Data Aware Sparse Non–Negative Signal Processing

Konstantinos Voulgaris

A thesis submited for the degree of Doctor of Philosophy

The University of Edinburgh

2021

Abstract

Greedy techniques are a well established framework aiming to reconstruct signals which are sparse in some domain of representations. They are renowned for their relatively low computational cost, that makes them appealing from the perspective of real time applications. Within the current work we focus on the explicit case of sparse non–negative signals that finds applications in several aspects of daily life e.g., food analysis, hazardous materials detection etc. The conventional approach to deploy this type of algorithms does not employ benefits from properties that characterise natural data, such as lower dimensional representations, underlying structures. Motivated by these properties of data we are aiming to incorporate methodologies within the domain of greedy techniques that will boost their performance in terms of: 1) computational efficiency and 2) signal recovery improvement (for the remainder of the thesis we will use the term acceleration when referring to the first goal and robustness when we are referring to the second goal). These benefits can be exploited via data aware methodologies that arise, from the Machine Learning and Deep Learning community.

Within the current work we are aiming to establish a link among conventional sparse non–negative signal decomposition frameworks that rely on greedy techniques and data aware methodologies. We have explained the connection among data aware methodologies and the challenges associated with the sparse non–negative signal decompositions: 1) acceleration and 2) robustness. We have also introduced the standard data aware methodologies, which are relevant to our problem, and the theoretical properties they have. The practical implementations of the proposed frameworks are provided here. The main findings of the current work can be summarised as follows:

- We introduce novel algorithms, theory for the Nearest Neighbor problem.
- We accelerate a greedy algorithm for sparse non–negative signal decomposition by incorporating our algorithms within its structure.
- We introduce a novel reformulation of greedy techniques from the perspective of a Deep Neural Network that boosts the robustness of greedy techniques.
- We introduce the theoretical framework that fingerprints the conditions that lay down the soil for the exact recovery of the signal.

Lay Summary

Sparse signal processing is a well established framework within the signal processing community. Its application has been proven beneficial in several aspects of daily life e.g. Magnetic Resonance Imaging (MRI) where the time for image acquisition was significantly reduced, which benefits the patients as well as health care economics. Within the current work we have a particular interest for a special class of sparse signals i.e. non–negative signals, that find applications in real life problems e.g., hazardous materials detection, food analysis etc.

Within the literature there exist several methodologies aiming to solve the sparse non–negative signal processing problem. The current work emphasizes on techniques that solve the problem within a time frame suitable for real time applications. A popular approach that meets this sort of criterion are greedy techniques. However, the operational time required by these techniques to carry out the assigned task scales with the size of the problem i.e. size of data stacked in system memory. Hence, the generic framework of greedy techniques is not well suited while the size of data increases. Moreover, these techniques are approximate i.e. given a sample that contains a hazardous material the corresponding algorithm may not detect it. This sort of characteristic may be fatal considering the type of applications we focus on. Within the current work we are aiming to develop methodologies that: 1) reduce the operational time of greedy techniques 2) improve the performance of greedy techniques in terms of recovering the actual elements that constitute the input signal.

In recent years data aware methodologies more formally known as machine learning and deep learning, have introduced novel solutions in many real life problems. These sort of techniques learn to perform a particular task by using known data as an example. Within the current work we are aiming to develop data aware methodologies that will reduce the computational cost of greedy techniques and improve their performance in terms of detecting the correct elements that constitute the input signal.

Declaration of originality

I hereby declare that the research recorded in this thesis and the thesis itself was composed and originated entirely by myself in the School of Engineering at The University of Edinburgh.

Konstantinos Voulgaris

Acknowledgements

There are many people who have contributed to this work in their own way, technical or non–technical, throughout the period of this project. I will start by thanking my supervisor Associate Professor Mehrdad Yaghoobi Vaighan for giving me the opportunity of doing this PhD in the first place as well as for the nice collaboration we had during this period. Moreover I would like to thank my second supervisor Professor Mike E. Davies for his insightful comments on this work.

The current work has been fully supported by the UK Engineering and Physical Sciences Research Council (EPSRC) under grant numbers EP/S000631/1 and EP/K014277/1. I would like to thank the EPSRC for their financial support and travel grants, which allowed me to present the papers at several conferences.

I would also like to thank my colleagues for creating a nice working environment at AGB 2.01 office where I spent most of my time during this PhD work. Special thanks to my great colleague and good mate Mikey Sheehan for the nice chats we had and for being there to share a cup of coffee when needed. Many thanks to all the people we spent our Wednesday evenings together playing football at Peffermill fields. Spending my time with them and their remarkable team spirit was one of the things that helped me to blow off the steam and disconnect from the day by day work.

On a relatively more personal note I would like to thank all of my friends. I would like to express my sincere thanks and my gratitude to all of them for shaping my character and my attitude and made the experiences we shared together unforgettable. Special thanks to my brother in arms Dimitrios Tycheros, and also to Maria Mizamidou and Shivani Khanna for being always there to share the good and the bad times of this period.

Finally, I would like to thank my family, my father Antonis, my mother Paraskevi and my sister Vasiliki, for their consistent and unconditional support. This work is dedicated to my niece Konstantina who was born during the period of this PhD work.

Acronyms and Abbreviations

AENN	Adaptive Embedded Nearest Neighbor
BP	Basis Pursuit
CS	Compressed Sensing
CoSaMP	Compressive Sampling Matching Pursuit
DeepMP	Deep Matching Pursuit
DeepMPSM	Deep Matching Pursuit with Support Manipulation
DL	Deep Learning
DNN	Deep Neural Network
ENN	Embedded Nearest Neighbor
FNNOMP	Fast Non Negative Orthogonal Matching Pursuit
IHT	Iterative Hard Thresholding
ISTA	Iterative Soft Thresholding Algorithm
LASSO	Least Absolute Shrinkage and Selection Operator
LISTA	Learned Iterative Soft Thresholding Algorithm
MIPS	Maximum Inner Product Search
MP	Matching Pursuit
NNS	Nearest Neighbor Search
NNMP	Non Negative Matching Pursuit
NuMax	Nuclear norm minimization with Max–norm constraints
OMP	Orthogonal Matching Pursuit

PCA	Principal Component Analysis
RIP	Restricted Isometry Property
RP	Random Projections
SVD	Singular Value Decomposition
tML	traditional Machine Learning

Nomenclature

NN_K	index in $\hat{\Phi}$ that corresponds to the Nearest Neighbor in R^K
NN_M	index in Φ that corresponds to the Nearest Neighbor in R^M
Q	embedding operator
Q_K	K first columns of Q_N
Q_N	$N \times N$ orthonormal matrix
s	support set that contains the non–zero indexes in x
W_i	set of weights at layer i
x	coefficient vector
x_s	restricted set of coefficients to the support set s
y	signal
$(\cdot)^\dagger$	pseudoinverse
$(\cdot)^T$	transpose
δ	constant used as an upper range bound
$\hat{\Phi}$	representation of Φ in R^K
$\hat{y}$	representation of y in R^K
$\mathcal{H}_\lambda$	hard thresholding operator
$\mathcal{S}_\lambda$	soft thresholding operator
μ	the coherence of the dictionary Φ
$\perp$	orthogonality between two vectors
Φ	The dictionary of atoms

ϕ_i	the ith column of Φ
Φ_s	restricted dictionary to the support set s
k	number of non–zero coefficients in x
x_i	the ith element of vector x
$\approx$	approximately equal
$\mathcal{O}(\cdot)$	computational complexity

Contents

Chapter 1

Introduction

1.1 Research Problem

Sparse signal processing is a framework that has been of particular interest within the signal processing community in the last decade. There exist many signals (e.g. images, audio etc) which are sparse with respect to a domain of representation. This type of approaches are appealing when considering real–time applications e.g. handheld spectrometers or mobile phones, since they enable the development of robust methodologies on a limited hardware topology.

The application of these methodologies has already been proven beneficial in several ways, i.e. Magnetic Resonance Imaging (MRI) where the time for image acquisition was significantly reduced, which benefits the patients as well as health care economics [1]. The preliminary focus of this project is on the development of non–negative sparse signal processing applications. The development of these applications may be beneficial in several areas such as defense, security or food industry.

Considering the algorithmic framework upon which we can develop these applications the most popular, in terms of robustness, is the one that solves the l_1–norm minimization problem also known as the Basis Pursuit (BP) [2]. The main drawback when considering this kind of approach as a potential strategy though is the fact that they are characterized by a heavy computational workload.

The greedy sparse approximation algorithms are generally characterised by a low computational cost that makes them suitable for real–time and large scale approximations. One simple greedy algorithm is Matching Pursuit (MP) [3] which builds a sparse representation of a signal on an iterative manner. In order to compensate the disadvantages of MP (e.g., a selected atom may be reselected, the representation found by the algorithm is not the best representation using the selected atoms), the Orthogonal Matching Pursuit (OMP) algorithm [4],[5] is introduced. OMP projects the input signal $\boldsymbol{y}$ onto the selected support set s at the expense of some extra computational cost. This type of projection can be found by solving the following optimization problem:

$$\tilde{\boldsymbol{x}}_s = \arg\min_{\boldsymbol{x}_\mathrm{s}} ||\boldsymbol{y} - \boldsymbol{\Phi}_s \boldsymbol{x}_s||, \tag{1.1}$$

where $\boldsymbol{\Phi}_s$ and $\boldsymbol{x}_s$ the sub–dictionary and coefficient vector accordingly with respect to the support s.

Greedy techniques may be appealing in general given that the corresponding computational cost is relatively low compared to alternative approaches such as BP, however the computational cost scales with the number of elements M and the dimensions of each element N. Hence, the relatively low computational workload of greedy techniques grows along the size of the associated dictionary. Furthermore, the decomposition process of greedy techniques heavily relies on the coherence in between the dictionary elements. The underlying coherence between the dictionary elements is of a critical importance given that it affects the performance of the associated algorithms in terms of signal reconstruction.

Within the current work we are aiming to boost the performance of greedy techniques in terms of: 1) acceleration i.e. reduce the corresponding computational cost and 2) signal reconstruction. Our motivation is driven by several applications that can benefit from this sort of approaches e.g., hazardous materials detection, food analysis etc. A natural question that rises though is the following: Given that greedy techniques are a well established framework, what is the new idea that the current work brings into the agenda?

For the last decade the research community is oriented into the development of data aware methodologies seeking for better solutions into real life problems. The main scope of the current work is to develop frameworks that incorporate the underlying properties that characterise natural data, e.g., lower dimensional representation, data structure etc and upgrade the performance of greedy techniques.

Considering the framework upon which these properties may be exploited we focus on data aware techniques that arise from the area of machine learning. In that sense, the current work is aiming to incorporate machine learning methodologies within a well established concept in the signal processing community, such as sparse signal processing, with a focus on the non–negativity case. In summary, the motivation of the current work is to act as a bridge in between data aware techniques and traditional signal processing.

1.2 Thesis Statements

Within the current work we are aiming to develop methodologies that can be introduced in the standard sparse non–negative setting with an aim to improve the performance of existing methodologies with respect to two main criteria: 1) acceleration and 2) robustness.

Our preliminary goal is to lay down the soil for algorithms that meet the tight criteria of real time applications. By considering this parameter as a guideline for our work, a natural question that rises is how fast these algorithms can solve the given problem. In other words, what is the computational workload that these methodologies carry out?

Another critical parameter though, and given the sensitivity of the targeted applications (e.g., detection of hazardous materials, food analysis etc), is the robustness of these methodologies. Within the current work are considering an algorithm to be robust when it fully reconstructs the input signal. Given that the current work mainly focus is on algorithms which are fast in terms of the computational cost, but approximate with respect to the solution of the given problem, the second direction of this project is the improvement of existing algorithms in terms of robustness i.e., improve the true positives detection ratio and their contribution in the input signal.

The main motivation that pushes our work forward is the improvement that data aware processes have delivered in several areas of research i.e. image processing [6], audio processing [7] etc. Within the current work the main focus is on techniques that exploit benefit from properties that typically characterise data such as: 1) lower dimensional representation, and 2) underlying structure.

We here focus on data that originally live in a high dimensional space R^N. Within the term lower dimensional representation, what we actually mean is that the original signal $\boldsymbol{y} \in R^N$ has a lower dimensional representation $\hat{\boldsymbol{y}} \in R^K$ which contains a relatively similar amount of information.

From the perspective of underlying structure of the data, the main characteristic related to sparse signal processing is coherence. In particular, from the compressive sensing framework point of view [8], a signal may be successfully reconstructed as long as it is 1) sparse in some domain of representation and 2) the dictionary $\mathbf{\Phi}$ is incoherent. In the ideal scenario, and considering incoherence as the main parameter of our problem, the points $\boldsymbol{\phi}_i \in \mathbf{\Phi}$ are mutually orthogonal. This is not typically the case from a real data perspective though. This results in a situation where the input signal may not be successfully reconstructed and as a result the corresponding algorithm yields an approximate solution. Within the current work we are seeking for methodologies that improve the reconstruction performance of existing algorithms and we investigate how these methodologies alter the underlying structure of the dictionary $\mathbf{\Phi}$.

1.3 Main Contributions

The current thesis introduces reformulations of the standard framework for greedy techniques with a focus on sparse non–negative signal processing. Our work is motivated by the properties that characterise natural data and can be exploited via machine learning and deep learning techniques. Overall we address our contributions as a bridge in between data aware methods such as machine learning and traditional signal processing techniques. More specifically the main contributions of this thesis are:

1. We introduce the theoretical framework that guarantees the acquisition of the exact–Nearest Neighbor (NN) in lower dimensional spaces via linear embeddings.
2. An empirically fast and exact algorithm called Embedded Nearest Neighbor (E–NN) for the general Nearest Neighbor Search (NNS) problem [9]. A theoretically robust and practically fast algorithm called Adaptive Embedded Nearest Neighbor (AE–NN) for the general NNS. We demonstrate the benefits of following such an approach in terms of acceleration, by incorporating the algorithms in the structure of a non–negative sparse signal decomposition algorithm such as Fast Non–Negative Orthogonal Matching Pursuit (FNNOMP) [10] and accelerating the algorithm in the time domain. We extend our results to the Maximum Inner Search problem (MIPS) [11]. We demonstrate that our approaches benefit from a particular property that may characterise natural signals i.e., with lower dimensional representation.
3. Greedy techniques are a well known framework for sparse signal decomposition in their conventional form, however the associated literature is relatively poor when it comes to the learning version of greedy techniques. We here introduce an unfolded version of matching pursuit for sparse non–negative signal processing in a form of neural network [12] called Deep Matching Pursuit (DeepMP), that

boosts the performance of greedy techniques in terms of support recovery and signal reconstruction.

4. We introduce the theoretical framework that provides an interpretation upon the results of DeepMP along with modifications on the standard structure of the model that boosts its performance in terms of support–signal recovery. The associated work has been submitted to related academic journals.

1.4 Thesis Organization

In Chapter 2, we start by making a thorough review in the general sparse signal processing problem. The background of several decomposition algorithms in terms of properties and theoretical guarantees is discussed in detail. We establish the link in between sparse non–negative signal processing, and in particular greedy techniques, with data aware methodologies. We introduce the state of the art methods from this area related with the different aspects of the sparse recovery problem we are aiming to tackle.

In Chapter 3 we introduce our key novelties in the standard Nearest Neighbor problem both in theory and practice. We demonstrate the benefits, in terms of acceleration, of following such approach by incorporating the developed algorithm within the structure of FNNOMP. We compare our framework with state of the art greedy sparse non–negative decomposition algorithms.

In Chapter 4 we reformulate a conventional sparse non–negative signal processing algorithm by means of Deep Neural Network (DNN). We demonstrate the benefits of following such an approach in terms of signal recovery compared to conventional schemes. We provide a theoretical analysis upon the conditions that the particular framework may recover a signal exactly. We perform a comparison between our approach and state of the art learned sparse signal decomposition frameworks.

In Chapter 5, we conclude the outcome of the current work and we point out directions for future research.

List of Publications

1. “Accelerated Search for Non–Negative Greedy Sparse Decomposition via Dimensionality Reduction” , with M. E. Davies and M. Yahgoobi in Sensor Signal Processing for Defense (SSPD), 2019.

2. “DeepMP for Non–Negative Sparse Decomposition”, with M. E. Davies and M. Yahgoobi in European Signal Processing Conference, 2020.

Journal Submission: “Scalable Sparse Non-Negative Signal Processing using a Fast and Exact Nearest Neighbor Search” , with M. E. Davies and M. Yahgoobi in *IEEE Transactions on Signal Processing.*

Work in Progress: “Deep Matching Pursuit: A Deep Neural Network for Sparse Recovery with Theoretical Guarantees”, with M. E. Davies and M. Yahgoobi.

Chapter 2

Background

2.1 Introduction

Within the current chapter we introduce the fundamentals of sparse signal processing. We start our analysis by drawing the link between sparse signal processing and the Compressed Sensing (CS) model. Then, we introduce the sparse signal problem and the limitations that arise when considering a direct approach to solve it. We will then continue by providing an overview of existing reconstruction algorithms, the motivation these methods rely on as well as their theoretical properties. Finally, we state the objectives of the current work and we draw the link in between sparse non–signal processing and data aware methodologies.

Compressive sensing can be generally described as a technique for the efficient reconstruction and acquisition of signals by solving an undetermined system of equations. It is known for the potential to overcome the Shannon–Nyquist minimum sampling rate theorem [13] and the limitations that arise from it, e.g., the associated rate is too high, the vast majority of signals are not band limited, therefore the ideal rate may not be feasible etc. The breakthrough in the CS theory took place in 2004 when it was proven that the signals can be reconstructed by a number of measurements which is much lower than that required by the Shannon–Nyquist theorem. This is feasible under the following conditions: 1) sparsity, and 2) sensing matrix separability e.g., incoherence. During the past decade CS has made a distinct impact in several applications such as MRI [14], compressive imaging architectures [15], environmental monitoring [16].

There exist several applications in which the signals are sparse, with image and audio based applications being among the most popular ones. Within the current work, we have an interest for the explicit case of sparse non–negative signals that finds applications in several areas of daily life such as spectral and multi–spectral unmixing [17],[18], microarray analysis [19] and Raman spectral deconvolution [20] being a few examples of them.

Sparse signal recovery is a well established regime within the literature. Consequently there exists a wide range of algorithms that fulfill such task. The main focus of the current work is the development of algorithms that meet the tight criteria of real time applications. Based on this guideline, we classify the existing algorithms with respect to two basic criteria: 1) computational complexity, and 2) signal recovery.

One of the critical components upon the decomposition relies on is the dictionary $\mathbf{\Phi}$. From the perspective of signal reconstruction in the vast majority of practical applications the dictionaries do not comply with the incoherence principle. The latter may result to the degradation in the recovery performance of the corresponding algorithm. On the other hand, the size of the problem, i.e. the dimensions M, N of the dictionary $\mathbf{\Phi}$ plays a key part in the overall decomposition process in terms of operational time. In cases where the dimensions of $\mathbf{\Phi}$ are significantly large, the overall computational complexity and thus the operational time of the decomposition algorithm increase significantly.

In order to overcome the impracticalities introduced by the conventional schemes for sparse non–signal recovery, we are aiming to exploit benefits from the area of data aware methodologies such as Machine Learning (ML). In general ML is a paradigm that has attracted lots of attention in the last few years and finds applications in several areas of daily life such as image–audio processing [6][7], text retrieval [21], finance [22] etc. This is mainly due to the impressive success of a specific area of machine learning, referring to Artificial Neural Networks (ANNs), in a wide range of real–time applications in areas where ANNs deliver state of the art results. Within the current work, we address Machine learning as a broad class of algorithms where ANNs are the dominant subclass. We address the rest of the methodologies as traditional Machine Learning (tML) where i.e. Support Vector Machines (SVM) [23], logistic regression [24], etc are a few of them. On the other hand, when we are considering ANN methodologies we typically refer to them as deep learning.

A natural question that rises though, is the following: What is the new approach that data aware methodologies bring into the game from the perspective of sparse signal recovery problem? When considering the associated computational workload as a part of the problem, our main goal is the introduction of techniques that reduce the size of the problem over M, N. This can be done via methodologies that fingerprint properties that characterise natural data such as: 1) lower dimensional representation and 2) underlying structure. In the tML literature there exist techniques that employ benefits from the first property (i.e., linear embeddings [25]) and techniques such as clustering [26] that employ benefits from the second property of natural data.

Multilabel classification [27] is a well established problem in the deep learning community. From the perspective of sparse signal processing problem it is related with the detection of the atoms that constitute the input signal. The latter implies that the development of a selection rule that boosts the performance in terms of atoms identification eventually leads to an improved performance over signal recovery. In the vast majority of cases though, DL architectures are treated as black box machines that deliver outstanding results. Moreover, this type of architectures are in many cases overparameterized, which results in a heavy computational workload that makes them prohibitive from the perspective of a real–time application. A systematic approach to overcome these drawbacks can be achieved via unfolding well understood iterative algorithms by means of a Deep Neural Network [28]. The latter lays down the soil for justification of the acquired results but also controls the number of parameters which are incorporated at the decomposition process. Another approach to boost the performance of an iterative algorithm in terms of atoms identification is the so–called preconditioning [29] framework. The main notion upon preconditioning relies on, is the design of a dictionary where the mutual coherence in between the associated atoms is significantly reduced and eventually leads to an improved performance over identification of atoms. Within the current work we are aiming to investigate the various properties and benefits that characterise the different approaches and introduce frameworks that boost the performance of existing algorithms in terms of signal recovery.

2.1.1 Sparse Model

Within the current section we introduce the fundamental principles of the sparse signal processing problem. We will then discuss the limitations that arise from a computational point of view, when considering the straight forward approach as a potential strategy to solve the problem. This is done in order to highlight the essential need for alternative approaches to perform signal reconstruction, that rely on iterative schemes.

In the rest of the thesis, bold letters denote vectors and matrices while non–bold ones denote scalars. Consider a set of N coefficients $\boldsymbol{x}(i)$ for $i = 1 \dots N$. Denote the vector elements $\boldsymbol{x}(i)$ as $\boldsymbol{x}$. The vector is considered to be sparse when the number of non–zero coefficients, denoted by k, is significantly smaller than the total number of samples N. Denote the set of indices in N as the so–called "support set" s, then $\boldsymbol{x}$ is represented as follows:

$$\boldsymbol{x}(i) = 0 \text{ for } i \notin \mathrm{s} \text{ and } \boldsymbol{x}(i) = a_i \text{ for } i \in \mathrm{s} = \{s_1, \cdots, s_k\}, \text{with } k \ll N. \tag{2.1}$$

The number of nonzero coefficients is commonly denoted by $||\boldsymbol{x}||_0 = k$. The function $||\boldsymbol{x}||_0$ is typically referred as the pseudonorm.

A measurement of vector $\boldsymbol{x}$ is defined as a linear combination of its elements $\boldsymbol{x}(i)$. The m–th measurement will be denoted by $\boldsymbol{y}(m)$. Considering the case where M measurements are available, they can be written in the form of a system of M linear equations:

$$\boldsymbol{y}(m) = \sum_{j=1}^{N} \boldsymbol{\phi}_j(m)\boldsymbol{x}(j), m = 1, 2 \ldots M, \;\; M < N, \tag{2.2}$$

where $\boldsymbol{\phi}_j(m)$ are the weighting coefficients. A representation of the system in a matrix formulation can be written as:

$$\begin{bmatrix} y(1) \\ y(2) \\ \vdots \\ y(M) \end{bmatrix} = \begin{bmatrix} \phi_1(1) & \phi_2(1) & \ldots & \phi_N(1) \\ \phi_1(2) & \phi_2(2) & \ldots & \phi_N(2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_1(M) & \phi_2(M) & \ldots & \phi_N(M) \end{bmatrix} \begin{bmatrix} x(1) \\ x(2) \\ \vdots \\ x(N) \end{bmatrix}, \tag{2.3}$$

$$\boldsymbol{y} = \boldsymbol{\Phi}\boldsymbol{x}.$$

The measurements model introduced in (2.3) describes the general concept of a linear system. From a sparse signal processing perspective though, only a few atoms $\boldsymbol{\phi}_i \in \boldsymbol{\Phi} : i \in \mathrm{s}$, actually contribute to the representation $\boldsymbol{y}$. This results a compression of the general measurements model that can be written as:

$$\begin{bmatrix} y(1) \\ y(2) \\ \vdots \\ y(M) \end{bmatrix} = \begin{bmatrix} \phi_{s_1}(1) & \phi_{s_2}(1) & \ldots & \phi_{s_k}(1) \\ \phi_{s_1}(2) & \phi_{s_2}(2) & \ldots & \phi_{s_k}(2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_{s_1}(M) & \phi_{s_2}(M) & \ldots & \phi_{s_k}(M) \end{bmatrix} \begin{bmatrix} x_{s_1} \\ x_{s_2} \\ \vdots \\ x_{s_k} \end{bmatrix}, \tag{2.4}$$

$$\boldsymbol{y} = \boldsymbol{\Phi}_s\boldsymbol{x}_s.$$

Eventually, the standard problem of finding the k components that contribute to the input signal $\boldsymbol{y}$ boils down to a system of M equations with k unknowns where $k < M$. Since the number of equations is greater than the number of unknowns, the linear system for $\boldsymbol{\Phi} \in R^{M \times N}$, $\boldsymbol{y} \in R^M$ can be solved via the trivial mean squared error problem as:

$$\epsilon^2 = ||\boldsymbol{y} - \boldsymbol{\Phi}_s\boldsymbol{x}_s||^2 = (\boldsymbol{y} - \boldsymbol{\Phi}_s\boldsymbol{x}_s)^T(\boldsymbol{y} - \boldsymbol{\Phi}_s\boldsymbol{x}_s) = ||\boldsymbol{y}||^2 - 2\boldsymbol{x}_s^T\boldsymbol{\Phi}_s^T\boldsymbol{y} + \boldsymbol{x}_s^T\boldsymbol{\Phi}_s^T\boldsymbol{\Phi}_s\boldsymbol{x}_s \quad (2.5)$$

By means of derivation over the vector of unknowns, the residual error is minimized when,

$$\frac{\mathrm{d}\epsilon^2}{\mathrm{d}\boldsymbol{x}_s^T} = -2\boldsymbol{\Phi}_s^T\boldsymbol{y} + 2\boldsymbol{\Phi}_s^T\boldsymbol{\Phi}_s\boldsymbol{x}_s = 0 => \boldsymbol{x}_s = (\boldsymbol{\Phi}_s^T\boldsymbol{\Phi}_s)^{-1}\boldsymbol{\Phi}_s^T\boldsymbol{y} = \boldsymbol{\Phi}^\dagger\boldsymbol{y} \quad (2.6)$$

2.1.2 Direct search

The system of equations introduced in (2.4) when considering a direct search approach to recover $\boldsymbol{x}_s$ requires an overall number of $\binom{N}{k}$ combinations of all possible non–zero positions. Considering the pool of available solutions, the ideal one is the combination of atoms that results in a minimum mean squared error calculated as: $\epsilon^2 = ||\boldsymbol{y}-\boldsymbol{\Phi}_s\boldsymbol{x}_s||^2$. From a theoretical point of view, $\epsilon^2 = 0$ for the true solution of the problem, i.e. the correct combination of atoms. Nevertheless, the overall number of combinations is $\binom{N}{k}$. Moreover, the exact cardinality of the support $|\mathrm{s}| = k$ is unknown for the vast majority of problems.

For any reasonable size for N and k the overall number of combinations $\binom{N}{k}$ is extremely large and therefore computationally intractable since the particular problem is NP hard. Due to the practical limitations of the direct search, several reconstruction algorithms have been proposed. Within the next section we will discuss the most common approaches followed for the sparse signal recovery problem.

2.2 Sparse Signal Decomposition Algorithms

There exist several techniques proposed in the literature aiming to reconstruct a signal that is sparse in some domain of representation. A simple but efficient type of approach is the one that is introduced in the so–called greedy techniques. The main notion upon the associated algorithms, is the selection of the atom that reduces the energy of the signal the most. The most popular among these algorithms are Matching Pursuit(MP) [3] and Orthogonal Matching Pursuit (OMP) [4][5]. Another popular approach is the one that relaxes the sparsity condition via the l_1 norm. The associated algorithm is the one that solves the Basis Pursuit (BP) problem [2].

The current section, conducts a review on these methods and highlights the main idea upon these approaches rely on. At an initial stage though, we will discuss the fundamental conditions which are necessary to recover a signal exactly.

2.2.1 Reconstruction conditions

For a vector $\boldsymbol{x}$ that is k sparse and $M \geqslant 2k$ the available set of measurements, the recovery solution is unique if all the measurement subdictionaries $\mathbf{\Phi}_{2K}$ are nonsingular for all the possible positions $\{k_1, k_2, \dots, k_{2k}\}$ and for any combination of $2k$ indices from all possible measurements $\{0, 1, 2, \dots, M-1\}$. It means that $\operatorname{rank}(\mathbf{\Phi}_{2k}) = 2k$ for every $\mathbf{\Phi}_{2k}$. With respect to the aforementioned analysis, we can deduce that the solution for a k sparse signal is unique if:

$$\operatorname{spark}\{\mathbf{\Phi}\} > 2k,$$

where $\operatorname{spark}\{\mathbf{\Phi}\}$ is the spark of the matrix that corresponds to the smallest number of linearly dependent columns/rows.

For any squared matrix the corresponding determinant equals the product of the corresponding eigenvalues $\det\{\mathbf{\Phi}_{2k}\} = d_1 d_2 \dots d_{2k}$. The uniqueness condition $\operatorname{rank}(\mathbf{\Phi}_2 k)$ can then be written as:

$$\min_i \, d_i > 0$$

If a subdictionary $\mathbf{\Phi}_{2k}$ is of an order of $M \times 2k$ then the rank of this matrix can be checked by considering $\mathbf{\Phi}_{2k}^T \mathbf{\Phi}_{2k}$. It means that there is no need for combinations over measurements (to form a quadratic matrix $2k \times 2k$ from $M \times 2k$) if $M > 2k$ since the rank of $M \times 2k$ matrix $\mathbf{\Phi}_{2k}$ can be checked by calculating the rank of a $2k \times 2k$ matrix $\mathbf{\Phi}_{2k}^T \mathbf{\Phi}_{2k}$ using:

$$\operatorname{rank}(\mathbf{\Phi}_{2k}) = \operatorname{rank}(\mathbf{\Phi}_{2k}^T \mathbf{\Phi}_{2k})$$

Let us consider a practical setting where any of the determinants $\det\{\Phi_{2k}\}$ or $\det\{\Phi_{2k}^T \Phi_{2k}\}$, is nearly 0. In such case, the theoretical foundation for the uniqueness of the solution would be met. However, the analysis and potential inversion would be highly sensitive to noisy measurements. Hence, a practical requirement is that the determinant is not just different from zero, but that it sufficiently differs from zero so that an inversion stability and noise robustness is achieved.

Restricted Isometric Property (RIP): From the matrix theory we know that the norm of a matrix Φ_{2k} satisfies:

$$\lambda_{\min} \leqslant \frac{||\mathbf{\Phi}_{2k} \boldsymbol{x}_{2k}||_2^2}{||\boldsymbol{x}_{2k}||_2^2} = \frac{\boldsymbol{x}_{2k}^T \mathbf{\Phi}_{2k}^T \mathbf{\Phi}_{2k} \boldsymbol{x}_2 k}{x_{2k}^T x_{2k}} \leqslant \lambda_{\max} \tag{2.7}$$

where λ_{min} is the minimum and λ_{max} the maximum eigenvalue of the matrix $\boldsymbol{\Phi}_{2k}^T\boldsymbol{\Phi}_{2k}$, $||\boldsymbol{x}||_2^2 = |\boldsymbol{x}(0)|^2 + \ldots |\boldsymbol{x}(N-1)|^2$ is the squared l_2–norm of $\boldsymbol{x}$. The isometry property for linear transformation matrix holds if:

$$||\boldsymbol{\Phi x}||_2^2 = ||\boldsymbol{x}||_2^2 \text{ or } \frac{||\boldsymbol{\Phi x}||_2^2}{||\boldsymbol{x}||_2^2} = 1$$

The RIP condition for matrix Φ_{2k} holds if:

$$1 - \delta_{2k} \leqslant \frac{||\boldsymbol{\Phi}_{2k}\boldsymbol{x}_{2k}||_2^2}{||\boldsymbol{x}_{2k}||_2^2} \leqslant 1 + \delta_{2k} => (1-\delta_{2k})||\boldsymbol{x}_{2k}||_2^2 \leqslant ||\boldsymbol{\Phi}_{2k}\boldsymbol{x}_{2k}||_2^2 \leqslant (1+\delta_{2k})||\boldsymbol{x}_{2k}||_2^2 \tag{2.8}$$

for any $2k$–sparse vector $\boldsymbol{x}_{2k}$, where $0 \leqslant \delta_{2k} < 1$ is the isometric constant [30],[31],[32].

The main idea upon the RIP condition relies on, is that it allows the dictionary $\boldsymbol{\Phi}$ to project $\boldsymbol{x}_{2k}$ while approximately maintaining the pairwise Euclidean distances in between the pairs of points. Considering two different k–sparse vectors $\boldsymbol{x}_{2k}^{\star}$ and $\boldsymbol{x}_{2k}^{'}$, the corresponding measurements $\boldsymbol{y}^{\star}$ and $\boldsymbol{y}^{'}$ cannot live in the nullspace of $\boldsymbol{\Phi}$. The rule of thumb in general is that a smaller δ_{2k} results in an improved signal recovery. It has been proven that in cases where $\boldsymbol{\Phi}$ aligns with the RIP condition for an appropriate δ_{2k}, several reconstruction algorithms may recover the sparse signal from a set of noisy measurements [33].

From equations (2.7) and (2.8) we can write:

$$\delta_{2k} = \max\{1 - \lambda_{\text{min}}, \lambda_{\text{max}} - 1\} \tag{2.9}$$

The most common definition for the isometry constant is $\lambda_{\text{max}} - 1$ and it is calculated as the maximum eigenvalue of matrix $\boldsymbol{\Phi}_{2k}^T\boldsymbol{\Phi}_{2k} - \boldsymbol{I}$. Typically, the matrix columns are normalized. In any other case, the normalization factors should be added.

For a k–sparse vector $\boldsymbol{x}$ and a measurement matrix $\boldsymbol{\Phi}$ the RIP is satisfied if the measurement matrix satisfies the RIP for $2k$–sparse vector $\boldsymbol{x}$ with $0 \leqslant \delta_{2k} < 1$. Note that if the RIP is satisfied then $\lambda_{min} > 0$. It means that all the $\boldsymbol{\Phi}_{2k}^T\boldsymbol{\Phi}_{2k}$ subdictionaries are nonsignular [34].

The RIP property for small δ_{2k} is closer to the isometry property and introduces an improvement on the solution stability. It can be related to the conditional number of the matrix. The conditional number of the matrix $\boldsymbol{\Phi}_{2k}^T\boldsymbol{\Phi}_{2k}$ is defined as the ratio of its maximal and minimal eigenvalues:

$$\text{cond}\{\boldsymbol{\Phi}_{2k}^T\boldsymbol{\Phi}_{2k}\} = \frac{\lambda_{\text{max}}}{\lambda_{\text{min}}} \tag{2.10}$$

.

When the subdictionary $\mathbf{\Phi}_{2k}$ aligns with the RIP with δ_{2k} then:

$$\text{cond}\{\mathbf{\Phi}_{2k}^T\mathbf{\Phi}_{2k}\} \leqslant \frac{1+\delta_{2k}}{1-\delta_{2k}}$$

For small values of δ_{2k} the conditional number is very close to 1, meaning stable invertibility and low sensitivity to the input noise i.e. small variations to the measurements do not result to large deviation to the result.

Consequently, the design of the dictionary $\mathbf{\Phi}$ is of critical importance for the sparse recovery problem. Even though, it is generally difficult to verify the RIP condition for dictionaries, it has been shown that matrices such as Bernoulli, Gaussian align with the RIP condition with high probability [35].

A more practical approach to guarantee exact signal recovery is based on the mutual coherence of the corresponding dictionary.

Incoherence condition. The mutual coherence of a given dictionary $\mathbf{\Phi}$ is defined as the maximum absolute value of the normalized scalar product of its two atoms:

$$\mu(\mathbf{\Phi}) = \max_{i\neq j} \frac{|\langle \boldsymbol{\phi}_i, \boldsymbol{\phi}_j \rangle|}{||\boldsymbol{\phi}_i||_2 \cdot ||\boldsymbol{\phi}_j||_2}, \forall \phi_i, \phi_j \in \Phi. \tag{2.11}$$

The desired scenario is the one where the coherence is very small (i.e. the incoherence should be very high). In the ideal case $\mu(\mathbf{\Phi}) = 0$, i.e. the coherence matrix $\mathbf{\Phi}^T\mathbf{\Phi} = \boldsymbol{I}_N$. Nonetheless $\mu(\mathbf{\Phi})$ for a known dictionary $\mathbf{\Phi}$ with dimensions $M \times N$ $(M < N)$ cannot be arbitrarily small. The Welch upper bound relation holds:

$$\mu(\mathbf{\Phi}) \geqslant \sqrt{\frac{N-M}{M(N-1)}} \tag{2.12}$$

.

The reconstruction of a k–sparse vector $\boldsymbol{x}$ from M measurements is unique if:

$$k < \frac{1}{2}(1 + \frac{1}{\mu(\mathbf{\Phi})}) \tag{2.13}$$

.

The coherence $\mu(\mathbf{\Phi})$ can be used to determine the lower bound in the spark as follows:

$$\text{spark}(\mathbf{\Phi}) \geqslant (1 + \frac{1}{\mu(\mathbf{\Phi})}) \tag{2.14}$$

If $\boldsymbol{x}$ is a solution of a system of equations $\boldsymbol{y} = \boldsymbol{\Phi x}$, such that:

$$||\boldsymbol{X}||_0 = k < \frac{1}{2}(1 + \frac{1}{\mu(\boldsymbol{\Phi})}) \leqslant \frac{1}{2}\text{spark}(\boldsymbol{\Phi}) \tag{2.15}$$

then $\boldsymbol{x}$ is the sparsest solution. The RIP condition can be bounded by means of $\mu(\boldsymbol{\Phi})$ as follows:

$$\delta_{2k} \leqslant (1 - k)\mu(\boldsymbol{\Phi}). \tag{2.16}$$

2.2.2 l_0 norm based reconstruction

The formal approach to perform signal reconstruction, states that $\boldsymbol{x}_s$ can be directly acquired by its measurements represented by $\boldsymbol{y}$, by finding the sparsest $\boldsymbol{x}$ that maps onto $\boldsymbol{y}$. Hence, by introducing the number of non–zero coefficients by means of the l_0 pseudonorm: $k = ||\boldsymbol{x}||_0$. The optimization problem can then be formulated as in (2.17):

$$\begin{aligned} \tilde{x} := \quad & \text{argmin } ||\boldsymbol{x}||_0 \\ & \boldsymbol{y} = \boldsymbol{\Phi x} \end{aligned} \tag{2.17}$$

The minimization problem introduced in (2.17) can be solved in an implicit way by a class of algorithms. For example, in specific applications we may predict the number of components or we can estimate the position of non–zero coefficients. In such case the computational complexity can be significantly reduced compared to the direct search method [36],[3]. In general though, the l_0 norm minimization problem is NP hard. This results in an optimization framework where the computational workload is rather intensive. The most common approach to fulfil such task is via an exhaustive search. An alternative approach is the one that uses a series of smoothed objectives, that eventually converges to l_0 in the limit. This type of approach is optimized in an iterative manner [37].

2.2.3 l_0 sparse approximation

The standard l_0–norm based minimization problem counts the number of non–zero coefficients and can be formulated as: $||\boldsymbol{x}||_0 = \sum_{i=1}^{N} f(x_i)$, where,

$$f(x_i) = \begin{cases} 0 & x_i = 0 \\ 1 & \text{otherwise} \end{cases} \tag{2.18}$$

The original minimization problem introduced in (2.17), can be relaxed by means of Lagrangian multipliers as follows:

$$\tilde{x} := \text{argmin } ||\boldsymbol{y} - \boldsymbol{\Phi x}||_2^2 + \lambda \sum_{i=1}^{N} f(x_i) \tag{2.19}$$

A popular approach that solves the problem introduced in (2.19) is the so–called Iterative Hard Thresholding (IHT) algorithm. This sort of approach implements the gradient descent with a thresholding operation in an iterative manner:

$$\boldsymbol{x}^{k+1} = H(\boldsymbol{x}^n - \boldsymbol{\Phi}^T(\boldsymbol{y} - \boldsymbol{\Phi x}^k), b) \tag{2.20}$$

The $H(\boldsymbol{a}, b)$ operator is the the so–called hard–thresholding operator [38] which sets all the elements in $\boldsymbol{a}$ below b (in terms of magnitudes) to 0, while b is the thresholding constant and $\boldsymbol{x}^k$ represents the estimated $\boldsymbol{x}$ at the k–iteration. IHT has been successfully used in the compressed sensing regime [39].

2.3 l_1–norm based reconstruction

As previously mentioned, the minimization of the l_0 norm is an NP hard problem that cannot be solved practically via an iterative algorithm or by means of linear programming. In order to avoid dealing with the complexity of an NP hard problem, such as the l_0 minimization problem, alternative approaches have been followed in order to reconstruct a sparse signal.

It has been proven, that under specific requirements, the l_1–norm minimization problem produces the same outcome as the l_0–norm minimization [40][41][42][43]. The l_1 norm reconstruction problem is then formulated as introduced in (2.21). In general, the equivalence of the l–0 norm to the l_1–norm is defined by the restricted isometric property (RIP).

$$\begin{aligned} \tilde{\boldsymbol{x}} := \quad & \underset{\boldsymbol{x}}{\text{argmin}} \; ||\boldsymbol{x}||_1 \\ & \text{s.t } \boldsymbol{y} = \boldsymbol{\Phi x} \end{aligned} \tag{2.21}$$

For a k sparse vector and a dictionary $\boldsymbol{\Phi}$, the solution of the l–0 pseudo norm minimization problem is equivalent to (2.21) as long as:

$$1 - \delta_{2K} \leqslant \frac{||\boldsymbol{\Phi}_{2k}\boldsymbol{x}_{2k}||_2^2}{||\boldsymbol{x}_{2k}||^2} \leqslant 1 + \delta_{2k} \tag{2.22}$$

where $0 \leqslant \delta_{2k} < \sqrt{2} - 1$ for all the submatrices of the dictionary $\boldsymbol{\Phi}$ [32]. For the l_0–norm the RIP constant range was $0 \leqslant \delta_{2k} < 1$ [34].

The optimization problem introduced in (2.21) is more formally known as the Basis Pursuit (BP) problem and it was proposed by the authors in [2]. There exist two basic techniques aiming to solve the BP problem. The original paper proposes interior–point methods of linear programming. An alternative approach to that, was proposed by the authors in [44] where BP is solved via a Block Coordinate Relaxation method. A key characteristic of these techniques though is that they are characterised by a heavy computational workload: "Strictly speaking BP is not an algorithm but a principle" [45].

The problem introduced in (2.21) can be reformulated in various ways. One such approach is the Langrangian formulation where the problem is formulated as:

$$\begin{aligned} f(x) &= \underset{x}{\operatorname{argmin}}\ ||\boldsymbol{y} - \boldsymbol{\Phi x}||_2^2 + \lambda||\boldsymbol{x}||_1 \\ &= ||\boldsymbol{y}||^2 - \boldsymbol{X}^T\boldsymbol{\Phi}^T\boldsymbol{y} - \boldsymbol{y}^T\boldsymbol{\Phi x} + \boldsymbol{x}^T\boldsymbol{\Phi}^T\boldsymbol{\Phi x} + \lambda\boldsymbol{x}^T\text{sign}(x) \end{aligned} \tag{2.23}$$

The solution of the optimization problem introduced in (2.23) is unique. This result can be proven by showing the convexity of the quadratic part. The remaining part of the cost function $||\boldsymbol{x}||_1$ is convex and the objective function is not bounded when $||\boldsymbol{x}|| \to \infty$. Nevertheless a necessary condition to reconstruct the sparse signal $\boldsymbol{x}$ exactly, is the uniqueness of the solution. This is satisfied in cases where the input signal is sufficiently sparse and the dictionary $\boldsymbol{\Phi}$ satisfies the so–called Exact Recovery Condition (ERC) [46, pp31].

The particular formulation of the sparse signal reconstruction problem in (2.23) introduces a tradeoff in between data fidelity at the minimization error $\boldsymbol{y} - \boldsymbol{\Phi x}$ and the minimization of the l_1–norm which can be addressed as a form of regularisation. The main scope of the regularization is to promote sparsity. There are many ways to solve the stated problem with respect to the Langrangian formulation. One of the most popular approaches is the so called Least Absolute Selection and Shrinkage Operator (LASSO) formulation.

