal Processing*, 15(2):190–203, 2021.

Ce Ju and Cuntai Guan. Deep optimal transport on spd manifolds for domain adaptation. *arXiv preprint arXiv:2201.05745*, 2022.

Srđan Kitić, Laurent Albera, Nancy Bertin, and Rémi Gribonval. Physics-driven inverse problems made tractable with cosparse regularization. *IEEE Transactions on Signal Processing*, 64(2):335–348, 2015.

Erich Kobler, Alexander Effland, Karl Kunisch, and Thomas Pock. Total deep variation for linear inverse problems. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7549–7558, 2020.

Max Kochurov, Rasul Karimov, and Serge Kozlukov. Geoopt: Riemannian optimization in pytorch. *arXiv preprint arXiv:2005.02819*, 2020.

Andreas Kofler, Markus Haltmeier, Tobias Schaeffter, Marc Kachelrieß, Marc Dewey, Christian Wald, and Christoph Kolbitsch. Neural networks-based regularization for large-scale medical image reconstruction. *Physics in Medicine & Biology*, 65(13):135003, 2020.

Soheil Kolouri, Yang Zou, and Gustavo K Rohde. Sliced wasserstein kernels for probability distributions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5258–5267, 2016.

Marine Le Morvan, Julie Josse, Erwan Scornet, and Gaël Varoquaux. What'sa good imputation to predict with missing values? *Advances in Neural Information Processing Systems*, 34:11530–11540, 2021.

Bruno Lecouat, Jean Ponce, and Julien Mairal. A flexible framework for designing trainable priors with adaptive smoothing and game encoding. In *Advances in neural information processing systems*, 2020.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Jaakko Lehtinen, Jacob Munkberg, Jon Hasselgren, Samuli Laine, Tero Karras, Miika Aittala, and Timo Aila. Noise2noise: Learning image restoration without clean data. In *International Conference on Machine Learning*, pages 2965–2974. PMLR, 2018.

Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In *Advances in neural information processing systems*, pages 6389–6399, 2018.

Yuelong Li, Or Bar-Shira, Vishal Monga, and Yonina C Eldar. Deep algorithm unrolling for biomedical imaging. *arXiv preprint arXiv:2108.06637*, 2021.

Yujie Li, Shuxue Ding, Zhenni Li, Xiang Li, and Benying Tan. Dictionary learning in the analysis sparse representation with optimization on stiefel manifold. In *IEEE Global Conference on Signal and Information Processing*, pages 1270–1274. IEEE, 2017.

Jialin Liu and Xiaohan Chen. Alista: Analytic weights are as good as learned weights in lista. In *International Conference on Learning Representations (ICLR)*, 2019.

Jiaming Liu, Yu Sun, Cihat Eldeniz, Weijie Gan, Hongyu An, and Ulugbek S Kamilov. Rare: Image reconstruction using deep priors learned without groundtruth. *IEEE Journal of Selected Topics in Signal Processing*, 14(6):1088–1099, 2020.

Ignace Loris and Caroline Verhoeven. On a generalization of the iterative soft-thresholding algorithm for the case of non-separable penalty. *Inverse Problems*, 27(12):125007, 2011.

Julien Mairal, Francis Bach, J. Ponce, and Guillermo Sapiro. Online learning for matrix factorization and sparse coding. *Journal of Machine Learning Research*, 11, 2009.

Benoît Malézieux, Thomas Moreau, and Matthieu Kowalski. Understanding approximate and unrolled dictionary learning for pattern recovery. *International Conference on Learning Representations*, 2022a.

Benoît Malézieux, Thomas Moreau, and Matthieu Kowalski. Apprentissage de dictionnaire par différentiation automatique pour la résolution de problèmes inverses. *GRETSI*, 2022b.

Stéphane Mallat. *A Wavelet Tour of Signal Processing*. Academic press, 2008.

Stéphane Mallat and Zhifeng Zhang. Matching pursuit with time-frequency dictionaries. *IEEE Transactions on Signal Processing*, 41:3397 – 3415, 1994.

Tudor Manole, Sivaraman Balakrishnan, and Larry Wasserman. Minimax confidence intervals for the sliced wasserstein distance. *Electronic Journal of Statistics*, 16(1):2252–2345, 2022.

Robert J McCann. Polar factorization of maps on riemannian manifolds. *Geometric & Functional Analysis GAFA*, 11(3):589–608, 2001.

Arthur Mensch, Julien Mairal, Bertrand Thirion, and Gaël Varoquaux. Dictionary learning for massive matrix factorization. In *International Conference on Machine Learning*, pages 1737–1746. PMLR, 2016.

