POLITECNICO
DI TORINO

POLITECNICO DI TORINO

SCUOLA DI DOTTORATO

Dottorato in Ingegneria elettronica e delle telecomunicazioni – XXVIII ciclo

Tesi di Dottorato

Autoregressive process parameters estimation from Compressed Sensing measurements and Bayesian dictionary learning

Matteo Testa

Tutore
prof. Enrico Magli

Coordinatore del corso di dottorato
prof. Ivo Montrosset

Maggio 2016

Summary

The main contribution of this thesis is the introduction of new techniques which allow to perform signal processing operations on signals represented by means of compressed sensing. Exploiting autoregressive modeling of the original signal, we obtain a compact yet representative description of the signal which can be estimated directly in the compressed domain. This is the key concept on which the applications we introduce rely on.

In fact, thanks to proposed the framework it is possible to gain information about the original signal given compressed sensing measurements. This is done by means of autoregressive modeling which can be used to describe a signal through a small number of parameters. We develop a method to estimate these parameters given the compressed measurements by using an ad-hoc sensing matrix design and two different coupled estimators that can be used in different scenarios. This enables centralized and distributed estimation of the covariance matrix of a process given the compressed sensing measurements in a efficient way at low communication cost. Next, we use the characterization of the original signal done by means of few autoregressive parameters to improve compressive imaging. In particular, we use these parameters as a proxy to estimate the complexity of a block of a given image. This allows us to introduce a novel compressive imaging system in which the number of allocated measurements is adapted for each block depending on its complexity, *i.e.,* spatial smoothness. The result is that a careful allocation of the measurements, improves the recovery process by reaching higher recovery quality at the same compression ratio in comparison to state-of-the-art compressive image recovery techniques. Interestingly, the parameters we are able to estimate directly in the compressed domain not only can improve the recovery but can also be used as feature vectors for classification. In fact, we also propose to use these parameters as more general feature vectors which allow to perform classification in the compressed domain. Remarkably, this method reaches high classification performance which is comparable with that obtained in the original domain, but with a lower cost in terms of dataset storage.

In the second part of this work, we focus on sparse representations. In fact,

a better sparsifying dictionary can improve the Compressed Sensing recovery performance. At first, we focus on the original domain and hence no dimensionality reduction by means of Compressed Sensing is considered. In particular, we develop a Bayesian technique which, in a fully automated fashion, performs dictionary learning. More in detail, using the uncertainties coming from atoms selection in the sparse representation step, this technique outperforms state-of-the-art dictionary learning techniques. Then, we also address image denoising and inpainting tasks using the aforementioned technique with excellent results. Next, we move to the compressed domain where a better dictionary is expected to provide improved recovery. We show how the Bayesian dictionary learning model can be adapted to the compressive case and the necessary assumptions that must be made when considering random projections. Lastly, numerical experiments confirm the superiority of this technique when compared to other compressive dictionary learning techniques.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Compressed Sensing (CS) [9] [10] is a well-established paradigm in which signal acquisition and compression become a single operation. Being able to sample a signal well below the Nyquist rate has made CS a popular approach over the last few years. The key concept which CS relies on, is that the acquired signal must be sparse in some domain. This assumption allows to provide theoretical guarantees under which the signal can be exactly recovered with overwhelming probability. Nevertheless, there are still some aspects of the CS which can be investigated and that can be improved. The aspects we address in this thesis are two. The first one is related to the fact most of the CS literature addresses the scenario where first a signal is acquired using CS, and then the signal is recovered using a nonlinear algorithm. However, since the CS measurements still contain much of the information of the original signal it would be of great interest to be able to extract some parameters directly in the compressed domain. These parameters can then be used directly for specific applications or used in the recovery process to improve its performance.

The second problem we address in this work is the dictionary learning problem. This problem aims to find the best dictionary which is able to sparsify a given set of signals denoted as training signals. This problem has great importance and covers a lot of applications in the signal processing field. As an example, image processing tasks such as inpainting and denoising among others, are efficiently solved by means of this technique. Images are an extremely good example of a case where the dictionary learning problem is perfectly suited. This is due to the fact that images tends to be compressible rather than sparse in standard basis such as wavelet or DCT. Hence, state-of-the-art techniques which use the sparsity as the main tool to perform image processing tasks take great advantage by using an improved sparse representation. Moreover, the dictionary learning problem can also be applied to CS measurements. In this case the main advantage of the technique is that by using a better dictionary specifically suited for the data considered, the recovery process which seeks for the sparsest solution is improved.

In the next sections we describe more in detail these two problems.

1.1 Compressive signal processing

As previously discussed, CS shifts the computational cost to the recovery stage. While in most cases this is an appreciated feature since it allows to shift the computational burden out of the acquisition process, in other cases the recovery process may still be too computational complex or not needed. In these scenarios, one wishes to infer some information about the signal that has been acquired, e.g. estimate one or more parameters that describe the signal in order to perform some detection or classification task. In CS, because of the complexity of the signal recovery process, it is inconvenient to first recover the signal and then estimate its parameters. Rather, it is much more desirable to perform estimation directly on the compressed measurements. Moreover, if the signal is not exactly sparse, the error introduced by the recovery process may be propagated through signal processing tasks and decrease the overall performance. Hence, the idea behind compressive signal processing is that of using the compressed measurements as a direct source of information bypassing the recovery process keeping in mind that most of the information of the original signal is still preserved during the CS process.

In their seminal work on compressive signal processing [11], the authors laid the foundations of the three main building blocks of signal processing. More in detail inference, estimation and filtering operations were derived in the compressed domain. Even though operations in the compressed domain lead to suboptimal results, they showed that a with a fairly small number of random projections is possible to achieve very good accuracy.

In fact, a lot of applications can take advantage by the compressive signal processing technique. Applications in which the signal is compressively acquired (by means of *e.g.,* CS-ADC such as [12]) and only some information about the signal is needed are those in which this paradigm lead to the biggest advantages. As an example let us consider the frequency estimation in wideband signals. This kind of signals are efficiently tackled with the CS acquisition scheme since it allows sub-Nyquist sampling avoiding the need of expensive high frequency ADC components. At this point, standard CS schemes would imply the signal recovery first and the frequency estimation later. In this case it is fair to assume the signal to be sparse in the frequency domain, however if the sparsifying basis (*e.g.,* DFT basis) does not contains the exact frequencies of the sinusoidal components of the signal, the recovered signal will be compressible but not exactly sparse. Hence, estimating the frequency on such a signal is likely to lead to estimation errors. Moreover, this procedure has higher computational costs requiring both the recovery and the estimation processes. In this case being able to directly infer the frequency in the compressed domain can

reduce the overall computational cost and improve the estimate accuracy.

Besides analyzing the signal itself, knowledge of such signal parameters may also be useful during the reconstruction stage. For example, in many CS applications, no knowledge about the nature of the compressed signal is available, except for the assumption that it is sparse in some domain. However, natural signals are typically not exactly sparse, but rather approximately sparse or "compressible". In many cases, such compressibility implies that the signal spectrum is decreasing [13]. This kind of information on the signal structure has indeed been used to improve the CS reconstruction [14].

To briefly summarize, performing operations in the compressed domain not only allows to directly extract useful information from the randomly projected signal, but in the cases in which is needed, it can also improve the recovery by providing additional information to the process. Following this idea, the main contribution of this thesis is a new approach which allows to model the original signal with few parameters that can be efficiently estimated in the compressed domain. In particular, the main achievements obtained in this field can be summarized as follows:

- *Compressive covariance estimation* Compressive estimation of the covariance matrix by means of few parameters allows in distributed and centralized settings to perfom signal processing operations in the compressed domain efficiently and at low communication cost.

- *Adaptive compressive imaging* Estimating the complexity of an image given the compressed measurements allows to adaptively sense an image allocating the correct number of measurements thus reaching higher recovery quality at the same compression ratio.

- *Compressive classification* Extracting features vectors for classification directly in the compressed domain allows to obtain excellent classification performance whilst keeping storage costs low.

1.2 Dictionary learning

Let us consider the role of the dictionaries for sparse representations before discussing the problem related to learning the best dictionaries given a set of training signals: the dictionary learning problem.

Representing a signal requires the selection of a dictionary, i.e., a set of "atoms" or vectors in the signal space, a linear combination of which represents the given signal (alternative representations based on the use of manifolds [15, 16] are also relevant but will not be discussed here). The obvious and simplest choice of a dictionary is a basis, the smallest possible dictionary with the capability of representing the whole

signal space. Simple as they are, the scarce expressiveness of such dictionaries led to the ongoing development of overcomplete dictionaries [17].

The transition to overcomplete dictionaries was gradual. Analytical complete dictionaries were introduced first, which made use of different transforms such as DCT, Wavelet or Gabor. The limitations of such transforms were soon brought to light. Indeed, the work in [18] pointed out the deficiencies of the popular orthogonal wavelet transforms, namely its sensitivity to translation, dilation and rotation, resulting in the development of the Steerable Wavelet Transform. Early approaches towards overcomplete dictionaries tried to preserve the favorable orthogonality properties of bases but soon proved to be insufficient.

Parallel work suggested the use of collections of data to better describe signals, rather than the use of mathematical synthetic functions. The works in [19] and [20] were very influential towards the recent advances in dictionary learning and sparse signal representation. These advances eventually led to the development of the K-SVD technique [2] which, employing a greedy iterative approach by alternating between the sparse coding and dictionary update steps, achieves state-of-the-art performance. However, the main drawback of this technique is that it typically requires the knowledge of the sparsity of the training signals and the variance of the noise which corrupts the data.