The LASSO formulation with the l_1 norm minimization term does not have a

Algorithm 1 Iterative Shrinkage Thresholding Algorithm.

```
Initialisation: $x = \vec{0}, L >$ largest eigenvalue of $\boldsymbol{\Phi}^T\boldsymbol{\Phi}$
while $k <$ some fixed threshold
    $\boldsymbol{x}^{k+1} = \text{soft}(z, \theta)(\boldsymbol{x}^k + \frac{1}{\lambda}\boldsymbol{\Phi}^T(\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{x}^k))$
end
$x \leftarrow \sum_k s_k$
```

closed form solution. It is typically solved via an iterative approach. In order to derive such an iterative scheme a nonnegative term is $h(\boldsymbol{x})$ with zero values as the solution $\boldsymbol{x}_s$ is added to the cost function introduced in (2.23), where $h(\boldsymbol{x})$ is defined as:

$$h(\boldsymbol{x}) = (\boldsymbol{x} - \boldsymbol{x}_s)^T(L\boldsymbol{I} - \boldsymbol{\Phi}^T\boldsymbol{\Phi})(\boldsymbol{x} - \boldsymbol{x}_s), \tag{2.24}$$

$h(\boldsymbol{x})$ does not change the solution of the original minimization problem. The new cost function is then defined as:

$$g(\boldsymbol{x}) = f(\boldsymbol{x}) + (\boldsymbol{x} - \boldsymbol{x}_s)^T(L\boldsymbol{I} - \boldsymbol{\Phi}^T\boldsymbol{\Phi})(\boldsymbol{x} - \boldsymbol{x}_s), \tag{2.25}$$

where the constant L is a constant added such that the additional term is always nonnegative [34]. This essentially means that $L > \lambda_{\max}$, where $\lambda_{\max}$ corresponds to the largest eigenvalue of $\boldsymbol{\Phi}^T\boldsymbol{\Phi}$. The gradient operator of $g(\boldsymbol{x})$ then reads:

$$\nabla g(\boldsymbol{x}) = \frac{\partial g(\boldsymbol{x})}{\partial \boldsymbol{x}^T} = -2\boldsymbol{\Phi}^T\boldsymbol{y} + 2\boldsymbol{\Phi}^T\boldsymbol{\Phi}\boldsymbol{x} + \lambda\text{sign}(\boldsymbol{x}) + 2(L\boldsymbol{I} - \boldsymbol{\Phi}^T\boldsymbol{\Phi})(\boldsymbol{x} - \boldsymbol{x}_s) \tag{2.26}$$

The solution $\nabla g(\boldsymbol{x}) = 0$ then reads:

$$\begin{aligned} \boldsymbol{\Phi}^T\boldsymbol{y} + \frac{\lambda}{2}\text{sign}(\boldsymbol{x}) - (L\boldsymbol{I} - \boldsymbol{\Phi}^T\boldsymbol{\Phi})\boldsymbol{x}_s + L\boldsymbol{x} = 0 \\ => \boldsymbol{x} + \frac{\lambda}{2L}\text{sign}(\boldsymbol{x}) = \frac{1}{L}\boldsymbol{\Phi}^T(\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{x}_s) + \boldsymbol{x}_s \end{aligned}$$

The associated iterative scheme is then derived as follows:

$$\boldsymbol{x}_{k+1} + \frac{\lambda}{2}\text{sign}\boldsymbol{x}_{k+1} = \frac{1}{L}\boldsymbol{\Phi}^T(\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{x}_k) + \boldsymbol{x}_k$$

The soft thresholding rule is then used as solution for the following equation:

$$\boldsymbol{x} + \lambda\text{sign}(\boldsymbol{x}) = \boldsymbol{y} \tag{2.27}$$

Algorithm 2 MP

```
Initialization: $s = \varnothing, j = 0, \boldsymbol{r}_0 = \boldsymbol{y}$.
while $j < K$ & $\max(\boldsymbol{\Phi}^T \boldsymbol{r}_k > 0)$.
    i $[\mu, \iota]$=max $|\boldsymbol{\Phi}^T \boldsymbol{r}_j|$
    ii $s = s \cup \mu$.
    iii $\boldsymbol{r}_{j+1} = \boldsymbol{r}_j - \boldsymbol{\Phi}_\mu \iota$
    iv $j \leftarrow j + 1$
end
```

It is defined by the function soft as follows:

$$x_i = \text{soft}(z, \theta) = \begin{cases} z + \theta & z < -\theta \\ 0 & \text{if } |z| \leqslant \theta \\ z - \theta & \text{if } z > \theta \end{cases} \tag{2.28}$$

or alternatively,

$$\text{soft}(z, \theta) = sign(z) \max\{0, |z_i| - \theta_i\} \tag{2.29}$$

The corresponding iterative scheme is the so–called Iterative Soft–Thresholding Algorithm (ISTA) [47] as introduced in **Algorithm 1**. The langrangian constant λ is used as a form of a balance term in between the data fidelity error and the l_1 norm. $\eta_{\lambda/L}$ is the so–called sofmax function defined in (2.29). The thresholds θ_i are typically set to $\theta_i = \frac{\lambda}{L}$, where L is commonly set to : $L = \max\{\text{eig}\{\boldsymbol{\Phi}^T \boldsymbol{\Phi}\}\}$ (where eig denotes the eigenvalues of the matrix). The standard setting for ISTA can be modified to accelerate the convergence via the Fast ISTA (FISTA) algorithm. Note though, that these are just a few of the approaches to solve (2.23) [47].

2.4 Greedy techniques

The class of convex optimization algorithms that solves the least squares problem, as introduced in (2.30), is proven to yield sparse solutions with a relatively low computational cost. The most popular techniques aiming to solve the problem introduced in (2.30) are the so–called greedy techniques. This type of techniques is known for its simplicity and the relatively low computational cost that makes them appealing from the perspective of a real time application.

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & ||\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}||_2^2 \\ \text{subject to} \quad & ||\boldsymbol{x}||_0 = k. \end{aligned} \tag{2.30}$$

2.4.1 Matching Pursuit

A simple greedy algorithm is called Matching Pursuit (MP), as introduced in **Algorithm 2**, and it was firstly proposed by the authors in [3]. MP builds the sparse representation of a signal by iteratively adding the most correlated element of the dictionary, more formally known as the atom, to the set of selected elements. MP has drawn the attention of various research communities, i.e. in the statistics community where it is more formally known as Projection Pursuit [48],[49] or the approximation community where it is more formally called the Pure greedy algorithm [50].

In a finite dimensional space, MP has been proven to converge exponentially [51],i.e. for some $0 < \beta < 1$,

$$||\boldsymbol{r}_m||^2 = ||\boldsymbol{y}_m - \boldsymbol{y}||^2 \leqslant \beta^m, m \geqslant 1. \tag{2.31}$$

Moreover, the authors in [51] introduced the criteria that guarantee that MP will select correct atoms within the first k iterations of the algorithm. The results introduced in the particular work rely on the so–called stability condition. In practice the corresponding analysis is an extension of the results for OMP introduced [45] and the so–called Exact Recovery Condition (ERC). The ERC condition will be more thoroughly discussed within the part of the subsection of Orthogonal Matching Pursuit (OMP).

2.4.2 Orthogonal Matching Pursuit

One of the key disadvantages of MP is that the representation found by the algorithm is not the best representation using the selected atoms. Moreover, the algorithm, may also reselect already selected atoms in the later iterations. In that sense, even in cases where the dictionary $\boldsymbol{\Phi}$ and the sparsity of the signal k comply with the stability condition criteria, this means that after k iterations we may end up with an s : |s| $< k$ or in other words will s not be fully recovered. As a result of the particular characteristic, the convergence of the algorithm may slow down.

Orthogonal Matching Pursuit (OMP) algorithm was introduced to compensate these issues [4],[5]. This is achieved via the incorporation of an orthogonalization pro-

Algorithm 3 OMP

```
Initialization: $s = \emptyset, j = 0, \boldsymbol{r}_0 = \boldsymbol{y}$.
**while** $j < K$ & $\max(\boldsymbol{\Phi}^T \boldsymbol{r}_k > 0)$.
    i $[\mu, \iota]$=max $|\boldsymbol{\Phi}^T \boldsymbol{r}_j|$
    ii $s = s \cup \mu$.
    iii $\boldsymbol{r}_{j+1} = \boldsymbol{r}_j - \boldsymbol{\Phi}_s \boldsymbol{\Phi}_s^{\dagger} \boldsymbol{y}$
    iv $j \leftarrow j + 1$
**end**
**output:**$\boldsymbol{x}_s = \boldsymbol{\Phi}_s^{\dagger} \boldsymbol{y}$
```

cedure that projects the input signal $\boldsymbol{y}$ onto the support set s. Such a projection can be found by,

$$\tilde{\boldsymbol{x}}_s := \underset{\boldsymbol{x}_s}{\text{argmin}}||\boldsymbol{y} - \boldsymbol{\Phi}_s \boldsymbol{x}_s||, \tag{2.32}$$

where $\boldsymbol{\Phi}_s$ and $\boldsymbol{x}_s$ are respectively the sub–dictionary and coefficient vector restricted to the support s, while $\boldsymbol{\Phi}_s^{\dagger}$ is defined as in (2.33). The OMP algorithm is analytically presented in **Algorithm 3**.

$$\boldsymbol{\Phi}_s^{\dagger} = (\boldsymbol{\Phi}_s^T \boldsymbol{\Phi}_s)^{-1} \boldsymbol{\Phi}_s \tag{2.33}$$

On the other hand this type of methodology yields solutions which empirically are in general less accurate compared to the minimization of the ℓ_1 norm [2]. However, the authors in [45] set up the conditions that guarantee the reconstruction of the exact signal $\boldsymbol{x}_s$ within the first k iterations of the algorithm.

The main outcome of the work introduced in [45] is that for certain type of dictionaries, the so–called quasi–incoherent dictionaries, OMP will recover the exact signal (or in other words no false positive will be introduced in s) under the so–called Exact Recovery Condition (ERC) which is as follows:

$$\max_{\boldsymbol{y}}||\boldsymbol{\Phi}_s^{\dagger} \boldsymbol{y}||_1 < 1, \tag{2.34}$$

where $\boldsymbol{\Phi}_s^{+}$ corresponds to the sub–dictionary of points that do not contribute in $\boldsymbol{y}$. Fundamentally, the ERC condition states that the sparsest representation of $\boldsymbol{y}$ is unique. From the perspective of the iterative scheme introduced in **Algorithm 3** this means that no false positive will be introduced within the first k iterations of the algorithm and the signal $\boldsymbol{x}_s$ will be exactly recovered after k iterations. Note though that the ERC condition holds for every signal with an k–representation provided that:

$$k < \frac{1}{2}(\mu^{-1} + 1) \tag{2.35}$$

The bound introduced in (2.35) can be reformulated as:

$$k < \frac{1}{2}(\frac{1}{\mu} + 1) => 2k < \frac{1}{\mu} + 1 => 2k\mu < 1 + \mu => 2k\mu - \mu < 1 \tag{2.36}$$

The ERC more generally holds whenever:

$$\mu_1(k-1) + \mu_1(k) < 1 \tag{2.37}$$

where μ and $\mu_1(k)$ are defined in equations (2.38), (2.39) respectively:

$$\mu = \max_{i \neq j} \frac{|\langle \boldsymbol{\phi}_i, \boldsymbol{\phi}_j \rangle|}{||\boldsymbol{\phi}_i||_2 \cdot ||\boldsymbol{\phi}_j||_2} \tag{2.38}$$

$$\mu_1(k) = \max_{i} \max_{|s|=K, i \notin s} \sum_{j \in s} |\langle \phi_i, \phi_j \rangle| \tag{2.39}$$

The function introduced in (2.39) is known as the Babel function and can be upper bounded as follows:

$$\mu_1(k) \leqslant k\mu, \tag{2.40}$$

and therefore

$$\mu_1(k-1) \leqslant (k-1)\mu \tag{2.41}$$

$$\overset{(2.40)+(2.41)}{=>} \quad \mu_1(k) + \mu_1(k-1) \leqslant k\mu + (k-1)\mu = 2k\mu - \mu < 1 \tag{2.42}$$

Hence, the bound introduced by the Babel function as in (2.37) is a relaxation of the bound introduced in (2.35) and with respect to the reformulation introduced in (2.36). In order to get a more intuitive interpretation of the bound introduced in (2.35) we can think of two hypothetical scenarios. Note that this sort of practical intuition will be followed when we discuss about other bounds as well. We follow this type of approach to provide a more practical understanding regarding the different bounds and their practical limitations.

For the first hypothetical case we can assume that we are interested in an application where more or less we know that the maximum number of contributing components in $\boldsymbol{y}$ is $k = 3$. By plugging this size of k into (2.35), we can derive that we need a dictionary $\boldsymbol{\Phi}$ with $\mu < 0.2$ to make sure that every signal with $k = 1, \ldots 3$, is exactly recovered within the first k iterations of OMP.

For the second hypothetical case we will address the whole issue the other way around, i.e., we are given a specific dictionary $\mathbf{\Phi}$, hence μ is known, therefore we are seeking to figure out the maximum level of sparsity in x_s so that the signal can be exactly recovered. Let as assume that $\mu = 0.5$, by plugging this quantity into (2.35) then we have that $k < 1.5$, hence the maximum level of sparsity we fully recover is for $k = 1$. The numerical examples provided here, are given in order to highlight that these bounds hold for specific type of dictionaries. These bounds hold typically for some k's for the so–called *quasi–incoherent* dictionaries, i.e. dictionaries where the cumulative coherence function grows slowly. Therefore, in the most commonly met dictionaries false positives are introduced in the selection step of the algorithm and eventually OMP yields approximate solutions.

2.4.3 OMP Implementations

From the perspective of OMP there exist several implementations that may be found in the literature. The straight forward approach is the one that incorporates the so–called matrix inversion lemma and constructs $\mathbf{\Phi}^\dagger$ at each iteration of the algorithm. At iteration k the particular approach requires the inversion of a $k \times k$ matrix. When k increases, the computational workload becomes very expensive, i.e., $\mathcal{O}(k^3)$. In order to bypass the high computational workload implied by the naive approach matrix factorization techniques have been proposed. QR factorization is among the most popular ones. By incorporating QR factorization we reformulate the subdictionary $\mathbf{\Phi}_s$ as introduced in (2.43).

$$\mathbf{\Phi}_s = \boldsymbol{Q}_s \cdot \boldsymbol{R}_s \tag{2.43}$$

Algorithm 4 OMP via QR factorization

```
Initialization: $s = \varnothing, j = 0, r_0 = y, \xi = \varnothing$.
while $j < K$ & $\max(\Phi^T r_k > 0)$.
      i  $[\mu, \iota]$=max $|\Phi^T r_j|$
     ii  $s = s \cup \mu$.
    iii  Update $Q, R$ factors.
     iv  $z_j = Q_j^T y$.
      v  $\xi \leftarrow [z_j, \xi]$
     vi  $r_{j+1} = r_j - Q_j z_j$
    vii  $j \leftarrow j + 1$
end
$x_s = R_s^{-1} \xi$
```

Algorithm	Complexity
Pseudoinverse	$MN + Mk + Mk^2 + k^3$
QR	$MN + Mk$; update (step 4) k^2

Table 2.1: Computational Complexity of different OMP implementations

From the perspective of an online algorithm such as OMP, the $\boldsymbol{Q}, \boldsymbol{R}$ factors can be constructed by means of the Gram–Schmidt procedure on an iterative manner. At the $k+1$–th iteration the $\boldsymbol{Q}, \boldsymbol{R}, \boldsymbol{R}^{-1}$ are computed as follows:

$$\boldsymbol{Q}_{k+1} = \begin{bmatrix} \boldsymbol{Q}_k & \boldsymbol{q}_{k+1} \end{bmatrix}, \ \boldsymbol{R}_{k+1} = \begin{bmatrix} \boldsymbol{R}_k & v \\ a & \omega \end{bmatrix}, \ \boldsymbol{R}_{k+1}^{-1} = \begin{bmatrix} \boldsymbol{R}_k^{-1} & -\frac{\boldsymbol{R}_k^{-1} v}{\omega} \\ 0 & \frac{1}{\omega} \end{bmatrix}, \tag{2.44}$$

where $\boldsymbol{q}_{k+1} = \frac{\boldsymbol{\psi_{k+1}}}{||\boldsymbol{\psi}_{k+1}||}$, $\boldsymbol{\psi}_{k+1} = (\boldsymbol{I} - \boldsymbol{Q}_k \boldsymbol{Q}_k^T)\boldsymbol{\phi}_{k+1}$,$v = \boldsymbol{Q}^T \boldsymbol{\phi}_{k+1}$ and $\omega = ||\boldsymbol{\psi}_{k+1}||$.

Based on (2.43) the Moore–Penrose pseudoinverse can be reformulated as follows:

$$\boldsymbol{\Phi}_s^\dagger = (\boldsymbol{\Phi}_s^T \boldsymbol{\Phi}_s)^{-1} \boldsymbol{\Phi}_s^T \overset{(3.37)}{=} ((\boldsymbol{Q_s R_s})^T (\boldsymbol{Q_s R_s}))^{-1} \boldsymbol{R}_s^T \boldsymbol{Q}_s^T \overset{\boldsymbol{Q}_s^T \boldsymbol{Q}_s = I_{|s|}}{=} \boldsymbol{R}_s^{-1} \boldsymbol{R}_s^{-T} \boldsymbol{R}_s^T \boldsymbol{Q}_s^T = \boldsymbol{R}_s^{-1} \boldsymbol{Q}_s^T \tag{2.45}$$

In practice the QR factorization provides a simplified and computationally cheaper approach to construct the Moore–Penrose pseudoinverse. From the practicalities of the problem, the outcome of (2.45) simply highlights that similarly to the output for $\boldsymbol{\Phi}^\dagger$ that comes from (2.6), in cases where the support set is successfully acquired over the k iterations, then OMP by means of QR factorization fully reconstructs $\boldsymbol{x}_s$. According to the study conducted by the authors in [52], the implementation that incorporates the QR factorization within the iterative procedure is the fastest among the proposed ones for medium to large problem sizes (i.e., large N). An analytical derivation in terms of computational complexity for the cost of the k–th iteration is provided in Table 2.1.

A possibly naive question that nevertheless occurred to the author though is the following: Given an implementation of OMP with respect to $\boldsymbol{\Phi}^\dagger$ and an implementation of OMP with respect to the QR factorization, should we expect that the two approaches will yield a different solution after k iterations? The output of (2.45) demonstrates that the two approaches are equivalent when the same support s is given. So, in practice the question is whether the two approaches yield the same s after k iterations. In order to fingerprint the answer to this question we will perform an analysis over the iterations of the two approaches.

Given a set of measurements $\boldsymbol{y}$, within the first iteration of the corresponding implementation, the two approaches will pick exactly the same candidate given that the selection rule: $\max\limits_{\boldsymbol{\phi}_i \in \boldsymbol{\Phi}} |\boldsymbol{\phi}_{\boldsymbol{i}}^T \boldsymbol{y}|$ is identical (similarly MP will always select the same candidate with any OMP implementation for the first iteration). So the question is whether they will pick the same candidate within the next iteration. In order to show that, we have to demonstrate that within the next iteration the same residual set of measurements $\boldsymbol{r}_1$ will be provided at the selection step of the algorithm. Assuming that within the first iteration of the algorithm the two implementations of the algorithm pick $\boldsymbol{\phi}_1$ as a potential candidate. Then the update step of the two approaches is then formulated as follows:

1) OMP implementation with respect to $\boldsymbol{\Phi}^{\dagger}$:

$$\boldsymbol{r}_1 = \boldsymbol{y} - \boldsymbol{\phi}_1(\boldsymbol{\phi}_1^T \boldsymbol{\phi}_1)^{-1} \boldsymbol{\phi}_1^T \boldsymbol{y} = \boldsymbol{y} - \boldsymbol{\phi}_1 \boldsymbol{\phi}_1^{\dagger} \boldsymbol{y} \tag{2.46}$$

2) OMP implementation with respect to the QR factorization:

$$\boldsymbol{r}_1^{'} = \boldsymbol{y} - \boldsymbol{Q}_1 \boldsymbol{z}_1, \tag{2.47}$$

where $z_1 = \boldsymbol{Q}_1^T \boldsymbol{y}$. Note that with respect to the QR factorization we have the following relationship:

$$\boldsymbol{\phi}_1 = \boldsymbol{Q}_1 \boldsymbol{R}_1 => \boldsymbol{\phi}_1 \boldsymbol{R}_1^{-1} = \boldsymbol{Q}_1 \tag{2.48}$$

By plugging the outcome of equation (2.48) into (2.47) the linear relationship can be rewritten as follows:

$$\boldsymbol{r}_1^{'} = \boldsymbol{y} - \boldsymbol{Q}_1 \boldsymbol{z}_1 = \boldsymbol{y} - \boldsymbol{\phi}_1 \boldsymbol{R}_1^{-1} \boldsymbol{Q}_1^T \boldsymbol{y} \overset{(2.45)}{=} \boldsymbol{y} - \boldsymbol{\phi}_1 \boldsymbol{\phi}_1^{\dagger} \boldsymbol{y} \tag{2.49}$$

From (2.49) and (2.46) we have that $\boldsymbol{r}_1 = \boldsymbol{r}_1^{'}$. So, within the second iteration of the different implementations once again the same atom will be selected provided that their identical selection rule is given the same set of residual measurements. By repeating the aforementioned procedure we can show that the two implementations will result to an identical support set and consequently to an identical reconstructed signal $\boldsymbol{x}$. In practice, the OMP implementation by means of the QR factorization can be addressed as a faster implementation of the $\boldsymbol{\Phi}^{\dagger}$ for medium to large dictionaries $\boldsymbol{\Phi}$.

A key difference of OMP, and in general any kind of algorithm that incorporates an orthogonalization procedure, is that it converges in terms of $||\boldsymbol{y} - \hat{\boldsymbol{y}}||$ (where $\boldsymbol{y}$ the input signal and $\hat{\boldsymbol{y}}$ the reconstructed signal by the associated algorithm) within a finite number of steps (N in case of OMP). The formulation of OMP via the QR factorization sheds the light on this characteristic. The underlying procedure that takes place via orthogonalization is the construction of an orthonormal basis $\boldsymbol{Q}$. Within the j–th iteration of the algorithm, OMP constructs and eventually projects $\boldsymbol{y}$ onto the j–th element of the basis. Nonetheless, a signal $\boldsymbol{y} \in R^N$ can be analytically expressed in any N–orthonormal basis $\boldsymbol{Q}_N$:

$$\boldsymbol{Q}_N \cdot \boldsymbol{Q}_N^T = \boldsymbol{I}_N = \boldsymbol{Q}_N^T \boldsymbol{Q}_N, \tag{2.50}$$

where $\boldsymbol{I}_N$ the identity matrix.

For example, let us assume that an N–orthonormal basis is given. The projection of $\boldsymbol{y}$ onto this basis is obtained as follows:

$$\boldsymbol{y}^{'} = \boldsymbol{Q}_N^T \cdot \boldsymbol{y} \overset{\boldsymbol{Q}_N^T \cdot \boldsymbol{Q}_N = \boldsymbol{I}_N}{=>} \boldsymbol{Q}_N \cdot \boldsymbol{y}^{'} = \boldsymbol{y} \tag{2.51}$$

Within the N–th iteration of OMP via QR factorization the update step of the algorithm reads:

$$\boldsymbol{r}_{n+1} = \boldsymbol{r}_n - \boldsymbol{Q}_n z_n = \boldsymbol{y} - \boldsymbol{Q}_N \boldsymbol{z} \overset{(2.51)}{=>} \boldsymbol{r}_{n+1} = \boldsymbol{0}, \tag{2.52}$$

$\boldsymbol{Q}_n$ is the n–th element of the orthonormal basis, $\boldsymbol{Q}_N$ the full orthonormal basis and $\boldsymbol{z} \in R^N$. In order to draw the equivalence between equations (2.51) and (2.52) we can address the process as follows: From the perspective of OMP the orthonormal basis $\boldsymbol{Q}_N$ and eventually the coefficients that correspond to the projection of each element of the basis expressed by $\boldsymbol{y}^{'}$ are unknown. At each iteration of OMP, we add a new orthonormal element $\boldsymbol{Q_j}$ at the orthonormal matrix $\boldsymbol{Q}$, and calculate the associated coefficient z_j which is then added at the vector $\boldsymbol{z}$. After N iterations $\boldsymbol{Q}$ is a full orthonormal basis and $\boldsymbol{z}$ a vector with N entries similarly to $\boldsymbol{y}^{'}$.

2.4.4 Compressive Sampling Matching Pursuit

Another variant of the MP framework is the so called Compressive Sampling Matching Pursuit (CoSaMP) algorithm introduced in [53]. Within the particular framework a k–sparse signal is reconstructed on an iterative manner. At the selection step of the algorithm, the measurements of the residual vector $\boldsymbol{r}$ are projected onto the atoms of the dictionary $\boldsymbol{\Phi}$ and the $2k$ atoms with the highest contribution in y are selected as potential candidates. The set of potential candidates is then expanded with respect to the candidates selected at the preceding iteration. At the next step of the algorithm, the pseudoinverse $\boldsymbol{\Phi}^{\dagger}$ is constructed with respect to the extended set of candidates and it is then projected onto the measurements $\boldsymbol{y}$. Eventually, the k elements with the

Algorithm 5 CoSaMP

```
Input: $k, \boldsymbol{\Phi}, \boldsymbol{y}$.
Initialization: $\boldsymbol{x} \leftarrow 0_{N\times 1}, \boldsymbol{r}_0 = \boldsymbol{y}$
while $j < K$
    1. $S_1 \leftarrow$positions of $2k$ highest values in $\boldsymbol{\Phi}^T \boldsymbol{r}_k$
    2. $S_2 \leftarrow$ positions of nonzero coefficients in $\boldsymbol{x}$.
    3. $S = S_1 \cup S_2$
    4. $\boldsymbol{\Phi}_S \leftarrow$ columns from matrix $\boldsymbol{A}$ selected by set $S$.
    5. $\boldsymbol{c} = (\boldsymbol{\Phi}_S^T \boldsymbol{\Phi}_S)^{-1} \boldsymbol{\Phi}_S^T \boldsymbol{y}$
    6. Select the $k$ coefficients with the highest magnitudes in $\boldsymbol{c}$ and place them at
       the corresponding index in $\boldsymbol{x}$.
    7. $\boldsymbol{r}_k = \boldsymbol{y} - \boldsymbol{\Phi x}$
    8. $j \leftarrow j + 1$
end
```

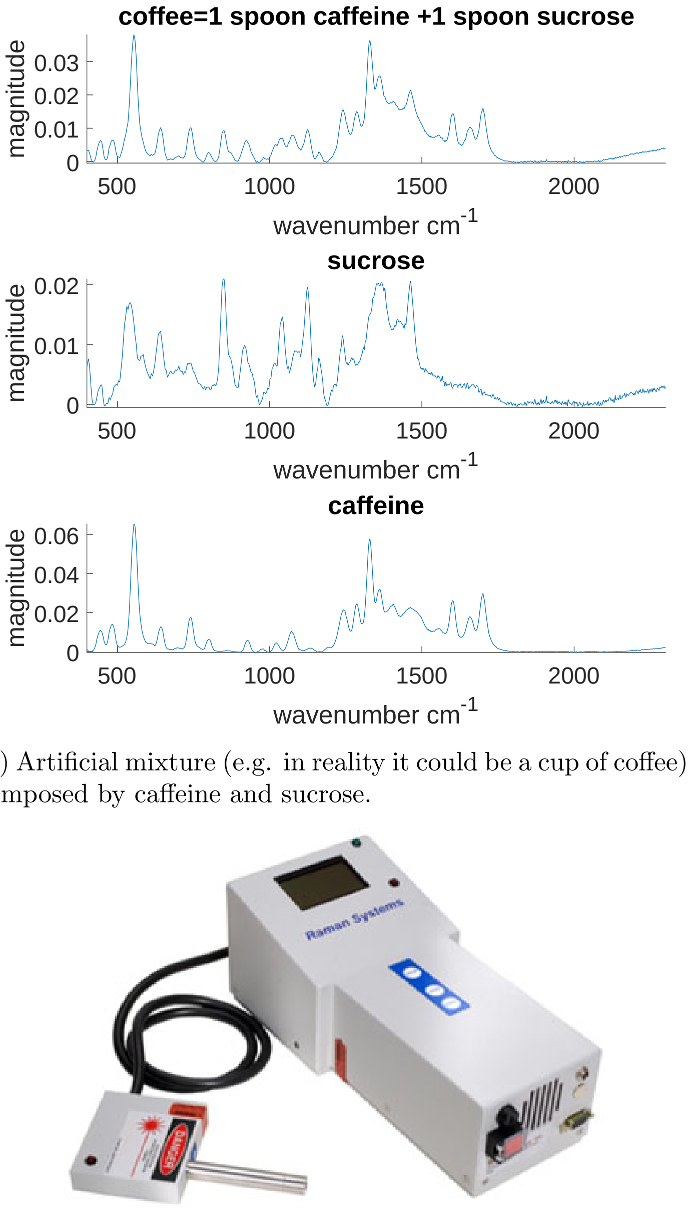
highest contribution are selected as the current estimate of the sparse signal $\boldsymbol{x}$. The new measurements $\boldsymbol{r}_k$ are then obtained by subtracting the current estimate from $\boldsymbol{y}$, and the process repeats until the terminating criterion is met. The overall procedure is demonstrated in **Algorithm 5**.

2.4.5 Sparse Non–Negative Signal Processing

The preliminary focus of this project is on the development of non–negative sparse signal processing applications. There are many applications for which the coefficient vectors are not only sparse, but they are also non–negative. Spectral and multi–spectral unmixing [17],[18], microarray analysis [19] and Raman spectral deconvolution [20] are a few examples. The development of these applications may be beneficial in several areas such as defense, security or food industry.

One of the characteristic cases where sparse non–negative signal processing could be of a major interest is Raman spectroscopy. This type of technique is used to observe vibrational, rotational and other low–frequency modes in a system. The method is widely used in several scientific disciplines such as cancer detection [54], nanotechnology [55] as well as in industrial applications [56], as well as in the pharmaceutical industry [57],[58] where *Raman* spectroscopy is applied in order to identify pharmaceutical ingredients and their polymorphic forms, but also in food safety [59] and hazardous materials detection [60].

Even though Raman spectroscopy is a very reliable method when dealing with a



(a) Artificial mixture (e.g. in reality it could be a cup of coffee) composed by caffeine and sucrose.

(b) Raman Spectrometer.

Figure 2.1: a) top: An example of an input signal $\boldsymbol{y}$ that consists of caffeine and sucrose b) bottom: a Raman Spectrometer source.

single element, the task of identifying the elements of a chemical mixture is not that simple. From the perspective of signal processing this corresponds to signals which are sparse in the corresponding domain of representation. Sparse signal processing is appealing when considering real–time applications e.g., handheld spectrometers or mobile phones, since it enables the development of robust methodologies on a limited hardware topology.

In order to provide a deeper insight regarding the Raman spectral decomposition problem and draw the link with the sparse signal processing aspect of it we demonstrate a practical example in Figure 2.1. Within the subfigure (b) we demonstrate a Raman spectrometer, while on the left column we demonstrate 3 different spectras. Within our disposal we have a database of Raman dictionary, where the spectra were collected and provided by [61] where among the available samples is the digital fingerprint of caffeine and sucrose. The corresponding digital fingerprint of each element is demonstrated in the provided figure. We generate a synthetic mixture of these two spectras which is a linear combination of caffeine and sucrose where the associated coefficient is set to $a = 1$ and $b = 1$ accordingly.

From the perspective of a realistic scenario this could be addressed as the digital fingerprint that corresponds to a cup of coffee where caffeine and sucrose contribute equally. The process of spectral decomposition can then be described as follows: The *Raman* spectrometer acquires the digital fingerprint of the coffee. The next task is on one hand the detection of the atoms that contribute in the corresponding sample and on the other hand is to estimate the contribution of each element. For the particular part of the process there is the need for the development of a decomposition algorithm.

From the perspective of sparse signal processing, caffeine and sucrose can be addressed as atoms $\boldsymbol{\phi}_1$,$\boldsymbol{\phi}_2$ that belong to the dictionary $\boldsymbol{\Phi}$. The input sample of caffeine is the signal $\boldsymbol{y}$. In practice the digital fingerprint of $\boldsymbol{\phi}_1$ and $\boldsymbol{\phi}_2$ are available to the system memory of the *Raman* spectrometer. The goal is then the composition of $\boldsymbol{y}$ with respect to the elements which are available in the system memory. In practice, the available samples are expected to consist of only a few elements from the available database. In that sense, the given sample is considered to be sparse with respect to the particular domain of representation.

Note though, that Raman spectrometry is one of the areas where the developed techniques may be applied. Within the current work we are aiming to develop algorithms that align within the general concept of sparse non–negative signal processing and not explicitly with Raman spectrometry. Moreover, we are aiming to use the particular framework as a baseline to draw more general conclusions regarding the characteristics of the data our techniques employ and potentially lead to an improvement in terms of computational efficiency and robustness (e.g., structure, lower dimensional representation etc).

2.5 Problem formulation

The mathematical framework that demonstrates the best performance in terms of signal recovery is the one that solves the problem by means of l_1 norm minimization and the so called Basis Pursuit algorithm which is the alternative to greedy techniques. The main drawback when considering this kind of approach as a potential strategy though, is the fact that they are characterized by a heavy computational workload. The class of convex optimization algorithms that meets the time constraint criterion is the one that solves the least squares problem, as introduced in (2.30). Essentially the original minimization problem introduced in (2.30) is reformulated by adding a constraint that guarantees the non–negativity of the coefficients and takes the form introduced in (2.53).

We focus on the least squares problem and in particular techniques which satisfy both of these criteria. The basis of fast greedy sparse approximation from the OMP perspective relies on the QR factorization. Adopting the concept of QR factorization in the content of Non–Negative Sparse Signal Processing is not straight forward. The main reason is that when we develop an application for a specific type of dictionaries, i.e. as the Raman dictionary, we do not comply with fundamental principles of CS theory such as incoherence and restricted isometry property (RIP). As a result unwanted atoms may be selected while decomposing the signal. Eventually this means that we will end up acquiring negative coefficients in the place that corresponds to the contribution of these atoms. In order to overcome these impracticalities, the authors in [10] introduced the Fast Non–Negative Orthogonal Matching Pursuit (FNNOMP) algorithm that guarantees that acquired coefficients are non–negative. The structure of the FNNOMP algorithm is presented in **Algorithm 7**. From the perspective of the pure greedy MP framework the variant for the Non–Negative framework is presented in **Algorithm 6**. Within the literature that may serve the purpose of sparse non–negative signal processing e.g. Non Negative OMP (NNOMP) and Non–Negative CoSamp (NN CoSaMP). A comparative analysis in between the methods we propose here and the aforementioned algorithms with respect to computational cost and signal reconstruction performance is provided in Chapter 3.

The problem of sparse non–negative signal processing has many similarities with the sparse non–negative matrix factorization problem [64](NNMF). The particular problem can be described as follows: given a matrix $X \in R^N \times D$, learn the matrices $W \in R^{N\times D}$ and $H \in R^{D\times K}$: $\boldsymbol{X} = \boldsymbol{WH}$. For the explicit case where $D = 1$ the dimensions of the sparse non–negative signal processing and sparse NNMF are identical. In that case X can be interpreted as $\boldsymbol{y}$, $\boldsymbol{W}$ as $\boldsymbol{\Phi}$ and $\boldsymbol{H}$ as $\boldsymbol{x}$ from the sparse

Algorithm 6 Non-Negative Matching Pursuit algorithm (NNMP)

```
1: initialisation: $s = \varnothing$, $k = 0$ and $\boldsymbol{r}_0 = \boldsymbol{y}$
2: While $k < K$ & $\max(\boldsymbol{\Phi}^T \boldsymbol{r}_k) > 0$
3: $s_k = \mathbf{0}$
4: $(\zeta, \iota) \leftarrow \max(\boldsymbol{\Phi}^T \boldsymbol{r}_k)$
5: $s_k[\iota] = \zeta$
6: $\boldsymbol{r}_{k+1} \leftarrow \boldsymbol{r}_k - \zeta \boldsymbol{\phi}_\iota$
7: $k \leftarrow k + 1$
8: end
9: $x \leftarrow \sum_k s_k$
```

NNMP to the sparse non–negative signal processing perspective. However, there is a distinct difference from the one framework to the other, i.e. $\boldsymbol{W}$ is learned while Φ is known at the standard sparse non–negative signal processing problem. However, in Chapter 4, we propose a framework where Φ is also learned via training. In that sense, the corresponding methodology can be addressed as a sparse NNMF learned framework for the explicit case where $D = 1$.

$$\begin{aligned} & \underset{x}{\text{minimize}} && ||\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}||_2^2 \\ & \text{subject to} && \boldsymbol{x} \geqslant 0. \\ & && ||\boldsymbol{x}||_0 = k. \end{aligned} \tag{2.53}$$

Greedy techniques in general introduce an appealing framework in terms of computational complexity, nonetheless the associated computational cost heavily depends on the size of the problem. In particular, the corresponding computational workload is carried out by the selection step of the algorithm (step 1 in **Algorithm 7** and step 4 in **Algorithm 6** accordingly). The computational workload of the particular step can be summarised as MN numerical operations. In that sense, the computational complexity is associated with the dimensions M, N of $\boldsymbol{\Phi}$. As a result the computational cost scales linearly across two parameters: 1) the cardinality N of the dictionary, 2) the dimension of the space M that the points live in. Fundamentally, these two parameters reflect the size of the problem.

Generally speaking OMP, MP are fast algorithms. However in several applications where M, N are large then the associated operational workload becomes a bottleneck. The first challenge we are willing to tackle through this work is the reduction of the corresponding computational workload.

$$z_{j+1} \leqslant z^t = \begin{cases} \min\limits_{\gamma_i < 0} \frac{|x_i|}{|\gamma_i|} & \exists i, \gamma_i \leqslant 0 \\ \infty, & \text{otherwise} \end{cases} \tag{2.54}$$

Algorithm 7 FNNOMP

```
Initialization: $s = z_0 = \emptyset, j = 0, \boldsymbol{r}_0 = \boldsymbol{y}$
while $j < K\&$ max$(\boldsymbol{\Phi}^T \boldsymbol{r}_k > 0)$
    1. $[\mu, \iota] \leftarrow \text{sort}_{\downarrow}(\boldsymbol{\Phi}^T \boldsymbol{r}_k)$
    2. $p \leftarrow 1$
    3. $p^c \leftarrow \mu$
    4. $z^c = 0$
    5. while $\sim$ Terminate & $p < N$
    6.     $z_t$ from (2.54).
    7.     z$\leftarrow \boldsymbol{\psi}_\mu^T \boldsymbol{y}$: $\boldsymbol{\psi}_\mu = \frac{\boldsymbol{q}}{||\boldsymbol{q}||_2}$, $\boldsymbol{q} = (\boldsymbol{I} - \boldsymbol{\Psi}\boldsymbol{\Psi}^T)\boldsymbol{\phi}_\mu$
    8.     Update based on Table I
    9. end while
    10. $s = s \cup \mu$
    11. Update $\Psi$ and R$^{-1}$
    12. $z_{j+1} \leftarrow [z_j, z_{j+1}]$
    13. $r_{j+1} = \leftarrow r_j - z_{j+1}\psi_{j+1}$
    14. $j \leftarrow j + 1$
end while
output: $x|_s \leftarrow R^{-1} z_j$
```

if	then
$0 < z \leqslant z^t, z > z^c$	$z_{j+1} \leftarrow z$, Terminate
$0 < z \leqslant z^t, z \leqslant z^c$	$z_{j+1} \leftarrow z^c, p \leftarrow p^c$, Terminate
$z > z^c \geqslant z^t$	$p = p + 1,$
$z \geqslant z^c > z^t$	$z_{j+1} \leftarrow z^c, p \leftarrow p^c$, Terminate
$z > z^t > z^c$	$z^c \leftarrow z^t, p^c \leftarrow p$
$z < 0$	Terminate

Table 2.2: Decision rules that guarantee the positivity of the coefficients

For the explicit case of sparse non–negative signals though, there is a simple but distinct difference in between the standard OMP,MP and the non–negative OMP,MP setting. At the selection step of the algorithm is formed as $\boldsymbol{\Phi}^T \boldsymbol{r}_k$ for the non–negativity setting, or in other words the $|\cdot|$ term is neglected. This is due to the fact that we are seeking explicitly for atoms that are positively correlated with $\boldsymbol{r}_k$. The latter results to the search of the Nearest Neighbour in between the input signal and the associated dictionary Φ given that $||\boldsymbol{\phi}_i|| = 1, \forall \boldsymbol{\phi}_i \in \Phi$. This outcome can be demonstrated as follows:

$$\min_{\boldsymbol{\phi}_i \in \boldsymbol{\Phi}} d(\boldsymbol{y}, \boldsymbol{\phi}_i) = \min_{\boldsymbol{\phi}_i \in \boldsymbol{\Phi}} (d(\boldsymbol{y}, \boldsymbol{\phi}_i))^2 = \min_{\boldsymbol{\phi}_i \in \boldsymbol{\Phi}} ||\boldsymbol{\phi}_i||^2 - 2\boldsymbol{\phi}_i^T \boldsymbol{y} + ||\boldsymbol{y}||^2$$
$$\overset{||\boldsymbol{\phi}_i||=1, \forall \boldsymbol{\phi}_i \in \boldsymbol{\Phi}}{=} \min_{\boldsymbol{\phi}_i \in \boldsymbol{\Phi}} - 2\boldsymbol{\phi}_i^T \boldsymbol{y} + ||\boldsymbol{y}||^2 = 2 \cdot \min_{\boldsymbol{\phi}_i \in \boldsymbol{\Phi}} \cdot (-\boldsymbol{\phi}_i^T \cdot \boldsymbol{y}) = \max_{\boldsymbol{\phi}_i \in \boldsymbol{\Phi}} \boldsymbol{\phi}_i^T \boldsymbol{y} \tag{2.55}$$

The Nearest Neighbor Search (NNS) problem is a well established framework within the Computer Science community. The most classical approaches are the ones that accelerate the NN search via methods that arise from the traditional Machine Learning regime and in particular clustering. In practice, these techniques are aiming to reduce the size of the problem in terms of N. An alternative approach to clustering is the one that embeds the dictionary in a lower dimensional space R^K. From the perspective of the problem size, this sort of approach relaxes the problem in terms of M. However, this type of approach yields an approximate solution to NNS. Within the current work, we are aiming to exploit benefits from the research conducted by the particular community, and incorporate methodologies that will introduce an acceleration for greedy techniques in the non–negativity setting.

The second goal of the current work is to improve the performance of greedy techniques in the non–negativity setting in terms of signal recovery. A critical parameter to the that direction is the acquisition of the ground truth support set s. This essentially means that the selection step of the associated algorithms play a critical role in the overall performance of greedy techniques. The conventional approach followed by the current framework is utilizing the known dictionary $\boldsymbol{\Phi}$ at the selection step of the algorithm. This results to a framework that the support recovery is highly dependent on the underlying structure of $\boldsymbol{\Phi}$. In cases for example, where the points in $\boldsymbol{\Phi}$ are highly correlated with each other then most likely a false positive may be introduced in s. For example, going back to the case of the synthetic mixture demonstrated before. In the ideal case, the OMP,MP framework will detect coffee, sucrose within the first two iterations of the algorithm. Nevertheless, there may exist elements in the dictionary that have a very similar digital fingerprint with either of these components. In such case, the conventional OMP,MP framework may introduce a false positive in s, i.e. brown sugar instead of sucrose. This mistake can be critical in several applications. For example let us address the false positive from an application related to hazardous materials detection. In that scenario a false positive means that the selection step detected a non explosive material and therefore the given sample does not correspond to a dangerous substance while in practice this is not the case and we are dealing with an explosive ingredient. In such case, the outcome of the false positive could be fatal.

The acquisition of s is quite relevant with the so–called multilabel classification problem in the deep learning regime. However, one of the common characteristics of the DL models is that in many cases they are overparameterised. From the perspective of the OMP,MP framework, overparameterization may result to significant increase in the computational complexity of the selection step. Eventually this may result to an increase in the computational workload compared to the standard OMP,MP framework. Within the current work, we are aiming to incorporate methodologies from the DL regime that will introduce a boost in the performance in the OMP,MP framework without altering its computationally appealing structure. By doing that we are aiming to take advantages of DL, i.e. better parameterize the model weights.

2.6 Data aware methodologies

Within the current section we review data aware methodologies that may serve the main goals of the current work which are stated as follows: 1) acceleration and 2) signal recovery improvement for greedy techniques. The order of presentation will take place with respect to the order of the goals of the current project stated before.

2.6.1 Exact Nearest Neighbor search via Data structures

The nearest neighbour search is a well established problem that finds applications in several areas such as pattern recognition [65], data mining [66], data compression [67] etc. But many other tasks also require nearest neighbor queries, i.e. localised support vector machines [68], reinforcement learning [69] etc. The standard setting for (NNS) can be described as follows:

Given a space of points R^M, a dictionary $\mathbf{\Phi} \subset R^M$ and a distance function: $d : R^M \times R^M \rightarrow R$, the nearest neighbor (NN) of $\boldsymbol{y}$ in $\mathbf{\Phi}$ is defined as follows:

$$\boldsymbol{\phi}_{NN} = \arg\min_{\boldsymbol{\phi}_i \in \mathbf{\Phi}} d(\boldsymbol{\phi}_i, \boldsymbol{y}) \tag{2.56}$$

The naive approach to query the NN of $\boldsymbol{y}$ in $\mathbf{\Phi}$ is the one that carries out a linear scan over all the data points in $\mathbf{\Phi}$. Given a dictionary $\mathbf{\Phi} \subset R^M$ with $|\mathbf{\Phi}| = N$, the overall computational cost required to carry out the assigned task can be summarised as: MN numerical operations.