Arthur Mensch, Julien Mairal, Bertrand Thirion, and Gaël Varoquaux. Stochastic subsampling for factorizing huge matrices. *IEEE Transactions on Signal Processing*, 66(1): 113–128, 2017.

Dimitri Meunier, Massimiliano Pontil, and Carlo Ciliberto. Distribution regression with sliced Wasserstein kernels. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 15501–15523. PMLR, 17–23 Jul 2022.

Francesco Mezzadri. How to generate random matrices from the classical compact groups. *arXiv preprint math-ph/0609050*, 2006.

Florent Michel, Benoît Malézieux, Thomas Moreau, and Matthieu Kowalski. L'entropie comme mesure de difficulté des problèmes inverses. *GRETSI*, 2023.

Thomas Moreau and Joan Bruna. Understanding neural sparse coding with matrix factorization. In *International Conference on Learning Representation*, 2017.

Thomas Moreau and Alexandre Gramfort. Dicodile: Distributed convolutional dictionary learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

Thomas Moreau, Laurent Oudre, and Nicolas Vayatis. Dicod: Distributed convolutional coordinate descent for convolutional sparse coding. In *International Conference on Machine Learning*, pages 3626–3634. PMLR, 2018.

Thomas Moreau, Mathurin Massias, Alexandre Gramfort, Pierre Ablin, Pierre-Antoine Bannier, Benjamin Charlier, Mathieu Dagréou, Tom Dupre la Tour, Ghislain Durif, Cassio F Dantas, et al. Benchopt: Reproducible, efficient and collaborative optimization benchmarks. *Advances in Neural Information Processing Systems*, 35:25404–25421, 2022.

Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

Kimia Nadjahi, Alain Durmus, Umut Simsekli, and Roland Badeau. Asymptotic guarantees for learning generative models with the sliced-wasserstein distance. *Advances in Neural Information Processing Systems*, 32, 2019.

Kimia Nadjahi, Alain Durmus, Lénaïc Chizat, Soheil Kolouri, Shahin Shahrampour, and Umut Simsekli. Statistical and topological properties of sliced probability divergences. *Advances in Neural Information Processing Systems*, 33:20802–20812, 2020.

Valeriya Naumova and Karin Schnass. Dictionary learning from incomplete data for efficient image restoration. In *2017 25th European Signal Processing Conference (EUSIPCO)*, pages 1425–1429. IEEE, 2017.

Luis Fernando Nicolas-Alonso and Jaime Gomez-Gil. Brain computer interfaces, a review. *sensors*, 12(2):1211–1279, 2012.

Jonathan Niles-Weed and Philippe Rigollet. Estimation of wasserstein distances in the spiked transport model. *Bernoulli*, 28(4):2663–2688, 2022.

Thomas Oberlin and Mathieu Verm. Regularization via deep generative models: an analysis point of view. In *2021 IEEE International Conference on Image Processing (ICIP)*, pages 404–408. IEEE, 2021.

Bruno A. Olshausen and David J Field. Sparse coding with an incomplete basis set: A strategy employed by \protect{V1}. *Vision Research*, 37(23):3311–3325, 1997.

Alexey Ozerov and Cédric Févotte. Multichannel nonnegative matrix factorization in convolutive mixtures for audio source separation. *IEEE transactions on audio, speech, and language processing*, 18(3):550–563, 2009.

Neal Parikh, Stephen Boyd, et al. Proximal algorithms. *Foundations and trends® in Optimization*, 1(3):127–239, 2014.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, pages 8026–8037, 2019.

Yagyensh Chandra Pati, Ramin Rezaiifar, and Perinkulam Sambamurthy Krishnaprasad. Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In *Proceedings of 27th Asilomar conference on signals, systems and computers*, pages 40–44. IEEE, 1993.

François-Pierre Paty and Marco Cuturi. Subspace robust wasserstein distances. In *International conference on machine learning*, pages 5072–5081. PMLR, 2019.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, Edouard Duchesnay, and Gilles Louppe. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12, 2012.

Ofir Pele and Michael Werman. Fast and robust earth mover's distances. In *2009 IEEE 12th international conference on computer vision*, pages 460–467. IEEE, 2009.

Han Peng, Weikang Gong, Christian F Beckmann, Andrea Vedaldi, and Stephen M Smith. Accurate brain age prediction with lightweight deep neural networks. *Medical image analysis*, 68:101871, 2021.

Xavier Pennec. Manifold-valued image processing with spd matrices. In *Riemannian geometric statistics in medical image analysis*, pages 75–134. Elsevier, 2020.

Xavier Pennec, Pierre Fillard, and Nicholas Ayache. A riemannian framework for tensor computing. *International Journal of computer vision*, 66(1):41–66, 2006.