In this work we propose a novel Bayesian dictionary learning technique which not only aims to solve the problems related to K-SVD but also achieves better performance. The main idea is that of approaching the problem from a Bayesian point of view. This allows us to take into account all the uncertainties, which are due to the selection of the atoms in the dictionary, and hence to improve the overall process. Moreover, by correctly modeling all the considered parameters, they do not need to be known in advance since all the quantities including sparsity and noise variance are automatically estimated. Numerical experiments confirm the excellent performance achieved by this technique in both the original and the compressed domains.

Lastly, it is worth noting that since the dictionary learning technique is more general approach, it has been successfully employed to solve different signal processing tasks. In particular, the dictionary learning problem has been widely applied in image processing and machine learning. Applications include image denoising and deblurring [21, 22], image super-resolution [23], image restoration [24], classification and clustering [25], and face recognition [26] among many others.

To summarize, the main contributions in this field which we propose in this work are the following:

- *Bayesian dictionary learning* Fully automatic Bayesian dictionary learning in the original domain allows to obtain better dictionaries which can improve the performance of image processing tasks such as inpainting and denoising.

- *Compressive Bayesian dictionary learning* Being able to learn a dictionary given the compressed measurements has a lot of advantages over standard CS recovery techniques. In fact, better dictionaries lead to improved recovery when the sparsifying bases are not known or when they are not able to correctly sparsify a signal.

1.3 Thesis organization

The remainder of this work is organized as follows.

In Chapter 2 we introduce the Compressed Sensing framework. In particular we focus on the theoretical assumptions behind the problem and we give a brief overview of the recovery algorithms. Chapter 3 introduces the compressed autoregressive estimation framework. We start by introducing the autoregressive modeling which is at the core of the framework we are proposing. Additionally, we also provide some notation which will be of paramount importance in the following chapters. Next, we derive a design that includes a sensing matrix with a specific structure which operates in conjunction with the estimators. The compressive estimators are two: a more general AR(p) estimator and a specific Bayesian AR(1) estimator which shows improved performance in critical situations, *i.e.,* for highly compressed signals. We also show two applications which directly arise from the frameworks and which are used as a test for the estimation accuracy of the compressed estimators.

Chapter 4 describes a novel adaptive compressive imaging system. To the best of our knowledge, this is the first adaptive scheme for CS image recovery which completely operates in the compressed domain. Remarkably, adapting the measurements for each block of which the image is composed based on the estimated complexity through the AR estimation framework, allow us to reach up to 6dB PSNR gains with respect to standard CS techniques.

Chapters 5 and 6 describe two compressive signal processing operations enabled by the proposed AR framework. In particular, in Chapter 5 we focus on the compressive covariance estimation problem which recently attracted a lot of attention within the CS community. Then, in Chapter 6 we show how we can use the compressively estimated AR parameters to perform efficient classification in the compressed domain.

We then move to the distributed setting in Chapter 7. In fact, we consider the distributed estimation of the covariance matrix of a colored noise process affecting CS measurements. In particular, we show that the proposed method outperforms distributed covariance estimation techniques in terms of both reduced communication cost and increased accuracy. Lastly, we also show the good performance of such method inside a compressive signal processing task, namely a compressive detection task.

In Chapter 8, we shift the focus on improved sparse representations. In particular, we propose a new Bayesian dictionary learning algorithm which is able to learn the best sparsifying dictionary given a set of signals. The approach we introduce, exploiting a hierarchical Bayesian modeling is able to estimate all the parameters of the model within the learning process. Since this approach achieves excellent performance finding good dictionaries and hence improving the sparse representation, we include this technique within the CS sensing process by adjusting the necessary parts in Chapter 9. The result is that it is possible to obtain good recovery results for signal compressed by means of CS which are not sparse in standard basis.

Lastly, in Chapter 10 we draw some conclusions and discuss possible future work.

1.4 Publications

In this section we provide a list of the publications which are the outcome of the research carried out during the PhD program.

- Accepted/Published

    1. M. Testa, E. Magli, Distributed covariance estimation for compressive signal processing, *2015 49th Asilomar Conference on Signals, Systems and Computers*, Pacific Grove, CA, 2015, pp. 676-680.
    2. M. Testa, E. Magli, Autoregressive process parameter estimation from compressed sensing measurements, *2015 49th Asilomar Conference on Signals, Systems and Computers*, Pacific Grove, CA, 2015, pp. 488-492.
    3. M. Testa, E. Magli, Compressive classification based on autoregressive features, *Communications (COMM), 2016 11th International Conference on*, Bucharest *accepted*

- Submitted

    1. M. Testa, J.G. Serra, R. Molina, A.K. Katsaggelos ,Bayesian K-SVD, *submitted at* IEEE Transactions on Image Processing
    2. M. Testa, E. Magli, Compressive estimation and imaging based on autoregressive models,*submitted at* IEEE Transactions on Image Processing

Chapter 2

Compressed Sensing

In this chapter we briefly introduce the theory underlying CS: a now well-established approach for signal acquisition that makes it possible to collapse sampling and compression in a single operation.

2.1 Introduction

Historically, in signal acquisition, to guarantee no signal distortion, the number of samples has always been limited by the bandwidth of the signal (according to the Nyquist-Shannon sampling theorem). This means that the minimum sampling rate must be at least twice the maximum frequency found in the signal itself, leading to a big number of samples. Compression and in particular transform coding frameworks emerged to address this problem. Given a signal $\mathbf{x} \in \mathbb{R}^N$ it can be expressed in terms of N orthonormal vectors $\mathbf{\Psi}_i \in \mathbb{R}^N$ that form the orthonormal basis $\mathbf{\Psi} = \begin{bmatrix} \mathbf{\Psi}_1 & \mathbf{\Psi}_2 & \dots & \mathbf{\Psi}_N \end{bmatrix}$. So $\mathbf{x}$ can be written as

$$\mathbf{x} = \sum_{i=1}^{N} s_i \mathbf{\Psi}_i \quad \mathbf{x} = \mathbf{\Psi}\mathbf{s} \quad \mathbf{s} \in \mathbb{R}^N$$

where $\mathbf{s}$ is the signal representation in the $\mathbf{\Psi}$ domain.

Most of the signals of interest, like audio and images, are *approximately sparse* or *compressible*. This means that there exists a basis $\mathbf{\Psi}$ in which the signal can be well represented with only $k \ll n$ non-zero coefficients s_i, differently from truly *sparse* signals where the k non-zero coefficients lead to an *exact* representation. This has largely been exploited in lossy *transform coding* where large signals are represented in an efficient way by only coding largest coefficients and their position. Examples of transform coding techniques are JPEG [27], JPEG2000 [28] and MP3 [29] standards.

Drawbacks of this method arise from the need of sampling the whole signal independently from its sparsity, when one may only need to extract some features from it.

As introduced in [9] and [10] CS framework emerged in analogy with the classical paradigm that relies on sampling the whole signal at first, accordingly to the Nyquist-Shannon sampling theorem, and then processing it. The key concept of CS is again signal sparsity, but instead of sampling the whole signal and then compressing it, the signal is directly downsampled according to its sparsity and then its reconstruction is guaranteed thanks to some properties that will be more in-detail explained in the following sections.

In the sensing problem we want to acquire a *linear* and *non-adaptive* reduced set of the samples of the original signal $\mathbf{x} \in \mathbb{R}^N$ but still keeping most of the information which is contained in it. In order to have an operator which can be efficiently applied to the signal, it has to be linear and it does not have to depend on the signal itself, namely it must be non-adaptive.

We denote as $\mathbf{\Phi} \in \mathbb{R}^{M,N}$ the sensing matrix and $\mathbf{\Psi} \in \mathbb{R}^{N,N}$ the matrix containing the basis in which the signal is known to be sparse: the so-called sparsifying basis or dictionary. We denote as N the the dimension of the signal and $M < N$ is the reduced dimension of the signal in the measurement domain (where the downsampling occurs). So we can write

$$\mathbf{y} = \mathbf{\Phi x} = \mathbf{\Phi \Psi s} = \mathbf{\Theta s}$$

where $\mathbf{y}$ is the *measurements vector*.

To guarantee signal recovery given $\mathbf{y}$, two important properties that are used to analyze the behavior of CS recovery techniques will be examined: *RIP* and *coherence*. Without loss of generality, from now on where no differently specified, the sparsifying basis $\mathbf{\Psi}$ will be an unitary matrix. This will allow us to consider $\mathbf{x}$ sparse in the original domain.

2.2 RIP

Let us start by imposing some conditions based on the results we want to obtain. To recover the signal, $\mathbf{\Phi}$ must preserve most of the information contained in the signal when dimensionality is reduced from N to M. Thus if a signal is k-sparse, namely it belongs to the set of k-sparse signals Σ_k, the mapping must not bring any significant distortions, *i.e.* if a signal does not belong to the nullspace of $\mathbf{\Psi}$, the signal projected by $\mathbf{\Phi}$ on $\mathbb{R}^M$ still must not belong to the nullspace of $\mathbf{\Psi}$.
More formally this can be written as:

$$\mathbf{\Phi x}_1 \neq \mathbf{\Phi x}_2 \;\; if \;\; \mathbf{x}_1 \neq \mathbf{x}_2 \;\; \forall \mathbf{x}_1, \mathbf{x}_2 \in \Sigma_K$$

where

$$\Sigma_K = \{\mathbf{x} : ||\mathbf{x}||_0 \leq k\}.$$

The *restricted isometry property* (RIP for short) is here introduced as a useful tool to determine if the sensing matrix is going to introduce any distortion in terms of ℓ_2 distance. This property, that is a more tractable and stable derivation of another useful property called *null space property* NSP, has been firstly described by Candès and Tao in [30].