In order to bypass the computational workload associated with the naive–brute

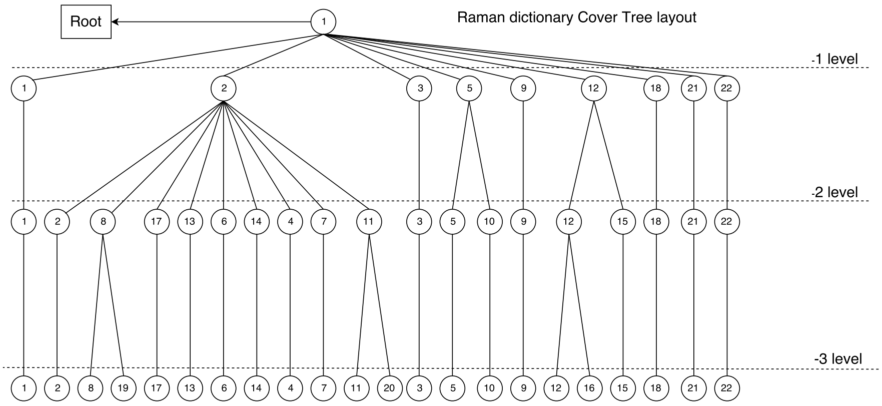


Figure 2.2: The figure demonstrates the layout of a Cover Tree for a *Raman* dictionary of 22 atoms. The layout is constructed with respect to the nesting,covering and separation criteria of the method.

force approach data structures have been proposed with an aim to accelerate the process. The main goal of these approaches is to accelerate the process by fingerprinting the underlying structure that may be present in $\mathbf{\Phi}$. One of the most popular, and probably the oldest method to address such task, is the so called K–d trees. Originally the particular approach for data living in relatively low dimensions (i.e. $K = 3$) but an extension to higher dimensions is straight forward. Generally speaking, K–d trees can be addressed as a sort of binary tree structure where data in each node is a M –dimensional point.

The most recent advances in the area of data structures is the introduction of the so called *Cover Tree* [70] which has drawn a lot of attention in the last few years. We will here discuss more in depth this sort of approach in order to provide a deeper insight about the functionality of these structures overall.

The Cover Tree data structure. A *cover tree* T on a dataset S is a leveled tree where each level is a cover for the level below it. Each level is assigned with an index of integer scale i which decreases while descending over T. Let C_i represent the set of nodes at the i–th level of T. A cover tree T on a dataset S obeys the following invariants for all i:

- (nesting) $C_i \subset C_{i-1}$.
- (covering) For every $\boldsymbol{p} \in C_{i-1}$, there exists a $\boldsymbol{q} \in C_i$ satisfying $d(\boldsymbol{p}, \boldsymbol{q}) \leqslant 2^i$, and exactly one such $\boldsymbol{q}$ is a parent of $\boldsymbol{p}$.
- (separation) For all $\boldsymbol{p}, \boldsymbol{q} \in C_i, d(\boldsymbol{p}, \boldsymbol{q}) > 2^i$,

where $\boldsymbol{p}, \boldsymbol{q} \in \mathbf{\Phi}$.

The invariants upon T are constructed which in practice are the same as the ones introduced in the navigating nets [71]. The only difference is related with invariant 2 where in the concept of cover trees only one parent is required instead of all possible nodes which is the common practice in the navigating nets setting.

In order to familiarise the reader with these structures and make the transition to the rest of the associated analysis smoother, we construct a T with respect to the associated cover tree invariants described above and dictionary of Raman spectra with $N = 22$ and $M = 1750$. The corresponding T layout is demonstrated in Figure 2.2. Note that from the perspective of our problem, the data points are stacked column wise in $\mathbf{\Phi}$ hence the associated index of each node (*i.e.* 9) stands for the corresponding column in $\mathbf{\Phi}$.

In order to understand the functionality of these methods and draw the link with the sparse non–negative problem we can sketch an intuitive example. In particular we are considering an input signal $\boldsymbol{y} : \boldsymbol{y} = a_4\boldsymbol{\phi}_4 + a_{10}\boldsymbol{\phi}_{10}$, where $a_4, a_{10} \geqslant 0$. Then within the first iteration of the decomposition algorithm $\boldsymbol{y}$ will acquire the NN, i.e. $\boldsymbol{\phi}_{10}$ via the search algorithm introduced in **Algorithm 8**. This algorithm can be plugged in at the selection step of the decomposition algorithm e.g. FNNOMP.

In the ideal scenario, while descending down T, the algorithm will pass from node 1 in level 0 of T, in level -1 will make the distance computations with points living in this particular level and then it will descend over node 8. At level -2 it will perform the computations only with the children of node 8 and then it will descend over node 11, the node that results to the minimum Euclidean distance, down to the bottom of T. So, in total we will conduct an overall number of pairwise distance measurements which can be summarised as follows: a single distance measurement with node 1 at level 0. Eight distance measurements at level -1 with nodes: $2, 3, 5, 9, 12, 18, 21, 22$. Eight distance measurements at level -2 with nodes: $8, 17, 18, 8, 14, 4, 7, 11$. One distance measurement with node 10 at level -2. Overall, the search will take place with respect to an overall number of $\Xi = 18 < N = 22$ distance measurements. The latter can be materialised in terms of an acceleration metric as follows:

$$\text{Acceleration} = \frac{MN \text{ numerical operations}}{M\Xi \text{ numerical operations}} = \frac{N}{\Xi} = \frac{22}{18} \approx 1.2 \tag{2.57}$$

So in practice the brute force search is 1.2 times slower or seeing it the other way around, NN search via the *Cover Tree* is 1.2 times faster than brute force search. The metric introduced in (2.57) is more formally known as the d–metric and it was first introduced in [72]. This type of metric is used in [70] from the authors to materialise the acceleration that the corresponding structure introduces compared to the brute

Algorithm 8 NN search

```
Set $Q_\infty$ where $C_\infty$ is the root level $T$.
for $i$ from $T$ down to $-\infty$(depth of the tree).
    • Set $Q = \{\text{Children(q)} : q \in Q_i\}$
    • Form cover set $Q_{i-1} = \{q \in Q : d(p,q) \leqslant d(p,Q) + 2^i\}$.
Return arg $\min_{q \in Q_{-\infty}} d(p,q)$
```

force search.

From the perspective of the sparse non–negative problem we address the dictionary $\boldsymbol{\Phi}$ as 2D entity where the 1st dimension M corresponds to the number of rows of Φ while the 2nd dimension corresponds to the number of columns. Fundamentally, these type of structures reduce the dimensions of the problem in terms of N (i.e. we do not perform all the pairwise comparisons but with a subset $\boldsymbol{\Xi} \subset \boldsymbol{\Phi} : |\boldsymbol{\Xi}| \leqslant N$). So overall, data structures can be addressed with a dimensionality reduction technique from the perspective of the $2D$ entity.

Nonetheless reduction in distance measurements implied by (2.57) is an indirect measure of acceleration. Reduction in the number of distance computations informs us what improvement in efficiency could be if bookkeeping and memory access costs were greatly reduced. Due to the curse of dimensionality, it is unlikely that there exists a general efficient solution to the exact k–NN problem [73]. A more comprehensive study of data structures with respect to the real time domain in various datasets and dimensions can be found in [74]. Within the current work we are interested in acceleration in terms of the real time domain. The acquired results demonstrate that tree structures are in several cases faster than the brute force search on the real–time domain in a relatively low dimension (i.e. $M < 40$).

2.6.2 Approximate Nearest Neigbhor Search: Linear Embeddings

Due to the limitations introduced by the *curse of dimensionality*, the research followed an alternate direction where instead of looking for the exact NN the associated methodologies result in an approximate solution. This type of methodology is the one that incorporates a linear embedding in the overall NN procedure. Addressing the overall functionality of the particular approach from the perspective of an online algorithm, the dictionary $\boldsymbol{\Phi}$ is embedded from the high dimensional space R^N to a lower dimensional space R^K via a linear operator $f : R^N \rightarrow R^K$ (We denote the representation of $\boldsymbol{\Phi}$ in the lower dimensional space as $\hat{\boldsymbol{\Phi}}$). This part of the process takes place offline. On the other hand the query point $\boldsymbol{y}$, is embedded in the lower dimensional space in an online manner. The motivation that lies upon this particular approach, is that the brute force search performed in the lower dimensional space, i.e. KN operations, is computationally cheaper and therefore more appealing in the real time setting.

From the perspective of the 2D entity $\mathbf{\Phi}$ this particular approach relaxes the problem in terms of M. In order to provide an intuitive overview of the problem, we demonstrate the size of the problem in the original problem in its 2D matrix form, where the N spectra are stacked over the columns of $\mathbf{\Phi}$ while each $\boldsymbol{\phi}_i$ consists of M coefficients. The representation $\hat{\mathbf{\Phi}}$ in the lower dimensional space, is a compressed version of the original dictionary. The number of elements N remains the same while the coefficients for each atom are of a number K where $K < M$. By following this approach, we may reduce the computational cost but on the other hand we alter the underlying structure of the dictionary. By that we mean, that the pairwise distances from R^N to R^K are distorted. From the perspective of NN, this will lead to situations where $NN_M \neq NN_K$ (where NN stands for the nearest neighbor and M, K the size of the Euclidean space).

In order to provide a better understanding regarding this issue we demonstrate an intuitive example in Figure 2.3. In particular we are considering a dictionary $\mathbf{\Phi} = \{\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, \boldsymbol{\phi}_3, \boldsymbol{\phi}_4, \boldsymbol{\phi}_5, \boldsymbol{\phi}_\mu\}$. Assuming that we are given an input signal $\boldsymbol{y}$, then $NN_M = \mu$, where μ holds for the index of the point in the matrix $\mathbf{\Phi}$. At the embedded space though, the relative positions between the dictionary points and the representation $\hat{\boldsymbol{y}}$ of the input signal in R^K change. As a result $NN_K = 5$ in the embedded space. Hence the particular approach yields an approximate solution to the NN problem.

Addressing the particular outcome from the perspective of an online algorithm e.g., FNNOMP, the particular approach could be incorporated as the selection step of the algorithm. The standard structure of the algorithm introduces an empirical error on the sparse non–negative signal processing problem. By incorporating an approximation framework within the structure of FNNOMP this essentially means that we will end up approximating the empirical error introduced by the standard form of the algorithm.

Given that this approach is appealing due to its computational simplicity, there are two main directions that can be followed when considering replacing the selection step of FNNOMP with such an approach: 1) introduce an upper bound at the approximation error that the particular approach may introduce to the empirical error of FNNOMP 2) incorporate a rule that compensates the error introduced at the embedded space and eventually results in an exact solution to the NN problem.

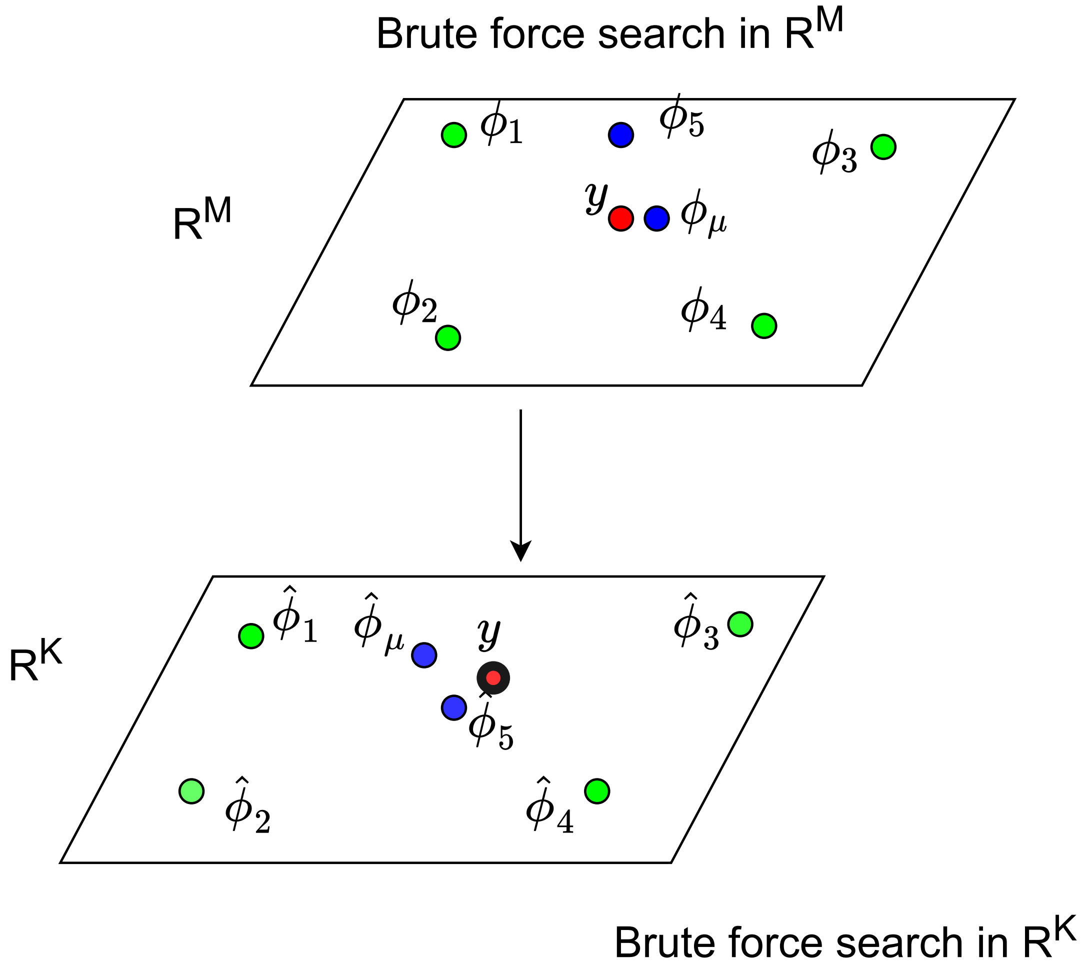


Figure 2.3: Example of a mismatch from R^M to R^K

2.6.3 Random Projections

One of the most classical approaches in the linear embeddings setting is Random Projections (RP). This type of methodology is simple in terms of construction and implementation. Note also, that unlike the other linear operators RP is a data oblivious operator. The construction of associated operator does not take into account any sort of information that can be exploited from a known dictionary, i.e. $\boldsymbol{\Phi}$. Typically the embedding operator $\boldsymbol{Q} \in R^{K \times N}$ takes the form:

$$\boldsymbol{Q} = \frac{1}{\sqrt{K}}\boldsymbol{A}, \tag{2.58}$$

where the entries in $\boldsymbol{A}$ are drawn from a Gaussian distribution $N(0,1)$.

Random projections are a key ingredient of the *locality sensitivity hashing* framework that finds applications in data hiding and security applications [75],[76],[77]. From the perspective of the approximate NN framework random projections are used in cases such as Spotify, a digital music dictionary where this method is used to find the approximate NN music recommendations as part of their open source system [78].

From a theoretical point of view, *random projections* relies on the so–called Johnson–Lindenstrauss lemma was first introduced in 1984, which is as follows [79]:

Johnson–Lindenstrauss Lemma. Let $\epsilon \in (0, \frac{1}{2})$. Let $\Phi \subset R^N$ be a set of N points and $K = \frac{20logN}{\epsilon^2}$. There exists a *Lipscitz* mapping: $f : R^N \rightarrow R^K$ such thing that $\forall \phi_i, \phi_j \in \Phi$:

$$(1-\epsilon)||\boldsymbol{\phi}_i - \boldsymbol{\phi}_j||^2 \leqslant ||f(\boldsymbol{\phi}_i) - f(\boldsymbol{\phi}_j)|| \leqslant (1+\epsilon)||\boldsymbol{\phi}_i - \boldsymbol{\phi}_j||^2$$

We are also interested to know what does this bound mean in practice. For starters we will take the upper bound for $\epsilon = \frac{1}{2}$, considering the point cloud Φ of raman data with $M = 1507$, $N = 4041$, which results in the lowest bound in terms of K. By plugging this value into the quantity $K = \frac{20logN}{\epsilon^2} = 664$. What is gonna happen if we want to have a smaller ϵ though? For example for $\epsilon = 0.1$? Then $K \approx 166000 > N$. So, in order to achieve a low distortion in the pairwise distances we end up embedding in a space of much higher dimension than the original space.

2.6.4 Principal Component Analysis (PCA)

Principal component analysis is an unpsupervised learning method that arise from the area of tML and is one of the most common, and most likely the oldest, dimensionality reduction techniques. Before moving forward and make a thorough introduction on PCA, we will first try to answer the following question: PCA or Singular Value Decomposition (SVD)?

Given a dictionary $\boldsymbol{\Phi} \in R^{M \times N}$, this can be decomposed by means of its singular values as follows:

$$\boldsymbol{\Phi} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T, \tag{2.59}$$

where $\boldsymbol{U} \in R^{N \times N}$ an orthonormal matrix that consists of the associated eigenvectors: $\boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{I}_M = \boldsymbol{U}\boldsymbol{U}^T$. $\boldsymbol{S} \in R^{M \times N}$ a diagonal matrix where its consists of the associated eigenvalues at the main diagonal, and $\boldsymbol{V} \in R^{N \times N}$ an orthonormal matrix:

$$\boldsymbol{V}^T\boldsymbol{V} = \boldsymbol{I}_N = \boldsymbol{V} \cdot \boldsymbol{V}^T \tag{2.60}$$

In order to apply PCA, we apply SVD on top of the matrix $\Phi\Phi^T$. The latter can be rewritten as follows:

$$\mathbf{\Phi\Phi}^T \overset{(2.59)}{=} \boldsymbol{USV}^T\boldsymbol{VS}^T\boldsymbol{U}^T \overset{(2.60)}{=} \boldsymbol{U\Lambda U}^T, \tag{2.61}$$

where Λ in practice contains the squares of the diagonal entries in $\boldsymbol{S}$. In order to incorporate PCA in the concept the embedding operator $\boldsymbol{Q} \in R^{N\times K}$ is a subset of $\boldsymbol{U}$. As it can be seen from (2.61) and (2.59) though the orthogonal matrix $\boldsymbol{U}$ is identical in between these techniques. So whether you apply PCA or SVD in $\mathbf{\Phi}$, the outcome in terms of $\boldsymbol{U}$ is going to be the same.

In contrast to *random projections*, PCA is a data aware deterministic technique, i.e. the embedding operator $\boldsymbol{Q}$ is constructed with respect to the given dictionary $\mathbf{\Phi}$. It is appealing due to its simplicity implementation wise and finds applications in several areas i.e. finance [80], image compression [81]. A key characteristic from the perspective of the NN problem, is that the corresponding operator may distort the pairwise distances arbitrarily and the embedding is not considered to be isometric. Unlike *random projections*, it is not aligned with some sort of theoretical foundation such as the *Johnsson Lindenstrauss* lemma.

2.6.5 *NuMax*: Nuclear norm minimization with Max–norm constraints

Within the concept of data aware embedding operators, one of the most recent advances is associated with the so–called *NuMax* algorithm [82]. From a general point of view, the main idea that *NuMax* relies on, is the construction of an embedding operator $\boldsymbol{Q}$ that is nearly isometric as *random projections* but in the same time data aware like PCA. A key characteristic of this operator is that the associated operator may distort the pairwise distances up to an ϵ. This comes in contrast with PCA where the distances are distorted arbitrarily, while the oblivious nature of *random projections* cannot leverage any special geometric structure that may characterise natural data [82].

In contrast with PCA, this data aware operator does not construct with respect to $\mathbf{\Phi}$ directly. The embedding operator is constructed with respect to the so called secant set $S(\mathbf{\Phi})$, where:

$$S(\mathbf{\Phi}) = \{\frac{\boldsymbol{\phi}_i - \boldsymbol{\phi}_j}{||\boldsymbol{\phi}_i - \boldsymbol{\phi}_j||_2}, \boldsymbol{\phi}_i, \boldsymbol{\phi}_j \forall \boldsymbol{\phi}_i, \boldsymbol{\phi}_j \in \mathbf{\Phi}, i \neq j\} \tag{2.62}$$

The problem of acquiring the embedding operator $\boldsymbol{Q}$ that introduces a minimal distortion δ in $S(\mathbf{\Phi})$ is cast and solved as an optimization problem over the space of Positive Semidefinite Matrices. $\boldsymbol{P}$ is defined as: $\boldsymbol{P} = \boldsymbol{Q}^T\boldsymbol{Q} \in R^{M\times M}$. Then rank($\boldsymbol{P}$)= K. There are also the constraints such that $||| \boldsymbol{Q}\boldsymbol{v}_i||_2^2 - 1| = |\boldsymbol{v}_i^T\boldsymbol{P}\boldsymbol{v}_i - 1|$ is greater than δ $\forall \boldsymbol{v}_i \in S(\mathbf{\Phi})$. Let $\mathbf{1}_S$ denote the S–dimensional all ones vector, and let $\mathcal{A}$ denote the linear operator that maps a symmetric matrix $\boldsymbol{X}$ to the S–dimensional vector $\mathcal{A} : \boldsymbol{X} \rightarrow (\boldsymbol{v}_i^T\boldsymbol{X}\boldsymbol{v}_i)_{i=1}^S$. Then we obtain the linear operator $\boldsymbol{Q}$ that minimizes the optimization problem introduced in (2.63):

$$\begin{aligned} &\underset{P^T=\boldsymbol{P}\succeq\mathbf{0}}{\text{minimize}} && \text{rank}(\boldsymbol{P}) \\ &\text{subject to} && ||\mathcal{A}(\boldsymbol{P}) - \mathbf{1}_S|| \leqslant \delta \end{aligned} \tag{2.63}$$

Rank minimization is typically an non–convex and NP–hard. Therefore the authors motivated by [83] introduced a nuclear norm relaxation in (2.63):

$$\begin{aligned} &\underset{P^T=\boldsymbol{P}\succeq\mathbf{0}}{\text{minimize}} && ||\boldsymbol{P}||_* \\ &\text{subject to} && ||\mathcal{A}(\mathbf{P}) - \mathbf{1}_S|| \leqslant \delta \end{aligned} \tag{2.64}$$

In practice the *NuMax* problem introduced in (2.64) is solved via an algorithm that is based on the Alternating Direction Method of Multipliers (ADMM). The user needs to provide $\mathbf{\Phi}$ and δ to the framework. In practice *NuMax* constructs a linear operator $\boldsymbol{Q}$ that distorts all the elements in $S(\mathbf{\Phi})$ up to δ. The latter can be demonstrated as follows:

$$(1-\delta)||\frac{\boldsymbol{\phi}_i - \boldsymbol{\phi}_j}{||\boldsymbol{\phi}_i - \boldsymbol{\phi}_j||}||^2 \leqslant ||\boldsymbol{Q}\frac{\boldsymbol{\phi}_i}{||\boldsymbol{\phi}_i - \boldsymbol{\phi}_j||} - \boldsymbol{Q}\frac{\boldsymbol{\phi}_j}{||\boldsymbol{\phi}_i - \boldsymbol{\phi}_j||}|| \leqslant (1+\delta)||\frac{\boldsymbol{\phi}_i - \boldsymbol{\phi}_j}{||\boldsymbol{\phi}_i - \boldsymbol{\phi}_j||}||^2 \tag{2.65}$$

From the perspective of CS, the *NuMax* problem yields an operator $\boldsymbol{Q} \in R^{M\times N}$ that satisfies the RIP on $S(\mathbf{\Phi})$ with the isometry constant being equal to δ. Note though, that similarly to PCA the particular data aware and deterministic framework does not align with any sort of theoretical justification.

2.6.6 Sparse Signal Reconstruction: a Learning Perspective

Sparse coding has drawn a lot of attention within the deep learning community in the last few years. This is due to the potentiality of extracting features from raw data, especially in cases when the the basis vectors of the dictionary are learned from unlabeled data. There exist several approaches that have been proposed to learn such dictionary. Numerous applications of these approaches have been introduced in areas

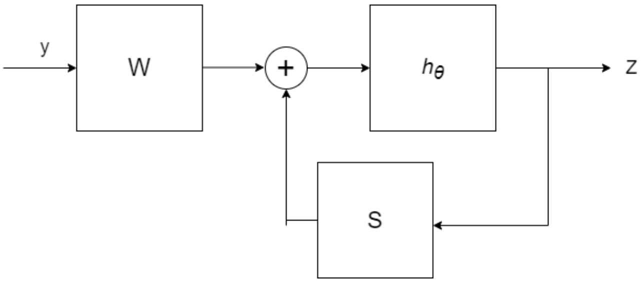


Figure 2.4: ISTA block representation, $\boldsymbol{W}$, $\boldsymbol{S}$ the associated matrices and h_θ the non–linear activation function.

such as visual neuroscience [28][84], image restoration [85],[86], while they have been also used for the extraction of features for object recognition [87],[88],[89].

A major issue with unsupervised learning methods for sparse coding is that the associated algorithm is computationally expensive that makes its implementation prohibitive from a perspective of a real time application. The authors in [90] introduced the Learned Iterative Soft Thresholding Algorithm (LISTA), a scheme that computes an approximation of the sparse code in a fixed time frame. Under the assumption that the atoms of the sparse coder have been trained and they are fixed, the main idea this method relies on is the training of a non–linear encoder that predicts the optimal sparse code after training the network with examples of input sparse vectors paired with the associated coefficients. After training, the network introduces a framework with a prefixed computational complexity and can be used for the approximation of sparse codes with a predescribed expected error.

The core idea upon this method relies on, is the unfolding of a conventional sparse coding algorithm such as ISTA by means of a Deep Neural Network (DNN). Within Figure 2.4 we represent a single iteration of ISTA as a system of blocks. At each iteration of that scheme, the corresponding information passes from the blocks $\boldsymbol{W} \in R^{M\times N}$ and $\boldsymbol{S} \in R^{N\times N}$. Then the the vector that comes as an outcome of the addition of the outputs from these blocks is passed through the non–linear activation function h_θ, where h_θ is defined as in (2.28), in order to get the desired sparse code. The process is repeated over the k iterations of the algorithm.

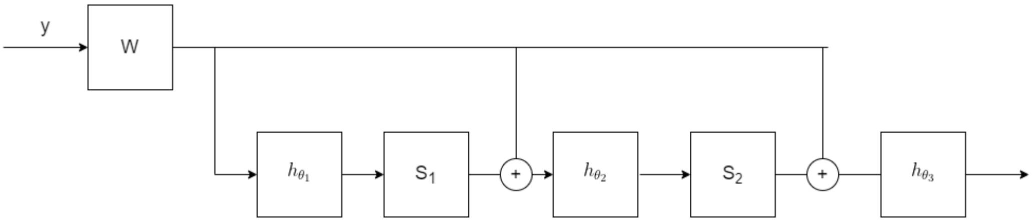


Figure 2.5: LISTA block representation

	LISTA	ISTA
Decomposition parameters	$MN + kN^2 + k$	$MN + N^2 + 1$
Numerical Operations for k iterations	$MN + kN^2 + kN$	$MN + kN^2 + kN$

Table 2.3: Comparison in between computational cost and decomposition parameters in between ISTA,LISTA

By reformulating ISTA by means of deep neural network, as in Figure 2.5, we employ benefits from the fact that for a size of k iterations–layers of the network we operate on a framework with a fixed computational cost. Note that in practice, each iteration of ISTA takes the form of a block in the LISTA scheme where each block consists of layers $\boldsymbol{S}_k$, $\boldsymbol{W}$ and $\boldsymbol{h}_{\theta_k}$. On the other hand, the deep learning framework releases the degrees of flexibility of the decomposition process given that a higher number of parameters is incorporated in the overall procedure. In particular, instead of utilizing a single set of parameters $\boldsymbol{W}$, $\boldsymbol{S}$, $\boldsymbol{h}_\theta$ over the k iterations a different set of parameters S_k and θ_k is utilised over the blocks of the DNN. A more analytical representation about the computational complexity and the decomposition parameters utilized by each framework are demonstrated in Table 2.3. The latter, introduces an improved performance of LISTA over ISTA with respect to the empirical error introduced in (2.66).

$$\epsilon = ||\boldsymbol{x} - \boldsymbol{x}^\star||, \tag{2.66}$$

where $\boldsymbol{x}$ the ground truth sparse code and $\boldsymbol{x}^\star$ the approximated sparse code by the corresponding algorithm.

From a theoretical point of view, the key contribution in the LISTA framework was introduced by the authors in [91]. In particular, for $B > 0$ and $s < \frac{(1+\frac{1}{\hat{\mu}(W)})}{2}$ then the error converges as follows:

$$||\boldsymbol{x} - \boldsymbol{x}^\star|| \leqslant sB\text{exp}(-cl), \tag{2.67}$$

where l the number of layers in LISTA and $\hat{\mu}(W)$ is defined as follows:

$$\hat{\boldsymbol{\mu}}(W) = \max \frac{\boldsymbol{W}(:,i)^T \boldsymbol{\Phi}(:,j)}{||\boldsymbol{W}(:,i)||_2|\boldsymbol{\Phi}(:,j)||_2}, \text{with } i \neq j, \tag{2.68}$$

where $\boldsymbol{\Phi}$ the standard dictionary and $\boldsymbol{W}$ the set of weights learned from LISTA. Fundamentally, the bound holds for signals with a unique representation with respect to $\hat{\mu}(\boldsymbol{W})$ and s. In that case the signal will be exactly recovered when $l \to \infty$. The latter implies that for this explicit class of signals that have a unique representation an architecture with an infinite number of layers is required to fully reconstruct the signal. This results to a scheme which is practically intractable in terms of computational workload and hardware resources.

The interested reader will observe that there is a strong similarity into the unique representation of a signal in between LISTA and OMP. In particular for LISTA, this is materialized as $s < \frac{(1+\frac{1}{\hat{\mu}(\boldsymbol{W})})}{2}$ while in OMP is it defined as $s < \frac{(1+\frac{1}{\mu(\boldsymbol{\Phi})})}{2}$, where $\mu(\boldsymbol{\Phi})$ as in (2.38). In order to provide an interpretation of these bounds we can sketch the following example: Let us assume that we are given a dictionary with $\mu(\boldsymbol{\Phi}) = 0.2$. The bound of OMP implies that all the signals with sparsity up to $|\mathrm{s}| = 3$ can be exactly recovered within a finite number of $|\mathrm{s}| = 3$ iterations. Let us assume that LISTA after training has $\hat{\mu}(\boldsymbol{W}) = 0.2$. Then according to the associated bound, the framework can recover the signals with a sparsity up to $|\mathrm{s}| = 3$ but for an infinite number of layers–computations.

A learned variant of various greedy techniques (e.g., OMP, MP, etc) was proposed by the authors in [92]. Motivated by LISTA the authors of the proposed framework unfold the aforementioned greedy techniques by means of a neural network. In practice this approach has many similarities with the dictionary learning regime. This is due to the fact that over the different layers K of the network a single dictionary $\boldsymbol{D}$ is shared , the so–called regular dictionary which is learned via training. $\boldsymbol{D}$ is also shared in between the selection step and the update step of the framework. Essentially the motivation of the particular work is to learn a single dictionary $\boldsymbol{D}$ that demonstrates a better performance in terms of signal decomposition compared to the standard dictionary $\boldsymbol{\Phi}$. This type of approach is relatively different from the commonly followed approach in LISTA where the weights are not shared and therefore a higher number of parameters is incorporated at the decomposition process. Note also, that in contrast with LISTA the particular approach lacks any sort of theoretical justification.

2.6.7 Dictionary Preconditioning for Greedy Algorithms

From the perspective of the conventional OMP,MP framework a significant amount of work has been conducted aiming to improve the signal recovery of the associated algorithms. A key characteristic of the conducted analysis, is that the overall functionality of the OMP,MP framework is distinguished into two basic steps: 1) sensing step 2) reconstruction step.

The authors in [29] introduced the so–called preconditioning framework. The fundamental idea upon the particular framework relies on is the replacement of the sensing step of the conventional MP,OMP framework with a different approach. By doing that, the preconditioning framework is essentially aiming to develop a selection rule that will boost the performance in terms of the support recovery s that will eventually lead to the boost in the performance in terms of $\boldsymbol{x}_s$ recovery. Nevertheless, a key goal of this approach though is to fulfill such task without increasing the computational complexity of the original selection step which can be summarised as MN operations.

From a more practical point of view, the preconditioning framework replaces the dictionary $\boldsymbol{\Phi} \in R^{M\times N}$ with a sensing dictionary $\boldsymbol{\Psi} \in R^{M\times N}$. Essentially, each atom $\boldsymbol{\phi}_i \in \Phi$, with $\boldsymbol{\phi}_i \in R^M$, is assigned to an alternative representation $\boldsymbol{\psi}_i \in R^M$. The main goal of this approach is the design of a sensing matrix $\boldsymbol{\Psi}$ where the so–called cross cumulative coherence is significantly lower than the cumulative coherence in $\boldsymbol{\Phi}$. This sort of improvement eventually reduces the number of false positives introduced in s.

Theorem 1. Let $\boldsymbol{y}$ be a signal exactly k–sparse in $\boldsymbol{\Phi}$, i.e. $\boldsymbol{y} = \sum_{i\in I} \boldsymbol{x}_i\boldsymbol{\phi}_i$. OMP,MP using the sensing matrix $\boldsymbol{\Psi}$, will always select components of the true support I if:

$$||(\boldsymbol{\Phi}_I^T\boldsymbol{\Psi}_I)^{-1}\boldsymbol{\Phi}_I^T\boldsymbol{\Psi}_{\bar{I}})||_{1,1} < 1, \tag{2.69}$$

which is always satisfied if:

$$\bar{\mu}_{\boldsymbol{\psi}}(k) + \bar{\mu}_{\boldsymbol{\psi}}(K-1) < \beta_{\boldsymbol{\psi}} \tag{2.70}$$

$$\bar{\mu}_{\boldsymbol{\psi}}(k, \boldsymbol{\Phi}, \boldsymbol{\Psi}) = \max_i \max_{|I|=k, i\notin I} \sum_{j\in I} |\langle \boldsymbol{\psi}_i, \boldsymbol{\phi}_j\rangle| \tag{2.71}$$

$$\beta_{\boldsymbol{\psi}}(\boldsymbol{\Phi}, \boldsymbol{\Psi}) = \min_i |\langle \boldsymbol{\psi}_i, \boldsymbol{\phi}_i\rangle| \tag{2.72}$$

Note from the following, the fact that this particular framework selects k ground truth atoms does not necessarily mean that $|s| = k$ (i.e. a ground truth atom may be selected twice). This is due to the fact that the residual is not orthogonal to the previously selected ψ_i's: $\boldsymbol{r}_k \not\perp \boldsymbol{\psi}_i : i \in s => \langle \boldsymbol{r}_k, \boldsymbol{\psi}_i\rangle \neq 0$, therefore ψ_i may be reselected. This comes in contrast with the conventional OMP framework where the residual vector r_k is always orthogonal to the previously selected atoms: $\boldsymbol{r}_k \perp \boldsymbol{\phi}_i : i \in s => \langle \boldsymbol{r}_k, \boldsymbol{\psi}_i\rangle = 0$, therefore $|s| = k$ after k iterations.

In order to demonstrate this, we can think of the following example. 1) OMP with $\boldsymbol{\Phi}$ as the sensing matrix OMP$_{\boldsymbol{\Phi}}$ 2) OMP with $\boldsymbol{\Psi}$ as a sensing matrix OMP$_{\boldsymbol{\Psi}}$. Let us assume that within the first iteration the first element $\boldsymbol{\phi}_1$ for OMP$_{\boldsymbol{\Phi}}$ and $\boldsymbol{\psi}_1$ is the one with the highest contribution in the input signal $\boldsymbol{y}$ for OMP$_{\boldsymbol{\Psi}}$. Then considering the implementation of OMP with respect to the QR factorization the intermediate steps are formulated as follows:

1. $\boldsymbol{Q}_1 = \boldsymbol{\phi}_1$ and $\boldsymbol{z}_1 = \boldsymbol{\phi}_1^T\boldsymbol{y}$

2. $\boldsymbol{z}_1 = \boldsymbol{\phi}_1^T\boldsymbol{y}$

3. $\boldsymbol{r}_1 = \boldsymbol{y} - \boldsymbol{Q}_1\boldsymbol{z}_1 = \boldsymbol{y} - \boldsymbol{\phi}_1\boldsymbol{z}_1$

Within the next iteration we have the following relationship for OMP$_{\boldsymbol{\Phi}}$: $\boldsymbol{\phi}_1^T\boldsymbol{r}_1 = \boldsymbol{\phi}_1^T(\boldsymbol{y} - \boldsymbol{\phi}_1 z_1) = \boldsymbol{\phi}_1^T\boldsymbol{y} - \boldsymbol{\phi}_1^T\boldsymbol{\phi}_1 z_1 \overset{\boldsymbol{\phi}_1^T\boldsymbol{\phi}_1=1}{=} \boldsymbol{\phi}_1^T\boldsymbol{y} - z_1 \overset{\text{from } 2}{=} z_1 - z_1 = 0$. Hence ϕ_1 will not be reselected. Within the next iteration we have the following relationship for OMP$_{\boldsymbol{\Psi}}$: $\boldsymbol{\psi}_1^T r_1 = \boldsymbol{\psi}_1^T(\boldsymbol{y} - \boldsymbol{\phi}_1 z_1) = \boldsymbol{\psi}_1^T\boldsymbol{y} - \boldsymbol{\psi}_1^T\boldsymbol{\phi}_1 z_1 \overset{\boldsymbol{\phi}_1^T\boldsymbol{\phi}_1=1}{=} \boldsymbol{\psi}_{\mathbf{1}}^T\boldsymbol{y} - z_1 \overset{\text{from } 2}{=} \boldsymbol{\psi}_1^T\boldsymbol{y} - \boldsymbol{\phi}_1^T\boldsymbol{y} = (\boldsymbol{\phi}_1 - \boldsymbol{\psi}_1)^T\boldsymbol{y} \overset{\phi_1 \neq \boldsymbol{\psi}_{\mathbf{1}}}{\neq} 0$. Hence $\boldsymbol{\psi}_1$ may be reselected.

The problem of designing the sensing matrix $\boldsymbol{\Psi}$ for the dictionary $\boldsymbol{\Phi}$ that results to a reduction in the cumulative coherence is formulated as looking for a gram matrix $\boldsymbol{G} = \boldsymbol{\Psi}^T\boldsymbol{\Phi}$ which has only ones in the diagonal and the diagonal elements are of an absolute value $\mu = \sqrt{\frac{N-M}{M(N-1)}}$. In practice the authors are aiming to design a sensing dictionary that for a known M, N reaches the lower bound in terms of coherence implied by the Welch bound introduced in (2.12). The process of the associated algorithm is as follows: Initialization:

- $\mathcal{G} := \{\boldsymbol{G} = \boldsymbol{\Psi}^T\boldsymbol{\Phi}, \boldsymbol{\Psi} \in R^{N \times M}\}$
- $\mathcal{H} := \{\boldsymbol{H} \in R^{N \times N},$ with $H_{ii} = 1$ and $|H_{ij}| \leqslant \mu,$ for $i \neq j\}$

Then the associated algorithm acquires the sensing matrix $\boldsymbol{\Psi}$ by solving the minimization problem introduced in

$$\min||\boldsymbol{G} - \boldsymbol{H}||_F \text{ s.t } \boldsymbol{G} \in \mathcal{G}, \boldsymbol{H} \in \mathcal{H} \tag{2.73}$$

This type of problems is solved via Projection Onto Convex Sets (POCS). The associated algorithm operates on an iterative manner as introduced in **Algorithm 9**.

2.7 Conclusion

In this Chapter we have reviewed the mathematical model for the standard sparse recovery problem. We discuss the mathematical formulation upon different decomposition algorithms and we review them in terms of theoretical properties and computational complexity. The link between data aware methodologies and greedy techniques for sparse non–negative signal processing is then drawn, which is the main goal of

Algorithm 9 Preconditioning

```
1: Input: $\mathbf{\Phi}, \mu_{min} = \sqrt{\frac{N-M}{M(N-1)}}$
2: **Initialization**: $\boldsymbol{G}^0 = \mathbf{\Phi}^T\mathbf{\Phi}$.
   at $k$–th iteration
3: Shrink: Calculate $\boldsymbol{H} \in R^{N\times N}$whose entries $h^k_{i,j}$ satisfy
```

$$h^k_{i,j} = \text{soft}(z, \theta) = \begin{cases} 1 & \text{of } i = j \\ g^{k-1}_{i,j} & \text{if } g^{k-1}_{i,j} \leqslant \mu_{min} \\ \text{the sign if} g^{k-1}_{i,j} \times \mu_{min}, & \text{if } g^{k-1}_{i,j} > \mu_{min} \end{cases} \tag{2.74}$$

```
4: Provisional Sensing matrix: $\mathbf{\Psi}^k = (\boldsymbol{H}^k\mathbf{\Phi}^\dagger)^T$.
5: Update Sensing Matrix: Multiply the matrix $\mathbf{\Psi}^k$ with the diagonal matrix containing
   $(\boldsymbol{\phi}_i^T\boldsymbol{\psi}_j)^{-1}$ on the main diagonal such that the new matrix $\mathbf{\Psi}^k$ satisfies $\langle\boldsymbol{\phi}_j, \boldsymbol{\psi}_j\rangle = 1$.
6: Pseudo–Gram matrix: Compute $\boldsymbol{G}^k = (\mathbf{\Psi}^k)^T\mathbf{\Phi}$.
7: if $k = K$ stop and output $\mathbf{\Psi}$, else continue.
```

the current work. We present the different data aware methodologies that are related with the different aspects of the problem we are willing to tackle. A discussion upon the different properties of the data that these methods are aiming to exploit is also conducted. We evaluate the different data aware methodologies with respect to two basic criteria: 1) computational complexity, 2) theoretical properties. We highlight the improvement that may be introduced in the sparse non–negative signal processing problem with respect to two basic criteria that severely affect the performance of real–world applications: 1) acceleration, and 2) robustness.

Chapter 3

Fast and Exact Nearest Neighbor Search: Lower Dimensional Representation and Sparse Signal Processing

3.1 Introduction

The operational time of a given algorithm is a key requirement when considering it as a potential candidate for a real–time application. Modern times are characterised by a significant increase on the data size which severely affects the computational complexity of commonly used algorithms. Such an example is FNNOMP where the operational time of the algorithm heavily relies on the selection step of the algorithm which can be summarised as: MN operations, where N is the number of points and M the dimensions of the Euclidean space they live in.

A common approach which is followed in order to reduce the computational complexity of these methods is to reduce the computational cost from the perspective of M. In such case the point cloud is embedded in a lower dimensional space $R^K : K < M$. By doing that we are aiming to benefit from the lower computational cost that characterises the brute force search in a lower dimensional space R^K. The latter from the perspective of computational cost can be be summarised as: KM operations.

In this chapter, we introduce novel algorithms for the so–called Nearest Neighbor problem that bridge the gap in between approximate to exact NN algorithms. The chapter is organised as follows: In sections 3.1–3.4 we introduce a novel algorithm which demonstrates the acquisition of the exact NN is feasible in terms of exact recovery as well as acceleration. This work was published in the *Senson Signal Processing for Defense* conference 2019. Nevertheless, the algorithm is theoretically fragile given that the associated recovery condition cannot be generalised for any unseen point. Therefore, at the remainder of the chapter we extend our work as follows: we introduce a theoretically robust framework that extends the theoretical foundation of embedding overall and leads to the development of a novel and theoretically robust algorithm for the standard NN problem. We extend our results to the Maximum Inner Product Search problem (MIPS) as well. We demonstrate that this sort of approach may benefit from a particular property that may characterise natural data i.e. lower dimensional representations. We demonstrate the benefits of such an approach in terms of acceleration by incorporating our algorithms to the selection of FNNOMP .

3.2 Linear Embeddings

In this section we introduce the guidelines for reducing the size of problem via a Linear Embedding. The standard notion regarding dimensionality reduction is that by having an input signal $\boldsymbol{y} \in R^M$, the dimension of the signal is reduced via a linear operator $\boldsymbol{Q} : R^M \rightarrow R^K$, with $K < M$, that embeds the input signal into the lower dimensional space. The projection of the signal $\hat{\boldsymbol{y}}$ in R^K is then computed as follows: $\hat{\boldsymbol{y}} = \boldsymbol{Q}^T \boldsymbol{y}$.

