Structured Sparse Models with Applications

A DISSERTATION

SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL

OF THE UNIVERSITY OF MINNESOTA

BY

Pablo G. Sprechmann

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

Doctor of Philosophy

Prof. Tryphon T. Georgiou

September, 2012

Acknowledgements

Foremost, I would like to thank my thesis advisor Prof. Guillermo Sapiro. He has been an outstanding mentor and a permanent source of encouragement. I admire his passion for research and dedication. This thesis would not have been the same without his insightful suggestions and creativity. But I would also like to thank him for his generosity and patience, and for the excellent example he has provided as a person.

I would like to thank Profs. Arindam Banerjee, Tryphon Georgiou Jarvis Haupt and Mos Kaveh for serving on my committee. I am grateful for the time that they dedicated for reviewing my work and for sharing interesting comments and discussions with me. I would like to thank Prof. Georgiou for "adopting" me as a student during the last months of my program, after Prof. Sapiro moved out from the University of Minnesota. I would also like to thank the grants that financially support our research.

I have had the great fortune of collaborating with wonderful people during my graduate studies: Prof. Alex Bronstein, Pablo Cancela, Prof. Yonina Eldar and Dr. Ignacio Ramirez. I have greatly benefited from their knowledge and great ideas, but most importantly I have enjoyed their company and friendship.

This work benefited significantly from interaction with many colleagues throughout these years. I hesitate to include a list with all the many names, as I will surely omit some who deserve mention. I will only list those who have been part of Prof. Sapiro's group: Iman Aganj, Xue Bai, Leah Bar, Alexey Castrodad, Julio Duarte, Marcelo Fiori, Jordan Hashemi, Jinyoung Kim, Oleg Kuybeda, Federico Lecumberry, Mona Mahmoudi, Ignacio Ramirez,

Thiago Spina, Mariano Tepper and Guoshen Yu. Thanks to them for all the great times. A special thanks goes to Julien Marial, for sharing his superb sparse coding code with me before it became publicly available.

I would also like to express my gratitude to the Instituto de Ingeniería Eléctrica, Universidad de la República, Uruguay, and all my colleagues there. Specially I would like to thank Prof. Gregory Randall who supported me and encouraged me to pursue the PhD.

None of this would have been possible without my family: my wonderful parents, my brother Martin and sister Magdalena. I want to thank them for their continuous support. Finally, I would like to thank a lot to my beloved girlfriend Mackenzie.

Abstract

In sparse modeling, it is assumed that a signal can be accurately represented by a sparse linear combination of atoms from a (learned) dictionary. A large class of signals, including most natural images and sounds, is well described by this model, as demonstrated by numerous state-of-the-art results in various signal processing applications. Along with the practical success, a rich theory of sparse or compressible signal recovery has been developed.

Sparse models assume minimal prior knowledge about the data, asserting that the signal has many coefficients close or equal to zero when represented in a given domain. From a data modeling point of view, sparsity can be seen as a form of regularization, that is, as a device to restrict or control the set of coefficient values which are allowed in the model to produce an estimate of the data. In this way, flexibility of the model (that is, the ability of a model to fit given data) is reduced, and robustness is gained by ruling out unrealistic estimates of the coefficients.

Implicitly, standard sparse models give the same relevance to all of the very large number of subsets of sparse nonzero coefficients (a number which grows exponentially with the number of atoms in the dictionary). This assumption can be easily proved false in many practical cases. Signals have in general a richer underlying structure that is merely disregarded by the model. In many situations, standard sparse models represent a very good trade off between model simplicity and accuracy. However, many practical situations could greatly benefit from exploiting the structure present in the data, potentially for interpretability purposes, improve performance and faster processing.

The main goal of this thesis is to explore different ways of including data structure into sparse models and to evaluate them in real image and signal processing applications. The main directions of research are: *(i)* extending sparse models through imposing structure in the sparsity patterns of non-zero coefficients in order to stabilize the estimation and

account for valuable prior knowledge of the signals; *(ii)* analyzing how this impacts in challenging real applications where the problem of estimating the model coefficients is very ill-posed. As a fundamental example, the problem of monaural source separation will be extensively evaluated throughout the thesis; *(iii)* studying ways of exploiting the underlying structure of the data in order to speed up the coding process. One of the most important challenges in sparse modeling is the relatively high computational complexity of the inference algorithms, which is of critical importance when dealing with very large scale (modern-size) applications as well as real-time processing.

Contents

List of Tables

List of Figures

1 Introduction

In recent years, sparse representations have received a lot of attention from the signal processing community. This is due in part to the observation that an important variety of signals (i.e. audio frames and natural images patches) can be represented sparsely in certain basis, such as DCT or Wavlets [69]. This is, the signals can be accurately approximated by a linear combination of a few elements (atoms) of some (often) redundant basis, here referred as dictionaries. This property was widely used in lossy image compression, as the removal of many small coefficients results in insignificant information loss.

Sparse modeling aims at learning non-parametric dictionaries from the data itself. In addition to being very attractive at the theoretical level, a large class of signals, including most natural images and sounds, is well described by this model, as demonstrated by numerous state-of-the-art results in various signal processing applications. A recent comprehensive review can be found in [9].

From a data modeling point of view, the sparsity assumption can be seen as a form of regularization, that is, as a device to restrict or control the set of coefficient values which are allowed in the model to produce an estimate of the data. The idea is that, by reducing the flexibility of the model (that is, the ability of a model to fit given data), one gains robustness by ruling out unrealistic estimates of the coefficients. In traditional sparse models, regularization translates to the requirement that "a few non-zero coefficients should be enough to represent the data well".

However, a great deal of flexibility in sparse models is still present in the usually very

large possible number of non-zero coefficients subsets, a number which grows exponentially with the number of atoms in the dictionary. Implicitly, standard sparse models give the same relevance to all of the usually very large number of subsets of sparse non-zero coefficients. It turns out that, in certain applications, the reduction in the flexibility of the model provided by traditional sparse models may not be enough to produce both stable and accurate estimates.

In general, certain combinations of non-zero coefficients (or even magnitudes) are preferred. Many natural signals have a richer underlying structure that is merely disregarded by standard sparse models. While in many situations standard sparse models represent a very good trade off between model simplicity and accuracy, many practical situations could greatly benefit from exploiting the structure present in the data, potentially for interpretability purposes, improve performance, and faster processing.

The fundamental idea behind sparse models is probably one of the most intuitive heuristic principles in the modeling of natural data: preferring a simple explanation to a more complex one. The power of using simple representations has been shown in a vast variety of applications in very diverse areas of science, including as a particular case the representation and analysis of matrix-valued data. For matrix models, the rank provides a very natural way to describe complexity. The intuition behind this approach is that many types of high-dimensional data have a much smaller intrinsic dimensionality. Because of this, dimensionality reduction techniques have been deeply studied for the last two decades or more. Principal component analysis (PCA) is probably the most widely used statistical technique for this purpose [52]. It is well known that the performance of PCA is highly sensitive to the presence of samples not following the low dimensional model; even a single outlier in the data matrix can render the estimation of the low rank component arbitrarily far from the true low rank matrix. This motivated an important amount of work dedicated to robustifying PCA [12, 108]. In a series of recent works [12, 126], a very elegant solution to

this problem was developed, in which the low rank matrix is determined as the minimizer of a convex program. A very neat analogy between sparse models and low rank recovery problems can be established, see Chapter 2 for details. Both, sparse modeling and low rank representations can be seen as a first order approximation of the data.

This particular formulation of robust PCA has spawned a lot of interest in the computer vision community and was successfully used in applications such as face recognition and modeling [83, 119, 120], background modeling [85, 133], and large scale image tag transduction [74]. Again here, there are many problems in which low rank models have shown to be effective despite the fact that the data to which it is applied still presents further structure that is not considered (or ignored) by the model. However. also in this case, it could be of great benefit to many practical applications to exploit the higher order structure of the problem.

The main questions are: how to exploit the structure in the data without loosing the benefits of sparse and low rank modeling? How can one further reduce that flexibility of these successful models in a meaningful way that can help stabilizing challenging ill-posed problems? Clearly these questions do not have a unique answer. This thesis is about exploring how to exploit the structure present in the data in order to extend and/or improve these models by incorporating prior (or learned) information to the specific problems.

One of the particular goals of this thesis is to evaluate the influence of the proposed models in real world applications in the fields of machine learning, image, video, and audio processing. However, the problem of monaural source separation receives special attention throughout this thesis. In this problem the main goal is to separate or isolate the different individual sources that are present in the mixture. Although important advances have been obtained throughout the years, this is still considered an open and difficult problem, mainly due to its high degree of under-determination. It is then crucial to use all the known available information to constraint the problem in a meaningful way, making it a very good

test-case for the purposes of this work.

The thesis is organized as follows. In Chapter 2, a brief review of sparse models and robust low rank representation is provided. Then, five different ways exploiting structure in sparse models are described in the following chapters. A list and a brief summary of each of them is given bellow. Finally, conclusions are drawn in Chapter 8.

Classification and clustering via sparse modeling:.

In this Chapter 3 we propose a new clustering framework within the sparse modeling and dictionary learning setting. Instead of searching for the set of centroid that best fit the data, as in k-means type of approaches that model the data as distributions around discrete points, we optimize for a set of dictionaries, one for each cluster, for which the signals are best reconstructed in a sparse coding manner. Thereby, we are modeling the data as a union of learned low dimensional subspaces, and data points associated to subspaces spanned by just a few atoms of the same learned dictionary are clustered together. In order to include structure of the problem, an incoherence promoting term encourages dictionaries associated to different classes to be as independent as possible, while still allowing for different classes to share features.

We first illustrate the proposed framework with examples on standard image and speech datasets in the supervised classification setting, obtaining results comparable to the state-of-the-art with this simple approach. We then present experiments for fully unsupervised clustering on extended standard datasets and texture images, obtaining excellent performance.

Hierarchical sparse models:.

In traditional sparse models, the sparse coefficients are obtained by treating each variable individually, without considering its position in the coefficient vector. Existing relationships between the different variables are essentially disregarded. In factor analysis problems, where one is interested in finding those factors which play a role in predicting response variables, an explanatory factor is often known to be represented by groups of input variables. In those cases, selecting the input variables in groups rather than individually produces better results [129].

The key idea in structured sparse modeling is to reduce the flexibility of the model (and obtain more stable estimates which at the same time are often faster to compute), by imposing (or promoting) further restrictions on the possible subsets of the estimated active coefficients. This is, instead of considering the atoms as singletons, the atoms are grouped according to a predefined structure. Depending on the application at hand, different structures might be considered. A particularly interesting way to add structure (and robustness) to the representation is to consider the simultaneous encoding of multiple signals. This is a natural collaborative filtering approach to sparse coding.

Chapter 4 presents an important particular case of structured sparsity: collaborative hierarchical sparse models. These models are evaluated in many settings ranging from object recognition to source separation and identification.

Audio signal modeling based on structured sparsity and GMM:.

In Chapter 5, we study how to incorporate physical structure present in harmonic

sources in the audio source separation problem. The decomposition of time-frequency representations, such as the power spectrogram, in terms of elementary atoms of a dictionary has become a popular tool in audio processing in the last decade. In particular, non-negative matrix factorization (NMF), [58], is closely related to sparse modeling and has lead to very good results in a variety of audio processing applications [41, 106]. We explore different possibilities of adding prior information such as the pitch information of harmonic sounds or temporal correlations to the representation via structured sparse modeling.

Learning efficient structured models:.

In Chapter 6, we present a comprehensive framework for structured sparse coding and robust low dimensional modeling, extending the recent ideas of using trainable fast regressors [39]. For this purpose, we propose efficient feed forward networks whose architecture faithfully approximate the exact solvers at a fraction of the complexity of the standard optimization methods. In this way, the structure in the data allows us to train the encoders in a meaningful way, so that they can faithfully approximate the solution of structured coding and low rank approximations for vectors following the same distribution as the training sample.

We further show that by using different training objective functions, the trainable sparse encoders can be adapted to dynamically changing input data. Hence, the encoders are no longer restricted to be mere approximants of the exact sparse code for a pre-given dictionary, as in earlier formulations, but can be rather used as full-featured sparse encoders or even modelers.

A simple implementation shows several orders of magnitude speedup compared to the

state-of-the-art at minimal performance degradation, making the proposed framework suitable for real time and large-scale applications. We present an extensive experimental evaluation in several computer vision and pattern recognition applications.

Real-time source separation via trainable encoders:.

Chapter 7 is dedicated to study the applicability of the trainable encoders proposed in Chapter 6 to the problem of monaural audio source separation. Specifically, to the problems of singing-voice/music-accompaniment separation and speech denoising in the presence of non-stationary noise.

The proposed approach builds upon the recent work by Huang et. al. [44]. In [44], the authors obtained state-of-the-art results in the singing-voice/music-accompaniment separation problem using robust PCA in the time-frequency domain. In that work, the repetitive structure of the accompaniment is represented with a low-rank linear model and the singing voice is regarded as sparse outliers.

In this work, we introduce a natural way of incorporating low-rank regularization to NMF. This regularization is used in two different ways. First, to propose a non-negative variant of robust PCA, termed as robust low-rank non-negative matrix factorization (RNMF). Second, to propose a model for representing speech signals that combines sparse representations with low rank modeling. As in Chapter 6, we propose two efficient feed-forward architectures that approximate the solution of the optimization problems with low latency and a fraction of the complexity of the exact method.

In this application, learning the underlying structure of the data plays a new and crucial role. As mentioned above, in [44] very good separation results were obtained by modeling the music accompaniment as low rank and the singing-voice as the remaining outliers. As

discussed in Chapter 5, however, using a richer model for representing the singing voice (e.g., the harmonic structure makes the voice patterns highly structured) would naturally produce a better performance. We propose to fill in the gap between the RPCA model and the real signals by incorporating learning, that is, by incorporating data structure that was ignored by the RPCA model.

Extensive experimental evaluation shows the benefit of this approach. The proposed method was implemented on portable devices, producing on the fly high quality separation results.

2 Background

In this chapter we present a brief summary of sparse modeling an low rank recovery.

2.1 Lasso and group lasso

Assume we have a set of data samples $\mathbf{x}_j \in \mathbb{R}^m, j = 1, \ldots, n$, and a dictionary of p atoms in $\mathbb{R}^m$, assembled as a matrix $\mathbf{D} \in \mathbb{R}^{m \times p}$, $\mathbf{D} = [\mathbf{d}_1 \mathbf{d}_2 \ldots \mathbf{d}_p]$. Each sample $\mathbf{x}_j$ can be written as $\mathbf{x}_j = \mathbf{D}\mathbf{a}_j + \epsilon$, $\mathbf{a}_j \in \mathbb{R}^p$, $\epsilon \in \mathbb{R}^m$, that is, as a linear combination of the atoms in the dictionary $\mathbf{D}$ plus some perturbation ϵ, satisfying $\|\epsilon\|_2 \ll \left\|\mathbf{x}_j\right\|_2$. The basic underlying assumption in sparse modeling is that, for all or most j, the "optimal" $\mathbf{a}_j$ has only a few nonzero elements. Formally, if we define the ℓ_0 cost as the pseudo-norm counting the number of nonzero elements of $\mathbf{a}_j$, $\left\|\mathbf{a}_j\right\|_0 := |\{k : a_{kj} \neq 0\}|$, then we expect that $\left\|\mathbf{a}_j\right\|_0 \ll p$ and $\left\|\mathbf{a}_j\right\|_0 \ll m$ for all or most j.

Seeking the sparsest representation $\mathbf{a}$ is known to be NP-hard. To determine $\mathbf{a}_j$ in practice, a multitude of efficient algorithms have been proposed, which achieve high correct recovery rates. The ℓ_1-minimization method is the most extensively studied recovery technique. In this approach, the non-convex ℓ_0 norm is replaced by the convex ℓ_1 norm, leading to

$$\min_{\mathbf{a} \in \mathbb{R}^p} \|\mathbf{a}\|_1 \quad \text{s.t.} \quad \left\|\mathbf{x}_j - \mathbf{D}\mathbf{a}\right\|_2^2 \leq \epsilon. \tag{2.1.1}$$

The use of general purpose or specialized convex optimization techniques allows for efficient reconstruction using this strategy. The above approximation is known as the Lasso

[105] or Basis Pursuit [15, 22]. A popular variant is to use the unconstrained version

$$\min_{\mathbf{a}\in\mathbb{R}^p} \frac{1}{2} \left\|\mathbf{x}_j - \mathbf{D}\mathbf{a}\right\|_2^2 + \lambda \left\|\mathbf{a}\right\|_1, \tag{2.1.2}$$

where λ is an appropriate parameter value, usually found by cross-validation, or based on statistical principles [36].

The fact that the $\|\cdot\|_1$ regularizer induces sparsity in the solution $\mathbf{a}_j$ is desirable not only from a regularization point of view, but also from a model selection perspective, where one wants to identify the relevant factors (atoms) that conform each sample $\mathbf{x}_j$. In many situations, however, the goal is to represent the relevant factors not as singletons but as groups of atoms. For a dictionary of p atoms, we define groups of atoms through their indices, $G \subseteq \{1,\ldots,p\}$. Given a group G of indexes, we denote the sub-dictionary of the columns indexed by them as $\mathbf{D}_{[G]}$, and the corresponding set of reconstruction coefficients as $\mathbf{a}_{[G]}$. Define $\mathcal{G} = \{G_1,\ldots,G_q\}$ to be a partition of $\{1,\ldots,p\}$. In order to perform model selection at the group level (relative to the partition $\mathcal{G}$), the Group Lasso problem was introduced in [129],

$$\min_{\mathbf{a}\in\mathbb{R}^p} \frac{1}{2} \left\|\mathbf{x}_j - \mathbf{D}\mathbf{a}\right\|_2^2 + \lambda\psi_{\mathcal{G}}(\mathbf{a}), \tag{2.1.3}$$

where $\psi_{\mathcal{G}}$ is the Group Lasso regularizer defined in terms of $\mathcal{G}$ as $\psi_{\mathcal{G}}(\mathbf{a}) := \sum_{G\in\mathcal{G}} \left\|\mathbf{a}_{[G]}\right\|_2$. The function $\psi_{\mathcal{G}}$ can be seen as a generalization of the ℓ_1 regularizer, as the latter arises from the special case $\mathcal{G} = \{\{1\},\{2\},\ldots,\{p\}\}$ (the groups are singletons), and as such, its effect on the groups of $\mathbf{a}$ is also a natural generalization of the one obtained with the Lasso: it "turns on/off" atoms in groups.

2.2 Dictionary learning

Now, what about the actual dictionary Φ? State-of-the-art results have shown that it should in general be learned from data. Given a set of signals $\{\mathbf{x}_i\}_{i=1\ldots m}$ in $\mathbb{R}^n$, the goal is to find a dictionary $\Phi \in \mathbb{R}^{n\times k}$ such that each signal in the set can be represented as a sparse linear combination of its atoms. In this thesis we use a variation of [66], where learning the dictionary is done by seeking a (local) solution to the following optimization problem,

$$\min_{\Phi,\{\mathbf{a}_i\}_{i=1,\ldots,m}} \sum_{i=1}^{m} \|\mathbf{x}_i - \Phi\mathbf{a}_i\|_2^2 + \lambda\|\mathbf{a}_i\|_1, \tag{2.2.4}$$

while restricting the atoms to have norm less than one. The optimization is carried out using an iterative approach that is composed of two (convex) steps: the sparse coding step on a fixed Φ and the dictionary update step on fixed $\mathbf{a}$.

2.3 Low rank recovery

Principal component analysis (PCA) is the most widely used statistical technique for dimensionality reduction reduction, with applications for example in computer vision, machine learning, and bioinformatics (among many others). Given a data matrix $\mathbf{X} \in \mathbb{R}^{m\times n}$ (each column of $\mathbf{X}$ is an m-dimensional data vector), it is decomposed as

$$\mathbf{X} = \mathbf{L} + \mathbf{N}, \tag{2.3.5}$$

where $\mathbf{L} \in \mathbb{R}^{m\times n}$ is a low rank matrix and $\mathbf{N} \in \mathbb{R}^{m\times n}$ is a perturbation matrix. PCA finds the smallest subspace solution to represent the data given a maximum tolerance in the least squared sense (LS), or conversely, finds the LS-optimal linear approximation for a given dimension of $\mathbf{L}$. The desired decomposition is obtained retaining the dominant eigenvectors of the data covariance matrix so that the restrictions (in LS or dimension) are met.

PCA is known to produce very good results when the perturbation is small [52]. However, its performance is highly sensitive to the presence of samples not following model (2.3.5), even a single outlier in the data matrix $\mathbf{X}$ can render the estimation of the low rank component arbitrarily far from the true matrix $\mathbf{L}$. This motivated an important amount of work dedicated to robustifying PCA, see [12, 108] and references therein. In a series of recent works [12, 126], a very elegant solution to this problem was developed, in which the low rank matrix is determined as the minimizer of a convex program. The basic idea is to add a new term in the decomposition (2.3.5) to account for the presence of outliers, $\mathbf{X} = \mathbf{L} + \mathbf{N} + \mathbf{O}$, where $\mathbf{O} \in \mathbb{R}^{m \times n}$ is an error matrix with a sparse number of non-zero coefficients with arbitrarily large magnitude. *Robust* PCA is then obtained solving

$$\min_{\mathbf{L},\mathbf{O} \in \mathbb{R}^{m \times n}} \|\mathbf{L}\|_* + \lambda \|\mathbf{O}\|_1 \qquad \text{s.t.} \quad \|\mathbf{X} - \mathbf{L} - \mathbf{O}\|_F^2 \leq \epsilon, \tag{2.3.6}$$

where $\|\mathbf{L}\|_*$ denotes the matrix nuclear norm, defined as the sum of the singular values of $\mathbf{L}$ (the convex surrogate of the rank), λ is a positive scalar parameter controlling the sparsity level in the outliers, and ϵ is a parameter controlling the error of the approximation. As the ℓ_1-norm encourages sparsity with vectors, the nuclear norm promotes low-rank in matrices. This particular formulation of robust PCA has spawned a lot of interest in the computer vision community and was successfully used in applications such as face recognition and modeling [83, 119], face recovery in video streaming [120], background modeling [85, 133], and large scale image tag transduction [74].

In this thesis, we tackle the robust PCA problem by solving the unconstrained optimization problem

$$\min_{\mathbf{L},\mathbf{O} \in \mathbb{R}^{m \times n}} \frac{1}{2} \|\mathbf{X} - \mathbf{L} - \mathbf{O}\|_F^2 + \lambda_* \|\mathbf{L}\|_* + \lambda \|\mathbf{O}\|_1 . \tag{2.3.7}$$

This formulation is equivalent to (2.3.6) in the sense that for every $\epsilon > 0$ one can find a $\lambda_* > 0$ such that (2.3.6) and (6.4.8) admit the same solutions.

Finally, many works have use the noisless formulation for the RPCA problem. In that case, the RPCA can be solved simply by minimizing the convex program

$$\min_{\mathbf{L},\mathbf{O}\in\mathbb{R}^{m\times n}} \|\mathbf{L}\|_* + \lambda \|\mathbf{O}\|_1 \text{ s.t. } \mathbf{X} = \mathbf{L} + \mathbf{O}. \tag{2.3.8}$$

3 Classification and clustering via dictionary learning with structured incoherence and shared features

3.1 Chapter summary

In this chapter we propose a new clustering framework within the sparse modeling and dictionary learning setting. Instead of searching for the set of centroid that best fit the data, as in k-means type of approaches that model the data as distributions around discrete points, we optimize for a set of dictionaries, one for each cluster, for which the signals are best reconstructed in a sparse coding manner. Thereby, we are modeling the data as a union of learned low dimensional subspaces, and data points associated to subspaces spanned by just a few atoms of the same learned dictionary are clustered together. An incoherence promoting term encourages dictionaries associated to different classes to be as independent as possible, while still allowing for different classes to share features. This term directly acts on the dictionaries, thereby being applicable both in the supervised and unsupervised settings. Using learned dictionaries for classification and clustering makes this method robust and well suited to handle large datasets. The proposed framework uses a novel measurement for the quality of the sparse representation, inspired by the robustness of the ℓ_1 regularization term in sparse coding. In the case of unsupervised classification and/or clustering,

an initialization based on combining sparse coding with spectral clustering is proposed. This initialization clusters the dictionary atoms, and therefore is based on solving a low dimensional eigen-decomposition problem, being applicable to large datasets. We first illustrate the proposed framework with examples on standard image and speech datasets in the supervised classification setting, obtaining results comparable to the state-of-the-art with this simple approach. We then present experiments for fully unsupervised clustering on extended standard datasets and texture images, obtaining excellent performance.

3.2 Introduction and basic formulation

In this work we propose a framework for clustering datasets that are well represented in the sparse modeling framework with a set of learned dictionaries. Given K clusters, we learn K dictionaries for representing the data, and then associate each signal to the dictionary for which the "best" sparse decomposition is obtained. Note that it is not that each data point belongs to a union of subspaces as for example in [29, 34]. Comparing with block/group sparsity, here a single dictionary (block) is selected per data point, and the point is sparsely represented (subspace) with atoms only from this dictionary.[1] Also in contrast with more classical subspace clustering, data points in the same class can belong to more than one subspace, since each dictionary represents a large number of subspaces (each sparsity pattern defines one subspace). The model is then very rich and non-linear.

The first building block of the proposed clustering framework is based on considering

$$\min_{\mathbf{D}_i, C_i} \sum_{i=1}^{K} \sum_{\mathbf{x}_j \in C_i} \mathscr{R}(\mathbf{x}_j, \mathbf{D}_i), \tag{3.2.1}$$

where $\mathbf{D}_i = [\mathbf{d}_1 | \mathbf{d}_2 \dots | \mathbf{d}_{k_i}] \in \mathbb{R}^{n \times k_i}$ is a dictionary of k_i atoms associated with the class

[1] Sharing atoms between the classes, and therefore having non-empty intersecting subspaces is permitted in our framework as well, see Section 3.3.1

C_i, $\mathbf{x}_j \in \mathbb{R}^n$ are the data vectors, and $\mathscr{R}$ is a function that measures how good the sparse decomposition for the signal $\mathbf{x}_j$ under the dictionary $\mathbf{D}_i$ is. In the general case, different dictionaries may have different number of atoms, k_i might be cluster dependent. This problem is closely related with the k-q-flat algorithm that aims at finding the closest k q-dimensional flats to a dataset [111]. However, there are major differences between the two. In particular, the framework here proposed, following the sparse representation approach, considers a large number of flats per class, and does not assume a pre-defined, or even constant across classes, (q) dimension, resulting in a richer space for representing and clustering the signals.

To complete the model, we add a block/dictionary incoherence term, inspired in part by the works on standard sparse coding, e.g., [21, 27, 29, 110], where it was shown that both the speed and accuracy of sparse coding techniques such as soft-thresholding and orthogonal matching pursuit depend on the incoherence between the dictionary atoms. Here we add a term $\mathscr{Q}(\mathbf{D}_i, \mathbf{D}_j)$ that promotes incoherence between the different dictionaries, thereby obtaining a general energy of the form

$$\min_{\mathbf{D}_i, C_i} \sum_{i=1}^{K} \sum_{\mathbf{x}_j \in C_i} \mathscr{R}(\mathbf{x}_j, \mathbf{D}_i) + \eta \sum_{i \neq j} \mathscr{Q}(\mathbf{D}_i, \mathbf{D}_j). \tag{3.2.2}$$

This energy will then lead to the learning of dictionaries optimized to properly represent the corresponding class, due to $\mathscr{R}$, while at the same time being weak for the other classes, due to the term $\mathscr{Q}$. We will later show how classes can still share atoms, an important property for classification algorithms [107], and a unique characteristic of our proposed model. Note that in contrast with prior work on dictionary learning for classification, this novel cross dictionary learning term $\mathscr{Q}$ is independent of the data, is intrinsic to the dictionaries being learned, thereby rendering itself also to the case of unsupervised or semi-supervised classification and clustering. For the experiments in this study we use the terms $\mathscr{Q}(\mathbf{D}_i, \mathbf{D}_j) =$

$\|\mathbf{D}_i^T \mathbf{D}_j\|_F^2$, where the subscript F denotes Frobenius norm.

We propose a measurement $\mathscr{R}$ for the quality of the sparse representation that naturally takes into account both the reconstruction error and the sparseness (complexity) of the representation on the corresponding learned dictionary. Such measurement can be applied to image patches directly or to image features, e.g., SIFT as in [127]. In practice this measurement has shown enormous discrimination power. To further show this, we performed experiments in the supervised classification setting using labeled data; we first learned a dictionary for each class (with the incoherence promoting term $\mathscr{Q}$), and then classified each testing signal according to this measure. This very simple approach gives results comparable with the state-of-the-art for several benchmark datasets. Thereby, as a by-product of our proposed clustering framework, we obtain a very simple and efficient supervised classification technique as well.

In the unsupervised clustering case, the initialization is very important for the success of the algorithm. Due to the cost associated with the procedure, repeating random initializations is practically impossible. Thus a "smart" initialization is needed. We propose an approach that combines sparse coding with spectral clustering [77], and is applicable to large datasets.

Ideas related to the ones here proposed were previously employed for subspace clustering [31, 38, 92], clustering using the so-called ℓ_1-*graph* by Huang and Yan (see description in [122]), and label propagation [16]. In contrast with our proposed dictionary learning framework, these works model all the data points in a given class as belonging to the same unique subspace, while we model them as "belonging" to the same dictionary, a richer nonlinear model since each subset of atoms from the dictionary represents a different subspace. Moreover, these very inspiring approaches all use the data itself as dictionary, sparsely representing every data point as a linear combination of the rest of the data. Such representation is computationally expensive (virtually unusable for datasets of thousands of points).

In addition, the large redundancy and coherence expected from using the data itself as dictionary is prompt to make the sparse coding very unstable: as mentioned above, it is well known that such coding techniques strongly depend on the internal coherence of the dictionary. Furthermore, the performance of these methods decreases when the number of clusters grows. We propose as part of our framework a method to bypass this problem that divides the clustering problem into several binary ones. In a natural way, we use the proposed energy function to decide which partition to choose. Such binary division framework is not so natural for these other related clustering methods.

The chapter is organized as follows. In Section 3.3 we define the measure $\mathscr{R}$ and analyze its discriminative power providing examples of supervised classification. In Section 3.4 we present the proposed clustering algorithm, together with some theoretical guarantees and experimental results. Finally, we conclude the chapter in Section 3.5.

3.3 The sparse representation quality $\hat{\mathscr{R}}$ and supervised classification

A common approach when using dictionaries for classification is to train class specific dictionaries using labeled data and then assign each testing signal to the class for which the best reconstruction is obtained [67, 84]. The measure employed for this task is often the reconstruction error, $\mathscr{R}(\mathbf{x}, \mathbf{D}) = \|\mathbf{x} - \mathbf{D}\mathbf{a}\|_2^2$, where $\mathbf{a}$ is the optimal coefficient vector in the sparse coding. While this strategy leads to very good results, it does not take into account the actual sparsity of the reconstruction. Suppose that we have two dictionaries for which almost the same reconstruction error is obtained, but one of them requires double the atoms than the other. In such a situation one would rather select the dictionary that gives the sparsest solution (simplest, following Akaike's Information Principle [1]), even if the reconstruction error is slightly larger.

dataset	proposed	data	A	B	C	SVM	k-NN
MNIST	1.26	1.35	3.41	**1.05**	-	1.4	5.0
USPS	3.98	4.14	**3.56**	4.38	6.05	4.2	5.2
ISOLET	**3.01**	3.34	4.3	3.4	-	3.3	8.7

Table 3.1: Error rate (in percentage) for the algorithm discussed in Section 3.3. The "data" column corresponds to using our discrimination function with dictionaries formed with the whole training dataset. MNIST: (A) is the best reconstructive method presented in [68], while (B) is the best discriminative one. USPS: (A) is the best reconstructive and (B) is the best discriminative method, both reported in [68]. (C) is the best result obtained in [43] (only USPS available). ISOLET: (A) is the supervised k-q-flats and (B) is the k-metrics in [104]. We also compare with an SVM with Gaussian kernel and the Euclidean k-NN.

In practice, this problem can be addressed using a small pre-defined sparsity level L in an ℓ_0 approach. This strategy is not longer valid when the convex relaxation (2.1.2) is employed (such relaxation is critical for classification tasks requiring robustness and stability). In this situation, comparing the reconstruction errors alone has little meaning. We propose then to use the actual cost function in (2.1.2) as a measure of performance, as in the dictionary learning (2.2.4), $\hat{\mathscr{R}}(\mathbf{x},\mathbf{D}) = \min_{\mathbf{a}} \|\mathbf{x}-\mathbf{D}\mathbf{a}\|_2^2 + \lambda\|\mathbf{a}\|_1$. This alternative takes into account both the reconstruction error and the complexity of the sparse decomposition. The reconstruction error measures the quality of the approximation while the complexity is measured by the ℓ_1 norm of the optimal $\mathbf{a}$.

Let $\mathbf{X}_i$, $i = 1,\ldots,K$, be a collection of K (labeled) classes of signals and $\mathbf{D}_i$ the corresponding dictionaries trained for each of them independently following for example (2.2.4). This gives, for each class, a (reconstructive) dictionary unaware of the task (classification/clustering) and of the data in the other classes. Thereby, as detailed in the introduction, it is more appropriate to add the dictionary incoherence $\mathscr{Q}(\mathbf{D}_i,\mathbf{D}_j)$, and the proposed optimization is

$$\min_{\{\mathbf{D}_i,A_i\}_{i=1\ldots K}} \sum_{i=1}^{K} \left\{ \|\mathbf{X}_i - \mathbf{D}_i\mathbf{A}_i\|_2^2 + \lambda \sum_{j=1}^{m_i} \|\mathbf{a}_i^j\|_1 \right\} + \eta \sum_{i\neq j} \left\|\mathbf{D}_i^T\mathbf{D}_j\right\|_F^2 . \tag{3.3.3}$$

Here we used the standard notation $\mathbf{A}_i = [\mathbf{a}_i^1 \ldots \mathbf{a}_i^{m_i}] \in \mathbb{R}^{k_i \times m_i}$, each column $\mathbf{a}_i^j$ is the sparse code corresponding to the signal $j \in [1..m_i]$ in class i. Note that the first term in the optimization is as in (2.2.4), where each dictionary is optimized for the data from its own class. The second term provides the coupling. In contrast with works such as [67], the coupling is between the dictionaries, the labeled data points do not form part of this term, thereby this can be used also in the non-supervised learning process, see next section.

Once the dictionaries have been learned, the class $\hat{j}_0$ for a given new signal $\mathbf{x}$ is found by solving $\hat{j}_0 = \arg\min_{j=1,\ldots,K} \hat{\mathscr{R}}(\mathbf{x}, \mathbf{D}_j)$.[2] This procedure is very simple and its few parameters can be found via cross-validation.

3.3.1 Sharing atoms

In practice, it turns out that even though we impose incoherence in the dictionaries, atoms representing common features in all classes tend to appear repeated almost exactly in dictionaries corresponding to different classes. Being so common, these atoms are used often and their associated reconstruction coefficients have a high absolute value $|\mathbf{a}_r|, r \in \{1,\ldots,k_i\}$, thus making the reconstruction costs $\hat{\mathscr{R}}(\mathbf{x}, \mathbf{D}_i)$ similar. By ignoring the coefficients associated to these common atoms when computing $\hat{\mathscr{R}}$, we can improve the discriminatory power of the system. The natural way to detect such atoms is to inspect the already available $\mathbf{D}_i^T\mathbf{D}_j$ matrices, whose absolute values represent the inner products between atoms. In the following experiments, a threshold of 0.95 consistently improves the results, sometimes significantly. Note that this procedure accounts for allowing classes to

[2] We actually obtain more than this information, since for each $\mathbf{x}$ we compute all the $\hat{\mathscr{R}}$s for all the K classes, and thereby can provide a soft classification with probabilities, or a feature vector for an SVM.

Figure 3.1: Atoms discarded due to excessive coherence. From left to right: 1 vs. 9, 3 vs. 5, 4 vs. 7, 5 vs. 8, 8 vs. 9. Notice how these atoms have learned features shared between different classes.

share features [107], and the corresponding subspaces to have intersections, in contrast for example with [31]. Figure 3.1 illustrates examples of automatically learned shared atoms in the task of learning to classify digits from the MNIST dataset.

3.3.2 Experimental results

We first test this simple classification method with standard datasets, the MNIST and USPS digit datasets and the ISOLET data that consists of 617 audio features extracted from 200 speakers saying each letter of the alphabet twice. We used in every case the usual training/testing split. In Table 3.1 we present the obtained results. We compare our results with several much more sophisticated classification algorithms. The results obtained are comparable and sometimes even better. We also compare with the standard Euclidean k-NN and with SVM with a Gaussian kernel. In all our experiments we used a penalty parameter $\lambda = 0.1$. The size of the dictionary depends on the number of training samples as well as the intrinsic complexity of the data. For MNIST, which has many samples, our best results were obtained with $k = 800$. In contrast, USPS and ISOLET have much less samples and more variability, leading to a much smaller dictionaries of size $k = 80$ and $k = 60$ respectively. These already state-of-the-art results can be further improved for example using the $\hat{\mathscr{R}}_i$ in an SVM.

One could think of using the whole training datasets as dictionaries for each class as

with the approaches mentioned in the introduction [31, 92, 122]. In that case, in all our experiments the error rates obtained are not better than the ones reported in Table 3.1. Using the data as dictionaries has the additional disadvantage that the computational cost of the classification becomes prohibitive,[3] and the method is highly susceptible to label errors due to the high coherence of the "dictionary."

Finally, we illustrate the discrimination power of the measure $\hat{\mathscr{R}}$ in a more challenging scenario using images from the Grasz02 dataset [78]. We address the object detection task by learning dictionaries for the local SIFT descriptors of an object class.

We chose the "bike" class from the Graz02 as an example, and test our proposed framework in two different weakly supervised settings. In the first setting, along with the training images, we provide the algorithm with a bounding box enclosing the bikes present in each of these images. In the second, the only supplied information is whether a bike is present or not in each of the training images. Clearly, the second case is more challenging. On each image we extract 128 dimensional SIFT descriptors from patches of 32×32 pixels computed over a grid with spacing of 4 pixels. For the first setting, we randomly pick $300,000$ SIFT descriptors from inside and outside of the bounding box respectively, and learn corresponding dictionaries with 500 atoms each. In the second setting, the bike and background dictionaries were learned from all the patches extracted from images marked as either containing a bike or purely background respectively.

In both cases, the dictionary for the class "bike" was learned iteratively, keeping the 90% of the descriptors that were more clearly assigned to the class "bike" at each iteration. This allows us to gradually discard background descriptors labeled as "bike." The choice of training and testing images was performed as it is usual for this dataset, where the first 300 images are split in two, the odd images for training, and the even images for testing.

Since classified patches overlap, each pixel in the image has several possible energy

[3] The cost of learning the dictionaries in our approach is off-line.

values $\hat{\mathscr{R}}$ for each of the two dictionaries, one per patch covering it. This spatial redundancy helps the algorithm to determine a more accurate energy value at the pixel level by means of a simple spatial average, with a Gaussian kernel, giving more weight to patches in which the pixel is closer to the center. Using a Gaussian regularization on the energy images has proven to improve the results. This is in part due to the way the ground truth masks are defined, Figure 3.2. The wheels are labeled as belonging to the class "bike" while most of the time one can see the background behind them. A strategy that considers the features globally or at several scales would help [57, 127], but this is beyond the scope of this example.

In Figure 3.2 we show the detection results obtained with this framework. We also show the corresponding precision vs. recall curve for the whole testing set. The results are very good, comparable to state-of-the-art, considering that we are using one single dictionary to categorize each of these highly complex categories. In this object localization application, the cancelation of atoms of high coherence has a crucial role because of the similarities that both classes have at the local level. If one uses directly dictionaries trained independently for each class, then most of the diagonal vertexes on the image tend to be classified as "bikes" and the opposite happens with horizontal and vertical edges, which are very frequent in urban environments.

3.4 Dictionary learning for clustering

We now proceed to extend the above dictionary learning and sparse coding frameworks to unsupervised clustering. Given a set of signals, $\{\mathbf{x}_j\}_{j=1\ldots m}$ in $\mathbb{R}^n$, and the number of clusters/classes, K,[4] we want to find the set of K dictionaries $\mathbf{D}_i \in \mathbb{R}^{n\times k_i}$, $i = 1,\ldots,K$, that best represents the data. We formulate this as an energy minimization problem of the form

[4]When K is over-estimated, a micro-detailed partition is observed.

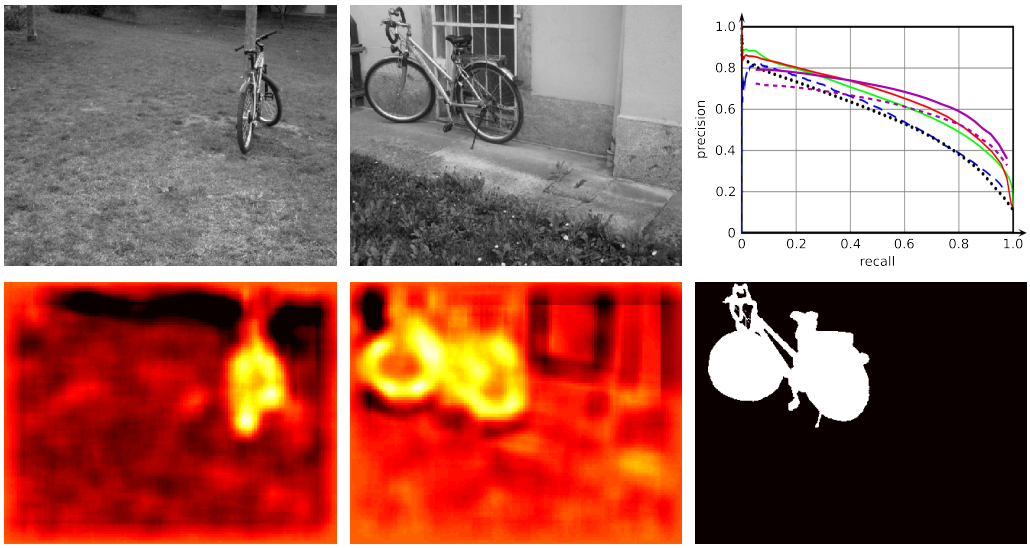


Figure 3.2: Bike detection on the Graz dataset using the measure $\hat{\mathscr{R}}$. The two leftmost columns show the obtained detection for two sample images. Bottom right: shows the ground truth for the middle row. Top right: precision vs. recall curve for several algorithms [80] (blue,dashed), [113] (black,dotted), [67] (red and green), and the proposed algorithm, using a bounding box (magenta,solid), and weakly supervised (magenta,dashed).

of Equation (3.2.2), and use the measure proposed in Section 3.3,

$$\min_{\mathbf{D}_i, C_i} \sum_{i=1}^{K} \sum_{\mathbf{x}_j \in C_i} \min_{\mathbf{a}_i^j} \|\mathbf{x}_j - \mathbf{D}_i \mathbf{a}_i^j\|_2^2 + \lambda \|\mathbf{a}_i^j\|_1 + \eta \sum_{i \neq j} \left\| \mathbf{D}_i^T \mathbf{D}_j \right\|_F^2, \tag{3.4.4}$$

where as before, the atoms of all the dictionaries are restricted to have unit norm. In contrast with (3.3.3), class assignments are unknown, and the optimization is carried out iteratively using a Lloyd's-type algorithm: *Assignment step:* The dictionaries are fixed and each signal is assigned to the cluster for which the best representation is obtained: $C_{j_0} := \left\{ \mathbf{x} : \hat{\mathscr{R}}(\mathbf{x}, \mathbf{D}_{j_0}) \leq \hat{\mathscr{R}}(\mathbf{x}, \mathbf{D}_i) \; \forall i = 1, \ldots, K \right\}$ (omitting the contribution of shared atoms). *Update step:* The new dictionaries are computed fixing the assignments found in the previous step. This is the dictionary learning problem (2.2.4), with the addition of the incoherence term.

The algorithm stops when the relative change in the energy is less than a given constant.

In practice few iterations are needed to reach good results. While the energy is being reduced at every step, there is no guarantee of arriving to a global minimum. In this setting, repeated initializations are computationally very expensive, thus a good initialization is required. This is explained next.