Gabriel Peyré and Jalal M Fadili. Learning analysis sparsity priors. In *Sampta'11*, pages 4–pp, 2011.

Gabriel Peyré, Marco Cuturi, et al. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019.

Farhad Pourkamali-Anaraki, Stephen Becker, and Shannon M Hughes. Efficient dictionary learning via very sparse random projections. In *2015 International Conference on Sampling Theory and Applications (SampTA)*, pages 478–482. IEEE, 2015.

Lindsey Power, Cédric Allain, Thomas Moreau, Alexandre Gramfort, and Timothy Bardouille. Using convolutional dictionary learning to detect task-related neuromagnetic transients and ageing trends in a large open-access dataset. *NeuroImage*, 267:119809, 2023.

Julien Rabin, Gabriel Peyré, Julie Delon, and Marc Bernot. Wasserstein barycenter and its application to texture mixing. In *International Conference on Scale Space and Variational Methods in Computer Vision*, pages 435–446. Springer, 2011.

Zaccharie Ramzi, Philippe Ciuciu, and Jean-Luc Starck. Xpdnet for mri reconstruction: an application to the fastmri 2020 brain challenge. 2020.

Audrey Repetti, Matthieu Terris, Yves Wiaux, and Jean-Christophe Pesquet. Dual forward-backward unfolded network for flexible plug-and-play. In *2022 30th European Signal Processing Conference (EUSIPCO)*, pages 957–961. IEEE, 2022.

JH Rick Chang, Chun-Liang Li, Barnabas Poczos, BVK Vijaya Kumar, and Aswin C Sankaranarayanan. One network to solve them all–solving linear inverse problems using deep projection models. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5888–5897, 2017.

Pedro Luiz Coelho Rodrigues, Christian Jutten, and Marco Congedo. Riemannian procrustes analysis: transfer learning for brain–computer interfaces. *IEEE Transactions on Biomedical Engineering*, 66(8):2390–2401, 2018.

Yaniv Romano, Michael Elad, and Peyman Milanfar. The little engine that could: Regularization by denoising (red). *SIAM Journal on Imaging Sciences*, 10(4):1804–1844, 2017.

Raif M Rustamov and Subhabrata Majumdar. Intrinsic sliced wasserstein distances for comparing collections of probability distributions on manifolds and graphs. *arXiv preprint arXiv:2010.15285*, 2020.

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.

David Sabbagh, Pierre Ablin, Gaël Varoquaux, Alexandre Gramfort, and Denis A Engemann. Manifold-regression to predict from meg/eeg brain signals without source modeling. *Advances in Neural Information Processing Systems*, 32, 2019.

David Sabbagh, Pierre Ablin, Gaël Varoquaux, Alexandre Gramfort, and Denis A Engemann. Predictive regression modeling with meg/eeg: from source power to signals and cognitive states. *NeuroImage*, 222:116893, 2020.

Meyer Scetbon, Michael Elad, and Peyman Milanfar. Deep k-svd denoising. *IEEE Transactions on Image Processing*, 30:5944–5955, 2021.

Evan Schwab, Benjamin D Haeffele, René Vidal, and Nicolas Charon. Global optimality in separable dictionary learning with applications to the analysis of diffusion mri. *SIAM Journal on Imaging Sciences*, 12(4):1967–2008, 2019.

Amirreza Shaban, Ching-An Cheng, Nathan Hatch, and Byron Boots. Truncated backpropagation for bilevel optimization. In *International Conference on Artificial Intelligence and Statistics*, pages 1723–1732. PMLR, 2019.

David Spiegelhalter. How old are you, really? communicating chronic risk through 'effective age'of your body and organs. *BMC medical informatics and decision making*, 16(1):1–6, 2016.

Jean-Luc Starck. Sparsity and inverse problems in astrophysics. *Journal of Physics: Conference Series*, 699, 2016.

Christoph Studer and Richard G Baraniuk. Dictionary learning from sparsely corrupted or compressed signals. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3341–3344. IEEE, 2012.

Ju Sun, Qing Qu, and John Wright. Complete dictionary recovery over the sphere i: Overview and the geometric picture. *IEEE Transactions on Information Theory*, 63(2): 853–884, 2016.

Zoltán Szabó, Barnabás Póczos, and András Lőrincz. Online group-structured dictionary learning. In *CVPR 2011*, pages 2865–2872. IEEE, 2011.

Julián Tachella, Dongdong Chen, and Mike Davies. Sampling theorems for learning from incomplete measurements. *arXiv preprint arXiv:2201.12151*, 2022.