Linear embedding is a standard approach in many applications where we seek for a low–dimensional representation of data living on a high–dimensional space. There exist different methods to perform the embedding, e.g., principal component analysis [93], random projections [94] etc. A common characteristic of these embeddings is that the relevant position between dictionary elements is changed when the points are embedded from R^M to R^K. In that sense, given a pair of elements $\boldsymbol{\phi}_i, \boldsymbol{\phi}_c \in R^M$ and their representations $\hat{\boldsymbol{\phi}}_i, \hat{\boldsymbol{\phi}}_c \in R^K$, we usually have: $d(\boldsymbol{\phi}_i, \boldsymbol{\phi}_c) \neq d(\hat{\boldsymbol{\phi}}_{\boldsymbol{i}}, \hat{\boldsymbol{\phi}}_{\boldsymbol{c}})$ (where $d(\boldsymbol{\phi}_i, \boldsymbol{\phi}_c) = ||\boldsymbol{\phi}_i - \boldsymbol{\phi}_c||_2$). For an algorithm that searches for the nearest neighbor of $\boldsymbol{y}$ in $\boldsymbol{\Phi}$, this may lead to a situation in which $\text{NN}_M \neq \text{NN}_K$ where NN is the abbreviation for the Nearest Neighbor and M, K corresponds to the dimensions of each euclidean space. At this section we introduce the Embedded Nearest Neighbor (E–NN) algorithm that under a specific condition the search in the lower dimensional space eventually yields the nearest neighbor in the original domain. In that sense we are seeking an embedding that yields a minimum distortion from $R^M \rightarrow R^K$. This aspect of the problem can be addressed in terms of a reformulation of the Constructive Johnson–Lindenstrauss [95] introduced in (3.1) where $d(\boldsymbol{b}, \boldsymbol{t})$, where $\boldsymbol{b}, \boldsymbol{t} \in \mathcal{A} \subset R^M$. Let $\boldsymbol{Q}$ distort the distance for

Algorithm 10 Embedded NN (E–NN)

```
Input: $\Phi, \hat{\Phi}, Q, y$.
$\hat{y} = Qy$.
Form set S=$\{i : \ d(\hat{\phi}_i, \hat{y}) \leqslant \ d(y, \mathrm{NN}_K) + \delta\}, \forall \hat{\phi}_i \in \hat{\Phi}$.
return $arg\ min_{i \in S}\ d(y, \Phi_i)$ .
```

at most $\epsilon_{b,t}$. We then have:

$$\begin{aligned}(1-\epsilon_{bt})d(\boldsymbol{b},\boldsymbol{t}) \leqslant d(\hat{\boldsymbol{b}},\hat{\boldsymbol{t}}) &\leqslant (1+\epsilon_{bt})d(\boldsymbol{b},\boldsymbol{t}) \\ d(\boldsymbol{z},\boldsymbol{t}) - \epsilon_{bt}d(\boldsymbol{b},\boldsymbol{t}) \leqslant d(\hat{\boldsymbol{b}},\hat{\boldsymbol{t}}) &\leqslant d(\boldsymbol{b},\boldsymbol{t}) + \epsilon_{bt}d(\boldsymbol{b},\boldsymbol{t}) \\ d(\boldsymbol{b},\boldsymbol{t}) - \delta \leqslant d(\hat{\boldsymbol{b}},\hat{\boldsymbol{t}}) &\leqslant d(\boldsymbol{b},\boldsymbol{t}) + \delta\end{aligned} \tag{3.1}$$

where,

$$\delta = \max_{\boldsymbol{b},\boldsymbol{t} \in \mathcal{A} \subset R^M} \ \epsilon_{bt}d(\boldsymbol{b},\boldsymbol{t}), \tag{3.2}$$

Theorem 2.$\forall \boldsymbol{b}, \boldsymbol{t} \in \mathcal{A}$ with a δ coming from (3.2) and $\forall \boldsymbol{y} \notin \mathcal{A}$ with max $\epsilon_{yt}d(\boldsymbol{y},\boldsymbol{t}) \leqslant \delta$, the E–NN introduced in Algorithm 1 guarantees the acquisition of the exact NN.

Proof: Considering three points $\boldsymbol{y}, \boldsymbol{b}, \boldsymbol{t}$ where $d(\boldsymbol{y},\boldsymbol{b}) \leqslant d(\boldsymbol{y},\boldsymbol{t})$. Then there exist 4 characteristic cases for pairwise distances.

- The case where both distances shrink: $d(\hat{\boldsymbol{y}},\hat{\boldsymbol{b}}) \leqslant d(\boldsymbol{y},\boldsymbol{b})$, $d(\hat{\boldsymbol{y}},\hat{\boldsymbol{t}}) \leqslant d(\boldsymbol{y},\boldsymbol{t})$. Then by incorporating (3.1) :
  $d(\hat{\boldsymbol{y}},\hat{\boldsymbol{b}}) \leqslant d(\boldsymbol{y},\boldsymbol{b}) \leqslant d(\boldsymbol{y},\boldsymbol{t}) + \delta$.
- The case where both distances stretch. Then from (3.1) we have:
  $d(\hat{\boldsymbol{y}},\hat{\boldsymbol{b}}) - \delta \leqslant d(\boldsymbol{y},\boldsymbol{b}) \leqslant d(\boldsymbol{y},\boldsymbol{t})$
  $\Rightarrow d(\hat{\boldsymbol{y}},\hat{\boldsymbol{b}}) \leqslant d(\boldsymbol{y},\boldsymbol{t}) + \delta$
- the case where $d(\hat{\boldsymbol{y}},\hat{\boldsymbol{b}})$ stretches: $d(\boldsymbol{y},\boldsymbol{b}) \leqslant d(\hat{\boldsymbol{y}},\hat{\boldsymbol{b}})$, $d(\hat{\boldsymbol{y}},\hat{\boldsymbol{t}})$ shrinks: $d(\boldsymbol{y},\boldsymbol{t}) \leqslant d(\hat{\boldsymbol{y}},\hat{\boldsymbol{t}}) + \delta$. Then it follows:
  $d(\hat{\boldsymbol{y}},\hat{\boldsymbol{b}}) \leqslant d(\boldsymbol{y},\boldsymbol{b}) + \delta \leqslant d(\boldsymbol{y},\boldsymbol{t}) + \delta$.
- The case where $d(\hat{\boldsymbol{y}},\hat{\boldsymbol{b}})$ shrinks: $d(\hat{y},\hat{b}) \leqslant d(\boldsymbol{y},\boldsymbol{b})$, $d(\hat{\boldsymbol{y}},\hat{\boldsymbol{t}})$ stretches: $d(\boldsymbol{y},\boldsymbol{t}) \leqslant d(\hat{\boldsymbol{y}},\hat{\boldsymbol{t}})$. Then:
  $d(\hat{\boldsymbol{y}},\hat{\boldsymbol{b}}) \leqslant d(\boldsymbol{y},\boldsymbol{b}) \leqslant d(\boldsymbol{y},\boldsymbol{t}) + \delta$. □

The analysis provided by proof of *Theorem 2* simply states that in cases where $NN_M \neq NN_K$, assuming that $b = NN_M$ and $t = NN_K$, then $d(\hat{\boldsymbol{y}}, \hat{\boldsymbol{\phi}}_{NN_M}) \leqslant d(\boldsymbol{y}, \boldsymbol{\phi}_{NN_K}) + \delta$.

The complexity of the E–NN introduced in **Algorithm 10** varies over steps 2–4 of the algorithm. At step 2 the input signal $\boldsymbol{y} \in R^M$ is embedded in R^K via the linear operator $\boldsymbol{Q} \in R^{K\times M}$. Hence the complexity of step 2 is $\mathcal{O}(KM)$. At step 3 we conduct a number of N distance computations over K–dimensional vectors. The computational cost of the corresponding operations is $\mathcal{O}(KN)$. Finally, at the last step of the algorithm we perform a number of $|S|$ distance computations on the original space R^M. The computational cost of the step is $\mathcal{O}(|S|M)$.

As it can be derived from the analysis there are two critical parameters to benefit from the brute force search in the lower dimensional space. The lower dimensional representation of the dataset expressed by K and the cardinality of S on the update step which depends on δ.

Essentially we are seeking for an embedding $\boldsymbol{Q}$:

$$\delta = \min_{\boldsymbol{Q}} \max_{i,c} |||\boldsymbol{Q}(\boldsymbol{\phi}_i - \boldsymbol{\phi}_c)||_2 - ||\boldsymbol{\phi}_i - \boldsymbol{\phi}_c||_2|, \forall i, c \in \Phi. \tag{3.3}$$

The most common approach to construct a dimension reduction is PCA. The embedding to the K–dimensional space is simply performed by taking the K dominant eigenvectors of the data covariance matrix. The main drawback of PCA though is that it distorts pairwise distances arbitrarily. In that sense the distance distortion may be significantly larger from the one pair of points to the other.

An alternative to PCA is the approach of *random projections*. According to the Johnson–Lindenstrauss lemma, given any point cloud $\boldsymbol{\Omega}$ in R^M, there exists an embedding $\boldsymbol{Q}$ of dimension $K = \mathcal{O}(\log|\boldsymbol{\Omega}|)$ with minimal distortion of the $\binom{|\boldsymbol{\Omega}|}{2}$ pairwise distances between the $|\boldsymbol{\Omega}|$ points. This linear embedding is easy to implement in practice. We simply construct a matrix $\boldsymbol{Q} \in R^{K\times M}$ with elements drawn randomly from a certain probability distribution. The authors in [82], introduced a deterministic framework, called *NuMax*, that constructs linear and near–isometric embeddings for data that live in a high–dimensional space. Given a set of training points $\boldsymbol{\Phi} \in R^M$, the authors consider a *secant set* $S(\boldsymbol{\Phi})$ consisting of all pairwise difference vectors of $\boldsymbol{\Phi}$ that live on the unit sphere. The problem is formulated as an affine rank minimization problem to construct $\boldsymbol{Q}$ such that the norms of all vectors in $S(\boldsymbol{\Phi})$ are preserved up to a distortion parameter.

We aim to solve the problem introduced in Equation (3.3) empirically for dictionary *Raman* spectra with $M = 1507$ and $N = 4041$ [61] and a dictionary of Swiss Roll data [96] which is a synthetic machine Learning dataset of points that lie on a 2–D manifold

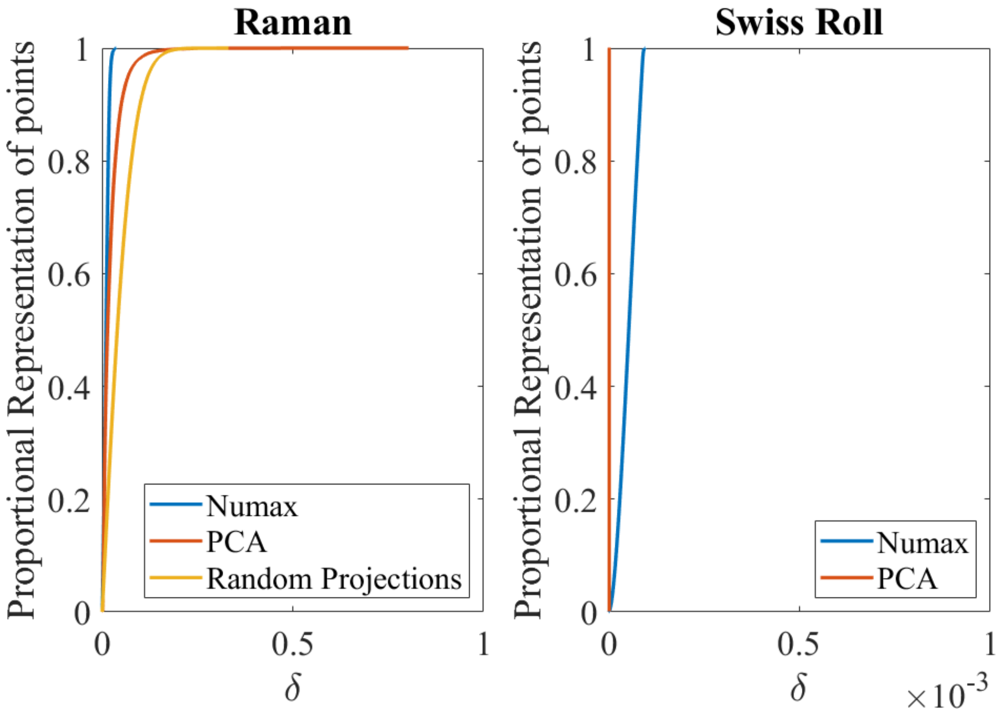


Figure 3.1: The figure demonstrates the empirical cumulative distribution function (CDF) of δ over Φ. The distortion on Φ introduced by random embeddings into Swiss Roll is much larger than *Numax* and PCA hence it is not demonstrated.

but embedded in R^{1507}. We found that the minimization problem introduced by the *NuMax* algorithm yields a matrix $\boldsymbol{Q} \in R^{K,M}$ with $K = 172$ for *Raman* while for the Swiss Roll case $K = 3$. Then we construct $\boldsymbol{Q}$ for PCA and *random projections* by setting $K = 172$ and $K = 3$ accordingly such that we can investigate which method serves the purpose for R^K.

The performance for each method is evaluated with respect to the error distortion function $\delta(\phi_i, \phi_c)$ as follows:

$$\delta(\boldsymbol{\phi}_i, \boldsymbol{\phi}_c) = |d(\boldsymbol{\phi}_i, \boldsymbol{\phi}_c) - d(\hat{\boldsymbol{\phi}}_i, \hat{\boldsymbol{\phi}}_c)|. \tag{3.4}$$

The obtained results are demonstrated in Figure 3.1. In practice at the particular figure we demonstrate the evaluation of the performance for each method with respect to the optimization problem introduced in (3.3). The *NuMax* algorithm outperforms PCA and *random projections* for the *Raman* dictionary. The latter indicates that a data aware operator that focus on the preservation of pairwise distances from R^M to R^K, demonstrates a better in terms of maintaining the underlying structure of the dataset. However, the is no theoretical guarantee that the one method may outperform the other. The latter can be validated only empirically. Moreover, the performance of each method is data dependent, e.g. *NuMax* and PCA demonstrate a relatively similar

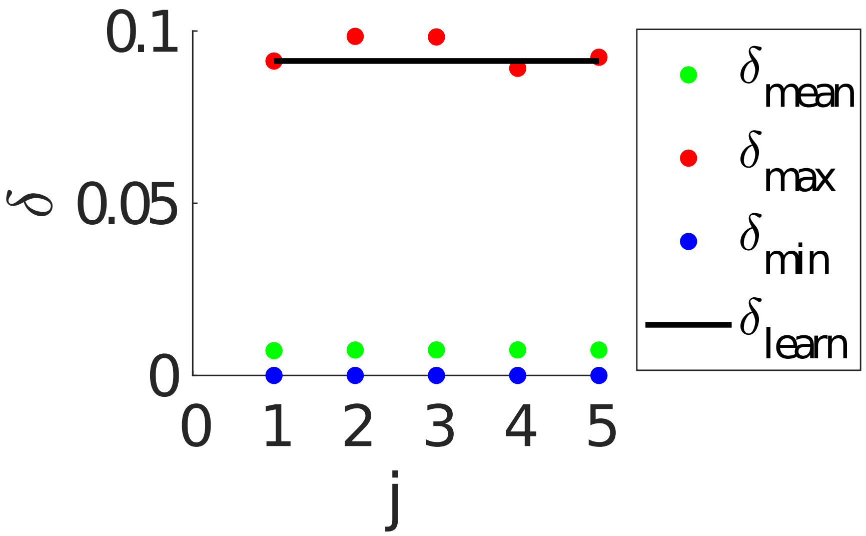


Figure 3.2: The figure demonstrates the range of distortion over sparsity. $\delta_{mean}(j) = \frac{1}{|\mathcal{Y}||\Phi|}(\sum_{y_m \in \mathcal{Y}} \sum_{\phi_i \in \Phi} \delta(\phi_i, y_m)), \delta_{max}(j) = \max\, \delta(\phi_i, y_m),\ \delta_{min}(j) = \min\, \delta(\phi_i, y_m)$

performance for the Swiss Roll data for lower K than the *Raman* data.

3.2.1 The case of mixtures

Within our framework we set δ with respect to the knowledge derived from elements that belong to an available dictionary $\boldsymbol{\Phi}$. The case of mixtures $\boldsymbol{y}$ is slightly different. In particular, each $\boldsymbol{y}$ with sparsity (number of contributing atoms) up to j is formulated as a linear combination of $\boldsymbol{\phi}_i \in \boldsymbol{\Phi}$ as follows: $\boldsymbol{y} = \sum_{w=1}^{j} a_w \boldsymbol{\phi}_w$.

This essentially means that there is not any particular knowledge regarding $\delta(\boldsymbol{\phi}_i, \boldsymbol{y})$. Hence, an obvious question is whether $\boldsymbol{y}$ is consistent with the choice of δ. Given that according to the results introduced in Figure 3.1 the $\boldsymbol{Q}$ obtained by the *NuMax* algorithm yields the best results we perform a simulation study for $\boldsymbol{y}$ over a sparsity level up to 5. The distortion is then evaluated according to the error distortion function introduced in (3.4) with $\boldsymbol{y}$ taking the place of $\boldsymbol{\phi}_i$ and $\hat{\boldsymbol{y}}$ the place of $\hat{\boldsymbol{\phi}}_i$ accordingly. For each j we generate a set of mixtures $\mathcal{Y} = \{y_m\}_{m=1}^{L}$ via 10000 (denoted as L) Monte Carlo simulations. The obtained results are demonstrated in Figure 3.2. Note that $a_w \sim U[0,1]$ and $||\boldsymbol{y}||_2 = ||\hat{\boldsymbol{y}}||_2 = 1$.

The results indicate that $\delta_{\max}$ flunctuates around δ_{learn}. We empirically observe that the maximum pairwise distortion $\forall \boldsymbol{\phi}_i \in \boldsymbol{\Phi}$ (denoted as δ_{learn}) exceeded only 0.003% over the total number of simulations. Even in these cases, the algorithm acquires the exact NN. This is happening due to the fact that the pairwise distortion is on average much lower than δ_{max} and a lower δ hence serves the purpose.

Algorithm 11 E–NN on FNNOMP

```
Initialization: s $= z_0 = \emptyset, j = 0, \boldsymbol{r}_0 = \boldsymbol{y}$.
while $j < K \& \max(\boldsymbol{\Phi}^T \boldsymbol{r}_k > 0)$.
    $\mu \leftarrow$ Embedded–NN.
    $p \leftarrow 1$.
    $p^c \leftarrow \mu$.
    $z^c = 0$
    while $\sim$ Terminate & $p < N$
        $z_t$ from (3.17).
        z$\leftarrow \boldsymbol{\psi}_\mu^T \boldsymbol{r}_k$: $\boldsymbol{\psi}_\mu = \frac{\boldsymbol{q}}{||\boldsymbol{q}||_2}$, $\boldsymbol{q} = (\boldsymbol{I} - \boldsymbol{\Psi\Psi}^T)\boldsymbol{\phi}_\mu$
        Update based on Table 3.1
    end while
    s = s $\cup\, \mu$.
    Update $\boldsymbol{\Psi}$ and R$^{-1}$
    $z_{k+1} \leftarrow [z_k, z_{k+1}]$
    $\boldsymbol{r}_{k+1} =\leftarrow \boldsymbol{r}_k - z_{k+1}\boldsymbol{\psi}_{k+1}$
    $k \leftarrow k + 1$
end while .
output: $\boldsymbol{x}|_s \leftarrow \boldsymbol{R}^{-1} z_j$
```

3.3 FNNOMP acceleration via a NNS approach

In this section we introduce an update on the structure of FNNOMP, which was introduced in Algorithm 2, with respect to the algorithm introduced in Algorithm 10. The first change in the structure takes place in the selection step of FNNOMP [10, p2] where we replace with E–NN. At a preprocessing stage of our method we embed $\boldsymbol{\Phi}$ in R^K via $\boldsymbol{Q}$. A common phenomenon in sparse non–negative decomposition is that a selected atom may be rejected by the non–negativity criteria introduced in Table 3.2 and with respect to equation (3.17). Consequently, we need to modify the content in Table 3.2 compared to the original FNNOMP version. A key aspect of the changes is the insertion of the Updated–NN algorithm, as introduced in Algorithm 3, such that E–NN adopts on the non–negativity setting. All the changes in the overall structure of FNNOMP are highlighted with red.

In practice U–NN can be addressed as a next NN Algorithm. In that sense anytime that the NN acquired by E–NN and indexed by μ is rejected by the criteria introduced

if	then
$0 < z \leqslant z^t, z > z^c$	$z_{j+1} \leftarrow z$, Terminate
$0 < z \leqslant z^t, z \leqslant z^c$	$z_{j+1} \leftarrow z^c, p \leftarrow p^c$, Terminate
$z > z^c \geqslant z^t$	$p = p + 1$, $\mu \leftarrow$U–NN
$z \geqslant z^c > z^t$	$z_{k+1} \leftarrow z^c, p \leftarrow p^c$, Terminate
$z > z^t > z^c$	$z^c \leftarrow z^t, p^c \leftarrow p$, $\mu \leftarrow$U–NN
$z < 0$	Terminate

Table 3.1: Non–negativity criteria FNNOMP

Algorithm 12 Updated NN

```
1: Input: $\boldsymbol{\Phi}, \hat{\boldsymbol{\Phi}}, \boldsymbol{y}, \boldsymbol{\phi}_{NN_M}$, S.
2: $\mathrm{S} = \mathrm{S} - NN_M$.
3: Form set $\mathrm{S}' = \{i : \ d(\hat{\boldsymbol{\phi}}_i, \hat{\boldsymbol{y}}) \leqslant \min d(\boldsymbol{y}, \boldsymbol{\phi}_{NN_K}) + \delta\}$.
4: Form set $S'' = S' - S$.
5: Return $arg\ min_{\phi_i \in S \cup S''}\ d(\boldsymbol{y}, \mathrm{S}) \cup d(\boldsymbol{y}, \mathrm{S}'')$ .
```

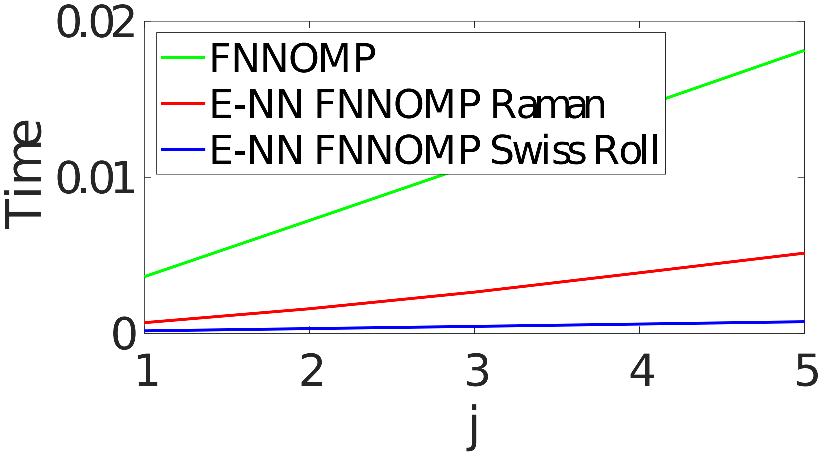
$$z_{j+1} \leqslant z^t = \begin{cases} \min\limits_{\gamma_i < 0} \frac{|x_i|}{|\gamma_i|} & \exists i, \gamma_i \leqslant 0 \\ \infty, \ \text{otherwise} & \end{cases} \tag{3.5}$$

In Table II, the task of U–NN is the acquisition of the next closest point to $\boldsymbol{y}$. To do as such we need to reject μ from S. This is done in step 2 of the algorithm.

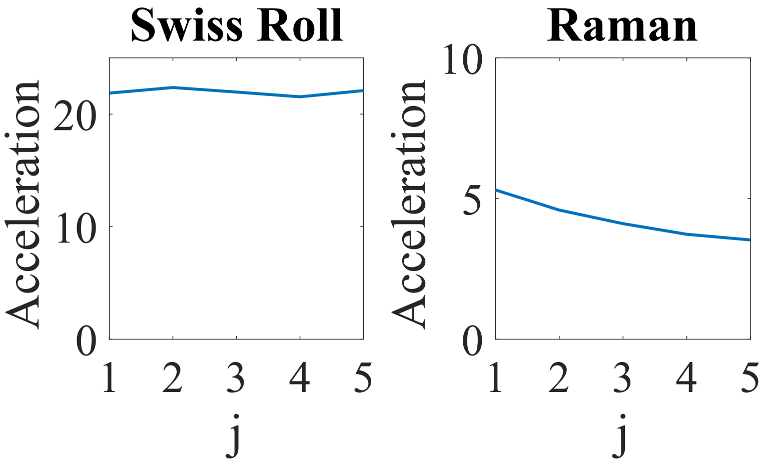
The implementation of E–NN provides U–NN with the full set of distance measurements in R^K and a number of distance measurements equal to $|S| - 1$ in R^K since μ is rejected in Step 2. Hence no additional distance computation is conducted in Step 3 of U–NN but a simple logical comparison that yields a new set of indices. Given that for some of these indices the distance in R^M is already available from E–NN we introduce Step 4 in order to avoid the recomputation. We then compute the distances for $\boldsymbol{\phi}_i \in S''$ and then we perform a comparison with the measurements of $\boldsymbol{\phi}_i \in S$ in order to find the next NN in R^M.

3.4 Results

In this section we evaluate the performance of the proposed algorithm with respect to FNNOMP. Based on the results introduced in Figure 3.1 we select the $\boldsymbol{Q}$ obtained by the *NuMax* algorithm as the linear operator that projects offline the dictionary $\boldsymbol{\Phi}$ and online the mixture $\boldsymbol{y}$ in R^K while for the Swiss Roll we select the $\boldsymbol{Q}$ obtained by PCA. We set $\delta = 0.09$ for the *Raman* dictionary and $\delta = 0$ for the *Swiss Roll.* We then generate signal mixtures of varying sparsity j from the elements in $\boldsymbol{\Phi}$.



(a)



(b)

Figure 3.3: Top of the figure: Elapsed time for each of the algorithms. Bottom: Acceleration over sparsity. Where $\text{Acceleration}(j) = \frac{\text{Time FNNOMP}(j)}{\text{Time E-NN FNNOMP}(j)}$.

The obtained results demonstrated in Figure 3.3 show that E–NN FNNOMP is generally faster than FNNOMP. In particular, for the case of *Raman* spectra the algorithm is approximately 4.5 times faster on average compared to FNNOMP while in case of *Swiss Roll* the acceleration is roughly 25 times. Given that the two datasets have the same cardinality (i.e. $|\mathbf{\Phi}| = 4041$), the reason why the algorithm is faster for the case of *Swiss Roll* compared to the case of *Raman* spectra has to do with the fact that $K = 172$ for *Raman* while $K = 3$ for *Swiss Roll.* Generally speaking, the lower the K the higher the acceleration.

The overall performance of the algorithm though decays over sparsity for the *Raman* spectra. Given that the computational cost at steps 1 and 2 of E–NN, the only parameter related to the complexity that may vary over j is $|S|$. In order to obtain a better understanding regarding that issue we demonstrate the average number of points

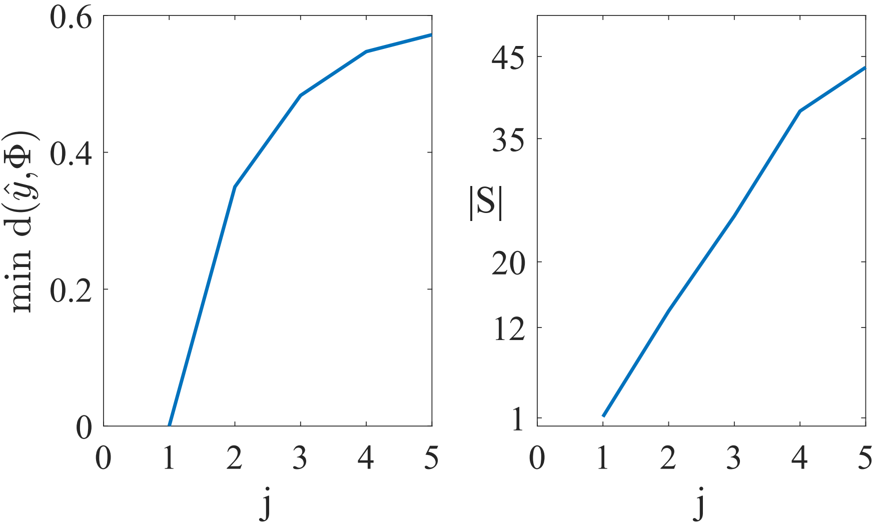


Figure 3.4: Average points in Step 4 of E–NN over sparsity.

per iteration of the algorithm in Figure 3.4. As can be seen from the results, the task of signal decomposition in the lower dimensional space becomes more difficult while sparsity increases. This is obviously not the case for the search in Swiss Roll. Essentially the acceleration factor remains constant. This happens because $\delta = 0$ hence the update step of E–NN is unnecessary. This means that in practice we compare the implementation of FNNOMP into different domains. This phenomenon may occur when all of the points that live in R^M in practice live in the same subspace R^K. As it can be seen from the *Raman* dictionary though this is not something to be expected in a realistic setting.

3.5 Review on E–NN

We here introduce two algorithms: 1) An Approximate Nearest Neighbor (Ap–NN) algorithm presented in Algorithm 13 2) and an Embedded–Nearest Neighbor algorithm (E–NN) [9]. At Step 2 in Algorithm 13 and Algorithm 14 accordingly, the input signal $\boldsymbol{y} \in R^M$ is embedded to a lower dimensional space R^K via a linear operator $\boldsymbol{Q} : R^M \rightarrow R^K$. At Step 3, a brute force search is operated in R^K in between $\hat{\boldsymbol{y}}$ and $\hat{\boldsymbol{\phi}}_i$ $\forall \hat{\boldsymbol{\phi}}_i \in \hat{\boldsymbol{\Phi}}$, where $\hat{\boldsymbol{y}}, \hat{\boldsymbol{\phi}}_{\boldsymbol{i}}$ are defined as follows:

Algorithm 13 Ap-NN

```
1: Input: $\hat{\Phi}, Q, y$.
2: $\hat{y} = Q^T y$.
3: $NN_K = \min_{\phi_i \in \Phi} d(\hat{y}, \hat{\phi}_i)$.
```

Algorithm 14 Embedded NN (E–NN)

```
1: Input: $\Phi, \hat{\Phi}, Q, y$.
2: $\hat{y} = Q^T y$.
3: Form set S=$\{i : \ d(\hat{y}, \hat{\phi}_i) \leqslant \ d(y, \phi_{NN_K}) + \delta\}, \forall \hat{\phi}_i \in \hat{\Phi}$.
4: return $argmin_{i \in S} \ d(y, \phi_i)$ .
```

- $\hat{\boldsymbol{y}} = \boldsymbol{Q}^T y$.
- $\hat{\boldsymbol{\phi}}_i = \boldsymbol{Q}^T \boldsymbol{\phi}_i$.
- $\hat{\boldsymbol{\Phi}} = \boldsymbol{Q}^T \boldsymbol{\Phi}$.

Note that NN stands for the nearest neighbor and the index M, K corresponds to the dimension of the Euclidean space the data points live in. In practice ϕ_{NN_M} corresponds to the representation of the nearest neighbor point in R^M while NN_M corresponds to the index of the particular point in matrix $\boldsymbol{\Phi}$ (note that the points of the dictionary $\boldsymbol{\Phi}$ are stacked column wise on a matrix form). The representation for each atom in R^M and R^K, is assigned with the same index in $\boldsymbol{\Phi}$, and $\hat{\boldsymbol{\Phi}}$ accordingly. Based on the definition for the NN search we obtain the following outcomes:

Outcome 1. $d(\boldsymbol{y}, \boldsymbol{\phi}_{NN_M}) \leqslant d(\boldsymbol{y}, \boldsymbol{\phi}_{NN_K})$.

Outcome 2. $d(\hat{\boldsymbol{y}}, \hat{\boldsymbol{\phi}}_{NN_K}) \leqslant d(\hat{\boldsymbol{y}}, \hat{\boldsymbol{\phi}}_{NN_M})$.

The distortion of the pairwise distances in between the dictionary points from R^N to R^K is materialized as follows:

$$\delta_{ij} = |d(\boldsymbol{\phi}_i, \boldsymbol{\phi}_j) - d(\hat{\boldsymbol{\phi}}_i, \hat{\boldsymbol{\phi}}_j)| \tag{3.6}$$

Given that a mismatch between the Nearest Neighbor from R^N to R^K may occur i.e., $NN_M \neq NN_K$ in step 3 of Ap–NN and E–NN, and an update step is introduced in Step 4 of E–NN to compensate the error. Essentially up to step 3 E–NN is an approximation algorithm. The additional step introduced by the algorithm is aiming to form a bridge in between the approximate to the exact solution of the problem. A critical parameter in the update step is the introduction of the penalty factor δ, where δ is defined as in (3.7).

$$\delta = \max_{\boldsymbol{\phi}_i, \boldsymbol{\phi}_j \in \boldsymbol{\Phi} \subset R^M} |d(\boldsymbol{\phi}_i, \boldsymbol{\phi}_j) - d(\hat{\boldsymbol{\phi}}_i, \hat{\boldsymbol{\phi}}_{\boldsymbol{j}})| \tag{3.7}$$

The outcome of the penalty factor is the result of the following analysis.

$$\begin{gathered} d(\boldsymbol{\phi}_i, \boldsymbol{\phi}_j) - \delta_{(i,j)} \leqslant d(\hat{\boldsymbol{\phi}}_i, \hat{\boldsymbol{\phi}}_j) \leqslant d(\boldsymbol{\phi}_i, \boldsymbol{\phi}_j) + \delta_{(i,j)} \\ d(\boldsymbol{\phi}_i, \boldsymbol{\phi}_j) - \delta \leqslant d(\hat{\boldsymbol{\phi}}_{\boldsymbol{i}}, \hat{\boldsymbol{\phi}}_{\boldsymbol{j}}) \leqslant d(\boldsymbol{\phi}_i, \boldsymbol{\phi}_j) + \delta \end{gathered} \tag{3.8}$$

Theorem 2.$\forall \boldsymbol{\phi}_i, \boldsymbol{\phi}_j \in \boldsymbol{\Phi}$ and $\forall \boldsymbol{y} \notin \boldsymbol{\Phi}$, if $\delta_{(\boldsymbol{y},\boldsymbol{\phi}_{NN_M})} = |d(\boldsymbol{y},\boldsymbol{\phi}_{NN_M}) - d(\hat{\boldsymbol{y}},\hat{\boldsymbol{\phi}}_{NN_M})| \leqslant \delta$, with δ coming from (3.2), the E–NN introduced in Algorithm 1 guarantees the acquisition of the exact NN.

By plugging $\boldsymbol{y}$ in (3.8) and δ in Outcome 1 we obtain the following relationships.

1. $d(\hat{\boldsymbol{y}},\hat{\boldsymbol{\phi}}_{NN_M}) \leqslant d(\boldsymbol{y},\boldsymbol{\phi}_{NN_M}) + \delta$.

2. $d(\hat{\boldsymbol{y}},\hat{\boldsymbol{\phi}}_{NN_K}) \leqslant d(\boldsymbol{y},\boldsymbol{\phi}_{NN_K}) + \delta$.

3. $d(\boldsymbol{y},\boldsymbol{\phi}_{NN_M}) \leqslant d(\boldsymbol{y},\boldsymbol{\phi}_{NN_K}) =>$
   $d(\boldsymbol{y},\boldsymbol{\phi}_{NN_M}) + \delta \leqslant d(\boldsymbol{y},\boldsymbol{\phi}_{NN_K}) + \delta$.

Proof. From Outcome 2, we have:

$$\begin{aligned}
&d(\hat{\boldsymbol{y}},\hat{\boldsymbol{\phi}}_{NN_K}) \leqslant d(\hat{\boldsymbol{y}},\hat{\boldsymbol{\phi}}_{NN_M}) \overset{1)}{=>} \\
&d(\hat{\boldsymbol{y}},\hat{\boldsymbol{\phi}}_{NN_K}) \leqslant d(\hat{\boldsymbol{y}},\hat{\boldsymbol{\phi}}_{NN_M}) \leqslant d(\boldsymbol{y},\boldsymbol{\phi}_{NN_M}) + \delta \overset{3)}{=>} \\
&d(\hat{\boldsymbol{y}},\hat{\boldsymbol{\phi}}_{NN_K}) \leqslant d(\hat{y},\hat{\phi}_{NN_M}) \leqslant d(\boldsymbol{y},\boldsymbol{\phi}_{NN_M}) + \delta \leqslant d(\boldsymbol{y},\boldsymbol{\phi}_{NN_K}) + \delta
\end{aligned} \tag{3.9}$$

□

The analysis provided by the proof of *Theorem 2* simply states that cases where $NN_M \neq NN_K$, then $d(\hat{\boldsymbol{y}},\hat{\boldsymbol{\phi}}_{NN_M}) \leqslant d(\boldsymbol{y},\boldsymbol{\phi}_{NN_K}) + \delta$.

Nevertheless, the version of the E–NN introduced in [9] has a basic disadvantage. The validity of the penalty factor holds if $\delta_{(y,NN_M)} \leqslant \delta$. In particular, the overall idea upon the update step relies on is that the representation of ϕ_{NN_M} in the lower dimensional space lives within a certain range from the representation of ϕ_{NN_K}. However, for $d(y,\phi_N N_M)$ we have the following relationship:

$$d(\hat{\boldsymbol{y}},\hat{\boldsymbol{\phi}}_{NN_M}) \leqslant d(\boldsymbol{y}\boldsymbol{\phi}_{NN_M}) + \delta_{(\boldsymbol{y},\boldsymbol{\phi}_{NN_M})} \tag{3.10}$$

where $\delta_{(\boldsymbol{y},\boldsymbol{\phi}_{NN_M})}$ is defined as:

$$\delta_{(\boldsymbol{y},\boldsymbol{\phi}_{NN_M})} = |d(\boldsymbol{y},\boldsymbol{\phi}_{NN_M}) - d(\hat{\boldsymbol{y}},\hat{\boldsymbol{\phi}}_{NN_M})| \tag{3.11}$$

Then E–NN will retrieve the exact NN as long as:

$$d(\hat{\boldsymbol{y}},\hat{\boldsymbol{\phi}}_{NN_M}) \leqslant d(\boldsymbol{y},\boldsymbol{\phi}_{NN_M}) + \delta_{(\boldsymbol{y},\boldsymbol{\phi}_{NN_M})} \leqslant d(\boldsymbol{y},\boldsymbol{\phi}_{NN_M}) + \delta \tag{3.12}$$

The relationship introduced in (3.12) holds as long as:

$$\begin{aligned} &\delta_{(\boldsymbol{y},\boldsymbol{\phi}_{NN_M})} \leqslant \delta => \\ |d(\boldsymbol{y},\boldsymbol{\phi}_{NN_M}) - d(\hat{\boldsymbol{y}},\hat{\boldsymbol{\phi}}_{NN_M})| &\leqslant \max_{\boldsymbol{\phi}_i,\boldsymbol{\phi}_j \in \boldsymbol{\Phi} \subset R^M} |d(\boldsymbol{\phi}_i,\boldsymbol{\phi}_j) - d(\hat{\boldsymbol{\phi}}_i,\hat{\boldsymbol{\phi}}_j)| \end{aligned} \tag{3.13}$$

In practice though, there is no guarantee that this requirement is fulfilled during the online process. The only available information during the online process is δ. In order to derive information regarding $\delta_{(\boldsymbol{y},\boldsymbol{\phi}_{NN_M})}$ we need to execute the following steps: 1) Brute force search in R^N in order to find NN_M 2) Embed $\boldsymbol{y}$ in R^K and calculate $d(\hat{\boldsymbol{y}},\hat{\boldsymbol{\phi}}_{NN_M})$. In case we would like to follow such an approach during the online process, by the time we execute step 1, then computationally we have already conducted the naive way of executing the NNS. In practice by using δ or any other arbitrarily selected threshold, we hope that the condition is fulfilled, but we can never be certain about it. Hence, Algorithm 1 is fragile from a theoretical point of view.

From a theoretical point of view and from the perspective of the NNS problem the goal of the current work is to introduce a theoretically robust algorithm that guarantees the recovery of the exact NN within the linear embeddings framework. Note that the bulk of the theoretical analysis that dominates the particular framework relies on the *JL* lemma. However, the associated analysis introduces an upper bound ϵ that holds explicitly for points $\boldsymbol{\phi}_i \in \Phi$ and it is not informative regarding an out of sample point $\boldsymbol{y} \notin \boldsymbol{\Phi}$. It is unclear whether this sort of analysis can be extended for $\boldsymbol{y}$ as well. In that sense, the associated algorithms that come along with the lemma, whether this is an Ap–NN with a Q that comes along with the associated analysis or LSH, inherit the drawback of the associated theoretical analysis.

The NNS problem is a well established framework within the academic community. Surprisingly though, there is not any straightforward comparison in between the two main approaches of the problem : 1) exact NNS 2) approximate NNS. This sort of comparison could take place with respect to two basis criteria: 1) acceleration 2) exact recovery. From a practical point of view, the current work conducts a straightforward comparison in between an exact solution of the problem and an approximate solution of the problem.

3.6 Linear Embedding Properties

In various applications, it is preferred to seek for a low–dimensional representation of points that originally live on a higher dimensional space. This is typically done via an embedding, i.e. a linear operator $Q : R^M \to R^K$. There exist two popular cases for linear embeddings, 1) data-adaptive *e.g.*, PCA [93] and 2) predefined, *e.g.*, RP [97]. A key characteristic of PCA is the fact that the linear operator is constructed in a relatively simple manner. This type of linear operation involves the mapping of N–dimensional data into a K–dimensional space with respect to the K dominant eigenvectors of the data covariance matrix. Strictly speaking though, PCA is a projection and not an embedding i.e., it is not one to one. However, it is known that applying a linear operator like PCA can typically lead to distances being distorted in an arbitrary manner. Essentially the particular type of embedding may map a pair of points that are distinct in R^M to be indistinguishable in R^K.

The authors in [82] introduced the *NuMax* algorithm which is a novel data–adaptive framework aiming to produce a linear operator distorting the pairwise distances up to an ϵ. Nevertheless, there is no guarantee that this ϵ can be generalised for an out of sample point $\boldsymbol{y} \notin \boldsymbol{\Phi}$.

The alternative to data–adaptive is the data–oblivious embeddings i.e. the RP approach. In particular, given a finite point cloud $\boldsymbol{\Phi}$ in R^M, there exists an embedding $\boldsymbol{Q}$ of dimension $K = \mathcal{O}(\log|\boldsymbol{\Phi}|)$ with minimal distortion of the $\binom{|\boldsymbol{\Phi}|}{2}$ pairwise distances between the $|\boldsymbol{\Phi}|$ points [97], where $| \cdot |$ indicates the cardinality of the set. A key aspect of the particular approach is that the mapping is near isometric in the sense that the embedding introduces a small distortion in the pairwise distances at least with high probability. This linear embedding is easy to implement in practice. We simply construct a matrix $\boldsymbol{Q} \in R^{K \times M}$ with elements drawn randomly from a certain probability distribution and then we scale it with an appropriate factor.

The common classification framework for linear embeddings, is the one that evaluates the distortion on pairwise distances in between points of known databases [82],[9] , or alternatively whether they are associated with some sort of theoretical justification (e.g., the JL lemma for RP). From the perspective of an online algorithm aiming to acquire the exact NN though, the classification is somewhat different. In particular, we classify the linear operators as follows: 1) Linear embeddings that shrink and stretch distances, and 2) Linear embeddings that explicitly shrink distances, i.e., non–expansive operators. The first class of linear embeddings is associated with E–NN. For the second class of embeddings, we introduce Adaptive Exact Nearest Neighbor (AE–NN) which

Algorithm 15 Adaptive Embedded NN (A E–NN)

```
1: Input: $\boldsymbol{\Phi}, \hat{\boldsymbol{\Phi}}, \boldsymbol{Q}, \boldsymbol{y}$.
2: $\hat{\boldsymbol{y}} = \boldsymbol{Q}^T \boldsymbol{y}$.
3: Form set S=$\{i : \ d(\hat{\boldsymbol{\phi}}_{\boldsymbol{i}}, \hat{\boldsymbol{y}}) \leqslant \ d(\boldsymbol{y}, \boldsymbol{\phi}_{\boldsymbol{NN_K}})\}, \forall \hat{\boldsymbol{\phi}}_{\boldsymbol{i}} \in \hat{\boldsymbol{\Phi}}$.
4: return $argmin_{i \in S} \ d(\boldsymbol{y}, \boldsymbol{\phi}_{\boldsymbol{i}})$ .
```

guarantees the acquisition of the exact NN in R^M.

Given that the operator shrinks the pairwise distances $d(\hat{\boldsymbol{y}}, \hat{\boldsymbol{\phi}}_i)$ in the embedded space R^K, then:

1. $d(\hat{\boldsymbol{y}}, \hat{\boldsymbol{\phi}}_{\boldsymbol{NN_M}}) \leqslant d(\boldsymbol{y}, \boldsymbol{\phi}_{\boldsymbol{NN_M}})$.
2. $d(\hat{\boldsymbol{y}}, \hat{\boldsymbol{\phi}}_{\boldsymbol{NN_K}}) \leqslant d(\boldsymbol{y}, \boldsymbol{\phi}_{\boldsymbol{NN_K}})$.

Theorem 3. For any operator $\boldsymbol{Q} : R^M \to R^K$ that shrinks distances, the Adaptive E–NN (AE–NN) introduced in Algorithm 15 guarantees the acquisition of the exact Nearest Neighbor.