Definition 2.1. The matrix $\mathbf{\Phi}$ satisfies the RIP of order k, if there exist an isometry constant $\delta_K \geq 0$ and an integer value k such that

$$(1 - \delta_k)||\mathbf{x}||_2^2 \leq ||\mathbf{\Phi x}||_2^2 \leq (1 + \delta_k)||\mathbf{x}||_{l_2}^2$$

holds for any $\mathbf{x} \in \Sigma_k$.

If we consider a sensing matrix $\mathbf{\Phi}$ which satisfies the RIP of order $2k$ and an isometry constant $\delta_{2k} \ll 1$, we can see that distances between k-sparse signals are approximately preserved:

$$(1 - \delta_{2K})||x_1 - x_2||_2^2 \leq ||\mathbf{\Phi}x_1 - \mathbf{\Phi}x_2||_2^2 \leq (1 + \delta_{2K})||x_1 - x_2||_2^2$$

A generic mapping satisfying this condition is also called *bi-Lipschitz* mapping.

The construction of such matrices is challenging due the computational complexity required by the combinatorial computation required to verify all $\binom{n}{k}$ matrices which satisfy the RIP property for the considered order.

It is also possible to construct these matrices in deterministic way [31] [32] but it is inefficient since these techniques require a value of M too large compared to the given N, e.g., in [32] the required M is $O(k^2 \log N)$.

2.3 Coherence

To avoid the previous method complexity in favor of some more computationally preferable approach, a property called *coherence* can be exploited. It takes into account both the sparsifying basis and the sensing matrix in order to assess some bounds on recovery performance.

Given the sensing basis $\mathbf{\Phi}$ and the sparsifying basis $\mathbf{\Psi}$, coherence is defined as

$$\mu(\mathbf{\Psi}, \mathbf{\Phi}) = \sqrt{N} \max_{1 \leq k,j \leq N} |\langle \mathbf{\Phi}_k, \mathbf{\Psi}_j \rangle| \tag{2.1}$$

where $\mu \in \left[1, \sqrt{N}\right]$.

The more the elements of the two bases $\mathbf{\Phi}$ and $\mathbf{\Psi}$ are correlated, the larger coherence μ. If coherence is large it means that vectors $\mathbf{\Phi}_j$ cannot sparsely represents vectors of $\mathbf{\Psi}_j$ and vice versa [10] [9]. In other words low coherence means that the columns of the two matrices are mutually orthogonal. An example of pair of basis for which incoherence is maximal ($\mu = 1$) are delta spikes as $\mathbf{\Phi}_j(t) = \delta(t-k)$ and Fourier sinusoides as $\mathbf{\Psi}_j(t) = \frac{1}{\sqrt{n}} e^{i2\pi j \frac{t}{n}}$ [33]. As an example, in [34] the authors show a set of matrices $\mathbf{A} \in \mathbb{C}^{M \times M^2}$ for which the coherence is small $\mu = \sqrt{N}/\sqrt{M}$ and RIP constant is $\delta_{2k} = C\frac{k}{\sqrt{M}}$.

The goal is to obtain very low coherence bases since, as will become more evident in Section 2.5, this property can be efficiently used to lower the required number of measurements.

2.4 Random matrices

Among all the possible methods to construct sensing matrices which can be used in CS, random ones play a very important role. In fact, whilst deterministic sensing matrix construction is computationally expensive, it has been proved that random matrices not only are easier to construct, but they also show statistical guarantees in terms of RIP and coherence.

In fact, the turning point of CS is that if $\mathbf{\Phi}$ elements are chosen to be independent and identically distributed (i.i.d) random variables with zero mean and variance $\frac{1}{M}$, with high probability the coherence of $\mathbf{\Phi}$ with any sparsifying basis $\mathbf{\Psi}$ is low [9].

Furthermore, if this kind of matrices are distributed according to Gaussian, Bernoulli or any other sub Gaussian distribution, then they will satisfy the RIP property with high probability [11].

2.4.1 RIP in random matrices

Starting from the *RIP* it is interesting to analyze it is correlation with M: the reduced number of acquired samples. The starting point as in [35] is the so-called *concentration inequality*:

$$P\left(\left|||\mathbf{Ax}||_2^2 - ||\mathbf{x}||_2^2\right| \geq \delta||\mathbf{x}||_2^2\right) \leq 2e^{-c_0\delta^2 M}, \quad 0 < \delta < 1 \tag{2.2}$$

where c_0 is some positive constant and the probability it is taken over all $M \times N$ matrices A.
In other words, it expresses the probability that the random variable $||\mathbf{Ax}||_2^2$ is strongly concentrated around its mean value $||\mathbf{x}||_2^2$.
This inequality is satisfied [36] by two classes of random matrices: the *Gaussian* and

Bernoulli distributed ones. Then, using the concentration inequality it is possible to show [35] that an estimate of the *RIP* constants is given by the following theorem.

Theorem 2.1. *Assume* $\mathbf{A} \in \mathbb{R}^{M \times N}$ *to be a random matrix satisfying the concentration property (2.2). Then there exists a constant* C *depending only on* c_0 *such that the restricted isometry constant of* $\mathbf{A}$ *satisfies* $\delta_k \leq \delta$ *with probability exceeding* $1 - \epsilon$ *provided*

$$M \geq C\delta^{-2} \left(k \log (N/M) + \log \epsilon^{-1} \right).$$

At this point if we combine these results about the *RIP* constants estimates with the recovery due to ℓ_1-minimization (discussed in detail in further sections) it is possible to prove [37] that any k-sparse signal can be stably recovered using a random matrix that satisfies (2.2) with high probability by choosing

$$M \geq Ck \log (N/k) \tag{2.3}$$

It is a very strong yet surprising statement since it means that given a suitable $M \ll N$, if the sensing matrix has randomly distributed entries (Gaussian or Bernoulli), then with high probability the signal can be exactly recovered from its measurements.

2.4.2 Coherence in random matrices

A similar result that bounds from below the number of measurements M required to achieve exact recover, can be obtained by exploiting the coherence property. Let us start with a simple model. Let us denote as Ω the random uniformly-sampled subset of dimension M of the original signal $\mathbf{x}$, as T the signal support, as $z(t)\ \forall\ t \in T$ the sign sequence valued ± 1 with probability 0.5 and as μ the coherence (2.1). Then, as stated in [38] we have

Theorem 2.2. *Let* $\mathbf{U} = \mathbf{R\Phi\Psi}$ *be an orthogonal matrix with* $|\mathbf{U}_{k,j}| \leq \mu(\mathbf{U})$. *Fix a subset* T *of the original domain. Choose a subset* Ω *of the measurements domain of size* $|\Omega| = M$, *and sign sequence* $\mathbf{z}$ *on* T *uniformly at random. Suppose that*

$$M \geq C_0 |T| \mu^2(U) \log (N/\delta)$$

and also $M \geq C_0' \log (N/\delta)^2$ *for some fixed numerical constants* C_0 *and* C_0'. *Then with probability exceeding* $1-\delta$, *every signal* $\mathbf{x}$ *in the support of* T *with signs matching* $\mathbf{z}$ *can be recovered from* $\mathbf{y} = \mathbf{\Phi x}$ *by solving an* ℓ_1*-minimization problem.*

As for the *RIP*, if M is chosen according to the bound above, then we have exact signal recovery with overwhelming probability.

By analyzing the bound, it is straightforward to see how the coherence μ directly influences the number of needed M; in fact the bigger the coherence between basis

the larger the required number of measurements and vice versa. It is not surprising then, that the interest is focused in low coherence basis due to the fact that if we are in the best possible scenario, namely $\mu = 1$ (largely uncorrelated basis), then we can reach exact signal recovery with the smallest possible number of measurements M which is proportional to the logarithm of N.

2.4.3 Universality in random matrices

Another advantage given by random matrix construction is that using such approach it is not necessary to know in advance (at the encoding stage) the sparsifying basis $\mathbf{\Psi}$. In fact, we can write the measurements as

$$\mathbf{y} = \mathbf{\Phi}\mathbf{\Psi}\mathbf{s}$$

but if we know that $\mathbf{\Phi}$ entries are chosen according to a Gaussian distribution and $\mathbf{\Psi}$ is the orthonormal basis in which the signal is sparse, it follows that $\mathbf{\Theta} = \mathbf{\Phi}\mathbf{\Psi}$ is also distributed according to a Gaussian distribution. Without knowing which is the basis $\mathbf{\Psi}$, for sufficiently high M as seen above, the *RIP* property can be satisfied. This property is often called *universality* since the sparsifying basis does not need to be known at the encoding stage since any orthonormal basis multiplied by a Gaussian sensing basis results in a Gaussian distributed basis $\mathbf{\Theta}$. Moreover, interesting results come from [35] where the authors show that very similar results can also be obtained for the case of sub-Gaussian distributions.

2.5 Recovery

The recovery problem is about recovering a sparse signal, without knowing the exact position of non-zero coefficients, given only a small set of randomly-projected measurements.