3.4.1 Initialization: spectral clustering meets dictionary learning

The initialization for the algorithm presented in the previous section can be given as a set of K dictionaries or as an initial partition of the data, this is the C_i sets. We propose two closely related algorithms one corresponding to each of these two alternatives. In both cases the main idea is to construct a similarity matrix and use it as the input for a spectral clustering algorithm [118].

Let $\mathbf{D}_0 \in \mathbb{R}^{n \times k_0}$ be an initial global dictionary trained (with internal incoherence) to reconstruct the data for the whole (unlabeled) set $X := [\mathbf{x}_1, \ldots, \mathbf{x}_m]$. For each signal $\mathbf{x}_j$ we have the corresponding sparse representation $\mathbf{a}_j$. Let us define $\mathbf{A} = [\mathbf{a}_1, \ldots, \mathbf{a}_m] \in \mathbb{R}^{k_0 \times m}$. Two signals belonging to the same cluster are expected to have decompositions that use similar atoms. Thus one can measure the similarity of two signals by comparing the corresponding sparse representations. Inversely, the similarity of two atoms can be determined by comparing how many signals use them simultaneously, and how they contribute, in their sparse decomposition. We compute two matrices representing each one of these cases respectively:

- *Clustering the signals:* Construct a similarity matrix $\mathbf{S}_1 \in \mathbb{R}^{m \times m}$, $\mathbf{S_1} := |\mathbf{A}^T\mathbf{A}|$.
- *Clustering the atoms:* Construct a similarity matrix $\mathbf{S}_2 \in \mathbb{R}^{k_0 \times k_0}$, $\mathbf{S_2} := |\mathbf{A}\mathbf{A}^T|$.

In both cases the similarity matrix obtained is positive semidefinite and can be associated with a graph, $G_1 := \{\mathbf{X}, \mathbf{S}_1\}$ and $G_2 := \{\mathbf{D}, \mathbf{S}_2\}$, where the data or the atoms are the sets of vertexes with the corresponding $\mathbf{S}_i$ as edge weights matrixes. This graph is partitioned

using standard spectral clustering algorithms to obtain the initialization for the algorithm described in the previous section.

As we mentioned before, G_1 is closely related with the ℓ_1-graph. In that case, the weights of the graph are determined using the sparse decomposition of the signals with the data itself as a dictionary. When the number of signals m is large, the computational cost of constructing the similarity matrix is too expensive. Also the spectral clustering algorithm requires the computation of the largest singular values (and corresponding singular vectors), which is also computationally demanding when m is large (although not so demanding if only a few eigenvectors are needed). In the case of G_2, clustering the atoms bypasses these difficulties, the size of $\mathbf{S}_2$ depends on the significantly smaller size of the initial dictionary k_0. This parameter does not depend on the amount of data, it just needs to be large enough to model it properly, and is often just in the hundreds. Note that the obtained sub-dictionaries may have different cardinalities (different k_i), reflecting different complexities of the associated clusters.

When the number of clusters, K, is large, the performance of the initial clusterization decreases. We propose a more robust initialization. Starting with the whole set as the only partition, at each iteration we subdivide in two sets each of the current partitions, keeping the division that produces the biggest decrease in the cost energy defined in Equation (3.4.4). The procedure stops when the desired number of clusters is reached. This can be applied for any of the two graphs presented in this section, and such partition is consistent with the energy driving the clustering.

3.4.2 Clustering results

We now apply the proposed algorithm to several clustering problems and texture segmentation. We first clustered the digits form 0 to 5 ($K = 6$) from the testing set of MNIST and the training set of USPS (ignoring the labels, of course). We also clustered the last six letters

Dataset	k-means	$\eta = 0$	$\eta \neq 0$
MNIST	21.2	6.9	**3.0**
USPS	22.3	2.9	**2.0**
ISOLET	20.0	6.0	**1.5**
Brodatz(x2)	-	2.5	**0.4**

Table 3.2: Error rate (in percentage) for the clustering algorithm discussed in Section 3.4. In MNIST and USPS we used digits form 0 to 5 and for ISOLET we used the last 6 letters. The result is the average performance over 10 random realizations.

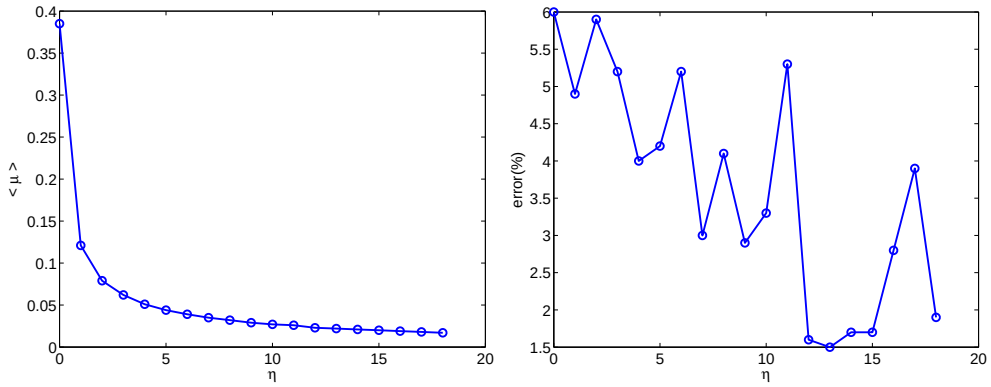


Figure 3.3: Effect of incoherence in clustering performance of the ISOLET database. Left: average incoherence between all the dictionaries vs. η. Right: classification error vs. η.

of ISOLET ($K = 6$), combining the standard training and testing sets. We further applied the clustering scheme to the combined patches of randomly chosen samples of 2 textures from the Brodatz database. The results are reported in Table 3.2 for different values of the incoherence penalty term η. The size of the initial dictionaries are $k = 120$ for USPS , $k = 300$ for MNIST and $k = 90$ for ISOLET. The dictionaries representing each cluster are of size 60, 25 and 15 respectively. The initial clustering of the data was done using spectral clustering on the graph G_1. In all the cases involving images, the atoms learned for each cluster were visually identifiable with the classes they represented.

Effect of imposing incoherence: As can be see in Table 3.2, encouraging incoherence in the dictionaries is of paramount importance, first to the initial dictionaries as internal

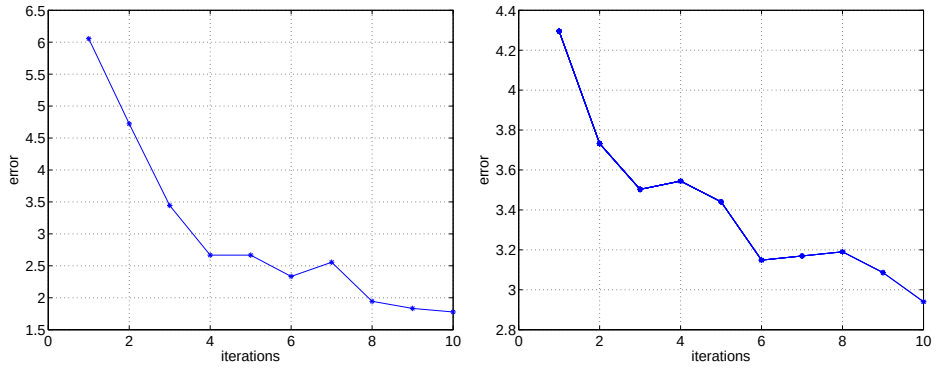


Figure 3.4: Classification error vs. number of iterations for the clustering algorithm when applied to the ISOLET (left) and USPS (right) datasets.

incoherence, and then between dictionaries during the iterative clustering. To further understand this effect, we show in Figure 3.3 how the strength of the incoherence term, controlled by the parameter η, affects the reconstruction error. Also shown is the actual average incoherence obtained between the cluster dictionaries, that is the average value of $|\mathbf{d}_i^T \mathbf{d}_j|$ between all possible pairs of atoms $\mathbf{d}_i$ and $\mathbf{d}_j$ from all the dictionaries involved.

Effect of the iterative clustering: In Figure 3.4 we show an example of how the classificaion error is monotonically reduced for succesive iterations of the proposed algorithm, thus empirically assessing its stability.

Texture segmentation: We also apply our clustering algorithm for the texture segmentation problem. The goal is to assign each pixel on an objective image to one of K possible textures. The approach is related to the one used in [84] for the supervised case. Overlapping 16×16 patches from the original images and used as input signals.

After each iteration (that is, before recomputing the dictionaries), we obtain K different energy values for each patch, each corresponding to a candidate texture class. Following the same strategy than in the object detection example of Section 3.3, we use the energy value obtained for each patch to construct K different "energy images," and combine such

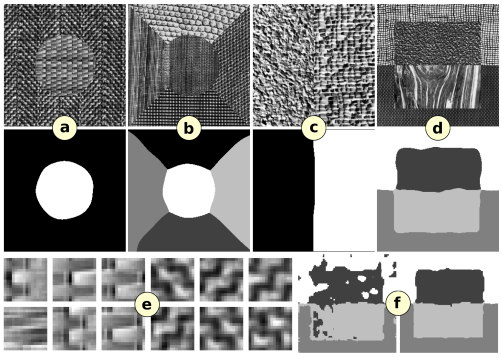


Figure 3.5: Texture segmentation results on mosaics from the Brodatz database. (a)-(d) above are the mosaics and below the respective segmentation. (f) shows the first and third iteration of the iterative clustering algorithm.

energies to obtain the pixel-wise classification.

In Figure 4.5 we show some of the results. The number of patches extracted was on the order of several thousands, so the initialization with G_2 was applied. The algorithm gave sub-dictionaires that have a cardinality that intuitively reflects the complexity of the corresponding texture (in other words, k_i was not constant). We got very low rates of missclassified pixels, for example in Figure 4.5(c), where we obtained 0.25%, which is better than the 0.37% obtained in [67] for the supervised case (which was, as far as we know, the best reported result in the literature for that image).

3.5 Conclusions

A framework for classification and clustering based on dictionary learning and sparse representations was introduced in this work. The basic idea is to simultaneously learn a set of dictionaries that optimally represent each one of the classes. Toward this goal, we introduced a new measurement of representation quality, a new term that promotes incoherence between the dictionaries, and an initialization procedure that combines sparse coding, dictionary learning, and spectral clustering. The clusters are allowed to share atoms. The

obtained model is much richer than standard subspace clustering algorithms, since multiple subspaces represent a given class, and classes can have intersecting subspaces. Soft-clustering can be obtained as well in this framework, and the experimental results can be further improved using these soft measures in an SVM.

4 Collaborative hierarchical sparse modeling

4.1 Chapter summary

Sparse modeling is a powerful framework for data analysis and processing. Traditionally, encoding in this framework is performed by solving an ℓ_1-regularized linear regression problem, commonly referred to as *Lasso* or *Basis Pursuit*. In this work we combine the sparsity-inducing property of the Lasso at the individual feature level, with the block-sparsity property of the *Group Lasso*, where sparse groups of features are jointly encoded, obtaining a sparsity pattern hierarchically structured. This results in the *Hierarchical Lasso (HiLasso)*, which shows important practical advantages. We then extend this approach to the collaborative case, where a set of simultaneously coded signals share the same sparsity pattern at the higher (group) level, but not necessarily at the lower (inside the group) level, obtaining the collaborative HiLasso model *(C-HiLasso)*. Such signals then share the same active groups, or classes, but not necessarily the same active set. This model is very well suited for applications such as source identification and separation. An efficient optimization procedure, which guarantees convergence to the global optimum, is developed for these new models. The underlying presentation of the framework and optimization approach is complemented by experimental examples and theoretical results regarding recovery guarantees.

4.2 Introduction and motivation

Sparse signal modeling has been shown to lead to numerous state-of-the-art results in signal processing, in addition to being very attractive at the theoretical level. The standard model assumes that a signal can be efficiently represented by a sparse linear combination of atoms from a given or learned dictionary. The selected atoms form what is usually referred to as the *active set*, whose cardinality is significantly smaller than the size of the dictionary and the dimension of the signal.

In recent years, it has been shown that adding structural constraints to this active set has value both at the level of representation robustness and at the level of signal interpretation (in particular when the active set indicates some physical properties of the signal); see [129, 47, 29, 123] and references therein. This leads to *group* or *structured* sparse coding, where instead of considering the atoms as singletons, the atoms are grouped, and a few groups are active at a time. An alternative way to add structure (and robustness) to the problem is to consider the simultaneous encoding of multiple signals, requesting that they all share the same active set. This is a natural collaborative filtering approach to sparse coding; see, for example, [109, 19, 14, 72, 30].

In this work we extend these approaches in a number of directions. First, we present a hierarchical sparse model, where not only a few (sparse) groups of atoms are active at a time, but also each group enjoys internal sparsity.[1] At the conceptual level, this means that the signal is represented by a few groups (classes), and inside each group only a few members are active at a time. A simple example of this is a piece of music (numerous applications in genomics and image processing exist as well), where only a few instruments are active at a time (each instrument is a group), and the sound produced by each instrument at each instant is efficiently represented by a few atoms of the sub-dictionary/group corresponding to it. Thereby, this proposed hierarchical sparse coding framework permits

[1] While we consider only 2 levels of sparsity, the proposed framework is easily extended to multiple levels.

to efficiently perform source identification and separation, where the individual sources (classes/groups) that generated the signal are identified at the same time as their representation is reconstructed (via the sparse code inside the group). An efficient optimization procedure, guaranteed to converge to the global optimum, is proposed to solve the hierarchical sparse coding problems that arise in our framework. Theoretical recovery bounds are derived, which guarantee that the output of the optimization algorithm is the true underlying signal.

Next, we go one step beyond. Continuing with the above example, if we know that the same few instruments will be playing simultaneously during different passages of the piece, then we can assume that the active groups at each instant, within the same passage, will be the same. We can exploit this information by applying the new hierarchical sparse coding approach in a collaborative way, enforcing that the same groups will be active at all instants within a passage (since they are of the same instruments and then efficiently representable by the same sub-dictionaries), while allowing each group for each music instant to have its own unique internal sparsity pattern (depending on how the sound of each instrument is represented at each instant). We propose a collaborative hierarchical sparse coding framework following this approach, *(C-HiLasso)*, along with an efficient optimization procedure. We then comment on results regarding the correct recovery of the underlying active groups.

The proposed optimization techniques for both *HiLasso* and *C-HiLasso* is based on the Proximal Method [76], more specifically, on its particular implementation for sparse problems, *Sparse Reconstruction by Separable Approximation* (SpaRSA) [124]. This is an iterative method which solves a subproblem at each iteration which, in our case, has a closed form and can be solved in linear time. Furthermore, this closed form solution combines a vector thresholding and a scalar thresholding, naturally yielding to the desired hierarchical sparsity patterns.

The rest of the chapter is organized as follows: Section 4.3 provides an introduction to

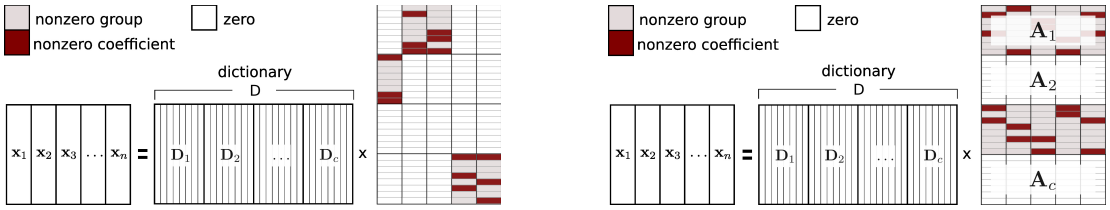


Figure 4.1: Sparsity patterns induced by HiLasso (left) and C-HiLasso (right) model selection programs.

traditional sparse modeling and presents our proposed HiLasso and C-HiLasso models. We discuss their relationship with the recent works of [47, 33, 82, 56, 49, 102]. In Section 4.4 we describe the optimization techniques applied to solve the resulting sparse coding problems and we discuss its relationship with other optimization methods recently proposed in the literature [49, 48]. We also comment on existing results regarding correct recovery of group-sparse patterns in the collaborative case. Experimental results and simulations are given in Section 4.6, and finally concluding remarks are presented in Section 4.7. Theoretical recovery guarantees for HiLasso in the noiseless setting are reported in Appendix A, demonstrating improved performance when compared with Lasso and Group Lasso.

4.3 Collaborative hierarchical sparse coding

4.3.1 The hierarchical lasso

The Group Lasso trades sparsity at the single-coefficient level with sparsity at a group level, while, inside each group, the solution is generally dense. Let us consider for example that each group is a sub-dictionary trained to efficiently represent, via sparse modeling, an instrument, a type of image, or a given class of signals in general. The entire dictionary $\mathbf{D}$ is then appropriate to represent all classes of the signal as well as mixtures of them, and Group Lasso will properly represent (dense) mixtures with one group or sub-dictionary per class. At the same time, since each class is properly represented in a sparse mode via its

corresponding group or sub-dictionary, we expect sparsity inside its groups as well (which is not achieved by Group Lasso, whose solutions are dense inside each group). This will become even more critical in the collaborative case, where signals will share groups because they are of the same class, but will not necessarily share the full active sets, since they are not the same signal. To achieve the desired in-group sparsity, we simply re-introduce the ℓ_1 regularizer together with the group regularizer, leading to the proposed *Hierarchical Lasso (HiLasso)* model,[2]

$$\min_{\mathbf{a}\in\mathbb{R}^p} \frac{1}{2} \left\|\mathbf{x}_j - \mathbf{D}\mathbf{a}\right\|_2^2 + \lambda_2 \psi_{\mathcal{G}}(\mathbf{a}) + \lambda_1 \|\mathbf{a}\|_1 . \tag{4.3.1}$$

The hierarchical sparsity pattern produced by the solutions of (4.3.1) is depicted in Figure 4.1(left). For simplicity of the description, we assume that all the groups have the same number of elements. The extension to the general case is obtained by multiplying each group norm by the square root of the corresponding group size. This model then achieves the desired effect of promoting sparsity at the group/class level while at the same time leading to overall sparse feature selection. As mentioned above, additional levels of hierarchy can be considered as well, e.g., with groups inside the blocks. This is relevant for example in audio analysis.

As with models such as Lasso and Group Lasso, the optimal parameters λ_1 and λ_2 are application and data dependent. In some specific cases, closed form solutions exist for such parameters. For example, for signal restoration in the presence of noise using Lasso ($\lambda_2 = 0$), the GSURE method provides a simple way to compute the optimal λ_1 [36]. As extending such methods to HiLasso (or the C-HiLasso model presented next) is beyond the scope of this work, we rely on cross-validation for the choice of such parameters. The selection of λ_1 and λ_2 has an important influence on the sparsity of the obtained solution. Intuitively, as λ_2/λ_1 increases, the group constraint becomes dominant and the solution tends to be more sparse at a group level but less sparse within groups (see Figure 4.2).

[2]We can similarly define a hierarchical sparsity model with ℓ_0 instead of ℓ_1.

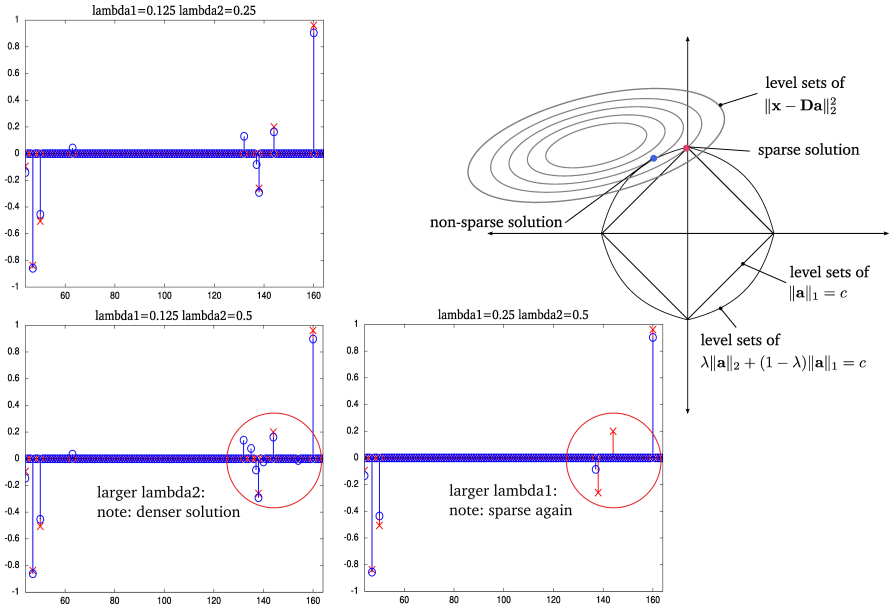


Figure 4.2: Effect of different combinations of λ_1 and λ_2 on the solutions of the HiLasso coding problem. The estimate that is closest to $\mathbf{a}_0$ in ℓ_1 norm is shown in the top left. As the ratio λ_2/λ_1 increases (bottom left), the level sets of the regularizer $\psi_{\mathcal{G}}(\cdot)$ become rounder, thus encouraging denser solutions. This is depicted in the rightmost figure for a simple case of $q = 1$ groups. Increasing λ_1 again (bottom right) increases sparsity, although here the final effect is too strong and some non-zero coefficients are not detected.

This relation allows in practice to intuitively select a set of parameters that performs well. We also noticed empirically that the selection of the parameters is quite robust, since small variations in their numerical value don't change considerably the obtained results.

Some recent modeling frameworks for sparse coding do not rely on the selection of such model parameters, e.g., following the Minimum Description Length criterion in [88], or non-parametric Bayesian techniques in [132]. Applying such techniques to the here proposed models is subject of future research.

4.3.2 Collaborative hierarchical lasso

In numerous applications, one expects that certain collections of samples $\mathbf{x}_j$ share the same active components from the dictionary, that is, that the indices of the nonzero coefficients in $\mathbf{a}_j$ are the same for all the samples in the collection. Imposing such dependency in the ℓ_1 regularized regression problem gives rise to the so called collaborative (also called "multitask" or "simultaneous") sparse coding problem [109, 72, 30, 112]. Considering the coefficients matrix $\mathbf{A} = [\mathbf{a}_1, \ldots, \mathbf{a}_n] \in \mathbb{R}^{p\times n}$ associated with the reconstruction of the samples $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n] \in \mathbb{R}^{m\times n}$, this model is given by

$$\min_{\mathbf{A}\in\mathbb{R}^{p\times n}} \frac{1}{2}\|\mathbf{X} - \mathbf{D}\mathbf{A}\|_F^2 + \lambda \sum_{k=1}^{p} \left\|\mathbf{a}^k\right\|_2, \tag{4.3.2}$$

where $\mathbf{a}^k \in \mathbb{R}^n$ is the k-th row of $\mathbf{A}$, that is, the vector of the n different values that the coefficient associated to the k-th atom takes for each sample $j = 1, \ldots, n$. If we now extend this idea to the Group Lasso, we obtain a *collaborative Group Lasso* (*C-GLasso*) formulation,

$$\min_{\mathbf{A}\in\mathbb{R}^{p\times n}} \frac{1}{2}\|\mathbf{X} - \mathbf{D}\mathbf{A}\|_F^2 + \lambda\psi_{\mathcal{G}}(\mathbf{A}), \tag{4.3.3}$$

where $\psi_{\mathcal{G}}(\mathbf{A}) = \sum_{G\in\mathcal{G}} \left\|\mathbf{A}^G\right\|_F$, and $\mathbf{A}^G$ is the sub-matrix formed by all the rows belonging to group G. This regularizer is the natural collaborative extension of the regularizer in (2.1.3).

In this work, we take an additional step and treat this together with the hierarchical extension presented in the previous section. The combined model that we propose, *C-HiLasso*, is given by

$$\min_{\mathbf{A}\in\mathbb{R}^{p\times n}} \frac{1}{2}\|\mathbf{X} - \mathbf{D}\mathbf{A}\|_F^2 + \lambda_2\psi_{\mathcal{G}}(\mathbf{A}) + \lambda_1 \sum_{j=1}^{n} \left\|\mathbf{a}_j\right\|_1. \tag{4.3.4}$$

The sparsity pattern obtained using (4.3.4) is shown in Figure 4.1(right). The C-GLasso is a particular case of our model when $\lambda_1 = 0$. On the other hand, one can obtain independent

Lasso solutions for each $\mathbf{x}_i$ by setting $\lambda_2 = 0$. We see that (4.3.4) encourages all the signals to share the same groups (classes), while the active set inside each group is signal dependent. We thereby obtain a collaborative hierarchical sparse model, with collaboration at the class level (all signals collaborate to identify the classes), and freedom at the individual levels inside the class to adapt to each particular signal. This new model is particularly well suited, for example, when the data vectors have missing components. In this case combining the information from all the samples is very important in order to obtain a correct representation and model (group) selection. This can be done by slightly changing the data term in (4.3.4). For each data vector $\mathbf{x}_j$ one computes the reconstruction error using only the observed elements. Note that the missing components do not affect the other terms of the equation. Examples will be shown in Section 4.6.

4.3.3 Relationship to recent literature

A number of recent works have addressed hierarchy, grouping and collaboration within the sparse modeling community. We now discuss the ones most closely related to the proposed HiLasso and C-HiLasso models.

In [47], the authors propose a general framework in which one can define a regularization term to encourage a variety of sparsity patterns, and provide theoretical results (different to the ones developed here) for the single-signal case. The HiLasso model presented here, in the single signal scenario, can be seen as a particular case of that model (where the groups in [47] should be blocks and singletons), although the particularly and important case of hierarchical structure introduced here is not mentioned in that paper. In [33] the authors simultaneously (see [100]) proposed a model that coincides with ours again in the single-signal scenario. None of these approaches develop the collaborative framework introduced here, nor the theoretical guarantees. The recovery of mixed signals with ℓ_0 optimization was addressed in [102]. This model does not include block sparsity

(no hierarchy), collaboration, or the theoretical results we obtain here.

The special case of C-HiLasso when $\lambda_1 = 0$, C-GLasso, is investigated in [7], where a theoretical analysis of the signal recovery properties of the model is developed. Collaborative coding with structured sparsity has also been used recently in the context of gene expression analysis [82, 56]. In [82], the authors propose a model, that can be interpreted as a particular case of the collaborative approach presented here, in which a set of signals is simultaneously coded using a small (sparse) number of atoms of the dictionary. They modify the classical collaborative sparse coding regularization so that each signal can use any subset of the detected atoms. This is equivalent to our model when the groups have only one element and therefore there is no hierarchy in the coding. A collaborative model is presented in [56], where signals sharing the same active atoms are grouped together in a hierarchical way by means of a tree structure. The regularization term proposed is analogous to the one proposed in our work, but it is used to group signals rather than atoms (features), having once again no hierarchical coding.

Tree-based sparse coding has also been used recently to learn dictionaries [49, 48]. Under this model, if a particular learned atom is not used in the decomposition of a signal, then none of its descendants (in terms of the given tree structure) can be used. Although not explicitly considered in these works, the HiLasso model is an important particular case, among the wide spectrum of hierarchical sparse models considered in this line of work, where the hierarchy has two levels and no single atoms are in the upper level.

To conclude, while particular instances of the proposed C-HiLasso have been recently reported in the literature, none of them are as comprehensive. C-HiLasso includes both collaboration, at a block/group level, and hierarchical coding. Such collaborative hierarchical structure is novel and fundamental to address new important problems such as collaborative source identification and separation. The new theoretical results presented here extend the block sparsity results of [29, 28], complementing the modeling and algorithmic work.

4.4 Optimization

4.4.1 Single-signal problem: HiLasso

In the last decade, optimization of problems of the form of (2.1.2) and (2.1.3) have been deeply studied, and there exist very efficient algorithms for solving them. Recently, Wright et. al [124] proposed a framework, SpaRSA, for solving the general problem

$$\min_{\mathbf{a}\in\mathbb{R}^p} f(\mathbf{a}) + \lambda\psi(\mathbf{a}). \tag{4.4.5}$$

be a smooth and convex function, while $\psi : \mathbb{R}^p \to \mathbb{R}$ only needs to be finite and convex in $\mathbb{R}^p$. This formulation, which is a particular case of the Proximal Method framework developed by Nesterov [76], includes as important particular cases the Lasso, Group-Lasso and HiLasso problems by setting $f(\cdot)$ as the reconstruction error and then choosing the corresponding regularizers for $\psi(\cdot)$. When the regularizer, $\psi(\cdot)$, is group separable, the optimization can be subdivided into smaller problems, one per group. The framework becomes powerful when these sub-problems can be solved efficiently. This is the case of the Lasso and Group Lasso (with non overlapping groups) settings, and also of the HiLasso, as we will show later in this Section. In all cases, the solution of the sub-problems are obtained in linear time.

The SpaRSA algorithm generates a sequence of iterates $\{\mathbf{a}^{(t)}\}_{t\in\mathbb{N}}$ that, under certain conditions, converges to the solution of (4.4.5). At each iteration, $\mathbf{a}^{(t+1)}$ is obtained by solving

$$\min_{\mathbf{z}\in\mathbb{R}^p} (\mathbf{z}-\mathbf{a}^{(t)})^\intercal \nabla f(\mathbf{a}^{(t)}) + \frac{\alpha^{(t)}}{2}\left\|\mathbf{z}-\mathbf{a}^{(t)}\right\|_2^2 + \lambda\psi(\mathbf{z}), \tag{4.4.6}$$

for a sequence of parameters $\{\alpha^{(t)}\}_{t\in\mathbb{N}}$, $\alpha^{(t)} = \alpha_0\eta^t$, where $\alpha_0 > 0$ and $\eta > 1$ need to be chosen properly for the algorithm to converge (see [124] for details). It is easy to show

that (4.4.6) is equivalent to

$$\min_{\mathbf{z}\in\mathbb{R}^p} \frac{1}{2}\left\|\mathbf{z}-\mathbf{u}^{(t)}\right\|_2^2 + \frac{\lambda}{\alpha^{(t)}}\psi(\mathbf{z}), \tag{4.4.7}$$

where $\mathbf{u}^{(t)} = \mathbf{a}^{(t)} - \frac{1}{\alpha^{(t)}}\nabla f(\mathbf{a}^{(t)})$. In this new formulation, it is clear that the first term in the cost function can be separated element-wise. Thus, when the regularization function $\psi(\mathbf{z})$ is group separable, so is the overall optimization, and one can solve (4.4.7) independently for each group, leading to

$$\mathbf{a}_{[G]}^{(t+1)} = \arg\min_{\mathbf{z}\in\mathbb{R}^{|G|}} \frac{1}{2}\left\|\mathbf{z}-\mathbf{u}_{[G]}^{(t)}\right\|_2^2 + \frac{\lambda}{\alpha^{(t)}}\psi_{\mathcal{G}}(\mathbf{z}),$$

$\mathbf{z}_{[G]}$ being the corresponding variable for the group. In the case of HiLasso, this becomes,

$$\mathbf{a}_{[G]}^{(t+1)} = \arg\min_{\mathbf{z}\in\mathbb{R}^{|G|}} \frac{1}{2}\|\mathbf{z}-\mathbf{w}\|_2^2 + \frac{\lambda_2}{\alpha^{(t)}}\|\mathbf{z}\|_2 + \frac{\lambda_1}{\alpha^{(t)}}\|\mathbf{z}\|_1, \tag{4.4.8}$$

where we have defined $\mathbf{w} = \mathbf{u}_{[G]}^{(t)}$. Problem (4.4.8) is a second order cone program (SOCP), for which one could use generic solvers. However, since it needs to be solved many times within the SpaRSA iterations, it is crucial to solve it efficiently. It turns out that (4.4.8) admits a closed form solution with cost linear in the dimension of $\mathbf{w}$. By inspecting the subgradient of (4.4.8) for the case where the optimum $\mathbf{z}^* \neq 0$,

$$\mathbf{w} - \left(1 + \frac{\tilde{\lambda}_2}{\|\mathbf{z}^*\|_2}\right)\mathbf{z}^* \in \tilde{\lambda}_1 \partial \left\|\mathbf{z}^*\right\|_1,$$

where we have defined $\tilde{\lambda}_2 = \lambda_2/\alpha^{(t)}$ and $\tilde{\lambda}_1 = \lambda_1/\alpha^{(t)}$. If we now define $C(\mathbf{z}^*) = 1 + \tilde{\lambda}_2/\|\mathbf{z}^*\|_2$, we observe that each element of $C(\mathbf{z}^*)\mathbf{z}^*$ is the solution of the well known scalar soft thresholding operator,

$$z_i^* = \frac{1}{C(\mathbf{z}^*)}\operatorname{sgn}(w_i)\max\{0, |w_i| - \tilde{\lambda}_1\} = \frac{h_i}{C(\mathbf{z}^*)}, \quad i = 1, \ldots, g, \tag{4.4.9}$$

where we have defined $h_i = \text{sgn}(w_i)\max\{0, |w_i| - \tilde{\lambda}_1\}$, the result of the scalar thresholding of $\mathbf{w}$. Taking squares on both sides of (4.4.9) and summing over $i = 1, \ldots, g$ we obtain

$$\left\|\mathbf{z}^*\right\|_2^2 = C^2(\mathbf{z}^*)\,\|\mathbf{h}\|_2^2 = \frac{\|\mathbf{z}^*\|_2^2}{(\|\mathbf{z}^*\|_2 + \tilde{\lambda}_2)^2}\,\|\mathbf{h}\|_2^2,$$

from which the equality $\|\mathbf{z}^*\|_2 = \|\mathbf{h}^*\|_2 - \tilde{\lambda}_2$ follows. Since all terms are positive, this can only hold as long as $\|\mathbf{h}^*\|_2 > \tilde{\lambda}_2$, which gives us a vector thresholding condition on the solution $\mathbf{z}^*$ in terms of $\|\mathbf{h}\|_2$. It is easy to show that $\|\mathbf{h}^*\|_2 \leq \tilde{\lambda}_2$ is a sufficient condition for $\mathbf{z}^* = 0$. Thus we obtain,

$$\mathbf{a}_{[G]}^{(t+1)} = \begin{cases} \frac{\max\{0, \|\mathbf{h}\|_2 - \tilde{\lambda}_2\}}{\|\mathbf{h}\|_2}\mathbf{h} & , \quad \|\mathbf{h}\|_2 > 0 \\ \mathbf{0} & , \quad \|\mathbf{h}\|_2 = 0. \end{cases} \tag{4.4.10}$$

The above expression requires g scalar thresholding operations, and one vector thresholding, which is also linear with respecto to the group size g. Therefore, for all groups, the cost of solving the subproblem (4.4.8) is linear in m, the same as for Lasso and Group Lasso. The complete HiLasso optimization algorithm is summarized in Algorithm 1. The parameter η has very little influence in the overall performance (see [124] for details); we used $\eta = 2$ in all our experiments. Note that, as expected, the solution to the sub-problem for the cases $\lambda_2 = 0$ or $\lambda_1 = 0$, corresponds respectively to scalar soft thresholding and vector soft thresholding. In particular, when $\lambda_2 = 0$, the proposed optimization reduces to the Iterative Soft Thresholding algorithm [21].

4.4.2 Optimization of the collaborative HiLasso

The multi-signal (collaborative) case is equivalent to the one-dimensional case where the signal is a concatenation of the columns of $\mathbf{X}$, and the dictionary is an $nm \times np$ block-diagonal matrix, where each of the n blocks is a copy of the original dictionary $\mathbf{D}$. However,

Algorithm 1: HiLasso optimization algorithm.

Input: Data $\mathbf{X}$, dictionary $\mathbf{D}$, group set $\mathcal{G}$, constants $\alpha_0 > 0$, $\eta > 1$, $c > 0$, $0 < \alpha_{\min} < \alpha_{\max}$
Output: The optimal point $\mathbf{a}^*$
Initialize $t := 0, \mathbf{a}^{(0)} := \mathbf{0}$;
while *stopping criterion is not satisfied* **do**
  choose $\alpha^{(t)} \in [\alpha_{\min}, \alpha_{\max}]$;
  set $\mathbf{u}^{(t)} := \mathbf{a}^{(t)} - \frac{1}{\alpha^{(t)}} \nabla f(\mathbf{a}^{(t)})$;
  while *stopping criterion is not satisfied* **do**
    // Here we use the group separability of (4.4.7) *and solve* (4.4.8) *for each group*
    for $i := 1$ *to* q **do**
      Compute $\mathbf{a}_{[G]}^{(t+1)}$ as the solution to (4.4.10);
    end
    set $\alpha^{(t+1)} := \eta \alpha^{(t)}$;
  end
  set $t := t + 1$;
end

in practice, it is not needed to build such (possibly very large) dictionary, and we can operate directly with the matrices $\mathbf{D}$ and $\mathbf{X}$ to find $\mathbf{A}$. If we define the matrix $\mathbf{U}^{(t)} \in \mathbb{R}^{m \times n}$ whose i-th column is given by $\mathbf{u}_i^{(t)} = \mathbf{a}_i^{(t)} - \frac{1}{\alpha^{(t)}} \nabla f(\mathbf{a}_i^{(t)})$, we get the following SpaRSA iterates,

$$\mathbf{A}^{(t+1)} = \arg\min_{\mathbf{Z} \in \mathbb{R}^{m \times n}} \frac{1}{2} \left\| \mathbf{Z} - \mathbf{U}^{(t)} \right\|_F^2 + \frac{\lambda_2}{\alpha^{(t)}} \|\mathbf{Z}\|_F + \frac{\lambda_1}{\alpha^{(t)}} \sum_{j=1}^{n} \left\| \mathbf{z}_j \right\|_1 ,$$

which again is group separable, so that it can be solved as q independent problems in the corresponding bands of $\mathbf{U}^{(t)}$,

$$(\mathbf{A}^{(t+1)})^G = \arg\min_{\mathbf{Z} \in \mathbb{R}^{g \times n}} \frac{1}{2} \left\| \mathbf{Z} - (\mathbf{U}^{(t)})^G \right\|_F^2 + \frac{\lambda_2}{\alpha^{(t)}} \|\mathbf{Z}\|_F + \frac{\lambda_1}{\alpha^{(t)}} \sum_{j=1}^{n} \left\| \mathbf{z}_j \right\|_1 .$$

The correspondent closed form solutions for these subproblems, which are obtained in an analogous way to (4.4.9)–(4.4.10), are given by

$$(\mathbf{A}^{(t+1)})^G = \begin{cases} \frac{\max\{0, \|\mathbf{H}\|_F - \tilde{\lambda}_2\}}{\|\mathbf{H}\|_F} \mathbf{H} & , \quad \|\mathbf{H}\|_F > 0 \\ \mathbf{0} & , \quad \|\mathbf{H}\|_F = 0 \end{cases} , \quad h_{ij} = \operatorname{sgn}(w_{ij}) \max\{0, |w_{ij}| - \tilde{\lambda}_1\}, \tag{4.4.11}$$

and we have defined $\mathbf{W} := (\mathbf{U}^{(t)})^G$.

As mentioned in Section 4.3.3, [48] addresses a wide spectrum of hierarchical sparse models for coding and dictionary learning. They propose a proximal method optimization procedure that, when restricted to the formulation of HiLasso, is very similar to the one developed in Section 4.4.1. The main difference with our method is that they solve the sub-problem (4.4.7) using a dual approach (based on conic duality) that finds the exact solution in a finite number of operations. Our method, being tailored to the specific case of HiLasso, provides such solution in closed form, requiring just two thresholdings, both linear in the dimension of $\mathbf{X}$, $n \times m$.

4.5 Recovery guarantees

As with the traditional sparse regression problems, Lasso and Group Lasso, it is of special interest to know under which conditions the HiLasso will be able to recover the true underlying sparse vector of coefficients $\mathbf{a}$ given an observation $\mathbf{x}$. This is of paramount importance to the source identification and classification problems, where one is interested more in $\mathbf{a}$ and its active set rather than in recovering $\mathbf{a}$. This has been carefully studied in Appendix A, where we provide conditions for the existence of unique solutions of the HiLasso model and for the correct recovery of the active set.

4.6 Experimental results

In this section we show the strength of the proposed HiLasso and C-HiLasso models. We start by comparing our model with the standard Lasso and Group Lasso using synthetic data. We created q dictionaries, $\mathbf{D}_r, r = 1, \ldots, q$, with $g = 64$ atoms of dimension $m = 64$, and i.i.d. Gaussian entries. The columns were normalized to have unit ℓ_2 norm. We then randomly chose $k = 2$ groups to be active at each time (on all the signals). Sets of $n = 200$

normalized testing signals were generated, one per active group, as linear combinations of $s \ll 64$ elements of the active dictionaries, $\mathbf{x}_j^r = \mathbf{D}_r \mathbf{a}_j^r$. The mixtures were created by summing these signals and (eventually) adding Gaussian noise of standard deviation σ. The generated testing signals have a hierarchical sparsity structure and while they share groups, they do not necessarily share the sparsity pattern inside the groups. We then built a single dictionary by concatenating the sub-dictionaries, $\mathbf{D} = [\mathbf{D}_1, \ldots, \mathbf{D}_q]$, and used it to solve the Lasso, Group Lasso, HiLasso and C-HiLasso problems. Table 4.1 summarizes the Mean Squared Error (MSE) and Hamming distance of the recovered coefficient vectors $\mathbf{a}_j, j = 1, \ldots, n$. We observe that our model is able to exploit the hierarchical structure of the data as well as the collaborative structure. Group Lasso selects in general the correct blocks but it does not give a sparse solution within them. On the other hand, Lasso gives a solution that has nonzero elements belonging to groups that were not active in the original signal, leading to a wrong model/class selection. HiLasso gives a sparse solution that picks atoms from the correct groups but still presents some minor mistakes. For the collaborative case, in all the tested configurations, no coefficients were selected outside the correct active groups, and the recovered coefficients are consistently the best ones.

In all the examples, and for each method, the regularization parameters were the ones for which the best results were obtained. One can scale the parameter λ_2 to account for different number of signals. This situation is analogous to a change in the size of the dictionary, thus, λ_2 should be proportional to the square root of the number of signals to code.

We then experimented with the USPS digits dataset, which has been shown to be well represented in the sparse modeling framework [89]. Here the signals are vectors containing the unwrapped gray intensities of 16×16 images ($m = 256$). We obtained each of the $n = 200$ samples in the testing data set as the mixture of two randomly chosen digits, one from each of the two drawn sets of digits. In this case we only have ground truth

$\sigma = 0.1$	417 / 22.0 330 / 19.8	1173 / 361.6 **163** / **13.3**
$\sigma = 0.2$	564 / 21.6 399 / 22.7	1182 / 378.3 **249** / **17.1**
$\sigma = 0.4$	965 / 22.7 656 / **19.5**	1378 / 340.3 **595** / 27.4

$s = 8$	388 / 22.0 272 / 19.5	1184 / 318.2 **96** / **16.2**
$s = 12$	1200 / 36.2 704 / **26.5**	1166 / 350.4 **413** / 29.1
$s = 16$	1641 / 43.9 1100 / **32.2**	1093 / 338.6 **551** / 35.0

$q = 4$	1080 / 27.8 1009 / **29.8**	1916 / 221.7 **742** / 30.2
$q = 8$	1200 / 36.2 704 / **26.5**	1166 / 350.4 **413** / 29.1
$q = 12$	1030 / 41.8 662 / **26.4**	840 / 447.7 **4** / 29.8

Table 4.1: Simulated signal results. In every table, each 2×2 cell contains the MSE ($\times 10^4$) and Hamming distance (MSE/Hamming) for Lasso (top,left), GLasso (top,right), HiLasso (bottom,left) and C-HiLasso (bottom,right). In the first case (left) we vary the noise σ while keeping $q = 8$ and $s = 8$ fixed. In the second and third cases we have $\sigma = 0$. For the second experiment (center) we fixed $q = 8$ while changing s. In the third case we fix $s = 12$ and vary the number of groups q.

at the group level. We measure the recovery performance in terms of the average MSE of the recovered signals, AMSE $= \frac{1}{nq}\sum_{r=1}^{q}\sum_{j=1}^{n}\left\|\mathbf{x}_j^r - \hat{\mathbf{x}}_j^r\right\|_2^2$, where $\mathbf{x}_j^r$ is the component corresponding to source r in the signal j, and $\hat{\mathbf{x}}_j^r$ is the recovered one.