Proof. Assuming that a mismatch in the embedded space occurs, hence from **Outcome 2**:

$$\begin{aligned} & d(\hat{\boldsymbol{y}}, \hat{\boldsymbol{\phi}}_{\boldsymbol{NN_K}}) \leqslant d(\hat{\boldsymbol{y}}, \hat{\boldsymbol{\phi}}_{\boldsymbol{NN_M}}) \overset{1)}{=>} \\ & d(\hat{\boldsymbol{y}}, \hat{\boldsymbol{\phi}}_{\boldsymbol{NN_K}}) \leqslant d(\hat{\boldsymbol{y}}, \hat{\boldsymbol{\phi}}_{\boldsymbol{NN_M}}) \leqslant d(\boldsymbol{y}, \boldsymbol{\phi}_{\boldsymbol{NN_M}}) \overset{\textbf{Outcome 1}}{=>} \\ & d(\hat{\boldsymbol{y}}, \hat{\boldsymbol{\phi}}_{\boldsymbol{NN_K}}) \leqslant d(\hat{\boldsymbol{y}}, \hat{\boldsymbol{\phi}}_{\boldsymbol{NN_M}}) \leqslant d(\boldsymbol{y}, \boldsymbol{\phi}_{\boldsymbol{NN_M}}) \leqslant d(\boldsymbol{y}, \boldsymbol{\phi}_{\boldsymbol{NN_K}}) \end{aligned} \tag{3.14}$$

Hence $d(\boldsymbol{y}, \boldsymbol{\phi}_{\boldsymbol{NN_K}})$ can be used as an upper bound in R^K in order to find the NN in R^M via the update step of the algorithm.

□

The case of *k*–nearest neighbors. Given a point $\boldsymbol{y} \in R^M$ and a number k, return the k–closest points to $\boldsymbol{y}$ in R^M. In this case the algorithm can be modified as follows:

- In Step 3, replace $d(\boldsymbol{y}, \boldsymbol{\phi}_{NN_K})$ by $d(\boldsymbol{y}, \boldsymbol{\phi}_{NN_K})(k)$.
- In Step 4, find the subset of points of interest in S by brute force.

The proof for k–NN case is a straight forward extension for the 1–NN as introduced in proof of ***Theorem 3*** where we assume that $k = 1$ (i.e. $d(\boldsymbol{y}, \boldsymbol{\phi}_{NN_M})(1)$). For $k = 1$,

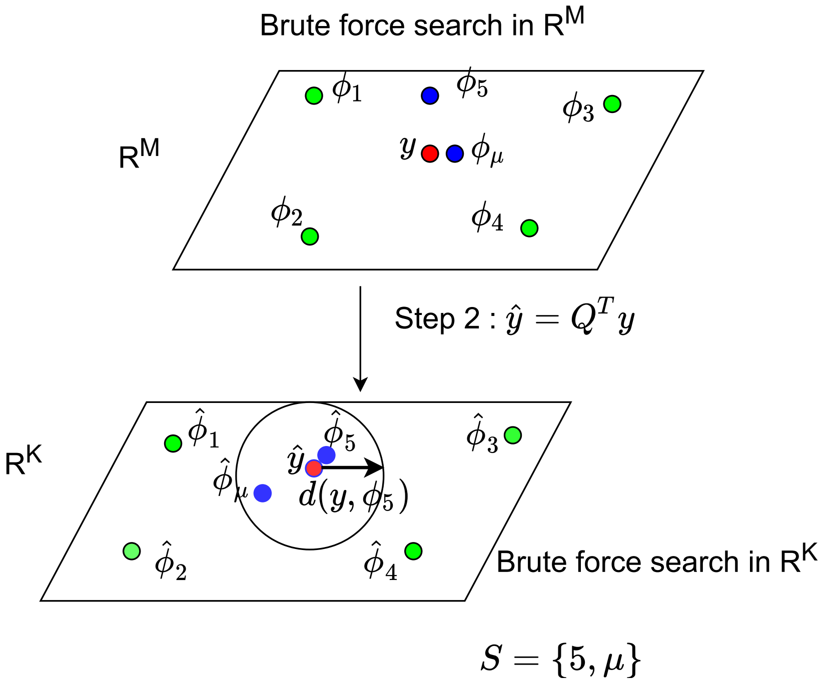


Figure 3.5: The figure demonstrates the operations of the algorithm on a step wise manner.

which is the standard NNS problem, we have discarded the corresponding index for simplicity. For the case of k–NN we have to plug in k at the corresponding index.

For better understanding of the process, we provide a visual demonstration of the step–wise process operated by the AE–NN algorithm in Figure 3.5. Specifically, we consider the case in which a mismatch between R^M and R^K occurs. More specifically, $NN_M = \mu$, $NN_K = 5$. We highlight with blue the points with the index that corresponds to NN_M and NN_K accordingly. The rest of the points are colored with green in order to highlight that they are not involved in the mismatch and the update step of AE–NN. Given that $NN_M = \mu$, then we have **fact 1**: $d(\boldsymbol{y}, \boldsymbol{\phi}_{NN_M}) \leqslant d(\boldsymbol{y}, \boldsymbol{\phi}_5)$.

The overall functionality of AE–NN can be described as follows: In Step 2 the input signal $\boldsymbol{y}$ is embedded in the subspace R^K via the non–expansive linear operator $\boldsymbol{Q}$. We obtain all the pairwise distance measurements in the embedded space R^K. Given that the positions in between the points of the cloud and the input signal $\boldsymbol{y}$ alter, a mismatch occurs from R^N to R^K. Hence: $NN_K = 5$. From that we have **fact 2**: $d(\hat{\boldsymbol{y}}, \hat{\boldsymbol{\phi}}_5) \leqslant d(\hat{\boldsymbol{y}}, \hat{\boldsymbol{\phi}}_{NN_M})$.

Upon the acquisition of the measurements in the embedded space, an extra measurement is conducted in between the representation in R^M of the closest point in R^K and the original representation of the input signal. In the scenario of Figure 3.5 this

Algorithm 16 Adaptive Embedded MIPS (Adaptive E–MIPS)

```
Input: $\boldsymbol{\Phi}, \hat{\boldsymbol{\Phi}}, \boldsymbol{Q}, \boldsymbol{y}$.
$\hat{\boldsymbol{y}} = \boldsymbol{Q}\boldsymbol{y}$.
Form set S=$\{i : \langle \hat{\boldsymbol{\phi}}_i, \hat{\boldsymbol{y}} \rangle \geqslant \langle \boldsymbol{y}, \boldsymbol{\phi}_{NN_K} \rangle + \frac{||\hat{\boldsymbol{y}}||^2 - ||\boldsymbol{y}||^2}{2} + \frac{||\hat{\boldsymbol{\phi}}_i||^2 - ||\boldsymbol{\phi}_{NN_K}||^2}{2}\}, \forall \hat{\boldsymbol{\phi}}_i \in \hat{\boldsymbol{\Phi}}$.
return $argmax_{i \in S} \langle y, \phi_i \rangle$ .
```

holds for $d(\boldsymbol{y}, \boldsymbol{\phi_5})$. From a geometrical point of view a ball is drawn around $\hat{\boldsymbol{y}}$ with a radius $d(\boldsymbol{y}, \boldsymbol{\phi_5})$. Given that $\boldsymbol{Q}$ fulfills a particular property, $\boldsymbol{Q}$ is a non–expansive operator, we have the following facts:

Fact 3: $d(\hat{\boldsymbol{y}}, \hat{\boldsymbol{\phi}}_{\boldsymbol{NN_M}}) \leqslant d(\boldsymbol{y}, \boldsymbol{\phi}_{\boldsymbol{NN_M}})$ and **fact 4**: $d(\hat{\boldsymbol{y}}, \hat{\boldsymbol{\phi}}_{\boldsymbol{5}}) \leqslant d(\boldsymbol{y}, \boldsymbol{\phi_5})$. From **fact 2** we have that: $d(\hat{\boldsymbol{y}}, \hat{\boldsymbol{\phi}}_{\boldsymbol{5}}) \leqslant d(\hat{\boldsymbol{y}}, \hat{\boldsymbol{\phi}}_{\boldsymbol{NN_M}})$. Then from **fact 3** we have: $d(\hat{\boldsymbol{y}}, \hat{\boldsymbol{\phi}}_{\boldsymbol{5}}) \leqslant d(\hat{\boldsymbol{y}}, \hat{\boldsymbol{\phi}}_{\mu}) \leqslant d(\boldsymbol{y}, \boldsymbol{\phi}_{\mu})$. Finally from **fact 1** and **fact 4** we have that: $d(\hat{\boldsymbol{y}}, \hat{\boldsymbol{\phi}}_5) \leqslant d(\hat{\boldsymbol{y}}, \hat{\boldsymbol{\phi}}_{\mu}) \leqslant d(\boldsymbol{y}, \boldsymbol{\phi}_{\mu}) \leqslant d(\boldsymbol{y}, \boldsymbol{\phi}_5)$. This essentially means that $\hat{\boldsymbol{\phi}}_{\boldsymbol{\mu}}$ is located within the volume of the ball or in other words $\mu \in S$. Hence, when we calculate the exact distances in the original space for the subset S, the exact NN can be located.

3.6.1 Maximum Inner Product Search: The case of non–expansive operators

The NNS problem finds many similarities with the so–called Maximum Inner Product Search problem (MIPS). That type of problem has received a significant attention in recent years as an essential step in many machine learning algorithms. Applications of MIPS can be found for example in matrix–factorization–recommender systems [98], multiclass prediction [99], structural SVM [100] and vision applications [99]. The MIPS problem can be formally described as follows: given a point cloud of N points $\Phi = \{\phi_1, \phi_2, \cdots, \phi_N\}$ where $\boldsymbol{\phi} \in R^M$ and a query point $\boldsymbol{y} \in R^M$ the goal is to find $\boldsymbol{\phi}_{\text{MIPS}}$ that maximizes the inner product $\langle q, \boldsymbol{\phi}_{\text{MIPS}} \rangle$. In other words the MIPS problem can be summarised as follows:

$$\boldsymbol{\phi}_{\text{MIPS}} := \underset{\phi_i \in \Phi}{\operatorname{argmax}} \langle \boldsymbol{y}, \boldsymbol{\phi}_i \rangle \tag{3.15}$$

MIPS has been of significant interest the last few years [101]. Despite the different approaches that may be followed to solve the problem, they all share a common characteristic: the problem is solved approximately. We here introduce Adaptive Embedded MIPS which is a bridge in between approximate to exact solutions for the MIPS problem.

Theorem 4 . For any operator $\boldsymbol{Q} : R^M \rightarrow R^K$ that shrinks distances between points, the Adaptive E–MIPS introduced in Algorithm 3 guarantees the acquisition of the point that maximizes the MIPS.

Proof. Given that any non–operator shrinks the pairwise distances assuming that a mismatch occurs, then with respect to ***Theorem 3*** we have the following:

$$\begin{aligned}
d(\hat{\boldsymbol{y}}, \hat{\boldsymbol{\phi}}_{NN_M}) &\leqslant d(\boldsymbol{y}, \boldsymbol{\phi}_{NN_K}) => \\
d(\hat{\boldsymbol{y}}, \hat{\boldsymbol{\phi}}_{NN_M})^2 &\leqslant d(\boldsymbol{y}, \boldsymbol{\phi}_{NN_K})^2 => \\
\langle \hat{\boldsymbol{y}}, \hat{\boldsymbol{\phi}}_{NN_M} \rangle &\geqslant \langle \boldsymbol{y}, \boldsymbol{\phi}_{NN_K} \rangle + \frac{||\hat{\boldsymbol{y}}||^2 - ||\boldsymbol{y}||^2 + ||\hat{\boldsymbol{\phi}}_{NN_M}||^2 - ||\boldsymbol{\phi}_{\boldsymbol{NN_K}}||^2}{2}
\end{aligned} \tag{3.16}$$

□

From a computational point of view AE–NN and AE–MIPS carry out the same workload for the same dimensional subspace R^K. In particular, step 1 carries out a number of KM operations for both frameworks. In step 2, the bound is in practice the same given that $\forall i \in S$ for AE–NN, we have the following relationship: $d(\boldsymbol{y}, \boldsymbol{\phi_i}) \leqslant d(\boldsymbol{y}, \boldsymbol{\phi}_{NN_K})$ that leads to $d(\boldsymbol{y}, \boldsymbol{\phi}_i)^2 \leqslant d(\boldsymbol{y}, \boldsymbol{\phi}_{NN_K})^2$, which is in practice the bound for AE–MIPS. At step 3, AE–NN requires extra vector–vector operations to calculate the Euclidean distance that results in a minimal additional computational overhead of $2N$ operations.

3.6.2 Non–Negative Sparse Signal Processing

In this section we introduce an update on the structure of FNNOMP, as introduced in Algorithm 2, with respect to the algorithm introduced in Algorithm 15. The first change in the structure takes place in the selection step of FNNOMP [10, pp2] where we make AE–NN adaptive. A common phenomenon in sparse non–negative decomposition, is that a selected atom may be rejected by the non–negativity criteria, introduced by the authors in [10], which are demonstrated in Table 3.2 and with respect to equation (3.17). Consequently, we need to modify the content in Table 3.2 compared to the original FNNOMP version. A key aspect of the changes is the insertion of the Adaptive Updated–NN (AU–NN) algorithm, as introduced in **Algorithm 17**, such that AE–NN is adapted on the non–negativity setting. All the changes in the overall structure of FNNOMP are highlighted with red.

Adaptive E–NN passes to AU–NN the full set of measurements D in the embedded space R^K and a set of measurements $\{d(\boldsymbol{y}, \boldsymbol{\phi}_i)\}_{i \in S}$ obtained by step 4 of the AE–NN algorithm. In practice Adaptive Updated–NN can be addressed as a next NN Algo-

if	then
$0 < z \leqslant z^t, z > z^c$	$z_{j+1} \leftarrow z$, Terminate
$0 < z \leqslant z^t, z \leqslant z^c$	$z_{j+1} \leftarrow z^c, p \leftarrow p^c$, Terminate
$z > z^c \geqslant z^t$	$p = p + 1$, $\mu \leftarrow$U–NN
$z \geqslant z^c > z^t$	$z_{j+1} \leftarrow z^c, p \leftarrow p^c$, Terminate
$z > z^t > z^c$	$z^c \leftarrow z^t, p^c \leftarrow p$, $\mu \leftarrow$U–NN
$z < 0$	Terminate

Table 3.2: Criteria that guarantee the positivity of the coefficients

rithm. In that sense anytime that the NN acquired by AE–NN and indexed by μ is rejected by the criteria introduced in Table 3.2, the task of AU–NN is the acquisition of the next closest point to $\boldsymbol{y}$. As such we need to discard μ from S. This is done in step 2 of the algorithm. In order to provide a better understanding regarding the functionality of AU–NN we consider a dictionary $\Phi = \{\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, \boldsymbol{\phi}_3, \boldsymbol{\phi}_4, \boldsymbol{\phi}_5, \boldsymbol{\phi}_6, \boldsymbol{\phi}_7, \boldsymbol{\phi}_\mu\}$, hence $D = \{d(\hat{\boldsymbol{y}}, \hat{\boldsymbol{\phi}}_i)\}_{\hat{\boldsymbol{\phi}}_i \in \hat{\boldsymbol{\Phi}}}$.

$$z_{j+1} \leqslant z^t = \begin{cases} \min\limits_{\gamma_i < 0} \frac{|x_i|}{|\gamma_i|} & \exists i, \gamma_i \leqslant 0 \\ \infty, \text{ otherwise} & \end{cases} \tag{3.17}$$

The case of a mismatch from R^M to R^K: $NN_K = 6, NN_M = \mu$. Such case is demonstrated in Figure 3.6. $S = \{\boldsymbol{\phi}_6, \boldsymbol{\phi}_7, \boldsymbol{\phi}_\mu\}$. Assuming that μ is rejected by the Non–Negativity criteria of FNNOMP we discard the corresponding index from S at step 2 of AU-NN and we are looking for the next NN. Given that the upper bound $d(\boldsymbol{y}, \boldsymbol{\phi}_6)$ is independent to μ then the upper bound for AU–NN is similar to AE–NN. Therefore at step 3 of AU–NN we have $S' = S - \mu$ and at step 4 we have $S'' = \varnothing$. Consequently no additional distance computation in R^M is required.

The case where $NN_M = NN_K$. Such case is demonstrated in Figure 3.7. At the particular scenario the closest point at the first run of AE –NN adaptive $\boldsymbol{\phi}_{\mu_1}$ is rejected by the non–negativity criteria of FNNOMP then the bound for AU–NN is updated since $d(\boldsymbol{y}, \boldsymbol{\phi}_{\mu_1}) \leqslant d(\boldsymbol{y}, \boldsymbol{\phi}_{\mu_2})$ where μ_2 is the 2nd NN of $\hat{\boldsymbol{y}}$ in the embedded space. Hence, it is most likely (though not necessary) that an extra point is introduced in our example $S' : S \cup \{4\}$, and $S" = \{4\}$ in step 4 of the algorithm. As a result an extra measurement $d(\boldsymbol{y}, \boldsymbol{\phi}_4)$ in R^M is required which increases the initial computational cost introduced by AE–NN.

3.7 Acceleration Ratio

AE–NN is an algorithm that guarantees the acquisition of the exact NN from R^N to R^K in any case. Nevertheless, this does not necessarily mean that AE–NN will be faster than brute force search in R^N in any case. Within the current section we will discuss in which cases this sort of approach will be faster than linear search and in what scenarios this acceleration will not be fulfilled.

In order to obtain a rough understanding regarding this issue, we can compare the computational complexity of the two approaches: linear search versus AE–NN. The overall computations required for the linear search are MN. For the AE–NN algorithm the stepwise operations analysis are as follows:

Algorithm 17 Adaptive E–NN on FNNOMP

```
Initialization: $s = z_0 = \varnothing, j = 0, \boldsymbol{r}_0 = \boldsymbol{y}$.
while $j < K \& \max(\boldsymbol{\Phi}^T \boldsymbol{r}_k > 0)$.
    i $\mu \leftarrow$ AE–NN.
    ii $p \leftarrow 1$.
    iii $p^c \leftarrow \mu$.
    iv $z^c = 0$
    v while $\sim$ Terminate & $p < N$
    vi     $z_t$ from (3.17).
    vii    z$\leftarrow \psi_\mu^T r_k$: $\psi_\mu = \frac{q}{||q||_2}$, $q = (I - \Psi\Psi^T)\phi_\mu$
    viii   Update based on Table 3.2
    ix end while
    x $s = s \cup \mu$.
    xi Update $\Psi$ and R$^{-1}$
    xii $z_{j+1} \leftarrow [z_j, z_{j+1}]$
    xiii $r_{j+1} \leftarrow r_j - z_{j+1}\psi_{j+1}$
    xiv $j \leftarrow j + 1$
end while .
output: $x|_s \leftarrow R^{-1} z_j$
```

Algorithm 18 Adaptive Updated NN (AU–NN)

```
Input: $\Phi, \hat{\Phi}, y, \mu, S, D$.
$D(\mu) = \infty, S = S - \mu$.
Form set $S' = \{i : \; D(i) \leqslant \; d(y, \phi_{NN_D})\}$.
Form set $S'' = S' - S$.
return $arg\ min_{\phi_i \in S \cup S''}\ d(y, S) \cup d(y, S'')$ .
```

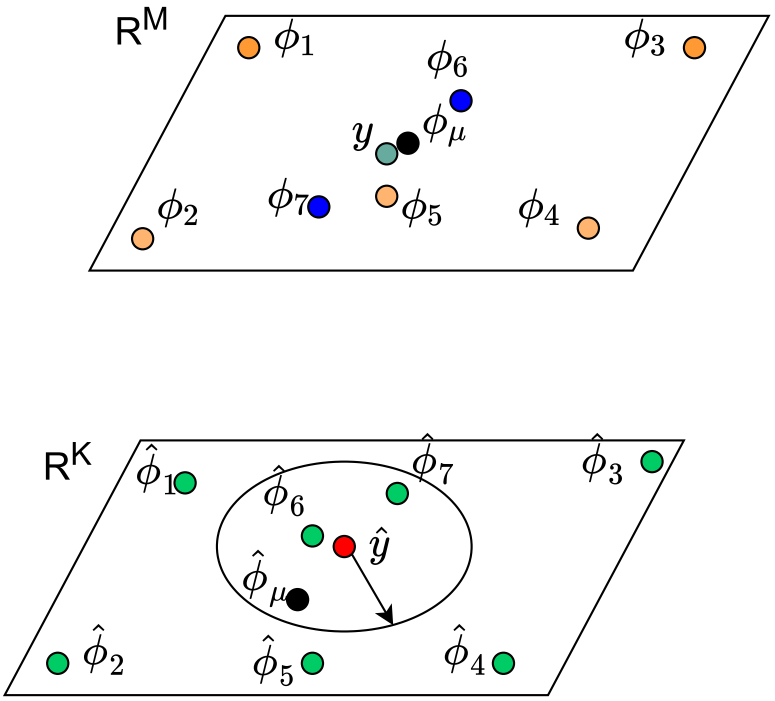


Figure 3.6: The mismatch case AU–NN.

1. Step 2. KM operations.
2. Step 3. KN operations.
3. Step 4. $|S|M$ operations.

Derived on this, we introduce the Acceleration Ratio metric introduced in (3.18) to provide a measure of the speed up that may be obtained by AE–NN.

$$\text{Acceleration Ratio} = \frac{MN}{KM + KN + |S|M} \tag{3.18}$$

The first issue we will address here, is for what size of K we expect that this type of approach will start being faster than linear search in R^N. Firstly though, we will split the computational cost of AE–NN into two basic categories:

1. **Fixed computational cost**: step 2 and step 3 of the algorithm
2. **Dynamic computational cost**: Step 4 of the algorithm.

The main computational workload of AE–NN is carried out by step 2 of the algorithm which is up to KN numerical operations. For any $K < M$, then since $N > 0$

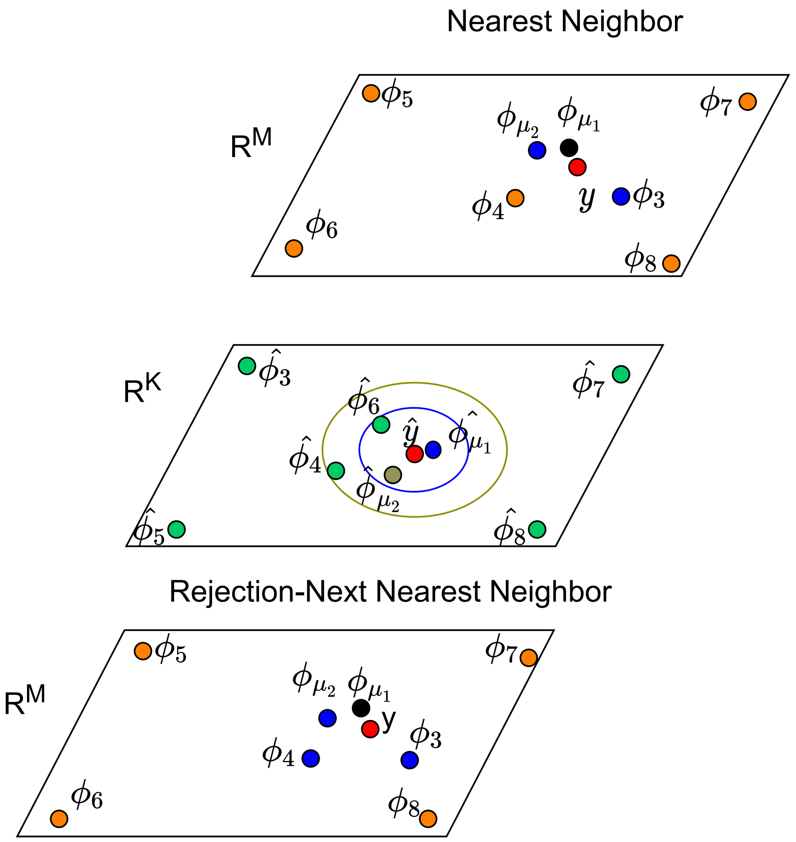


Figure 3.7: The $NN_M = NN_K$ case U–NN. The measurements in R^M obtained from the run of E–adaptive are highlighted with blue. ϕ_{μ_1} is rejected by the non–negativity criteria of FNNOMP hence it is colored with black. U–NN conducts an extra measurement $d(y, \phi_4)$ in R^M hence ϕ_4 it is also colored with blue.

then $KN < NM$, which is the computational workload of linear search. Nonetheless, the fixed computational cost of AE–NN includes the online embedding process of $\boldsymbol{y}$ in the lower dimensional space R^K which is up to KM operations. So, in order to start having some benefits in terms of acceleration the following requirement needs to be fulfilled:

$$K < \frac{MN}{M+N}. \tag{3.19}$$

The outcome of (3.19) is what we call the *Lower Acceleration Bound.*

Nevertheless, it is expected that in several cases the algorithm may be slower than the linear search given the extra computations expressed by **Dynamic computational cost**. The Acceleration Ratio can be used as guideline in order to understand in which cases this may happen:

$$KM + KN + |S|M > MN => |S| > \frac{MN - KM - KN}{M}. \tag{3.20}$$

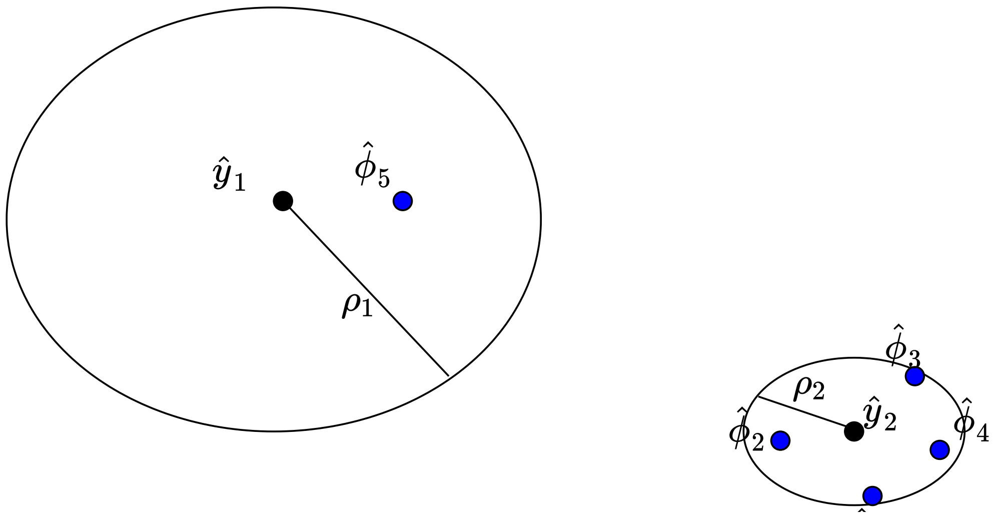


Figure 3.8: The figure demonstrate characteristic cases for the update step of the algorithm.

A natural consideration that may occur though, is what is the association between $|S|$ and $d(\boldsymbol{y}, \boldsymbol{\phi}_{NN_K})$, or in other words what is the relationship between the bound on step 2 and the number of points in S. Given that the upper bound for any in input $\boldsymbol{y}$ is $d(\boldsymbol{y}, \boldsymbol{\phi}_{NN_K})$, a ball is practically formed in step 2 of the algorithm.

From a theoretical point of view and when considering a continuous space, within a ball there may exist an infinite amount of points regardless of the size of the radius. Given that we perform the search on a finite point cloud this is not the case. In such case $|S|$ relies on two basic parameters: 1) the volume of the ball $\rho_i = d(\boldsymbol{y}_i, \boldsymbol{\phi}_{NN_{K_i}})$ (where NN_{K_i} denotes NN_K for each input signal $\boldsymbol{y}_i$) 2) and the density of the points on the particular area of the point cloud in R^K. In order to provide a deeper insight regarding this issue we demonstrate an example in Figure 3.8.

In particular we are considering a point cloud $\boldsymbol{\Phi} = \{\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, \cdots, \boldsymbol{\phi}_N\}$ with $N = 5$ and two different cases for input signals: $\boldsymbol{y}_1, \boldsymbol{y}_2$. Then for each signal we have the following relationships: $NN_{K_1} = 5$ and $NN_{K_2} = 2$. As it can be seen from the figure $\rho_1 > \rho_2$ which means that $d(\boldsymbol{y}_1, \boldsymbol{\phi}_{NN_{K_1}}) > d(\boldsymbol{y}_2, \boldsymbol{\phi}_{NN_{K_2}})$. Nevertheless, $|S|_1 < |S|_2$. This is due to the fact that the area around point 2 is more dense in terms of population compared to the area around point 5.

The example simply demonstrates that there may exist extreme cases where the associated bound may be wide and still obtain the optimal acceleration, while on the other hand the algorithm may perform poorly even in cases where the associated bound is relatively small.

3.7.1 Data Characteristics

Within the current section we provide a deeper insight regarding a particular property that subspace methods are aiming to tackle, i.e., the representation that natural signals may have in a lower dimensional space. Moreover, we will also demonstrate how the embedding operation: $\boldsymbol{Q} : R^M \rightarrow R^K$ may affect the underlying structure of the dictionary from R^N to R^K. The analysis will take place with respect to methodologies that yield a full orthnonormal basis such as PCA and Discrete Cosine Transform (DCT) [102]. The two approaches are different in the sense that the PCA coefficients are acquired on a linear fashion while the DCT coefficients on a non–linear fashion. However the operation of embedding the signal in R^K is linear in both cases. We perform the analysis with respect to these operators due to the fact that they are aligned with the requirement of AE–NN, in the sense that the associated embedding operator is orthonormal. This sort of embeddings do not expand distances by definition [25].

Let as assume that we are given such a basic $\boldsymbol{Q}_N$ provided by either PCA or DCT or any other method. Then the following relationships hold:

$$\boldsymbol{Q}_N^T \boldsymbol{Q}_N = \boldsymbol{I}_N = \boldsymbol{Q}_N^T \boldsymbol{Q}_N \tag{3.21}$$

The projection of the atom $\boldsymbol{\phi}_i$ on the complete orthonormal basis $\boldsymbol{Q}_N$ can be then obtained via the following relationship:

$$\boldsymbol{\phi}_i^\star = \boldsymbol{Q}_N^T \boldsymbol{\phi}_i, \tag{3.22}$$

The two signals $\boldsymbol{\phi_i}, \boldsymbol{\phi}_i^\star$ may be different in terms of coefficients, but they are identical by means of energy. This can be demonstrated as follows:

$$||\boldsymbol{\phi}_i^\star||_2^2 = \boldsymbol{\phi}_i^{\star T} \boldsymbol{\phi}_i^\star \overset{(3.22)}{=} \boldsymbol{\phi}_i^T \boldsymbol{Q}_N \boldsymbol{Q}_N^T \boldsymbol{\phi}_i \overset{(3.21)}{=} \boldsymbol{\phi}_i^T \boldsymbol{\phi}_i = ||\boldsymbol{\phi}_i||^2 \tag{3.23}$$

The original spectra $\boldsymbol{\phi}_i$ can be simply reconstructed via the measurements in $\boldsymbol{\phi}_i^\star$ and with respect to (3.22) as follows:

$$\boldsymbol{\phi}_i^\star = \boldsymbol{Q}_N^T \boldsymbol{\phi}_i => \boldsymbol{Q}_N \boldsymbol{\phi}_i^\star = \boldsymbol{Q}_N \boldsymbol{Q}_N^T \boldsymbol{\phi}_i \overset{(3.21)}{=>} \boldsymbol{Q}_N \boldsymbol{\phi}_i^\star = \boldsymbol{\phi}_i \tag{3.24}$$

The standard notion about PCA, is that we select the K–th principle eigenvectors with respect to some rule of thumb (i.e. number of the associated eigenvalues that preserve more than 90% of the energy on the points that lay in $\boldsymbol{\Phi}$). That process is equivalent to performing some sort of thresholding (i.e. preserve the K principle coefficients) on the coefficients of $\boldsymbol{\phi}_i^{\star}$:

$$\phi_i^{'}(m) := \begin{cases} \phi_i^{\star}(m), & \text{if } m \leqslant K. \\ 0, & \text{otherwise.} \end{cases} \tag{3.25}$$

We denote the vector of K non–zero coefficients in $\boldsymbol{\phi}_i^{'}$ as $\hat{\boldsymbol{\phi}}_i$. The relationship introduced in (3.25) can then be reformulated as follows:

$$\boldsymbol{\phi}_i = \boldsymbol{Q}_N \boldsymbol{\phi}_i^{\star} \approx \boldsymbol{Q}_N \boldsymbol{\phi}_i^{'} = \boldsymbol{Q}_K \hat{\boldsymbol{\phi}}_i, \tag{3.26}$$

where $\boldsymbol{Q}_K \in R^{N \times K}$ stands for the subset of K principle eigenvectors. Note that the equality holds, i.e. as in (3.27), when all of the $m+1, \cdots, N$ coefficients in $\boldsymbol{\phi}_i^{'}$ are actually 0. In such case the following relationship holds:

$$\boldsymbol{\phi}_i = \boldsymbol{Q}_K \hat{\boldsymbol{\phi}}_{\boldsymbol{i}} \tag{3.27}$$

By embedding $\boldsymbol{\Phi}$ in a lower dimensional space R^K, in practice we are exploiting the property that natural signals are characterised by some sort of lower dimensional representation:

$$\begin{aligned} ||\boldsymbol{\phi}_i||^2 = \boldsymbol{\phi}_i^T \boldsymbol{\phi}_i \overset{(3.26)}{\approx} \hat{\boldsymbol{\phi}}_{\boldsymbol{i}}{}^T \boldsymbol{Q}_K^T \boldsymbol{Q}_K \hat{\boldsymbol{\phi}}_{\boldsymbol{i}} \overset{\boldsymbol{Q}_K^T \boldsymbol{Q}_K = \boldsymbol{I}_K}{=} ||\hat{\boldsymbol{\phi}}_{\boldsymbol{i}}||^2 \\ => ||\boldsymbol{\phi}_{\boldsymbol{i}}||^2 \approx |\hat{\boldsymbol{\phi}}_{\boldsymbol{i}}||^2, \end{aligned} \tag{3.28}$$

where $\boldsymbol{\phi}_i \in R^N$ and $\hat{\boldsymbol{\phi}}_{\boldsymbol{i}} \in R^K$. Note the following, in cases where (3.27) holds, then:

$$||\boldsymbol{\phi}_i||_2 = ||\hat{\boldsymbol{\phi}}_{\boldsymbol{i}}||_2 \tag{3.29}$$

This is the way that the current work addresses the notion of lower dimensional representation, i.e. that a signal that originally lives in R^N has a compact representation (i.e., retains its energy) in a lower dimensional space R^K. We denote the lower dimensional representation of a a signal as $\mathcal{D}(\boldsymbol{\phi}_i)$. In accordance with the analysis upon the sparsity of $\boldsymbol{\phi}_i$ with respect to $\boldsymbol{Q}_N$, the relationship introduced in (3.29) holds in cases where:

$$||\boldsymbol{\phi}_i^{'}||_0 = K \tag{3.30}$$

In order to provide a more practical intuition regarding this analysis we demonstrate such an example in Figure 3.9. In particular we demonstrate the standard representation of a $\boldsymbol{\phi}_i \in \boldsymbol{\Phi}$ at subfigure a). Then we project $\boldsymbol{\phi}_i$ on the orthonormal bases $\boldsymbol{Q}_{PCA}$ and $\boldsymbol{Q}_{DCT}$ obtained by PCA and DCT via the process described in (3.21). The corresponding representations are demonstrated in subgfigures b) and c) accordingly. The red line is utilised in order to highlight the point beyond which the associated coefficients are 0. In particular we have that $\hat{\boldsymbol{\phi}}_{PCA}(k) = 0$ for $k > 280$ while $\hat{\boldsymbol{\phi}}_{DCT}(k) = 0$ for $k > 450$ in accordance with the analysis provided in (3.25) and $||\boldsymbol{\phi}_i|| = ||\hat{\boldsymbol{\phi}}_{PCA_{280}}|| = ||\hat{\boldsymbol{\phi}}_{DCT_{450}}|| = 1$, where the index 280 and 450 correspond to the number of coefficients utilised at the vector. In that sense, ϕ_i has a lower dimensional representation $\mathcal{D}_K$ which is exactly $\mathcal{D}_K = 280$ for PCA and $\mathcal{D}_K = 450$ for DCT. In practise Figure 3.9 demonstrates the representation of ϕ in 3 different orthonormal bases: 1) The standard representation $\boldsymbol{\phi}_i$ where $\boldsymbol{Q}_N = \boldsymbol{I}_N$ with $\boldsymbol{I}_N$ being the identity matrix 2) $\boldsymbol{Q}_{PCA}$ is the orthonormal basis obtained by PCA and 3) $\boldsymbol{Q}_{DCT}$ is the orthonormal basis obtained by DCT. In this particular scenarion $\boldsymbol{\phi}_i$ has a more sparse–compact representation on the PCA basis.

By fingerprinting the lower dimensional representation of each point in the point cloud $\boldsymbol{\Phi}$ in step 2 of AE–NN, we are aiming to obtain a layout of $\hat{\boldsymbol{\Phi}}$ in a subspace R^K which introduces a minimal deviation within the underlying structure of the dictionary compared to the original layout of $\boldsymbol{\Phi} \in R^N$. That essentially means that the pairwise distances are approximately preserved from the one space to the other. Let us consider a pair of points ϕ_1,ϕ_2, then from (3.26) we have the following relationships:

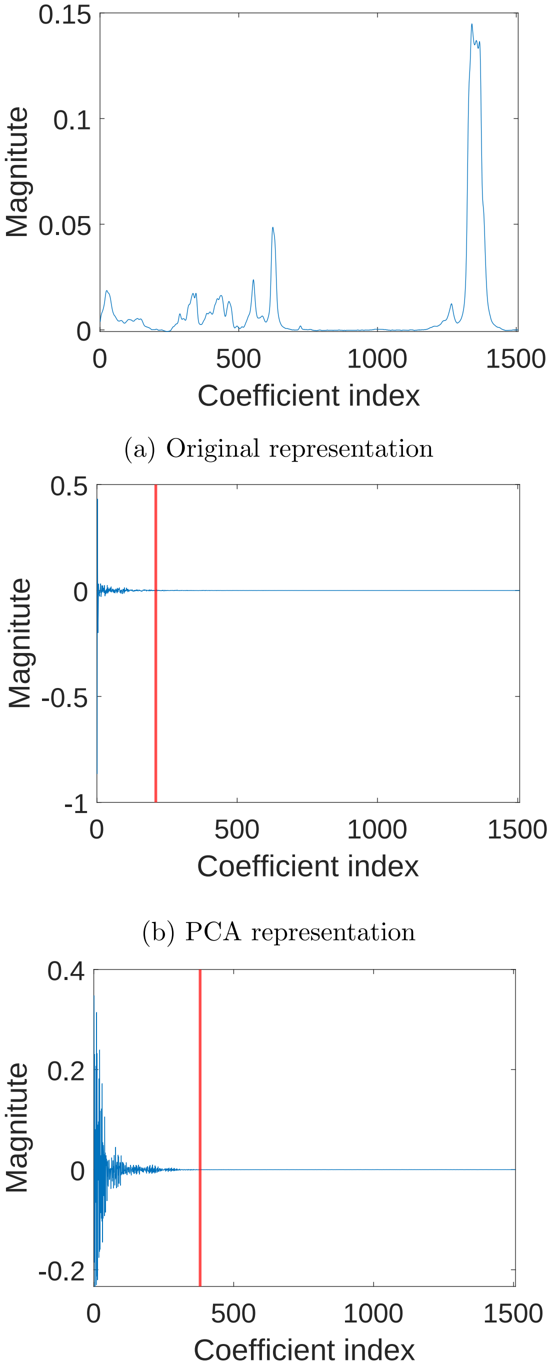
1. $\boldsymbol{\phi}_1 \approx \boldsymbol{Q}_K \hat{\boldsymbol{\phi}}_1$.
2. $\boldsymbol{\phi}_2 \approx \boldsymbol{Q}_K \hat{\boldsymbol{\phi}}_2$.
3. $\boldsymbol{Q}_K^T \boldsymbol{Q}_K = \boldsymbol{I}_K$.

$$\begin{aligned} ||\boldsymbol{\phi}_1 - \boldsymbol{\phi}_2||_2^2 &= (\boldsymbol{\phi}_1 - \boldsymbol{\phi}_2)^T (\boldsymbol{\phi}_1 - \boldsymbol{\phi}_2) \\ &\overset{1),2)}{\approx} (\hat{\boldsymbol{\phi}}_1 - \hat{\boldsymbol{\phi}}_2)^T \boldsymbol{Q}_K^T \boldsymbol{Q}_K (\hat{\boldsymbol{\phi}}_1 - \hat{\boldsymbol{\phi}}_2) \overset{3)}{=} ||\hat{\boldsymbol{\phi}}_1 - \hat{\boldsymbol{\phi}}_2||_2^2 \end{aligned} \tag{3.31}$$

Note that the approximation drops and equality holds when both signals align with (3.27). In such case the following outcome holds:

$$||\boldsymbol{\phi}_1 - \boldsymbol{\phi}_2||_2^2 = ||\hat{\boldsymbol{\phi}}_1 - \hat{\boldsymbol{\phi}}_2||_2^2 \tag{3.32}$$

This implies that in cases where all the signals have a lower dimensional repre-



(a) Original representation

(b) PCA representation

(c) DCT representation

Figure 3.9: Representation of $\boldsymbol{\phi}_i$ in different orthonormal bases

sentation K, then by embedding $\boldsymbol{\Phi}$ in R^K, we obtain a $\hat{\boldsymbol{\Phi}}$ with identical underlying structure compared to $\boldsymbol{\Phi}$. In such case the following relationship holds:

$$\boldsymbol{\phi}_i = \boldsymbol{Q}_K \hat{\boldsymbol{\phi}}_i \ \forall \boldsymbol{\phi}_i \in \Phi. \tag{3.33}$$

However, a key question that may occur is: What does this knowledge provide from the perspective of an online algorithm given that $\boldsymbol{y} \notin \boldsymbol{\Phi}$? Within the current work we have a particular interest for signals that are sparse with respect to a known dictionary: $\boldsymbol{y} = \sum_{i=1}^{k} a_i \hat{\boldsymbol{\phi}}_i : \boldsymbol{\phi}_i \in \boldsymbol{\Phi}$. For this particular class of signals we can demonstrate the following:

$$\begin{aligned}
&\boldsymbol{y} = \sum_{i=1}^{k} a_i \boldsymbol{\phi}_i \overset{(3.33)}{=} \sum_{i=1}^{k} \boldsymbol{Q}_K a_i \hat{\boldsymbol{\phi}}_i = \boldsymbol{Q}_K \hat{\boldsymbol{y}} \overset{(3.33)}{=>} \\
&\boldsymbol{y} - \boldsymbol{\phi}_j = \boldsymbol{Q}_K \hat{\boldsymbol{y}} - \boldsymbol{Q}_K \hat{\boldsymbol{\phi}}_j = \boldsymbol{Q}_K(\hat{\boldsymbol{y}} - \hat{\boldsymbol{\phi}}_j) => \\
&||\boldsymbol{y} - \boldsymbol{\phi}_j||_2^2 = (\boldsymbol{y} - \boldsymbol{\phi}_j)^T(\boldsymbol{y} - \boldsymbol{\phi}_j) = (\hat{\boldsymbol{y}} - \hat{\boldsymbol{\phi}}_j)^T \boldsymbol{Q}_K^T \boldsymbol{Q}_K (\hat{\boldsymbol{y}} - \hat{\boldsymbol{\phi}}_j) \overset{\boldsymbol{Q}_K^T \boldsymbol{Q}_K = \boldsymbol{I}_K}{=>} \\
&||\boldsymbol{y} - \boldsymbol{\phi}_j||_2^2 = ||\hat{\boldsymbol{y}} - \hat{\boldsymbol{\phi}}_j||_2^2 \ \forall \boldsymbol{\phi}_j \in \boldsymbol{\Phi}
\end{aligned} \tag{3.34}$$

Theorem 4. Given a lower dimensional space R^K, an orthonormal operator $\boldsymbol{Q}_K \in R^{N \times K}$, a dictionary $\boldsymbol{\Phi} : \mathcal{D}_{\boldsymbol{\phi}_i} = K \ \forall \boldsymbol{\phi}_i \in \boldsymbol{\Phi}$ and a signal $\boldsymbol{y}$ which is sparse with respect to $\boldsymbol{\Phi} : \boldsymbol{y} = \sum_{i=1}^{k} a_i \hat{\boldsymbol{\phi}}_i : i \in \mathrm{s}$, then the following outcomes always hold for AE–NN: 1) $NN_M = NN_K$ 2) $|S| = 1$.

Proof. From the definition of NNS we have the following relationship:

$$\begin{aligned}
&||\boldsymbol{y} - \boldsymbol{\phi}_{NN_M}|| \leqslant ||\boldsymbol{y} - \boldsymbol{\phi}_j|| \ \forall \boldsymbol{\phi}_j \in \boldsymbol{\Phi} : j \neq NN_M \overset{(3.34)}{=>} \\
&||\hat{\boldsymbol{y}} - \hat{\boldsymbol{\phi}}_{NN_M}|| = ||\boldsymbol{y} - \boldsymbol{\phi}_{NN_M}|| \leqslant ||\boldsymbol{y} - \boldsymbol{\phi}_j|| = ||\hat{\boldsymbol{y}} - \hat{\boldsymbol{\phi}}_j|| => \\
&||\hat{\boldsymbol{y}} - \hat{\boldsymbol{\phi}}_{NN_M}|| \leqslant ||\hat{\boldsymbol{y}} - \hat{\boldsymbol{\phi}}_j|| => NN_M = NN_K
\end{aligned} \tag{3.35}$$

From (3.35) we also have $||\boldsymbol{y} - \boldsymbol{\phi}_{NN_M}|| \leqslant ||\hat{\boldsymbol{y}} - \hat{\boldsymbol{\phi}}_j|| \forall j \neq NN_M$, hence the only point that lays within the bound is the representation of NN_M in R^K therefore $|S| = 1$. □

The outcome of *Theorem 4* states that for this explicit case Ar–NN will with a $\boldsymbol{Q}_K$ from PCA, DCT ,or any other operator which is a subset of an orthonormal basis, will retrieve the exact NN and the update step introduced by AE–NN FNNOMP is practically redundant.