2.5.1 Recovery in noiseless environment

As introduced in the previous section, the standard recovery algorithm used in CS is the ℓ_1-minimization. To understand why this kind of minimization is used, let us start by analyzing the two main characteristics we know about the signal:

- *Sparsity* For a certain integer k we know that $||\mathbf{x}||_0 \leq k$
- *Linear system of equations* It links the known measurements vector $\mathbf{y}$ and the unknown signal $\mathbf{x}$ through $\mathbf{y} = \mathbf{\Phi}\mathbf{x}$

It is straightforward to see that the linear system is ill-posed, in fact there are less equations than unknowns since by assumption $M < N$, thus there are infinitely many $\mathbf{x}$ that can satisfy the system $\mathbf{y} = \mathbf{\Phi x}$.
Here is where the role of the sparsity in $\mathbf{x}$ comes in: among all the possible solutions we want to pick the sparsest one. This problem can be written as

$$\begin{aligned} \underset{\tilde{\mathbf{x}} \in \mathbb{R}^N}{\text{minimize}} \quad & ||\tilde{\mathbf{x}}||_0 \\ \text{subject to} \quad & \mathbf{y} = \mathbf{\Phi}\tilde{\mathbf{x}}. \end{aligned}$$

It can be shown [39] that by using as few as $M = k + 1$ measurements by solving this ℓ_0-optimization problem, it is possible to exactly recover any k-sparse signal with high probability.

Unfortunately this technique is extremely costly from a computational point of view since it is an NP-hard combinatorial minimization problem which is intractable [40].

To overcome this, a natural solution may be trying to solve the problem as a standard least squares one. Unfortunately, if this problem is solved as a *least squares* (LS) problem thus minimizing the ℓ_2-norm, then we have no exact recovery. This is because by solving the LS problem, the solution is the vector which satisfies the system of linear equations with the minimal energy which however does not necessarily implies that the signal must be sparse.

Let us now briefly discuss the category of recovery algorithms which have been proposed in CS literature. They can be grouped in three main categories:

- combinatorial ones [41]
- greedy algorithms [42]
- convex problems *e.g.*, ℓ_1-minimization also known as *basis pursuit*

Combinatorial algorithms arise from *combinatorial group testing* problems where the goal is to design and minimize tests in order to obtain all the k non-zero coefficients from a larger vector $\mathbf{x}$. The advantage of these algorithms is the low requirement in terms of computational power required; the drawback is the higher number and the required structure for the acquired measurements.
Moreover some works [43] also shown that they can be seen as special cases of l_1-minimization, i.e., in [44] the authors show how combinatorial algorithms can be seen as recursive l_1-minimization problems using binary projection matrices.

A different approach is used in greedy algorithms: the approach on which greedy algorithms rely on is the iterative approximations of the signals coefficients and support, until convergence is reached. Two of the most simplest examples of these

iterative approaches are the *orthogonal matching pursuit* (OMP) [45] and the *iterative thresholding* algorithm. Both of them can almost reach the same guarantees on signal recovery as the l_1-minimization but using stronger constraints (e.g. smaller RIP constants).

The latter category, the one of ℓ_1 minimization problems is the one which has attracted most of the attention of the CS community. This is due to the large quantity of theoretical results which have been obtained in terms of recovery quality and measurements bounds, to the fact that is a convex problem and last but not least to the good performance this method can achieve. The basis pursuit problem can be written as

$$\begin{aligned} \underset{\tilde{\mathbf{x}}\in\mathbb{R}^N}{\text{minimize}} \quad & ||\tilde{\mathbf{x}}||_1 \\ \text{subject to} \quad & \mathbf{y} = \mathbf{\Phi}\tilde{\mathbf{x}} \end{aligned} \tag{2.4}$$

or equivalently,

$$\begin{aligned} \underset{\tilde{\mathbf{x}}\in\mathbb{R}^N}{\text{minimize}} \quad & ||\tilde{\mathbf{x}}||_1 \\ \text{subject to} \quad & \mathbf{x} \in \mathcal{B}(\mathbf{y}) \end{aligned} \tag{2.5}$$

with $\mathcal{B}(\mathbf{y}) = \left\{\mathbf{x} : \mathbf{\Phi}\mathbf{x} = \mathbf{y}\right\}$. Solving the basis pursuit problem is equivalent to find the smallest ℓ_1-norm ball that intersects the hyperplane generated by the linear equations system. The intersection point corresponds to the recovered $\mathbf{x}$.

To better understand how ℓ_1-minimization can effectively lead to an exact recovery, let us analyze Figure 2.1 which represents a (two dimensional) geometrical comparison between ℓ_1-norm and ℓ_2-norm minimization. As we can see, the ℓ_1-norm it is more likely to exactly recover the original signal because the ℓ_1-norm ball is a polytope instead of a sphere as the ℓ_2-norm. This means that the solution is unique only when the intersection happens at norm ball vertexes which correspond to sparser signals given that along the axis most of the components of the vector are zero.
Solving the ℓ_1-minimization can lead to excellent results, but we still need to be evaluate the accuracy of the signal recovery in terms of the recovery error. In a noiseless environment the result is strong: as stated in the following theorem is it possible to obtain *exact* signal recovery.

Theorem 2.3. *(Theorem 1.1 of [46]). Suppose that* $\mathbf{\Phi}$ *satisfies the RIP of order* $2k$ *with* $\sigma_{2k} < \sqrt{2} - 1$ *and we obtain measurements of the form* $\mathbf{y} = \mathbf{\Phi}\mathbf{x}$*. Then*

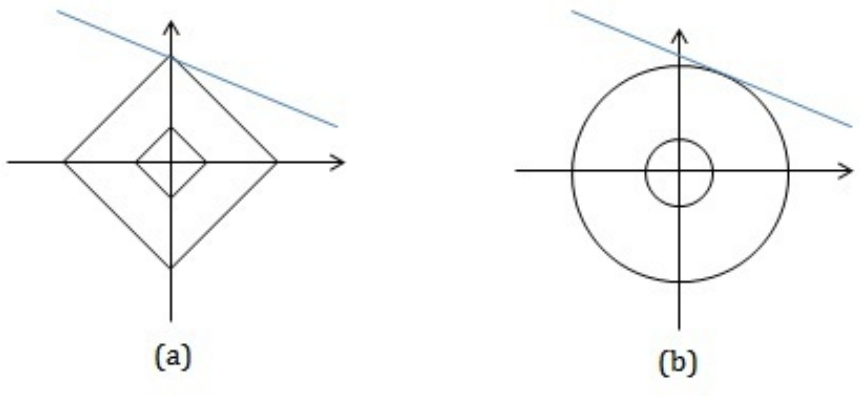


Figure 2.1: Comparison between two kind of l_p-norm minimization in order to exploit sparse signal recovery. (a) corresponds to the l_1-norm minization (b) corresponds to l_2-norm minization

$\mathcal{B}(\mathbf{y}) = \left\{\mathbf{z} : \mathbf{\Phi z} = \mathbf{y}\right\}$, *the solution* $\hat{\mathbf{x}}$ *to (*2.5*) obeys* [1]

$$||\hat{\mathbf{x}} - \mathbf{x}||_2 \leq C_0 \frac{\sigma_k(\mathbf{x})_1}{\sqrt{k}}.$$

In fact, with a fairly small number of measurements $O(k \log \frac{N}{k})$ (2.3) which are enough to to guarantee the RIP property to be satisfied for the considered sensing matrix, k-sparse signals are exactly recovered. It is worth noting that the most important word in the above statement is "exactly", in fact it is a quite strong result the fact that a sparse signal can be recovered without any information loss given just a small number of randomly acquired samples.

2.5.2 Recovery in noisy environment

Heretofore we tackled the signal recovery given the compressed measurements in a noise-free environment, unfortunately this is not the standard case. In fact, depending on the setting, signal is often corrupted by noise which may arise from different

[1] $\sigma_k(x)_1$ *quantifies the compressibility of* $\mathbf{x}$ *calculated by approximating it with a* k*-sparse signal. If* $\mathbf{x} \in \Sigma_k$ *then this quantity is equal to zero.*

sources such as electronic components or happens to be the result of operations like quantization.

In this section we analyze the signal recovery from noise corrupted measurements. In particular, in order to make the problem easily characterizable, the noise we take into account is additive white gaussian noise (AWGN) with zero mean and finite variance.

The ℓ_1 minimization problem needs to be adapted to this setting. In fact, the equality sign in (2.4) can not be satisfied in presence of additive noise. It is rather preferable to employ an ℓ_2 norm in order to guarantee the equality in the least squares sense.

The problem of recovering $\mathbf{x}$ from its compressed measurements $\mathbf{y} = \mathbf{\Phi x}$ becomes the one of recovering $\mathbf{x}$ given the noisy $\mathbf{y} = \mathbf{\Phi x} + \mathbf{\Phi n}$ where $\mathbf{n}$ is AWGN.

The convex optimization problem in (2.4) slightly changes since we have to keep into account some error bounds ϵ due to noise presence and becomes the so-called LASSO problem that is

$$\begin{aligned} \underset{\tilde{\mathbf{x}} \in \mathbb{R}^N}{\text{minimize}} \quad & ||\tilde{\mathbf{x}}||_1 \\ \text{subject to} \quad & ||\mathbf{\Phi}\tilde{\mathbf{x}} - \mathbf{y}||_2 \leq \epsilon \end{aligned}$$

This is the standard recovery problem used in CS literature for recovering noise-corrupted signals.
Let us now analyze the performance of the LASSO problem in analogy to what we have done with the basis pursuit problem. Starting from the bounded noise theorem (1.2) in [46] and using the concentration property from Gaussian distribution, assuming then $\epsilon = 1$, we can introduce two corollaries.