Using the usual training-testing split for USPS, we first learned a dictionary for each digit. We then created a single dictionary by concatenating them. In Table 4.2 we show the AMSE obtained while summing $k = 2$ different digits. We also consider the situation were only one digit is present. C-HiLasso automatically detects the number of sources while achieving the best recovery performance. As in the synthetic case, only the collaborative method was able to successfully detect the true active classes. In Figure 4.3 we relax the assumption that all the signals have to contain exactly the same type and amount of classes in the mixture, further demonstrating the flexibility of the proposed C-HiLasso model. We used a set containing 180 mixtures of digit images. The first 150 images are obtained as the sum/mixture of a number “3” and an number “5” (randomly selected). Each of the last 30 images in the set are the mixture of three numbers: “3” ,“5” and “7” (the 180

experiment	Lasso		GLasso		HiLasso		C-GLasso		C-HiLasso	
	AMSE	Hamm	AMSE	Hamm	AMSE	Hamm	AMSE	Hamm	AMSE	Hamm
1 digit	0.06	0.43	0.07	0.78	0.02	0.19	**0.01**	**0.02**	0.02	0.06
1 digit+n	0.08	1.31	0.08	0.87	0.04	0.48	0.05	0.25	**0.02**	**0.01**
2 digit	0.09	1.46	0.08	1.86	0.02	1.18	**0.01**	**0.74**	0.02	0.90
2 digit+n	0.11	2.21	0.08	1.99	0.04	1.46	0.09	1.60	**0.03**	**0.70**

Table 4.2: Noisy digit mixtures results. Four different cases are shown: when each signal is a single digit and when it is the mixture of two different (randomly selected) digits, with and without additive Gaussian noise with standard deviation 10% of the peak value. See also Figure 4.3.

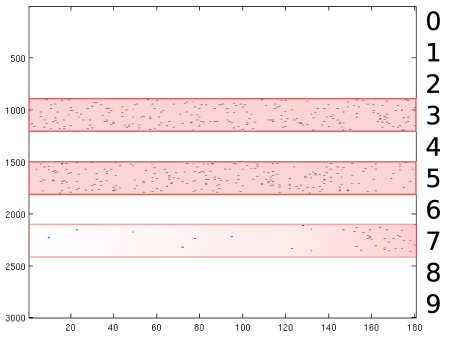


Figure 4.3: In this example we used C-HiLasso to analyze mixtures where the data set contains different number and types of sources/classes.

images are of course presented at random, the algorithm is not a priori aware which images contain 2 sources and which contain 3). The figure shows the active sets of the recovered coefficients matrix $\mathbf{A}$ as a binary matrix the same size as $\mathbf{A}$ (atom indices in the vertical and sample indices in the horizontal), where black dots indicate nonzero coefficients. C-HiLasso managed to identify the active blocks while the sub-dictionary corresponding to "7" is mostly active for the last 30 images. The accuracy of this result depends on the relationship between the sub-dictionaries corresponding to each digit.

We also used the digits dataset to experiment with missing data. We randomly discarded an average of 60% of the pixels per mixed image and then applied C-Hilasso. The algorithm is capable of correctly detecting which digits are present in the images. Some

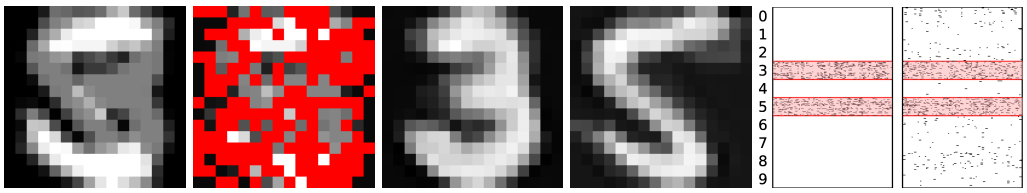


Figure 4.4: Example of recovered digits (3 and 5) from a mixture with 60% of missing components. From left to right: noiseless mixture, observed mixture with missing pixels highlighted in red, recovered digits 3 and 5, and active set recovered for all samples using the C-HiLasso and Lasso respectively. In the last two figures, the active sets are represented as in Figure 4.3.

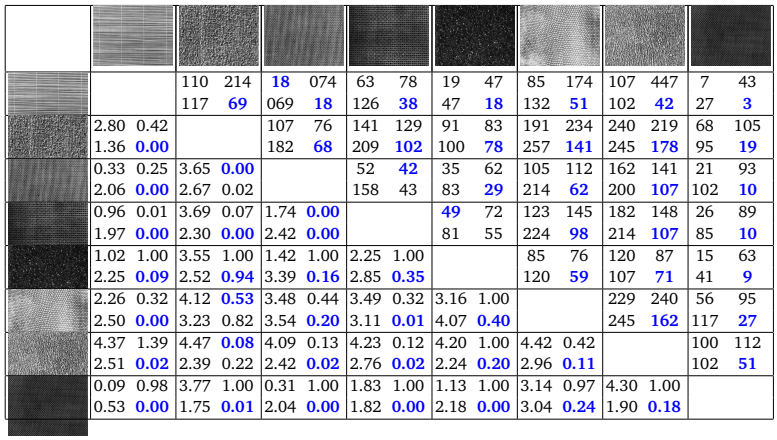
example results for this case are shown in Figure 4.4. Note that this is a quite different problem than the one commonly addressed in the matrix completion literature. Here we do not aim to recover signals that all belong to a unique unknown subspace, but signals that are the combination of two non-unique spaces to be automatically identified from the available dictionary. Such unknown spaces have common models/groups for all the signals in question (the coarse level of the hierarchy), but not necessarily the exact same atoms inside the groups and therefore do not necessarily belong to the same subspaces. Both levels of the hierarchy are automatically detected, e.g., the groups corresponding to "3" and "5," and the corresponding reconstructing atoms (subspaces) in each group, these last ones possibly different for each signal in the set. While we consider that the possible subspaces are to be selected from the provided dictionary (learned off-line from training data), in Section 4.7 we discuss learning such dictionaries as part of the optimization as well (see also [132, 95]). In such cases, the standard matrix completion problem becomes a particular case of the C-HiLasso framework (with a single group and all the signals having the same active set, subspace, in the group), naturally opening numerous theoretical questions for this new more general model.[3]

We also compared the performance of C-HiLasso, Lasso, GLasso and C-GLasso (without

[3]Prof. Carin and collaborators have new results on the case of a single group and signals in possible different subspaces of the group, an intermediate model between standard matrix completion and C-HiLasso (personal communication).

hierarchy) in the task of separating mixed textures in an image. In this case, the set of signals $\mathbf{X}$ corresponds to all 12×12 patches in the (single) image to be analyzed. We chose 8 textures from the Brodatz dataset and trained one dictionary for each one of them using one half of the respective images (these form the $g = 8$ groups of the dictionary). Then we created an image as the sum of the other halves of the $k = 2$ textures. One can think of this experiment as a generalization to the texture separation problem proposed in [96] (without additive noise), where only two textures are present. The experiment was repeated for all possible combinations of two textures from the 8 possible ones. The results are summarized in Table 4.3. A detailed example is shown in Figure 4.5. For each algorithm, the best parameters were chosen using grid search, ensuring that those were not in the edges of the grid. For Lasso and C-HiLasso the best λ_1 is 0.0625. For GLasso and C-GLasso, the best λ_2 was, respectively, 0.05 and 75 (for the collaborative setting, we heuristically scale λ_2 with the number of signals as $\sqrt{n}$. In this experiment, $n \approx 512^2$, leading to such large value of λ_2). From Table 4.3 we can conclude that the C-HiLasso is significantly better than the competing algorithms, both in the MSE of the recovered signals (we show the AMSE of recovering both active signals), and in the average Hamming distance between the recovered group-wise active sets and the true ones. In the latter case we observe that, in many cases, the C-HiLasso active set recovery performance is perfect (Hamming distance 0) or near perfect, whereas the other methods seldom approach a Hamming distance lower than 1.

Finally, we use C-HiLasso to automatically identify the sources present in a mixture of audio signals [99]. The goal is to identify the speakers talking simultaneously on a single recording. Here the task is not to fully reconstruct each of the unmixed sources from the observed signal but to identify which speakers are active. In this case, since the original sources do not need to be recovered, the modeling can be done in terms of features extracted from the original signals in a linear but non-bijective way.



		110	214	**18**	074	63	78	19	47	85	174	107	447	7	43
		117	**69**	069	**18**	126	**38**	47	**18**	132	**51**	102	**42**	27	**3**
2.80	0.42			107	76	141	129	91	83	191	234	240	219	68	105
1.36	**0.00**			182	**68**	209	**102**	100	**78**	257	**141**	245	**178**	95	**19**
0.33	0.25	3.65	**0.00**			52	**42**	35	62	105	112	162	141	21	93
2.06	**0.00**	2.67	0.02			158	43	83	**29**	214	**62**	200	**107**	102	**10**
0.96	0.01	3.69	0.07	1.74	**0.00**			**49**	72	123	145	182	148	26	89
1.97	**0.00**	2.30	**0.00**	2.42	**0.00**			81	55	224	**98**	214	**107**	85	**10**
1.02	1.00	3.55	1.00	1.42	1.00	2.25	1.00			85	76	120	87	15	63
2.25	**0.09**	2.52	**0.94**	3.39	**0.16**	2.85	**0.35**			120	**59**	107	**71**	41	**9**
2.26	0.32	4.12	**0.53**	3.48	0.44	3.49	0.32	3.16	1.00			229	240	56	95
2.50	**0.00**	3.23	0.82	3.54	**0.20**	3.11	**0.01**	4.07	**0.40**			245	**162**	117	**27**
4.37	1.39	4.47	**0.08**	4.09	0.13	4.23	0.12	4.20	1.00	4.42	0.42			100	112
2.51	**0.02**	2.39	0.22	2.42	**0.02**	2.76	**0.02**	2.24	**0.20**	2.96	**0.11**			102	**51**
0.09	0.98	3.77	1.00	0.31	1.00	1.83	1.00	1.13	1.00	3.14	0.97	4.30	1.00		
0.53	**0.00**	1.75	**0.01**	2.04	**0.00**	1.82	**0.00**	2.18	**0.00**	3.04	**0.24**	1.90	**0.18**		

Table 4.3: Texture separation results. The rows and columns indicate the active textures in each cell. The upper triangle contains the AMSE ($\times 10^4$) results, while the lower triangle shows the Hamming error in the group-wise active set recovery. Within each cell, results are shown for the Lasso (top left), Group Lasso (bottom left), Collaborative Group Lasso (top right) and Collaborative Hierarchical Lasso (bottom right). The best results are in blue bold.

Audio signals have in general very rich structures and their properties rapidly change over time. A natural approach is to decompose them into a set of overlapping local time-windows, where the properties of the signal remain stable. There is a straightforward analogy with the approach explained above for the texture segmentation case, where images were decomposed into collections of overlapping patches. These time-windows will collaborate in the identification.

A challenging aspect when identifying audio sources is to obtain features that are specific to each source and at the same time invariant to changes in the fundamental frequency (pitch) of the sources. In the case of speech, a common choice is to use the short-term power spectrum envelopes as feature vectors [86] (refer to [99] for details on the feature extraction process and implementation). The spectral envelope in human speech varies along time, producing different patterns for each phoneme. Thus, a speaker does not produce an unique spectral envelope, but a set of spectral envelopes that live in a union of manifolds. Since such manifolds are well represented by sparse models, the problem of speaker

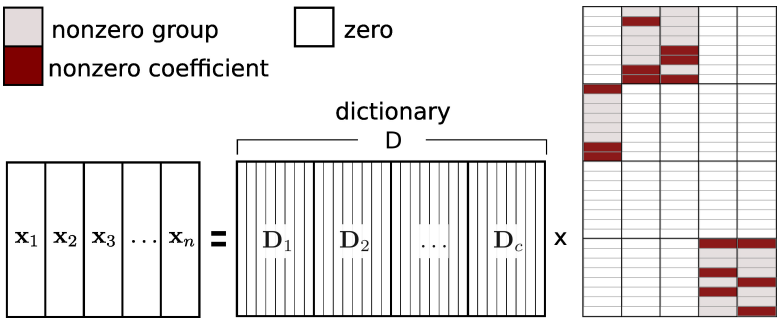


Figure 4.5: Texture separation results. Left to right: sample mixture, corresponding C-HiLasso separated textures, and comparison of the active set diagrams obtained by the Lasso (as in Figure 4.4).

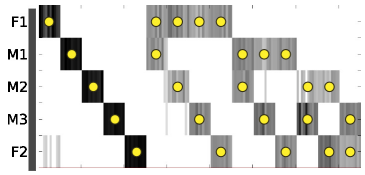


Figure 4.6: Speaker identification results. Each column corresponds to the sources identified for a specific time frame, the true ones marked by yellow dots. The vertical axis indicates the estimated activity of the different sources, where darker colors indicate higher energy.

identification is well suited for the proposed C-HiLasso framework, where each block in the dictionary is trained for the features corresponding to a given speaker, and the overlapping time-windows collaborate in detecting the active blocks.

For this experiment we use a dataset consisting of recordings of five different German radio speakers, two female and three male. Each recording is six minutes long. One quarter of the samples were used for dictionary training, and the rest for testing. For each speaker, we learned a sub-dictionary from the training dataset. For testing, we extracted 10 non-overlapping frames of 15 seconds each (including silences made by the speakers while talking), and encoded them using C-HiLasso. The experiment was repeated for all possible combinations of two speakers, and all the speakers talking alone. The results are presented

in Figure 4.6. C-HiLasso manages to detect automatically the number of sources very accurately, as well as the actual active speakers. Again, refer to [99] for comparisons with other sparse modeling methods (showing the clear advantage of C-HiLasso) and results obtained for the identification of wind instruments in musical recordings.

4.7 Conclusions

In this chapter, we introduced a new framework of collaborative hierarchical sparse coding, where multiple signals collaborate in their encoding, sharing code groups (models) and having (possible disjoint) sparse representations inside the corresponding groups. An efficient optimization approach was developed, which guarantees convergence to the global minimum, and examples illustrating the power of this framework were presented. At the practical level, we are currently continuing our work on the applications of this proposed framework in a number of directions, including collaborative instruments separation in music, signal classification, and speaker recognition, following the here demonstrated capability to collectively select the correct groups/models.

At the theoretical level, a whole family of new problems is opened by this proposed framework, some of which we already addressed in this work. A critical one is the overall capability of selecting the correct groups in the collaborative scenario, with missing information, and thereby of performing correct model selection and source identification and separation. Results in this direction will be reported in the future.

Finally, we have also developed an initial framework for learning the dictionary for collaborative hierarchical sparse coding, meaning the optimization is simultaneously on the dictionary and the code. As it is the case with standard dictionary learning, this is expected to lead to significant performance improvements (see [89] for the particular case of this with a single group active at a time).

5 Audio modeling via GMM-based structured sparsity

5.1 Chapter summary

In this chapter we propose a framework for modeling time-frequency representations of harmonic signals and the evaluate it on single channel audio source separation problems. This new framework is based on a audio signal model decouples between the information of pitch and timbre. The pitch is modeled using parametric harmonic filters, while for the timbre Gaussian modeling is exploited. The encoding of the signal is cast as an inverse problem, solved via an efficient maximum-a-posteriori approach. This basic framework is extended to include large timbre variability and pre-learned temporal dynamics. The proposed probabilistic setting leads to an implicit classification of harmonic and non-harmonic sounds by simply selecting the representation that maximizes the corresponding posterior probability. Learned temporal dynamics are also included via a first order hidden Markov model, allowing for the representation of richer signals such as speech. We evaluate the proposed framework with real and synthetic examples of source separation of musical instruments and non-stationary noise removal in speech, achieving better results than with benchmark non-negative spectrogram factorization.

5.2 Introduction

Single-channel source separation is a classical problem in audio processing that arises naturally in various scenarios. The main goal is to separate the different tracks corresponding to the individual sources that are present in the mixture. Although important advances have been obtained throughout the years, this is still considered an open and difficult problem, in part due to its high degree of under-determination. It is then crucial to use all the known physics and available information to constraint the problem in a meaningful way.

The decomposition of time-frequency representations, such as the power or magnitude spectrogram, in terms of elementary atoms of a dictionary has become a popular tool in audio processing. In particular, models involving non-negative matrix factorization (NMF), [58], lead to very good results in a variety of applications, including single-channel source separation [5, 117]. In this approach, the separation is carried out by decomposing the magnitude spectrogram of the mixture signal and then reconstructing groups corresponding to each single source, as detailed in Section 7.3. In the fully unsupervised setting these methods brake down when the sources have a significant time-overlap in the track. A considerable amount of work has been dedicated to adding constraints into the factorization in order to include prior information guiding and stabilizing the decomposition.

In this work we are interested in the modeling of quasi-harmonic audio signals, in which most of the produced sounds can be classified as *pitched* and the (fewer) remaining as *un-pitched*. The variability in the spectrum of an harmonic sound has two main components: the variation of the fundamental frequency and the changes in its spectral envelope (SE). Standard NMF presents some intrinsic difficulties for accurately representing these sounds when they are non-stationary, since it relies in a low dimensional frame-to-frame redundancy. Slight changes in the fundamental frequency with constant spectral envelope produce significant changes in the spectrogram, the same happening when changes in the spectral envelope occur with a fixed pitch. Classical NMF needs several dictionary atoms to

account for this variability, while the nature of these changes is rather simple. In addition, it is desirable for the obtained atoms to have a physically insightful harmonic structure, and this cannot be guaranteed with standard NMF. In order to overcome these difficulties, recent methods have proposed constraining the atoms to mimic the harmonic structure [6, 24, 41, 106, 115].

In this work we propose a simple model for representing the magnitude spectrogram of audio harmonic signals that decouples between the information of pitch and spectral envelope. This allows to efficiently represent a great deal of variability using very simple models for each component. Specifically, let $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_N] \in \mathbb{R}^{F \times N}$ be the power spectrogram of a signal containing, for now, an isolated instrument. Here F and N represent the total number of frequency and time bins, respectively. We can decompose $\mathbf{V}$ as

$$\mathbf{V} \approx \mathbf{H} \bullet \mathbf{E}, \tag{5.2.1}$$

where $\mathbf{E} = [\mathbf{e}_1, \ldots, \mathbf{e}_N] \in \mathbb{R}^{F \times N}$ is a non-negative matrix modeling the spectral envelope and its evolution in time, and $\mathbf{H} = [\mathbf{h}_1, \ldots, \mathbf{h}_N] \in \mathbb{R}^{F \times N}$ is a matrix with entries in the $[0, 1]$ interval enforcing the harmonic structure through an element-wise multiplication $\bullet$. With this physically meaningful representation, changes in pitch are captured in $\mathbf{H}$, while changes in the timbre appear in $\mathbf{E}$. The space of the SE is simple and can be accurately represented via Gaussian modeling as we will later demonstrate.

Given $\mathbf{H}$, finding $\mathbf{E}$ can be cast as an inverse problem with Gaussian Mixture Modeling (GMM). We use a formulation inspired by the excellent results reported by [128] for a number of inverse problems in image processing. In this formulation, the Gaussian parameters involved in the model and the signal representation are simultaneously estimated via a computationally efficient maximum a posteriori expectation-maximization (MAP-EM) algorithm.

Several works have explored the idea of separately modeling the timbre variability and the harmonic structure.[1] In [41] the authors proposed a method based on NMF that also uses parametric templates to represent the harmonic structure of notes, while timbre is represented by estimating amplitude of partials. This work is restricted to exploit an available score and the different notes produced by a source are assumed to share the same fixed relative harmonic amplitudes. In a series of works [24, 25], Durrieu and collaborators proposed a source/filter approach incorporated in the NMF framework. The authors first used their model, referred to as Instantaneous Mixture Model (IMM), for extracting the main melody of musical pieces, [25], and then extended these ideas for producing mid-level representations that are well suited for pitch estimation and audio source separation [24]. Our work is related to IMM since the model of the signals is learned from training data. In the context of multi-pitch estimation, in [115] the authors propose a model that combines NMF with harmonic structure and smoothness constrains in the timbre.

The successful results obtained by the prior art show that splitting the modeling allows for more stable and meaningful representations of harmonic signals. The fundamental differences between our approach and these previous works lies in the particular type of modeling that we consider for the spectral components. The GMM modeling allows a more flexible representation, capturing the frequency-dependent resonance of the instruments and their variability. The developed probabilistic approach intrinsically captures the dependencies between the atoms modeling the timbre, while allowing to easily and naturally include available prior information such as musical scores or temporal constraints. We further show that these models can be extended to handle a broader class of signals, such as human speech.

The chapter is organized as follows. In Section 5.3 we present the basic framework for harmonic signal modeling and describe the MAP-EM estimation algorithm. In Section 5.5

[1] Other particular relations with the literature are explicitly highlighted in Section 5.3.

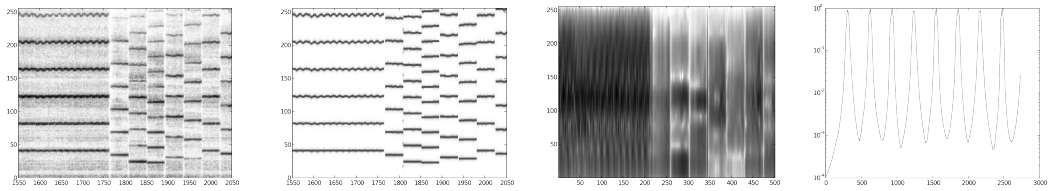


Figure 5.1: From left to right: Example of a spectrogram $\mathbf{V}$, the corresponding matrices $\mathbf{H}$ and $\mathbf{E}$, and a comb filter $\mathbf{h}_i$.

we present extensions of the model accounting for large timbre variabilities, non-harmonic sounds, and pre-learned temporal dynamics. We further show how the extended model can be used for speech. We evaluate the method with synthetic and real data in Section 5.6. Finally, conclusions are drawn in Section 5.7.

5.3 Modeling framework

5.3.1 Source separation via NMF

In NMF [58], the spectrogram of a signal is decomposed as the product of two non-negative matrices,

$$\mathbf{V} \approx \mathbf{W}\mathbf{H}, \quad \text{with} \quad \mathbf{W} \geq 0, \mathbf{H} \geq 0, \tag{5.3.2}$$

where $\mathbf{W} \in \mathbb{R}^{F \times Q}$, $\mathbf{H} \in \mathbb{R}^{Q \times N}$, and $Q \ll F, N$. The matrix $\mathbf{W}$ is a dictionary, and each column represents an atom. The matrix $\mathbf{H}$ codes the activation of each atom in the dictionary throughout the N frames. This representation provides a low-rank linear approximation of $\mathbf{V}$. The non-negativity constraints make the description physically meaningful and has been shown to lead to very good results in a variety of applications.

When NMF is used in the context of audio source separation, $\mathbf{V}$ is the magnitude or power spectrogram of a mixture signal and the dictionary $\mathbf{W}$ is divided into several sub-dictionaries, each corresponding to one of the present sources, $\mathbf{W} = [\mathbf{W}_1, \ldots, \mathbf{W}_c]$, where c

is the total number of sources. Then the decomposition (5.3.2) can be written as,

$$\mathbf{V} \approx [\mathbf{W}_1, \ldots, \mathbf{W}_c] \begin{bmatrix} \mathbf{H}_1 \\ \vdots \\ \mathbf{H}_c \end{bmatrix} = \sum_{r=1}^{c} \mathbf{W}_r \mathbf{H}_r, \tag{5.3.3}$$

where, as in (5.2.1), F and N represent the total number of frequency and time bins, respectively.

Assuming that $\mathbf{V}$ is the magnitude spectrogram, the individual magnitude spectrogram of each source can be reconstructed as $\hat{\mathbf{V}}_j = \mathbf{W}_j \mathbf{H}_j$, with $1 \leq i \leq c$. Each source is recovered by further filtering the mixture signal using the Wiener masks given by,

$$\mathbf{M}_j = \frac{\hat{\mathbf{V}}_j^2}{\sum_{r=1}^{c} \hat{\mathbf{V}}_r^2.}. \tag{5.3.4}$$

Despite the good results obtained by this model in several source separation problems, it presents some limitations for representing harmonic signals. This is important since harmonic signals appear naturally in many fundamental applications, for example, when modeling musical or speech signals, many frames will contain harmonic sounds corresponding to tones in musical signals or voiced sounds in speech. Using NMF for source separation implies assuming that the magnitude spectrogram of each source can be accurately modeled with a low-rank model which is not really the case for this type of signals. Small variations in the pitch significantly increase the rank of $\mathbf{V}$, forcing to increase the number of Q atoms composing the dictionaries. In under-determinate source separation this is highly undesirable, the models need to be as stable as possible in order to make sure that we can properly identify and reconstruct the multiple sources. See Figure 5.2 for a graphical illustration of this phenomenon. A possible solution to address this problem is to add a sparsity constraint, so that only a few atoms of $\mathbf{W}$ are used for representing each audio frame [47].

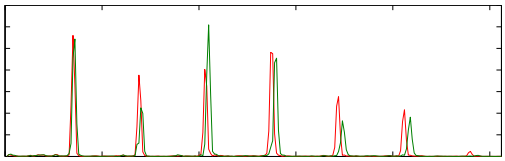


Figure 5.2: This figure exemplifies the sources of variations of harmonic signal. The graph shows the magnitude spectrogram of two frames corresponding of a violin recording. The violin is playing a given note with vibrato, hence the fundamental frequency oscillates around the desired pitch. These two frames present two different sources of variation: the fundamental frequency and the relative amplitude of their partials. These type of variations are difficult to represent accurately with a low rank model.

5.3.2 Single signal model

We present our proposed framework in a constructive way, by increasing it's generality throughout the different sections. In the most general setting, our model is designed to work with signals that can produce both pitched and un-pitched sounds, assuming the former to be the vast majority. In this section we first describe the complete framework for modeling (and separating) harmonic sources (that is, producing exclusively pitched sounds). This basic model captures the essence of our framework and it's extension to the most general case is rather simple, as described in detail in Section 5.5.

Lets assume that in the i-th frame we have an harmonic sound with a fundamental frequency f_i. We further assume that each source is stationary within a time frame. Then, in the Fourier domain, most of the energy of $\mathbf{v_i}$ is concentrated in bins corresponding to frequencies of the form kf_i, with $k \in \mathbb{Z}$. The proposed model (5.2.1), explicitly accounts for this by decomposing each frame in the spectrogram as

$$\mathbf{v}_i = \mathbf{h}_i \bullet \mathbf{e}_i + \mathbf{w}_i, \quad \text{for} \quad i = 1, \ldots, N,$$

where $\mathbf{h}_i$ is the power spectrum of a linear filter that enforces an harmonic constraint in the representation, $\mathbf{e}_i$ is the envelope of the spectral content, and $\mathbf{w}_i$ is a representation

error. We consider $\mathbf{h}_i$ to be the spectral response of a comb filter with unit amplitude and parametrized by its fundamental frequency, $\mathbf{h}_i = \mathbf{h}(f_i)$, as shown in Figure 5.1. The point-wise multiplication $\mathbf{h}_i \bullet \mathbf{e}_i$ corresponds to the filtering of $\mathbf{e}_i$. In every frame, $\mathbf{e}_i$ is assumed to be drawn from a learned Gaussian distribution. The representation error is assumed to be zero mean Gaussian with known or estimated signal-independent isotropic covariance $\sigma^2 I_d$.

In contrast to NMF, our proposed model is non-linear. We use a linear model only to represent the timbre of each instrument, while using the set of parametric filters to model the harmonic constraint, as shown in (5.4.11). This proposed simple model is less general than NMF since it is restricted to quasi-harmonic signals. As mentioned in Section 4.3, this is common to the works that split the modeling into harmonic and spectral envelope [41, 24, 25, 115].

As it will be later shown in Section 5.6, this framework produces good results for modeling multiple musical instruments. However, signals with large timbre variability (i.e., plucked string sounds, singing voice, or speech), in general require richer models than a single Gaussian distribution. To account for this, we propose a natural extension that uses a mixture of Gaussians instead. For the sake of simplicity in the explanation, we separately address this in Section 5.5.1.

5.3.3 Mixed signal model

Let us now assume that the observed signal is a mixture of c different harmonic sources. Generalizing (5.2.1), we want to decompose its power spectrum as

$$\mathbf{V} = \sum_{j=1}^{c} \mathbf{H}_j \bullet \mathbf{E}_j + \mathbf{W}. \tag{5.3.5}$$

The harmonic structure and spectral envelope for the j-th instrument are modeled by the matrices $\mathbf{H}_j = [\mathbf{h}_{j1}, \ldots, \mathbf{h}_{jN}]$ and $\mathbf{E}_j = [\mathbf{e}_{j1}, \ldots, \mathbf{e}_{jN}]$ respectively, following the single signal model presented in 5.3.2. The fundamental frequencies of each source for the different time frames are represented by the vectors $\mathbf{f}_j \in \mathbb{R}^N$ with $1 \leq j \leq c$, this is, the fundamental frequency for j-th source at the i-th frame will be denoted as $[\mathbf{f}_j]_i = f_{ji}$.

As with NMF, the model computation and the signal coding are performed simultaneously, and require estimating:

- The Gaussian parameters $\mathcal{G} = \left\{(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)\right\}_{1 \leq j \leq c}$ for each instrument.
- The set of envelopes, $\left\{\mathbf{E}_j\right\}_{1 \leq j \leq c}$, and the real fundamental frequencies, $\left\{\mathbf{f}_j\right\}_{1 \leq j \leq c}$, from the spectrum $\mathbf{V}$ and the estimated Gaussian distributions for each instrument, $\mathcal{G}$.

To solve this non-convex problem we use an adaptation of the efficient MAP-EM-type algorithm presented for image processing in [128], as explained in detail in Section 5.4.

We distinguish three different training regimes according to the level of supervision used for obtaining the distributions modeling the timbre of the sources: *supervised*, where $\mathcal{G}$ is pre-learned from available unmixed training data; *semi-supervised*, where a rough estimation of $\mathcal{G}$ is available beforehand (for example learned from an isolated recording of a given instrument roughly similar to the one present in the mixture); and *unsupervised*, where there is no model for the sources. In the supervised case, $\mathcal{G}$ is obtained by estimating the model from training data and later used (but not refined) in the testing, where only the specific envelopes for each frame and their corresponding fundamental frequencies are estimated. In the semi-supervised case, the distributions are only used as an initial condition and are further refined in the separation process.

One of the advantages of the proposed framework is that its probabilistic setting allows

to naturally incorporate prior information or smoothness constrains. In the following section we first discuss how to incorporate prior information on the fundamental frequencies, while in Section 5.5.3 we will describe how to include pre-learned temporal dynamics.

5.3.4 Including pitch priors

There are many situations in which prior information on the pitch of the sources is available and can therefore be used to guide the separation problem. Our probabilistic setting allows to include this prior knowledge as a probabilistic distribution over the possible fundamental frequencies.

Lets first consider the so called scored-informed source separation [41, 115, 98], in which the score of the piece is used as side information. The available score provides, for each source, a set of possible values for the fundamental frequency (one for each given instant), $\mathbf{f}_{0j} = [f_{0j1}, \ldots, f_{0jN}]$ with $1 \leq j \leq c$. In general $\mathbf{f}_{0j}$ is a very good approximation of the true values of the fundamental frequencies, $\mathbf{f}_j = [f_{j1}, \ldots, f_{jN}]$, but they cannot be assumed to be identical. A canonical example of such a situation is the vibrato, where the real fundamental frequency slightly oscillates around the note specified in the score. Thus, we model the fundamental frequency as a random variable with a time changing (real valued) distribution. We denote it's probability density function as $\alpha_i(f)$, with $i = 1, \ldots, N$. A simple choice (used in all the score informed experiments shown in Section 5.6.1) is to select this density as a Gaussian distribution centered at the fundamental frequency given by the score and with variance σ_0^2.[2]

Assuming to have a hint of the fundamental frequency of each source on each time frame is only valid for certain music separation problems. In a more general case, however one can often have some basic prior knowledge about the possible fundamental frequencies

[2]In all our experiments we considered σ_0 to be 1% of the value of f_{0i}. However, σ_0^2 might be instrument dependent, since instruments such as the cello are normally played with vibrato, while others have a more constant pitch.

that the sources might use. For example, when modeling speech signals, different speakers might have different registers, this is, different ranges of fundamental frequency variability. This means that prior distributions for the pitch of the different sources can be learned and used in the separation process. Unlike the score informed case, here the distribution of the fundamental frequency is not time dependent, $\alpha_i(f) = \alpha(f)$ for all $i = 1, \ldots, N$. Finally, if there is no prior information available, $\alpha(f)$ can be assumed uniform (with the appropriate interval).

5.4 Computational algorithm

In this section we present the computational algorithm used to perform the source separation problem following the model introduced in Section 5.3.3. We propose to use a MAP-EM-type algorithm for simultaneously estimating both, the timber distributions and the signal representations (this is, the fundamental frequencies and the spectral envelopes).

The MAP-EM algorithm is an iterative procedure that alternates between two steps. An *E-step* that estimates the spectral content of the sources assuming that the Gaussian parameters are known, and an *M-step* that reciprocally estimates the Gaussian parameters for each source while assuming that the spectral envelopes and fundamental frequencies are known. We now describe each one in detail.

5.4.1 E-Step: signal estimation

The *E-step* can be performed independently for each frame. We obtain the estimates by solving the MAP,

$$\{\tilde{\mathbf{e}}_{ji}, \tilde{f}_{ji}\}_{1\leq j\leq c} = \underset{\{\mathbf{e}_{ji}, f_{ji}\}_{1\leq j\leq c}}{\operatorname{argmax}} \log p(\mathbf{e}_{c1}, \ldots, \mathbf{e}_{ci}, f_{1i}, \ldots f_{ci} | \mathbf{v}_i, \alpha_{1i}, \ldots, \alpha_{ci}, \mathscr{G}), \tag{5.4.6}$$

where α_{ji} is the prior pitch probability density function for the j-th source in the i-th frame. Using the independence of each source, one can see that maximizing the cost function in (5.4.6) is equivalent to maximizing

$$\log p(\mathbf{v}_i|\mathbf{e}_{c1},\ldots,\mathbf{e}_{ci},f_{1i},\ldots f_{ci}) + \sum_{r=1}^{c}\log p(\mathbf{e}_{ri}|\mathscr{G}) + \sum_{r=1}^{c}\log \alpha_{ri}(f_{ri}). \tag{5.4.7}$$

In (5.4.7), the first term can be obtained from the noise probability density function, the second term from the Gaussian models for the timber of each source, and the last one using the prior model for the fundamental frequencies as explained in Section 5.3.4. Substituting in (5.4.7) these probability density functions and writing the masking filters $\mathbf{h}(f_{ji})$ as a linear operators $\mathbf{h}(f_{ji}) \bullet \mathbf{e}_{ji} = \mathbf{U}_{ji}\mathbf{e}_{ji}$, where $\mathbf{U}_{ji} = \mathrm{diag}(\mathbf{h}(f_{ji}))$, we obtain

$$\underset{\{\mathbf{e}_{ji},f_{ji}\}_{1\le j\le c}}{\operatorname{argmin}} \|\mathbf{v}_i - \sum_{r=1}^{c}\mathbf{U}_{ri}\boldsymbol{\mu}_r - \sum_{r=1}^{c}\mathbf{U}_{ri}\mathbf{e}_{ri}\|^2 + \sigma^2\sum_{r=1}^{c}\mathbf{e}_{ri}^T{\boldsymbol{\Sigma}_{\mathbf{r}}}^{-1}\mathbf{e}_{ri} - \sigma^2\sum_{r=1}^{c}\log\alpha_{ji}(f_{ji}) + \sigma^2\sum_{r=1}^{c}\log(|\boldsymbol{\Sigma}_{\mathbf{r}}|). \tag{5.4.8}$$

This sub-problem is non-convex when minimizing over both the envelopes and the fundamental frequencies, $\mathbf{e}_{ji}, f_{ji}$, with $1 \le j \le c$. We solve it by iteratively fixing one and optimizing over the others. Note that in this case, the last term in (5.4.8) is constant and can be eliminated from the objective function, however this will no longer be the case in Section 5.5.1 when using the GMM for modeling the $\mathbf{e}_{ji}$'s.

When fixing f_{ji}'s in (5.4.8), the problem is strictly convex and can be solved efficiently in closed form and independently from the prior pitch distribution,

$$\tilde{\mathbf{e}}_{ji} = \mathbf{U}_{ji}^T\boldsymbol{\Sigma}_j\left(I_d\sigma^2 + \sum_{r=1}^{c}\mathbf{U}_{ri}^T\boldsymbol{\Sigma}_r\mathbf{U}_{ri}\right)^{-1}(\mathbf{v}_i - \sum_{r=1}^{c}\boldsymbol{\mu}_r), \tag{5.4.9}$$

where I_d is the identity matrix of the appropriate size. In oder to have a physically meaningful decomposition, the obtained $\{\tilde{\mathbf{e}}_{ji}\}_{1\le j\le c}$ in (5.4.9) need to satisfy $\boldsymbol{\mu}_j + \tilde{\mathbf{e}}_{ji} \ge 0$. We observed in practice that this is always the case when the number of sources is small, although

it is not explicitly imposed in our formulation. This has also been observed in various image processing inverse problems [128] (where the pixel intensity is also non-negative). However, this might not happen when the number of sources in the mixture is large due to the high degree of under-determination. We then add to (5.4.8) a non-negativity constrain, $\boldsymbol{\mu}_j + \mathbf{e}_{ji} \geq 0$, $\forall j, i$. This constrained optimization is still convex and can be carried out using projected gradient methods [76]. The idea is to first compute the solution of the unconstrained problem given in (5.4.9) and, if the constrains are not automatically met, adjust it using an inexpensive iterative proximal method algorithm in the dual problem. The details on this technique are given in Section 5.4.4.

Since we are working with a short time analysis window, the set of distinguishable fundamental frequencies is limited in number, an then it can be naturally discretized. Then, solving the problem in (5.4.8), with the $\mathbf{e}_{ji}$'s fixed, reduces to evaluating the cost function in a small number of candidates distributed around the ones provided, e.g., by the score, $\{f_{0ji}\}_{1\leq j\leq c}$, and choosing the one for which the minimum is obtained. In the particular case discussed in Section 5.3.4, in which we have an available score to guide the separation, the las term in (5.4.8) simply becomes,

$$\log \alpha_{ji}(f_{ji}) = -\frac{\sigma^2}{\sigma_0^2}|f_{ri} - f_{0ri}|^2.$$

5.4.2 M-Step: model estimation

After the estimation of the envelopes and the fundamental frequencies, we recalculate the Gaussian parameters for all the instruments. This is done via the empirical mean and covariance,

$$\tilde{\boldsymbol{\mu}}_j = \frac{1}{N}\sum_{i=1}^{N}\mathbf{e}_{ji} \quad , \quad \tilde{\boldsymbol{\Sigma}}_j = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{e}_{ji} - \tilde{\boldsymbol{\mu}}_j)(\mathbf{e}_{ji} - \tilde{\boldsymbol{\mu}}_j)^T, \quad 1 \leq j \leq c. \tag{5.4.10}$$

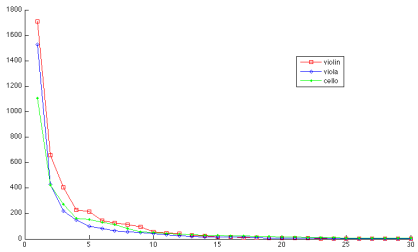


Figure 5.3: This figure show the energy captured by the eigenvalues of three different harmonic instruments: violin, viola and cello. In all three cases most the first 10 principal direction capture most of the energy and 99 % of it is retained only by the first 20 eigenvalues. This shows that the spectral envelope of these harmonic instruments is highly compressible.

Refer to [128] for a discussion on the optimality of this selection.

5.4.3 Structured estimation in PCA bases

Given a set of signals $\{\mathbf{e}_{ji}\}_{1\leq i\leq N}$, a Principal Component Analysis (PCA) basis is defined as the matrix $\mathbf{B}_j = [\mathbf{b}_{j1},\ldots,\mathbf{b}_{jF}]$ that diagonalizes the corresponding (estimated) covariance matrix, $\mathbf{\Sigma}_j = \mathbf{B}_j^T\mathbf{S}_j\mathbf{B}_j$, where $\mathbf{S}_j = \text{diag}(\boldsymbol{\lambda}_j)$ is a diagonal matrix, whose diagonal elements are given by the vector $\boldsymbol{\lambda}_j = [\lambda_{j1},\lambda_{j2},\ldots,\lambda_{jF}] \in \mathbb{R}^F$ of the corresponding sorted eigenvalues $\lambda_{j1} \geq \lambda_{j2} \geq \ldots \geq \lambda_{jF} \geq 0$. The columns of $\mathbf{B}_j$ are orthonormal and represent the principal directions of variation of $\{\mathbf{e}_{ji}\}_{1\leq i\leq N}$. The magnitude of the eigenvalues measure the energy of the variation in the corresponding directions.

Working in the PCA basis rather than the canonical one, $\mathbf{a}_{ji} = \mathbf{B}_j^T\mathbf{e}_{ji}$, allows to reduce the dimensionality of the data in a meaningful way. For example, when representing the timbre of various harmonic instruments, we can verify that they are highly compressible: the first few eigenvalues of the covariance matrices concentrate most of the total energy, see Figure 5.3.4. We can then write our model stated in (5.3.5) as

$$\mathbf{V} = \sum_{j=1}^{c}\mathbf{H}_j \bullet \mathbf{B}_j\mathbf{A}_j + \mathbf{W} \approx \sum_{j=1}^{c}\mathbf{H}_j \bullet \hat{\mathbf{B}}_j\mathbf{A}_j + \mathbf{W}, \tag{5.4.11}$$

where the matrices $\hat{\mathbf{B}}_j = [\mathbf{b}_{j1}, \ldots, \mathbf{b}_{jk}]$ conserve only the first $k \ll F$ principal directions. The MAP estimate (5.4.7) can be equivalently computed as (see also [128])

$$\{\tilde{\mathbf{a}}_{ji}, \tilde{f}_{ji}\}_{1\leq j\leq c} = \underset{\mathbf{a}_{ji}, f_{ji}}{\operatorname{argmin}} \quad \|\mathbf{v}_i - \textstyle\sum_{r=1}^{c} \mathbf{U}_{ri}\boldsymbol{\mu}_r - \sum_{r=1}^{c} \mathbf{U}_{ri}\hat{\mathbf{B}}_j\mathbf{a}_{ri}\|^2 + \sigma^2 \sum_{r=1}^{c}\sum_{p=1}^{k} \frac{|a_{ri}[p]|^2}{\lambda_r^p}$$
$$-\sigma^2 \textstyle\sum_{r=1}^{c} \log \alpha_{ji}(f_{ji}) + \sigma^2 \sum_{r=1}^{c} \sum_{p=1}^{k} \log(\lambda_r^p). \quad (5.4.12)$$

This formulation allows to clearly observe another interesting property of the proposed model. Inside each PCA basis the atoms are pre-ordered by their corresponding eigenvalues, and weighting term in (5.4.12) privileges the coefficients corresponding to the principal directions with larger energy. This representation stabilizes the decomposition, which is critical in these type of ill-posed source separation problems.[3]

5.4.4 Optimization

The fundamental computation involved in the signal estimation step described in Section 5.4.1 is the MAP estimation given in (5.4.8), or its lower dimensional version (5.4.12). In both cases, the closed form solution is no longer valid when adding a non-negativity constraints $\boldsymbol{\mu}_j + \mathbf{e}_{ji} \geq 0$ (or $\boldsymbol{\mu}_j + \hat{\mathbf{B}}_j\mathbf{a}_{ji} \geq 0$). This restriction is important to guarantee a physically meaningful solution, that is, the reconstructed power spectrum needs to be non-negative. In this section we describe a proximal method approach applied to the dual problem, that first computes the unconstrained solution and then refines this solution only if the restrictions are not automatically met. For simplicity in the notation we will derive the procedure for (5.4.12) while the extension to (5.4.8) is straight forward. We will start re-writing the unconstrained problem (5.4.12) as

$$\min_{\mathbf{a}_1,\ldots,\mathbf{a}_c} \|\mathbf{v} - \sum_{r=1}^{c} \mathbf{c}_r - \sum_{r=1}^{c} \mathbf{D}_r\mathbf{a}_r\|^2 + \sigma^2 \sum_{r=1}^{c} \mathbf{a}_r^T\mathbf{S}_r^{-1}\mathbf{a}_r, \quad (5.4.13)$$

[3]This is not the standard weighted norms, since the weights are collaborative, given by the eigenvalues.

where $\mathbf{c}_r = \mathbf{U}_{ri}\boldsymbol{\mu}_r \in \mathbb{R}^F$, $\mathbf{D}_r = \mathbf{U}_{ri}\hat{\mathbf{B}}_j \in \mathbb{R}^{F\times k}$ and $\mathbf{S}_r = \text{diag}(\hat{\boldsymbol{\lambda}}_j) \in \mathbb{R}^{k\times k}$ (the vectors $\hat{\boldsymbol{\lambda}}_r \in \mathbb{R}^k$ have the k largest (non-negative) eigenvectors of $\boldsymbol{\Sigma}_r$).[4] We dropped the last constant term in (5.4.12), since it is not relevant for the optimization method, but needs to be taken into account when comparing the MAP values for class/Gaussian selection. Problem (5.4.13) is a quadratic program that can be rewritten in a compact way as,

$$\min_{\mathbf{a}} \|\mathbf{y} - \mathbf{D}\mathbf{a}\|^2 + \mathbf{a}^T\mathbf{S}^{-1}\mathbf{a}, \tag{5.4.14}$$

where $\mathbf{y} = \mathbf{v} - \sum_{r=1}^{c}\mathbf{c}_r$, $\mathbf{D} = [\mathbf{D}_1, \ldots, \mathbf{D}_c] \in \mathbb{R}^{F\times kc}$, $\mathbf{a} = [\mathbf{a}_1^T, \ldots, \mathbf{a}_c^T]^T \in \mathbb{R}^{kc}$ and $\mathbf{S} = \text{diag}([\hat{\boldsymbol{\lambda}}_1^T, \ldots, \hat{\boldsymbol{\lambda}}_c^T]^T) \in \mathbb{R}^{kc\times kc}$. The optimum can be found in closed form via Wiener filtering as in (5.4.9),

$$\tilde{\mathbf{a}} = \mathbf{W}^{-1}\mathbf{D}^T\mathbf{y}, \tag{5.4.15}$$

with $\mathbf{W} = \sigma^2\mathbf{S}^{-1} + \mathbf{D}^T\mathbf{D}$. The matrix $\mathbf{W}$ is positive semidefinite and consequently it's inverse is well defined. This solution requires the inversion of a small (and well conditioned) matrix of dimensions $kc \times kc$.