3.7.2 Sparse Signal Processing: the Notion of Lower Dimensional Representation

The current subsection as well as the following one are not aiming to revisit a well established framework such as sparse signal processing. Our main scope is to highlight the similarities as well as the differences in between AE–NN and OMP. Moreover we are aiming to highlight the underlying process that takes place via orthogonalization.

The standard sparse signal processing model, implies that a signal $\boldsymbol{y}$ can be represented a linear combination of only a few elements of a known dictionary $\boldsymbol{\Phi}$. This linear relationship is more formally expressed as:

$$\boldsymbol{y} = \boldsymbol{\Phi}_s \boldsymbol{x}_s, \tag{3.36}$$

where $\boldsymbol{y} \in R^M$ $\boldsymbol{\Phi}_s \in R^{M \times k}$ and $\boldsymbol{x}_s \in R^k$. The subdictionary Φ_s can be represented by means of QR factorization as follows:

$$\boldsymbol{\Phi}_s = \boldsymbol{Q}_s \boldsymbol{R}_s, \tag{3.37}$$

where $\boldsymbol{Q}_s \in R^{M \times k}$ contains a subset of vectors that are mutually orthogonal and $\boldsymbol{R}_s \in R^{k \times k}$ an upper triangular matrix. By plugging (3.37) into (3.36), the original linear relationship is reformulated as follows:

$$\boldsymbol{y} = \boldsymbol{Q}_s \boldsymbol{R}_s \boldsymbol{x}_s \overset{\boldsymbol{R}_s \boldsymbol{x}_s = \boldsymbol{z}_s}{=} \boldsymbol{Q}_s \boldsymbol{z}_s, \tag{3.38}$$

where $z_s \in R^K$.

We can then obtain the following relationship:

$$||\boldsymbol{y}||^2 = \boldsymbol{y}^T \boldsymbol{y} \overset{(3.38)}{=} \boldsymbol{z}^T \boldsymbol{Q}^T \boldsymbol{Q} \boldsymbol{z} \overset{\boldsymbol{Q}^T \boldsymbol{Q} = \boldsymbol{I_K}}{=} \boldsymbol{z}_s^T \boldsymbol{z}_s = ||\boldsymbol{z}_s||^2 \tag{3.39}$$

The outcome of (3.39) implies that a signal which is K–sparse with respect to a known dictionary $\boldsymbol{\Phi}$, is a signal that can be exactly represented in a K–dimensional space, or else it has an K lower dimensional representation. This associated subspace R^K is explicitly designed with the $\boldsymbol{Q}_i$'s: $i \in s$.

Algorithm 19 Adaptive Embedded NN (AE–NN)

```
1: Input: $\boldsymbol{\Phi}, \hat{\boldsymbol{\Phi}}, \boldsymbol{Q}, \boldsymbol{y}$.
2: $\hat{\boldsymbol{y}} = \boldsymbol{Q}^T \boldsymbol{y}$.
3: Form set S=$\{i :\ d(\hat{\boldsymbol{\phi}}_i, \hat{\boldsymbol{y}}) \leqslant\ d(\boldsymbol{y}, \boldsymbol{\phi}_{\mathrm{NN}_K})\}, \forall \hat{\boldsymbol{\phi}}_i \in \hat{\boldsymbol{\Phi}}$.
4: return $arg\ min_{i \in S}\ d(\boldsymbol{y}, \boldsymbol{\phi}_i)$ .
```

3.7.3 Similarities in between AE–NN and OMP

Within the current subsection we are aiming to draw link in between the functionality of AE–NN and OMP. We will stretch the interest of the reader in step 2 of **Algorithm 17** and step 2 of **Algorithm 18** accordingly.

At step 2 of **Algorithm 17**, the corresponding algorithm is equipped with a matrix $\boldsymbol{Q} \in R^{M \times K}$. In practice this is a matrix with K columns that live in M dimensional space and they are orthonormal. The $\boldsymbol{Q}$ operator is practically obtained by performing PCA on $\boldsymbol{\Phi}$ and then picking the K first columns of $\boldsymbol{Q}_M \in R^{M \times M}$. The input signal $\boldsymbol{y} \in R^M$ is projected on the lower dimensional space R^K by performing a simple matrix–vector multiplication in between the transpose of $\boldsymbol{Q}$ and $\boldsymbol{y}$. In practice we multiply each of the K columns in $\boldsymbol{Q}_i : \boldsymbol{Q}_i \subset \boldsymbol{Q}$ with $\boldsymbol{y}$: $\hat{\boldsymbol{y}}(i) = \boldsymbol{Q}_i^T \boldsymbol{y}$. The outcome of the process would be the same if instead of performing a matrix–vector multiplication we were performing a number of M loops such that: $\hat{\boldsymbol{y}}(m) = \boldsymbol{Q}_m^T \boldsymbol{y}$, where $\boldsymbol{Q}_m$ is the m–th column of $\boldsymbol{Q}$ and $1 \leqslant m \leqslant K$.

In practice OMP performs exactly the same operation. The key difference is that in contrast to AE–NN where $\boldsymbol{Q}$ is plugged in via PCA, the elements of the matrix $\boldsymbol{Q}$ are unknown. They are constructed on the fly via i.e. the QR factorization with respect to the atoms that are picked at the selection step of the algorithm. In practice, the

Algorithm 20 OMP via QR factorization

```
1: Initialization: $s = \varnothing, j = 0, \boldsymbol{r}_0 = \boldsymbol{y}, \xi = \varnothing$.
2: while $j < K$ & max$(\boldsymbol{\Phi}^T \boldsymbol{r}_k > 0)$.
      i $[\mu, \iota]$=max $|\boldsymbol{\Phi}^T \boldsymbol{r}_j|$
      ii $s = s \cup \mu$.
      iii Update $\boldsymbol{Q}, \boldsymbol{R}$ factors.
      iv $z_j = Q_j^T y$.
      v $\xi \leftarrow [z_j, \xi]$
      vi $\boldsymbol{r}_{j+1} = \boldsymbol{r}_j - \boldsymbol{Q}_j \boldsymbol{z}_j$
      vii $j \leftarrow j + 1$
3: end
4: $\boldsymbol{x}_s = \boldsymbol{R}_s^{-1} \boldsymbol{\xi}$
```

coefficients $\boldsymbol{z}_j$ in OMP are obtained via the same operation that the coefficients $\hat{\boldsymbol{y}}(m)$ are obtained via AE–NN. Essentially, OMP via the orthogonalization step projects $\boldsymbol{y}$ in a lower dimensional space. The size of the Euclidean subspace increases at each iteration of the algorithm. The coefficients $\boldsymbol{x}_i : i \in s$ are then obtained by projecting the coefficients $\boldsymbol{z}_j$ of lower dimensional space R^K back to the original space R^N via the R operation. The latter explains one of the key differences in between OMP and MP, which is somehow hidden under the hood. On OMP the coefficients $\boldsymbol{x}_i : i \in \mathrm{s}$ are updated at each iteration k of the algorithm. This is not the case for MP though. The reason why this phenomenon occurs is that at each iteration of OMP, the signal is projected on a subspace with an increasing size. By projecting the coefficients $\boldsymbol{z}_j$ back to R^N via i.e. the $\boldsymbol{R}$ operation we demonstrate the representation of $\boldsymbol{y}$ in different subspaces, i.e. when we stop the algorithm at each iteration $k-1$ we have a representation of the signal in a R^{k-1} while in iteration k we have a representation of the signal in R^k etc.

The analysis conducted here is aiming to highlight that an algorithm that incorporates an orthogonalization procedure essentially fingerprints the lower dimensional representation of $\boldsymbol{y}$ with respect to $\boldsymbol{\Phi}$. This kind of characteristic holds for OMP and CoSaMP type algorithms.

Note though that there is a difference between the particular approach and the one followed by AE–NN. The functionality of AE–NN uses exactly the same operator $\boldsymbol{Q} \in R^{M \times K}$ for both $\boldsymbol{\Phi}$ and $\boldsymbol{y}$. In that sense, the dictionary $\boldsymbol{\Phi}$ and the input signal $\boldsymbol{y}$ are embedded in the same subspace R^K.

From the perspective of Sparse Signal Processing, this is not the case. In particular, let us assume that we are given two input signals $\boldsymbol{y}_1$,$\boldsymbol{y}_2$ with $|\mathrm{s}_1| = |\mathrm{s}_2| = k$ but $\mathrm{s}_1 \neq \mathrm{s}_2$. Then $\boldsymbol{Q}_1 \neq \boldsymbol{Q}_2$ where $\boldsymbol{Q}_1, \boldsymbol{Q}_2 \in R^{M \times k}$. In that sense, even though $\boldsymbol{y}_1, \boldsymbol{y}_2$ have the same size of lower dimensional representation k, with respect to $\boldsymbol{\Phi}$, the subspace R^k they live in differs and it is spanned by $\boldsymbol{Q}_1$,$\boldsymbol{Q}_2$ accordingly.

In total, by incorporating AE–NN at the selection step of an FNNOMP we fingerprint lower dimensional representation on a dual manner: 1) At the selection step we fingerprint the K lower dimensional representation of $\boldsymbol{\Phi}$ and $\boldsymbol{y}$ with respect to the same subspace R^K 2) While decomposing the signal over the iterations of FNNOMP we fingerprint the k lower dimensional representation of $\boldsymbol{y}$ with respect to $\boldsymbol{\Phi}$.

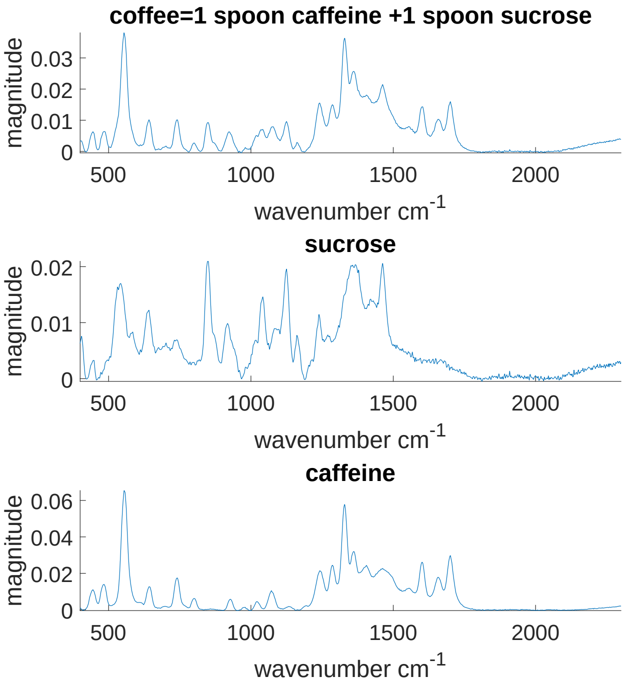


Figure 3.10: An example of an input signal $\boldsymbol{y}$ (plot on the top) that consists of caffeine and sucrose.

3.8 Simulation Results

Within the current section we compare state of the art OMP type algorithms for Sparse Non–Negative signal decomposition such as FNNOMP [10], NNOMP, [62], NN CoSaMP [63] with respect to operational time and approximation error.

The first set of simulations takes place with respect to the *Raman* data provided by [61]. The dimensions of the dictionary $\boldsymbol{\Phi}$ are $M = 1507$, $N = 4041$. We generate mixtures : $\boldsymbol{y} = \sum_{w=1}^{j} a_w \boldsymbol{\phi}_w$ where $a_w \sim U[0,1]$ and then each algorithm is assigned to decompose the input signal $\boldsymbol{y}$.

Raman spectroscopy is a technique used to observe vibrational, rotational and other low–frequency modes in a system. The method is widely used in different scientific disciplines i.e. cancer detection [54], nanotechnology [55] as well as in industrial applications [103],[56] such as the pharmaceutical industry where *Raman* spectroscopy is applied in order to identify pharmaceutical ingredients and their polymorphic forms, food safety [57], hazardous materials detection [59].

In order to provide a deeper insight regarding the raman spectral decomposition problem and draw the link with the sparse signal processing aspect of it we demonstrate a practical example in Figure 3.10. We demonstrate 3 different spectra. Within our disposal we have a database of *Raman* dictionary where among the available samples is the digital fingerprint of caffeine and sucrose. The corresponding digital fingerprint is demonstrated in the provided figure. We generate a synthetic mixture of these two spectras which is a linear combination of caffeine and sucrose where the associated coefficient is set to $a = 1$ and $b = 1$ accordingly. In the ideal scenario, an OMP type of algorithm such as FNNOMP will select the 2 components that constitute the signal within the first two iterations of the algorithm and $\boldsymbol{y}$ will be fully reconstructed. However, the selection rule of the OMP type algorithms is highly sensitive to the underlying structure of $\boldsymbol{\Phi}$. In particular, in cases where a ground truth atom $\boldsymbol{\phi}_i$ has a relatively similar digital fingerprint, i.e., they are highly coherent with each other, with a $\boldsymbol{\phi}_j$ (for example instead of sucrose it could pick brown sugar) that does not contribute to $\boldsymbol{y}$ it is quite possible that a false positive may be introduced in the support set s. The introduction of a false positive may be critical in several applications. Let us address the particular issue from the perspective of hazardous materials detection, a false positive may correspond to a case where the algorithm detects a non–hazardous material where in practice the mixture which is taken into account may consist of a hazardous material. On the other hand from the perspective of food safety the algorithm may detect an ingredient that the food is safe while in practise the mixture consists of an ingredient that demonstrates that the food is contaminated. Within the current work, we are aiming to accelerate such a framework without increasing the sensitivity of the selection rule. Therefore we introduce an exact NN algorithm. In contrast by incorporating an approximation algorithm ,i.e. Ap–NN, essentially we introduce a framework that increases the sensitivity of an existing framework.

For AE–FNNOMP we select: 1) PCA and we set $K = 210$ 2) DCT and we set $K = 370$ (more details on how to tune K in the remainder of the current section). For the Ap–NN algorithm we use PCA and DCT for the same K as in FNNOMP and a version of the Ap–NN algorithm where the associated embedding operator $\boldsymbol{Q}$ is designed with respect to the RP principles (i.e., the entries in $\boldsymbol{Q}$ are sampled independently from a Gaussian $N(0, 1)$ and then scaled by an appropriate factor). For the particular operator we set $K = 450$ which is the lowest bound for K with respect to the formula introduced in [104]. We plug in the algorithm at the selection step of FNNOMP. In case where an atom is rejected by the non–negativity criteria of FNNOMP we select the next NN in the embedded space R^K.

For the rest of the section, we use the empirical Acceleration Ratio as a metric as introduced in (3.40):

$$\text{Acceleration Ratio}(l) = \frac{\text{Time FNNOMP}(l)}{\text{Time \ NN–OMP type algorithm}(l)}, \qquad (3.40)$$

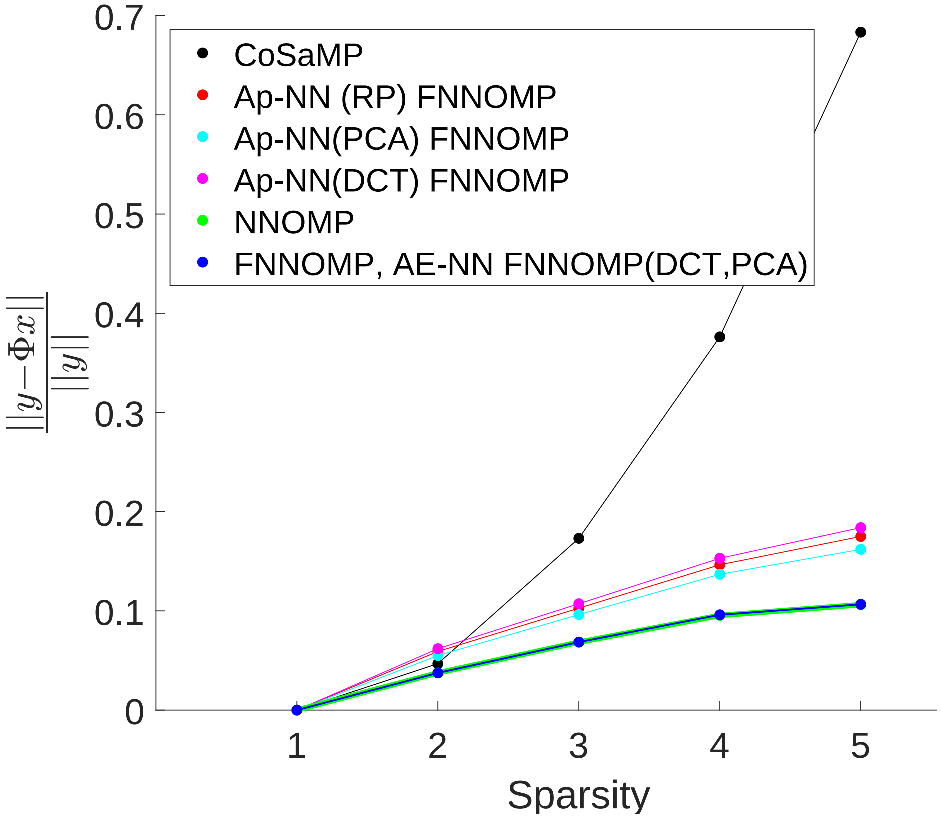


Figure 3.11: Comparisons of different sparse non–negative signal processing algorithms in terms of reconstruction error.

where l corresponds to the signal sparsity.

We perform a simulation study over 1000 realizations. We first analyse the performance of the algorithms in terms of acceleration. The obtained results are demonstrated in Figure 3.11.

From the perspective of reconstruction error FNNOMP and NNOMP demonstrate similar performance as it can be seen in Figure 3.11. The different versions of Ap–NN algorithm increase the reconstruction error compared to the result of the original algorithm that varies depending on the embedding operator. On the other hand, the AE–NN FNNOMP algorithm introduces an identical reconstruction error regardless of the operator, DCT or PCA. The latter demonstrates that the extra step introduced by AE–NN closes the gap which is originally introduced by the Ap–NN versions of the associated operators. As mentioned before, AE–NN is not aiming to introduce a new sort of selection rule that improves the recovery performance of the original framework. From the perspective of an approximation algorithm such as FNNOMP, AE–NN guarantees that the empirical approximation error introduced by the original algorithm will be exactly recovered from AE–NN. Note that an AE–NN FNNOMP with an RP type

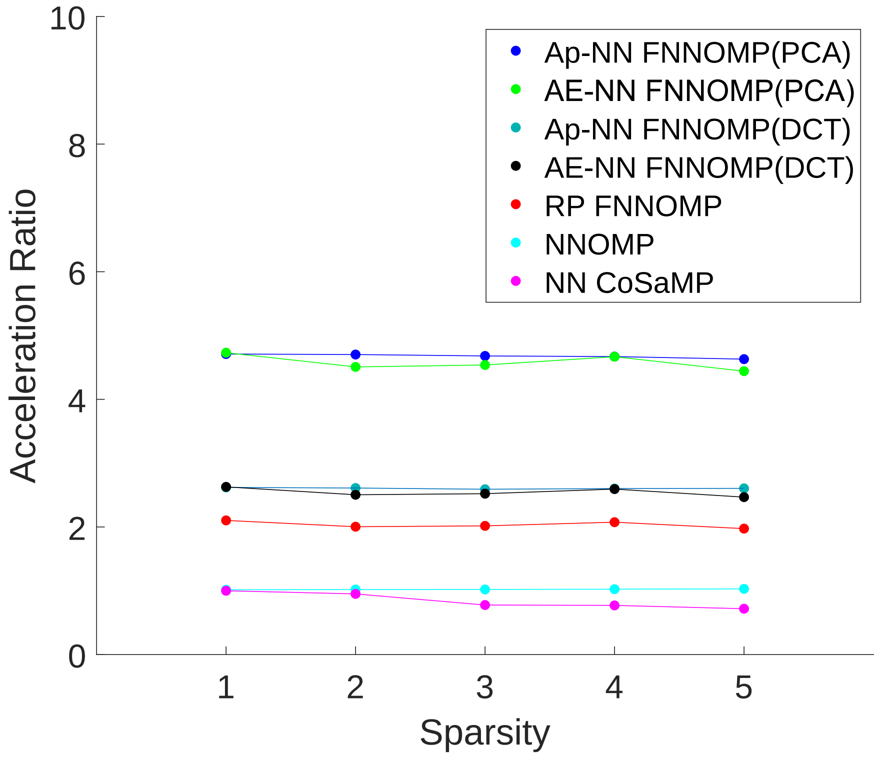


Figure 3.12: Comparison in between sparse non–negative signal processing algorithms in the time domain.

of operator is not feasible given that the associated operator is not an non–expansive one.

From the perspective of acceleration, the results indicate that FNNOMP and NNOMP demonstrate a relatively similar performance with respect to the particular criterion as demonstrated in Figure 3.12. The latter is not surprising given the fact that the main computational workload is carried out by the selection step of the algorithms which is in practise identical. The two algorithms outperform NN CoSaMP in terms of acceleration. AE–NN FNNOMP and RP–FNNOMP accelerate FNNOMP. This outcome highlights the benefits that can be exploited by performing the search in a lower dimensional space compared to R^N. AE–NN FNNOMP accelerates FNNOMP by approximately 4.5 times over sparsity while RP–FNNOMP accelerates FNNOMP but with a smaller factor i.e. approximately 2.1 times. This is due to the number of dimensions of the subspace we embed the dictionary which is higher compared to AE–NN FNNOMP. Generally speaking, the benefits in terms of acceleration increase while reducing the number of dimensions in R^K.

Our analysis does not consider the case of noisy data which is a common characteristic of real world applications. In practise, what we would expect in this sort of

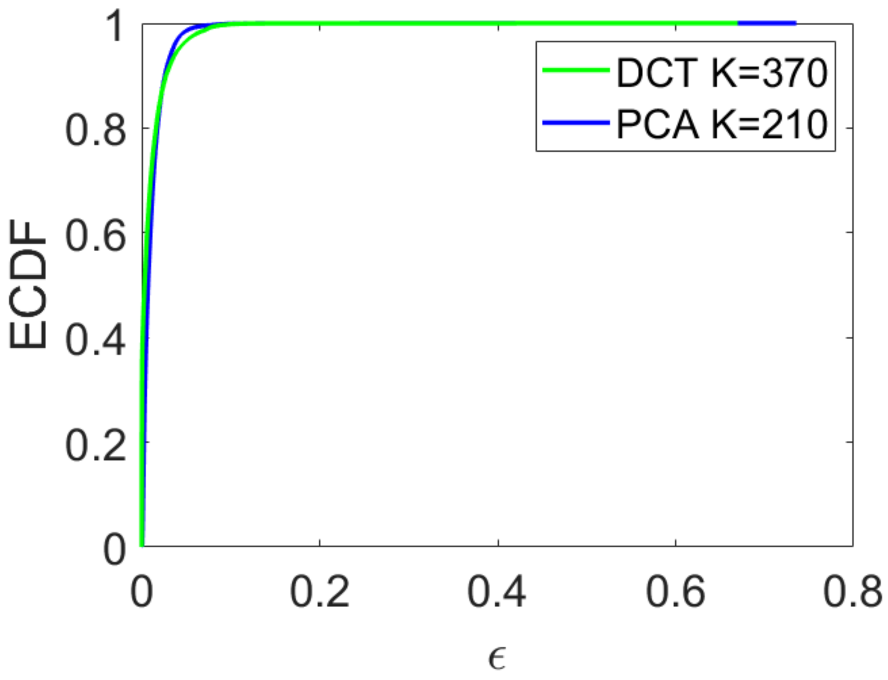
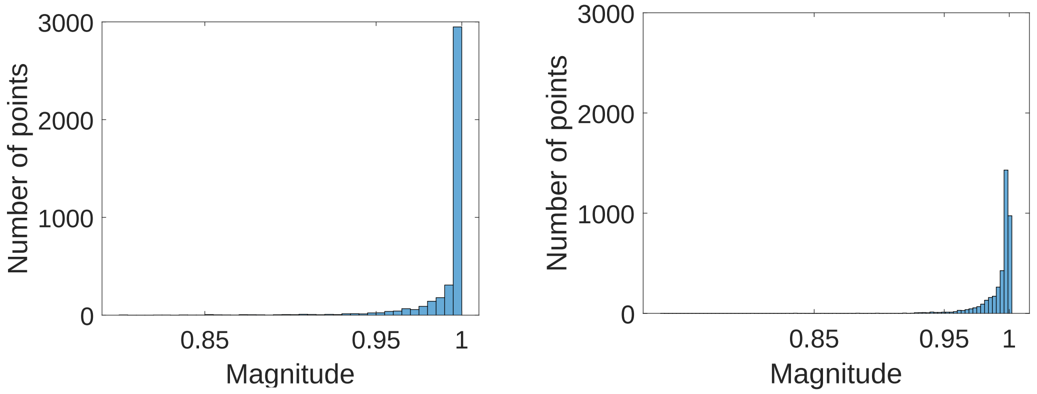


Figure 3.13: Structural analysis on *Raman* data with respect to ϵ.

scenario is that the empirical approximation error will increase depending on the Signal to Noise Ratio.

For the next part of the current section we discuss an empirical rule for tuning K for AE–NN. In order to fingerprint the alteration of the underlying structure we use the Empirical Cumulative Distribution Function (ECDF) introduced in (3.41). The analysis will takes place alongside with Figure 3.14 that demonstrates the histograms of point magnitude in the embedded space. Fundamentally, Figure 3.14 evaluates the two methods in terms of preserving the lower dimensional representation of the dataset. The results indicate that the overall performance of PCA and DCT in terms of ϵ is very similar. DCT demonstrates a slightly better performance in terms of the particular criterion. However, this performance takes place for a higher K compared to PCA. We here tune K such that the two methods have a relatively close performance. By tuning K for DCT to be equivalent to the one for PCA, DCT will demonstrate a lower performance compared to PCA. A closer look at the magnitudes of points provides an intuition why this phenomenon occurs. In particular, nearly 3000 $||\hat{\phi}_i||$'s have a magnitude $||\hat{\phi}_i|| = 1$ that means that these points retain their energy from R^N to R^K (note that $||\phi_i|| = 1$ within the OMP setting). This results to a distance between these points which is identical from R^N to R^K as explained in (3.32). However, for the bulk of points for both methods have a magnitude that lives within the range $[0.95, 1]$ that explains their very similar performance. However, PCA demonstrates this sort of performance in lower dimensions than DCT that can be interpreted as the benefit



(a) Magnitudes of points in R^{210} PCA.

(b) Magnitudes of points in R^{370} DCT

Figure 3.14: Points magnitudes in R^K

of utilizing a data aware operator (i.e. PCA) compared to a data oblivious one (i.e. DCT). Nevertheless, DCT serves the purpose of acceleration as well.

$$\mathrm{ECDF}(\epsilon) = \frac{1}{\binom{|\Phi|}{2}} \sum_{i=1}^{M} \sum_{j \neq i}^{M} \frac{||\boldsymbol{\phi}_i - \boldsymbol{\phi}_j|| - ||\hat{\boldsymbol{\phi}}_i - \hat{\boldsymbol{\phi}}_j||}{||\boldsymbol{\phi}_i - \boldsymbol{\phi}_j||} \leqslant \epsilon. \tag{3.41}$$

In order to provide an intuitive analysis on that aspect of the problem, i.e., the distortion of the underlying structure, we sketch the following example. We consider an input signal y and a dictionary $\hat{\boldsymbol{\Phi}}$ in R^K for $K = 250$ and $K = 150$ accordingly. Let us assume that $NN_{250} = NN_{150} = 1$. Then the associated bound is $d(\boldsymbol{y}, \boldsymbol{\phi}_1)$ in both cases. Nonetheless, the update step of the algorithm will have a different workload. This is due to the alteration on the underlying structure of the dictionary. In particular, due to the fact that the associated operator explicitly shrinks distances, then the points are mutually getting closer from R^{250} to R^{150}. This will eventually lead to an update step where $|S| = N$.

For the next part of this section we are discussing an explicit case that characterises non–isometric embeddings i.e., the pairwise distance may collapse: $||\hat{\boldsymbol{\phi}}_i - \hat{\boldsymbol{\phi}}_j|| \approx 0$. Note that this sort of scenario does not take place at the results demonstrated in Figure 3.13. In cases where $||\hat{\boldsymbol{\phi}}_i - \hat{\boldsymbol{\phi}}_j|| \approx 0$ then $\epsilon \approx 1$ which is not the case for PCA with $K = 210$ or DCT with $K = 370$ either. However, this is a scenario that may occur for lower K's or in other datasets. In order to address this aspect from the perspective of AE–NN we sketch the example demonstrated in Figure 3.16. In particular we are considering $||\hat{\boldsymbol{\phi}}_\mu - \hat{\boldsymbol{\phi}}_5|| \approx 0$, while $||\boldsymbol{\phi}_i - \boldsymbol{\phi}_j|| = ||\hat{\boldsymbol{\phi}}_i - \hat{\boldsymbol{\phi}}_j||$ for the rest of the pairs. In such case any time where $NN_K = 5$ or $NN_K = \mu$ then $5, \mu \in S$. However, the most important parameter for AE–NN is what happens with the vast majority of distances. So, in this particular scenario $|S| = 2$ instead of $|S| = 1$ which is the ideal scenario for AE–NN.

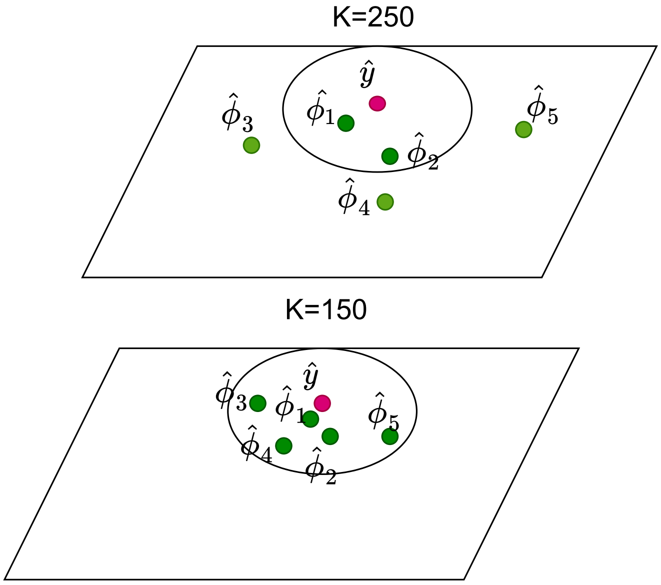


Figure 3.15: Examples Update Step different subspaces

This minor additional cost will not affect the overall performance of the algorithm.

For the next set of simulations, we use a dictionary $\mathbf{\Phi}$ of Swiss Roll data [96] which is a synthetic machine Learning dataset of points that live on a 2–D manifold embedded in R^{1507}. We generate a point cloud of $M = 4041$ points. We repeat the same simulation settings that we used for *Raman*. For AE–NN FNNOMP we set $K = 3$. The obtained results are demonstrated in Figure 3.17.

AE–NN FNNOMP is faster than FNNOMP but with a bigger margin, up to 25 times, this time. Given that the two dictionaries have similar dimensions M, N a natural question that rises is why we pick different K at each time and essentially affect the acceleration of AE–NN FNNOMP over FNNOMP. The main reason why this phenomenon occurs is that the points in *Swiss Roll* dictionary have a much lower dimensional representation compared to the ones in the *Raman* dictionary. In particular, we can embed the *Swiss Roll* dictionary into a lower K without causing a severe deviation to the pairwise distances compared to the one performed for the *Raman* data.

3.9 Summary

In this chapter we introduced novel algorithms for the generic NNS problem. From the perspective of the NNS problem, linear embeddings are renown for introducing fast but approximate solutions to the given problem. Within the current work we laid down the theoretical framework that guarantees the acquisition of the exact NN from

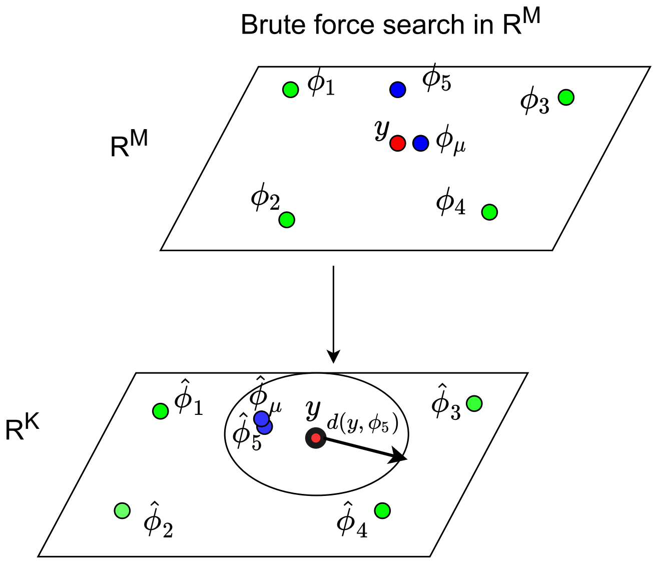


Figure 3.16: Examples where the pairwise distance collapse.

the original space R^M to its subspace R^K. A novel algotithm for the standard NNS problem is introduced as a result of the theoretical analysis.

The NNS problem finds application in several areas such as pattern recognition [65], data mining [66], data compression [67], data mining [113] to name a few. Within the current work we have a particular interest for Sparse Non–Negative signal processing. We incorporated our algorithms in the structure of FNNOMP, that fulfills such task, and we demonstrated the benefits of following such an approach in terms of acceleration. Finally, we demonstrate that our algorithms employs benefits from a particular property that may characterise natural signals, i.e., lower dimensional representation.

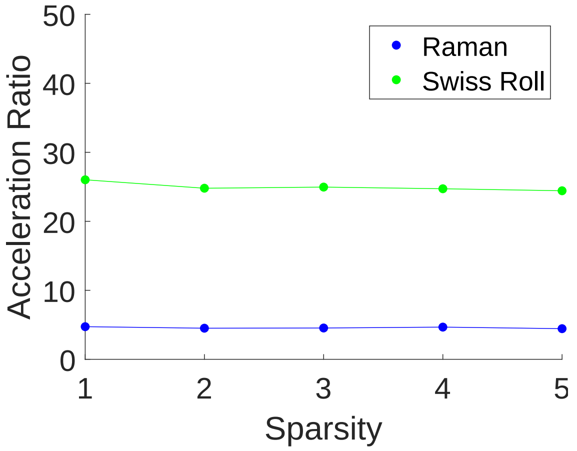


Figure 3.17: Acceleration factor for *Raman* and *Swiss Roll*. The two datasets have identical dimensions $M = 1507$, $N = 4041$. The acceleration factor varies due to the fact that the associated point clouds have a different lower dimensional representation.

Chapter 4

Signal Recovery and Robustness

4.1 Introduction

Within the current Chapter we are aiming to develop a framework that boosts the performance of greedy techniques in terms of signal recovery. Our motivation arises from the success that unfolded techniques have been introduced in the sparse coding regime.

From the perspective of greedy techniques a critical parameter that affects the overall performance is the selection step of the algorithm. The conventional approach that incorporates the standard dictionary $\mathbf{\Phi}$ at the selection step of the algorithm. In such case the recovery performance of the greedy algorithm heavily relies on the underlying structure of the known dictionary, i.e. several points may live close to each other. As a result, a false positive may be introduced in the support set s.

In order to overcome these impracticalities, a potential approach is the design of a selection rule that overcomes the limitations that may occur from the existing approach. A critical parameter when following this sort of direction though, is what is the price we have to pay in terms of computation. The classical approach, regardless of the limitations that may occur in terms of signal recovery, is characterised by its computational efficiency. In that sense a reformulation of the selection rule should take computational efficiency into account as well. Otherwise, the comparison of the new approach with the existing one may introduce a trade off between recovery and computational workload, i.e. the new approach may be more efficient in terms of recovery but more expensive in terms of computational complexity. Therefore, our work is aiming to develop a selection rule that improves the recovery performance of greedy

Algorithm 21 Non-Negative Matching Pursuit algorithm (NNMP)

```
1: initialisation: $s = \varnothing$, $k = 0$ and $\boldsymbol{r}_0 = \boldsymbol{y}$
2: while $k < K$ & $\max(\boldsymbol{\Phi}^T \boldsymbol{r}_k) > 0$
3: $s_k = \mathbf{0}$
4: $(\zeta, \iota) \leftarrow \max(\boldsymbol{\Phi}^T \boldsymbol{r}_k)$
5: $s_k[\iota] = \zeta$
6: $\boldsymbol{r}_{k+1} \leftarrow \mathcal{P}\{\boldsymbol{r}_k - \zeta \phi \iota\}$
7: $k \leftarrow k + 1$
8: end while
9: $x \leftarrow \sum_k s_k$
```

techniques but in the same time maintains the computational workload introduced by the conventional frameworks.

We here introduce DeepMP, a novel sparse decomposition deep neural network. We evaluate the performance of DeepMP compared to the conventional OMP,MP frameworks for sparse non–negative signal processing frameworks via simulations. We lay down the theoretical foundation that guarantees that the network will only recover ground truth atoms over the k layers. We compare our method with state of the art sparse coding frameworks such as LISTA and preconditioning.

4.2 DeepMP

We here introduce Deep Matching Pursuit (DeepMP), a Deep Neural Network for Sparse Non–Negative Signal Processing. The corresponding architecture can be unfolded as a sequence of blocks, where each of the blocks follows the decomposition process of Non–Negative Matching Pursuit (NNMP) introduced in **Algorithm 21**. Starting with the measurement $\boldsymbol{y}$ as the current residual signal $\boldsymbol{r}_k|_{k=0}$, the main steps of NNMP are: a) finding the best matched atom $\boldsymbol{\phi}_k$ to $\boldsymbol{r}_k$, and b) updating the residual $\boldsymbol{r}_k$ by subtracting the contribution of selected atom. The operator $\mathcal{P}$ is the identity matrix.

From the perspective of DeepMP, each of the basic steps of NNMP takes the form of a Neural Network layer. Step 4 of NNMP takes the form of a dense layer that consists of N neurons of size M. By doing that, in practice we replace the conventional dictionary $\boldsymbol{\Phi}$ by a new set of weights $\boldsymbol{W}_f^{(k)}$ that varies over the blocks of DeepMP (we use the index f to distinguish the set of selection step weights from the weights of the update step where we use the index b). On top of the $\boldsymbol{W}_f^{(k)}$ layer we apply the "hard-max" operation as the non–linear activation function, which is the projection onto the best one–sparse set, also known as the 1–sparse hard–thresholding [105]. In

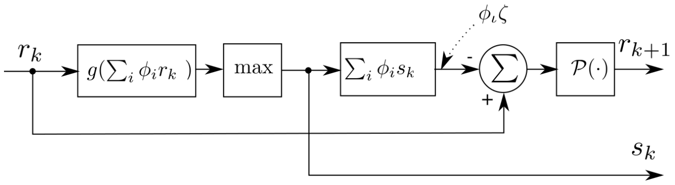


Figure 4.1: One step of non-negative matching pursuit algorithm. r_k and s_k are respectively the residual and the selected index at the k^{th} step of algorithm.

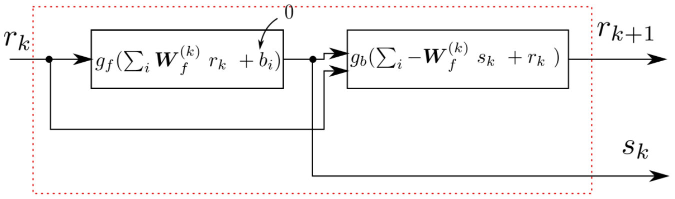

0
r_k
$g_f(\sum_i \boldsymbol{W}_f^{(k)}\ r_k\ +b_i)$
$g_b(\sum_i -\boldsymbol{W}_f^{(k)}\ s_k\ +r_k\ )$
r_{k+1}
s_k

Figure 4.2: DeepMP model: a representation of a single iteration of the NNMP algorithm in the form of a two layer neural network with a skip connection. g_f and g_b are respectively hard–max and linear/ReLU activation functions.

practice the"hard-max" operation replaces the standard "max" operation. Here the term "max" has a dual interpretation: 1) the index of the atom with the highest contribution 2) the corresponding coefficient. The replacement of the "max" operation by the "hard-max" operation is done mainly for implementation purposes and in order to be consistent with the conventional NNMP framework. The "hard-max" operation can be summarised as a special case of the hard thresholding operator $\tau(\boldsymbol{n}, z)$ which sets all the elements in $\boldsymbol{n}$ below z (in terms of mangitude) to zero. In case of the "hard-max" operation $\boldsymbol{n} = \boldsymbol{\Phi}^T y$ and $z = \max\ \boldsymbol{\Phi}^T y$. The conventional NNMP framework recovers a set of indices, while DeepMP returns a sparse vector where the non–zero indices correspond to the atoms that are selected at step 4 of the algorithm. At the update step of DeepMP the standard dictionary $\boldsymbol{\Phi}$ utilised by NNMP is replaced by a new set of weights $\boldsymbol{W}_b^{(k)}$. We then discard the contribution of the selected atom from the available set of measurements $\boldsymbol{r}_k$. Finally, the activation function $\mathcal{P}$ is applied on top of the vector–vector operation performed in step 6. The $\mathcal{P}$ function can be linear (i.e. the identity matrix) or non–linear (i.e. ReLU [106]). The outcome of step 6 is then passed to the next block of DeepMP.

In order to assure that the coefficients obtained at the selection step of the network are always non–negative, there are two approaches we can follow depending on the activation function P we incorporate at step 6 of the algorithm. In particular, in cases where P is linear then some coefficients in $\boldsymbol{r}_k$ may live in the negative orthant. This may lead to a situation where the outcome of the selection step yields a negative coefficient. The "hard-max" function by itself does not guarantee that the dominant coefficient is non–negative. In order to prevent the return of a negative coefficient we apply ReLU on top of "hard-max". Hence in cases where the dominant coefficient is negative, the selection step of DeepMP will yield 0. Note that by following this sort of approach and if we we keep $\boldsymbol{W}_b^k = \boldsymbol{W}_f^k = \boldsymbol{\Phi}$ fixed while training, DeepMP will demonstrate identical results with NNMP. In cases where P is ReLU, then $\boldsymbol{r}_k$ always lives in the positive orthant. However, this is not necessarily the case for the atoms in $\boldsymbol{W}_f^k$. Therefore, we constrain the weights to be non–negative while training, i.e. $\boldsymbol{W}_f^k = |\boldsymbol{D}_f^k|$ (where $\boldsymbol{D}_f^k$ is the set of weights that was originally learnt by DeepMP). From a practical point of view we do not observe a significant difference in between the two approaches in terms of support recovery. A key characteristic of DeepMP, is that the consecutive blocks consist of different set of weights $\boldsymbol{W}_f^k$. This comes in contrast with the conventional NNMP framework where the same set of weights $\boldsymbol{\Phi}$ are utilized over the k iterations.

The overall functionality of the decomposition process for DeepMP can be described by means of a (non–linear) system of equations,

$$\begin{pmatrix} \boldsymbol{r}_{k+1} \\ \boldsymbol{x}_{k+1} \end{pmatrix} = g_b \begin{pmatrix} -\boldsymbol{W}_b^{(k)} g_f(\boldsymbol{W}_f^{(k)^T} \boldsymbol{r}_k) + \boldsymbol{r}_k \\ g_f(\boldsymbol{W}_f^{(k)^T} \boldsymbol{r}_k) + \boldsymbol{x}_k \end{pmatrix}, \tag{4.1}$$

where g_f and g_b are respectively forward and backward functions which are hard-max and $\mathcal{P}$. Such a model can be represented as two layers of a neural network model per single iteration of the algorithm. By concatenation of K blocks of Figure 4.2. The depth of the network is then ruled by the signal k, i.e. the concatenation of k blocks of Figure 4.2 generates k sparse signals. The network takes then the form of a DNN of $2k$ layers. x can then be reconstructed by superposition of s_k.

In the conventional MP frameworks, the successful recovery of s heavily relies on the selection step of the algorithm where we select the atom in $\boldsymbol{\Phi}$ that has the highest contribution in the input signal. The core idea that DeepMP relies on, is the representation of a conventional framework such as NNMP by means of a Deep Neural Network. By fixing the weights of the network to be the same to $\boldsymbol{\Phi}$ over the k layers and setting $\boldsymbol{P}$ to be linear we obtain a structure that has identical performance to NNMP. The main motivation of this type of approach is that by introducing a data adaptive framework such as DeepMP, while training, the deep structure introduces a better performance in terms of support–signal recovery compared to NNMP.