Corollary 2.1. *(Corollary 1.1 of [47]) Suppose that* $\mathbf{\Phi}$ *satisfies the RIP of order* $2k$ *with* $\delta_{2k} < \sqrt{2} - 1$. *Furthermore, suppose that* $\mathbf{x} \in \Sigma_K$ *and that we obtain measurements of the form* $\mathbf{y} = \mathbf{\Phi}(\mathbf{x} + \mathbf{n})$ *where the entries of* $\mathbf{n}$ *are i.i.d.* $\mathcal{N}(0, \sigma^2)$. *Then when* $\mathcal{B}(\mathbf{y}) = \left\{z : ||\mathbf{\Phi x} - \mathbf{y}||_2 \leq 2\sqrt{M}\sigma\right\}$, *the solution* $\hat{\mathbf{x}}$ *to equation (2.5) obeys to*

$$||\hat{\mathbf{x}} - \mathbf{x}||_2 \leq 8 \frac{\sqrt{1 + \sigma_{2k}}}{1 - (1 + \sqrt{2})\sigma_{2k}} \sqrt{M}\sigma$$

with probability at least $1 - \exp(-C_0 M)$.

Losing a bit of the generality of corollary (2.1) and assuming $\mathbf{\Phi}$ to be column normalized, we obtain the following corollary:

Corollary 2.2. *(Corollary 1.2 of [47]) Suppose that* $\mathbf{\Phi}$ *has unit norm columns and satisfies the RIP of order* $2k$ *with* $\delta_{2k} < \sqrt{2} - 1$. *Furthermore, suppose that* $\mathbf{x} \in \Sigma_K$ *and that we obtain measurements of the form* $\mathbf{y} = \mathbf{\Phi}(\mathbf{x} + \mathbf{n})$ *where the entries of*

$\mathbf{n}$ *are i.i.d.* $\mathcal{N}(0,\sigma^2)$*. Then when* $\mathcal{B}(\mathbf{y}) = \left\{\mathbf{z} : ||\mathbf{\Phi}^\top(\mathbf{\Phi}\mathbf{z}-\mathbf{y})||_\infty \leq 2\sqrt{\log N}\sigma\right\}$*, the solution* $\hat{\mathbf{x}}$ *to equation (*2.4*) obeys*

$$||\hat{\mathbf{x}}-\mathbf{x}||_2 \leq 4\sqrt{2}\frac{\sqrt{1+\sigma_{2k}}}{1-(1+\sqrt{2})\sigma_{2k}}\sqrt{K\log N}\sigma$$

with probability at least $1-\frac{1}{N}$*.*

Next, given that the RIP property and the coherence can be put in relation, starting from the above results one can check then how coherence value can influence the recovery performances. However, the results obtained in this way are usually not as significant as the ones obtained by directly exploiting coherence property by itself [48] [49].

The following theorem showed in [49] is a consequence of coherence influence in signal recovery

Theorem 2.4. *Suppose that* $\mathbf{\Phi}$ *has coherence* μ *and that* $\mathbf{x} \in \Sigma_k$ *with* $k < \left(1/\mu + 1\right)/4$*. Furthermore, suppose that we obtain measurements of the form* $\mathbf{y} = \mathbf{\Phi}(\mathbf{x}+\mathbf{n})$*. Then when* $\mathcal{B}(y) = \left\{z : ||Az-y||_2 \leq \epsilon\right\}$*, the solution* $\hat{\mathbf{x}}$ *to (*2.4*) obeys*

$$||\mathbf{x}-\hat{\mathbf{x}}||_2 \leq \frac{||\mathbf{n}||_2+\epsilon}{\sqrt{1-\mu(4k-1)}}$$

As we can see this error bound is general, meaning that it also works when there is no additive noise, e.g., $\epsilon = 0$ and $||\mathbf{n}||_2 = 0$, and it is the result of a worst-case analysis.

To have more significant results let us start by an alternative yet equivalent formulation for the recovery problem, in which the parameter λ is introduced. It can be used, by adjusting it value, promote sparsity in reconstructed signal. The result is a quadratic program that can be solved by means of convex optimization, and it is:

$$\underset{\hat{\mathbf{x}}\in\mathbb{R}^N}{\text{minimize}}\frac{1}{2}||\mathbf{\Phi}\hat{\mathbf{x}}-\mathbf{y}||_2^2 + \lambda||\hat{\mathbf{x}}||_1 \tag{2.6}$$

Thus, using this formulation as starting point, as proved in [50] we have

Theorem 2.5. *Suppose that* $\mathbf{\Phi}$ *has coherence* μ *and that* $\mathbf{x} \in \Sigma_k$ *with* $k < 1/(3\mu)$*. Furthermore, suppose that we obtain measurements of the form* $\mathbf{y} = \mathbf{\Phi}(\mathbf{x}+\mathbf{n})$ *where the entries of* $\mathbf{n}$ *are i.i.d* $\mathcal{N}(0,\sigma^2)$*. Set*

$$\lambda = \sqrt{8\sigma^2(1+\alpha)\log(N-k)}$$

for some fairly small value $\alpha > 0$*. Then with probability exceeding*

$$\left(1 - \frac{1}{(N-k)^{\alpha}}\right)\left(1 - \exp\left(-k/7\right)\right)$$

the solution $\hat{\mathbf{x}}$ *to (*2.6*) is unique,* $supp(\hat{\mathbf{x}}) \subset supp(\mathbf{x})$*, and*

$$||\hat{x} - x||_2^2 \leq \left(\sqrt{3} + 3\sqrt{2(1+\alpha)\log\left(N-k\right)}\right)^2 k\sigma^2$$

In this last theorem we can see that the probability exponentially tends to 1 since both multiplicative terms tends to 1 as the problem size increases. This means that for large N and $N-k$, with high probability, the recovery error $||\hat{\mathbf{x}} - \mathbf{x}||_2^2$ is, up to a multiplicative constant, proportional to $k\sigma^2 \log\left(N-k\right)$.

Heretofore we described the basic assumptions, properties and recovery techniques owed to CS. Leveraging this, in the following Chapters we will introduce few novel methods to tackle CS problems.

Chapter 3

Compressive estimation of AR parameters

We leverage the concepts of AR modeling for compressive signal processing we previously discussed to introduce, in this chapter, techniques which can be used to estimate these parameters in the compressed domain. Among the different classes of the AR coefficients estimators available in literature, we start by focusing on the LS estimator. This choice, along with an ad-hoc sensing matrix design, allow us to explicitly estimate the regression coefficients.

3.1 Autoregressive modeling

As previously discussed, if we assume that the signal to be sensed can be approximated by a certain class of parametric signals, then estimating its parameters has important implications in terms of inferring signal characteristics, and using these to improve signal recovery. Hence, we seek a model that allows us to describe a signal by means of few parameters that can be efficiently estimated in the compressed domain and that are compact enough to be transmitted at low communication cost. We propose to employ autoregressive (AR) modeling to address this task. The reason is twofold: first, it is a general model able to describe signals' characteristics and second, it gives an extremely compact representation.

More in detail, an AR process of order p is a parametric model able to describe the time-varying nature of a process in which the output values linearly depend on their previous values. The strength of such model is the ability to describe a signal by means of few parameters. In fact, this modeling approach has been used in a variety of applications including, among others, linear predictive coding of speech [51], spectral estimation and biomedical signal processing [52]. It has also been successfully applied to model natural signals such as audio and images [53] [54]. Hence,

being able to extract this information given the compressed measurements opens the door to many interesting applications such as covariance estimation, adaptive CS and compressive classification.

Let us start with a more formal description of the model which will be used through the rest of the thesis. An AR process of order p is described by:

$$\mathbf{x}_t = \sum_{i=1}^{p} \mathbf{x}_{t-i}\mathbf{a}_i \; + \boldsymbol{\epsilon}_t, \tag{3.1}$$

where $\boldsymbol{\epsilon}$ is called driving process and $\mathbf{a}$ is the vector of the regression coefficients. In other words, it can be seen as a filtering operation over a process $\boldsymbol{\epsilon}$ with an all-pole filter with coefficients given by $\mathbf{a} = [\mathbf{a}_1 \; \dots \; \mathbf{a}_i \; \dots \; \mathbf{a}_p]^\top$. Given an AR(p) process $\mathbf{x} \in \mathbb{R}^N$, we define $\mathbf{x}^+$ as a subset of $\mathbf{x}$ composed by its samples with index from $(p+1)$ to N. Let us also define the matrix $\mathbf{X} \in \mathbb{R}^{(N-p)\times p}$ constructed in the following way

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_p & \mathbf{x}_{p-1} & \dots & \mathbf{x}_1 \\ \mathbf{x}_{p+1} & \mathbf{x}_p & \dots & \mathbf{x}_2 \\ \vdots & \dots & \dots & \vdots \\ \mathbf{x}_{N-1} & \mathbf{x}_{N-2} & \dots & \mathbf{x}_{N-p} \end{bmatrix}, \tag{3.2}$$

which allows us to rewrite (3.1) as

$$\mathbf{x}^+ = \mathbf{X}\mathbf{a}. \tag{3.3}$$

A concept of great importance is the stationarity of the process; a process is said to be wide sense stationary (WSS) if the first and second moments of the process do not vary with respect to time. This ensures the process to be stable as the regression evolves during time. Given the AR process, it is WSS if all the poles of the following lie inside the unitary circle

$$A(z) = 1 + z^{-1}\mathbf{a}_1 + \cdots + z^{-p}\mathbf{a}_p.$$

In the particular case of $p = 1$, it turns out that the stationarity of the process can be ensured by imposing $|\mathbf{a}_1| < 1$.