Now, as discussed in Section 5.4.1, the obtained coefficients $\tilde{\mathbf{a}} = [\tilde{\mathbf{a}}_1^T, \ldots, \tilde{\mathbf{a}}_c^T]^T$ in (5.4.15) need to satisfy, $\mathbf{D}_r\tilde{\mathbf{a}}_r + \mathbf{c}_r \geq 0$ for $r = 1, \ldots, c$. However, this might not happen automatically in all cases when dealing with a large number of sources. In order to overcome this problem we propose to solve a constrained (5.4.14) given by

$$\min_{\mathbf{a}} \|\mathbf{y} - \mathbf{D}\mathbf{a}\|^2 + \mathbf{a}^T\mathbf{S}^{-1}\mathbf{a}, \quad \text{s.t.} \quad \mathbf{D}_r\mathbf{a}_r + \mathbf{c}_r \geq 0 \quad \forall r. \tag{5.4.16}$$

This new problem is still strictly convex and can be solved via quadratic programing. However the computational cost would be very demanding. In the following we propose a tailored solution based on proximal methods which is considerably more efficient.

[4] For ease of notation, will assume that all matrices $\mathbf{D}_r$ have the same number of columns, k, for $r = 1, \ldots, c$. However the whole analysis and algorithm hold for the general case.

The Lagrangian of problem (5.4.16) is given by,

$$L(\mathbf{a}, \mathbf{z}) = \mathbf{a}^T \mathbf{W}\mathbf{a} - q(\mathbf{z})\mathbf{a} - \mathbf{z}^T \mathbf{c},$$

where $\mathbf{z} \in \mathbb{R}^{cF}$ are the Lagrange multipliers, $\mathbf{c} = [\mathbf{c}_1^T, \ldots, \mathbf{c}_c^T]^T \in \mathbb{R}^{cF}$, $q(\mathbf{z}) = 2\mathbf{y}^T \mathbf{D} + \mathbf{z}^T \mathbf{L}$, with $\mathbf{L} \in \mathbb{R}^{cF \times ck}$ the block diagonal matrix given by

$$\mathbf{L} = \begin{pmatrix} \mathbf{D}_1 & 0 & \ldots & 0 \\ 0 & \mathbf{D}_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \mathbf{D}_c \end{pmatrix}.$$

Then the dual problem of (5.4.16) can be expressed as,

$$\max_{\mathbf{z}} -\frac{1}{4} q(\mathbf{z})^T \mathbf{W}^{-1} q(\mathbf{z}) - \mathbf{z}^T \mathbf{c}, \quad \text{s.t.} \quad \mathbf{z} \geq 0. \tag{5.4.17}$$

Problem (5.4.16) satisfies Slater's condition and consequently strong duality holds (the duality gap between the optima of the primal and dual problems is zero). If the optima of problems (5.4.16) and (5.4.17) are found in $\mathbf{a}^*$ and $\mathbf{z}^*$ respectively, then the following identity holds,

$$\mathbf{a}^* = \mathbf{W}^{-1}\mathbf{D}^T \mathbf{y} + \frac{1}{2}\mathbf{W}^{-1}\mathbf{L}^T \mathbf{z}^*. \tag{5.4.18}$$

Note that $\mathbf{a}^*$ is composed by two terms, one that is equal to $\tilde{\mathbf{a}}$ in (5.4.15) (this is, the point in which the minimum of the unconstrained problem (5.4.14) is attained), and another one depending on the solution of its dual problem (5.4.17). From the optimality conditions of (5.4.16) and (5.4.17) one can see that $\mathbf{z}^* = 0$ if and only if the non-negativity constraints in (5.4.16) are automatically satisfied by $\tilde{\mathbf{a}}$. Thus, the second term in (5.4.18) can be thought

as a "correction" term that needs to be added when the solution of (5.4.16) in not physically meaningful. Based on this observations, we first compute the solution to the unconstrained optimization problem, then if the reconstructed spectrograms do not meet the non-negative constraint we proceed to compute the second term, $1/2\mathbf{W}^{-1}\mathbf{L}^T\mathbf{z}^*$, this is, to solve problem (5.4.17).

The dual problem of a constrained quadratic problem is also a constrained quadratic problem [8]. However, in contrast to what happened in (5.4.14), the projection onto the feasible set of (5.4.17) (the non-negative orthant) can be very efficiently computed. This important difference makes (5.4.17) particularly well suited for applying an accelerated projected gradient method [8, 76, 62].

Projected gradient methods solve constrained convex problems by iterating a gradient descent on the objective function and a projection into the feasible set. They were first used in the context of NMF in [62]. In this case, the gradient of the cost function in (5.4.17) can be computed in closed form as,

$$\mathbf{LWL}^T\mathbf{z}^{(n)} + \mathbf{LWDv} - \mathbf{c}.$$

While the projection into the feasible set reduces to $\max\{\mathbf{z}, 0\}$, the component-wise maximum function between zero and $\mathbf{z}$. Since the the cost function in [62] is Lipschitz, a constant descent step can be employed. The procedure is summarized in Algorithm 2, and represent a standard implementation of these methods. We refer the reader to [76] for a detailed discussion.

Algorithm 2: Optimization Algorithm

Input: Data
Output: The optimal point $\mathbf{a}$
set $\tilde{\mathbf{a}} = \mathbf{W}^{-1}\mathbf{D}^T\mathbf{y}$,;
if $\mathbf{D}_r\tilde{\mathbf{a}}_r + \mathbf{c}_r \geq 0$ *for* $r = 1, \ldots, c$ **then**
  set $\mathbf{a}^* = \tilde{\mathbf{a}}$;
else
  Initialize $v_1 := \frac{1}{2}, \boldsymbol{\gamma}^{(0)} := 0$;
  while *Stopping criterion* **do**
    Gradient step:
    $\mathbf{G}^{(n)} = \mathbf{z}^{(\mathbf{n})} - \frac{1}{d}(\mathbf{LWL}^T\mathbf{z}^{(n)} + \mathbf{LWDv} - \mathbf{c})$;
    Projection step:
    $\boldsymbol{\gamma}^{(n)} = \max\left\{\mathbf{G}^{(n)}, 0\right\}$;
    Point estimation:
    $\mathbf{z}^{(n)} = \boldsymbol{\gamma}^{(n)} + \frac{t_{n-1}-1}{t_n}(\boldsymbol{\gamma}^{(n)} - \boldsymbol{\gamma}^{(n-1)})$;
    $t_{n+1} = \frac{1}{2} + \frac{\sqrt{1+4t_n^2}}{2}$;
    $n = n + 1$;
  end
  set $\mathbf{a}^* = \tilde{\mathbf{a}} + \frac{1}{2}\mathbf{W}^{-1}\mathbf{L}^T\mathbf{z}^{(n)}$;
end

5.5 Speech signals

5.5.1 Large timbre variability

When signals present large timbre variability, the space of their spectral envelopes is no longer highly compressible and, consequently, modeling it with a single Gaussian would produce a distribution with several principal directions with a large variance. This effect reduces the good generalization properties discussed in Section 5.4.3. In order to address this, we propose to partition the timber space and separately represent each sub-part using a single (tighter) Gaussian model, following the idea used in the image processing domain in [128]. This Gaussian Mixture Model (GMM) can be thought as the union of several Gaussian models. This idea is well suited for capturing the non-stationarity of the audio signals; each sub-model would correspond to one out of several possible states of the signal,

for example different phonemes in speech. Then during a recording, the signal dynamically changes from one state to another.

Mathematically, the mixture model can be written exactly as in (5.3.5), this is, each frame is represented as,

$$\mathbf{v}_i = \sum_{j=1}^{c} \mathbf{h}_{ji} \bullet \mathbf{e}_{ji} + \mathbf{w}_i, \quad \text{for} \quad i = 1, \ldots, N, \tag{5.5.19}$$

but now, each spectral envelope $\mathbf{e}_{ji}$ is drawn from one of G possible Gaussians. We denote the GMM corresponding to the j-th source as $\mathscr{G}_j = \left\{(\boldsymbol{\mu}_{jg}, \boldsymbol{\Sigma}_{jg})\right\}_{1 \leq g \leq G}$ with $j = 1 \ldots c$.[5] Here $\mathbf{h}_{ji}$ still represents the harmonic filters introduced in Section 5.3 (parametrized with the fundamental frequency f_{ji}), and $\mathbf{w}_i$ is again the representation error.

With this new model, the signal estimation step described in Section 5.4.1 becomes more difficult. In the new E-step of the MAP-EM algorithm, the set of Gaussian distributions $\mathscr{G} = \left\{\mathscr{G}_j\right\}_{1 \leq j \leq c}$ are assumed fixed and, for each source, we now need to determine three things:

- The fundamental frequencies for each frame, f_{ji} for $i = 1, \ldots, N$.
- The index, g_{ji}, for $i = 1, \ldots, N$, of the Gaussian distribution in $\mathscr{G}_j$ that generates the spectral envelope of the frame.
- The specific spectral envelope $\mathbf{e}_{ji}$ that best represents the timbre for $i = 1, \ldots, N$.

As in our previous model, the estimation is again carried out via MAP. We solve the MAP given in (5.4.8) for every combination of Gaussian distributions. For each of such combination the problem can be written exactly as the one described in Section 5.3. This is, determining the fundamental frequencies and envelopes given a specific distribution for each source. We then select the combination of Gaussian distributions that produces the

[5] For the sake of simplicity in the notation, we assume the same number of Gaussians to model the spectral envelope of is the same for all the signals in the mixture. The extension to a general case is straightforward.

maximum score. Note that now, when comparing the MAP obtained using different Gaussian distributions, the last term in (5.4.8), $\sigma^2 \sum_{r=1}^{c} \log(|\mathbf{\Sigma_r}|)$, is different for each case and needs to be included for a meaningful comparison of the optimum values. This term is larger for tighter models (with faster decreasing singular vectors), and intuitively prevents the in-discriminative use of the larger ones. Naturally, here we can again use the dimensionality reduction technique described in Section 5.4.3.

This more general model is not yet well suited for working in the completely unsupervised case, since there is no clear way of connecting the different states of a signal. The new model is better suited for the supervised and semi-supervised regimes discussed in Section 5.3.3, since they provide an initial estimation of the timbre models.

The idea of modeling time-frequency frames using one out of many low-rank linear models rather than using a single (large) global one has been previously studied in [75]. The difference with our work is that here we only model the timbre, while in [75] the model is used to represent the entire magnitude spectrum; and, again, that we use Gaussian modeling rather than NMF.

5.5.2 Non-harmonic sounds

Until now, the signals were assumed to be completely harmonic on every frame, that is, all the sounds produced by the sources are always pitched. We now generalize this idea to include un-pitched sounds, assuming that pitched sounds are the vast majority.

Un-pitched sounds are modeled simply by considering $\mathbf{h}_i$ constant with value one, the whole spectrum of the signal is directly modeled by $\mathbf{e}_{ji}$, as commonly done in NMF. We again use a Gaussian distribution to model the spectral component. For each source we add a Gaussian distribution to its GMM, $\mathscr{G}_j$, that is exclusively used for representing the non-harmonic sounds. This includes the particular case in which $\mathscr{G}_j$ has two Gaussian distributions, one for pitched sounds and the other for un-pitched sounds. In this way, the

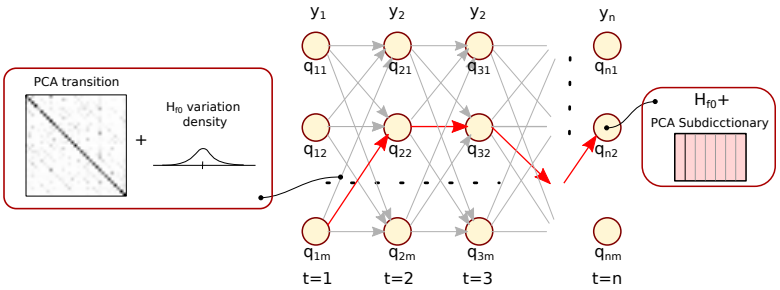


Figure 5.4: In this figure we show the basic scheme for the signal estimation including temporal dynamics. The path producing the MAP is estimated via Viterbi's algorithm considering the learned transition probabilities between the different signal states. In this case, the temporal dynamics are modeled via a learned Gaussain transition matrix and smoothness in the fundamental frequency.

decision if it is a pitched or an un-pitched sound is implicitly done selecting the distribution of $\mathcal{G}_j$ that produces the highest MAP score.

5.5.3 Temporal dynamics

Most of the factorization approaches for modeling audio signals do not take into account a very important aspect of sound: its temporal structure. In this section we take the proposed model one step further to include pre-learned temporal dynamics. In [75] the authors do include pre-learned temporal dependencies in the probabilistic NMF context. The idea is to split the dictionary into several sub-dictionaries and learn the temporal dynamics via a Hidden Markov Model (HMM) [87]. In this work, we integrate a similar idea into our framework considering a HMM that models the usage of the different Gaussian distributions modeling the timber (as introduced in Section 5.5.1), and the fundamental frequencies (as introduced in Section 5.3.4).

We consider that at each time frame the audio signal is in one out of several possible states, defined as the different possible pairs of Gaussian distribution and fundamental frequency. A particular realization of this signal will then have associated a corresponding trajectory in the state space. Clearly, not all the possible trajectories have the same

probability. For example, it is reasonable to expect that small changes in the fundamental frequencies between neighboring time frames are more likely than large ones. On the other hand, the different phonemes of a given language (and the probability of transition between phonemes), imply certain temporal constrains in the usage of the different Gaussians throughout time, that also translate into specific preferred trajectories in this state space. Hence, we propose to learn this transition probabilities from training data: coding clean isolated samples of the signals with the MAP criterion and estimating the transition matrices from the obtained states.

Following [75], we propose to use a first order HMM to learn these temporal dependencies and then simultaneously code the different frames in order to globally maximize the MAP. One can think of this procedure as a collaborative approach in which all frames are used together to find a solution that is globally better than maximizing the MAP independently for each frame.

We can now think of the magnitude spectrum $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_N]$ as a sequence of N input observations. Let $\mathbf{Q} = [\mathbf{q}_1, \ldots, \mathbf{q}_N]$ be the state of the signal for each corresponding time frame where, $\mathbf{q}_i = (g_i, f_i)^T$ is composed by the index of the Gaussian distribution from which the spectral envelope is drawn, g_i, and the fundamental frequency f_i (we include non-pitched sounds as a 0 fundamental frequency). Let M be the total number of possible states (which is equal to the product of the number of Gaussian distributions modeling the timbre and the number of possible discretized fundamental frequencies). The coding of the signal consists in recovering the optimal $\mathbf{Q}$ as well as the corresponding spectral envelopes for $\mathbf{E} = [\mathbf{e}_1, \ldots, \mathbf{e}_N]$. We do this by computing for each frame the MAP obtained for each possible state, and then, given the learned transition probabilities, obtaining the solution that produces the global MAP, see Figure 5.4.

Formally, we want to find $(\mathbf{E}, \mathbf{Q})$ such that the probability $p(\mathbf{V}|\mathbf{E}, \mathbf{Q})$ is maximized, as discussed in Section 5.4.1 for the signal estimation step of the MAP-EM algorithm. Maximizing

$p(\mathbf{V}|\mathbf{E},\mathbf{Q})$ with respect to $(\mathbf{E},\mathbf{Q})$ is equivalent to maximizing,

$$p(\mathbf{V},\mathbf{E},\mathbf{Q}) = p(\mathbf{V}|\mathbf{E},\mathbf{Q})p(\mathbf{E}|\mathbf{Q})p(\mathbf{Q}).$$

Now, since we are assuming a first order HMM, we obtain,

$$p(\mathbf{Q}) = p(\mathbf{q}_1)p(\mathbf{q}_2|\mathbf{q}_1)\dots p(\mathbf{q}_N|\mathbf{q}_{N-1}) = p(\mathbf{q}_1)\prod_{i=2}^{N} p(\mathbf{q}_i|\mathbf{q}_{i-1}).$$

This probabilities only depend on the entries of the transition matrix. On the other hand, each time frame is assumed to be independent from the others, thus we have

$$p(\mathbf{V}|\mathbf{E},\mathbf{Q})p(\mathbf{E}|\mathbf{Q}) = \prod_{i=1}^{N} p(\mathbf{v}_i|\mathbf{e}_i,\mathbf{q}_i)p(\mathbf{e}_i|\mathbf{q}_i),$$

where this last expression is exactly the MAP for each frame individually given a specific signal state as shown in (5.4.6). This problem can be solved with Viterbi's algorithm, see [87] for details. For each time frame i we want to compute Viterbi's functions $\delta(r)$ and $\varphi(r)$, $r = 1,\dots,M$, as

$$\delta_i(r) = \max_{\mathbf{e}_1\dots\mathbf{e}_i,\mathbf{q}_1,\mathbf{q}_{i-1}} p(\mathbf{e}_1\dots\mathbf{e}_i,\mathbf{q}_1,\dots\mathbf{q}_{i-1},\mathbf{v}_1,\dots,\mathbf{v}_i|\mathbf{q}_k = r).$$

This can be done recursively by solving,

$$\delta_{i+1}(r) = \left[\max_{1\le l\le M} \delta_i(l)\right] \max_{\mathbf{e}_i} p(\mathbf{v}_i|\mathbf{e}_i,\mathbf{q}_i = r)p(\mathbf{e}_i|\mathbf{q}_i = r)$$

Then we have,

$$\varphi_{i+1}(j) = arg \max_{1\le r\le M} \delta_i(r)p(\mathbf{q}_{i+1} = j|\mathbf{q}_i = r).$$

Finally, applying the backward procedure, one obtains the optimal state path with the corresponding coefficient vectors [87].

5.6 Experimental results

In this section we show results for different applications involving musical and speech data under supervised and semi-supervised settings. In all the experiments, the separation is calculated by a Wiener filter based on the time-frequency masks computed from our algorithm's output, see Section 7.3.

5.6.1 Music instruments separation experiments

Music database

The music database described in [41] was used to evaluate the score informed source separation application. It consists of 12 different sets of tracks for musical pieces by Bach, Beethoven, and Boccherini, synthesized from real string instrument notes (two violins, viola, and cello). The database includes 30 seconds of the tracks for each composition, with the correspondent aligned MIDI data synthesized by two different methods, from now on, $M1$ and $M2$. These two methods present distinct characteristics such as note decay time or fluctuations from the ideal pitch. Also, method $M2$ includes a reverberation effect, while $M1$ does not.

In addition, a real music example from the Development Set for MIREX 2007 MultiF0 Estimation Tracking Task is evaluated, containing a 52 seconds long musical piece played by 5 different wind instruments. The separated tracks for each instrument are available and the mixture is obtained by simply summing them. In this case the MIDI data is estimated from each original track.

	PLCA train M1	PLCA train M2	PDA	sPCA train M1	sPCA train M2	Oracle
SIR	22.8	14.2	20.2	17.9	17.8	20.5
SAR	11.5	6.3	7.7	11.0	10.9	13.6
SDR	11.1	4.8	7.2	10.8	10.0	12.7

Table 5.1: Results with synthesis method M1 as testing signals.

	PLCA train M1	PLCA train M2	PDA	sPCA train M1	sPCA train M2	Oracle
SIR	12.4	20.1	12.6	16.2	16.5	19.6
SAR	4.5	10.6	3.3	8.5	8.9	12.3
SDR	3.1	10.1	2.1	7.7	8.1	11.5

Table 5.2: Results with synthesis method M2 as testing signals.

Results

The experiments were carried out using a sampling frequency of 11025 Hz, with a frame length of 512 samples and a hop size of 128. The fundamental frequencies ranged from 60 Hz to 1280 Hz to cover all the possible instruments notes present in the database, and were discretized in a logarithmic scale of 1024 values. In this setting, only one Gaussian is selected to model the envelope of each instrument.

The results are presented in terms of the standard signal to interference, signal to artifacts, and signal to distortion ratios (SIR, SAR, SDR, respectively) as defined in [116]. The separation results are shown in tables 5.1 and 5.2, in which our method (sPCA) is compared with the results in [41] and a method based on probabilistic latent component analysis (PLCA) [35]. We also show the results when using the original tracks to build the Wiener masks in 5.3.4 to calculate the separation, considering this the best performance our algorithm can achieve; we refer to this as the Oracle method.

As expected, the Oracle method gives the best results for SAR and SDR. Also as expected, we obtain slightly better results when trainning the system with the same synthesis method as the test method. Compared to PLCA our results are lower but comparable when

training with the same synthesis method, but there is a bigger performance increase when a different synthesis method for the training data is used. This means that the proposed algorithm is less sensitive to the initialization than the PLCA-based one, having better generalization properties. In all cases the results obtained were better than those of PDA in terms of SAR and SDR.

5.6.2 Speech separation experiments

In this section we evaluate the performance of our proposed framework when representing speech signals.

Speech and noise databases

Speech experiments were conducted using part of the GRID database described in [18], which consists of a set of sentences recorded by 34 different speakers. Each sentence has 6 words, each from a different category, and in the following order: <command> <color> <preposition> <letter> <digit> <adverb>. The command can be one of: bin, lay, place, or set; the color can be: blue, green, red, or white; the prepositions are: at, by, in, or with; the letters can be any from A to Z excluding W; the digits from 0 to 9; and the adverb is one of: again, now, please, and soon. See [18] for additional details on the database.

Also, a noise database was selected to mix the clean speech files with real noisy environments at different Signal to Noise Ratios (SNR). The noise files used are part of the Aurora Database described in [81]. The noisy environments included in the experiments are: Babble, Car, Exhibition hall, Restaurant, Street, Airport, Train station, and Train.

Speech denoising with structured noise

We evaluated the performance of the denoising algorithm using two different criteria. In the first one we used, as before, the objective single channel separation measures: SAR, SIR

and SDR [116]. In the second evaluation we measured the intelligibility of the denoised speech as defined in [53]. Since in [53] there is no standard measure, we propose an evaluation procedure based on the recognition performance achieved by a state-of-the-art speech to text commercial software [45].

In all the speech experiments we used a set of 256 logarithmically spaced fundamental frequencies in the range from 75 Hz to 280 Hz. The frame analysis is based on a 512 samples Hanning window, with a hop size of 128. The number of Gaussian models to represent voiced sounds envelopes is set to 16, using 10 eigenvectors for each of them. Unvoiced sounds and the background noise were modeled with one Gaussian with 25 eigenvectors The modeling includes pre-learned temporal dependencies via an HMM, as described in Section 5.5.3. The temporal dynamics for each speaker were learned using a different set of noise free GRID sentences. Figure 5.5 shows an example of the transition matrices obtained for the Gaussians and fundamental frequencies. One can see that the latter presents a dominant diagonal, which means that smaller changes are more likely to occur than larger ones.

For our first evaluation we employed a subset of the GRID database composed of 30 sentences spoken by two males and two females. All the speech audio signals were downsampled to 8 kHz (the original sampling frequency of 25 kHz) to meet the noise's sampling rate. We first evaluated the algorithm considering three different SNRs: -3 dB, 0 dB and 3 dB. We selected 5 different noisy environments from the Aurora database, using 3 different instances of each of noise in the evaluation. Considering all the possible combinations, a total of 6750 noisy audio excerpts were evaluated.

Results are presented in Table 5.4. As a benchmark comparison, we also show the separation results when using standard NMF for modeling both the noise and speech signal. For each case we selected the number of the atoms for which the best results were obtained: 35 atoms for the speech dictionary and 10 for the background noise one. Most of the atoms

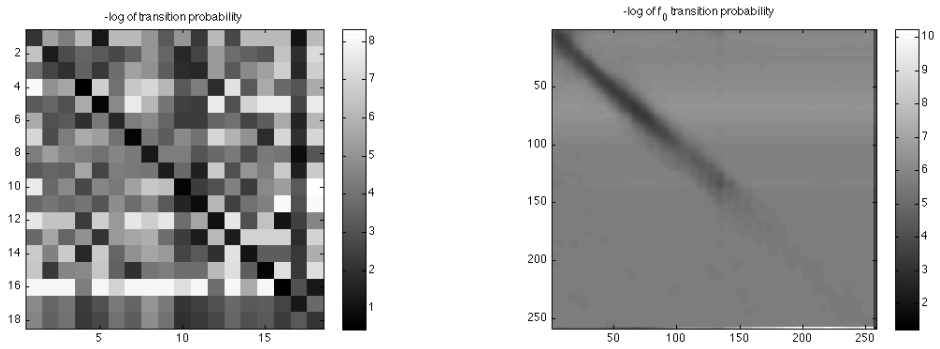


Figure 5.5: In this figure we show an example of the HMM transition matrices. (Left) Transition probability between Gaussians in voiced sounds (with states numbered 1-16), unvoiced sounds (state 17), and silence (state 18), (Right) Probability of of fundamental frequency change (states 1-256), unvoiced sounds (state 257), and silence (state 258)

	SAR	SIR	SDR	SNR
with HMM	3.53	13.17	2.80	3.35
w/o HMM	3.42	13.15	2.71	3.23

Table 5.3: Results for 5 different speakers of the GRID database at a SNR of -5dB. Results with and without the HMM are shown.

present a clear harmonic structure. It is clear that having only 35 atoms severely restricts the capability to represent a wide range of fundamental frequencies. Furthermore, even if 35 would have been enough to cover the main possible fundamental frequencies for a particular speaker, the timbre representation gets strongly reduced. This can be clearly appreciated when listening the denoised audio outputs, in which the intonation is well recovered but the phoneme variability is almost completely lost.

In Table 5.3 SIR, SAR, SDR and final SNR for a set of 5 speakers are reported for an initial SNR of -5dB with and without the HMM. An increase of around 0.1dB is obtained in all the measurements.

In our second evaluation we wanted to measure the intelligibility of the denoised audio. In order to do so, we fed it as input to the speech-to-text MacSpeech Scribe software [45],

sPCA	-3dB	0 dB	3 dB
SAR	4.99	7.09	8.79
SIR	13.75	13.39	12.28
SDR	4.21	5.94	6.92
SNR	4.56	5.92	6.63

NMF	-3dB	0 dB	3 dB
SAR	2.55	4.44	6.44
SIR	8.55	8.47	8.35
SDR	1.02	2.48	3.82
SNR	2.78	3.23	3.54

Table 5.4: SDR, SAR, SIR and final SNR for different initial SNRs for *(left)* the proposed method *(right)* standard NMF.

	Speaker 2	Speaker 7
Clean audio	99.2	98.3
Noisy audio	74.2	80.8
Denoised audio	88.3	82.5

Table 5.5: Intelligibility in terms of percentage of correctly transcribed words.

which is a state-of-the-art commercial application that reaches an accuracy of over 99% on clean speech signals. One of the reasons to select MacSpeech Scribe is that it permits to test the system on a prerecorded audio and, more importantly, can be trained with a user defined text and its corresponding audio. At the same time, it gives the possibility to include a corpus text file to train the vocabulary and grammar as part of the training.

The training process needs an initial calibration step with 15 seconds of audio and the corresponding transcription. After this calibration, the system needs at least 60 more seconds of annotated speech to fully train the system. We used a set of 100,000 randon valid GRID sentences for training the vocabulary and grammar. In all cases we set the initial profile to “British English,” as all the speakers in the GRID database have that accent.

As the initial calibration and first training step require good signal quality, we performed them using the clean original signals. This enables to continue training the system by progressively including the lower quality signals, training with both the noisy and denoised signals. At the same time, the training procedure intends to be as objective as possible and to give the best possible training of the software even for relatively low quality signals.

In Table 5.5 we show the results obtained over a test set of 20 phrases for two speakers, one male and one female. The performance is measured as the percentage of correct transcribed words. The results were evaluated for the clean original audio, the noisy audio at a SNR of 3 dB, and the denoised results for the latter. We observe that the proposed algorithm helps in the recognition process. The speech to text software performance decreases sharply for SNRs of 0dB and below, giving working points in which no useful comparison can be made.

5.7 Conclusions

In this chapter we introduced a new framework for representing quasi-harmonic audio signals that can be used to address audio source separation problems. This model is well suited in particular for tackling score-informed source separation of musical mixtures. The method has the ability to model the pitch and envelope of a sound source independently. This permits the representation of signals with combinations of pitches and envelopes not previously observed. Moreover, it allows to easily incorporate meta-data such as the musical score indicating the signal to be represented. In this way, two or more musical instruments with the same characteristics can be separated. The characterized set of signals can be extended to incorporate non-harmonic sounds by defining filters $\mathbf{h_i}$ with the appropriate masking of spectral components. The proposed probabilistic setting leads to an implicit classification of harmonic and non-harmonic sounds by simply selecting the representation that maximizes the corresponding posterior probability. Learned temporal dynamics are also included via a first order hidden Markov model, allowing for the representation of richer signals such as speech. The method has been evaluated in the score-informed monaural source separation problem with synthetic and real data showing a performance comparable with those reported in the literature. When including temporal dynamics the

model can capture the complexity of speech signals being very efficient for the denoising of speech in the presence of non-stationary noise.

6 Learning efficient structured models

6.1 Chapter summary

As described in the introduction, one of the objectives of this thesis was to exploit the data structure in order to produce efficient coding schemes, capable of running in real-time applications. In this chapter we present a comprehensive framework for structured sparse coding and robust low dimensional modeling extending the recent ideas of using trainable fast regressors to approximate exact sparse codes. For this purpose, we propose efficient feed forward networks whose architecture faithfully approximate the exact solvers with a fraction of the complexity of the standard optimization methods. We also show that by using different training objective functions, the trainable sparse encoders are no longer restricted to be mere approximants of the exact sparse code for a pre-given dictionary, as in earlier formulations, but can be rather used as full-featured sparse encoders or even modelers. A simple implementation shows several orders of magnitude speedup compared to the state-of-the-art at minimal performance degradation, making the proposed framework suitable for real time and large-scale applications. We present an extensive experimental evaluation in several computer vision and pattern recognition applications.

6.2 Introduction

Sparse coding and robust low dimensional modeling have become one of the hot topics of several communities in the last five years or so. As discussed in the introduction, both

models aim at finding a simple explanation for very complex high dimensional data.

The main challenge of all optimization-based sparse coding and modeling approaches is their relatively high computational complexity. Consequently, a significant amount of effort has been devoted to developing efficient optimization schemes [4, 21, 60, 125]. A very similar situation is observed in the robust low rank recovery problems. A great deal of optimization algorithms have been proposed for efficiently solving the different formulations low rank estimation [10, 12, 64, 74, 94, 71]. Despite the permanent progress reported in the literature, the state-of-the-art algorithms require tens or hundreds of iterations to converge, making them infeasible for real-time or very large (modern size) applications. Furthermore, the iterative nature of these optimization algorithms make the complexity data-dependent, which is a significant obstacle in time-critical settings.

In the sparse coding domain, techniques based on sparse representations in learned over-complete dictionaries produced outstanding results in various computer vision and signal processing problems, but they are often prohibitively costly for real-time computation. This motivated significant effort in the deep learning community aiming at overcoming this problem. On one hand, several works concentrated on proposing systems capable of producing sparse codes, aiming to bring the success of the exact sparse coding algorithms to extremely efficient deep learning schemes, e.g., [91, 37]. In a different approach, several works proposed learning non-linear regressors (or predictors) capable of producing good approximations of the true sparse codes in a fixed amount of time [46, 54]. The insightful work in [39] introduced an approach in which the regressors are multilayer artificial neural networks with an architecture inspired by first order optimization algorithms for solving sparse coding problems. These regressors are trained to minimize the mean squared error between the predicted and exact codes over a given training set, and produced high quality approximations of sparse codes for vectors following the same distribution as the training sample.

Motivated by the latter approach, in this work we propose to extend these ideas in a number of ways. First we show that these ideas can be generalized to handle structured sparse regression and robust low rank recovery problems. We propose to design regressors capable of approximating the true optimization algorithms in a very fast way. We follow [39] and base the architecture of the encoders on the iterations of exact algorithms. In each case we obtain problem-specific network architectures, which according to our experimental evaluation, play a central role in the obtained performance. The extension to robust low rank recovery problems, such as RPCA, is not straight forward. Unlike the standard sparse coding setting, the exact first order RPCA algorithms cannot be used directly, as each iteration involves SVD. As a remedy, we use an algorithm inspired by the non-convex optimization techniques proposed in [94].

We propose a training objective function that allows the encoders to be trained in an online manner on the very same data vectors fed to them. This also makes the fast encoders no more restricted to work with a specific distribution of input vectors known *a priori* (limitation existing, for example, in [39]), and removes the need to run the exact algorithms beforehand. While differently motivated, in the case in which the dictionary is learned, the framework is related to recent efforts in producing NN based sparse representations, see [37, 91] and references therein. It can be interpreted as an online trainable sparse auto-encoder [37] with a sophisticated encoder and simple linear decoder. The higher complexity of the proposed architecture in the encoder allows the system to produce accurate estimates of true structured sparse codes, as will be empirically shown in Section 7.6.

The differentiability of the proposed encoders with respect to the input, output and training data allows a very simple to handle changes in the input data or incorporating discriminative regularizations. For example, several applications of RPCA rely on the critical assumption that the given input vectors are *aligned* with respect to a group of geometric transformations [83]. The proposed regressors can handle these changes in a very straight

forward way, in contrast to the state-of-the art techniques for addressing this problem involve the computation of several RPCA problems by changing the individual transformations applied to each input data vector.

The rest of the chapter is organized as follow. In Section 6.3 and in Section 6.4 we introduce respectively the general structured sparse coding problem and the online RPCA one. For each case, we discuss the corresponding optimization algorithms that will be later used to inspire the architecture of the proposed encoders. In Section 6.5 we present the new sparse encoders and the new objective functions used for their training. Experimental results are presented in Section 7.6. Finally, conclusions are drawn in Section 6.7.

6.3 Structured sparse models

The underlying assumption of sparse models is that the input vectors can be reconstructed accurately as a linear combination of some (usually learned) basis vectors (factors or dictionary atoms) with a small number of non-zero coefficients. *Structured* sparse models further assume that the pattern of non-zero coefficients exhibits a specific structure known *a priori*.

Let $\mathbf{D} \in \mathbb{R}^{m \times p}$ be a dictionary with p m-dimensional atoms. We define groups of atoms through their indexes, $G \subseteq \{1, \ldots, p\}$. Then, we define a group structure, $\mathcal{G}$, as collection of groups of atoms, $\mathcal{G} = \{G_1, \ldots, G_{|\mathcal{G}|}\}$. For an input vector $\mathbf{x} \in \mathbb{R}^m$, the corresponding structured sparse code, $\mathbf{a} \in \mathbb{R}^p$, associated to the group structure $\mathcal{G}$, can be obtained by solving the convex program,

$$\min_{\mathbf{a} \in \mathbb{R}^p} \frac{1}{2} \|\mathbf{x} - \mathbf{D}\mathbf{a}\|_2^2 + \psi(\mathbf{a}), \quad \text{with} \quad \psi(\mathbf{a}) = \sum_{r \in \mathcal{G}} \lambda_r \left\|\mathbf{a}_r\right\|_2, \tag{6.3.1}$$

where the vector $\mathbf{a}_r \in \mathbb{R}^{|G_r|}$ contains the coefficients of $\mathbf{a}$ belonging to group r, and λ_r are scalar weights controlling the sparsity level.

The regularizer function ψ in (6.3.1) can be seen as a generalization of the ℓ_1 regularizer used in standard sparse coding, as the latter arises from the special case of singleton groups $\mathcal{G} = \{\{1\}, \{2\}, \ldots, \{p\}\}$ and setting $\lambda_r = 1$. As such, its effect on the groups of $\mathbf{a}$ is a natural generalization of the one obtained with standard sparse coding: it "turns on" and "off" atoms in groups according to the structure imposed by $\mathcal{G}$.

Several important structured sparsity settings can be cast as particular cases of (6.3.1): *sparse coding*, as mentioned above, which is often referred to as Lasso [105] or basis pursuit [15, 22]; *group sparse coding*, a generalization of the standard sparse coding to the cases in which the dictionary is sub-divided into groups that are known to be active or inactive simultaneously [129], in this case $\mathcal{G}$ is a partition of $\{1, \ldots, p\}$; *hierarchical sparse coding*, assuming a hierarchical structure of the non-zero coefficients [131, 50, 101]. This is a more general setting that contains as a particular case the HiLasso framework proposed as part of this thesis in Chapter 4. In this general setting, the groups in $\mathcal{G}$ form a hierarchy with respect to the inclusion relation (a tree structure), that is, if two groups overlap, then one is completely included in the other one; and *collaborative sparse coding* generalizing the concept of structured sparse coding to collections of input vectors by promoting given patterns of non-zero elements in the coefficient matrix [30, 101]. Again this concept was used in the C-HiLasso in Chapter 4.

6.3.1 Optimization algorithms for structured sparsity

State-of-the-art approaches for solving (6.3.1) rely on the family of proximal splitting methods (see [2, 17] and references therein). In what follows, we briefly introduce proximal methods and present a new algorithm for solving hierarchical sparse coding problems that will be further used to construct trainable sparse encoders.

Algorithm 3: Forward-backward splitting method.

input : Data $\mathbf{x}$, dictionary $\mathbf{D}$, regularizer ψ.
output: Sparse code $\mathbf{a}$.
Define $\mathbf{S} = \mathbf{I} - \frac{1}{\alpha}\mathbf{D}^{\mathrm{T}}\mathbf{D}$, $\mathbf{W} = \frac{1}{\alpha}\mathbf{D}^{\mathrm{T}}$, $t = \frac{1}{\alpha}$.
Initialize $\mathbf{a} = \mathbf{0}$ and $\mathbf{b} = \mathbf{W}\mathbf{x}$.
repeat
  $\mathbf{a} = \mathrm{prox}_{t\psi}(\mathbf{b})$
  $\mathbf{b} = \mathbf{b} + \mathbf{S}\mathbf{a}$
until *convergence* ;

Forward-backward splitting

The forward-backward splitting method is designed for solving unconstrained optimization problems in which the cost function can be split as

$$\min_{\mathbf{a}\in\mathbb{R}^m} f_1(\mathbf{a}) + f_2(\mathbf{a}), \tag{6.3.2}$$

where f_1 is convex and differentiable with a $\frac{1}{\alpha}$-Lipschitz continuous gradient, and f_2 is convex extended real valued and possibly non-smooth. Clearly, problem (6.3.1) falls in this category by considering $f_1(\mathbf{a}) = \frac{1}{2}\|\mathbf{x} - \mathbf{D}\mathbf{a}\|_2^2$ and $f_2(\mathbf{a}) = \psi(\mathbf{a})$.

The forward-backward splitting method with fixed constant step defines a series of iterates,

$$\mathbf{a}^{k+1} = \mathrm{prox}_{\alpha f_2}(\mathbf{a}^k - \frac{1}{\alpha}\nabla f_1(\mathbf{a}^k)), \tag{6.3.3}$$

where $\mathrm{prox}_{f_2}(\mathbf{a}) = \underset{\mathbf{u}\in\mathbb{R}^m}{\mathrm{argmin}}\, ||\mathbf{u} - \mathbf{a}||_2^2 + f_2(\mathbf{u})$ denotes the proximal operator of f_2. The procedure is given in Algorithm 3.

The forward-backward method becomes particularly interesting when the proximal operator of ψ can be computed exactly and efficiently, e.g., in standard or group-structured sparse coding. When the groups of $\mathcal{G}$ overlap arbitrarily, there is no efficient way of doing so directly. However, there exist important exceptions such as the hierarchical setting with tree-structured groups which is discussed in the sequel.

Proximal operators

To simplify the notation, we will henceforth formulate all the derivations for the case of two-level hierarchical sparse coding, referred as HiLasso [33, 101]. This captures the essence of hierarchical sparse models and the generalization to more layers [50] or to a collaborative scheme [101] is straightforward.

The HiLasso model was introduced for simultaneously promoting sparsity at both, group and coefficient level. Given a partition $\mathscr{P} = \{G_1, \ldots, G_{|\mathscr{P}|}\}$, the group structure $\mathscr{G}$ can be expressed as the union of two partitions: $\mathscr{P}$ and the set of singletons. Thus, the regularizer ψ becomes

$$\psi(\mathbf{a}) = \sum_{j=1}^{p} \lambda_j \left\|\mathbf{a}_j\right\|_1 + \sum_{r=1}^{|\mathscr{P}|} \mu_r \left\|\mathbf{a}_r\right\|_2 . \tag{6.3.4}$$

The proximal operator of ψ can then be computed in closed form. Given a partition of the group of indexes, $\mathscr{P}$, and a vector of thresholding parameters $\boldsymbol{\lambda} \in \mathbb{R}^{|\mathscr{P}|}$, we define the group separable operator $\pi_{\boldsymbol{\lambda}} : \mathbb{R}^p \to \mathbb{R}^p$ for $r = 1, \ldots, |\mathscr{P}|$ as

$$\pi_{\boldsymbol{\lambda}}(\mathbf{a})_r = \frac{\max\{0, \left\|\mathbf{a}_r\right\|_2 - \lambda_r\}}{\left\|\mathbf{a}_r\right\|_2} \mathbf{a}_r \tag{6.3.5}$$

if $\left\|\mathbf{a}_r\right\|_2 > 0$, and $\mathbf{0}$ otherwise. Note that $\pi_{\boldsymbol{\lambda}}$ applies a vector soft-thresholding to each group in $\mathscr{P}$. The proximal operator of (6.3.4) can be expressed as [50, 101],

$$\pi_{\lambda,\mu}(\mathbf{a}) = \pi_{\mu}(\pi_{\boldsymbol{\lambda}}(\mathbf{a})), \tag{6.3.6}$$

a composition of the proximal operators associated to each of the partitions in $\mathscr{G}$: $\mathscr{P}$ and the set of singletons.

The Lasso problem is a particular case of HiLasso with $\boldsymbol{\mu} = \mathbf{0}$ and $\boldsymbol{\lambda} = \lambda \mathbf{1}$, in which the proximal operator in (6.3.6) reduces to the scalar soft-thresholding operator and Algorithm 3 corresponds then to the popular iterative shrinkage and thresholding algorithm

Algorithm 4: BCoFB algorithm.

input : Data $\mathbf{x}$, structured dictionary $\mathbf{D}$, $\boldsymbol{\lambda}$, $\boldsymbol{\mu}$.
output: Structured sparse code $\mathbf{a}$.
Bound Lipschitz constant $\alpha \leq \max_r \left\|\mathbf{D}_r\right\|_2^2$
Define $\mathbf{S} = \mathbf{I} - \frac{1}{\alpha}\mathbf{D}^{\mathrm{T}}\mathbf{D}$, $\mathbf{W} = \frac{1}{\alpha}\mathbf{D}^{\mathrm{T}}$, $\mathbf{s} = \frac{1}{\alpha}\boldsymbol{\mu}$, and $\mathbf{t} = \frac{1}{\alpha}\boldsymbol{\lambda}$.
Initialize $\mathbf{a} = \mathbf{0}$ and $\mathbf{b} = \mathbf{W}\mathbf{x}$.
repeat
  $\mathbf{y} = \pi_{\mathbf{s},\mathbf{t}}(\mathbf{b})$
  $\mathbf{e} = \mathbf{y} - \mathbf{a}$
  $g = \arg\max_r \left\|\mathbf{e}_r\right\|_2$
  $\mathbf{b} = \mathbf{b} + \mathbf{S}_g \mathbf{e}_g$
  $\mathbf{a}_g = \mathbf{y}_g$
until *until convergence* ;
Output $\mathbf{a} = \pi_{\mathbf{s},\mathbf{t}}(\mathbf{b})$

(ISTA) [21, 4].