During the training process, DeepMP is provided with a set of artificial mixtures $\boldsymbol{y}$ which are k sparse with respect to $\boldsymbol{\Phi}$. The model yields a sparse set $\mathrm{s}^\star$ which is gradually constructed over the k layers of the model while decomposing $\boldsymbol{y}$. The performance of the model is then evaluated with respect to the fidelity in between s (the ground truth support vector) and $\mathrm{s}^\star$. In that sense, the different sets of weights $\boldsymbol{W}^k$ are learned while optimizing the fidelity in between s and $\mathrm{s}^\star$. At an initial stage we set $\boldsymbol{W}_f^k = \boldsymbol{\Phi}$. This

	DeepMP	NNMP
Sensing step	kMN	MN
Update step	kM	kM
Numerical Operations	kMN+kM	kMN+kM

Table 4.1: Comparison in between DeepMP,MP in terms of numerical operations over k

is done given that the standard dictionary $\mathbf{\Phi}$ is considered as a good starting point and from a practical point of view as the set of weights that yields a known empirical bound in terms of sparse code fidelity. While training we start from this bound, which is considered as the lower empirical bound, and we are aiming to learn sets of weights that improve this bound. On the other hand we keep $\boldsymbol{W}_f^k = \mathbf{\Phi}$ fixed while training given that empirically it demonstrates better performance.

From a more practical point of view, the reformulation of a conventional algorithm in a form a Neural Network releases the degrees of freedom within the decomposition process. In particular, given that a different set of weights is utilised over the k different layers at the selection step of the algorithm, a higher number of parameters is utilised over DeepMP compared to NNMP. The latter is presented in more detail in Table 4.1. In particular, the sensing step of DeepMP utilises k, different dictionaries each of them having MN parameters, over the different layers of DeepMP that results to an overall number of kMN parameters on the sensing step of the network. On the other hand, NNMP utilises the same set of weights $\mathbf{\Phi}$ over the k iterations that results to an overall number of MN parameters during the decomposition process. On the other hand at the update step, the two frameworks consist of a total number of kM parameters. In that sense, even if we utilise a $\boldsymbol{W}_b^k \neq \mathbf{\Phi}$ at the particular step we do not expect that it will result to a marginal improvement in the overall process. Note also that the number of parameters at the particular step is marginally smaller than the one corresponding at the selection step, in that sense the particular step is not expected to have a role as vital as the sensing step. From the perspective of computational complexity though, the two frameworks carry out an identical workload. Even though the dictionaries $\boldsymbol{W}^k$ may differ to $\mathbf{\Phi}$ in terms of weights, they are identical in terms of dimensions. In that sense, the two frameworks conduct the same number of numerical operations.

The main task of DeepMP is to recover the atoms that constitute the input signal $\boldsymbol{y}$. In practice the corresponding architecture performs a multilabel classification task [27]. The input signal $\boldsymbol{y}$ is decomposed with respect to the corresponding classes, while the weights are learned by optimizing the categorical cross–entropy loss function,

$$H(p,q) = -\sum_{j=1}^{|\mathbf{\Phi}|} \mathbf{1}_{s_a}(j) \log p(j,i) \tag{4.2}$$

where i corresponds to the i–th sample, j is the index of atoms and $\mathbf{1}_{s_a} : I \rightarrow \{0,1\}$ is the indicator function, defined as:

$$\mathbf{1}_{\mathrm{s}}(j) = \begin{cases} 1 \text{ if } j \in \mathrm{s} \\ 0 \text{ if } j \notin \mathrm{s} \end{cases}$$

Note that we do not use backpropagation while training.

4.2.1 Sparse Signal Decomposition

The main motivation for introducing an unfolding scheme is that this type of approach increases flexibility approach in the selection of the MP type algorithms. By doing that we are aiming to improve the prediction rate on the support set and eventually reduce the residual error compared to the standard MP framework.

Considering the set of sparse signals which are the main focus of the current work, the main goal of the decomposition algorithm is to identify the atoms which build up the input signal $\boldsymbol{y}$ with non–negative weights as follows:

$$\boldsymbol{y} = \sum_{l=1}^{k} a_w \boldsymbol{\phi}_i. \tag{4.3}$$

with $a_w \sim U[0,1]$, where $U[0,1]$ stands for the uniform distribution with 0 mean and unit variance.

The overall process can then be represented as an iterative algorithm. A common phenomenon that frequently takes place during the decomposition is the selection of unrelated atoms in the support set s_a, over the iterations of the algorithm.

The main reason why this phenomenon occurs, is the similarity between the atoms $\boldsymbol{\phi}_i$. In cases where the algorithm operates over a dictionary where the constituent atoms are highly coherent with each other, the algorithm may introduce a false positive to the support set s. Coherence measures the maximum similarity between two distinctive atoms of $\mathbf{\Phi}$. Given a pair of points $\boldsymbol{\phi}_i, \boldsymbol{\phi}_j \in \mathbf{\Phi}$ where $i \neq j$, the coherence can be formulated as follows:

	M	N	lr	final_lr	epochs
Synthetic data	30	200	$1e-3$	0.1	20
Raman dictionary	503	2521	$1e-3$	0.1	30

Table 4.2: Dimensions of the datasets and training parameters for each of the associated networks.

$$\mu(\mathbf{\Phi}) = \max_{i \neq j} \frac{|\langle \boldsymbol{\phi}_i, \boldsymbol{\phi}_j \rangle|}{||\boldsymbol{\phi}_i||_2 \cdot ||\boldsymbol{\phi}_j||_2} \tag{4.4}$$

where $|| \cdot ||_2$ indicates the Euclidean norm. By introducing the matrix $\boldsymbol{W}_f^{(k)}$ at the selection step of the algorithm, we are aiming for the points to be represented in a way that the mutual coherence of the points will decrease. In that sense by training the network we are expecting that the coherence of the corresponding representation $\boldsymbol{W}_b^{(k)}$ yields an outcome where ideally $\mu(\mathbf{\Phi}) \geqslant \mu(\boldsymbol{W}_f^{(k)})$, where $\mu(\mathbf{\Phi})$ and $\mu(\boldsymbol{W}_f^{(k)})$ are respectively the coherence in $\mathbf{\Phi}$ and $\boldsymbol{W}_f^{(k)}$.

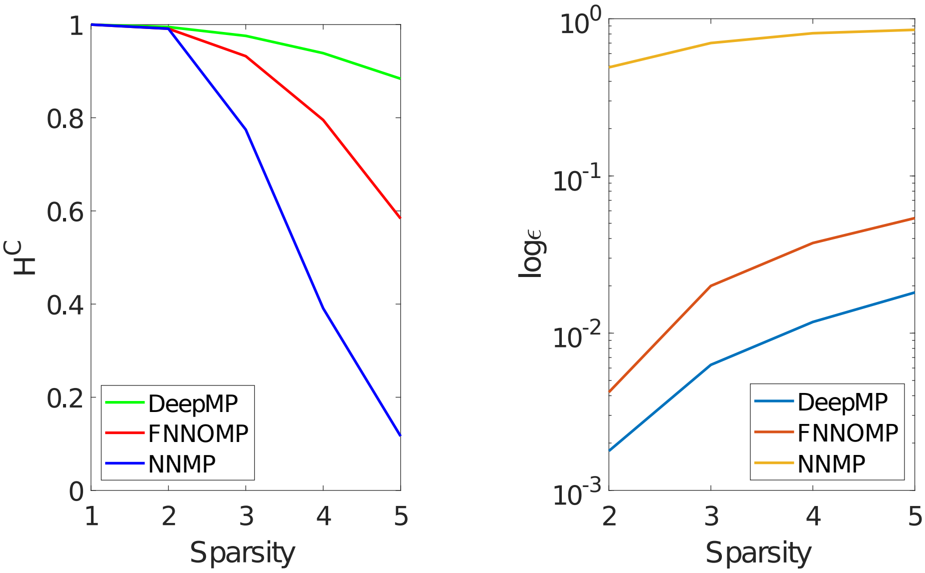
4.3 Experiments

Within the current section we evaluate the performance of DeepMP by some simulations. In order to evaluate the performance of DeepMP we are considering two datasets; a synthetic dataset $\mathbf{\Phi} \in R_+^{d \times N}$. The dictionary was randomly generated with an *i.i.d.* normal distribution and then projected onto the positive orthant and column normalised. A real dataset of *Raman* spectra, where each of the spectras consists of 503 wavenumbers that lay within the range of 306 to 1249 cm^{-1}, provided by [61] . We perform a number of 150000 trials for each dataset where only $\boldsymbol{W}_f^{(k)}$s were trained in the M-space while $\boldsymbol{W}_b^{(k)} = \mathbf{\Phi}$. This essentially means that we only train the weights that correspond to the selection step of the algorithm while the weights that correspond to the update step are kept fixed.

The DeepMP framework is optimized using the *AdaBound* algorithm [107]. More details about the datasets and the settings for the *AdaBound* algorithm can be found in TABLE 4.2.

As an evaluation metric for the exact recovery of the support set we are using the normalized *Hamming distance* complement [108]. The metric is defined as in (4.5).

$$\mathrm{H^C}(s_a, s_g) = \sum_{n=1}^{N} 1 - \frac{|s_a(n) - s_g(n)|}{k} \tag{4.5}$$



(a) Support Recovery

(b) Reconstruction error

Figure 4.3: The performance of different MP frameworks with the *Raman* dictionary.

where s_a is the support set acquired by the corresponding algorithm and s_g the ground truth, k is the sparsity level and $|\cdot|$ the cardinality operator. The performance on the reconstruction error for each sparsity k is evaluated with respect to ϵ as follows:

$$\epsilon(k) = \frac{1}{Z}\sum_{z=1}^{Z} \frac{||\boldsymbol{y}[z] - \boldsymbol{\Phi}\boldsymbol{x}[z]||_2}{||\boldsymbol{y}[z]||_2}, \tag{4.6}$$

where Z is the number of realizations.

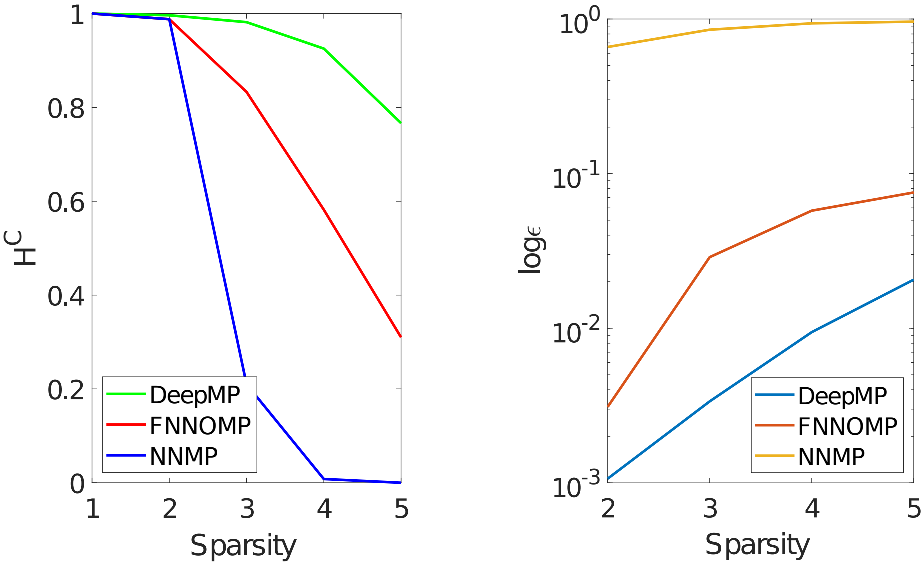
A basic expectation while training the selection step of the algorithm is the variation of the coherence between $\boldsymbol{w}_i^k, \boldsymbol{w}_j^k$ columns of $\boldsymbol{W}_b^{(k)}$. For the particular aspect of the problem we use the empirical cumulative distributed function (ECDF) as introduced in equation (4.7):

$$\mu_{\mathrm{ECDF}}(t) = \frac{1}{\binom{|\Phi|}{2}} \sum_{i=1}^{M} \sum_{j\neq i}^{M} \mu(\boldsymbol{\phi}_i, \boldsymbol{\phi}_j) \leqslant t. \tag{4.7}$$

where $t \in [0, 1]$.

4.3.1 Results

We here perform a simulation based evaluation of the different MP frameworks. We are particularly interested in signals which are very sparse. Hence we consider mixtures



(a) Support Recovery

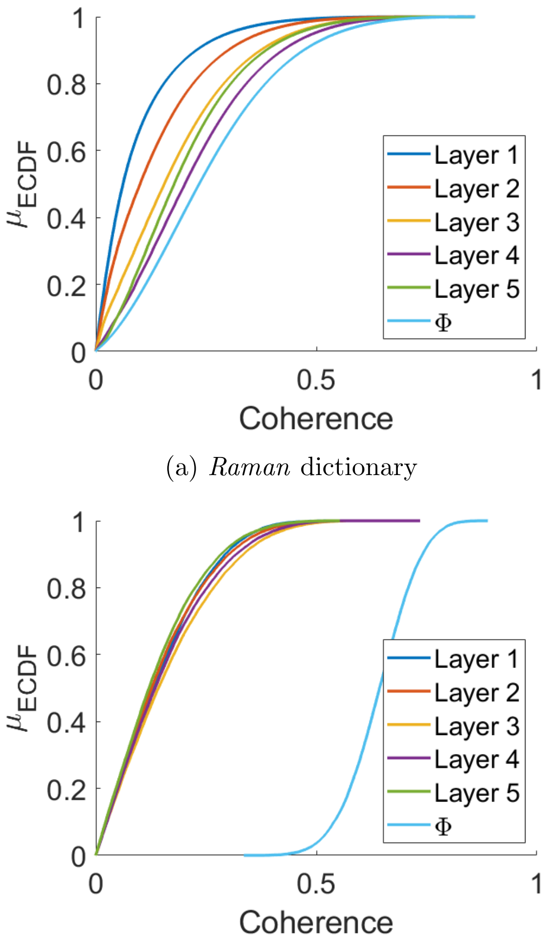
(b) Reconstruction error

Figure 4.4: The figure demonstrates the performance of the different MP frameworks for the Synthetic dictionary.

of signals $\boldsymbol{y}$ that consist of up to 5 atoms. From the perspective of the DeepMP framework this corresponds to concatenation of up to 5 different versions of the model with a varying depth over sparsity. These versions are independent, i.e. the 1st layer is different from the one model to the other. The obtained results for the *Raman* data and the synthetic data are demonstrated in Figures 4.3 and 4.4 accordingly.

As it can be seen from the results, DeepMP outperforms the NNMP and FNNOMP [10] with respect to the Hamming distance complement, while FNNOMP significantly outperforms NNMP. This essentially means that the higher number of parameters at the selection step of DeepMP introduces a more flexible approach which leads to a better exact recovery performance. The advantage of DeepMP becomes more significant over sparsity having less sparse signals, i.e. larger k. Hence, despite the fact that the performance of all the MP frameworks decays with K, the flexibility of DeepMP leads to a slower decay over sparsity and hence getting a better performance compared to FNNOMP and NNMP. The ϵ–error results also indicate that the improved exact recovery, leads to a better performance on the reconstruction of the input signal $\boldsymbol{y}$.

Despite demonstrated good results using DeepMP, a question is why it outperforms NNMP and FNNOMP. While a rigorous answer to this question is left for the future, we demonstrate the μ_{ECDF}'s of the two dictionaries and the trained model trained for $k = 5$ in figure 4.5. As it can be seen from the results, the network generates weight matrices with reduced coherences compared to the original ones. In that sense, the



(a) *Raman* dictionary

(b) Synthetic dictionary

Figure 4.5: The coherence of the Raman dictionary and the synthetic dictionary in comparison with the coherence of the weight matrices of DeepMP for $k = 5$.

corresponding point clouds consist of a set of atoms which are further apart the one to the other. This essentially means that DeepMP alters the underlying geometry of the selection step to avoid a misclassification. Given that the points are further apart, i.e. smaller coherence in average, the algorithm can easier pick the right atom without confusing it with its neighbors.

4.4 DeepMP and Exact Recovery

From a practical point of view, the outcome of DeepMP is that we replace $\mathbf{\Phi}$ by a different representation matrix. This sort of modification has similarities with the so–called preconditioning framework introduced in [29]. A key characteristic of this approach is that the MP functionality is split into two main steps: 1) sensing step (step 4

	DeepMP	Preconditioning
Sensing step	kMN	MN
Update step	kM	kM
Numerical Operations	kMN+kM	kMN+kM

Table 4.3: Comparison in between the Preconditioning framework and DeepMP in terms of computational cost.

in NNMP) 2) reconstruction step (step 6 in NNMP). The main scope of preconditioning, is the design of a sensing matrix $\boldsymbol{\Psi}$ that improves the performance of the MP framework in terms of support recovery compared to $\boldsymbol{\Phi}$. Despite the practical similarities between DeepMP and the preconditioning framework, there are two basic differences between the two approaches.

The first difference in between DeepMP and preconditioning is in the process upon which the representation matrices are acquired. From the perspective of DeepMP the model is trained to reconstruct artificial signals with respect to the exact recovery of the support set as introduced in (4.2). We thus indirectly enforce the network to construct layouts of representation matrices $\boldsymbol{W}^k$ with lower cumulative coherence. The latter will be more analytically demonstrated at Section 4.7.

On the other hand, the preconditioning framework finds the representation matrix $\boldsymbol{\Psi}$ by solving the optimization problem introduced in (4.8).

$$\min ||\boldsymbol{G} - \boldsymbol{H}||_F, \tag{4.8}$$

where $\boldsymbol{G} = \boldsymbol{\Psi}^{\boldsymbol{T}}\boldsymbol{\Phi}$ and $\boldsymbol{H}$ a $N \times N$ matrix with $\boldsymbol{H}_{ii} = 1$ and $|\boldsymbol{H}_{ij}| \leqslant \mu$ for $i \neq j$. By following this type of approach, the main goal is the construction of a representation matrix $\boldsymbol{\Psi}$, for the dictionary $\boldsymbol{\Phi}$, that gives low cumulative coherence and eventually leads to a better performance in terms of support set reconstruction.

The second difference is related to the flexibility of each approach. In [29], $\boldsymbol{\Phi}$ is replaced by a single set of weights $\boldsymbol{\Psi}$ which is fixed over the iteration k. In contrast, DeepMP consists of a sequence of matrices $\boldsymbol{W}^k$'s that varies over k providing a higher degree of flexibility. In summary, the preconditioning framework incorporates an overall number of parameters MN in the decomposition process while in case of DeepMP the number of parameters is kMN as expressed in section 4.3.

Inspired from the theoretical results of [29] and considering a number of iterations k (number of layers in DeepMP), the following theorem states the conditions that a correct atom is selected at each of the layers of DeepMP. Note that within our analysis we introduce some sort of abuse in the standard terminology of the mutual coherence term, as defined in (4.9), given that we use these terms for non–normalized $\boldsymbol{W}^k$'s. The minimal coherence term introduced in (4.10) corresponds to the similarity in between a dictionary element $\boldsymbol{\phi}_i$ and its corresponding representation in $\boldsymbol{W}^k$.

$$\bar{\mu}_w(\boldsymbol{\Phi}, \boldsymbol{W}^k, K) = \max_{|s|=K} \max_{i \notin s} \sum_{j \in s} |\langle \boldsymbol{W}_i^k, \boldsymbol{\phi}_j \rangle| \tag{4.9}$$

$$\beta(\boldsymbol{\Phi}, \boldsymbol{W}^k) = \min_i |\langle \boldsymbol{W}_i^k, \boldsymbol{\phi}_i \rangle| \tag{4.10}$$

Theorem 1. Let $\boldsymbol{y}$ be an exactly K–sparse signal in $\boldsymbol{\Phi}$, with support s, i.e. $\boldsymbol{y} = \sum_{i \in s} x_i \boldsymbol{\phi}_i$. Deep Matching Pursuit with linear P using representation matrices $\boldsymbol{W}^k$'s, selects components of the true support s, and its complement $\bar{s}$ if for every K we have,

$$||(\boldsymbol{\Phi}_s^T \boldsymbol{W}_s^k)^{-1} \boldsymbol{\Phi}_s^T \boldsymbol{W}_{\bar{s}}^k)||_{1,1} < 1, \tag{4.11}$$

where in general $||\boldsymbol{A}||_{p,q} = \sum_{j=1}^N (\sum_{i=1}^M |a_{ij}|)$. (4.11) is always satisfied if,

$$\bar{\mu}_w(\boldsymbol{\Phi}, \boldsymbol{W}^k, K) + \bar{\mu}_w(\boldsymbol{\Phi}, \boldsymbol{W}^k, K-1) < \beta(\boldsymbol{\Phi}, \boldsymbol{W}^k) \tag{4.12}$$

From (4.12) it follows:

$$\begin{aligned} &\bar{\mu}_w(\boldsymbol{\Phi}, \boldsymbol{W}^k, K-1) < \beta(\boldsymbol{\Phi}, \boldsymbol{W}^k) => \\ &=> \bar{\mu}_w(\boldsymbol{\Phi}, \boldsymbol{W}^k, K-1) - \beta(\boldsymbol{\Phi}, \boldsymbol{W}^k) < 0 \end{aligned} \tag{4.13}$$

Proof. Given an input signal $\boldsymbol{y} = \sum_{i \in s} x_i \boldsymbol{\phi}_i$ and under the assumption that correct atoms have been selected up to layer k, then the residual $\boldsymbol{r}_k$ is still a linear combination of the atoms in the true support:

$$\boldsymbol{r}_k = \boldsymbol{y} - \sum_{j \in s_k} b_j \boldsymbol{\phi}_j = \sum_{i \in s} x_i \boldsymbol{\phi}_i - \sum_{j \in s_k} b_j \boldsymbol{\phi}_j = \sum_{i \in s} c_i \boldsymbol{\phi}_i = \boldsymbol{\Phi}_s \boldsymbol{c}, \tag{4.14}$$

where $s_k \subset s$.

DeepMP selects a correct atom at the next layer, if the maximal contribution of an atom in the support s: $\max_{i \in s} |\langle \boldsymbol{W}_i^k, \boldsymbol{r}_k \rangle|$ is larger than the maximal contribution of an atom in the complement support $\bar{s}$, $\max_{j \in \bar{s}} |\langle \boldsymbol{W}_j^k, \boldsymbol{r}_k \rangle|$. Hence, we have to make sure that the following inequality holds:

$$\frac{\max_{j \in \bar{s}} |\langle \boldsymbol{W}_j^k, \boldsymbol{r}_k \rangle|}{\max_{i \in s} |\langle \boldsymbol{W}_i^k, \boldsymbol{r}_k \rangle|} = \frac{||(\boldsymbol{W}_{\bar{s}}^k)^T \boldsymbol{r}_k||_\infty}{||(\boldsymbol{W}_s^k)^T \boldsymbol{r}_k||_\infty} < 1 \tag{4.15}$$

We can use now the p, q–matrix norms for $1 \leqslant p, q \leqslant \infty$, defined as $||\boldsymbol{C}||_{p,q} = \max_{||\boldsymbol{x}||_p=1}||\boldsymbol{x}||_q$. Inserting $\boldsymbol{r} = \boldsymbol{\Phi}_s\boldsymbol{c}$ into (4.15) and under the assumption that $(\boldsymbol{W}_s^k)^T\boldsymbol{\Phi}_s$ is invertible , where $\boldsymbol{z} = (\boldsymbol{W}_s^k)^T\boldsymbol{\Phi}_s\boldsymbol{c}$, we can bound (4.15) as follows:

$$\begin{aligned}\frac{||(\boldsymbol{W}_{\bar{s}}^k)^T\boldsymbol{\Phi}_s\boldsymbol{c}||_\infty}{||(\boldsymbol{W}_s^k)^T\boldsymbol{\Phi}_s\boldsymbol{c}||_\infty} &= \frac{||(\boldsymbol{W}_{\bar{s}}^k)^T\boldsymbol{\Phi}_s((\boldsymbol{W}_s^k)^T\boldsymbol{\Phi}_s)^{-1}\boldsymbol{z}||_\infty}{||\boldsymbol{z}||_\infty}, \\ &\leqslant ||(\boldsymbol{W}_{\bar{s}}^k)^T\boldsymbol{\Phi}_s((\boldsymbol{W}_s^k)^T\boldsymbol{\Phi}_s)^{-1}||_{\infty,\infty},\end{aligned} \tag{4.16}$$

If we use the following duality norm relationship $||(\boldsymbol{W}_{\bar{s}}^k)^T\boldsymbol{\Phi}_s((\boldsymbol{W}_s^k)^T\boldsymbol{\Phi}_s)^{-1}||_{\infty,\infty} = ||(\boldsymbol{\Phi}_s^T\boldsymbol{W}_s^k)^{-1}\boldsymbol{\Phi}_s^T\boldsymbol{W}_{\bar{s}}^k||_{1,1}$ we show that (4.11) is sufficient condition for exact recovery.

In the second part of the proof, we show that (4.12) implies (4.11). The first step is to use the subaddition property of the norm as follows:

$$||(\boldsymbol{\Phi}_s^T\boldsymbol{W}_s^k)^{-1}\boldsymbol{\Phi}_s^T\boldsymbol{W}_{\bar{s}}^k||_{1,1} \leqslant ||(\boldsymbol{\Phi}_s^T\boldsymbol{W}_s^k)^{-1}||_{1,1}||\boldsymbol{\Phi}_s^T\boldsymbol{W}_{\bar{s}}^k||_{1,1} \tag{4.17}$$

The second term in the right hand side of (4.17) can be bounded by the mutual–coherence term as follows:

$$||\boldsymbol{\Phi}_s^T\boldsymbol{W}_{\bar{s}}^k||_{1,1} = \max_{j\in\bar{s}}\sum_{i\in s}|\langle\boldsymbol{W}_j^k, \boldsymbol{\phi}_i\rangle| \leqslant \bar{\mu}_w(\boldsymbol{\Phi}, \boldsymbol{W}^k, K), \tag{4.18}$$

where given a matrix $\boldsymbol{\Phi} \in R^{M\times N}$, the $l_{1,1}$ norm of $\boldsymbol{\Phi}$ is defined as follows:

$$||\Phi||_{1,1} = \sum_{j=1}^{M}\sum_{i=1}^{N}|a_{i,j}| \tag{4.19}$$

In order to find an upper bound for the first term in (4.17), we employ the property that in cases where $||\boldsymbol{C}||_{1,1} < 1$ then $||\boldsymbol{I}_K + \boldsymbol{C}||_{1,1} < (1 - ||\boldsymbol{C}||_{1,1})^{-1}$[109]. Set $\boldsymbol{C} = \boldsymbol{\Phi}_s^T\boldsymbol{W}_s^k - \boldsymbol{I}_K$,where $\boldsymbol{I}_K$ the identity matrix in R^K, then it follows:

$$\begin{aligned}||\boldsymbol{C}||_{1,1} &= \max_{i\in I}(|\langle\boldsymbol{W}_i^k, \boldsymbol{\phi}_i\rangle - 1| + \sum_{j\neq i}|\langle\boldsymbol{W}_j^k, \boldsymbol{\phi}_i\rangle|) \\ &\leqslant 1 - \beta(\boldsymbol{\Phi}, \boldsymbol{W}^k) + \bar{\mu}_w(\boldsymbol{\Phi}, \boldsymbol{W}^k, K-1) \overset{(4.13)}{<} 1,\end{aligned} \tag{4.20}$$

that results to the following:

$$\begin{aligned}||(\boldsymbol{\Phi}_s^T\boldsymbol{W}_s^k)^{-1}||_{1,1} &\leqslant (1 - (1 - \beta(\boldsymbol{\Phi}, \boldsymbol{W}^k) + \bar{\mu}_w(\boldsymbol{\Phi}, \boldsymbol{W}^k, K-1)))^{-1} \\ &\leqslant (\beta(\boldsymbol{\Phi}, \boldsymbol{W}^k) - \bar{\mu}_w(\boldsymbol{\Phi}, \boldsymbol{W}^k, K-1))^{-1}\end{aligned} \tag{4.21}$$

By combination of (4.18),(4.21) we get the desired result:

$$||(\mathbf{\Phi}_s^T \boldsymbol{W}_s^k)^{-1}\mathbf{\Phi}_s^T \boldsymbol{W}_{\bar{s}}^{\boldsymbol{k}}||_{1,1} \leqslant \frac{\bar{\mu}_w(\mathbf{\Phi}, \boldsymbol{W}^k, K)}{\beta - \bar{\mu}_w(\mathbf{\Phi}, \boldsymbol{W}^k, K-1)} < 1 \tag{4.22}$$

□

The outcome of the theoretical analysis fingerprints the conditions which are necessary in order to recover only correct atoms over the k layers of DeepMP. Typically greedy methods aim to obtain fast but approximate solutions. However, exact support recovery is of critical importance in several applications, i.e. Beamforming with multiple targets where the task is the detection of location of each target [110], DNA microanalysis [111], Raman spectroscopy [112] where i.e. the task is the detection of hazardous materials that may be present in a chemical mixture etc. The bound introduced in (4.22), is not incorporated while training DeepMP. This sort of approach could be a future direction for the current work. However, by evaluating the cross coherence function on W^k's after training, then for all the k's that comply with (4.22), we can be certain that the support will be recovered. The latter turns the process of testing the performance of DeepMP in terms of support recovery into a redundant process for this level of k. Essentially, this is one of the key advantages of model based learnt structures compared to ad hoc structures. The common practice for Deep Neural Networks is that they are treated as black box machines that deliver outstanding results. From the perspective of DeepMP, such an outstanding result, i.e. the full recovery of the support, takes place because the weights of the network align with (4.22).

4.5 Practical Considerations for DeepMP

The selection of a correct atom does not necessarily mean that within the k iterations we may acquire k different atoms. This is due to the fact that a previously selected atom may be reselected i.e. $|s| < k$ after k iterations. The latter holds for the preconditioning framework in general, i.e. for both MP and OMP. This is not the case i.e. for the standard OMP framework where the residual vector r_k is always orthogonal to a previously selected atom i.e. $\langle \boldsymbol{\phi}_i, \boldsymbol{r}_k \rangle = 0, \forall i \in \mathrm{s}$. Due to this property a new iteration of OMP always comes up with a new atom in s.

In order to assure that DeepMP always selects a new atom per each layer we introduce a modification on the standard framework for MP which is introduced in **Algorithm 20**. In particular we set the value of the inner product in between the

Algorithm 22 Matching Pursuit with Support Manipulation (MPSM).

```
initialisation: $sup = \emptyset$, $k = 0$ and $\boldsymbol{r}_0 = y$
while$k < K$.
    $s_k = \boldsymbol{0}$
    $S = |\boldsymbol{\Phi}^T \boldsymbol{r}_k|$
    $S_{sup} = 0$.
    $(\zeta, \iota) \leftarrow \max S$
    $s_k[\iota] = \zeta$
    $sup = sup \cup \iota$
    $\boldsymbol{r}_{k+1} \leftarrow \boldsymbol{r}_k - \zeta \phi_\iota$
    $k \leftarrow k + 1$
end while
$x \leftarrow \sum_k s_k$
```

current residual $\boldsymbol{r}_k$ and the previously selected atoms to 0. This modification can be incorporated by the standard preconditioning framework for OMP as well. The main scope of this work is to learn a better set of weights $\boldsymbol{W}_f^k$ compared to $\boldsymbol{\Phi}$. In practice, we do not expect that this type of modification will have a significant impact on training.

4.5.1 DeepMP and Exact Signal Recovery

Addressing the multilabel classification task from a Sparse Signal Processing point of view, this is related with the identification of the correct atoms that may contribute within a mixture. The main goal from a Sparse Signal Processing point of view though, is to acquire the exact coefficients in order to fully recover the input signal. From the perspective of the standard MP framework though , and consequently for DeepMP, the exact recovery of s does not guarantee the exact recovery of $\boldsymbol{x}_s$.

In order to assure that DeepMP recovers $\boldsymbol{x}_s$ exactly, any time s is reconstructed exactly as well, we introduce an additional step to the framework. A renown approach for exact signal recovery, given that the atoms are identified correctly, is to apply the pseudoinverse $\boldsymbol{\Phi}^\dagger = (\boldsymbol{\Phi}_{\boldsymbol{s}}^{\boldsymbol{T}} \boldsymbol{\Phi}_{\boldsymbol{s}})^{-1} \boldsymbol{\Phi}_{\boldsymbol{s}}^T$. It is well known that the pdeudoinverse reconstructs in input signal exactly if the exact support set s is provided. The latter can be demonstrated as follows:

$$\boldsymbol{y} = \boldsymbol{\Phi}_s \boldsymbol{x}_s => \boldsymbol{\Phi}_s^T \boldsymbol{y} = \boldsymbol{\Phi}_s^T \boldsymbol{\Phi}_s \boldsymbol{x}_s => (\boldsymbol{\Phi}_s^T \boldsymbol{\Phi}_s)^{-1} \boldsymbol{\Phi}_s^T \boldsymbol{y} = \boldsymbol{x}_s. \tag{4.23}$$

Note that the output introduced in (4.23) holds, as long as $\boldsymbol{\phi}_i, \boldsymbol{\phi}_j$ are linearly independent $\forall i, j \in$ s.

Therefore, we introduce a modified version for DeepMP in **Algorithm 23** which incorporates the pseudoinverse as a part of the process in order to guarantee the exact

Algorithm 23 Deep Matching Pursuit algorithm (NN DeepMP)+Pseudoinverse

```
initialisation: $s = \emptyset$, $k = 0$ and $\boldsymbol{r}_0 = y, x = \mathbf{0}$
while $k < K$ & $\max(\boldsymbol{\Phi}^T \boldsymbol{r}_k) > 0$
    $s_k = \mathbf{0}$
    $S = \boldsymbol{\Phi}^T \boldsymbol{r}_k$
    $S_{sup} = 0.$
    $(\zeta, \iota) \leftarrow \max S$
    $s_k[\iota] = \zeta$
    $s = s \cup \zeta$
    $\boldsymbol{r}_{k+1} \leftarrow \mathcal{P}\{\boldsymbol{r}_k - \zeta \phi_\iota\}$
    $k \leftarrow k + 1$
end while
$x_s \leftarrow (\boldsymbol{\Phi}_s^T \boldsymbol{\Phi}_s)^{-1} \boldsymbol{\Phi}_s^T \boldsymbol{y}$
```

acquisition of $\boldsymbol{x}_s$ when s is fully recovered after k iterations. In practice by introducing the update step on DeepMP we update the coefficients acquired over the layers of the network. The latter can be incorporated at the general MP framework as well.

4.6 Comparison LISTA–DeepMP

The essential need for algorithms that perform well in terms of signal reconstruction but, in the same time, they are computationally appealing was the main motivation of the authors to propose the Learned Iterative Soft Thresholding (LISTA) framework in [90]. The main characteristic of LISTA is that for the same number of iterations it outperforms ISTA in terms of signal reconstruction as defined in (4.24).

$$\epsilon = ||\boldsymbol{x} - \boldsymbol{x}^\star||_2 \tag{4.24}$$

, where $\boldsymbol{x}$ the ground truth vector and $\boldsymbol{x}^\star$ the approximation obtained by the algorithm. In practice, each iteration of ISTA corresponds to a layer of LISTA that consists of two blocks W and S which are learned via training. In order to draw the connection in between ISTA and LISTA, W corresponds to $\frac{1}{L}\Phi$ while S corresponds to $I - \frac{1}{L}\Phi^T\dot{\Phi}$, where I the identity matrix. Another key difference is that the non–linear function $\eta_{\lambda/L}$ is represented as: $\eta_\theta = \text{sign}(x)\max(0, |x| - \theta)$, where θ is a vector, where $\theta_i = \frac{\alpha}{L}$. α a parameter which is manually tuned while L is the maximum eigenvalue. A key

Algorithm 24 Iterative Shrinkage Thresholding Algorithm.

```
initialisation: $x = \vec{0}, L >$ largest eigenvalue of $\Phi^T\Phi$
while$k <$ some fixed threshold
    $x^{k+1} = \text{soft}(z, \theta)(x^k + \frac{1}{L}\Phi^T(y - \Phi x^k))$
end while
$x \leftarrow \sum_k s_k$
```

difference of LISTA compared to ISTA is that θ^k(note that k for the LISTA notation corresponds to a layer of the NN and not in the level of sparsity as it is commonly used in the sparse signal decomposition framework) is learned over the layers instead of being manually tuned. Similarly to DeepMP the weight matrix Φ is also replaced by learned weights W. The authors in [91] introduced a bound over the residual error for a sparse signal. In particular the error is bounded as follows:

$$||\boldsymbol{x}^k - \boldsymbol{x}||_2 \leqslant pBexp(-ck), \tag{4.25}$$

where $B > 0$, $c > 0$ and $p = ||x||_0$. Note that k in this framework corresponds to the number of layers and not in the level of sparsity which is the trivially used notation in the sparse signal decomposition regime. Essentially the bound states, that the recovery error converges to 0 as the number of layer goes to infinity. In practice, this means that in cases where exact signal recovery is the preliminary goal, the associated computational cost would be intractable given that the latter requires an architecture with an infinite number of layers (or an infinite number of iterations from the perspective of an iterative scheme). It is also worth mentioning that the bound introduced in (4.25) holds under the following constraint:

$$k < \frac{(1 + \frac{1}{\tilde{\mu}})}{2} \tag{4.26}$$

where k corresponds to the number of non–zero coefficients (level of sparsity in the signal) and $\tilde{\mu}$ is defined as in (4.27):

$$\tilde{\mu} = \max_{i \neq j} |\tilde{W_i}^T \Phi_j| \tag{4.27}$$

, where W the set of training weights.

Note that for a similar type of bound where,

$$s < \frac{(1 + \frac{1}{\mu})}{2} \tag{4.28}$$

, with

$$\mu = \max_{i \neq j} |\Phi_i^T \Phi_j| \tag{4.29}$$

introduced in 4.26 the author in [45, pp.2] has proven that OMP recovers x in exactly $|s|$ steps. The latter corresponds to a finite number of steps and consequently to a computationally tractable scheme. In that sense, implementing OMP for signals following the bound introduced in 4.26 is a computationally efficient approach assuming in cases where the exact recovery of a signal is the main goal. Nevertheless, it is worth mentioning that the author in [45] introduced a more relaxed bound with respect to the Babel function (4.30) which can be summarised as in (4.31).

$$\mu_1(s, \Phi) = \max_i \max_{|\boldsymbol{s}|=K, i \notin \mathbf{s}} \sum_{j \in \mathbf{s}} |\langle \phi_i, \phi_j \rangle| \tag{4.30}$$

, where s the support set.

$$\mu_1(s) + \mu_1(s-1) < 1 \tag{4.31}$$

In order to familiarise the reader and explain why we call the bound introduced in (4.31) more relaxed than the one in 4.26, we will first rewrite the bound in 4.26 as follows:

$$s < \frac{1}{2}(\frac{1}{\mu} + 1) => 2s < \frac{1}{\mu} + 1 => 2s\mu < 1 + \mu => 2s\mu - \mu < 1 \tag{4.32}$$

The Babel function introduced in (4.30) can be upper bounded as follows:

$$\mu_1(s) \leqslant s\mu \tag{4.33}$$

, and therefore

$$\mu_1(s-1) \leqslant (s-1)\mu \tag{4.34}$$

$$\overset{(4.33)+(4.34)}{=>} \quad \mu_1(s) + \mu_1(s-1) \leqslant s\mu + (s-1)\mu = 2s\mu - \mu < 1 \tag{4.35}$$

Hence, the bound introduced by the Babel function as introduced in (4.31) is smaller than the bound introduced in (4.28) and with respect to the reformulation introduced in (4.32).

By following the guidelines introduced by the author in [45] we can relax the bound for LISTA with respect to the Babel function as follows:

$$\bar{\mu}_w(s) + \bar{\mu}_w(s-1) < \beta_w \tag{4.36}$$

$$\bar{\mu}_w(s, \boldsymbol{\Phi}, \boldsymbol{W}^k) = \max_i \max_{|I|=s, i \notin I} \sum_{j \in I} |\langle \boldsymbol{W}_i^k, \boldsymbol{\phi}_j \rangle| \tag{4.37}$$

$$\beta_w(\boldsymbol{\Phi}, \boldsymbol{W}^k) = \min_i |\langle \boldsymbol{W}_i^k, \boldsymbol{\phi}_i \rangle| \tag{4.38}$$

Note though that the authors in [91] consider that $\boldsymbol{W}_i^T \boldsymbol{\Phi}_i = 1$, hence $\beta_w = 1$, and consequently the equation introduced in 4.12 can be rewritten as follows:

$$\bar{\mu}_w(s) + \bar{\mu}_w(s-1) < 1 \tag{4.39}$$

By following the same methodology with the bounds for OMP, we can show that:

$$\bar{\mu}_w(s) + \bar{\mu}_w(s-1) \leqslant 2s\tilde{\mu} - \tilde{\mu} < 1 \tag{4.40}$$

The relaxation on the bound corresponds to the structural relationship in between the weights of LISTA which is evaluated with respect to coherence. Nevertheless, the fact that LISTA needs and infinite number of layers, i.e., iterations, to recover these types of signals exactly remains the same. In contrast, from a theoretical point of view DeepMP+Pseudoinverse guarantees the acquisition of the exact signal within k layers (iterations) where k is a finite number.

From a practical point of view, DeepMP and LISTA share a common characteristic: They are model based Deep NN's, i.e. the corresponding structure is a reformulation of an existing algorithm, which comes on contrast with the commonly followed practise in Deep NN's where the models are constructed on an ad hoc basis. However, the two models are fundamentally different in the sense they are aiming to solve a different optimization problem which is (2.30) and (2.23) for DeepMP and LISTA accordingly. From a training perspective key difference of the two approaches is that LISTA learns a set of parameters $\boldsymbol{\theta}$ except from the weight matrices. This is not the case for DeepMP though. However, we can develop a customized cost function, i.e.: $|y - \Phi x| + \lambda |x|_1$, where $\lambda > 0$ could be a constant learned via training. A results based practical comparison in between the two models is provided in section 4.7.

4.6.1 Computational Complexity LISTA–DeepMP

Within the current section we will discuss about the computational complexity of the standard version of ISTA and MP. Note that the main motivation for unfolding a Neural Network (NN) in the form of these algorithms, is to boost their performance in terms of signal recovery while maintaining their appealing computational complexity. In that sense, the complexity analysis holds for both ISTA and MP , regardless their formulation (conventional or NN). Given a dictionary $\Phi \in R^{M \times N}$, the computational

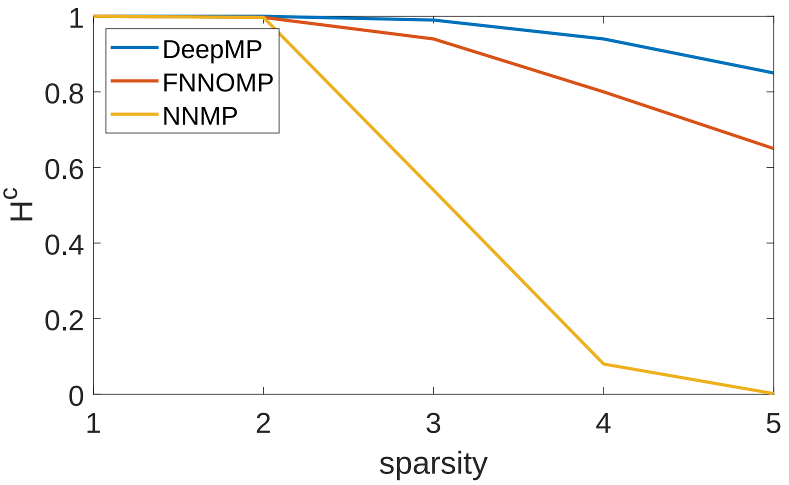


Figure 4.6: Comparison with respect to H^c synthetic data, $N = 40$.

complexity of MP for a single iteration , with respect to the selection step (step 4 in (Algorithm 1)) and the update step (step 7 in **Algorithm 2**) , can be summarised as follows: $k(MN + M) = kM(N + 1) \approx MN$(for a very large N). The complexity of the algorithm scales linearly over the number of iterations k. From the perspective of LISTA the computational complexity can be summarised as follows : $k2MN+MN$. By evaluating the two algorithms one by one in terms in terms of computational complexity and assuming that they perform the same number of iterations k, we can obtain the following outcome:

$$\begin{aligned}\text{Acceleration} &= \frac{\text{Numerical operations LISTA}}{\text{Numerical operations MP}}\\ &= \lim_{k\to\infty} \frac{MN + 2k \cdot MN}{kMN} \\ &= \lim_{k\to\infty} \frac{2k \cdot MN}{k \cdot MN} + \frac{MN}{k \cdot MN} = \lim_{k\to\infty} 2 + \frac{1}{k} = 2\end{aligned} \tag{4.41}$$

The outcome of (4.41) simply states that ISTA, and consequently LISTA, is 2 times slower than (Deep) MP. Note that the analysis is somewhat general and does not consider the case of having a structured dictionary that may help us having a fast dictionary–coefficient multiplication.

4.7 Results

We perform a simulation based comparison between different frameworks for sparse decomposition with the non–negativity setting. The evaluation takes place with respect to three datasets: 1) A set of synthetic data: We generate a random dictionary $\mathbf{\Phi} \in R^{M\times N}$ with $M = 30$. For the particular dataset we are considering two different cases for N: a) $N = 40$, b) $N = 200$. We then project $\mathbf{\Phi}$ on the positive orthant, then the unitary ball. This particular dataset will be used as the baseline dataset for the remainder of the current section, unless it is explicitly mentioned. 2) A dataset of *Raman* spectra provided by [61].