Another interesting feature of the AR modeling is that the parameters can be used to estimate the spectrum of a signal leading to the well-known parametric spectral estimation. In fact, it is known [55] that the Power Spectral Density (PSD) estimate of a signal R_y is related to the coefficients of the regression according to

$$R_y = \left| \frac{1}{A(\omega)} \right|^2,$$

where $A(\omega) = 1 + \mathbf{a}_1 e^{-i\omega} + \cdots + \mathbf{a}_p e^{-ip\omega}$. This gives us a hint of the usefulness of this modeling approach which, if the AR parameters are compressively estimated, allows us to perform parametric compressive spectral estimation.

Heretofore we described a model which can be used to characterize a signal and estimated its main characteristic, along with its properties. The next step, which we discuss in the following Chapter is how to estimate these parameters directly in the compressed domain. In particular, we introduce two different compressive AR estimators thanks which we develop key applications that employ the proposed compressive AR estimation framework to solve important practical estimation problems described in Chapters 4,5 and 6. In the next Section we start our discussion on the compressive estimation of the autoregressive parameters.

3.2 Compressed LS estimator

Let us start by reviewing the LS estimator in the uncompressed domain. Starting from the matrix vector representation of the AR process regression given by

$$\mathbf{x}^+ = \mathbf{X}\mathbf{a}, \tag{3.4}$$

it is straightforward to write the LS estimator of $\mathbf{a}$ as the solution to the following minimization problem

$$\arg\min_{\hat{\mathbf{a}}} \|\mathbf{x}^+ - \mathbf{X}\hat{\mathbf{a}}\|_2^2 \tag{3.5}$$

or, more concisely, as $\hat{\mathbf{a}} = \mathbf{X}^\dagger \mathbf{x}^+$ where "†" denotes the pseudo-inverse.

In order to have an analogous LS estimator for the compressed domain we need to employ a sensing matrix able to preserve the structure of the regression. As we can see from (3.2), the LS estimator for a process of order p, needs $p+1$ shifted versions of the input signal. Hence, the idea is to build a sensing matrix from which, given the output measurements, it is possible to extract the compressed $p+1$ shifted versions of $\mathbf{x}$. This means that the sensing matrix should be made of $p+1$ sub-blocks $\mathbf{\Phi}'$ where each of them senses a shifted version of the unknown signal $\mathbf{x}$.

Then, if we use (3.4), multiplying both sides by the sensing block $\mathbf{\Phi}'$ we obtain

$$\mathbf{y}^+ = \mathbf{Y}\mathbf{a}, \tag{3.6}$$

where $\mathbf{y}^+ = \mathbf{\Phi}'\mathbf{x}^+$ and $\mathbf{Y} = \mathbf{\Phi}'\mathbf{X}$. Hence, if the sensing matrix is made up of shifted sensing blocks it is possible to extract the quantities $\mathbf{y}^+$ and $\mathbf{Y}$ directly from the measurement vector $\mathbf{y}$.

More formally, let us assume that the main building block $\mathbf{\Phi}' \in \mathbb{R}^{\mu \times (N-p)}$ with $\mu = M/(p+1)$, has entries distributed according to $\boldsymbol{\phi}'_{ij} \sim \mathcal{N}(0, \frac{1}{M})$. Then, the

$$\begin{bmatrix} \phi'_{1,1} & \phi'_{1,2} & \phi'_{1,3} & \phi'_{1,4} & \phi'_{1,5} & \phi'_{1,6} & 0 & 0 \\ \phi'_{2,1} & \phi'_{2,2} & \phi'_{2,3} & \phi'_{2,4} & \phi'_{2,5} & \phi'_{2,6} & 0 & 0 \\ 0 & \phi'_{1,1} & \phi'_{1,2} & \phi'_{1,3} & \phi'_{1,4} & \phi'_{1,5} & \phi'_{1,6} & 0 \\ 0 & \phi'_{2,1} & \phi'_{2,2} & \phi'_{2,3} & \phi'_{2,4} & \phi'_{2,5} & \phi'_{2,6} & 0 \\ 0 & 0 & \phi'_{1,1} & \phi'_{1,2} & \phi'_{1,3} & \phi'_{1,4} & \phi'_{1,5} & \phi'_{1,6} \\ 0 & 0 & \phi'_{2,1} & \phi'_{2,2} & \phi'_{2,3} & \phi'_{2,4} & \phi'_{2,5} & \phi'_{2,6} \end{bmatrix}$$

Figure 3.1: Circulant blocks structure of $\mathbf{\Phi}$.

proposed sensing matrix $\mathbf{\Phi} \in \mathbb{R}^{M \times N}$ is made of $p+1$ circulant blocks of $\mathbf{\Phi}'$ as depicted in Fig. 3.1.

In order to obtain the measurements of shifted versions of the original signal (which are needed to obtain $\mathbf{y}^+$ and $\mathbf{Y}$), we exploit the structure of the sensing matrix in Fig. 3.1. As can be seen, each sub-block acquires a shifted version of the input signal through the sub-sensing matrix $\mathbf{\Phi}'$. Hence we can write $\mathbf{y}_{1+\mu(k-1)\to k\mu} = \mathbf{\Phi}'\mathbf{x}_{k\to(N-p+k-1)}$, which means that the vector $\mathbf{y}$ is made of $p+1$ blocks of length μ which are the measurements corresponding to different shifts of $\mathbf{x}$.

In particular, using (3.6) we define the compressed LS estimator for AR(p) coefficients, as:

$$\arg\min_{\hat{\mathbf{a}}} \|\mathbf{y}^+ - \mathbf{Y}\hat{\mathbf{a}}\|, \tag{3.7}$$

where the chosen M must be an integer multiple of $p+1$.

It is worth noting that the proposed estimator, working directly in the reduced space of the measurements domain, is computationally less demanding with respect to the corresponding LS estimator (3.5) in the uncompressed domain. The complexity is due to the computation of the pseudo-inverse of $\mathbf{X}$ ($\mathbf{Y}$) in the uncompressed (compressed) domain. When considering the uncompressed case, it strictly depends on the value of p and the length of signal N according to $O(p^2(N-p))$, where the most influential term is N because the order of the process is typically small. On the other hand, in the compressed domain the required computational power for the proposed estimator drastically reduces to $O(p^2\mu)$ with $\mu \ll (N-p)$.

Sensing matrix validation

We now numerically validate the proposed sensing matrix comparing its recovery performance to that of the most used ones in literature for which theoretical results on the recovery performance exists [56] [57]. In particular, we fix a sparsity level $s = 100$ and randomly pick the support of the non-zero components. Then we compare the recovery performance of different sensing matrices matrices by running 1000 different Monte Carlo runs over different M values by compressing sparse signals and then recovering them using LASSO. The recovery error is defined as $\|\mathbf{x} - \hat{\mathbf{x}}\|_2 / \|\mathbf{x}\|_2$ where $\hat{\mathbf{x}}$ is the recovered signal. The results (Fig. 3.2) show that

recovery error of the proposed matrix is slightly higher than the Gaussian sensing matrix and lower than that of the Bernoulli one. The recovery performance of the proposed matrix is hence comparable to that of a circulant matrix, which is a very popular choice, and has a negligible performance loss with respect to a Gaussian matrix. Similar results can be found for other values of M, N, s and p and are omitted for brevity.

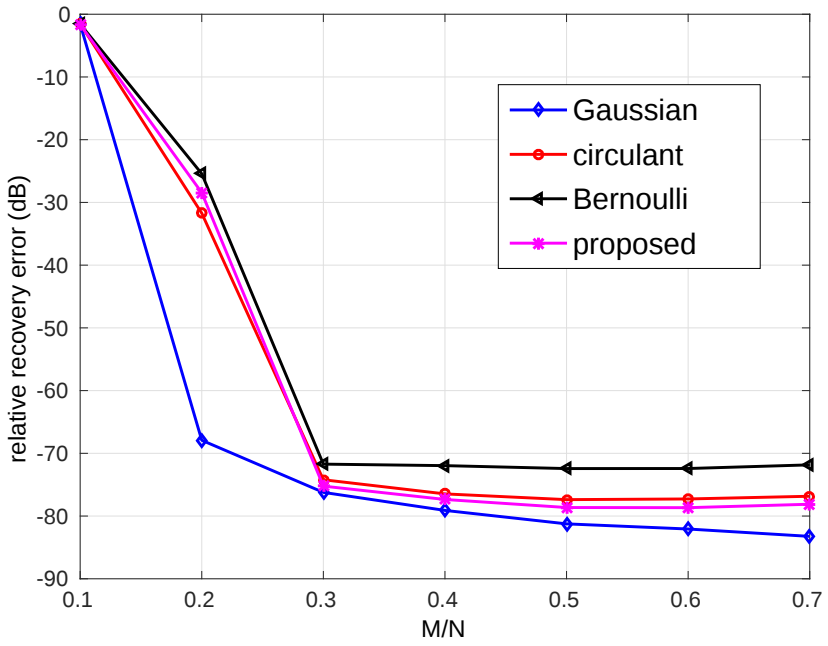


Figure 3.2: Comparison of recovery ability of different sensing matrices using a signal of length $N = 1000$, sparsity $s = 100$ and $p = 9$.

3.3 Compressive Bayesian AR(1) estimation

As previously discussed, the coefficients of an AR process can be estimated in an efficient way in the compressed domain and can be used to perform compressive covariance estimation. Among other things, they can be used to estimate the compressibility of a signal.