Block-coordinate forward-backward algorithm

In Algorithm 3, every iteration requires the update of all the groups of coefficients in the partition $\mathscr{P}$, according to (6.3.6). Inspired by the coordinate descent algorithm (CoD) introduced for standard sparse coding in [60], we propose a new variant of the forward-backward algorithm, referred to as Block-Coordinate Forward-Backward algorithm (BCoFB), that updates only one (carefully chosen) group at a time. The iterates of BCoFB are given by,

$$\begin{aligned} \mathbf{p} &= \pi_{\lambda,\mu}(\mathbf{a}^k), \\ \mathbf{a}^{k+1} &= \mathbf{a}^k, \\ \mathbf{a}_g^{k+1} &= \mathbf{p}_g - \tfrac{1}{\alpha}\mathbf{D}_g^{\mathrm{T}}(\mathbf{D}_g\mathbf{p}_g^k - \mathbf{x}_g), \end{aligned} \tag{6.3.7}$$

where g is the index of the group in $\mathscr{P}$ to be updated at the k-th iteration, according to some selection rule to be discussed next. BCoFB is a descent algorithm for any group selector, that is, the HiLasso objective function, $f(\mathbf{a}) = \frac{1}{2}\|\mathbf{x} - \mathbf{D}\mathbf{a}\|_2^2 + \psi(\mathbf{a})$, decreases:

Proposition 1. Let $\{\mathbf{a}^k\}_{k\in\mathbb{N}}$ be a sequence generated by the iterates defined in (6.3.7). For

any $k > 1$,

$$\frac{\alpha}{2}||\mathbf{a}_g^{k+1} - \mathbf{a}_g^k||_2^2 \leq f(\mathbf{a}^k) - f(\mathbf{a}^{k+1}),$$

for any possible group selection $g = 1 \ldots |\mathcal{G}|$.

Clearly, the group selection rule plays a fundamental role in the convergence rate of the algorithm. A natural idea is to update the group of coefficients that would produce the largest decrease in the cost function. However, this alternative is computationally very demanding, as it requires the evaluation of the cost function for every possible group update. The result of Proposition 1 provides, at any given iteration, a lower bound in the decrease of the cost function for each possible group update, expressed in terms of the norm of the difference between iterates (which only requires the evaluation of the proximal operator). A natural heuristic is, at each iteration, to update the group for which the largest lower bound is obtained. This is summarized in Algorithm 4.

In the case of standard sparsity, Algorithm 4 with $\alpha = 1$ is identical to CoD, which was derived from completely different arguments in [60] and later used in [39] to build trainable sparse encoders.

6.4 Online Robust PCA

In this section we present a formulation of robust PCA that later will be translated to the framework of trainable encoders. We start this section we defining robust low dimensional projections as a convex program and then present an optimization algorithm for solving it.

6.4.1 Robust PCA via non-convex factorization

We tackle the robust PCA problem by solving the unconstrained optimization problem

$$\min_{\mathbf{L},\mathbf{O}\in\mathbb{R}^{m\times n}} \frac{1}{2}\left\|\mathbf{X}-\mathbf{L}-\mathbf{O}\right\|_F^2 + \lambda_*\left\|\mathbf{L}\right\|_* + \lambda\left\|\mathbf{O}\right\|_1. \tag{6.4.8}$$

This formulation is equivalent to (2.3.6) in the sense that for every $\epsilon > 0$ one can find a $\lambda_* > 0$ such that (2.3.6) and (6.4.8) admit the same solutions.

As the ℓ_1-norm encourages sparsity with vectors, the nuclear norm promotes low-rank in matrices. In [93] it was shown that the nuclear norm of a matrix can be reformulated as a penalty over all possible factorizations,

$$\|\mathbf{L}\|_* = \min_{\mathbf{U},\mathbf{S}} \frac{1}{2}\|\mathbf{U}\|_F^2 + \frac{1}{2}\|\mathbf{S}\|_F^2 \qquad \text{s.t.} \quad \mathbf{US} = \mathbf{L}. \tag{6.4.9}$$

The minimum is achieved by the SVD: if $\mathbf{L} = \mathbf{U}_L\mathbf{\Sigma}\mathbf{V}_L^T$ then the minimum of (7.4.4) is $\mathbf{U} = \mathbf{U}_L\mathbf{\Sigma}^{\frac{1}{2}}$ and $\mathbf{S} = \mathbf{\Sigma}^{\frac{1}{2}}\mathbf{V}_L$. This factorization has been recently exploited in parallel processing across multiple processors to produce state-of-the-art algorithms for matrix completion problems [94] and it was used as an alternative approach to robustifying PCA in [71].

In (6.4.8), neither the rank of $\mathbf{L}$ nor the level of sparsity in $\mathbf{O}$ are assumed known *a priori*. However, in most computer vision applications, it is a reasonable to have a rough upper bound of the rank, say $\text{rank}(\mathbf{L}) = \bar{q} \leq q$. Combining this with (7.4.4), we can reformulate (6.4.8) as

$$\min_{\mathbf{U},\mathbf{S},\mathbf{O}} \frac{1}{2}\|\mathbf{X} - \mathbf{US} - \mathbf{O}\|_F^2 + \frac{\lambda_*}{2}(\|\mathbf{U}\|_F^2 + \|\mathbf{S}\|_F^2) + \lambda\|\mathbf{O}\|_1, \tag{6.4.10}$$

with $\mathbf{U} \in \mathbb{R}^{m\times q}$, $\mathbf{S} \in \mathbb{R}^{q\times n}$, and $\mathbf{O} \in \mathbb{R}^{m\times n}$. This decomposition reveals a lot of structure hidden in the problem. The low rank component can now be thought as an under-complete dictionary $\mathbf{U}$, with q atoms, multiplied by a matrix $\mathbf{S}$ containing in its columns the corresponding coefficients for each data vector in $\mathbf{X}$. This interpretation brings our problem close to that of dictionary learning in the sparse modeling domain discussed before.

While this new factorized formulation drastically reduces the number of optimization variables from $2nm$ to $q(n+m)$, problem (6.4.10) is no longer convex. Fortunately, it can be shown that any stationary point of (6.4.10), $\{\mathbf{U},\mathbf{S},\mathbf{O}\}$, satisfying $\|\mathbf{X} - \mathbf{US} - \mathbf{O}\|_F^2 \leq \lambda_*$ is

an optimal solution of (6.4.10) [70]. Thus, problem (6.4.10) can be solved using an alternating minimization or block coordinate scheme, in which the cost function is minimized with respect to each individual optimization variable while keeping the other ones fixed, without the risk of falling into a stationary point that may not be globally optimum.

6.4.2 Robust low dimensional projections

Let us assume that we have already learned a low dimensional model, $\mathbf{U} \in \mathbb{R}^{m\times q}$, from some data $\mathbf{X} \approx \mathbf{US} + \mathbf{O} \in \mathbb{R}^{m\times n}$. Suppose that we are given a new input vector $\hat{\mathbf{x}} \in \mathbb{R}^m$ drawn from the same distribution as $\mathbf{X}$. Then $\hat{\mathbf{x}}$ can be decomposed as $\hat{\mathbf{x}} = \mathbf{U}\hat{\mathbf{s}} + \hat{\mathbf{n}} + \hat{\mathbf{o}}$, where $\mathbf{U}\hat{\mathbf{s}}$ represents the low dimensional component, $\hat{\mathbf{n}} \in \mathbb{R}^m$ is a small perturbation and $\hat{\mathbf{o}} \in \mathbb{R}^m$ is a sparse outlier vector. We propose to do it by extending (6.4.10)

$$\min_{\mathbf{s}\in\mathbb{R}^q, \mathbf{o}\in\mathbb{R}^m} \frac{1}{2} \|\hat{\mathbf{x}} - \mathbf{U}\mathbf{s} - \mathbf{o}\|_2^2 + \frac{\lambda_*}{2} \|\mathbf{s}\|_2^2 + \lambda \|\mathbf{o}\|_1 . \tag{6.4.11}$$

Unlike dictionary learning problems, here the columns of the dictionary $\mathbf{U}$ are not constrained to have unit norm. In fact, the differences in the norms of the different atoms play a crucial role in the estimation weighting the relevance of the atoms in the low dimensional distribution and appearing in the objective function as the quadratic term $\|\mathbf{s}\|_2^2$. To give some further intuition we analyze the program (6.4.11) and its relation with the possible solutions of (6.4.10). As discussed in the previous section, if the upper bound for the rank of the true low dimensional model is correct, $\bar{q} \leq q$, any pair of matrices $\{\mathbf{U}, \mathbf{S}\}$ found as a stationary point of (6.4.10), satisfying $||\mathbf{X} - \mathbf{US} - \mathbf{O}||_F^2 \leq \lambda_*$, is guaranteed to verify, $\mathbf{L} = \mathbf{US}$. For simplicity in the notation and without loss of generality, in the sequel we assume that $\bar{q} = q$. Then, the solutions of program (7.4.8), $\{\hat{\mathbf{s}}_1, \hat{\mathbf{o}}_1\}$, satisfy, $\mathbf{U}\hat{\mathbf{s}}_1 = \tilde{\mathbf{U}}\hat{\mathbf{s}}_2$ and $\hat{\mathbf{o}}_1 = \hat{\mathbf{o}}_2$, where $\{\hat{\mathbf{s}}_2, \hat{\mathbf{o}}_2\}$ are the solutions obtained by substituting $\mathbf{U}$ by $\tilde{\mathbf{U}} = \mathbf{U}_L \mathbf{\Sigma}^{\frac{1}{2}}$ in

(7.4.8). Applying the change of coordinates $\mathbf{s} = \mathbf{\Sigma}^{-\frac{1}{2}}\mathbf{w}$, this new problem can be written as

$$\min_{\mathbf{w}\in\mathbb{R}^q,\mathbf{o}\in\mathbb{R}^m} \frac{1}{2}\left\|\hat{\mathbf{x}} - \mathbf{U}_L\mathbf{w} - \mathbf{o}\right\|_2^2 + \frac{\lambda_*}{2}\sum_{i=1}^{q}\frac{w_i^2}{\sigma_i} + \lambda\left\|\mathbf{o}\right\|_1, \tag{6.4.12}$$

where the w_i's are the individual coefficients of $\mathbf{w}$ and the σ_i's are the singular values of $\mathbf{L}$, $\mathbf{\Sigma} = \mathrm{diag}(\sigma_1, \ldots, \sigma_q)$. Note that $\mathbf{U}\hat{\mathbf{s}}_1 = \mathbf{U}_L\mathbf{w}$. The second term in the cost in (6.4.12) acts as a regularizer that encourages the usage of the coefficient of $\mathbf{w}$ corresponding to the dominant directions (larger singular values) of $\mathbf{L}$.

The robust low dimensional projection (7.4.8) is a convex program that can be solved using several methods. We are interested in choosing an optimization algorithm that can be further used to define the architecture of trainable encoders for simultaneously estimating $\mathbf{s}$ and $\mathbf{o}$. With this in mind, we choose to use the alternating minimization scheme, described in Algorithm 5. The solution of (6.4.14) is given by

$$\mathbf{s} = (\mathbf{U}^{\mathrm{T}}\mathbf{U} - \lambda_*\mathbf{I})^{-1}\mathbf{U}^{\mathrm{T}}(\mathbf{x}_t - \mathbf{o}) \quad \text{and} \quad \mathbf{o} = \pi_\lambda(\hat{\mathbf{x}} - \mathbf{U} * \mathbf{s}), \tag{6.4.13}$$

when fixing $\mathbf{o}$ and $\mathbf{s}$ respectively. Here $\pi_{\boldsymbol{\lambda}}$ is the scalar soft-thresholding operator with parameter $\boldsymbol{\lambda} \in \mathbb{R}^m$, which applies a soft-threshold λ_i to each component of the input vector. In this case, $\boldsymbol{\lambda} = \lambda\mathbf{1}$.

6.4.3 Online robust PCA

In Section 6.4.1 we assumed that the entire data matrix $\mathbf{X}$ was available a priori. We now address the case when the data samples $\{\mathbf{x}_t\}_{t\in\mathbb{N}}$, $\mathbf{x}_t \in \mathbb{R}^m$, arrive sequentially, the index t should be understood as a temporal variable. Online robust PCA aims at estimating and refining the model as the data comes in. The need for online algorithm appears naturally in a various applications, e.g., when large volumes of data are permanently generated over time. Other applications aim at estimating models for dynamic data constantly changing

Algorithm 5: Alternating minimization scheme for solving (6.4.11).

input : Data $\hat{\mathbf{x}}$, dictionary $\mathbf{U}$, parameters λ_* and λ.
output: Coefficient vector $\mathbf{s}$ and outlier vector $\mathbf{o}$.
Define $\mathbf{H} = (\mathbf{U}^T\mathbf{U} - \lambda_*\mathbf{I})^{-1}$ and $\mathbf{W} = \mathbf{U}\mathbf{H}\mathbf{U}^T$, $\boldsymbol{\lambda} = \lambda\mathbf{1}$.
Initialize $\mathbf{y} = \mathbf{0}$ and $\mathbf{b} = (\mathbf{I} - \mathbf{W})\mathbf{x}$.
repeat
  $\mathbf{o} = \pi_{\boldsymbol{\lambda}}(\mathbf{b})$
  $\mathbf{b} = \mathbf{b} + \mathbf{W}(\mathbf{o} - \mathbf{y})$
  $\mathbf{y} = \mathbf{o}$
until *until convergence* ;
Output $\mathbf{o}$ and $\mathbf{s} = \mathbf{H}(\mathbf{x} - \mathbf{o})$.

over time. Finally, online learning has been also extensively used when the available training data is simply too large to be handled altogether [20].

We propose to address online robust PCA extending the approach presented in Section 6.3. An alternating minimization algorithm for solving the online counterpart of (6.4.10) goes as follows. When a new data vector $\mathbf{x}_t$ is received, we first obtain its corresponding pair of variables $\{\mathbf{s}_t, \mathbf{o}_t\}$ given the current model estimate, $\mathbf{U}_{t-1}$.

$$\{\mathbf{s}_t, \mathbf{o}_t\} \quad = \quad \underset{\mathbf{s},\mathbf{o}}{\operatorname{argmin}} \frac{1}{2}\left\|\mathbf{x}_t - \mathbf{U}_{t-1}\mathbf{s} - \mathbf{o}\right\|_2^2 + \frac{\lambda_*}{2}\|\mathbf{s}\|_2^2 + \lambda\|\mathbf{o}\|_1 . \tag{6.4.14}$$

Then, we update the model using the projections, $\{\mathbf{s}_j\}_{j\leq t}$ and $\{\mathbf{o}_j\}_{j\leq t}$, computed during the previous steps of the algorithm,

$$\mathbf{U}_t \quad = \quad \underset{\mathbf{U}}{\operatorname{argmin}} \sum_{j=1}^{t} \beta_j \left(\frac{1}{2}\left\|\mathbf{x}_j - \mathbf{U}\mathbf{s}_j - \mathbf{o}_j\right\|_2^2 + \frac{\lambda_*}{2}\|\mathbf{U}\|_F^2\right), \tag{6.4.15}$$

where $\beta_j \in [0,1]$ is a "forgetting factor" that can be added to rescale the "old" information so that newer estimates $\{\mathbf{s}_j, \mathbf{o}_j\}$ have more weight. The use of forgetting factors is standard in online learning [71, 66].

Problem (6.4.14) is identical to (6.4.11) and can be solved using Algorithm 5 setting

$\mathbf{U} = \mathbf{U}_{t-1}$ and $\hat{\mathbf{x}} = x_t$. There are mainly two options for solving (6.4.15). The first one is to solve it recursively from previous estimates, using strategies such as block-coordinate descent methods with warm restarts [66] or recursive LS [71]. The other option is to directly solve the system of equations

$$\mathbf{U}_t \left(\sum_{j=1}^{t} \beta_j \mathbf{s}_j \mathbf{s}_j^{\mathrm{T}} + \lambda_* \mathbf{I} \right) = \sum_{j=1}^{t} \beta_j (\mathbf{x}_j - \mathbf{o}_j) \mathbf{s}_j^{\mathrm{T}}, \tag{6.4.16}$$

where again $\beta_j \in [0, 1]$ is a forgetting factor.

While the recursive strategies have in general less computational complexity, in particular for large scale problems, they require more memory storage. The optimal setting is consequently problem dependent.

6.5 Online robust PCA via fast trainable encoders

6.5.1 Trainable encoders for sparse decompositions

Given a possibly over-complete dictionary $\mathbf{D} \in \mathbb{R}^{m \times p}$ and a data vector $\mathbf{x} \in \mathbb{R}^m$, the standard sparse coding problem is to find a sparse coefficient vector $\mathbf{a} \in \mathbb{R}^p$ such that $\mathbf{x} \approx \mathbf{D}\mathbf{a}$. This can be solved via the convex program ($\lambda \in \mathbb{R}^+$)

$$\min_{\mathbf{a} \in \mathbb{R}^p} \frac{1}{2} \|\mathbf{x} - \mathbf{D}\mathbf{a}\|_2^2 + \lambda \|\mathbf{a}\|_1 . \tag{6.5.17}$$

In order to make sparse coding feasible in real-time settings, it has been recently proposed to learn non-linear regressors capable of producing good approximations of sparse codes in a fixed amount of time [46, 54]. The main idea is to construct a parametric regressor $\mathbf{h}(\mathbf{x}, \boldsymbol{\Theta})$, with some set of parameters, collectively denoted as $\boldsymbol{\Theta}$, that minimizes the

loss function

$$\mathscr{L}(\mathbf{\Theta}) \quad = \quad \frac{1}{n}\sum_{j=1}^{n} L(\mathbf{\Theta}, \mathbf{x}_j) \tag{6.5.18}$$

on a training set $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$. Here, $L(\mathbf{\Theta}, \mathbf{x}_j) = \frac{1}{2}\|\mathbf{a}_j^* - \mathbf{a}_j\|_2^2$, $\mathbf{a}_j^*$ is the exact sparse code of $\mathbf{x}_j$ obtained by solving problem (6.5.17), and $\mathbf{a}_j = \mathbf{h}(\mathbf{x}_j, \mathbf{\Theta})$ is its approximation. Since the application of off-the-shelf regressors produces relatively low-quality approximations [39], Gregor&LeCun proposed regressors implemented as a truncated form of the iterative shrinkage and thresholding algorithm (ISTA) [4] and coordinate descent algorithms (CoD) [60]. Essentially, these regressors are multi-layer artificial neural networks where each layer implements a single iteration of ISTA or CoD, approximating very well sparse codes for input vectors coming from the same distribution as the one used for training.

These sparse encoder architectures are continuous and almost everywhere $\mathscr{C}^1$ with respect to the parameters and the inputs. Differentiability with respect to the parameters allows the use of (sub)gradient descent methods for training, while differentiability with respect to the inputs allows backpropagation of the gradients and the use of the sparse encoders as modules in globally-trained systems.

6.5.2 Generalized trainable encoders

We now extend Gregor&LeCun's idea to be used in a general setting. This is, we tackle the general problem of hierarchical (structured) sparse code regressors and regressors capable of producing the robust projection of input vectors onto a low dimensional space. In both cases the idea is to construct architectures based on the exact optimization algorithms.

The learning of the network parameters can be done from training data as proposed in [39] and described in Section 6.5.1. This strategy leads to good results as discussed in Section 6.6, however, it suffers from one inherent drawback: it is hopeless to expect the

neural network regressor to produce good decompositions for any input data, preventing the encoders to be used in online applications. In the sequel we discuss a new training alternative to overcome this limitation.

As an alternative to learning, one could simply set the parameters as prescribed by Algorithm 3 and Algorithm 5, terminating it after a small number of iterations. However, it is by no means guaranteed that such a truncated algorithm will produce a good decomposition with the same (small) number of layers.

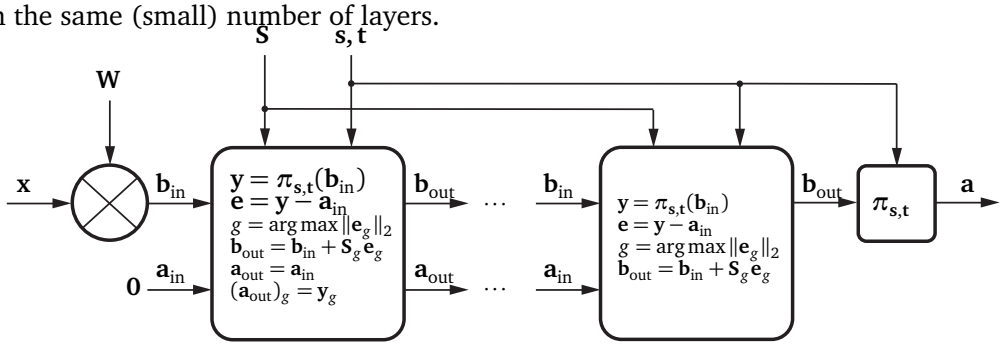


Figure 6.1: BCoFB structured sparse encoder architecture with two levels of hierarchy (a "HiLasso" network).

Hierarchical Sparse Encoders

We consider a feed-forward architecture based on the BCoFB, where each layer implements a single iteration of the BCoFB proximal method (Algorithm 4). The encoder architecture is depicted in Figure 6.2. Each layer essentially consists of the nonlinear proximal operator $\boldsymbol{\pi}_{\mathbf{s},\mathbf{t}}$ followed by a group selector and a linear operation $\mathbf{S}_g$ corresponding to that group. The network parameters are initialized as in Algorithm 4. In the particular case of $\alpha = 1$ and $\mathbf{s} = \mathbf{0}$, the CoD architecture is obtained.

A general sparse coding problem can be viewed as a mapping between a data vector $\mathbf{x}$ and the corresponding sparse code $\mathbf{a}$ minimizing an aggregate of a fitting term and a

(possibly, structured) regularizer,

$$f(\mathbf{x},\mathbf{a}) = \frac{1}{2}\|\mathbf{x}-\mathbf{D}\mathbf{a}\|_2^2 + \psi(\mathbf{a}). \tag{6.5.19}$$

Since the latter objective is trusted as an indication of the code quality, we can train the network to minimize the ensemble average of f on a training set with $\mathbf{a} = \arg\min f(\mathbf{x},\mathbf{a})$ replaced by $\mathbf{a} = \mathbf{h}(\mathbf{x},\boldsymbol{\Theta})$. This results in the training objective

$$\mathscr{L}(\boldsymbol{\Theta}) \quad = \quad \frac{1}{N}\sum_{n}^{N}\beta_j f(\mathbf{x}_n, \mathbf{h}(\mathbf{x}_n,\boldsymbol{\Theta})). \tag{6.5.20}$$

Given an application, one therefore has to select an objective with an appropriate regularizer ψ corresponding to the problem structure, and a sparse encoder architecture consistent with that structure, and train the latter to minimize the objective on a representative set of data vectors. We found that selecting the sparse encoder with the structure consistent with the training objective and the inherent structure of the problem is crucial for the production of high-quality sparse codes.

Learning robust low dimensional projections

Now, we aim at regressors capable of approximating the solution of (6.4.11) for a given fixed dictionary $\mathbf{U}$.

We consider a feed-forward architecture inspired by the alternating minimization scheme described in Algorithm 5, where each layer implements a single iteration. The parameters of the network are the matrices $\mathbf{W}$ and $\mathbf{H}$ and the threshold $\boldsymbol{\lambda}$.[1] The encoder architecture is depicted in Figure 6.2. Each layer essentially consists of the nonlinear thresholding operator $\pi_{\boldsymbol{\lambda}}$ followed by a linear operation $\mathbf{W}$. The network parameters are initialized as in Algorithm 5.

[1] In the network extra flexibility is obtained by learning different thresholds λ_i for each component.

As with structured coding, the robust projection (7.4.8) can be viewed as a mapping between a data vector $\mathbf{x}$ and the corresponding pair $\mathbf{a} = \{\mathbf{s}, \mathbf{o}\}$ minimizing the cost function,

$$f(\mathbf{x}, \mathbf{a}) = \frac{1}{2} \|\mathbf{x} - \mathbf{U}\mathbf{s} - \mathbf{o}\|_2^2 + \frac{\lambda_*}{2} \|\mathbf{s}\|_2^2 + \lambda \|\mathbf{o}\|_1 . \tag{6.5.21}$$

This objective is trusted as an indication of the decomposition quality as explained in Section 6.4.2. Then, the network can be trained to minimize the ensemble average of f on a training set with $\mathbf{a} = \arg\min f(\mathbf{x}, \mathbf{a})$ replaced by $\mathbf{a} = \mathbf{h}(\mathbf{x}, \boldsymbol{\Theta})$. This results in the training objective

$$\mathscr{L}(\boldsymbol{\Theta}) \quad = \quad \frac{1}{N} \sum_{j=1}^{N} \beta_j f(\mathbf{x}_j, \mathbf{h}(\mathbf{x}_j, \boldsymbol{\Theta})). \tag{6.5.22}$$

6.5.3 Finding the network parameters

The minimization of a loss function $\mathscr{L}(\boldsymbol{\Theta})$ with respect to $\boldsymbol{\Theta}$ requires the computation of the (sub)gradients $d\mathscr{L}(\boldsymbol{\Theta}, \mathbf{x}_t)/d\boldsymbol{\Theta}$, which is achieved by a backpropagation procedure. Backpropagation starts with differentiating $\mathscr{L}(\boldsymbol{\Theta}, \mathbf{x}_t)$ with respect to the output of the last network layer, and then propagating the (sub)gradients down to the input layer, multiplying them by the Jacobian matrices of the traversed layers. This results in a simple application of the chain rule,

$$\frac{dL}{d\boldsymbol{\Theta}} = \sum_{i=1}^{T} \frac{dL}{d\mathbf{a}^i} \frac{d\mathbf{a}^i}{d\boldsymbol{\Theta}}, \quad \text{with} \quad \frac{dL}{d\mathbf{a}^i} = \frac{dL}{d\mathbf{a}^{i+1}} \frac{d\mathbf{a}^{i+1}}{d\mathbf{a}^i}, \tag{6.5.23}$$

where $\mathbf{a}^i$ denotes the output of the i-th layer of a T-layer network.

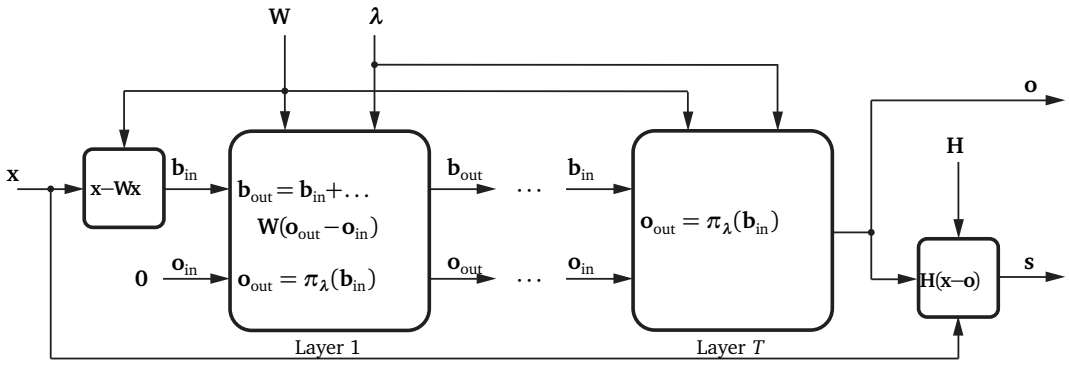


Figure 6.2: Low dimensional encoder architecture with T layers.

6.5.4 Online learning

A challenge often encountered in modern applications is that the flow of new input data is permanent. Then, the model need to be adapted constantly since the data distribution can change over time, calling for developing efficient online techniques [3, 71, 85, 120].

In this section we extend the ideas presented by Gregor&LeCun considering the neural network encoders not only as regressors approximating an iterative algorithm, but as full-featured encoders in their own right. Interpreting the neural networks as standalone robust encoders and removing the reference exact solutions of (7.4.8) and (6.5.19) makes the training problem completely unsupervised. Consequently, one may train the network on the very same data fed to it for robust low dimensional projections. This allows using the proposed framework in online learning applications as explained next. This can be done simply applying the stochastic gradiend descent algorithm as the data flows in. In that case, (6.5.20) and (6.5.22) simple need to be replaced by,

$$\mathscr{L}(\boldsymbol{\Theta}) \quad = \quad \frac{1}{t}\sum_{j=1}^{t}\beta_j f(\mathbf{x}_j, \mathbf{h}(\mathbf{x}_j, \boldsymbol{\Theta})). \tag{6.5.24}$$

where again $\beta_j \in [0,1]$ is a forgetting factor, that tend to zero as $t-j$ tends to infinity.

When the training of the regressors can be done online, one can further consider the online adaptation of the dictionary. This can be done, simply by treating **U** in (6.5.21) and **D** in (6.5.19) as optimization variables in the training, and minimizing $\mathscr{L}$ with respect to both the dictionaries and the network parameters, alternating between network training and dictionary update iterations. This essentially extends the proposed framework into a full-featured online learned encoder, that can be applied in the context of structured sparse encoders or robust PCA.

A full online scenario for training the encoders consists of (a) initializing the dictionary (b) fixing the dictionary in the training objective and adapting the network parameters to the newly arriving data using one of the wealth of online learning algorithms; and (c) fixing the robust low dimensional projections and adapting the dictionary using (6.4.15).

6.5.5 Geometric transformations

The underlying model used in robust PCA relies on the critical assumption that the given input vectors **X** are "aligned" with respect to each other. While this assumption holds for many applications (i.e., background subtraction with static cameras), it does not apply in all cases. The canonical example is face modeling, where the low dimensional model only holds if the facial images are pixel-wise aligned [83, 55]. Even small misalignments can break the structure in the data, the representation quickly degrades as the rank of the low dimensional component **US** increases and the matrix of outliers **O** loses its sparsity.

This challenging problem has been recently studied in the literature. In [55] the authors propose a pre-processing strategy to align the training images. In [83], the authors simultaneously align the input images and solve robust PCA with a sequence of convex problems.

Following [83], we propose to incorporate the optimization over geometric transformations of the input data into the representation problem. Then, the optimization problem

(6.4.10) becomes

$$\min_{\mathbf{U},\mathbf{S},\mathbf{O},\boldsymbol{\alpha}} \frac{1}{2}\left\|\mathbf{T}_{\boldsymbol{\alpha}}(\mathbf{X}) - \mathbf{U}\mathbf{S} - \mathbf{O}\right\|_F^2 + \frac{\lambda_*}{2}(\|\mathbf{U}\|_F^2 + \|\mathbf{S}\|_F^2) + \lambda\|\mathbf{O}\|_1, \tag{6.5.25}$$

where $\mathbf{T}_{\boldsymbol{\alpha}}$ is a parametrized operator (with a set of parameters collectively denoted as $\boldsymbol{\alpha} = [\boldsymbol{\alpha}^{\mathbf{1}}, \ldots, \boldsymbol{\alpha}^{\mathbf{n}}]$) that applies different geometric transformation, $\mathbf{T}_{\boldsymbol{\alpha}^{\mathbf{i}}}$, to each training vector $\mathbf{x}_i$. This formulation is highly non-convex and difficult to optimize. Interestingly, the framework of trainable regressors introduced in sections 6.5.2 and 6.5.4 is very well suited for producing accurate approximations of (6.5.25) at very mild extra computational expenses. We propose to replace the training objective function defined in (6.5.26) for

$$\mathscr{L}(\boldsymbol{\Theta}, \boldsymbol{\alpha}) \quad = \quad \frac{1}{t}\sum_{j=1}^{t} \beta_j f\left(\mathbf{T}_{\boldsymbol{\alpha}^j}(\mathbf{x}_j), \mathbf{h}(\mathbf{T}_{\boldsymbol{\alpha}^j}(\mathbf{x}_j), \boldsymbol{\Theta})\right). \tag{6.5.26}$$

The obtained regressors are conceptually very similar to the ones we had before and can still be trained in an online manner. When a new data vector $\mathbf{x}_t$ arrives, we compute it's robust low rank projection by minimizing $f(\mathbf{T}_{\boldsymbol{\alpha}^t}(\mathbf{x}_t), \mathbf{h}(\mathbf{T}_{\boldsymbol{\alpha}^t}(\mathbf{x}_t), \boldsymbol{\Theta}))$ over a the vector $\boldsymbol{\alpha}^{\mathbf{t}}$ parametrizing the geometric transformation.

This strategy can also be used in the standard robust PCA scenario, however, the representations $\mathbf{a} = \{\mathbf{o}, \mathbf{s}\}$ themselves are minimizers of a convex problem, making the minimization with respect to $\boldsymbol{\alpha}$ cumbersome and computationally expensive. On the other hand, when using a neural network encoder to produce the representation, $\mathbf{h}(\mathbf{x}_t, \boldsymbol{\Theta})$ is almost everywhere differentiable with respect to their input $\mathbf{x}_t$, which allows to find the (sub)gradient with respect to $\boldsymbol{\alpha}^{\mathbf{t}}$ by applying the chain rule. This, in turn, allows to minimize $f(\mathbf{T}_{\boldsymbol{\alpha}^t}(\mathbf{x}_t), \mathbf{h}(\mathbf{T}_{\boldsymbol{\alpha}^t}(\mathbf{x}_t), \boldsymbol{\Theta}))$ using standard (sub)gradient descent algorithms. In the same way, the transformation of all the training vectors is updated through the minimization of a loss function $\mathscr{L}(\boldsymbol{\Theta}, \boldsymbol{\alpha})$ with respect to $\boldsymbol{\alpha}$, following the ideas in Section 6.5.4.

Table 6.1: Misclassification rates on MNIST digits.

Code	**Dictionary size**	
	100	289
NN G-L	3.76%	5.98%
NN Lasso	2.65%	2.51%
Exact Lasso	1.99%	1.47%

6.6 Experimental results

All neural networks were implemented in Matlab with built-in GPU acceleration and executed on state-of-the-art Intel Xeon E5620 CPU and NVIDIA Tesla C2070 GPU. Even with this by no means optimized code, the propagation of 10^5 100-dimensional vector through a 10-layer structured network with the proposed BCoFB architecture takes only 3.6 seconds, which is equivalent to $3.6\mu sec$ spent per vector per layer. This is several orders of magnitude faster than the exceptionally optimized multithreaded SPAMS HiLasso code [50] executed on the CPU. Such benefits of parallelization are possible due to the fixed datapath and complexity of the neural network encoder compared to the iterative solver.

In all experiments, training was performed using gradient descent safeguarded by Armijo rule. We refer as *NN G-L* to the neural network sparse encoders obtained by minimizing Gregor&LeCun's objective function, this is, the ℓ_2 discrepancy with the output of the exact encoder. It will be explicitly stated when neural network sparse encoders are trained using a specific objective function (e.g., *NN Lasso*).

6.6.1 Classification

In this experiment, we evaluate the performance of unstructured neural network sparse encoders in the MNIST digit classification task. The MNIST images were resampled to 17×17

Table 6.2: Misclassification rates on the audio dataset.

Code	**Error rate**
NN G-L unstructured	6.08%
NN G-L structured	3.53%
NN discriminative structured	3.44%
Exact	2.35%

(289-dimensional) patches. A set of ten dictionaries was trained for each class. Classification was performed by encoding a test vector in each of the dictionaries and assigning the label corresponding to the smallest value of the full Lasso objective.

The following sparse coders were compared: exact sparse codes (*Exact Lasso*), unstructured *NN G-L*, and unstructured *NN Lasso* (a CoD network trained using the Lasso objective). Ten networks were trained, one per each class; all contained $T = 5$ CoD layers. $\lambda = 0.1$ was used in the Lasso objective. Dictionaries with 100 (under-complete) and 289 (complete) atoms were used. Further increase in the dictionary size did not exhibit significant performance improvement.

Table 6.1 summarizes the misclassification rates of each of the sparse encoders. Performance of the *NN G-L* sparse encoder decreases with the increase of the dictionary size, while the discrepancy with the exact codes drops. On the other hand, better performance in terms of the Lasso objective consistently correlates with better classification rates, which makes *NN Lasso* a more favorable choice. Dictionary adaptation in the training of the *NN Lasso* encoder brings only a small improvement in performance, diminishing with the dictionary size. We attribute this to the relative low complexity of the data.

6.6.2 Structured coding

In this section we evaluate the performance of the structured sparse encoders in a speaker identification task reproduced from Chapter 4. In that application we used HiLasso to automatically detect the present speakers in a given mixed signal. We repeat this experiments using the proposed efficient structured sparse encoders instead.

The dataset consists of recordings of five different radio speakers, two females and three males. One quarter of the samples was used for training, and the rest for testing. Within the testing data, two sets of waveforms were created: one containing isolated speakers, and another containing all possible combinations of mixtures of two speakers. Signals are decomposed into a set of overlapping local time frames of 512 samples with 75% overlap, such that the properties of the signal remain stable within each frame. An 80-dimensional feature vector is obtained for each audio frame as its short-time power spectrum envelope (refer to Chapter 4 for details). Five undercomplete dictionaries with 50 atoms were trained on the single speaker set minimizing the Lasso objective with $\lambda = 0.2$ (one dictionary per speaker), and then combined into a single structured dictionary containing 250 atoms. Increasing the dictionary size exhibited negligible performance benefits. Speaker identification was performed by first encoding a test vector in the structured dictionary and measuring the ℓ_2 energy of each of the five groups. Energies were sum-pooled over 500 time samples and the labels of the highest two were selected.

The following structured sparse encoders were compared: exact HiLasso codes with $\mu = 0.05$ (*Exact*), unstructured *NN G-L* trained on the exact HiLasso codes (*NN G-L unstructured*), structured *NN G-L* trained on the same codes (*NN G-L structured*), and a structured network with a discriminative cost function with regularization term in which the weights μ_r were set independently for each data vector to -1 to promote the activation of groups corresponding to knowingly active speakers, and to $+1$ to discourage the activation of groups corresponding to knowingly silent speakers (*NN discriminative structured*). All

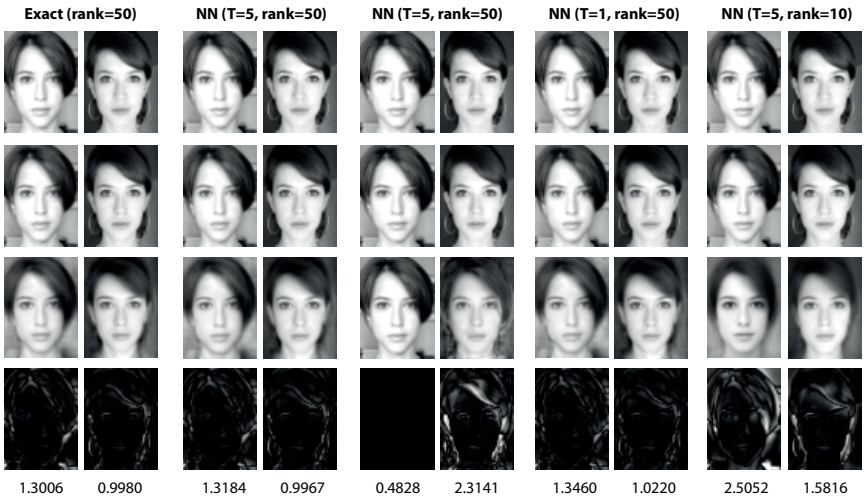


Figure 6.3: Robust PCA representation of faces from the faces sequence using different encoders (left-to-right groups): Algorithm 5; neural network encoder with five layers trained on 600 faces; the same encoder trained only on 100 faces (observe the overfitting phenomenon); a "shallow" single-layer encoder; and a five-layer encoder with lower rank q. First row: original face $\mathbf{x}$; second row: approximation $\mathbf{Us} + \mathbf{o}$; third row: low-rank approximation $\mathbf{Us}$; fourth row: sparse outliers $\mathbf{o}$. Left column contains a face from the training set; right column is a face from the test set unseen at training. The $\ell_2 + \ell_1$ cost is given for each case.

neural networks contained $T = 2$ layers with the BCoFB architecture.

Table 6.2 summarizes the obtained misclassification rates. It is remarkable that using a structured architecture instead of its unstructured counterpart with the same number of layers increases performance by nearly a factor of two. The use of the discriminative objective further improves performance. Surprisingly, using a network with only two layers cedes just about 1% of correct classification rate.

6.6.3 Robust PCA encoders

In order to evaluate the quality of different robust PCA encoders, neural networks with different number of layers were trained on 500 vectors from the faces dataset. The following training objectives were used: the ℓ_2 discrepancy between the exact representation $\mathbf{s}$ and $\mathbf{o}$

Table 6.3: Robust PCA representation accuracy (in the sense of the $\ell_2 + \ell_1$ cost) of the faces data using different encoders. The cost for the exact encoder is 1.290.

Encoder	**Untrained**	**Approx.**	$\ell_2 + \ell_1$	$\ell_2 + \ell_1$ **with U update**
Single layer	1.3355	1.3471	1.3460	1.3262
$(2, 10)$ **layers**	(1.325,1.297)	(1.326,1.298)	(1.325,1.297)	(1.317,1.288)

(denoted as *Approx.*); the $\ell_2 + \ell_1$ objective; and the $\ell_2 + \ell_1$ objective combined with the online update of the dictionary $\mathbf{U}$ (initial dictionary was computed using standard SVD). For reference, we also report the results produced by the exact Algorithm 5, and an *untrained* network with $\mathbf{W}$, $\mathbf{H}$ and $\boldsymbol{\lambda}$ initialized according to Algorithm 5 (being effectively a truncated version of the algorithm). The obtained representations are visualized in Figure 6.3. Table 6.3 summarizes the quality of the representations in terms of the $\ell_2 + \ell_1$ cost (lower numbers correspond to better quality). Note how sufficiently deep encoders with dictionary update slightly outperform the exact encoder without dictionary adaptation. Also note that using a neural network encoder to approximate the exact representations slightly degrades the $\ell_2 + \ell_1$ measure compared to the untrained encoder.

Figure 6.4 shows background and foreground separation via robust PCA on the surveillance data. The representation produced by a five-layer neural network with $q = 5$ is nearly identical to the output of the exact algorithm and to the output of the code from [64], used as reference, while being considerably faster.

6.6.4 Online learning

In this section, we evaluate the online learning capabilities of unstructured neural network sparse encoders. We performed two different experiments.

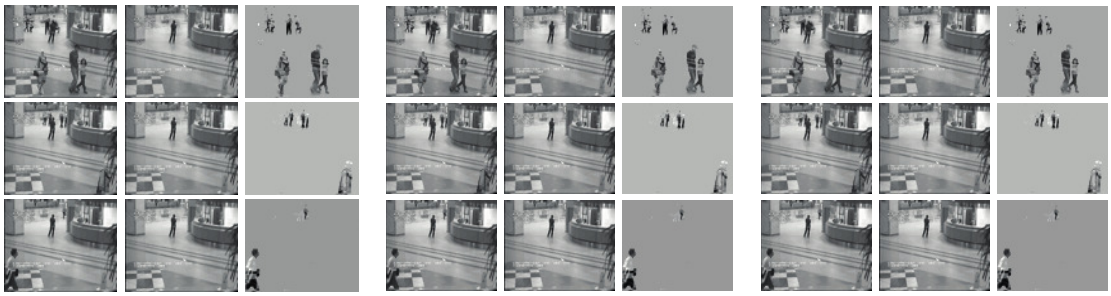


Figure 6.4: Robust PCA representation of several frames from the surveillance sequence obtained using the algorithm in [64] (left group), Algorithm 5 (middle group), and a five layer neural network encoder (right group). Columns in each group are, left-to-right: the reconstructed frame $\mathbf{Us} + \mathbf{o}$, its low-rank approximation $\mathbf{Us}$ (background), and the sparse outlier $\mathbf{o}$ (foreground). Each row corresponds to a different frame.