We first start with a comparison with the preconditioning framework. In particular we consider three different MP frameworks: 1) DeepMP, and 2) NNMP with preconditioning and 3) FNNOMP with preconditioning.

We compare the performance of the two versions of DeepMP compared to NNMP and FNNOMP [10] with respect to the exact recovery of the support set. As an evaluation metric for the exact recovery of the support set we are using the normalized *Hamming distance* complement [108] defined as follows (4.42),

$$H^C(s_a, s_g) = \sum_{t=1}^{T} 1 - \frac{|s_a(t) - s_g(t)|}{k}, \tag{4.42}$$

where T the number of test samples. $s_g(t)$ is the ground truth sparse vector that consists of 0's and 1's. The nonzero indexes of the vector correspond to the labels of the atoms that contribute to $y(t)$ and form the support set s. $s_a(t)$ corresponds to the support set, that takes the form of a sparse vector, acquired by each of the decomposition algorithms.

We generate a number of $500K$ samples for each level of sparsity and from them we train the models with respect to $425K$ while the remaining $75K$ points are the test samples. The coefficients are drawn from a uniform distribution of zero mean and unit variance. We optimize the models with respect to the adabound [107] optimizer with the settings presented in Table 4.4.

We will start the evaluation with respect to H^C for the *Random* dictionary with $N = 40$. The obtained results are demonstrated in Figure 4.6. As it can be seen from the results, DeepMP outperforms NNMP and FNNOMP with preconditioning with respect to particular metric. Both frameworks demonstrate identical performance

	M	N	lr	final_lr	epochs
Synthetic data	30	40	$1e-3$	0.1	10
Synthetic data	30	200	$1e-3$	0.1	20
Raman data	503	2521	$1e-3$	0.1	30

Table 4.4: Dimensions of the evaluation datasets and training parameters of the associated networks.

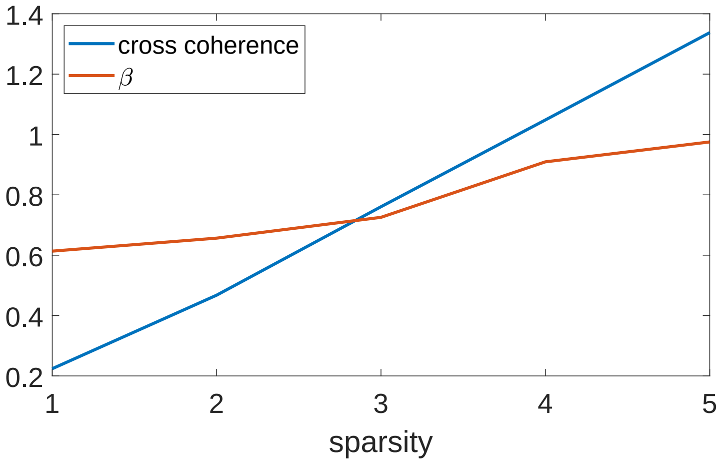


Figure 4.7: Comparison of β and cross coherence for Synthetic data, $N = 40$.

$H^C(1) = 1$. Hence both frameworks fully recover s for $k = 1$. However, $H^C(2) = 1$ for DeepMP while $H^C(2) = 0.98$ for both NNMP and FNNOMP with preconditioning. This essentially means, that DeepMP fully recovers s for $k = 2$ as well. A closer look to the growth of the cross coherence over k for DeepMP are demonstrated in figure 4.7 provide a better understanding why this phenomenon occurs. In particular, it is lower than β for a k up to 2, therefore no spiral atom is introduced in s.

We will continue our comparison for the *Random* dictionary with $N = 200$ and the *Raman* dictionary. The obtained results are demonstrated in Figures 4.8 and 4.9. The results indicate that DeepMP outperforms preconditioned NNMP and FNNOMP in terms of H^C. This essentially means that the performance of the MP framework can be improved, in terms of support recovery by incorporating a series of representation matrices $\boldsymbol{W}^k$ compared to a single $\boldsymbol{\Psi}$. The latter is not of surprise given that DeepMP consists of an overall number of kMN parameters at the selection step compared to the

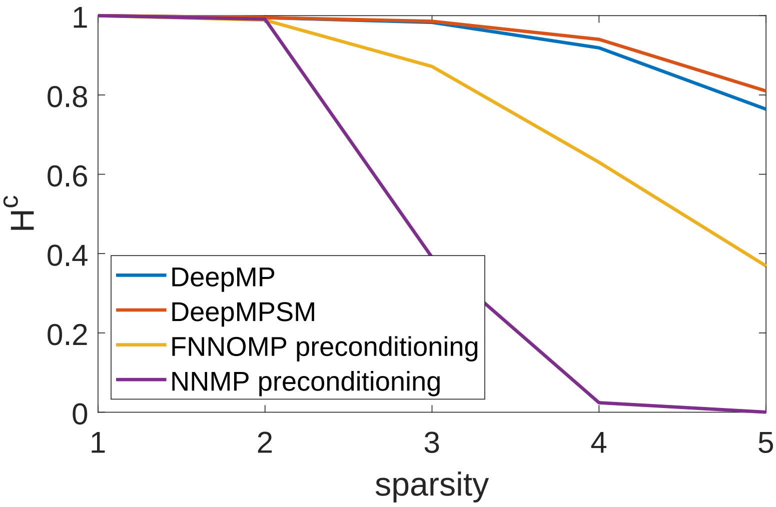


Figure 4.8: Comparison with respect to H^c Synthetic data.

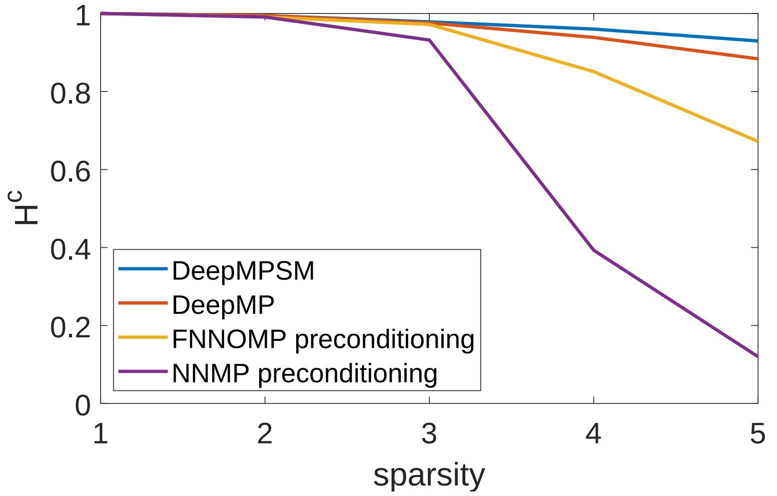


Figure 4.9: Comparison with respect to H^c Raman data.

MN parameters of the preconditioning framework when incorporated at the structure of the standard OMP,MP frameworks. Hence, DeepMP demonstrates a better performance due to the higher degree of flexibility that characterises the model. This implies that a sequential form of preconditioning introduces a more flexible approach compared to the standard form of preconditioning. However, in case of the *Random* dictionary, in contrast with the performance of the the network for $N = 40$, $H^C(2) < 1$. The latter indicates, that the growth of the coherence function and eventually the performance of support recovery relies on the redundancy of the dictionary, i.e. $\frac{N}{M}$.

For the next stage of the comparison we will evaluate the cumulative cross –coherence function (4.12) for the representation matrices of DeepMP and the cumulative cross–coherence function for the representation matrix $\mathbf{\Psi}$ of the preconditioning framework. The obtained results for the *Synthetic* dictionaries and the *Raman* data are

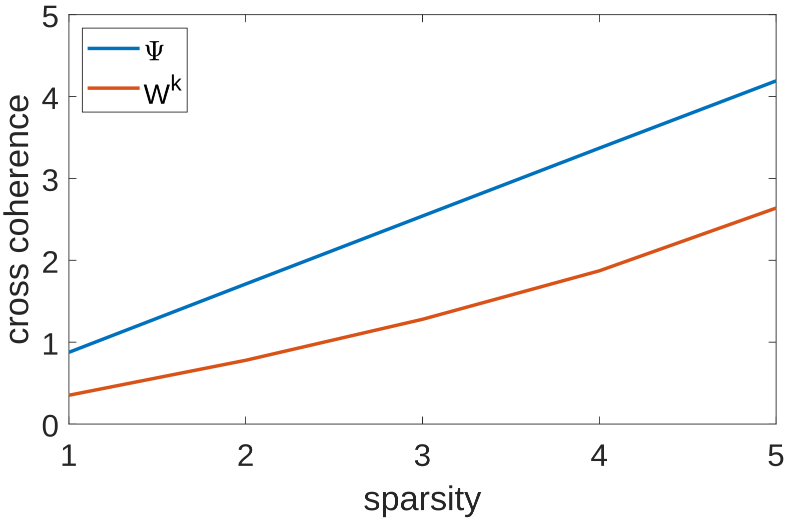


Figure 4.10: Cumulative (cross) coherence for Synthetic data.

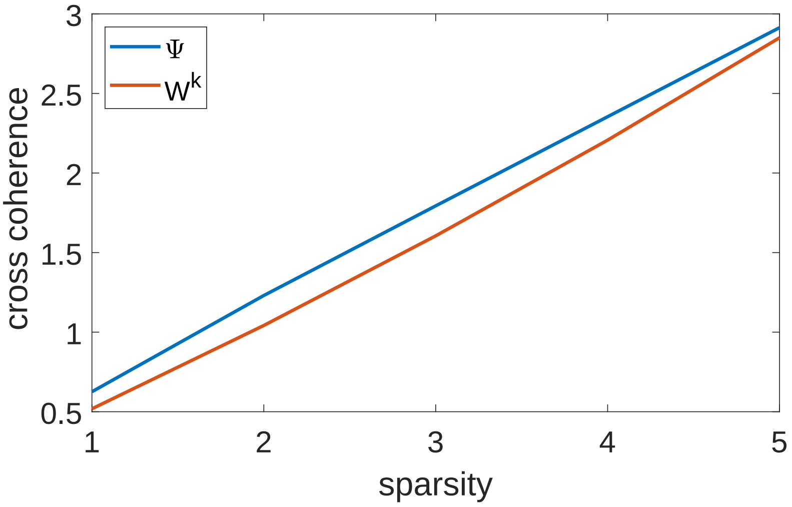


Figure 4.11: Cumulative (cross) coherence for Raman data.

demonstrated in Figures 4.10 and 4.11 accordingly. As it can be seen from the results the cumulative cross–coherence the $\boldsymbol{W}^k$'s (where $k = 5$ for DeepMP) is lower in both the data. This results in an improved performance of DeepMP over the preconditioning framework. The cumulative cross–coherence is much lower for the *Synthetic* data compared to the *Raman* data. This results to a bigger margin in terms of support recovery performance of DeepMP over the preconditioning framework for the two datasets.

Note the following, within our experimental analysis we tune the number of layers k in DeepMP to be equal to s. However, there is no restriction to follow this sort of approach for setting k in DeepMP. We can add more layers to the structure of the network that increases the number of parameters and hence the flexibility of the network, however this would result to an increase in the computational cost as well. The k approach we follow here, essentially evaluates the performance of DeepMP and greedy techniques in general, with respect to the ideal scenario i.e. retrieve the k elements that constitute $\boldsymbol{y}$ within the first k iterations–blocks of each approach.

Within the so far conducted analysis we have not evaluated DeepMP with respect some trivial challenges that characterise natural data i.e. the presence of noise. The preliminary focus of the current work was to introduce some sort of interpretability into Deep NN's such as DeepMP. From the perspective of DeepMP this has to do with the theoretical conditions that guarantee the acquisition of ground truth atoms within the first k blocks of the model. This sort of analysis lays highlights that the development of a training process that aligns with the theoretical criteria may boost the performance of the model. This is left for a future work though. This comes in contrast with the widely followed approach with Deep NN's where the developed methodologies are treated as some sort of black box machines that demonstrate outstanding results. However, the

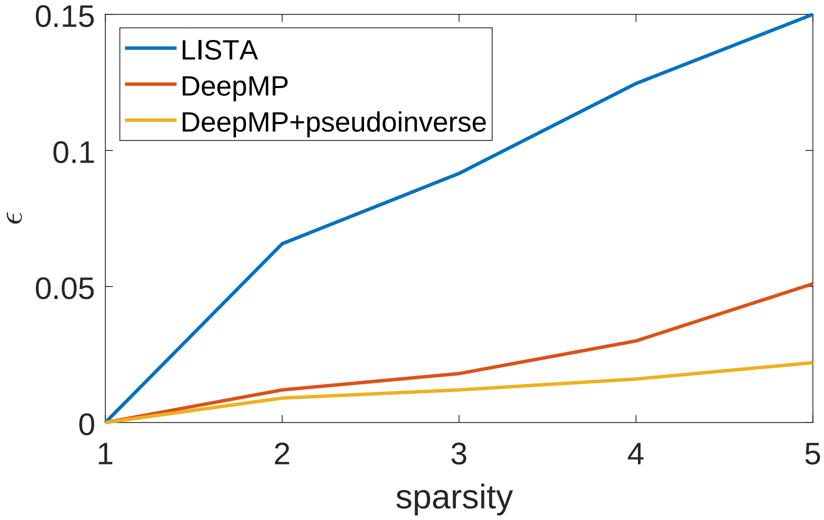


Figure 4.12: Comparison of the different Sparse Coding frameworks with respect to signal reconstruction on the Synthetic data.

evaluation with respect to noisy data can take place in the results section of a future submission. This sort of analysis may also extend this line of work to other aspects of Deep NN's i.e. generalization.

For the next step we conduct a comparison with respect to the exact signal recovery in between DeepMP, DeepMP+pseudoinverse and LISTA [90]. We generate signals of varying sparsity $k = 1, \ldots 5$. This results to an equivalent number of layers compared with DeepMP. For the fairness of comparison, we follow the same principle for LISTA in the sense that the number of layers is adapted to the sparsity of the signal. This results in models that have relatively similar capacity. The two models are evaluated with respect to their performance or recovering the ground truth sparse signal $\boldsymbol{x}$ as

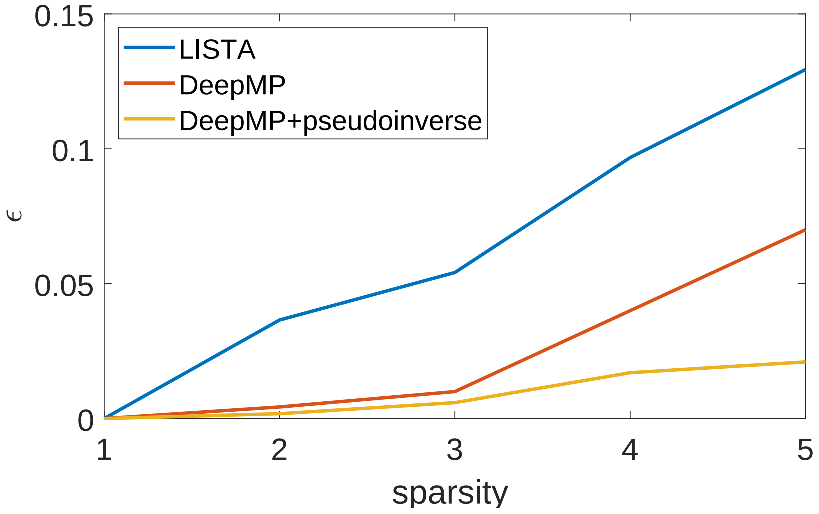


Figure 4.13: Comparison of the different Sparse Coding frameworks with respect to signal reconstruction on the Raman data.

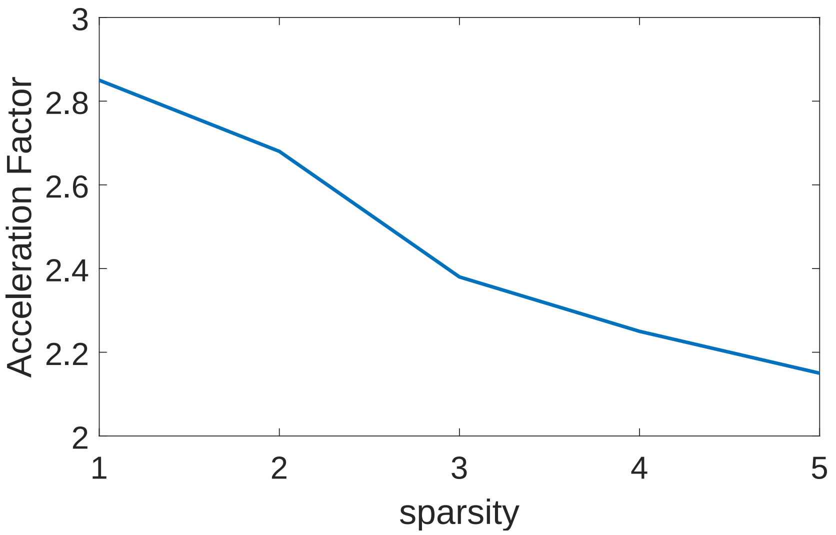


Figure 4.14: Acceleration of DeepMP compared to LISTA

introduced in (4.43). Given that LISTA may not acquire exactly k coefficients, we keep the k largest coefficients. Obtained results are demonstrated in Figure 4.12. Note that B is the total number of test points, while x_i is the ground truth signal, $x_i^\star$ the reconstructed signal by the recovery algorithm for the i–th test point.

$$\epsilon = \frac{1}{B}\sum_{i=1}^{B}\frac{||\boldsymbol{x_i} - \boldsymbol{x}_i^\star||_2^2}{||\boldsymbol{x}_i||_2^2} \tag{4.43}$$

As it can be observed in the results, DeepMP and its variant DeepMP+pseudoinverse demonstrate a better performance in terms of signal reconstruction compared to LISTA. The pseudoinverse introduces an improvement on the recovery of $\boldsymbol{x}$ at the cost of additional computational costs. The decay in performance over sparsity highlights that the task of signal reconstruction becomes more complicated and difficult while more components contribute in the signal.

Finally, we will focus on the computational cost that characterises each of these approaches, i.e. DeepMP and LISTA. The comparison is established with respect to the metric introduced in the following,

$$\text{Acceleration Factor}(k) = \frac{\text{Time LISTA}(k)}{\text{Time DeepMP}(k)} \tag{4.44}$$

As it can be seen from the results, DeepMP is faster than LISTA, with an acceleration factor of 2 over the iterations, which aligns with the analysis established in Section 4.6.1.

4.8 Summary

In this chapter we have investigated the benefits that can be obtained by unfolding conventional sparse signal decomposition frameworks and in particular NNMP by means of a Deep Neural Network. We first compare our method with the conventional greedy techniques for sparse non–negative signal decomposition techniques. The acquired results highlight that by learning dictionary layouts with lower coherence all over the dictionary points, we can boost the performance of greedy techniques. We continue by laying down the soil for drawing a theoretical analysis upon the performance of DeepMP. By doing that we introduce interpretability of the results of DeepMP. We compare DeepMP with the preconditioning framework. The results indicate that the higher degrees of flexiblity implied by DeepMP results to an improved performance compared to preconditioning. Finally, we compare DeepMP with LISTA a state of the art framework for sparse decomposition. DeepMP outperforms LISTA in terms of signal recovery while in the same time introduces a decomposition framework with lower computational cost. The performance of DeepMP may be improved in terms of signal recovery by incorporating the pdeudoinverse outside the model.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

Within the current thesis we focus on the improvement of greedy techniques from the perspective of sparse non–negative signal processing with respect to two main directions: 1) acceleration, and 2) signal recovery. To do as such we formed a bridge in between greedy techniques and data aware methodologies.

Our motivation is driven by the advantages that can be exploited via data aware methodologies and may be incorporated in the structure of conventional sparse signal decomposition techniques. The main characteristics of data we exploited here are, from the perspective of acceleration, the lower dimensional representation that may be present in natural data and can be exploited by data aware methodologies such as PCA. From the perspective of signal reconstruction, the performance of the conventional signal decomposition algorithms is one way or the other bounded by the underlying structure of the standard dictionary $\mathbf{\Phi}$. A key characteristic of data ware methodologies is that we can design dictionary layouts where atoms are mutually less coherent the one to the other.

The standard exact nearest neighbor algorithms suffer from the so–called curse of dimensionality in high dimensions. On the other hand, linear embbedings yield fast but approximate solutions approximate solutions to the NN search problem. Within the current work we laid down the soil for the development of a fast and theoretically robust framework that guarantees the acquisition of the exact NN from R^K to R^N.

Within the current work we propose a novel algorithm for the generic NN problem. In chapter 3, we demonstrated the benefits in terms of acceleration that can be obtained by incorporating such an approach within the structure of FNNOMP. We demonstrated that our approach employs benefits from the lower dimensional representation of natural data when present. We extended our results to the so–called MIPS problem.

The conventional decomposition schemes may demonstrate a poor performance in terms of signal decomposition. On the other hand crafted learned frameworks for sparse coding are characterised by a relatively high computational cost that makes the prohibitive from the perspective of a real time application. The unfolding of conventional signal decomposition algorithms such as MP, release the degrees of freedom of the decomposition process of the conventional schemes. In the same time, it results to a framework with a tractable computational workload. We demonstrated the benefits in terms of signals reconstruction, by unfolding a conventional sparse non–negative signal processing algorithm such as NNMP by means of Deep Neural network.

An extra benefit of unfolding is that it allows the introduction of some sort of interpretability for the acquired results. This comes in contrast with the typical approach followed in the Deep Learning regime where Deep NN's are treated as some sort of black box machine. The empirical results as well as the conducted theoretical analysis were demonstrated in Chapter 4.

An alternative approach that we could follow at this project would be the one where we could explicitly focus on either of our two preliminary goals: 1) acceleration 2) robustness, and extend the so far acquired analysis–results. That would essentially lead to a situation where e.g. from the perspective of acceleration we would focus on the Nearest Neighbour problem and all the various techniques that we could deploy on top of the ones we demonstrated on Chapter 3 of the current work. From the perspective of robustness we could investigate how the existing model could be extended into other scenarios i.e. train a model that incorporates a denoising process while training etc. However, we chose to introduce a more comprehensive approach from the perspective of greedy techniques and leave the these alternatives for future investigation.

5.2 Future directions

From the perspective of the NN search problem, we propose two algorithms: 1) A theoretically robust algorithm for the explicit case where the associated operator strictly shrinks distances, and 2) A heuristic but theoretically fragile algorithm for the general case of operators.

The first question that could be answered in the future is whether a theoretically robust framework could be developed for the general cases of linear operators as well. From the perspective of operators that strictly shrink distances we utilized PCA. Nonetheless, this does not necessarily mean that is the best operator to serve the purpose. Designing a non–expansive operator that has the ability to retain the structure of $\mathbf{\Phi}$ with a minimal distortion in lower dimensions than PCA would result to an acceleration of AE–NN. However, another direction could be the one where a non–linear operator is incorporated within the embedding step of AE–NN. This sort of operator would serve the purpose as long as it is a non–expansive one.

A critical parameter for the design of this operator would be the computational cost of the embedding operation at step 1 of AE–NN. In particular the embedding operator should introduce a computational cost of KM operations such that the associated workload in step 1 is similar in between PCA and the new operator. The benefits of following this approach would the the introduction of a smaller computational workload at the the update step of the algorithm that eventually would lead to an acceleration to the overall functionality of AE–NN.

Apart from the design of a linear or non-linear expansive operator, another direction could be the one where AE-NN is applied in other applications associated with the standard NNS problem. This type of applications could be related with pattern recognition [65],data mining [66], data compression [67], data mining [113] etc.

From the perspective of signal reconstruction a potentially new research direction could be the one that performs denoising and decomposition on an integrated manner. Note that within the current work we focus on the noiseless case and we demonstrate the associated empirical and theoretical results.

The overall idea that DeepMP relies on is the unfolding of the k iterations of NNMP within k layers of a Deep NN. A potential future direction could be the one where an extra layer is be added that preceds the decomposition process. The main scope of the particular layer would be to act as a denoiser. In the ideal scenario the DeepMP with a denoising layer would perform identically in terms of signal desomposition to DeepMP. The difference would be that DeepMP+denoising would be evaluated with respect to noisy measurements while DeepMP would be evaluated with noiseless measurements.

Another approach is to integrate AE–NN with DeepMP. In particular, given a set of k different sets of weights obtained via DeepMP after training we could apply AE–NN on a per layer manner. In such case, we would transfer the weights $\boldsymbol{W}^k$ and plug them into an iterative scheme via blocking [114]. At each iteration k of that scheme the associated set of weights $\boldsymbol{W}^k$ would be loaded from the system memory.

The search for each layer then, could be accelerated via AE–NN. In such case a different operator $\boldsymbol{Q}^k$ would be incorporated at each iteration of the algorithm associated with the corresponding set of weights $\boldsymbol{W}^k$.

Bibliography

[1] M. Lustig, D. L. Donoho, J M. Santos, and M. Pauly *Compressed Sensing MRI. IEEE Signal Processing Magazine*, 2008.

[2] S.S Chen, D.L Donoho and M.A Saunders,“ Atomic decomposition by basis pursuit“, *SIAM* J.Sci.Comput, vol.20,no.1 pp 33–61,1999.

[3] S. Mallat, and Z. Zhang, “Matching Pursuit with time frequency dictionaries “, *IEEE Trans. on Signal Processing, vol.41 no.12,pp 3397–3415,1993.*

[4] Y.C. Pati, R. Rezaiifar, and P.S. Krishnaprasad, “Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition,“ in *Asilomar Conference on Signals, Systems and Computers*, 1993, pp. 40–44.

[5] B.K. Natarajan, “Sparse approximate solutions to linear systems,“ *SIAM Journal of Comput*, vol. 24, no. 2, pp. 227–234, 1995.

[6] A. Maier, C. Syben, T. Lasser, C. Riess,“A gentle introduction to deep learning in medical image processing“, in *Z. für Med. Phys*, 2019.

[7] H. Purwins, B. Li, T. Virtanen, J. Schlüter, S.-Y. Chang, and T. Sainath, “Deep Learning for Audio Signal Processing,“ IEEE J. Sel. Topics Signal Process, 2019.

[8] E.J Candes, M.B Wakin “An Introduction To Compressive Sampling“. *IEEE Signal Processing Magazine*, 2008.

[9] K. A. Voulgaris, M. E. Davies and M. Yaghoobi “Accelerated Search for Non-Negative Greedy Sparse Decomposition via Dimensionality Reduction“, *Sensor Signal Processing in Defense (SSPD)*, 2019.

[10] Yaghoobi V., M, Wu, D & Davies, M , “Fast Non-Negative Orthogonal Matching Pursuit“ *IEEE Signal Processing Letters*, vol. 22,no. 9, 2015.

[11] H.F Yu, C.J Hsieh, Q. Lei, I.S Dhillon,“ A Greedy Approach for Budgeted Maximum Inner Product Search“, *Advances in Neural Information Processing Systems (NIPS)*, 2017.

[12] K. A. Voulgaris, M. E. Davies and M. Yaghoobi “Accelerated Search for Non-Negative Greedy Sparse Decomposition via Dimensionality Reduction“, *European Signal Processing Conference (EUSIPCO)*, 2020.

[13] S. Foucard and H. Rauhult *A Mathematical Introduction to Compressive Sensing.* Birkhuser, 2013.

[14] M. Lustig, D. Donoho and J. M.Pauly. Sparse MRI: “The application of compressed sensing for rapid MR imaging“, *Magnetic Resonance in Medicine*, 58:1182–1195,2007.

[15] A. Liutkus, D. Martina, S. Popoff, G. Chardon, O. Katz, G. Lerosey, S. Gigan, L. Daudet and I. Carron,“Imaging with nature: Compressive imaging using a multiply scattering medium.“*Scientific reports*, 4,2014.

[16] C. Wu, D. Wu, S. Yan, and Y. Guo. “Sensor deployment in bayessian compressive sensing based environmental monitoring“. IN *MOBIQUITOUS 2013*,2013.

[17] MD Iordache, JM Bioucas-Dias, and A Plaza, “Sparse unmixing of hyperspectral data,“ *Geoscience and Remote Sensing, IEEE Transactions. on*, vol. 49, no. 6, pp. 2014–2039, 2011.

[18] Y Qian, S Jia, J Zhou, and A. Robles–Kelly, “Hyperspectral unmixing via sparsity-constrained nonnegative matrix factorization,“ *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 49, no. 11, pp. 4282–4297, 2011.

[19] H. Kim and H. Park, “Sparse non-negative matrix factorizations via alternating non-negativity-constrained least squares for microarray data analysis,“ *Bioinformatics*, vol. 23, no. 12, pp. 1495–1502, 2007.

[20] D. Wu, M. Yaghoobi, S. I Kelly, M.E Davies, and R. Clewes, “A sparse regularized model for raman spectral analysis,“ in *Sensor Signal Processing for Defence*, Edinburgh, 2014.

[21] J. Chen, L. Zhang, C. Bai, K. Kpalma, “Review of Recent Deep Learning Based Methods for Image-Text Retrieval “, IEEE Conference on Multimedia Information Processing and Retrieval, 2020.

[22] J. Huang, J. Chai, S. Cho, “Deep learning in finance and banking: A literature review and classification “, 2020.

[23] J. Cervantes, F. Garcia–Lamont, L Rodriguez–Mazahua, A. Lopez, “A comprehensive survey on support vector machine classification: Applications, challenges and trends“, in *Neurocomputing*, 2020.

[24] S. Dreiseitla L. Ohno–Machado, “Logistic regression and artificial neural network classification models: a methodology review“ , in *Journal of Biomedical Informatics*, 2002.

[25] E. Grant, C. Hedge, P. Indtyk, “Nearly optimal linear embeddings into very low dimensions“, in *Global Conference on Signal and Information Processing*, 2013.

[26] R. Qaddoura, H. Farris, I. Aljarah, “An efficient clustering algorithm based on the k–nearest neighbors with an indexing ratio“, in *International Journal of Machine Learning and Cybernetics*, 2020.

[27] G. Tsoumakas, I. Katakis, I. Vlahavas, “Random k-Labelsets for Multi-Label Classification “, *IEEE Transactions on Knowledge and Data Engineering* 2010.

[28] B.A Olshausen, and D. Field. “Emergence of simple-cell receptive field properties by learning a sparse code for natural images“. *Nature*, 381(6583):607–609, 1996.

[29] K. Schnass, Pierre Vandergheynst, “Dictionary Preconditioning for Greedy Algorithms“, IEEE Transactions on Signal Processing, 2008.

[30] A.S. Bandeira,E. Dobrihan, D.G. Mixon and W.F.Sawing,“Certifying the Resticted Isometry Property is Hard“,*IEEE Transactions on Information Theory*,vol 59,no.6,2013, pp. 3448–3450.

[31] D. Blanchard, Cartis and J. Tanner “ Compressed Sensing: How Sharp Is the Restricted Isometry Property? “,*SIAM Rev*,vol.53,no.1,2011,pp105–125.

[32] E.J. Candes,“The restricted isometry property and its implications for compressed sensing“,*Comptes Rendus Mathematique*,vol.346,no. 9–10,2008,pp,589–592.

[33] M. A. Davenport, M. F. Duarte, Y. C. Eldar, and G. Kutyniok “Introduction to Compressed Sensing“,*Cambridge University Press*,2012.

[34] L. Stankovic, E. Sejdic, S. Stankovic, M. Dakovic,“A Tutorial on Sparse Signal Reconstruction and Its Applications in Signal Processing “,*Circuits Systems and Signal Processing*, 2019.

[35] F. Yang, S. Wang, and C. Deng. “Compressive Sensing of image reconstruction using multi–wavelet transforms“,In *2010 International Conference on Intelligent Computing and Intelligent Systems*,volume 1, pages 702–705. 2010.

[36] J. Bazerque, G. Giannakis,“Distributed spectrum sensing for cognitive radio networks by exploiting sparsity “, *IEEE Transaction on Signal Processing*,vol.58, no 3,2010,pp 1847–1862.

[37] H. Mohimani, M. Babaie–zadeh, and C. Jutten, “A fast approach for overcomplete sparse decomposition based on smoother l0 norm“, In, *IEEE Trans. on Signal Processing* ,2009,no.1 289–301.

[38] D.L Donoho, and J.M Johnstone, “ Ideal spatial adaptation by wavelet shrinkage“,*Biometrica 81*,no.3,pp425–455, *Biometrica 81.*

[39] T. Blumensath, M.E Davies, “Iterative Hard Threshlding for Compressed Sensing“, Applied and Computatotional Harmonic Analysis, 2009.

[40] E. Candes, J. Romberg,“Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information“,*IEEE Transactions on Information Theory*,vol. 52, no.2 2006,pp 489–509.

[41] E. Candes, M. Wakin, “An Introduction to Compressive Sampling “, *IEEE Signal Processing Magazine*,vol. 25,no.2 2008,pp21–30.

[42] D.L. Donoho,“Compressed sensing“,*IEEE Transactions on Information Theory*,vol.52,no.4,pp 1826–1306,2006.

[43] M.A Figueiredo, R.D Nowak, S.J. Wright, “Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems,“*IEEE Journal of Selected Topics in Signal Processing* vol.1 no. 4, pp 586–597.

[44] S. Sardy, A. G. Bruce and P. Tseng,“Block coordinate relaxation methods for nonparametric wavelet denoising “,*Comp. and Graph. Statist.*vol.9,no.2, 2000

[45] J. A Tropp “Greedy is Good: Algorithmic Results for Sparse Approximation“, *IEEE Transactions on Information Theory*,vol 50, No 10, October 2004.

[46] PhD thesis, “Adaptive Sparse Coding and Dictionary Selection“ M.Y. Vaighan, 2010.

[47] A. Beck, M. Teboulle, “A fast iterative shrinkage–thresholding algorithm for linear inverse problems “,*SIAM Journal Imaging Sci.*vol 2, no.1,pp 183-–197,2009.

[48] J. H Friedman and W. Stuetzle,“Projection pursuit regression”,*Journal of the American Statistical Association* vol, 76, pp 817–823.1981.

[49] P.J. Huber, “Projection pursuit ”, *Annals of Statistics* vol.13, no.2,pp 435–475, 1985.

[50] V. Temlyakov,“Weak Greedy Algorithms”,*Advances in Computational Mathematics* vol.12 , no.2,3,pp 213–227,2000.

[51] R. Gribonval, P. Vandergheynest,“On the exponential convergence of Matching Pursuits in quasi–incoherent dictionaries“,*IEEE Transactions in Information Theory*,2006,pp 256–261.

[52] B. Sturn, M. Christensen, “Comparison of orthogonal matching pursuit implementations“, EUSIPCO, 2012.

[53] D. Needel, J.A Tropp :“CoSaMP: Iterative signal recovery from incomplete and inaccurate samples“. In *Applied and Computational Harmonic Analysis.*

[54] A. Mahadevan–Jansen, R. Richards–Kortum, *Raman Spectroscopy For Cancer Detection: A Review.* Engineering in Medicine and Biology Society, 1997. Proceedings of the 19th Annual International Conference of the IEEE.

[55] H. Guo, L. He, B. Xing *Applications of surface-enhanced Raman spectroscopy in the analysis of nanoparticles in the environment* . Environmental Science:Nano, (2017).

[56] D.J Gardiner *Practical Raman spectroscopy*, Springer (1989).

[57] S. Faqurharson *Pharmaceutical Applications of Raman Spectroscopy* . American Pharmaceutical Review (2014).

[58] M. Yaghoobi, T. Grecu, S. Brookes, C.J Campbell, “Automatic Cocrystal Detection by Raman Spectral Deconvolution–Based Novelty Analysis “ in *Analytical Chemistry*,2021.

[59] M. Petersen, Z. Yu, X. Lu “Application of Raman Spectroscopic methods in Food Safety“ in *Biosensors*, 2014.

[60] M. Liszwenska, B. Bartosewics, B. Budner, B. J. Jankievicz, “Raman and SERS spectroscopies in the detection of hazardous materials“ in *Baltic URSI Symposium*, 2018.

[61] https://www.stjapan.de/.

[62] T. T. Nguyen, J. Idier, C. Soussen, El–Hadi Djermoune. “Non–Negative Orthogonal Greedy Algorithms“IEEE Transactions on Signal Processing,vol.76, 2019.

[63] D. Kim, J.P. Haldar :“Greedy algorithms for nonnegativity-constrained simultaneous sparse recovery”. In *Signal Processing*,2016.

[64] R.Peharz, F. Perncopf ,“Sparse nonegative matrix factorzation with l_0 constraints”. In *Neurocomputing Elsevier*, 2012. pp 38–46.

[65] T. Cover and P. Hart, "Nearest neighbor pattern classification",*IEEE Transactions on Information Theory*,1967,pp 22–27.

[66] C. Buckley, A. Singhal , M. Mitra and G. Salton. "New Retrieval Approaches Using SMART:TREC" 4. In. *Proceedings of the Fourth Text Retrieval Conference*, National Institute of Standards and Technology 1995.

[67] A. Gersho and R.M Gray. "Vector Optimization and Data Compression ", Kluwer, 1991.

[68] N. Segata and E. Blanzieri. "Fast and scalable local kernel machines", *Journal of Machine Learning Research*:11, 1883–1926, 2010.

[69] N. Tziotziotis, C. Dimitrakakis and K. Blekas, "Cover Tree Bayessian Reinforcement Learning",*Journal of Machine Learning Research*,15,2014.

[70] A. Beygelzimer, S. Kakade, J. Langford. " Cover Trees for Nearest Neighbor" . *Proceedings of the 23rd international conference on Machine learning* (2006).

[71] R. Krauthgamer and J. Lee. "Navigating nets: Simple algorithms for proximity search, Proceedings of the 15th Annual Symposium on Discrete Algorithms" (SODA), 791–801, 2004.

[72] K. Clarkson: "Nearest Neighbor Searching in Metric Space: Experimental Results for sb(S)", 2002.

.

[73] L. Boytsov, "Efficient and Accurate Non-Metric k-NN Search with Applications to Text Matching", *PhD Thesis*, 2018.

[74] A. Kibriya, E. Frank, "An Empirical Comparison of Exact Nearest Neighbour Algorithms ", *Proceedings of the 11th European conference on Principles and Practice of Knowledge Discovery in Databases*,2007.

[75] K. Liu, H. Kargupta, and J. Ryan, "Random projection-based multiplicative data perturbation for privacy preserving distributed data mining," *IEEE Transactions on knowledge and Data Engineering*, vol. 18, no. 1, pp. 92–106, 2006.

[76] S. Jassim, H. Al-Assam, and H. Sellahewa, "Improving performance and security of biometrics using efficient and stable random projection techniques," in *Image and Signal Processing and Analysis*, 2009. ISPA 2009. *Proceedings of 6th International Symposium on.* IEEE, 2009, pp. 556–561.

[77] B. Yang, D. Hartung, K. Simoens, and C. Busch, “Dynamic random projection for biometric template protection,“ in *Biometrics: Theory Applications and Systems (BTAS), 2010 Fourth IEEE International Conference on.* IEEE, 2010, pp. 1–7.

[78] M. Nabil , “Random Projection and Its Applications“, *https://arxiv.org/*,2017

[79] S. Kakade, G. Shakhnarovich,“Random Projections“, CMSC 35900, 2009.

[80] M. Waqar, H. Dadwood,P. Guo, M.B Shahnawaz, M.A Ghazanfar, “Prediction of Stock Market by Principal Component Analysis“, In. *International Conference on Computational Intelligence and Security*, 2017.

[81] S.C. Ng “Principal component analysis to reduce dimension on digital image“, In. *ELSEVIER*,113–119, 2017.

[82] C. Hegde, A. C. Sankaranarayanan, W. Yin, and R. G. Baraniuk, “NuMax: A Convex Approach for Learning Near-Isometric Linear Embeddings“, *IEEE Transactions on Signal Processing* vol. 63, 2015.

[83] M. Fazel, “Matrix rank minimization with applications,“ Ph.D. dissertation, Stanford Univ., Stanford, CA, USA, 2002.

[84] M. Elad, M. and M. Aharon. “Image denoising via learned dictionaries and sparse representation“. In *CVPR*, 2006.

[85] M.A. Ranzato, Y. Boureau, S. Chopra, and Y. LeCun, “A unified energy-based framework for unsupervised learning“. In *AI-Stats’07*, 2007.

[86] J. Mairal, M. Elad, and G. Sapiro, “Sparse representation for color image restoration.“ *IEEE T. Image Processing*, 17(1):53–69, January 2008.

[87] K. Kavukcuoglu, Koray, M.A. Ranzato, Marc’Aurelio, and Y. LeCun. “Fast inference in sparse coding algorithms with applications to object recognition“. Technical Report, *Computational and Biological Learning Lab*, Courant Institute, NYU, 2008.

[88] H. Lee,R. Grosse, R. Ranganath, R., and A.Y Ng, “Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations“. In *International Conference on Machine Learning.* ACMNew York, 2009.

[89] J. Yang, K. Yu, Y. Gong, and T. Huang. “Linear spatial pyramid matching using sparse coding for image classification“. In *CVPR 09*, 2009.

[90] K. Gregor and Y. LeCun,“Learning Fast Approximations of sparse coding “, *in Proceedings of the 27th International Conference on Machine Learning.* Omnipress, 2010, pp. 399–406.

[91] C. Xiaohan, et al, “Theoretical linear convergence of unfolded ISTA and its practical weights and thresholds“.*Advances in Neural Information Processing Systems*,2018.

[92] R. Khatib, D. Simon, M. Elad, “Learned Greedy Method (LGM): A Novel Neural Architecture for Sparse Coding and Beyond “, *arXiv*, 2020.

[93] B. Moore “Principal component analysis in linear systems: Controllability, observability, and model reduction“ *IEEE Trans. Autom. Control.* vol 26 no.1 pp. 17-32,1981.

[94] D. Achlioptas,“Database–friendly random projections“ in *Proc. Symp. Principles of Database Syst. (PODS)*, Santa Barbara, CA,USA, May 2001.

[95] J. Matusek “On variants of the Johnson–Lindenstrauss lemma“ *Random Structures and Algorithms, 33, 142–156.*

[96] S. Marshland “ Machine Learning: An Algorithmic Perspective“, Chapman & Hall/Crc Machine Learning & Pattern Recognition Chapter 10, 2009.

[97] P. Indyk, R. Motwani, “Approximate nearest neighbors: towards removing the curse of dimensionality”, *Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing* ,(1998).

[98] C. P., K. Y., and T. R.“Performance of Recommender algorithms on top–n recommendation tasks”. In *Proceedings of the fourth ACM conference on Recommender Systems.* 39–46. ACM, 2010.

[99] D. T. Ruzon, M. Segal, M. Shlens J, Vijayanarasimhan S, and Yagnik J. “Fast, accurate detection of 1000000 object classes on a single machine”. In *Computer Vision and Pattern Recognition(CVPR) 2013.*

[100] T. Joachims, “Training linear svms in linear time”. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining.* 217–226. ACM, 2006.

[101] A. Shrivastava, and P. Li, 2014. “ Assymetric lsh (alsh) for sublinear and maximum inner product search(mips)“. In *Advances in Neural Information Processing Systems.* 2321-2329.

[102] N. Ahmed, T. Natarajan, K.R Rao , “Discrete Cosine Transform” In *IEEE Transactions on Computers*, 1974.

[103] D.J Gardiner *Practical Raman spectroscopy*, Springer (1989).

[104] P. Frankl, H Maehara,“The Johnson–Lindenstrauss Lemma and the Sphericity of Some Graphs”, *Journal of Combinatorial Theory*, 1988.

[105] T. Blumensath and M. Davies,“Iterative Thresholding for Sparse Approximations“,*Journal of Fourier Analysis and Applications*, vol.14 no.5 pp.629-654,2008.

[106] V. Nair and G.E Hinton “Rectified linear units improve restricted boltzman machines“, *in Proceedings on the 27th international conference on machine learning(ICML-10)*, 2010,pp. 807–814.

[107] L. Luo, Y. Xiong, Y. Liu, X. Sun, “Adaptive Gradient Method With Dynamic Bound of Learning Rate“,*ICLR*,2019

[108] A. Bookstein, V. Kulyukin, T. Raita, “Generalised Hamming Distance“,*Springer*,2002.

[109] E. Kreyszig, “Introductory Functional Analysis with Applications“. New York: Wiley, 1989.

.

[110] Z. Bai, L. Shi, J. R. Jensen, J. Sun and M. G. Christensen,“Acoustic DOA estimation using space alternating sparse Bayesian learning “, *EURASIP Journal Audio, Speech and Music Processing*, 2021.

[111] Farzad Parvaresh, Haris Vikalo, Sidhant Misra, and Babak Hassibi, “Recovering Sparse Signals Using Sparse Measurement Matrices in Compressed DNA Microarrays“, *IEEE Journal of selected topics in Signal Processing*, Vol. 2, NO. 3, JUNE 2008.

[112] D Wu, M Yaghoobi, S. I Kelly, M. E Davies, and R Clewes, “A sparse regularized model for raman spectral analysis,“ in *Sensor Signal Processing for Defence*, Edinburgh, 2014.

[113] T. Bozkaya and M. Ozsoyoglu, “Ditance–Based Indexing for High–Dimensional Metric Spaces“. *In: Proceedings of the ACM SIGMOID International Conference on Management Data*, 1997.

[114] M. S. Lam, E.E. Rothberg and M.E Wold, “The Cache Performance and Optimizations of Blocked Algorithms,“ACM SIGPLAN Notices, Volume 26, Issue 4, 1991.