While the compressive AR(p) estimator we introduced in the previous section achieves excellent performance and it is more general since can be applied to autoregressive processes of any order p, it can be further improved and made more robust by employing Bayesian techniques. In fact, if we consider autoregressive processes of

the first order $p = 1$, it is possible to explicitly obtain a compressive Bayesian estimator as described below. It is also worth noting that when higher orders $p > 1$ are considered, the stationarity constrains we impose on the autoregressive coefficient can not be exploited due to its recursive dependency with the reflection coefficients. Thus, it is not possible to obtain an explicit compressive Bayesian AR(p) estimator without employing computationally expensive algorithms such as Markov chain Monte Carlo like techniques. Hence, in the following we focus on the particular case when the order reduces $p = 1$: the AR coefficient turns out to be the correlation coefficient among the samples of the signal. This information is closely related to the complexity of a signal and its inference can improve the knowledge of the uncompressed signal. In the following we introduce a novel Bayesian estimator for compressed AR(1) processes which leads to better performance for highly compressed signals.

3.3.1 Modeling

In order to improve the readability, let us denote with ρ the AR(1) coefficient. According to the CS scheme we previously introduced, the acquired measurements lead to the following observation model

$$\begin{aligned} \mathbf{\Phi}'\mathbf{x}^+ &= \mathbf{\Phi}'(\mathbf{x}^-\rho + \boldsymbol{\epsilon}) \\ \mathbf{y}^+ &= \mathbf{y}^-\rho + \boldsymbol{\zeta}, \end{aligned}$$

where $\mathbf{x}^- = \mathbf{x}_{1\to(N-p)}$.

The set of parameters $\mathbf{\Theta} = \{\rho, \sigma_\epsilon^2\}$ we wish to estimate includes the AR(1) coefficient and the variance of the Gaussian process of the AR model. Noting that $\boldsymbol{\zeta}$ follows a Gaussian distribution, we can write the probability of the observation model as:

$$p(\mathbf{y}^+|\mathbf{y}^-, \mathbf{\Theta}) = \mathcal{N}(\mathbf{y}^+|\mathbf{y}^-\rho, \mathbf{\Phi}\sigma_\epsilon^2\mathbb{I}\mathbf{\Phi}^\intercal).$$

Let us discuss the choice of the prior distributions for the set of parameters $\mathbf{\Theta}$. When choosing the probability distribution for ρ it is worth noting that, in order to ensure stability, the necessary condition for the stationarity of the process [58] requires the values of ρ to be bounded in the interval $(-1,1)$. Among the class of bounded probability distributions we choose the Beta distribution since it allows us to shape the signal distribution in a flexible way, which includes the non informative uniform distribution as a special case. The Beta distribution is a bounded distribution defined on the interval [0,1]. Hence, in order to bound the Beta distribution in the interval of interest $[-1,1]$, the probability of ρ can be defined as:

$$p(\rho) = \text{Beta}(0.5 + 0.5\rho|\rho_\alpha, \rho_\beta),$$

where ρ_α, ρ_β are hyperparameters controlling the shape of the distribution. Since we expect $\sigma_{\boldsymbol{\epsilon}}^2$ to be positive valued, we put on this parameter an inverse gamma distribution which is commonly used for variance modeling and, being conjugate prior with the Gaussian distribution, allows easier calculations. The probability of $\sigma_{\boldsymbol{\epsilon}}^2$ is hence defined as:

$$p(\sigma_{\boldsymbol{\epsilon}}^2) = \mathrm{IG}\left(\sigma_{\boldsymbol{\epsilon}}^2 \middle| a, b\right),$$

where a and b are two hyperparameters. According to the Bayesian modeling employed we have four hyperparamters $\boldsymbol{\Theta}_h = \{\rho^\alpha, \rho^\beta, a, b\}$ which, since no assumptions can be made, are manually set. This choice allows more flexibility, *e.g.,* given that ρ^α, ρ^β control the distribution on ρ, in image processing problems we may want to peak the distribution around 1 since an image patch has higher probability to be a smooth region than a high frequency one. Moreover, when $\rho^\alpha = \rho^\beta = 1$ this will result in a flat (uninformative) prior on ρ.

3.3.2 Inference

In order to obtain better results by taking into account the uncertainties of the estimates, we employ Bayesian inference. The goal is to obtain the probability distributions of the parameters in $\boldsymbol{\Theta}$ instead of point-wise estimates like *maximum likelihood* or *maximum-a-posteriori*.

If we consider the log joint distribution $\log p(\mathbf{y}^+, \mathbf{y}^-, \boldsymbol{\Theta})$, the presence of the Beta distribution which is not conjugate prior to the Gaussian distribution does not allow a direct Bayesian inference. For this reason we employ the variational Bayesian framework [59]. This approach, in particular using the Mean-Field approximation [59], seeks a set of disjoint set of distributions that approximate the full posterior which minimizes the Kullback-Leiber (KL) divergence. In particular we have:

$$p(\boldsymbol{\Theta}|\mathbf{y}^-, \mathbf{y}^+) \simeq q(\Theta) = q(\rho)q(\sigma_{\boldsymbol{\epsilon}}^2).$$

The best function $q(\bullet)$ in terms of KL-divergence is obtained as the expectation $q(\bullet) = \langle p(\boldsymbol{\Theta})\rangle_{\boldsymbol{\Theta}\setminus\bullet}$. Thus, given the log-joint distribution, we can write:

$$\begin{aligned}\log q(\rho) &= \langle \log p(\mathbf{y}^+, \mathbf{y}^-, \boldsymbol{\Theta})\rangle_{\boldsymbol{\Theta}\setminus\rho} = \\ &= c_0 - \frac{1}{2\sigma_{\boldsymbol{\epsilon}}^2}(\mathbf{y}^+ - \mathbf{y}^-\rho)^\intercal(\boldsymbol{\Phi}\boldsymbol{\Phi}^\intercal)^{-1}(\mathbf{y}^+ - \mathbf{y}^-\rho) + \\ &+ \underbrace{(\rho^\alpha - 1)\log(1+\rho)}_{\mathrm{e}} + \underbrace{(\rho^\beta - 1)\log(1-\rho)}_{\mathrm{g}},\end{aligned}$$

where c_0 is a constant term in ρ. Since this distribution does not allow an easy form, we propose to use an approximation for the terms e and g which is very strict

around 0. In particular we have:

$$e = (\rho^{\alpha} - 1)\log(1+\rho) \leq (\rho^{\alpha} - 1)\rho$$
$$g = (\rho^{\beta} - 1)\log(1-\rho) \leq (\rho^{\beta} - 1)(-\rho - \frac{\rho^2}{2}).$$

Using such approximations, the probability distribution of ρ becomes:

$$q(\rho) \sim \mathcal{N}\left(\rho \,\middle|\, -\frac{m}{2n}, \frac{1}{2n}\right) \tag{3.8}$$

having defined

$$\begin{aligned} m &= -\frac{1}{2}(\rho^{\beta} - 1) - \frac{1}{2\sigma_{\epsilon}^2}\mathbf{y}^{-\intercal}(\mathbf{\Phi}\mathbf{\Phi}^{\intercal})^{-1}\mathbf{y}^{-} \\ n &= (\rho^{\alpha} - 1) - (\rho^{\beta} - 1) + \frac{1}{\sigma_{\epsilon}^2}\mathbf{y}^{-\intercal}(\mathbf{\Phi}\mathbf{\Phi}^{\intercal})^{-1}\mathbf{y}^{+}. \end{aligned} \tag{3.9}$$

The same process applies for the parameter σ_{ϵ}^2. It can be shown that it is distributed as an inverse gamma defined by:

$$q(\sigma_{\epsilon}^2) \sim \mathrm{IG}\left(\sigma_{\epsilon}^2 \middle| \tilde{a}, \tilde{b}\right)$$

where

$$\begin{aligned} \tilde{a} &= a + \frac{M}{2} \\ \tilde{b} &= b + \frac{1}{2}\left(\mathbf{y}^{+} - \mathbf{y}^{-}\rho\right)^{\intercal}\left(\mathbf{\Phi}\mathbf{\Phi}^{\intercal}\right)^{-1}\left(\mathbf{y}^{+} - \mathbf{y}^{-}\rho\right). \end{aligned}$$

Then, as can be seen from (3.9), in order to compute $q(\rho)$ we only need the expectation of the inverse of σ_{ϵ}^2 which can be computed as:

$$\left\langle \frac{1}{\sigma_{\epsilon}^2} \right\rangle = \frac{\tilde{a}}{\tilde{b}}. \tag{3.10}$$

To conclude, the whole inference process of the AR(1) parameters is summarized in Algorithm 1. It is worth noting that in order to define a criterion for evaluating the convergence, a good metric is the difference in the likelihood $p\left(\mathbf{y}^{+}|\mathbf{\Theta}, \mathbf{y}^{-}\right)$ between two subsequent iterations.

3.3.3 Comparison

This algorithm, despite being specific for the AR(1) model, is able to improve the estimation performance when high compression ratios are employed. To better show

Algorithm 1 Bayesian AR(1) parameter estimation algorithm

INPUT: $\mathbf{y}^+, \mathbf{y}^-, \rho^\alpha, \rho^\beta, a, b$

INITIALIZE: $\hat{\rho} = 1, (1/\hat{\sigma_\epsilon^2}) = 1$

1: **while** not reached convergence **do**
2: Compute $\hat{\rho} = \langle q(\rho) \rangle$ according to (3.8) and (3.9)
3: Compute $\hat{\rho}^2 = \mathrm{Var}(q(\rho))$ according to (3.8) and (3.9)
4: Compute $\frac{\hat{1}}{\sigma_\epsilon^2} = \left\langle \frac{1}{q(\sigma_\epsilon^2)} \right\rangle$ according to (3.10)
5: **end while**

OUTPUT: $\hat{\rho}, \hat{\sigma_\epsilon^2} = \langle q(\sigma_\epsilon^2) \rangle$

this, in Figure 3.3 we can see a comparison of the two algorithms. For the experiment, a stationary AR(1) process was generated with $\rho \in (-1,1)$ and the signals were compressed using the sensing matrix design introduced in this chapter. For both algorithms we computed the NMSE defined as NMSE $= \left(\frac{\hat{\rho}-\rho_{\text{true}}}{\rho_{\text{true}}}\right)^2$. The results were averaged over 1000 experiments. It is worth noting that, according to the described setup, the convergence was usually reached in no more than 5 iterations.