Natural image patches

In this experiment we evaluated the encoders using natural image patches, since this is one of the most successful applications of dictionary learning in computer vision. As the input data we used 30×10^4 randomly located 8×8 patches from three images from the Brodatz texture dataset [90]. The patches were ordered in three consecutive blocks of 10^4 patches from each image. Dictionary size was fixed to 64 atoms. $\lambda = 1$ was used in the Lasso objective.

Online learning was performed in overlapping windows of $1,000$ vectors with a step of 100 vectors. We compared standard online dictionary learning (*Exact Lasso online*) with unstructured *NN Lasso* with dictionary adaptation in a given window (*NN Lasso online*), initialized by the network parameters from the previous windows. In the latter case, the dictionary was initialized by a random subset of 64 out of the first $1,000$ data vectors (therefore, no iterative sparse coding). As the reference, we also compared the following three offline algorithms trained on a distinct set of $6,000$ patches extracted from the same images: standard dictionary learning (*Exact Lasso offline*); unstructured *NN G-L* (*NN G-L offline*), and unstructured *NN Lasso* (*NN Lasso offline*). All neural networks used $T = 4$ CoD layers.

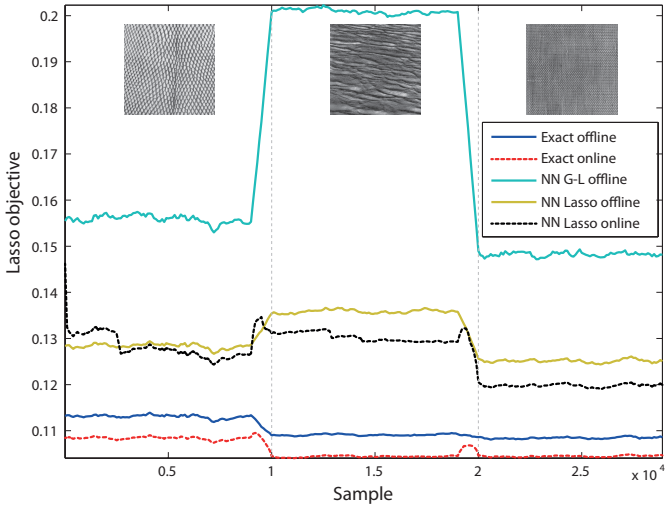


Figure 6.5: Performance of different sparse encoders measured using the Lasso objective as the function of sample number in the online learning experiment. Shown are the three groups of patches corresponding to different texture images from the Brodatz dataset.

Performance measured in terms of the Lasso objective is reported in Figure 6.6. Exact offline sparse encoder achieved the best results among all offline encoders. It is, however, outperformed by the exact online encoder due to its ability to adapt the dictionary to a specific class of data. The performance of the *NN Lasso online* encoder is slightly inferior to the *Exact Lasso offline*; the online version performs better after the network parameters and the dictionary adapt to the current class of data. Finally, the *NN G-L offline* encoder has the lowest, significantly inferior performance.

This experiment shows that, while the drop in performance compared to the exact encoder is relatively low, the computational complexity of the *NN Lasso online* encoder is tremendously lower and is fixed. This allows using sparse modeling in real-time online learning scenarios, which has been so far prohibitive with sparse coding involving optimization.

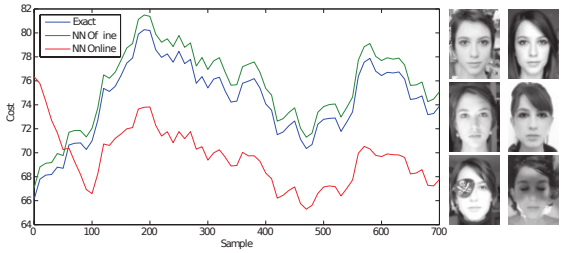


Figure 6.6: Performance, in the sense of the cost (6.4.8), of the online and offline encoders on the faces sequence. Representative faces are also shown.

Face modeling

In this experiment we evaluate the online learning capabilities of the proposed neural network encoders. As the input data we used the time-ordered sequence of 800 images from the faces dataset. Online learning was performed in overlapping windows of 100 images with a step of 10 images. We compared the exact algorithm, a five layer neural network encoder trained offline using the $\ell_2 + \ell_1$ objective (*NN offline*), and the same encoder trained online with adaptive $\mathbf{U}$. The dictionary was initialized using SVD. Performance measured in terms of the exact cost (6.4.8) is reported in Figure 6.6. The exact offline encoder is consistently slightly inferior to the exact algorithm. However, thanks to its capabilities to adapt to the changing data distribution, again here the online encoder starts outperforming the offline counterparts after a relatively brief period of initial adaptation.

6.6.5 Geometric transformations

We now evaluate the representation capabilities of the proposed neural network encoder in the presence of geometric transformations. A five layer encoder was trained on 600 images from the faces dataset. As the test set, we used the remaining 200 faces, as well as a collection of geometrically transformed images from the same test set. Sub-pixel planar translations were used for geometric transformations. The encoder was applied to the misaligned set, optimizing the $\ell_2 + \ell_1$ objective over the transformation parameters. For

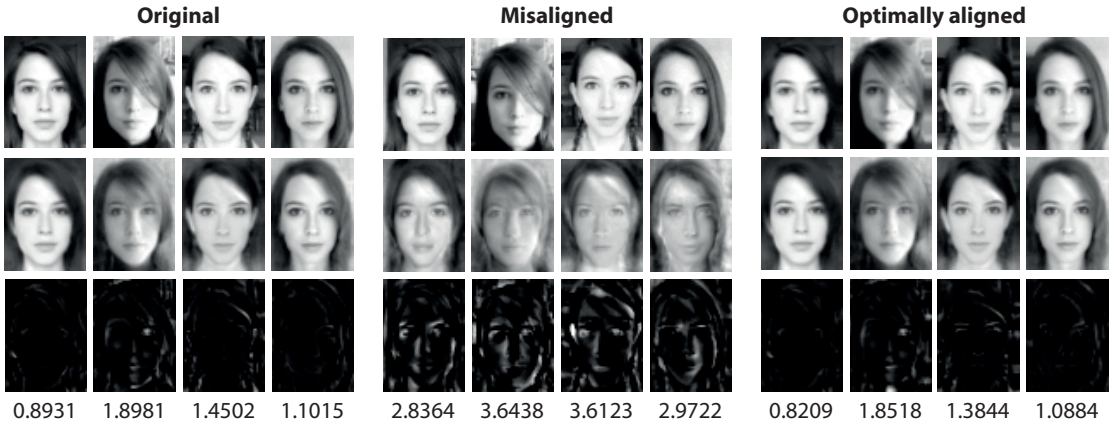


Figure 6.7: Robust PCA representation of the faces dataset in the presence of geometric transformations (misalignment). Left group: original faces; middle group: shifted faces; right group: faces optimally realigned during encoding. First row: reconstructed face $\mathbf{Us}+\mathbf{o}$; middle row: its low-rank approximation $\mathbf{Us}$; and bottom row: sparse outlier $\mathbf{o}$. The $\ell_2+\ell_1$ cost is given for each representation.

reference, the encoder was also applied to the transformed and the untransformed test sets without performing optimization. Examples of the obtained representations are visualized in Figure 6.7. Note the relatively larger magnitude and the bigger active set of the sparse outlier vector $\mathbf{o}$ produced for the misaligned faces, and how they are re-aligned when optimiziation over the transformation is allowed. Since the original data are only approximately aligned, performing optimal alignment during encoding frequently yields lower cost compared to the plain encoding of the original data.

6.7 Conclusion

Marrying ideas from convex optimization with multi-layer neural networks, we have developed in this work a comprehensive framework for modern sparse modeling and low rank recovery for real time and large scale applications. The framework includes different objective functions, from reconstruction to classification, allows different sparse coding structures from hierarchical to group similarity, and robust principal component analysis. The

framework allow an easy inclusion of the estimation of geometric transformation applied to training data as well as addressing online learning scenarios. A simple implementation already achieves several order of magnitude speedups when compared to the state-of-the-art, at minimal cost in performance, opening the door for practical algorithms following the demonstrated success of these modeling schemes in various applications.

7 Real-time source separation via trainable encoders

7.1 Chapter summary

In this chapter we evaluate a variation of the encoders presented in Chapter 6 in the context of monaural source separation. Our method builds upon recent works in the field of singing-voice/music-accompaniment separation using RPCA producing very successful results. The method decomposes the signal into a low-rank component corresponding to the accompaniment with its repetitive structure, and a sparse component corresponding to the voice with its quasi-harmonic structure.

In this paper we first introduce a way of combining NMF with low rank estimation. Then we use this regularization to propose two different low rank models. First a robustified version of NMF, that can also be thought as a non-negative variant of RPCA, termed as robust low-rank non-negative matrix factorization (RNMF). This new framework better suits audio applications. Then we propose a new sparse, low rank NMF that is very well suited for representing speech signals. In both cases propose efficient feed-forward architectures that approximate the exact solutions for these models with low latency and a fraction of the complexity of the original optimization method. These approximants allow incorporating elements of unsupervised, semi- and fully-supervised learning into the RPCA and RNMF frameworks. Our basic implementation shows several orders of magnitude speedup

compared to the exact solvers with no performance degradation, and allows online and faster-than-real-time processing. In the problem of singing-voice/music-accompaniment, an evaluation on the MIR-1K dataset demonstrates state-of-the-art performance.

7.2 Introduction

Non-negative matrix factorization (NMF) [58] was shown efficient for source separation leading to very good results [32, 97]. The separation via NMF is carried out by decomposing the time-frequency representation of the mixture signal and then performing reconstructions of groups corresponding to each single source. NMF has been shown to be better suited for separation tasks than other matrix decompositions due to (at least) two major reasons: First, the non-negativity constraints are well suited to the non-negative spectral representations since it produces a non-subtractive, part-based representation. Second, in the time-frequency domain, even non-stationary signals such as speech can be modeled accurately with-low rank models. However, the optimal rank of the representation is rarely known *a priori*, and there is a trade-off between generalization and accuracy: the more atoms the dictionary contains, the better is the signal reconstruction, but also the more degrees of freedom it give (which are undesirable in ill-posed problems like source separation). Consequently, the results normally depend on the number of atoms used in the decomposition and degrade quickly outside some (relatively narrow) sweet spot. In general, the selection of the size of the dictionary is done based on empirical observations or via cross-validation. Several regularization techniques of the NMF have been proposed to mitigate this problem and to improve the robustness of the separation.

We begin by introducing a regularized version of NMF in which high-rank representations are penalized. The regularization is related to the nuclear norm of the reconstructed spectrogram and shows robustness with respect to the size of the dictionaries. This new

regularization is used to build two new models for audio signal representation.

We first develop an extension of RPCA in which the low rank model is represented as a non-negative linear combination of non-negative basis vectors. This is done following recent results connecting non-convex optimization with nuclear norm optimization [93, 94] (further references are given later in the chapter). As with standard non-negative matrix factorization (NMF) methods, this new model is more appropriate to represent audio signals, being applied to the magnitude of the spectrum. The use of robust NMF (RNMF) is not restricted to this application and the usage in combination with divergences in lieu of Euclidean distances is straightforward. The proposed framework can also be seen as an extension of the robustification of NMF introduced in [130]; not only does our model consider a sparse variable accounting for outliers , but it also adds a regularization term that minimizes the rank of the linear model. Following the recent work by Huang and coauthors [44], we evaluate the proposed model in the challenging problem of separating the leading singing voice from the musical background from a monaural recording.

Secondly we combine the low rank NMF with sparse models. This setting is very well suited for representing speech signals. This new model shows to be highly discriminative, as shown by our experiments in the problem of speaker identification in environments heavily contaminated by non-stationary noise.

The main contribution of our work is to combine these ideas to work with tailored feed-forward neural networks opening up new uses as explained in the sequel. We induce multi-layer feed-forward networks capable of approximating the output of the exact optimization algorithms at a fraction of their computational cost and with no decrease in performance in our various experiments.

This new framework allows to incorporate unsupervised, semi- and fully-supervised learning into RPCA and RNMF. In this way, we aim at taking the advantages of the unsupervised methods while minimizing their drawbacks via realistic learning. When combined

with learning as here proposed, the obtained networks produce over 1 dB improvement in the signal-to-distortion ratio when compared to the optimization-based RPCA, and, after the offline learning, are computable online and faster than real time without the need to observe the whole audio file.

While capturing the great advantages of the NMF paradigm, the differentiability of the proposed encoders with respect to the input, output and the parameters allows their use in a much broader scenario. As an example, we define a discriminative training objective function that improves the (already very good) discriminative power of the proposed NMF model.

The proposed networks are closely related to the ones introduced in [39], used to produce meaningful audio features for music style and gender classification [40]. These approaches are examples of recent successful efforts in the machine learning community to produce fast trainable (auto-)encoders of sparse features of visual and audio signals (see [37, 91] and references therein). While the work in this paper comes from these ideas, it presents a fundamental difference in the sense that the proposed networks do not compute features, but perform the full separation of the singing voice from the musical accompaniment.

The chapter is organized in the following way. In Section 7.3 we introduce the proposed low rank NMF models. In Section 7.5 we discuss their implementation via the new feed-forward networks and their different ways of usage. A description of the singing-voice/music-accompaniment and speech denoising problems along with experimental results are given in Section 7.6 and Section 7.7 respectively. Finally, in Section 7.8 we draw conclusions and discuss future and ongoing work.

7.3 Non-negative matrix factorization

Given a non-negative matrix $\mathbf{X} \in \mathbb{R}^{F\times N}$, NMF aims at finding a factorization $\mathbf{X} \approx \mathbf{US}$ into non-negative matrices $\mathbf{U} \geq 0 \in \mathbb{R}^{F\times Q}$, $\mathbf{S} \geq 0 \in \mathbb{R}^{Q\times N}$. The dictionary size Q is generally chosen such that $FQ + QN \ll FN$. In audio processing, $\mathbf{X}$ is a non-negative representation of the time-frequency domain, obtained using the short-time Fourier transform (STFT) with F frequency samples and N time frames. In this case, $\mathbf{U}$ is a dictionary and each column represents an elementary spectral atom and $\mathbf{S}$ codes the activation of each atom in the dictionary throughout the frames; $\mathbf{US}$ is a low-rank linear approximation of $\mathbf{X}$.

The factorization is in general obtained by solving a minimization problem of the form,

$$\min_{\mathbf{U}\geq\mathbf{0},\mathbf{S}\geq\mathbf{0}} \sum_{i=1}^{N} d(\mathbf{x}_i|[\mathbf{US}]_i), \tag{7.3.1}$$

where d is a scalar cost function. Significant attention has been devoted in the literature to the case in which d belongs to the family of the so-called β-divergences defined as

$$d_\beta(\mathbf{a}|\mathbf{b}) = \sum_{j=1}^{F} \begin{cases} \frac{a_i}{b_i} - \log\frac{a_i}{b_i} - 1 & \text{if} \quad \beta = 0 \\ a_i \log a_i/b_i + (a_i - b_i) & \text{if} \quad \beta = 1 \\ \frac{1}{\beta(\beta-1)}(a_i^\beta + (\beta-1)b_i^\beta - \beta a_i b_i^{\beta-1}) & \text{otherwise.} \end{cases}$$

This family includes the three most widely used cost functions in NMF: the Euclidean distance ($\beta = 2$), the Kullback-Leibler divergence ($\beta = 1$), and the Itakura-Saito divergence ($\beta = 0$).

One of the most popular ways of solving (7.3.1) is a multiplicative gradient descent approach introduced in [58] that alternates a descent over $\mathbf{U}$ and $\mathbf{S}$,

$$\mathbf{U} \leftarrow \mathbf{U}.\frac{\mathbf{U}^\mathrm{T}((\mathbf{US})^{.(\beta-2)}.\mathbf{X})}{\mathbf{U}^\mathrm{T}(\mathbf{US})^{.(\beta-2)}}, \quad \mathbf{S} \leftarrow \mathbf{S}.\frac{((\mathbf{US})^{.(\beta-2)}.\mathbf{X})\mathbf{S}^\mathrm{T}}{(\mathbf{US})^{.(\beta-2)}\mathbf{S}^\mathrm{T}}, \tag{7.3.2}$$

where . denotes element-wise operations.

7.3.1 Online NMF

Uhen the entire audio signal is not available, one can think of the columns of $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_N)$ as arriving sequentially. Online NMF algorithms aim at computing the factorization as the data comes in. In general, they compute (or approximate) the projections of the incoming vectors $\mathbf{x}_t$ onto the current estimate of the dictionary $\mathbf{U}$, and then use the projections to refine $\mathbf{U}$. When the dictionary is available *a priori*, as explained in the sequel, then projection of the $\mathbf{x}_t$ is obtained by solving the optimization problem

$$\min_{\mathbf{s}\geq\mathbf{0}} d_\beta(\mathbf{x}_t|\mathbf{U}\mathbf{s}). \tag{7.3.3}$$

For a fixed dictionary, problem (7.3.1) is separable in the columns, and each subproblem corresponds exactly to (7.3.3). In many situations, in order to save computational time, (7.3.3) is not solved exactly, as discussed by [59]. [23] set the maximum number of iterations of the multiplicative algorithm to a value that empirically provides a good approximation. The update of the dictionary is performed by incorporating the information of the new projections [59].

7.4 Robust low-rank models for time frequency representations

In this section we introduce a low-rank regularized version of NMF. We further use it in two important cases, namely: robust NMF and low-rank sparse NMF. Through this section, we are limiting our attention to the Euclidean data term, as it lends itself easily to solution via proximal splitting methods (or alternating minimization schemes) that will motivate the construction of our fast NMF encoders. We will however reintroduce arbitrary data terms at the training stage as described in the sequel.

7.4.1 Low-rank NMF

Recent advances in convex optimization have shown that to substitute by its nuclear norm $\|\mathbf{US}\|_*$, defined as the sum of the singular values of the matrix, which is the tightest convex surrogate for the rank. Akin the ℓ_1-norm that encourages sparsity of vectors, the nuclear norm promotes low rank of matrices. It has been successfully used for finding low rank models in various contexts. The most significant one is probably the robust principal component analysis introduced in [12], see Section 2.3 for a brief overview. However, this regularizer is not well-suited for the alternating minimization involved in NMF, as it couples both the dictionary $\mathbf{U}$ and the activation $\mathbf{S}$. As done in Chapter 6 in the context of RPCA, we are going to use the recent result from [93] to decouple the regularizer. This is valid when one has a rough upper bound Q on the rank of the representation. As stated in (7.4.4), the nuclear norm of a matrix $\mathbf{L} \in \mathbb{R}^{F \times N}$ with $\text{rank}(\mathbf{L}) \leq Q$ can be reformulated as a penalty over al possible factorizations,

$$\|\mathbf{L}\|_* = \min_{\mathbf{U},\mathbf{S}} \frac{1}{2} \|\mathbf{U}\|_{\text{F}}^2 + \frac{1}{2} \|\mathbf{S}\|_{\text{F}}^2 \quad \text{s.t.} \quad \mathbf{US} = \mathbf{L}, \tag{7.4.4}$$

with $\mathbf{U} \in \mathbb{R}^{F \times Q}$, $\mathbf{S} \in \mathbb{R}^{Q \times N}$. The optimum is found via the SVD of $\mathbf{L}$. Now, given $\mathbf{L} = \mathbf{US}$ as a product of non-negative matrices $\mathbf{U}$ and $\mathbf{S}$, the optimum in (7.4.4) would be attained by a pair of not necessarily non-negative factors, $\hat{\mathbf{U}}$, $\hat{\mathbf{S}}$. Consequently, adding the non-negativity constraint to (7.4.4) produces

$$||\mathbf{US}||_* \leq \frac{1}{2} \min_{\mathbf{S} \geq \mathbf{0}, \mathbf{U} \geq \mathbf{0}} \left\{ ||\mathbf{U}||_{\text{F}}^2 + ||\mathbf{S}||_{\text{F}}^2 \text{ s.t. } \mathbf{US} = \mathbf{US} \right\} \leq \frac{1}{2} \|\mathbf{U}\|_{\text{F}}^2 + \frac{1}{2} \|\mathbf{S}\|_{\text{F}}^2 \tag{7.4.5}$$

Thus, the sum of the Frobenius norms of the non-negative matrices $\mathbf{U}$ and $\mathbf{S}$ gives an upper bound on the nuclear norm of their product, and motivates the proposed regularizer.

7.4.2 Robust NMF

We now extend (6.4.10) to consider a robust version of NMF. We to the low rank term an outlier matrix and constrain them to be non-negative,

$$\min_{\mathbf{U}\geq\mathbf{0},\mathbf{S}\geq\mathbf{0},\mathbf{O}\geq\mathbf{0}} \frac{1}{2}\left\|\mathbf{X}-\mathbf{U}\mathbf{S}-\mathbf{O}\right\|_F^2+\frac{\lambda_*}{2}(\left\|\mathbf{U}\right\|_F^2+\left\|\mathbf{S}\right\|_F^2)+\lambda\left\|\mathbf{O}\right\|_1. \tag{7.4.6}$$

This new formulation is no longer equivalent to the classical RPCA given in (6.4.10). In fact, applying (7.4.4) directly to the matrix $\mathbf{US}$, we obtain $\hat{\mathbf{U}}\hat{\mathbf{S}}$ with the factors $\hat{\mathbf{S}}$ and $\hat{\mathbf{U}}$ not being necessarily non-negative. Adding the non-negativity constraint produces the inequality

$$||\mathbf{US}||_*\leq\frac{1}{2}\min_{\hat{\mathbf{S}}\geq\mathbf{0},\hat{\mathbf{U}}\geq\mathbf{0}}||\hat{\mathbf{U}}||_{\mathrm{F}}^2+||\hat{\mathbf{S}}||_{\mathrm{F}}^2. \tag{7.4.7}$$

Thus, the sum of the Frobenius norms of the non-negative matrices $\mathbf{S}$ and $\mathbf{U}$ regularizes an upper bound of the nuclear norm of their product.

Standard NMF is obtained as a particular case by setting to zero both λ_* and λ, while the robust version of NMF introduced in [130] is obtained when only λ_* is selected as zero. In this work we use RNMF as stated in (7.4.6), however its extension to more general fitting terms such as β-divergences is straightforward. Problem (7.4.6) can be optimized using multiplicative algorithms, commonly used in the NMF context.

Robust non-negative projections

Let us now assume to be given a low dimensional model, $\mathbf{U}\in\mathbb{R}^{m\times q}$, learned from some data $\mathbf{X}\approx\mathbf{US}+\mathbf{O}\in\mathbb{R}^{m\times n}$. A new input vector $\mathbf{x}$ drawn from the same distribution as $\mathbf{X}$ can be decomposed into $\mathbf{x}=\mathbf{Us}+\mathbf{n}+\mathbf{o}$, where $\mathbf{Us}$ represents the low dimensional component,

n is a small perturbation, and **o** is a sparse outlier vector. It can be obtained via

$$\min_{\mathbf{s}\geq\mathbf{0},\mathbf{o}\geq\mathbf{0}} \quad \tfrac{1}{2}\left\|\mathbf{x}-\mathbf{U}\mathbf{s}-\mathbf{o}\right\|_2^2+\tfrac{\lambda_*}{2}\left\|\mathbf{s}\right\|_2^2+\lambda\left\|\mathbf{o}\right\|_1, \tag{7.4.8}$$

a convex problem similar to the one of standard sparse coding. The solution can be obtained via proximal methods [2], which split the objective function (7.4.8) into a smooth part (the first two terms), and a non-differentiable part (the ℓ_1 norm of the outliers vector). Proximal methods iterate between a gradient descent on the smooth function and an application of the proximal operator (which assumes a closed form of one-sided soft-thresholding), as detailed in Algorithm 6. This algorithm is conceptually very similar to the popular iterative shrinkage and thresholding algorithm (ISTA) [4]. We do not use this algorithm as an explicit tool, but rather as a motivation of the architecture of a feed-forward network capable of accurately performing the separation in real time, as discussed next.

Algorithm 6: RNMF given the dictionary **U**.

input : Data **x**, dictionary **U**.
output: Nonnegative coefficient vector **s** and nonnegative outlier vector **o**.
Define $\mathbf{H}=\mathbf{I}-\frac{1}{\alpha}\begin{pmatrix}\mathbf{U}^{\mathrm{T}}\mathbf{U}+\lambda_*\mathbf{I} & \mathbf{U}^{\mathrm{T}}\\ \mathbf{U} & (1+\lambda_*)\mathbf{I}\end{pmatrix}$, $\mathbf{W}=\frac{1}{\alpha}\begin{pmatrix}\mathbf{U}^{\mathrm{T}}\\ \mathbf{I}\end{pmatrix}$, and $\mathbf{t}=\frac{\lambda}{\alpha}\begin{pmatrix}\mathbf{0}\\ \mathbf{1}\end{pmatrix}$.
Initialize $\mathbf{z}=\mathbf{0}$, $\mathbf{b}=\mathbf{W}\mathbf{x}$.
repeat
  $\mathbf{y}=\max\{\mathbf{b}-\mathbf{t},0\}$
  $\mathbf{b}=\mathbf{b}+\mathbf{H}(\mathbf{y}-\mathbf{z})$
  $\mathbf{z}=\mathbf{y}$
until *convergence* ;
Output $(\mathbf{o},\mathbf{s})=\mathbf{z}$.

7.4.3 Sparse, low-rank models

In this section we present a new model time-frequency representations that that builds upon the low rank regularization introduced in Section 7.4.1. In particular, this model is very well suited for representing speech signals, as it will be further discussed in Section 7.7. It is

well established that speech signals can be accurately represented by a low rank model. For example, clean speech can be reconstructed with perceptually good quality with as few as 20 dictionary atoms as reported by [23]. However, the size of the dictionary that gives the best reconstruction results might vary with the application (see, e.g., the speech separation example in the supplementary material). The main reason for this behavior lies in the fact that having more dictionary atoms improves the approximation of the training data but also allows the dictionary tuned to a specific speaker to represent well other sources present in the mixture. Some recent works have shown that adding a sparsity constraint in $\mathbf{S}$ can improve this effect and provide better separation.

Specifically, we propose to model the time-frequency representations of the speech signals as a sparse linear combination of the atoms of an undercomplete dictionary by solving

$$\min_{\mathbf{U},\mathbf{S}} \frac{1}{2}\|\mathbf{X}-\mathbf{U}\mathbf{S}\|_{\mathrm{F}}^2 + \frac{\lambda_*}{2}(\|\mathbf{U}\|_{\mathrm{F}}^2 + \|\mathbf{S}\|_{\mathrm{F}}^2) + \lambda\,\|\mathbf{S}\|_1 \quad \text{s.t.} \quad \mathbf{U},\mathbf{S} \geq 0, \tag{7.4.9}$$

where λ_* and λ are parameters controlling, respectively, the low-rank and the sparsity constraints. A multiplicative gradient descent algorithm can also be used to solve (7.4.9).

Low-rank sparse NMF via convex optimization

As in the case of RNMF discussed in Section 7.4.2, we note that the proposed low-rank NMF problem (7.7.12) with fixed dictionaries can be solved using a first order convex optimization algorithm. The solution is obtained via proximal methods [2], which split the objective function (7.7.12) into a smooth part (the fitting and the low-rank terms), and a non-differentiable part (the ℓ_1 norm of the activation vector). Proximal methods iterate between a gradient descent on the smooth function and an application of the proximal operator (which assumes a closed form of one-sided soft-thresholding), as detailed in Algorithm 7. This algorithm is conceptually very similar to the popular iterative shrinkage and

Algorithm 7: Low-rank NMF given the noise and speech dictionaries $\mathbf{U}_\mathrm{n}$ and $\mathbf{U}_\mathrm{s}$.

input : Data $\mathbf{x}$, dictionary $\mathbf{U} = (\mathbf{U}_\mathrm{s}, \mathbf{U}_\mathrm{n})$.
output: Nonnegative coefficient vector $\mathbf{s} = (\mathbf{s}_\mathrm{s}; \mathbf{s}_\mathrm{n})$.
Define $\mathbf{B} = \mathbf{I} - \frac{1}{\alpha}\begin{pmatrix} \mathbf{U}_\mathrm{s}^\mathrm{T}\mathbf{U}_\mathrm{s} + \lambda_* \mathbf{I} & \mathbf{U}_\mathrm{s}^\mathrm{T}\mathbf{U}_\mathrm{n} \\ \mathbf{U}_\mathrm{n}^\mathrm{T}\mathbf{U}_\mathrm{s} & \mathbf{U}_\mathrm{n}^\mathrm{T}\mathbf{U}_\mathrm{n} + \lambda_* \mathbf{I} \end{pmatrix}$, $\mathbf{A} = \frac{1}{\alpha}\begin{pmatrix} \mathbf{U}_\mathrm{s}^\mathrm{T} \\ \mathbf{U}_\mathrm{n}^\mathrm{T} \end{pmatrix}$, and $\mathbf{t} = \frac{\lambda}{\alpha}\begin{pmatrix} \mathbf{1} \\ \mathbf{0} \end{pmatrix}$.
Initialize $\mathbf{s} = \mathbf{0}$, $\mathbf{b} = \mathbf{A}\mathbf{x}$.
repeat
  $\mathbf{y} = \max\{\mathbf{b} - \mathbf{t}, 0\}$
  $\mathbf{b} = \mathbf{b} + \mathbf{B}(\mathbf{y} - \mathbf{s})$
  $\mathbf{s} = \mathbf{y}$
until *convergence* ;
Output $(\mathbf{s}_\mathrm{s}, \mathbf{s}_\mathrm{n}) = \mathbf{z}$.

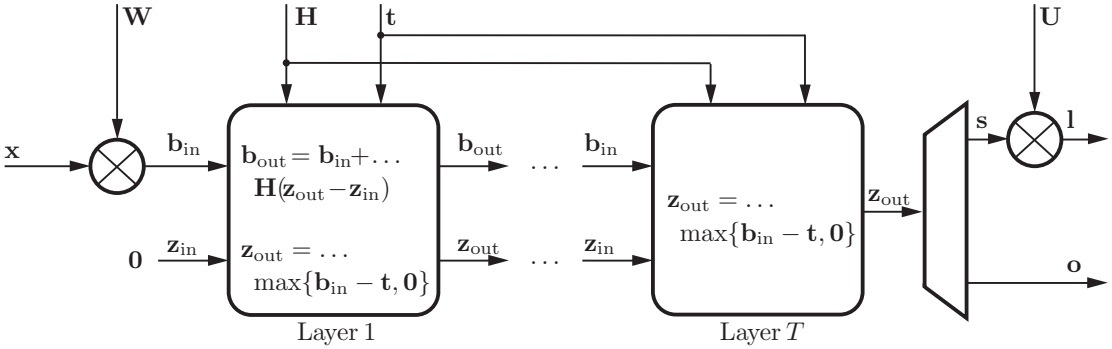


Figure 7.1: RNMF encoder architecture with T layers.

thresholding algorithm (ISTA) [4]. We do not use this algorithm as an explicit tool, but rather as a motivation of the architecture of a feed-forward network capable of accurately performing the separation in real time, as discussed next.

7.5 Fast robust sparse modeling

We used encoders as defined in Chapter 6 to efficiently solve both the RNMF and the sparse, low-rank NMF problems. We aim at constructing a parametric regressor $\mathbf{z} = (\mathbf{o}, \mathbf{s}) = \mathbf{h}(\mathbf{x}, \boldsymbol{\Theta})$, with some set of parameters, collectively denoted as $\boldsymbol{\Theta}$, capable of accurately performing the different source separation problems. This is, for a given training sample

$\mathscr{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$. Here, each $\mathbf{x}_i$ represents the magnitude spectrum of a mixture; training samples may come from many different sources (singers and songs, speech and noise, etc).

As in [39], we design an architecture for the encoders based on an exact optimization algorithm, in this case Algorithm 6 and Algorithm 7. We propose multi-layer artificial neural networks where each layer implements a single iteration of these algorithms. These networks are depicted in Figure 7.1 and Figure 7.2 where the different parameters of each networks are described.[1] On both cases, the encoder architectures are continuous and almost everywhere $\mathscr{C}^1$ with respect to the parameters, allowing the use of (sub)gradient descent methods for training. Again here, we train this regressors as explained in Section 6.5 using stochastic gradient descent.

We train the encoders by minimizing over $\mathscr{X}$ functions of the form

$$\mathscr{L}(\mathbf{\Theta}) \quad = \quad \frac{1}{|\mathscr{X}|} \sum_{x_i \in \mathscr{X}} L(\mathbf{\Theta}, \mathbf{x}_i), \tag{7.5.10}$$

where $L(\mathbf{\Theta}, \mathbf{x}_i)$ is a function that measures the quality of the code $\mathbf{z}_i = \mathbf{h}(\mathbf{x}_i, \mathbf{\Theta})$. Specifically, we iteratively select a random subset of $\mathscr{X}$ and then update the network parameters as $\mathbf{\Theta} \leftarrow \mathbf{\Theta} - \mu \frac{\partial \mathscr{L}(\mathbf{\Theta})}{\partial \mathbf{\Theta}}$, where μ is a decaying step, repeating the process until convergence. The decoder is just a linear operator given by a dictionary $\mathbf{U}$, see Figure 7.1.

Once trained, the parameters $\mathbf{\Theta}$ and the dictionary $\mathbf{U}$ are fixed, and the network is used to sequentially process new data. The latency of both RNMF and the sparse, low-rank NMF networks (referred henceforth as *NN RNMF* and *NN SLRNMF*, respectively) is of the order of a single STFT frame (hundreds of milliseconds), while the exact algorithms require the entire signal to be observed.

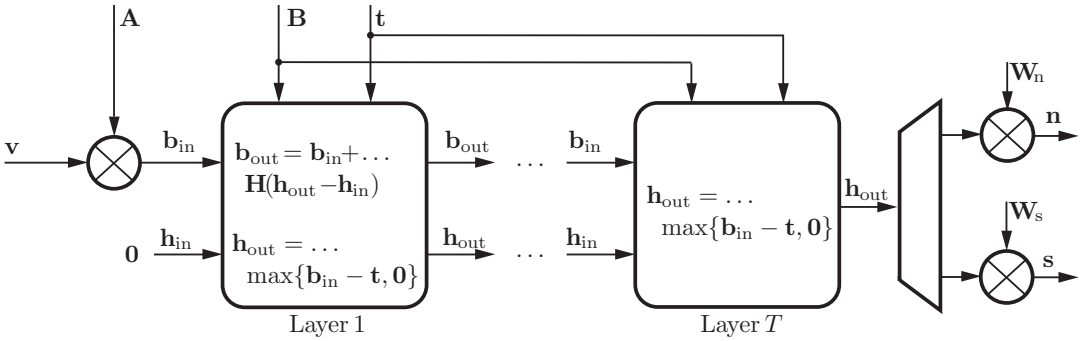


Figure 7.2: NN SLRNMF encoder architecture with T layers. The input noisy speech signal $\mathbf{v}$ is used to encode the representation coefficients $\mathbf{h}$, which are decoded into a clean speech signal $\mathbf{s}$ and noise $\mathbf{n}$.

7.5.1 Training regimes

Training of the proposed sparse, low rank NMF and RNMF encoders is possible under different regimes. We refer as *supervised* to the setting where the training set consists of the mixed signal $\mathbf{x}_i = \mathbf{o}_i^* + \mathbf{l}_i^*$, and the synchronized ground truth signals $\mathbf{o}_i^*$ and $\mathbf{l}_i^*$ (each vector corresponding to the magnitude spectrogram).

We refer as *semi-supervised* to the setting in which isolated samples of the different sources are available, but are not synchronized (the $\mathbf{x}_i$ are now either the voice or the accompaniment; speech or the noise; and so on). The training of the network is performed in the same way as the supervised case, but setting to zero the missing source.

Finally, in the *unsupervised* setting we only have access to mixtures as training data and the objective $L(\mathbf{\Theta}, \mathbf{x}_i)$ is used to directly minimize the cost in (7.4.6) and (7.4.9) respectively.

7.5.2 Dictionary adaptation

The performance of both these networks can be further improved if the dictionary $\mathbf{U}$ (decoder) is updated during the training. In both networks *NN RNMF* and *NN SLRNMF*, the

[1] In the network, extra flexibility is obtained by learning different thresholds t_i for each component.

dictionaries are updated via the standard multiplicative update,

$$\mathbf{U} \quad \leftarrow \quad \mathbf{U} \odot \frac{\mathbf{X}\mathbf{S}^{\mathrm{T}}}{\mathbf{U}(\mathbf{S}\mathbf{S}^{\mathrm{T}} + \lambda_{*}\mathbf{I})}, \tag{7.5.11}$$

where $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ is the input matrix, $\mathbf{S} = (\mathbf{s}_1, \ldots, \mathbf{s}_n)$ is the matrix of the corresponding codes, $\mathbf{Y} = (\mathbf{x}_1 - \mathbf{o}_1, \ldots, \mathbf{x}_n - \mathbf{o}_n)$, and $\odot$ and the fraction denote, respectively, element-wise multiplication and division. This update minimizes the objectives in (7.4.6) and (7.4.9) while fixing the remaining parameters, and is guaranteed to preserve the non-negativity of $\mathbf{U}$.

Analogously, in the semi- and fully-supervised scenarios, $\mathbf{U}$ can be updated by minimizing the corresponding $\mathscr{L}(\boldsymbol{\Theta})$ using the ground-truth separation.

7.6 Singing voice separation via robust NMF

7.6.1 Proposed method

Separating the leading singing voice from the musical background from a monaural recording is very challenging. Existing approaches can be classified according to the level of supervision that they require. Supervised approaches tend to have a model for either the musical background, the singing voice, or both, and in general map the mixture signals onto a feature space where the separation is performed, e.g. [79, 61, 114, 26]. A common drawback of these methods is the need to identify the vocal segments beforehand, typically using features such as the Mel-Frequency Cepstrum Coefficients (MFCC). Unsupervised approaches make basic fundamental assumptions requiring no prior training or particular features. For example, in [65] the authors tackle the separation by extracting the repeating background (music) from the non-repeating foreground (voice). Most relevant for our work is the method proposed in [44]. The authors model the repetitive structure

of the accompaniment with a low-rank linear model, while the singing voice is regarded as sparse and non-repetitive. The separation is performed using *robust* PCA (RPCA) [12], producing state-of-the-art results. Common drawbacks of unsupervised approaches include the requirement to observe the whole audio track to perform the separation and the fact that, unlike supervised models, the obtained sources might not follow known characteristics of the signals.

We consider the promising results presented in [44] as a starting point. Here we also model the background accompaniment as the low rank model and the voice as the outliers, but we use the proposed trainable encoders described in Section 7.5. Specifically, we use the *NN RNMF* encoders. As shown in the next section, we evaluate different training regimes. In the supervised setting, the objective function is

$$L(\boldsymbol{\Theta}, \mathbf{x}_i) = ||\mathbf{U}\mathbf{s}_i - \mathbf{l}_i^*||_2^2 + ||\mathbf{o}_i - \mathbf{o}_i^*||_2^2 \quad \text{with} \quad (\mathbf{o}_i, \mathbf{s}_i) = \mathbf{h}(\mathbf{x}_i, \boldsymbol{\Theta}).$$

7.6.2 Experimental evaluation

Dataset

We evaluate the separation performance of the proposed methods on the MIR-1K dataset [42], containing 1000 16 kHz clips extracted from 110 Chinese karaoke songs performed by 19 amateur singers (11 males and 8 females). Each clip duration ranges from 4 to 13 seconds, totaling about 133 minutes. We reserved about 23 minutes of audio sang by one male and one female singers (*abjones* and *amy*) for the purpose of training; the remaining 110 minutes of 17 singers were used for testing. The voice and the music tracks were mixed linearly with equal energy.

Table 7.1: Performance on the recovered vocal track on MIR-1K.

Method	**GNSDR**	**GSNR**	**GSAR**	**GSIR**
Ideal freq. mask	13.48	5.46	13.65	31.22
ADMoM RPCA [44]	5.00	2.38	6.68	13.76
Proximal RPCA	5.48	3.29	7.02	13.91
NN RPCA Untrained	5.30	2.66	6.80	13.00
NN RPCA Unsupervised	5.62	2.87	6.90	14.02
NN RPCA Supervised	6.38	3.18	7.22	16.47
NN RPCA Dict. update	6.42	3.19	7.23	16.57
Multiplicative RNMF	5.60	3.39	6.94	14.67
NN RNMF Untrained	1.62	0.00	5.85	5.13
NN RNMF Unsupervised	5.00	2.66	6.63	11.89
NN NMF Supervised	6.36	3.37	7.10	16.96
NN RNMF Dict. update	**6.55**	**3.55**	**7.24**	**17.65**

Evaluation

As the evaluation criteria, we used the BSS-EVAL metrics [116], which calculate the *source-to-distortion ratio* (SDR),[2] the *source-to-artifacts ratio* (SAR), and the *source-to-interference ratio* (SIR). As in [44], we computed the global normalized SDR,

$$\text{GNSDR} = \sum_{i=1}^{N} \delta_i (\text{SDR}(\hat{s}, s) - \text{SDR}(x, s)),$$

where $\hat{s}$ and s are the corresponding original and estimated voice signal, x is the mixture, δ_i is the relative duration of each of the N testing pieces. Prefix "G" indicates average sample performance,e.g. GSAR. We also computed the *signal-to-noise ratio* (SNR).

[2] In this work the SDR is computed using the latest release of the BSS-EVAL code. The reported values are higher (equally for all algorithms) than the ones reported in [44], since they used the older release of that package.

Comparison of separation methods

We evaluated the proposed *NN RNMF* using the different training settings discussed in Section 7.5.1. In all our examples (except when explicitly mentioned), we used $T = 10$ layers and $q = 20$. We compare these result against three exact solvers: *ADMoM RPCA* solving (2.3.8) with $\lambda = 1/\sqrt{n}$ (as suggested in [44]) via the alternating direction method of multipliers [63], for which the code from [44] was used; *Proximal RPCA* solving (6.4.8) using the proximal method from [12], with $\lambda = \sqrt{2n}\sigma$ and $\lambda_* = \sqrt{2}\sigma$ with $\sigma = 0.3$ set following [12]; and *Multiplicative RNMF* solving (7.4.6) using the standard multiplicative algorithm. We also include the *NN RPCA* studied in the robust PCA problem in Section 6.5. In that case, no non-negativity constraints are imposed.

In all experiments, the spectrogram of each mixture was computed using a window size of 1024 and a step size of 256 samples (at 16 KHz sampling rate). Training was performed using 1000 safe-guarded gradient descent iterations on a random subset of 10.000 spectral frames for training and the same amount of distinct frames for cross-validation.

Table 7.1 summarizes the performance of the compared methods. The best performance is achieved by the *NN RNMF* with trained dictionary. The use of the proximal RPCA algorithm allowing for inexact reconstruction of the data (thus accounting for unstructured noise) gives almost 0.5 dB improvement over [44]. The use of unsupervised training was more successful in the *NN RPCA*; however, both *NN RPCA* and *NN RNMF* outperform ADMoM RPCA.

The complexity of the proposed systems is significantly lower to the one of exact algorithms: our unoptimized Matlab code that uses GPU acceleration is capable of computing the networks about 70 faster than real time, while a preliminary implementation on iPhone 4S is online and $6-7$ times faster than real time (after offline training).

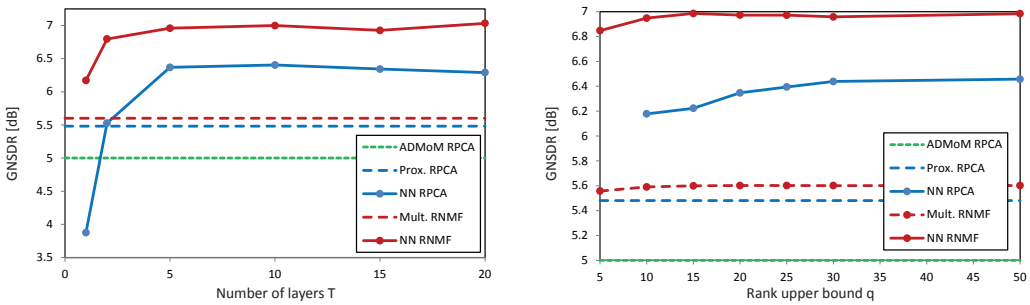


Figure 7.3: Performance of the supervised *NN RPCA* and *NN RNMF* on the MIR-1K dataset for different number of layers T (left, q fixed to 20), and values of the rank bound q (right, T fixed to 10). GNSDR of the recovered vocal track is used as the comparison criterion. For reference, the performance of exact RPCA and RNMF is given.