Wen Tang, Emilie Chouzenoux, Jean-Christophe Pesquet, and Hamid Krim. Deep transform and metric learning network: Wedding deep dictionary learning and neural network. *Neurocomputing*, 509:244–256, 2022.

Jason R Taylor, Nitin Williams, Rhodri Cusack, Tibor Auer, Meredith A Shafto, Marie Dixon, Lorraine K Tyler, Richard N Henson, et al. The cambridge centre for ageing and neuroscience (cam-can) data repository: Structural and functional mri, meg, and cognitive data from a cross-sectional adult lifespan sample. *neuroimage*, 144:262–269, 2017.

Yann Thanwerdas and Xavier Pennec. O (n)-invariant riemannian metrics on spd matrices. *Linear Algebra and its Applications*, 661:163–201, 2023.

Tibshirani and Ryan. The lasso problem and uniqueness. *Electronic Journal of Statistics*, 7, 2012. doi: 10.1214/13-EJS815.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society Series B*, 58:267–288, 1996.

Ryan J. Tibshirani. The lasso problem and uniqueness. *Electronic Journal of Statistics*, 7 (1):1456–1490, 2013.

Bahareh Tolooshams and Demba Ba. Pudle: Implicit acceleration of dictionary learning by backpropagation. *arXiv preprint*, 2021.

Bahareh Tolooshams, Sourav Dey, and Demba Ba. Deep residual autoencoders for expectation maximization-inspired dictionary learning. *IEEE Transactions on Neural Networks and Learning Systems*, PP:1–15, 2020. doi: 10.1109/TNNLS.2020.3005348.

Paul Tseng. Approximation accuracy, gradient methods, and error bound for structured convex optimization. *Mathematical Programming*, 125:263–295, 2010. doi: 10.1007/s10107-010-0394-2.

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.

Sharan Vaswani, Aaron Mishkin, Issam Laradji, Mark Schmidt, Gauthier Gidel, and Simon Lacoste-Julien. Painless stochastic gradient: Interpolation, line-search, and convergence rates. *Advances in neural information processing systems*, 32:3732–3745, 2019.

Pierre-Hugo Vial, Paul Magron, Thomas Oberlin, and Cédric Févotte. Learning the proximity operator in unfolded admm for phase retrieval. *IEEE Signal Processing Letters*, 29:1619–1623, 2022.

Cédric Villani. *Optimal transport: old and new*, volume 338. Springer, 2009.

Bang Cong Vu. A splitting algorithm for dual monotone inclusions involving cocoercive operators. *Advances in Computational Mathematics*, 38(3):667–681, 2013.

Zaiwen Wen and Wotao Yin. A feasible method for optimization with orthogonality constraints. *Mathematical Programming*, 142(1-2):397–434, 2013.

Brendt Wohlberg. Efficient algorithms for convolutional sparse representations. *IEEE Transactions on Image Processing*, 25(1):301–315, 2015.

Jonathan R Wolpaw. Brain–computer interfaces. In *Handbook of Clinical Neurology*, volume 110, pages 67–74. Elsevier, 2013.

Tong Tong Wu, Kenneth Lange, et al. Coordinate descent algorithms for lasso penalized regression. *Annals of Applied Statistics*, 2(1):224–244, 2008.

Yande Xiang, Zhitao Lin, and Jianyi Meng. Automatic qrs complex detection using two-level convolutional neural network. *Biomedical engineering online*, 17(1):1–17, 2018.

Alba Xifra-Porxas, Arna Ghosh, Georgios D Mitsis, and Marie-Hélène Boudrias. Estimating brain age from structural mri and meg data: Insights from dimensionality reduction techniques. *NeuroImage*, 231:117822, 2021.

Hao Xu. Unsupervised manifold learning with polynomial mapping on symmetric positive definite matrices. *Information Sciences*, 609:215–227, 2022.

Mehrdad Yaghoobi, Sangnam Nam, Rémi Gribonval, and Mike E Davies. Constrained overcomplete analysis operator learning for cosparse signal modelling. *IEEE Transactions on Signal Processing*, 61(9):2341–2355, 2013.

Or Yair, Felix Dietrich, Ronen Talmon, and Ioannis G Kevrekidis. Domain adaptation with optimal transport on the manifold of spd matrices. *arXiv preprint arXiv:1906.00616*, 2019.

John Zarka, Louis Thiry, Tomas Angles, and Stephane Mallat. Deep network classification by scattering and homotopy dictionary learning. In *International Conference on Learning Representations*, 2019.

Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising. *IEEE Transactions on Image Processing*, 26(7):3142–3155, 2017.

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*, 44(10):6360–6376, 2021.