Parameter selection

We also evaluated the performance of the supervised RPCA and RNMF networks as a function of the two principal parameters: the number of layers T and the rank bound q, see Figure 7.3.

Supervised learning has a dramatic effect on the performance of the networks. With just two layers, the RPCA network already outperforms the exact RPCA algorithms; as a reference, an untrained network, with the parameters $\mathbf{U}, \mathbf{S}$, and $\mathbf{t}$ set according to Algorithm 6, requires over 15 layers to approach this performance. This phenomenon is even more pronounced in the case of RNMF. The influence of the number of layers quickly saturates; slight oscillations in the GNDSR are due to the randomization used at training.

In contrast, the effect of q is less dramatic. The networks outperform the exact algorithms already for $q = 5$ and the performance saturates for $q \geq 30$. This is radically different from the behavior of standard NMF approaches, in which setting the number of columns in the non-negative factor $\mathbf{U}$ significantly affects the performance. In fact, RNMF with $\lambda_* = 0$ as [130] yields 5.60 dB GNSDR for $q = 1$, which drops to 2.88 dB for $q = 3$ and to -2.5 dB for $q = 10$.

Table 7.2: Performance of NN RNMF on the vocal trak of *Sunrise* song. Audio files are available for download

Method	**NSDR**	**SNR**	**SAR**	**SIR**
Ideal freq. mask	14.98	5.84	18.46	39.40
ADMoM RPCA [44]	1.61	2.99	11.13	6.60
Supervised (*MIR-1K*)	7.16	4.86	14.21	13.25
Supervised (*We are in love*)	7.85	5.47	15.35	13.59
Supervised (*Sunrise*)	10.93	5.67	16.16	19.20
Semi-supervised (*We are in love*)	7.35	4.69	11.39	20.01
Semi-supervised (*Sunrise*)	8.46	5.11	12.20	23.97

Supervised training settings

We evaluated the influence of the different training regimes on the performance of the networks on Shannon Hurley's song *Sunrise*, available from `archive.org`. The song was resampled at 16 kHz and voice was artificially mixed with the guitar accompaniment with equal energies. Three distinct datasets were used for training the nets: two singers from MIR-1K used in the previous experiments; another Shannon Hurley's song *We are in love*; and the same *Sunrise*, song on which the testing was performed (given only for comparison). Supervised and semi-supervised regimes were used.

Table 7.2 summarizes the obtained results. RNMF networks trained using mixtures from MIR-1K outperform [44] by nearly 5.5 dB GNSDR; training on more singer-specific data (*We are in love* song) improves this result by about 0.7 dB. ; finally, training on a mixture from the same song yields over 3.5 dB improvement. We conclude that training the networks on unrelated singers and accompaniments already achieves very high performance. Semi-supervised training on the *We are in love* song yields a minor improvement over MIR-1K, and cedes 0.5 dB to the fully-supervised training. We conclude that in the absence of synchronized voice and music tracks for supervised training, semi-supervised training still produces comparable results.

7.7 Speech denoising and speaker identification using low-rank sparse models

7.7.1 Proposed method

As mentioned in the introduction the problem of speech denoising can be seen as source separation problem, and consequently tackled via NMF. However, the standard setting of NMF is not well suited for online or real time processing, since it requires observing the whole dataset. Recent studies have proposed supervised and semi-supervised approaches for performing online and real time NMF. [121] proposed an online NMF algorithm in the context of document clustering and [59] proposed an online NMF with the Itakura-Saito (IS) divergence for analyzing long recordings. In the speech denoising setting, [51] proposed supervised and semi-supervised algorithms for online real-time speech denosing assuming that a model of the speaker is available, while [23] propose a dual approach in which the model of the noise is the one known.

In line with the literature on NMF-based source separation, we propose to model the speech and the noise by a pair of pre-trained dictionaries $\mathbf{U}_\mathrm{s}$ and $\mathbf{U}_\mathrm{n}$. We represent the speech signal with the sparse low rank model introduced in Section 7.4.3. Given a noisy signal $\mathbf{X}$, we decompose it into speech and noise signals by finding the activation matrices $\mathbf{S}_\mathrm{s}$ and $\mathbf{S}_\mathrm{n}$ minimizing

$$\min_{\mathbf{S}_\mathrm{s}\geq\mathbf{0},\mathbf{S}_\mathrm{n}\geq\mathbf{0}} \frac{1}{2}\|\mathbf{X}-\mathbf{U}_\mathrm{n}\mathbf{S}_\mathrm{n}-\mathbf{U}_\mathrm{s}\mathbf{S}_\mathrm{s}\|_\mathrm{F}^2+\frac{\lambda_*}{2}(\left\|\mathbf{S}_\mathrm{s}\right\|_\mathrm{F}^2+\left\|\mathbf{S}_\mathrm{n}\right\|_\mathrm{F}^2)+\lambda\left\|\mathbf{S}_\mathrm{s}\right\|_1 \tag{7.7.12}$$

Note that the ℓ_2 regularization terms on the dictionaries are superfluous, since they latter are assumed fixed. Also observe that we impose sparsity of the activation corresponding to the speech signal only, as noise is poorly described by sparse activation. The model can be easily adapted to source separation; in that $\mathbf{S}_n$ would correspond to another speaker and

Table 7.3: Performance of denoising methods on the GRID dataset with different background noises, in terms of GSDR in dB. For each method, two numbers are given corresponding to the noise-specific (first) and noise-agnostic (second) settings.

Method	Exact NMF (noise only)	Exact NMF (noise+voice)	NN SLRNMF (Untrained)	NN SLRNMF (Sup. $\beta = 2$)	NN SLRNMF (Sup. $\beta = 0$)	NN SLRNMF (Unsup.)
street	4.67 | 2.57	6.90 | 6.77	6.07 | 6.37	7.21 | 7.21	8.08 | 7.70	6.22 | 6.50
restaurant	3.37 | 2.52	6.20 | 6.18	4.92 | 5.42	6.45 | 6.27	7.49 | 7.33	5.14 | 5.57
car	6.57 | 3.13	7.89 | 7.02	6.80 | 6.68	8.13 | 7.61	8.95 | 8.23	7.00 | 6.84
exhibition	7.38 | 3.14	8.85 | 7.95	7.78 | 7.79	9.15 | 8.60	10.07 | 9.46	7.95 | 7.88
train	6.53 | 3.24	8.48 | 7.21	7.55 | 6.95	8.71 | 7.78	9.22 | 8.01	7.70 | 7.06
airport	4.07 | 2.86	6.71 | 6.47	5.40 | 5.77	7.07 | 6.88	7.63 | 7.13	5.60 | 5.92
average	5.43 | 2.91	7.51 | 6.93	6.42 | 6.50	7.79 | 7.39	8.57 | 7.98	6.60 | 6.63

its sparsity would be enforced through an ℓ_1 term. As an illustration to the regularization effect of the low-rank model is shown in Figure 2 in the supplementary material.

Problem (7.7.12) is column-wise separable and it can be solved using a simple adaptation of the multiplicative algorithms described in Section 7.3. Once $\mathbf{S}_s$ and $\mathbf{S}_n$ are obtained, the time-frequency representation of the speech and the noise is estimated as $\mathbf{U}_s\mathbf{S}_s$ and $\mathbf{U}_n\mathbf{S}_n$, respectively. Next, a time-frequency mask is constructed and the speech is recovered from the mixture by Wiener filtering, as is standard in NMF-based source separation, see Section 7.3.

The proposed model can also be used for speaker identification in the presence of noise. When $\mathbf{U}_n$ matches the noise and $\mathbf{U}_s$ is tuned to a particular speaker, the minimal cost attained in (7.7.12) is likely to be small. On the other hand, using a dictionary $\mathbf{U}'_s$ tuned for another speaker, the cost is likely to be higher as the dictionary is less suitable for the given data. This suggests a very commonly used classification scheme: a collection of dictionaries is trained, each for an individual speaker. At testing, a collection of data vectors is encoded in each of the dictionaries by solving (7.7.12). The class assignment is made based on the minimum cost attained by the solutions.

7.7.2 Experimental evaluation

We evaluated the separation performance of the proposed methods on a subset of the GRID dataset [18] containing 10 distinct speakers; each speaker comprising 1000 short clips of the form "*put red at G9 now*". Three sets of 200 distinct clips each were used for training, validation, and testing. The GRID clips were resampled to 8 KHz and artificially contaminated by six categories of noise recorded from different real environments (street, restaurant, car, exhibition, train, and airport) taken from the AURORA corpus [81].[3] The voice and the noise clips were mixed linearly with equal energy (0 dB SNR).

As the evaluation criteria, we again used the *source-to-distortion ratio* (SDR) from the BSS-EVAL metrics [116]. Following [44], we computed the global SDR (GSDR) by averaging the SDR over all test clips from the same speaker and noise weighted by the clip duration.

Comparison of denoising methods

We evaluated the proposed NN SLRNMF encoders with the different training settings discussed in Section 7.5.1. In all our examples we used $T = 10$ layers and $q = 50$. As a reference, we also evaluate untrained networks with parameters initialized according to Algorithm 7. We compare these result against exact low-rank NMF with noise model only, and that involving both noise and voice models; both with the Euclidean data term ($\beta = 2$). We used $\lambda = \sqrt{2N}\sigma$ and $\lambda_* = \sqrt{2}\sigma$ with $\sigma = 0.3$ set following [12]; such a setting guarantees that if the data $\mathbf{X}$ consist of n frames of zero-mean white noise of variance σ^2, then both $\mathbf{U}_\mathrm{n}\mathbf{S}_\mathrm{n}$ and $\mathbf{U}_\mathrm{s}\mathbf{S}_\mathrm{s}$ are zero.

In all experiments, the spectrogram of each mixture was computed using a window of size 512 and a step size of 128 samples (at 8 KHz sampling rate). Training was performed using 1500 safe-guarded gradient descent iterations on a random selection of $10K$ spectral

[3] Note that both datasets are publicly available; the exact files used can be provided upon request.

frames for training and the same amount of distinct frames for cross-validation. All methods were trained in two distinct settings: the *noise-specific* setting in which the noise category is assumed to be known as the training is performed only on that noise; and the *noise-agnostic* setting, in which the noise is only known to belong to one of the six categories and the training is performed on a random selection of all the noises.

Table 7.3 summarizes the performance of the compared methods. We observe that the introduction of a low-rank sparse voice model improves the quality of denoising by exact NMF algorithms by over 2 dB GSDR in the noise-specific setting, and over 4 dB GSDR in the noise-agnostic one. The NN NMF encoder trained in the supervised regime to produce the best approximation of the voice and noise tracks known at training consistently outperforms the exact NMF algorithms and NN NMF encoders trained in other regimes, achieving over 7 dB GSDR in both the noise-specific and noise-agnostic settings. The use of the Itakura-Saito divergence ($\beta = 0$) in the supervised setting brings further improvement by about 1 dB.

The complexity of the proposed NN NMF systems is significantly lower to the one of exact algorithms: a preliminary implementation on iPad is over four times faster than real time. Also, the latency of our implementation is in the order of hundreds of milliseconds, while the exact algorithms require a significant amount of data to be observed.

Comparison of speaker identification methods

We evaluated the classification capabilities of different NN SLRNMF encoders encoders with different training settings used in the separation experiments. We compare these result against exact sparse, low-rank NMF with noise and voice models. Table 7.4 summarizes the classification rates of the compared methods. In the noisy case (for which all training was performed), the best performance is achieved when using the NN NMF encoders with the discriminative loss. In the noiseless case, however, the best performance is achieved by NN SLRNMF trained in the unsupervised regime. We attribute this to the fact that while being

Table 7.4: Performance of speaker identification methods on the GRID dataset with different background noises in terms of classification rate. For each method, two numbers are given corresponding to the noise-specific (first) and noise-agnostic (second) settings.

Method	**Exact NMF (noise+voice)**	**NN SLRNMF (Untrained)**	**NN SLRNMF (Sup. $\beta = 2$)**	**NN SLRNMF (Unsup.)**	**NN SLRNMF (Discrim.)**
street	0.86 | 0.93	0.83 | 0.92	0.91 | 0.63	0.85 | 0.95	0.91 | 0.94
restaurant	0.91 | 0.90	0.85 | 0.89	0.89 | 0.83	0.92 | 0.92	0.90 | 0.97
car	0.90 | 0.94	0.91 | 0.91	0.91 | 0.65	0.93 | 0.95	0.96 | 0.87
exhibition	0.93 | 0.94	0.92 | 0.89	0.91 | 0.65	0.95 | 0.95	0.95 | 0.96
train	0.93 | 0.94	0.91 | 0.90	0.88 | 0.77	0.94 | 0.95	0.96 | 0.95
airport	0.92 | 0.94	0.88 | 0.91	0.85 | 0.65	0.93 | 0.95	0.98 | 0.94
average	0.91 | 0.93	0.88 | 0.90	0.89 | 0.69	0.92 | 0.94	0.94 | 0.94
noiseless	0.96 | 0.96	0.94 | 0.94	0.94 | 0.94	0.97 | 0.97	0.91 | 0.91

more discriminative, classifiers trained with $\mathscr{L}_{\text{dis}}$ have weaker generalization capabilities and perform worse under conditions distinct from those at training.

7.8 Conclusion

Marrying ideas from convex optimization with multi-layer neural networks, we have developed efficient architectures for real-time online single-channel separation. We introduced a way of combining NMF with low rank estimation and proposed two models for representing audio signals, namely: a robustified version of NMF and a sparse, low rank NMF. We successfully applied these models for the problems of of singing voice from musical accompaniment and speech denoising in the presence of non-stationary noise respectively. Our approach achieves state-of-the-art results on the MIR-1K datasets with orders of magnitude improvement in runtime and latency.

8 Conclusions

The main conclusion of this thesis is that, in many situations, sparse models and low rank representations can be greatly improved by incorporating structure of the problem or the data. Specifically:

New dictionary learning schemes:. we have shown that by promoting block-incoherence in the learned dictionaries, this is, promoting a discriminative structure, can significantly improve the performance of sparse models in classification and clustering tasks. The benefits of imposing these features on dictionaries were demonstrated in tasks such as object recognition and classification.

Structured sparse coding:. we have shown that prior knowledge of the structure in the pattern of non-zero coefficients can be included into sparse models. In particular, we introduced a new framework of collaborative hierarchical sparse coding, where multiple signals collaborate in their encoding, sharing code groups (models) and having (possible disjoint) sparse representations inside the corresponding groups. At a practical level, this model is very well suited for tackling source separation problems as demonstrated in speaker recognition and separation problems.

Physical structure and temporal dynamics:. we have developed a new model for representing quasi-harmonic signals that takes into account the physical properties of the signals. This new framework is based on a audio signal model that decouples between the information of pitch and timbre. Learned temporal dynamics can also included via a first order hidden Markov model, allowing for the representation of richer signals such as speech.

Learning efficient sparse and low rank representations:. we have proposed a new way of computing (or approximating) sparse codes. The underlying structure in the data is used to train learnable encoders capable of producing accurate approximation of the true sparse codes at a small fraction of the computational expense of the exact solvers with almost no performance loss. This encoders allow to incorporate elements of supervised learning into sparse models with great practical advantages. We evaluated these encoders in various real time video and audio processing applications obtaining results that, in some cases, outperform the state-of-the-art.

9 Bibliography

[1] H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19, 1974.

[2] F. Bach, R. Jenatton, J. Mairal, and G. Obozinski. Convex optimization with sparsity-inducing norms. In *Optimization for Machine Learning*. MIT Press, 2011.

[3] L. Balzano, R. Nowak, and B. Recht. Online identification and tracking of subspaces from highly incomplete information. In *Proc. of 48th Allerton Conference*, September 2010.

[4] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Img. Sci.*, 2:183–202, March 2009.

[5] N. Bertin, R. Badeau, and G. Richard. Blind signal decompositions for automatic transcription of polyphonic music: Nmf and k-svd on the benchmark. In *ICASSP*, pages 65–68, 2007.

[6] N. Bertin, R. Badeau, and E. Vincent. Enforcing harmonicity and smoothness in Bayesian non-negative matrix factorization applied to polyphonic music transcription. *IEEE Trans. Audio, Speech & Lang. Process.*, 18:538–549, 2010.

[7] P. Boufounos, G. Kutyniok, and H. Rauhut. Sparse recovery from combined fusion frame measurements. arXiv:0912.4988v1, 2010.

[8] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, Mar. 2004.

[9] A. M Bruckstein, D. L Donoho, and M. Elad. From sparse solutions of systems of equations to sparse modeling of signals and images. *SIAM Review*, 51(1):34, 2009.

[10] Jian-Feng Cai, Emmanuel J. Candès, and Zuowei Shen. A singular value thresholding algorithm for matrix completion. *SIAM J. on Opt.*, 20(4):1956–1982, 2010.

[11] E. Candès. The restricted isometry property and its implications for compressed sensing. *C. R. Acad. Sci. Paris S'er. I Math.*, 346:589–592, 2008.

[12] E. Candès, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *Journal of the ACM*, 58(3), May 2011.

[13] E. Candès, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Trans. Inform. Theory*, 52(2):489–509, February 2006.

[14] J. Chen and X. Huo. Theoretical results on sparse representations of multiple-measurement vectors. *IEEE Trans. Sig. Proc.*, 54(12):4634–4643, Dec. 2006.

[15] S. Chen, D. Donoho, and M. Saunders. Atomic decomposition by basis pursuit. *SIAM J. Scientific Computing*, 20(1):33–61, 1999.

[16] H. Cheng, Z. Liu, and J. Yang. Motion segmentation via robust subspace separation in the presence of outlying, incomplete, or corrupted trajectories. In *ICCV*, 2009.

[17] P.L. Combettes and J.-C. Pesquet. Proximal splitting methods in signal processing. In *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, pages 185–212. Springer, 2011.

[18] M. Cooke, J. Barker, S. Cunningham, and X. Shao. An audio-visual corpus for speech perception and automatic speech recognition. *Journal of the Acoustical Society of America*, 120(5 Pt 1):2421–2424, 2006.

[19] S. Cotter, B. Rao, K. Engan, and K. Kreutz-Delgado. Sparse solutions to linear inverse problems with multiple measurement vectors. *IEEE Trans. Sig. Proc.*, 53(7):2477–2488, July 2005.

[20] A. d'Aspremont, L. El Ghaoui, M. Jordan, and G. Lanckriet. A direct formulation for sparse PCA using semidefinite programming. *Neural Information Processing Systems*, 17, 2004.

[21] I. Daubechies, M. Defrise, and C. De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Comm. on Pure and Applied Mathematics*, 57:1413–1457, 2004.

[22] D. Donoho. Compressed sensing. *IEEE Trans. on Inf. Theory*, 52(4):1289–1306, Apr 2006.

[23] Zhiyao Duan, Gautham J. Mysore, and Paris Smaragdis. Online plca for real-time semi-supervised source separation. In *LVA/ICA*, pages 34–41, 2012.

[24] J.-L. Durrieu, B. David, and G. Richard. A musically motivated mid-level representation for pitch estimation and musical audio source separation. *Selected Topics in Signal Processing, IEEE Journal of*, 5(6):1180 –1191, Oct. 2011.

[25] J.-L. Durrieu, G. Richard, B. David, and C. Févotte. Source/filter model for unsupervised main melody extraction from polyphonic audio signals. *IEEE Trans. Audio, Speech & Lang. Process.*, 18(3):564–575, 2010.

[26] Jean-Louis Durrieu, Bertrand David, and Gaël Richard. A musically motivated mid-level representation for pitch estimation and musical audio source separation. *J. Sel. Topics Signal Processing*, 5(6):1180–1191, 2011.

[27] M. Elad. Optimized projections for compressed-sensing. *IEEE Trans. SP*, 55(12):5695–5702, Dec. 2007.

[28] Y. C. Eldar, P. Kuppinger, and H. Bölcskei. Block-sparse signals: Uncertainty relations and efficient recovery. *IEEE Trans. SP*, 58(6):3042–3054, June 2010.

[29] Y. C. Eldar and M. Mishali. Robust recovery of signals from a structured union of subspaces. *IEEE Trans. Inform. Theory*, 55(11):5302–5316, Nov. 2009.

[30] Y. C. Eldar and H. Rauhut. Average case analysis of multichannel sparse recovery using convex relaxation. *IEEE Trans. Inform. Theory*, 56(1):505–519, 2010.

[31] E. Elhamifar and R. Vidal. Sparse subspace clustering. In *CVPR*, 2009.

[32] C. Fevotte, N. Bertin, and J.-L. Durrieu. Nonnegative matrix factorization with the itakura-saito divergence. with application to music analysis. *Neural Computation*, 21(3):793–830, Mar. 2009.

[33] J. Friedman, T. Hastie, and R. Tibshirani. A note on the group lasso and a sparse group lasso. preprint (2010).

[34] A. Ganesh, Z. Zhou, and Y. Ma. Separation of a subspace-sparse signal: Algorithms and conditions. In *ICASSP*, volume 14, april 2009.

[35] J. Ganseman, P. Scheunders, G. J. Mysore, and J. S. Abel. Evaluation of a score-informed source separation system. In *ISMIR*, pages 219–224, 2010.

[36] R. Giryes, M. Elad, and Y. C. Eldar. The projected GSURE for automatic parameter tuning in iterative shrinkage methods. Submitted to *Applied and Computational Harmonic Analysis*, 2010.

[37] I. Goodfellow, Q. Le, A. Saxe, H. Lee, and A. Y. Ng. Measuring invariances in deep networks. In *In NIPS*, pages 646–654. 2009.

[38] B.V. Gowreesunker and Ahmed H. Tewfik. A novel subspace clustering method for dictionary design. In *ICA, Lecture Notes in Computer Science*, pages 34–41. Springer, 2009.

[39] K. Gregor and Y. LeCun. Learning fast approximations of sparse coding. In *ICML*, pages 399–406, 2010.

[40] Mikael Henaff, Kevin Jarrett, Koray Kavukcuoglu, and Yann LeCun. Unsupervised learning of sparse features for scalable audio classification. In *ISMIR*, 2011.

[41] R. Hennequin, B. David, and R. Badeau. Score informed audio source separation using a parametric model of non-negative spectrogram. In *ICASSP*, May 2011.

[42] C.L. Hsu and J.S.R. Jang. On the improvement of singing voice separation for monaural recordings using the MIR-1K dataset. *IEEE Trans. on Audio, Speech, and Lang. Proc.*, 18(2):310–319, 2010.

[43] K. Huang and S. Aviyente. Sparse representation for signal classification. In *NIPS*, 2006.

[44] P-S. Huang, S.D. Chen, P. Smaragdis, and M. Hasegawa-Johnson. Singing-voice separation from monaural recordings using robust principal component analysis. In *ICASSP*, 2012.

[45] Nuance Inc. Macspeech scribe. http://www.nuance.com/, 2011.

[46] K. Jarrett, K. Kavukcuoglu, M.A. Ranzato, and Y. LeCun. What is the best multi-stage architecture for object recognition? In *CVPR*, pages 2146–2153, 2009.

[47] R. Jenatton, J.-Y. Audibert, and F. Bach. Structured variable selection with sparsity-inducing norms. Technical report, arXiv:0904.3523, 2009.

[48] R. Jenatton, J. Mairal, G. Obozinski, and F. Bach. Proximal methods for hierarchical sparse coding. Technical Report HAL : inria-00516723, INRIA, 2010.

[49] R. Jenatton, J. Mairal, G. Obozinski, and F. Bach. Proximal methods for sparse hierarchical dictionary learning. In *ICML*, June 2010.

[50] R. Jenatton, J. Mairal, G. Obozinski, and F. Bach. Proximal methods for hierarchical sparse coding. *JMLR*, 12:2297–2334, 2011.

[51] Cyril Joder, Felix Weninger, Florian Eyben, David Virette, and Björn Schuller. Real-time speech separation by semi-supervised nonnegative matrix factorization. In *LVA/ICA*, pages 322–329, 2012.

[52] I. T. Jolliffe. *Principal Component Analysis*. Springer, second edition, 2002.

[53] J. Joseph. *Why only two ears? Some indicators from the study of source separation using two sensors*. PhD thesis, Indian Institute of Science, 2004.

[54] K. Kavukcuoglu, M.A. Ranzato, and Y. LeCun. Fast inference in sparse coding algorithms with applications to object recognition. *arXiv:1010.3467*, 2010.

[55] I. Kemelmacher-Shlizerman and S. M. Seitz. Face reconstruction in the wild. In *ICCV*, pages 1746–1753, 2011.

[56] S. Kim and E. P. Xing. Tree-guided group lasso for multi-task regression with structured sparsity. In *ICML*, June 2010.

[57] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR '06: Proc. of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2169–2178, Washington, DC, USA, 2006. IEEE Computer Society.

[58] D.D. Lee and H.S. Seung. Learning parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

[59] A. Lefèvre, F. Bach, and C. Févotte. Online algorithms for nonnegative matrix factorization with the itakura-saito divergence. In *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, Mohonk, NY, Oct. 2011.

[60] Y. Li and S. Osher. Coordinate descent optimization for ℓ_1 minimization with application to compressed sensing; a greedy algorithm. *Inverse Problems and Imaging*, 3:487–503, 2009.

[61] Y. Li and D. Wang. Separation of singing voice from music accompaniment for monaural recordings. *IEEE Transactions on Audio, Speech & Language Processing*, 15(4):1475–1487, 2007.

[62] C.-J. Lin. Projected gradient methods for non-negative matrix factorization. *Neural Computation*, 19:2756–2779, 2007.

[63] Z. Lin, M. Chen, and Y. Ma. The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices. *Arxiv preprint arXiv:1009.5055*, 2010.

[64] Z. Lin, A. Ganesh, J. Wright, L. Wu, M. Chen, and Y. Ma. Fast convex optimization algorithms for exact recovery of a corrupted low-rank matrix. preprint, July 2009.

[65] A. Liutkus, Z. Rafii, R. Badeau, B. Pardo, and G. Richard. Adaptive filtering for music/voice separation exploiting the repeating musical structure. In *ICASSP*, 2012.

[66] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online dictionary learning for sparse coding. In *ICML*, 2009.

[67] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Discriminative learned dictionaries for local image analysis. In *CVPR*, 2008.

[68] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Supervised dictionary learning. In *NIPS*, volume 21, pages 1033–1040. 2009.

[69] S. Mallat. *A Wavelet Tour of Signal Proc.: The Sparse Way, 3rd edition*. Academic Press, 2008.

[70] M. Mardani, G. Mateos, and G. B. Giannakis. Unveiling network anomalies in large-scale networks via sparsity and low rank. In *Proc. of 44th Asilomar Conf. on Signals, Systems, and Computers, PaciiňĄc Grove, CA*, November 2011.

[71] Gonzalo Mateos and Georgios B. Giannakis. Robust PCA as bilinear decomposition with outlier-sparsity regularization. *arXiv.org:1111.1788*, 2011.

[72] M. Mishali and Y. C. Eldar. Reduce and boost: Recovering arbitrary sets of jointly sparse vectors. *IEEE Trans. Sig. Proc.*, 56(10):4692–4702, Oct. 2008.

[73] B. Moghaddam, Y. Weiss, and S. Avidan. Spectral bounds for sparse PCA: Exact & greedy algorithms. *Neural Information Processing Systems*, 18, 2006.

[74] Yadong Mu, Jian Dong, Xiaotong Yuan, and Shuicheng Yan. Accelerated low-rank visual recovery by random projection. In *CVPR*, pages 2609–2616, 2011.

[75] G. J. Mysore, P. Smaragdis, and B. Raj. Non-negative hidden markov modeling of audio with application to source separation. In *LVA/ICA*, pages 140–148, 2010.

[76] Y. Nesterov. Gradient methods for minimizing composite objective function. In *CORE Discussion Paper 2007/76, Center for Operations Research and Econometrics (CORE)*. Catholic University of Louvain, Louvain-la-Neuve, Belgium, 2007.

[77] A. Y. Ng, M. Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *NIPS*, volume 14. 2002.

[78] A. Opelt, A. Pinz, M. Fussenegger, and P. Auer. Generic object recognition with boosting. *IEEE Trans. on PAMI*, 28(3), March 2006.

[79] Alexey Ozerov, P. Philippe, Frédéric Bimbot, and Rémi Gribonval. Adaptation of bayesian models for single-channel source separation and its application to voice/music separation in popular songs. *IEEE Transactions on Audio, Speech & Language Processing*, 15(5):1564–1578, 2007.

[80] Caroline Pantofaru, Gyuri Dorko, Cordelia Schmid, and Martial Hebert. Combining regions and patches for object class localization. In *The Beyond Patches Workshop in conjunction with the IEEE conference on Computer Vision and Pattern Recognition*, pages 23 – 30, June 2006.

[81] D. Pearce and H.-G. Hirsch. The aurora experimental framework for the performance evaluation of speech recognition systems under noisy conditions. In *INTERSPEECH*, pages 29–32, 2000.

[82] J. Peng, J. Zhu, A. Bergamaschi, W. Han, D. Noh, J. Pollack, and P. Wang. Regularized multivariate regression for identifying master predictors with application to integrative genomics study of breast cancer. *Annals of Applied Statistics*, 4(1):53–77, 2010.

[83] Y. Peng, A. Ganesh, J. Wright, W. Xu, and Y. Ma. RASL: Robust alignment by sparse and low-rank decomposition for linearly correlated images. In *CVPR*, pages 763–770, 2010.

[84] G. Peyré. Sparse modeling of textures. *Journal of Mathematical Imaging and Vision*, 34:17–31, May 2009.

[85] C. Qiu and N. Vaswani. Real-time robust principal components' pursuit. *arXiv.org:1111.1788*, 2011.

[86] L. Rabiner and B.-H. Juang. *Fundamentals of Speech Recognition*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993.

[87] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[88] I. Ramírez and G. Sapiro. Sparse coding and dictionary learning based on the MDL principle. In *Proc. ICASSP*, May 2011.

[89] I. Ramirez, P. Sprechmann, and G. Sapiro. Classification and clustering via dictionary learning with structured incoherence. In *CVPR*, June 2010.

[90] T. Randen and J. H. Husoy. Filtering for texture classification: a comparative study. *IEEE Trans. Pattern Anal. Mach. Intell.*, 21(4):291–310, 1999.

[91] M. Ranzato, F. J. Huang, Y-L. Boureau, and Y. LeCun. Unsupervised learning of invariant feature hierarchies with applications to object recognition. In *CVPR*, 2007.

[92] S. R. Rao, R.Tron, R. Vidal, and Y. Ma. Motion segmentation via robust subspace separation in the presence of outlying, incomplete, or corrupted trajectories. In *CVPR*, 2008.

[93] B. Recht, M. Fazel, and P. A. Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Rev.*, 52(3):471–501, 2010.

[94] Benjamin Recht and Christopher Ré. Parallel stochastic gradient algorithms for large-scale matrix completion. `http://pages.cs.wisc.edu/~brecht/papers/11.Rec.Re.IPGM.pdf`, 2011.

[95] K. Rosenblum, L. Zelnik-Manor, and Y. C. Eldar. Sensing matrix optimization for block-sparse decoding. arXiv:1009.1533, Sep. 2010.

[96] N. Shoham and M. Elad. Alternating KSVD-denoising for texture separation. In *The IEEE 25-th Convention of Electrical and Electronics Engineers in Israel*, 2008.

[97] P. Smaragdis. Convolutive speech bases and their application to supervised speech separation. *IEEE Transactions on Audio, Speech & Language Processing*, 15(1):1–12, 2007.

[98] P. Sprechmann, P. Cancela, and G. Sapiro. Gaussian mixture models for score-informed instrument separation. In *ICASSP*, Mar. 2012.

[99] P. Sprechmann, I. Ramirez, P. Cancela, and G. Sapiro. Collaborative sources identification in mixed signals via hierarchical sparse modeling. In *Proc. ICASSP*, May 2011.

[100] P. Sprechmann, I. Ramirez, and G. Sapiro. Collaborative hierarchical sparse modeling. In *CISS*, Mar. 2010.

[101] P. Sprechmann, I. Ramírez, G. Sapiro, and Y. C. Eldar. C-hilasso: A collaborative hierarchical sparse modeling framework. *IEEE Trans. Signal Process.*, 59(9):4183–4198, 2011.

[102] J. Starck, M. Elad, and D. Donoho. Image decomposition via the combination of sparse representations and a variational approach. *IEEE Trans. Image Proc.*, 14:1570–1582, 2004.

[103] M. Stojnic. Block-length dependent thresholds in block-sparse compressed sensing. arXiv:0907.3679, July 2009.

[104] A.D. Szlam and G. Sapiro. Discriminative k-metrics. In *ICML*, 2009.

[105] R. Tibshirani. Regression shrinkage and selection via the LASSO. *J. Royal Stat. Society: Series B*, 58(1):267–288, 1996.

[106] S.K. Tjoa, M.C. Stamm, W. S. Lin, and K.J.R. Liu. Harmonic variable-size dictionary learning for music source separation. In *ICASSP*, pages 413–416, Dallas, TX, Mar. 2010.

[107] A. Torralba, K. Murphy, and W. Freeman. Sharing visual features for multiclass and multiview object detection. *IEEE Trans. PAMI*, 29(5):854–869, 2007.

[108] F. De La Torre and M. J. Black. A framework for robust subspace learning. *International Journal of Computer Vision*, 54:2003, 2003.

[109] J. Tropp. Algorithms for simultaneous sparse approximation. part II: Convex relaxation. *Signal Processing*, 86(3):589–602, 2006.

[110] J. A. Tropp. Greed is good: Algorithmic results for sparse approximation. *IEEE Trans. IP*, 50:2231–2242, 2004.

[111] P. Tseng. Nearest q-flat to m points. *J. Optim. Theory Appl.*, 105(1):249–252, 2000.

[112] B. Turlach, W. Venables, and S. Wright. Simultaneous variable selection. *Technometrics*, 27:349–363, 2004.

[113] T. Tuytelaars and C. Schmid. Vector quantizing feature space with a regular lattice. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8, 2007.

[114] Shankar Vembu and Stephan Baumann. Separation of vocals from polyphonic audio recordings. In *ISMIR*, pages 337–344, 2005.

[115] E. Vincent, N. Bertin, and R. Badeau. Adaptive harmonic spectral decomposition for multiple pitch estimation. *IEEE Trans. Audio, Speech & Lang. Process.*, 18(3):528–537, 2010.

[116] E. Vincent, R. Gribonval, and C. Févotte. Performance measurement in blind audio source separation. *IEEE Trans. Audio, Speech & Lang. Process.*, 14:1462–1469, 2006.

[117] T. Virtanen. Monaural sound source separation by nonnegative matrix factorization with temporal continuity and sparseness criteria. *IEEE Trans. Audio, Speech & Lang. Process.*, 15(3):1066–1074, 2007.

[118] U. von Luxburg. A tutorial on spectral clustering. *CoRR*, abs/0711.0189, 2007.

[119] A. Wagner, J. Wright, A. Ganesh, Z. Zhou, H. Mobahi, and Y. Ma. Towards a practical face recognition system: Robust alignment and illumination via sparse representation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(2):372 – 386, 2011.

[120] Tan Wai-tian, Cheung Gene, and Ma Yi. Face recovery in conference video streaming using robust principal component analysis. In *ICIP*, pages 3225–3228, 2011.

[121] F. Wang, P. Li, and A. C. König. Efficient document clustering via online nonnegative matrix factorizations. In *SDM*, pages 908–919, 2011.

[122] J. Wright, Y. Ma, J. Mairal, G. Spairo, T. Huang, and S. Yan. Sparse representations for computer vision and pattern recognition. In *Proc. IEEE*, 2009, to appear.

[123] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 31(2):210–227, 2009.

[124] S. Wright, R. Nowak, and M. Figueiredo. Sparse reconstruction by separable approximation. *IEEE Trans. Sig. Proc.*, 57(7):2479–2493, 2009.

[125] Z. J. Xiang, H. Xu, and P. J. Ramadge. Learning sparse representations of high dimensional data on large scale dictionaries. *In NIPS*, 24:900–908, 2011.

[126] H. Xu, C. Caramanis, and S. Sanghavi. Robust PCA via outlier pursuit. In *NIPS*, pages 2496–2504. 2010.

[127] J. Yang, K. Yu, Y. Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *CVPR*, 2009.

[128] G. Yu, G. Sapiro, and S. Mallat. Solving inverse problems with piecewise linear estimators: From gaussian mixture models to structured sparsity. *CoRR*, abs/1006.3056, 2010. http://arxiv.org/abs/1006.3056.

[129] M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *J. Royal Stat. Society, Series B*, 68:49–67, 2006.

[130] L. Zhang, Z. Chen, M. Zheng, and X. He. Robust non-negative matrix factorization. *Frontiers of Electrical and Electronic Engineering in China*, 6:192–200, 2011.

[131] P. Zhao, G. Rocha, and B. Yu. The composite absolute penalties family for grouped and hierarchical variable selection. *Annals of Statistics*, 37(6A):3468, 2009.

[132] M. Zhou, H. Chen, J. Paisley, L. Ren, G. Sapiro, and L. Carin. Non-parametric bayesian dictionary learning for sparse image representations. In *Proceedings of Advances in Neural Information Processing Systems (NIPS)*, 2009.

[133] Z. Zhou, X. Li, J. Wright, E. J. Candès, and Y. Ma. Stable principal component pursuit. In *ISIT*, pages 1518–1522, 2010.

[134] H. Zou, T. Hastie, and R. Tibshirani. Sparse principal component analysis. *Journal of Computational and Graphical Statistics*, 15(2), 2003.

A Theoretical guarantees for HiLasso

This appendix is dedicated to ellaborate in the theoretical guarantees studied for the hierarchical sparse coding scheme introduced in Chapter 4.

In our current theoretical analysis, we study the case of a single measurement vector (signal) $\mathbf{x}$ (we comment on the collaborative case at the end of this section), and assume that there is no measurement noise or perturbation, so that $\mathbf{x} = \mathbf{D}\mathbf{a}$. Without loss of generality, we further assume that the cardinality $|G_r| = g, r = 1, \ldots, |\mathcal{G}|$, that is, all groups in $\mathcal{G}$ have the same size. The goal is to recover the code $\mathbf{a}$, from the observed $\mathbf{x}$, by solving the noise-free HiLasso problem:

$$\min_{\mathbf{a}\in\mathbb{R}^p} \left\{ \lambda \psi_{\mathcal{G}}(\mathbf{a}) + (1-\lambda)\|\mathbf{a}\|_1 \quad \text{s.t.} \quad \mathbf{x} = \mathbf{D}\mathbf{a} \right\}. \tag{A.0.1}$$

Note that we have replaced the two regularization parameters λ_1 and λ_2 by a single parameter λ, since scaling does not effect the optimal solution. Therefore, we can always assume that $\lambda_1 + \lambda_2 = 1$.

Our goal is to develop conditions under which the HiLasso program of (A.0.1) will recover the true unknown vector $\mathbf{a}$. As we will see, the resulting set of recoverable signals is a superset of those recoverable by Lasso, that is, HiLasso is able to recover signals for which Lasso (or Group Lasso) will fail to do so.

We assume throughout this section that $\mathbf{a}$ has group sparsity k, namely, no more than k of the group vectors $\mathbf{a}_{[g_i]}, i = 1, \ldots, |\mathcal{G}|$, have non-zero norm. In addition, within each group, we assume that not more than s elements are non zero, that is, $\|\mathbf{a}_{[g]}\|_0 \leq s$.

For $\lambda = 1$, (A.0.1) reduces to the Group Lasso problem, (2.1.3), whereas with $\lambda = 0$, (A.0.1) becomes equivalent to the Lasso problem, (2.1.2). Both cases have been treated previously in the literature and sufficient conditions have been derived on the sparsity levels and on the dictionary $\mathbf{D}$ to ensure that the resulting optimization problem recovers the true unknown vector $\mathbf{a}$. For example, in [29, 13, 11], conditions are given in terms of the restricted isometry property (RIP) of $\mathbf{D}$. In an alternative line of work, recovery conditions are based on coherence measures, which are easier to compute [28, 110]. Here, we follow the same spirit and consider coherence bounds that ensure recovery using the HiLasso approach. We also draw from [30] to briefly describe conditions under which the probability of error of recovering the correct groups, using the special case of the C-HiLasso with $\lambda_1 = 0$ (C-GLasso), falls exponentially to 0 as the number of collaborating samples N grows. Finally, recent theoretical results on block sparsity were reported in [103]. In particular, bounds on the number of measurements required for block sparse recovery were developed under the assumption that the measurement matrix $\mathbf{D}$ has a basis of the null-space distributed uniformly in the Grassmanian. The model is a block-sparse model, without hierarchical or collaborative components.

In this section we extend the group-wise indexing notation to refer both to subsets of rows and columns of arbitrary matrices as $\mathbf{W}_{[F,G]} := \{w_{ij} : i \in F, j \in G\}$. This is, $\mathbf{W}_{[F,G]} = \mathbf{I}_{[F]}^T \mathbf{W} \mathbf{J}_{[G]}$, where $\mathbf{I}$ and $\mathbf{J}$ are the identity matrices of the column and row spaces of $\mathbf{W}$ respectively. We define the sets $\Omega = \{1, 2, \ldots, p\}$ and $\Gamma = \{1, 2, \ldots, g\}$, and use $\overline{S}$ to denote the complement of a set of indices S, either with respect to Ω or Γ, depending on the context. The set difference between S and T is denoted as $S \setminus T$, $\emptyset$ represents the empty set, and $|S|$ denotes the cardinality of S.

A.1 Block-sparse coherence measures

We begin by reviewing previously proposed coherence measures. For a given dictionary $\mathbf{D}$, the (standard) coherence is defined as $\mu := \max_{i,j \neq i \in \Gamma} |\mathbf{d}_i^\intercal \mathbf{d}_j|$. This coherence was extended to the block-sparse setting in [28], leading to the definition of *block coherence*:

$$\mu_B := \max \left\{ \frac{1}{g} \rho(\mathbf{D}_{[G]}^\intercal \mathbf{D}_{[F]}),\ G, F \in \mathscr{G}, G \neq F \right\},$$

where $\rho(\cdot)$ is the spectral norm, that is, $\rho(\mathbf{Z}) := \lambda_{\max}^{1/2}(\mathbf{Z}^\intercal \mathbf{Z})$, with $\lambda_{\max}(\mathbf{W})$ denoting the largest eigenvalue of the positive semi-definite matrix $\mathbf{W}$. An alternate atom-wise measure of block coherence is given by the *cross coherence*,

$$\chi := \max \left\{ \max \left\{ |\mathbf{d}_i^T \mathbf{d}_j|,\ i \in G, j \in F \right\} G, F \in \mathscr{G}, G \neq F \right\}. \tag{A.1.2}$$

When $g = 1$ (each block is a singleton), $\mathbf{D}_{[G_r]} = \mathbf{d}_r$, so that as expected, $\chi = \mu_B = \mu$. While μ_B and χ quantify global properties of the dictionary $\mathbf{D}$, local block properties are characterized by the *sub-coherence*, defined as

$$\nu := \max \left\{ \max \left\{ |\mathbf{d}_i^T \mathbf{d}_j|,\ i, j \in G, i \neq j \right\}\ G \in \mathscr{G} \right\}. \tag{A.1.3}$$

We define $\nu = 0$ for $g = 1$. Clearly, if the columns of $\mathbf{D}_{[g]}$ are orthonormal for each group G, then $\nu = 0$. Assuming the columns of $\mathbf{D}$ have unit norm, it can be easily shown that μ, ν, χ and μ_B all lie in the range $[0, 1]$. In addition, we can easily prove that $\nu, \mu_B, \chi \leq \mu$. In our setting, $\mathbf{a}$ is block sparse, but has further internal structure: each sub-vector of $\mathbf{a}$ is also sparse. In order to quantify our ability to recover such signals, we expect that an appropriate coherence measure will be based on the definition of block sparsity, but will further incorporate the internal sparsity as well. Let $\mathbf{M} := \mathbf{D}^\intercal \mathbf{D}$ denote the Gram matrix of $\mathbf{D}$. Then, the standard block coherence μ_B is defined in terms of the largest singular

value of an off-diagonal $g \times g$ sub-block of $\mathbf{M}$. In a similar fashion, we will define *sparse block coherence* measures in terms of *sparse singular values*. As we will see, two different definitions will play a role, depending on where exactly the sparsity within the block enters. To define these, we note that the *spectral norm* $\rho(\mathbf{Z})$ of a matrix $\mathbf{Z}$ can be defined as

$$\rho(\mathbf{Z}) := \max_{\mathbf{u},\mathbf{v}} |\mathbf{u}^\intercal \mathbf{Z} \mathbf{v}| \qquad \text{s.t.} \quad \|\mathbf{u}\|_2 = 1, \|\mathbf{v}\|_2 = 1.$$

Alternatively, we can define $\rho(\mathbf{Z})$ as the largest singular value of $\mathbf{Z}$, $\rho(\mathbf{Z}) := \sigma_{\max}(\mathbf{Z}) = \sqrt{\lambda_{\max}(\mathbf{Z}^\intercal \mathbf{Z})}$,

$$\lambda_{\max}(\mathbf{Z}^\intercal \mathbf{Z}) := \max_{\mathbf{v}} \mathbf{v}^\intercal (\mathbf{Z}^\intercal \mathbf{Z}) \mathbf{v} \qquad \text{s.t.} \quad \|\mathbf{v}\|_2 = 1.$$

We now develop sparse analogs of $\rho(\mathbf{Z})$ and $\lambda_{\max}(\mathbf{Z}^\intercal \mathbf{Z})$. As we will see, the simple square-root relation no longer holds in this case. The *largest sparse singular value* is defined as [20]:

$$\rho^{ss}(\mathbf{Z}) := \max_{\mathbf{u},\mathbf{v}} |\mathbf{u}^\intercal \mathbf{Z} \mathbf{v}| \qquad \text{s.t.} \quad \|\mathbf{u}\|_2 = 1, \|\mathbf{v}\|_2 = 1, \|\mathbf{u}\|_0 \leq s, \|\mathbf{v}\|_0 \leq s. \tag{A.1.4}$$

Similarly, the *largest sparse eigenvalue* of $\mathbf{Z}^\intercal \mathbf{Z}$ is defined as [20, 134, 73],

$$\lambda^s_{\max}(\mathbf{Z}^\intercal \mathbf{Z}) := \max_{\mathbf{v}} \mathbf{v}^\intercal (\mathbf{Z}^\intercal \mathbf{Z}) \mathbf{v} \qquad \text{s.t.} \quad \|\mathbf{v}\|_2 = 1, \|\mathbf{v}\|_0 \leq s. \tag{A.1.5}$$

The *sparse matrix norm* is then given by

$$\rho^s(\mathbf{Z}) := \sqrt{\lambda^s_{\max}(\mathbf{Z}^\intercal \mathbf{Z})}. \tag{A.1.6}$$

Note that, in general, $\rho^s(\mathbf{Z})$ is not equal to $\rho^{ss}(\mathbf{Z})$. It is easy to see that $\rho^{ss}(\mathbf{Z}) \leq \rho^s(\mathbf{Z})$. For any matrix $\mathbf{Z}$, $\rho^{ss}(\mathbf{Z}) = \rho(\mathbf{Z}_{[F,G]})$ and $\rho^s(\mathbf{Z}) = \rho(\mathbf{Z}_{[T]})$, where F, G, T are subsets of $\Gamma = \{1, 2, \ldots, g\}$ of size s, chosen to maximize the corresponding singular value. Using

(A.1.4) and (A.1.6), we define two sparse block coherence measures:

$$\mu_B{}^{ss} := \max\left\{\frac{1}{g}\rho^{ss}(\mathbf{D}_{[G]}^{\mathsf{T}}\mathbf{D}_{[F]}),\, G,F\in\mathscr{G}, G\neq F\right\}, \tag{A.1.7}$$

$$\mu_B{}^{s} := \max\left\{\frac{1}{g}\rho^{s}(\mathbf{D}_{[G]}^{\mathsf{T}}\mathbf{D}_{[F]}),\, G,F\in\mathscr{G}, G\neq F\right\}. \tag{A.1.8}$$

The choice of scaling is to ensure that $\mu_B{}^{s}, \mu_B{}^{ss} \leq \mu_B$.

Note that, while $\rho^s(\mathbf{Z})$ (also referred to in the literature as *sparse principal component analysis* (SPCA)) and $\rho^{ss}(\mathbf{Z})$ are in general NP-hard to compute, in many cases they can be computed exactly, or approximated, using convex programming techniques [20, 134, 73].

The following proposition establishes some relations between these new definitions and the standard coherence measures.

Proposition 2. The sparse block-coherence measures $\mu_B{}^{ss}, \mu_B{}^{s}$ satisfy

$$0 \leq \mu_B{}^{ss} \leq \frac{s}{g}\mu, \quad 0 \leq \mu_B{}^{s} \leq \sqrt{\frac{s}{g}}\mu. \tag{A.1.9}$$

Proof: The inequalities $\mu_B{}^{ss}, \mu_B{}^{s} \geq 0$ follow immediately from the definition. We obtain the upper bounds by rewriting $\rho^{ss}(\mathbf{Z})$ and $\rho^{s}(\mathbf{Z})$ and then using the Geršgorin Theorem,

$$\rho^{ss}(\mathbf{Z}) = \lambda_{\max}^{1/2}(\mathbf{Z}_{[F,G]}^{\mathsf{T}}\mathbf{Z}_{[F,G]}) \overset{(a)}{\leq} \sqrt{\max_l \sum_{r=1}^{s} |e_{lr}|} \leq \sqrt{s \max_{l,r} |e_{lr}|}, \tag{A.1.10}$$

$$\rho^{s}(\mathbf{Z}) = \lambda_{\max}^{1/2}(\mathbf{Z}_{[T]}^{\mathsf{T}}\mathbf{Z}_{[T]}) \overset{(b)}{\leq} \sqrt{\max_l \sum_{r=1}^{s} |e'_{lr}|} \leq \sqrt{s \max_{l,r} |e'_{lr}|}, \tag{A.1.11}$$

where e_{lr} and e'_{lr} are the elements of $\mathbf{E} = \mathbf{Z}_{[F,\Gamma]}^{\mathsf{T}}\mathbf{Z}_{[F,\Gamma]}$ and $\mathbf{E}' = \mathbf{Z}^{\mathsf{T}}\mathbf{Z}$, and (a), (b) are a consequence of Geršgorin's disc theorem.

The entries of $\mathbf{Z} = \mathbf{D}_{[G_i]}^{\mathsf{T}}\mathbf{D}_{[G_j]}$ for $i \neq j$ have absolute value smaller than or equal to μ, and the size of $\mathbf{Z}$ is $g \times g$. Therefore, $|e_{k\ell}| \leq s\mu^2$ and $|e'_{k\ell}| \leq g\mu^2$. Substituting these values

into (A.1.10) and (A.1.11) concludes the proof of the upper bounds on ${\mu_B}^{ss}$ and ${\mu_B}^{s}$. ■

A.2 Recovery proof

Our main recovery result is stated as follows. Suppose that $\mathbf{a}$ is a block k-sparse vector with blocks of length g, where each block has sparsity exactly s,[1] and let $\mathbf{x} = \mathbf{D}\mathbf{a}$. We rearrange the columns in $\mathbf{D}$ and the coefficients in $\mathbf{a}$ so that the first k groups, $\{G_1, G_2, \ldots, G_k\}$ correspond to the non-zero (active) blocks. Within each block $G_i, i \leq k$, the first s indices, represented by the set S_i, correspond to the s nonzero coefficients in that block, and the index set $T_i = G_i \setminus S_i$ represents its $(g-s)$ inactive elements, so that $G_i = [S_i \; T_i]$. The set $G_0 = \bigcup_{i=1}^{k} G_i$ contains the indices of all the active blocks of $\mathbf{a}$, whereas $\overline{G_0} = \Omega \setminus G$ contains the inactive ones. Similarly, $S_0 = \bigcup_{i=1}^{k} S_i$ contains the indices of all the active coefficients/atoms in $\mathbf{a}$ and $\mathbf{D}$ respectively, $\overline{S_0} = \Omega \setminus S_0$ indexes the inactive coefficients/atoms in $\mathbf{a}/\mathbf{D}$, and $T_0 = \bigcup_{i=1}^{k} T_i$ indexes the inactive coefficients/atoms within the active blocks. These indexing conventions are exemplified in Figure A.1(left). With these conventions we can write $\mathbf{x} = \mathbf{D}_{[G_0]}\mathbf{a}_{[G_0]} = \mathbf{D}_{[S_0]}\mathbf{a}_{[S_0]}$.

An important assumption that we will rely on throughout, is that the columns of $\mathbf{D}_{[G_0]}$ must be linearly independent for any G_0 as defined above.Under this assumption, $\mathbf{D}_{[S_0]}^{\mathsf{T}}\mathbf{D}_{[S_0]}$ is invertible and we can define the pseudo-inverse $\mathbf{H} := (\mathbf{D}_{[S_0]}^{\mathsf{T}}\mathbf{D}_{[S_0]})^{-1}\mathbf{D}_{[S_0]}^{\mathsf{T}}$. For reasons that will become clear later, we will also need a second, oblique pseudo-inverse, $\mathbf{Q} := (\mathbf{D}_{[S_0]}^{\mathsf{T}}(\mathbf{I}-\mathbf{P})\mathbf{D}_{[S_0]})^{-1}\mathbf{D}_{[S_0]}^{\mathsf{T}}(\mathbf{I}-\mathbf{P})$, where $\mathbf{P}$ is an orthogonal projection onto the range of $\mathbf{D}_{[T_0]}$, that is, $\mathbf{P}\mathbf{D}_{[T_0]} = \mathbf{D}_{[T_0]}$. It is easy to check that

$$\mathbf{Q}\mathbf{D}_{[T_0]} = \mathbf{0}, \quad \text{and} \quad \mathbf{Q}\mathbf{D}_{[S_0]} = \mathbf{I}. \tag{A.2.12}$$

[1] These conditions are non-limiting, since we can always complete the vector with zeros.

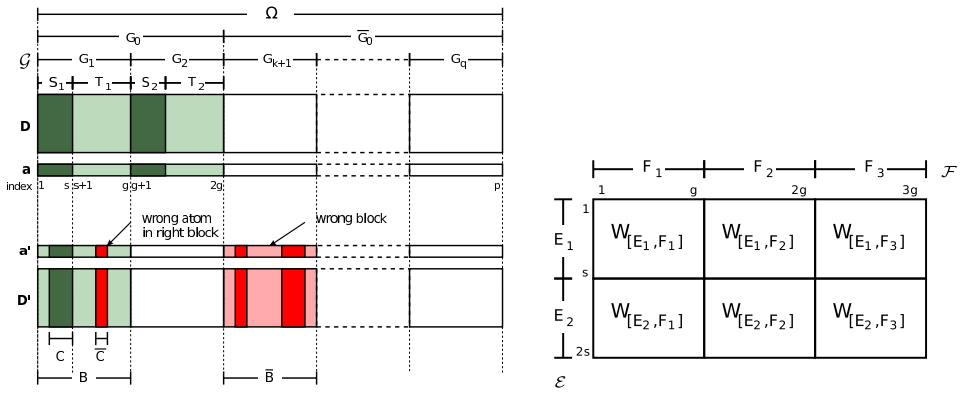


Figure A.1: Left: Indexing conventions, here shown for $g = 8$, $k = 2$ and $s = 3$. Shaded regions correspond to active elements/atoms. Active blocks are light-colored, and active elements/coefficients are dark colored. Here $\mathbf{a}'$ represents an alternate representation of $\mathbf{x}$, $\mathbf{x} = \mathbf{D}\mathbf{a}'$. Blocks and atoms that are not part of the true solution $\mathbf{a}$ are marked in red. Right: partitioning of a matrix $\mathbf{W}$ performed by the measure $\rho_{[\mathscr{E},\mathscr{F}]}(\mathbf{W})$ with $\mathscr{E} = \{E_1, E_2\}$ and $\mathscr{F} = \{F_1, F_2, F_3\}$, where $|E_i| = s$ and $|F_j| = g$.

Equipped with these definitions we can now state our main result.

Theorem 1. Let $\mathbf{a}$ be a block k-sparse vector with blocks of length g, where each block has sparsity s. Let $\mathbf{x} = \mathbf{D}\mathbf{a}$ for a given matrix $\mathbf{D}$. A sufficient condition for the HiLasso algorithm (A.0.1) to recover $\mathbf{a}$ from $\mathbf{x}$ is that, for some $\alpha \leq 1$,

$$\rho_{[\mathscr{S}_0,\overline{\mathscr{G}}_0]}(\mathbf{Q}\mathbf{D}_{[\overline{G}_0]}) < \alpha, \tag{A.2.13}$$

$$\|\mathbf{H}\mathbf{D}_{[\overline{G}_0]}\|_{1,1} < \gamma, \quad \gamma \leq 1 + \frac{\lambda(1-\alpha)}{\sqrt{g}(1-\lambda)}, \tag{A.2.14}$$

$$\|\mathbf{H}\mathbf{D}_{[T_0]}\|_{1,1} < 1. \tag{A.2.15}$$

Here $\rho_{[\mathscr{E},\mathscr{F}]}(\mathbf{Z}) := \max_{F\in\mathscr{F}} \sum_{E\in\mathscr{E}} \rho(\mathbf{Z}_{[E,F]})$, is the block spectral norm defined in [28], the blocks defined by the sets of index sets $\mathscr{E}$ and $\mathscr{F}$ (see Figure A.1(right)). We also define $\mathscr{S}_0 = \{S_i : i = 1,\ldots,k\}$, $\overline{\mathscr{G}}_0 = \{G_i : i = k+1,\ldots,|\mathscr{G}|\}$ and $\mathscr{T}_0 = \{T_i : i = 1,\ldots,k\}$. Finally, $\|\mathbf{Z}\|_{1,1} := \max_r \left\|\mathbf{z}_r\right\|_1$, where $\mathbf{z}_r$ is the r-th column of $\mathbf{Z}$.

The above theorem can be interpreted as follows. With $\gamma = 1$, the conditions (A.2.14)-(A.2.15) are sufficient both for Lasso ($\lambda = 0$) and HiLasso to recover $\mathbf{a}$. However, if there

exists a $\gamma > 1$ for which condition (A.2.14) holds, then HiLasso will be able to recover $\mathbf{a}$ in a situation where Lasso is not guaranteed to do so. The idea is that, for $0 < \lambda < 1$, HiLasso trades off between the minimization of its ℓ_1 and ℓ_2 terms, by tightening the ℓ_2 term ($\alpha \leq 1$) to improve group recovery, while loosening the ℓ_1 term ($\gamma > 1$). Also, although not yet clear from conditions (A.2.13)–(A.2.15), we will see in Theorem 2 that the final data independent bounds are also a relaxation of the ones corresponding to Group Lasso when the solutions are block-dense. Therefore, the proposed model outperforms both standard Lasso and Group Lasso with regard to recovery guarantees. This is also reflected in the experimental results presented in the next section.

The sufficient conditions (A.2.13)–(A.2.15) depend on $\mathbf{D}_{[S_0]}$ and therefore on the nonzero blocks in $\mathbf{a}$, G_0, and the nonzero locations within the blocks, S_0, which, of course, are not known in advance. Nonetheless, Theorem 2 provides sufficient conditions ensuring that (A.2.13)–(A.2.15) hold, which are independent of the unknown signals, and depend only on the dictionary $\mathbf{D}$.

We now prove Theorem 1.

Proof: To prove that (A.0.1) recovers the correct vector $\mathbf{a}$, let $\mathbf{a}'$ be an alternative solution satisfying $\mathbf{x} = \mathbf{D}\mathbf{a}'$. We will show that $\lambda\psi_{\mathscr{G}}(\mathbf{a}) + (1-\lambda)\|\mathbf{a}\|_1 < \lambda\psi_{\mathscr{G}}(\mathbf{a}') + (1-\lambda)\left\|\mathbf{a}'\right\|_1$. Let the set G_0 contain the indices of all elements in the active blocks of $\mathbf{a}$. Let G_0' contain the indices of the active blocks in $\mathbf{a}'$. Then $\mathbf{x} = \mathbf{D}_{[G_0]}\mathbf{a}_{[G_0]} = \mathbf{D}_{[G_0']}\mathbf{a}'_{[G_0']}$.

By our assumptions, in each block of $\mathbf{a}_{[G_0]}$ there are exactly s nonzero values. Let the set $S_0 \subset G_0$ contain the indices of all nonzero elements in $\mathbf{a}$. We thus have $|S_0| = ks$. Using (A.2.12) we can write

$$\mathbf{a}_{[S_0]} = \mathbf{Q}\mathbf{D}_{[S_0]}\mathbf{a}_{[S_0]} = \mathbf{Q}\mathbf{D}_{[G_0]}\mathbf{a}_{[G_0]} = \mathbf{Q}\mathbf{D}_{[G_0']}\mathbf{a}'_{[G_0']}. \tag{A.2.16}$$

To proceed, we separate G_0' into two parts: $B = G_0' \cap G_0$, and $\overline{B} = G_0' \setminus G_0$, so that

$G_0' = [B\ \overline{B}]$ and $\mathbf{D}_{[G_0']}\mathbf{a}'_{[G_0']} = \mathbf{D}_{[B]}\mathbf{a}'_{[B]} + \mathbf{D}_{[\overline{B}]}\mathbf{a}'_{[\overline{B}]}$. We can now rewrite (A.2.16) as

$$\mathbf{a}_{[S_0]} = \mathbf{Q}\mathbf{D}_{[B]}\mathbf{a}'_{[B]} + \mathbf{Q}\mathbf{D}_{[\overline{B}]}\mathbf{a}'_{[\overline{B}]}, \tag{A.2.17}$$

and use the triangle inequality to obtain

$$\psi_{\mathscr{G}}(\mathbf{a}_{[S_0]}) \leq \psi_{\mathscr{G}}(\mathbf{Q}\mathbf{D}_{[B]}\mathbf{a}'_{[B]}) + \psi_{\mathscr{G}}(\mathbf{Q}\mathbf{D}_{[\overline{B}]}\mathbf{a}'_{[\overline{B}]}). \tag{A.2.18}$$

We now analyze the two terms in the right hand side of (A.2.18) using [28, Lemma 3]:

Lemma 1. Let $\mathbf{v} \in \mathbb{R}^p$ be a vector, $\mathbf{Z} \in \mathbb{R}^{m\times p}$ be a matrix, $\mathscr{F}$ be a partition of $\Omega = \{1, 2, \ldots, p\}$, and $\mathscr{E}$ a partition of $\{1, \ldots, m\}$. We then have that, $\psi_{\mathscr{G}}(\mathbf{Z}\mathbf{v}) \leq \rho_{[\mathscr{E},\mathscr{F}]}(\mathbf{Z})\psi_{\mathscr{G}}(\mathbf{v})$.[2]

Since $\overline{B} \subset \overline{G_0}$, it follows from (A.2.13) that $\rho_{[\mathscr{S}_0,\overline{\mathscr{B}}]}(\mathbf{Q}\mathbf{D}_{[\overline{B}]}) < \alpha$ (here $\overline{\mathscr{B}}$ is the set of the blocks that comprise $\overline{B}$). To analyze $\rho_{[\mathscr{S}_0,\mathscr{B}]}(\mathbf{Q}\mathbf{D}_{[B]})$, we use its definition,

$$\rho_{[\mathscr{S}_0,\mathscr{B}]}(\mathbf{Q}\mathbf{D}_{[B]}) = \max_{F\in\mathscr{B}} \sum_{E\in\mathscr{S}_0} \rho((\mathbf{Q}\mathbf{D})_{[E,F]}) = \max_{F\in\mathscr{B}} \sum_{S_j : j=1,\ldots,k} \rho((\mathbf{Q}\mathbf{D})_{[S_j,F]}), \tag{A.2.19}$$

and analyze each of its terms. By definition of $\mathscr{B}$, each $F \in \mathscr{B}$ corresponds to some $G_i = [S_i\ T_i]$ for some $i \leq k$. We can thus write $(\mathbf{Q}\mathbf{D})_{[S_j,F]} = [\,(\mathbf{Q}\mathbf{D})_{[S_j,S_i]}\ (\mathbf{Q}\mathbf{D})_{[S_j,T_i]}\,]$. Then, by recalling that $\mathbf{Q}\mathbf{D}_{[T_0]} = \mathbf{0}$ we see that $(\mathbf{Q}\mathbf{D})_{[S_j,T_i]} = \mathbf{0}$ for all i, j. Now, when $i = j$ we have $(\mathbf{Q}\mathbf{D})_{[S_j,S_i]} = \mathbf{I}$, thus $\rho((\mathbf{Q}\mathbf{D})_{[S_j,F]}) = \rho([\,\mathbf{I}\ \mathbf{0}\,]) = 1$. When $i \neq j$, $(\mathbf{Q}\mathbf{D})_{[S_j,S_i]} = \mathbf{0}$, and $\rho((\mathbf{Q}\mathbf{D})_{[S_j,F]}) = \rho([\,\mathbf{0}\ \mathbf{0}\,]) = 0$ in that case. From (A.2.19) we conclude that $\rho_{[\mathscr{S}_0,\mathscr{B}]}(\mathbf{Q}\mathbf{D}_{[B]}) = 1$. Plugging into (A.2.18) leads to

$$\psi_{\mathscr{G}}(\mathbf{a}) < \psi_{\mathscr{G}}(\mathbf{a}'_{[B]}) + \alpha\psi_{\mathscr{G}}(\mathbf{a}'_{[\overline{B}]}). \tag{A.2.20}$$

For the ℓ_1 term, we follow the same path as (A.2.16) and (A.2.17), now using the

[2] Note that the statement of Lemma 1 as shown here is actually a slight generalization of [28, Lemma 3], where the groups in the partitions need not have the same size.

Moore-Penrose pseudo-inverse $\mathbf{H}$ instead, yielding $\mathbf{a}_{[S_0]} = \mathbf{HD}_{[B]}\mathbf{a}'_{[B]} + \mathbf{HD}_{[\overline{B}]}\mathbf{a}'_{[\overline{B}]}$, from which $\|\mathbf{a}\|_1 \leq \left\|\mathbf{HD}_{[B]}\mathbf{a}'_{[B]}\right\|_1 + \left\|\mathbf{HD}_{[\overline{B}]}\mathbf{a}'_{[\overline{B}]}\right\|_1$ follows. Using the fact that $\|\mathbf{Wv}\|_{1,1} \leq \|\mathbf{W}\|_{1,1}\|\mathbf{v}\|_1$ [110], we get $\|\mathbf{a}\|_1 \leq \left\|\mathbf{HD}_{[B]}\right\|_{1,1}\left\|\mathbf{a}'_{[B]}\right\|_1 + \left\|\mathbf{HD}_{[\overline{B}]}\right\|_{1,1}\left\|\mathbf{a}'_{[\overline{B}]}\right\|_1$. Now, since $B \subset G_0$, and $\left\|\mathbf{HD}_{[G_0]}\right\|_{1,1} = 1$, we have that $\left\|\mathbf{HD}_{[B]}\right\|_{1,1} \leq 1$. Together with condition (A.2.15) this yields,

$$\|\mathbf{a}\|_1 < \left\|\mathbf{a}'_{[B]}\right\|_1 + \gamma\left\|\mathbf{a}'_{[\overline{B}]}\right\|_1. \tag{A.2.21}$$

Combining (A.2.20) and (A.2.21) into the HiLasso cost function we get

$$\lambda\psi_{\mathscr{g}}(\mathbf{a}) + (1-\lambda)\|\mathbf{a}\|_1 \;<\; \lambda\left[\,\psi_{\mathscr{g}}(\mathbf{a}'_{[B]}) + \alpha\psi_{\mathscr{g}}(\mathbf{a}'_{[\overline{B}]})\,\right] + (1-\lambda)\left[\,\left\|\mathbf{a}'_{[B]}\right\|_1 + \gamma\left\|\mathbf{a}'_{[\overline{B}]}\right\|_1\,\right]. \tag{A.2.22}$$

Now, to finish the proof, we need to bound the right hand side of (A.2.22) by $\lambda\psi_{\mathscr{g}}(\mathbf{a}') + (1-\lambda)\left\|\mathbf{a}'\right\|_1$, in order to show that the alternate $\mathbf{a}'$ is not a minimum of the HiLasso problem. For any γ satisfying

$$\gamma \leq 1 + \frac{\lambda(1-\alpha)\psi_{\mathscr{g}}(\mathbf{a}'_{[\overline{B}]})}{(1-\lambda)\left\|\mathbf{a}'_{[\overline{B}]}\right\|_1},$$

we have,

$$\lambda[\,\psi_{\mathscr{g}}(\mathbf{a}'_{[B]}) + \alpha\psi_{\mathscr{g}}(\mathbf{a}'_{[\overline{B}]})\,] + (1-\lambda)[\,\left\|\mathbf{a}'_{[B]}\right\|_1 + \gamma\left\|\mathbf{a}'_{[\overline{B}]}\right\|_1\,] \leq \lambda\psi_{\mathscr{g}}(\mathbf{a}') + (1-\lambda)\left\|\mathbf{a}'\right\|_1, \tag{A.2.23}$$

where we have used the fact that $\left\|\mathbf{a}'\right\|_1 = \left\|\mathbf{a}'_{[B]}\right\|_1 + \left\|\mathbf{a}'_{[\overline{B}]}\right\|_1$ and $\psi_{\mathscr{g}}(\mathbf{a}') = \psi_{\mathscr{g}}(\mathbf{a}'_{[B]}) + \psi_{\mathscr{g}}(\mathbf{a}'_{[\overline{B}]})$. To obtain a signal independent relationship between γ and α, we bound $\psi_{\mathscr{g}}(\mathbf{a}'_{[\overline{B}]})$ in terms of $\left\|\mathbf{a}'_{[\overline{B}]}\right\|_1$,

$$\left\|\mathbf{a}'_{[\overline{B}]}\right\|_1 = \sum_i \left\|\mathbf{a}'_{[\overline{R}_i]}\right\|_1 \leq \sum_i \sqrt{g}\left\|\mathbf{a}'_{[\overline{B}_i]}\right\|_2 = \sqrt{g}\psi_g(\mathbf{a}'_{[\overline{B}]}),$$

resulting in the condition

$$\gamma \leq 1 + \frac{\lambda(1-\alpha)}{(1-\lambda)\sqrt{g}} \leq 1 + \frac{\lambda(1-\alpha)\psi_{\mathscr{G}}(\mathbf{a}'_{[\overline{B}]})}{(1-\lambda)\left\|\mathbf{a}'_{[\overline{B}]}\right\|_1},$$

which completes the proof. ■

We conclude that we can guarantee recovery for every choice of λ as long as (A.2.13)–(A.2.15) are satisfied. Note that when $\lambda = 0$ (Lasso mode) we get $\gamma \leq 1$, and, as expected, (A.2.14)–(A.2.15) reduce to the Lasso recovery condition. Also, if $\alpha = 1$ we have $\gamma \leq 1$, meaning that we must tighten the constrains related to the ℓ_2 part of the cost function in order to relax the ℓ_1 part. For $\gamma > 1$, the HiLasso conditions are a relaxation of the Lasso conditions, thus allowing for more signals to be correctly recovered.

Theorem 2 below provides signal independent replacements of the conditions (A.2.13)–(A.2.15). The signal independent bound for (A.2.13) derived here, depends on coherence measures between the dictionary $\mathbf{D}$ and its image under the projection $\mathbf{I}-\mathbf{P}$, $\mathbf{C} = (\mathbf{I}-\mathbf{P})\mathbf{D}$. Since $\mathbf{P}$ depends on S_0, $\mathbf{C}$ itself is signal dependent. Thus, we need to maximize also over all possible sets S_0. These are defined as follows,

$$\nu_P := \max\left\{\max\left\{\max\left\{\frac{\mathbf{d}_i^\intercal \mathbf{c}_j}{(\mathbf{d}_i^\intercal \mathbf{c}_i)^{1/2}(\mathbf{d}_j^\intercal \mathbf{c}_j)^{1/2}},\, i,j \in G, i \neq j\right\},\, G \in \mathscr{G}\right\},\, S_0\right\}, \tag{A.2.24}$$

$$\mu_P^s := \max\left\{\max\left\{\frac{1}{g}\rho^s(\mathbf{D}_{[G]}^\intercal \mathbf{C}_{[F]}),\, G,F \in \mathscr{G}, G \neq F\right\},\, S_0\right\}, \tag{A.2.25}$$

$$\mu_P^{ss} := \max\left\{\max\left\{\frac{1}{g}\rho^{ss}(\mathbf{D}_{[G]}^\intercal \mathbf{C}_{[F]}),\, G,F \in \mathscr{G}, G \neq F\right\},\, S_0\right\}, \tag{A.2.26}$$

$$\zeta := \max\left\{\max\{(\mathbf{d}_i^\intercal \mathbf{c}_i)^{-1/2} : i = 1,\ldots,p\},\, S_0\right\}. \tag{A.2.27}$$

We are now in position to state the theorem.

Theorem 2. Let χ, ν_P, μ_P^s, μ_P^{ss} and ζ be the coherence measures defined respectively in

(A.1.2) and (A.2.24)–(A.2.27). Then the conditions (A.2.13)–(A.2.15) are satisfied if

$$\frac{\zeta^2 k g \mu_P^s}{1-(s-1)\nu_P + g\mu_P^{ss}(k-1)\zeta^2} \leq \alpha, \tag{A.2.28}$$

$$\frac{ks\chi}{1-(s-1)\nu-(k-1)s\chi} < \gamma, \tag{A.2.29}$$

$$\frac{ks\nu}{1-(s-1)\nu-(k-1)s\chi} < 1. \tag{A.2.30}$$

We also require the denominators in (A.2.28)–(A.2.30) to be positive. Note that, although the interpretation of (A.2.28) is rather counter-intuitive, it is easy to check that $\mu_P^s, \mu_P^{ss} \leq \mu_B$. This can be seen when $s = g$ (a case included in our theorems), in which case $\mathbf{P} = \mathbf{0}$, $\mathbf{C} = \mathbf{D}$, and $\mu_P^s = \mu_P^{ss} = \mu_B$. Therefore, the condition (A.2.28) is a relaxation of the standard (dense) block-sparse recovery one [28, Theorem 2].

Proof: Recall that $\mathbf{QD}_{[\overline{G_0}]} = \left(\mathbf{D}_{[S_0]}^\intercal \mathbf{C}_{[S_0]}\right)^{-1} \mathbf{D}_{[S_0]}^\intercal \mathbf{C}_{[\overline{G}_0]}$. Since $\rho_{[\cdot,\cdot]}(\cdot)$ is submultiplicative, [28],[3]

$$\rho_{[\mathscr{S}_0,\overline{\mathscr{G}}_0]}(\mathbf{QD}_{[\overline{G_0}]}) \leq \rho_{[\mathscr{S}_0,\mathscr{S}_0]}((\mathbf{D}_{[S_0]}^\intercal \mathbf{C}_{[S_0]})^{-1})\, \rho_{[\mathscr{S}_0,\overline{\mathscr{G}}_0]}(\mathbf{D}_{[S_0]}^\intercal \mathbf{C}_{[\overline{G}_0]}). \tag{A.2.31}$$

Applying the definitions of $\rho_{[\mathscr{S}_0,\overline{\mathscr{G}}_0]}$ and μ_P^s we have,

$$\rho_{[\mathscr{S}_0,\overline{\mathscr{G}}_0]}(\mathbf{D}_{[S_0]}^\intercal \mathbf{C}_{[\overline{G}_0]}) = \max_{F\in\overline{\mathscr{G}}_0} \sum_{E\in\mathscr{S}_0} \rho(\mathbf{D}_{[E]}^\intercal \mathbf{C}_{[F]}) \leq k \max_{F\in\overline{\mathscr{G}}_0} \max_{E\in\mathscr{S}_0} \{\rho(\mathbf{D}_{[E]}^\intercal \mathbf{C}_{[F]})\} \leq k g \mu_P^s, \tag{A.2.32}$$

where the last inequality in (A.2.32) derives from (A.2.25) and the fact that each $E \in \mathscr{S}_0$ belongs to some G_i, and $|E| = s$, thus playing the role of the set T in the definition of $\rho^s(\cdot)$. Our goal is now to obtain a bound for $\rho_{[\mathscr{S}_0,\mathscr{S}_0]}((\mathbf{D}_{[S_0]}^\intercal \mathbf{C}_{[S_0]})^{-1})$. To this end, we define

[3] There is a slight abuse of notation here, in that, in our case of non-square blocks, each norm $\rho_{[\cdot,\cdot]}(\cdot)$ in the right hand of the submultiplicativity inequality (A.2.31) is actually a different norm. However, it is easy to see that the referred inequality holds in this case as well.

$\mathbf{Z} = \mathbf{D}_{[S_0]}^{\intercal}\mathbf{C}_{[S_0]}$, and rewrite it as $\mathbf{Z} = \Lambda^{-1}(\mathbf{I} - (\mathbf{I} - \Lambda\mathbf{Z}\Lambda))\Lambda^{-1}$. Here Λ is a $ks \times ks$ block-diagonal scaling matrix to be defined later. Assume for now that $\rho_{[\mathscr{S}_0,\mathscr{S}_0]}(\mathbf{I} - \Lambda\mathbf{Z}\Lambda) < 1$. This allows us to apply the following result from [28]:

Lemma 2. Suppose that $\rho_{[\mathscr{E},\mathscr{F}]}(\mathbf{W}) < 1$. Then $(\mathbf{I}+\mathbf{W})^{-1} = \sum_{k=0}^{\infty}(-\mathbf{W})^k$.

By applying Lemma 2 to $\mathbf{W} = -\mathbf{I} + \Lambda\mathbf{Z}\Lambda$ we can write $\mathbf{Z}^{-1} = \Lambda\left[\sum_{i=0}^{\infty}(\mathbf{I} - \Lambda\mathbf{Z}\Lambda)^i\right]\Lambda$. With this,

$$\begin{aligned}\rho_{[\mathscr{S}_0,\mathscr{S}_0]}(\mathbf{Z}^{-1}) &\overset{(a)}{\leq} \left[\rho_{[\mathscr{S}_0,\mathscr{S}_0]}(\Lambda)\right]^2 \rho_{[\mathscr{S}_0,\mathscr{S}_0]}\left(\sum_{i=0}^{\infty}(\mathbf{I} - \Lambda\mathbf{Z}\Lambda)^i\right) \overset{(b)}{\leq} \left[\rho_{[\mathscr{S}_0,\mathscr{S}_0]}(\Lambda)\right]^2 \sum_{i=0}^{\infty}\rho_{[\mathscr{S}_0,\mathscr{S}_0]}\left((\mathbf{I} - \Lambda\mathbf{Z}\Lambda)^i\right) \\ &\overset{(c)}{\leq} \left[\rho_{[\mathscr{S}_0,\mathscr{S}_0]}(\Lambda)\right]^2 \sum_{i=0}^{\infty}\left(\rho_{[\mathscr{S}_0,\mathscr{S}_0]}(\mathbf{I} - \Lambda\mathbf{Z}\Lambda)\right)^i \overset{(d)}{\leq} \frac{[\rho_{[\mathscr{S}_0,\mathscr{S}_0]}(\Lambda)]^2}{1 - \rho_{[\mathscr{S}_0,\mathscr{S}_0]}(\mathbf{I} - \Lambda\mathbf{Z}\Lambda)},\end{aligned} \tag{A.2.33}$$

where in (a) and (c) we applied the submultiplicativity of $\rho_{[\cdot,\cdot]}(\cdot)$, (b) is a consequence if the triangle inequality, and (d) is the limit of the geometric series, which is finite when $\rho_{[\mathscr{S}_0,\mathscr{S}_0]}(\mathbf{I} - \Lambda\mathbf{Z}\Lambda) < 1$.

We now bound $\rho_{[\mathscr{S}_0,\mathscr{S}_0]}(\mathbf{I} - \Lambda\mathbf{Z}\Lambda)$. First, note that, since Λ is block-diagonal, we have that $(\mathbf{I} - \Lambda\mathbf{Z}\Lambda)_{[S_i,S_j]} = \mathbf{I}_{[S_i,S_j]} - \Lambda_{[S_i,S_i]}\mathbf{Z}_{[S_i,S_j]}\Lambda_{[S_j,S_j]}$. We then choose Λ to be a diagonal matrix with $\Lambda_{ii} = (\mathbf{d}_i^{\intercal}\mathbf{c}_i)^{-1/2}, i \in S_0$. With this choice, we have that the diagonal elements of $\mathbf{I}_{[S_j,S_j]} - \Lambda_{[S_j,S_j]}\mathbf{Z}_{[S_j,S_j]}\Lambda_{[S_j,S_j]}$ are equal to 1 for all j, and the off-diagonal elements are bounded by ν_P. Using Geršgorin Theorem we then have that

$$\rho(\mathbf{I}_{[S_j,S_j]} - \Lambda_{[S_j,S_j]}\mathbf{Z}_{[S_j,S_j]}\Lambda_{[S_j,S_j]}) \leq (s-1)\nu_P,\ j = 1,\ldots,k. \tag{A.2.34}$$

As for the off-diagonal $s \times s$ blocks of $\mathbf{I} - \Lambda\mathbf{Z}\Lambda$, we have $(\mathbf{I} - \Lambda\mathbf{Z}\Lambda)_{[S_i,S_j]} = -\Lambda_{[S_i,S_i]}\mathbf{Z}_{[S_i,S_j]}\Lambda_{[S_j,S_j]}$. We then have

$$\rho((\mathbf{I} - \Lambda\mathbf{Z}\Lambda)_{[S_i,S_j]}) \overset{(a)}{\leq} \rho(\Lambda_{[S_i,S_i]})\rho(\mathbf{Z}_{[S_i,S_j]})\rho(\Lambda_{[S_j,S_j]}) \overset{(b)}{\leq} \zeta(g\mu_P^{ss})\zeta, \tag{A.2.35}$$

where in (a) we used the submultiplicativity of $\rho(\cdot)$, and (b) derives from the definition of μ_P^{ss}, and the fact that, with our choice of Λ we have $\rho(\Lambda_{[S_i,S_i]}) \leq \zeta$ for all i. Now we can write the definition of $\rho_{[\mathscr{S}_0,\mathscr{S}_0]}(\mathbf{I}-\Lambda\mathbf{Z}\Lambda)$ and bound its summation using (A.2.34)–(A.2.35),

$$\rho_{[\mathscr{S}_0,\mathscr{S}_0]}(\mathbf{I}-\Lambda\mathbf{Z}\Lambda) \leq \max_{S_j:j\leq k} \Big\{ \rho(\mathbf{I}_{[S_j,S_j]} - \Lambda_{[S_j,S_j]}\mathbf{Z}_{[S_j,S_j]}\Lambda_{[S_j,S_j]}) + \dots \\ \dots \sum_{S_i:i\leq k, i\neq j} \rho\left(\Lambda_{[S_i,S_i]}(\mathbf{I}-\Lambda\mathbf{Z}\Lambda)_{[S_i,S_j]}\Lambda_{[S_j,S_j]}\right) \Big\} \leq (s-1)\nu_P + g\mu_P^{ss}\zeta^2. \tag{A.2.36}$$

By our choice of Λ, $\rho(\Lambda_{[S_i,S_i]}) \leq \zeta$ and $\rho(\Lambda_{[S_i,S_j]}) = 0$ for $i \neq j$. Therefore $\rho_{[\mathscr{S}_0,\mathscr{S}_0]}(\Lambda) \leq \zeta$ as well. Using this together with (A.2.36) in (A.2.33), we obtain

$$\rho_{[\mathscr{S}_0,\mathscr{S}_0]}(\mathbf{Z}^{-1}) \leq \frac{\zeta^2}{1-(s-1)\nu_P + g\mu_P^{ss}(k-1)\zeta^2}. \tag{A.2.37}$$

To ensure that $\rho_{[\mathscr{S}_0,\mathscr{S}_0]}(\mathbf{I}-\Lambda\mathbf{Z}\Lambda) < 1$, we need the denominator in the above equation to be positive. Now (A.2.28) follows by plugging (A.2.32) and (A.2.37) into (A.2.31),

$$\rho_{[\mathscr{S}_0,\overline{\mathscr{G}}_0]}(\mathbf{Q}\mathbf{D}_{[\overline{G}_0]}) \leq \frac{\zeta^2 k g \mu_P^s}{1-(s-1)\nu_P + g\mu_P^{ss}(k-1)\zeta^2}.$$

Finally, we use the same ideas to bound $\|\mathbf{H}\mathbf{D}_{[\overline{G}_0]}\|_{1,1}$ and derive (A.2.29). Specifically,

$$\|\mathbf{H}\mathbf{D}_{[\overline{G}_0]}\|_{1,1} \leq \|(\mathbf{D}_{[S_0]}^\intercal \mathbf{D}_{[S_0]})^{-1}\|_{1,1} \|\mathbf{D}_{[S_0]}^\intercal \mathbf{D}_{[\overline{G}_0]}\|_{1,1}. \tag{A.2.38}$$

Now

$$\|(\mathbf{D}_{[S_0]})^\intercal \mathbf{D}_{[\overline{G}_0]}\|_{1,1} = \max_{j\in\overline{G}_0} \sum_{i\in S_0} |\mathbf{d}_i^\intercal \mathbf{d}_j| \overset{(a)}{\leq} ks\chi, \tag{A.2.39}$$

where (a) follows from the definition of χ and the fact that $|S_0| = ks$. It remains to develop

a bound on $\|(\mathbf{D}_{[S_0]}^\intercal \mathbf{D}_{[S_0]})^{-1}\|_{1,1}$. To this end we express $\mathbf{D}_{[S_0]}^\intercal \mathbf{D}_{[S_0]} = \mathbf{I} + \mathbf{W}$, and bound

$$\|\mathbf{W}\|_{1,1} = \max_{r \le k} \left\{ \max_{i \in S_r} \left\{ \sum_{j \in S_r, j \neq i} |\mathbf{d}_i^\intercal \mathbf{d}_j| + \sum_{j \in S_0 \setminus S_r} |\mathbf{d}_i^\intercal \mathbf{d}_j| \right\} \right\} \le (s-1)\nu + s(k-1)\chi. \quad \text{(A.2.40)}$$

since for all $S_r, r \le k$, and all $i \in S_r$, the first sum has $(s-1)$ nonzero elements bounded by ν, and the second sum has $s(k-1)$ elements bounded by χ. Now, by requiring $(s-1)\nu + s(k-1)\chi < 1$ we can apply Lemma 2 to $\mathbf{W}$ and follow the same path as the one that leads to (A.2.36), now using the matrix norm properties of $\|\cdot\|_{1,1}$, to obtain,

$$\|(\mathbf{D}_{[S_0]}^\intercal \mathbf{D}_{[S_0]})^{-1}\|_{1,1} \le \frac{1}{1 - (s-1)\nu + s(k-1)\chi}. \quad \text{(A.2.41)}$$

Again, $(s-1)\nu + s(k-1)\chi < 1$ is implicit in the requirement that the above denominator be positive. Plugging (A.2.41) and (A.2.39) into (A.2.38) yields (A.2.29).

The proof for (A.2.30) is analogous to that of (A.2.29), only that now the upper bound on $|\mathbf{d}_i^\intercal \mathbf{d}_j|, i \in S_0, j \in T_0$, is $\nu \le \mu$. Continuing as before leads to (A.2.30). ■

Theorems 1 and 2 are for the non-collaborative case. For the collaborative case there exist results that show that both the C-Lasso [30] and C-GLasso [7] will recover the true shared active set with a probability of error that vanishes exponentially with N. Since the in-group active sets are not necessarily equal for all samples in $\mathbf{X}$, C-HiLasso could only help in recovering the group sparsity pattern. Since the C-GLasso is a special case of C-HiLasso when $\lambda_1 = 0$, we can conjecture that when $\lambda_1 > 0$, the accuracy of the C-HiLasso in recovering the correct groups will improve with larger N. Furthermore, since our results for HiLasso improve on those of the Group Lasso, it is to be expected that the accuracy of C-HiLasso, for an appropriate $\lambda_1 > 0$, will be better than that of C-GLasso.

As an intuitive explanation to why this may happen, the proofs in [30] and [7] assume

a continuous probability distribution on the non-zero coefficients of the signals, and give recovery results for the average case. On the other hand, the in-group sparsity assumption of C-HiLasso implies that only s out of g samples will be nonzero within each group. This implies that, for the same group sparsity pattern, there will be much less (exactly a fraction s/g) non-zero elements in the possible signals compared to the ones that can occur under the hypothesis of C-GLasso. Since any assumed distribution of the signals under the in-group sparsity hypothesis has to be concentrated on this much smaller set of possible signals, they should be easier to recover correctly from solutions to the C-HiLasso program, compared to the dense group case of C-GLasso.