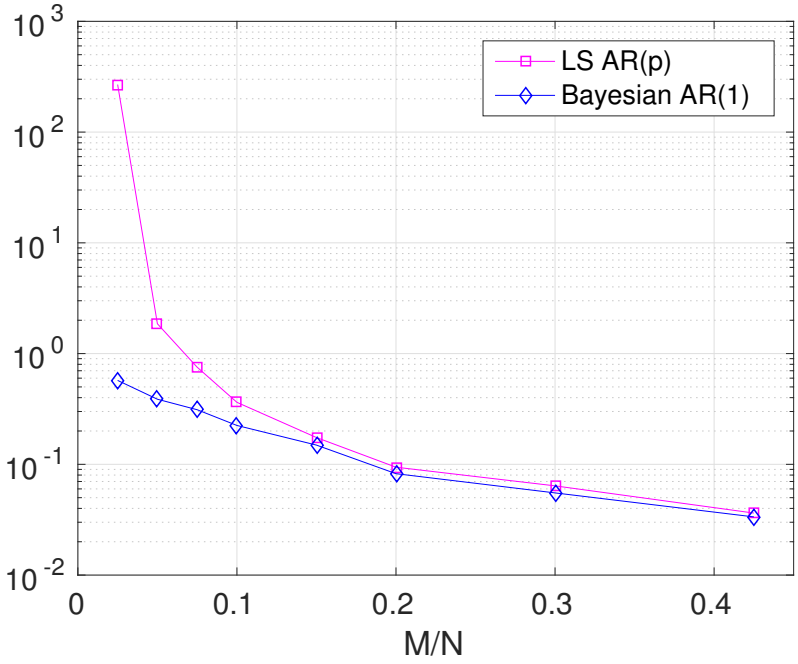


Figure 3.3: NMSE comparison of LS method in (3.7) and the Bayesian method described in Algorithm 1. In this experiment $\rho = 0.8$, $N = 100$. Only high compression ratios are shown since the two methods tends to converge as M/N approaches 1.

We can see that, for extremely compressed signals whose compression ratio M/N ranges between 0.01 and 0.3 the Bayesian AR(1) specific algorithm achieves significantly lower estimation errors. We rely on this important feature to design an adaptive compressive imaging scheme that is explained more in detail in the next section.

3.4 Experimental results

In this section we evaluate the performance of the proposed technique with few example applications. In particular we focus on the compressive LS AR(p) estimator,

since an application suited for the Bayesian AR(1) estimator is described in detail in the following Chapter.

3.4.1 Sparsely-driven AR recovery

Here we consider a process obtained by filtering a sparse driving process. This model is of interest in speech processing, since the speech signal can be modeled as an AR process where the residual (driving process of the model) is sparse as in Multi-Pulse excitation coding [51].

To tackle this problem we need to introduce a basis able to represent our process sparsely. Since, by assumption, the driving noise of AR(p) process is sparse, our sparsifying basis will be that filtering matrix which reverses the AR(p) filtering effect.

Given the AR(p) process $\mathbf{x}$ and the sparse driving process $\mathbf{u}$, we can write $\mathbf{u} = \mathbf{A}\mathbf{x}$, where $\mathbf{A}$ is the matrix performing the inverse filtering operation on $\mathbf{x}$, *i.e,* the lower-triangular Toeplitz matrix created from the length-N vector $[1\ \mathbf{a}^\top\ 0\ \dots\ 0]$. Then, the basis we are looking for is $\mathbf{H} \triangleq \mathbf{A}^{-1}$.

When considering the compressed measurements we can equivalently write $\mathbf{y} = \mathbf{\Phi x} = \mathbf{\Phi H u}$, which leads to the following LASSO problem for the AR(p) recovery:

$$\arg\min_{\mathbf{u}} \|\mathbf{y} - \mathbf{\Phi}\tilde{\mathbf{H}}\mathbf{u}\|_2 + \lambda\|\mathbf{u}\|_1, \tag{3.11}$$

where $\tilde{\mathbf{H}}$ is the sparsifying basis constructed from the estimate of $\mathbf{a}$ in the CS domain. It is worth noting that using the proposed estimator along with the LASSO in (3.11) to improve the signal recovery only requires the CS measurements and hence no side informations or training data must be known.

Hence, in order to validate the technique we generate a sparse synthetic signal and use LASSO in (3.11) to recover the signal (Fig. 3.4). The error metric we use for this experiment is the same as for the sensing matrices comparison. The results are averaged over 100 different trials. In particular, we compare the proposed method with the recovery using the Discrete Fourier Transform (DFT) sparsifying basis, and the one proposed in [3]. It is important to highlight that, while both the proposed and the DFT-based recovery do not require any additional information apart for the CS measurements, the recovery in [3] requires the *a priori* knowledge of the AR coefficients of the process. The DFT-based recovery shows a very large error, showing that the DFT basis is not good for sparsifying this class of signals. Conversely, the other two methods show lower error, which decreases as M increases. Moreover, the errors converge as $M \to N$ because the two methods use the same sparsifying basis, but constructed with different AR coefficients estimates (from CS measurements and from original signal). Finally, in Fig. 3.5 we show the recovery of a *natural* signal: a vocalized tract of a speech signal. For this experiment, in

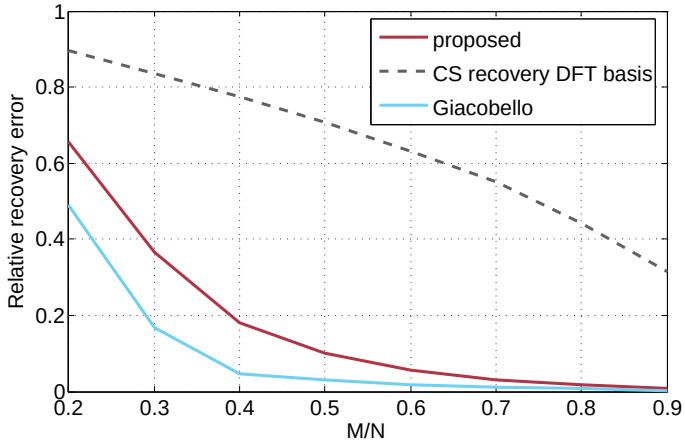


Figure 3.4: Relative recovery error comparison of a synthetic sparse signal with $N = 1000$, and sparsity $s = 100$. We compare the recovery proposed by Giacobello *et al.* in [3], a CS recovery assuming the sparsity to be in the frequency domain and the proposed method with regression coefficients estimated from the measurements.

order to run a fair comparison, we compare the techniques which require only the CS measurements: the DFT-based recovery and the proposed technique. As we can see, with a small number of measurements, the signal recovered using the proposed method approximates the original one very well. In contrast, the recovery assuming DFT-based sparsity is not able to approximate the original signal accurately. In fact, the MSE of the proposed recovery (shown in Fig. 3.5) is -30.46 dB conversely to the DFT-based which is -23.4 dB.

3.4.2 Compressive spectral estimation

Here we show an example of spectral estimation performed in the compressed domain using the proposed LS estimator. Since, as shown in Section 3.1, there is a direct relationship between the AR coefficients and the PSD of the signal, the accuracy of the estimated parameters can also be seen from the PSD perspective. In Fig. 3.6(b) we show the comparison between the spectrum of a signal made of three sinusoidal components estimated using the FFT for the original signal and the compressive spectral estimation for CS measurements. As can be seen, the spectrum generated with the estimated AR coefficients accurately approximates the spectrum spikes of the original signal.

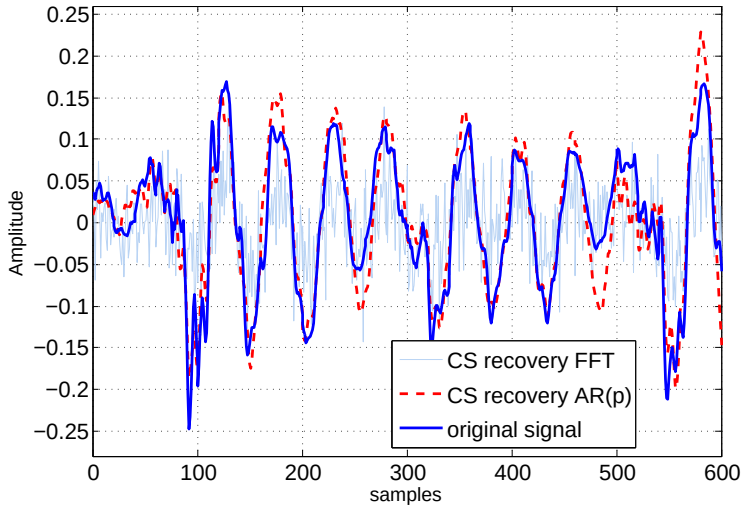


Figure 3.5: Recovery comparison of a vocalized tract of speech signal of length $N = 600$ with $M = 200$ and $p = 10$. We compare the original signal with the recovered versions made by using the DFT as a sparsifying basis, and the LASSO with the estimated regression coefficients.

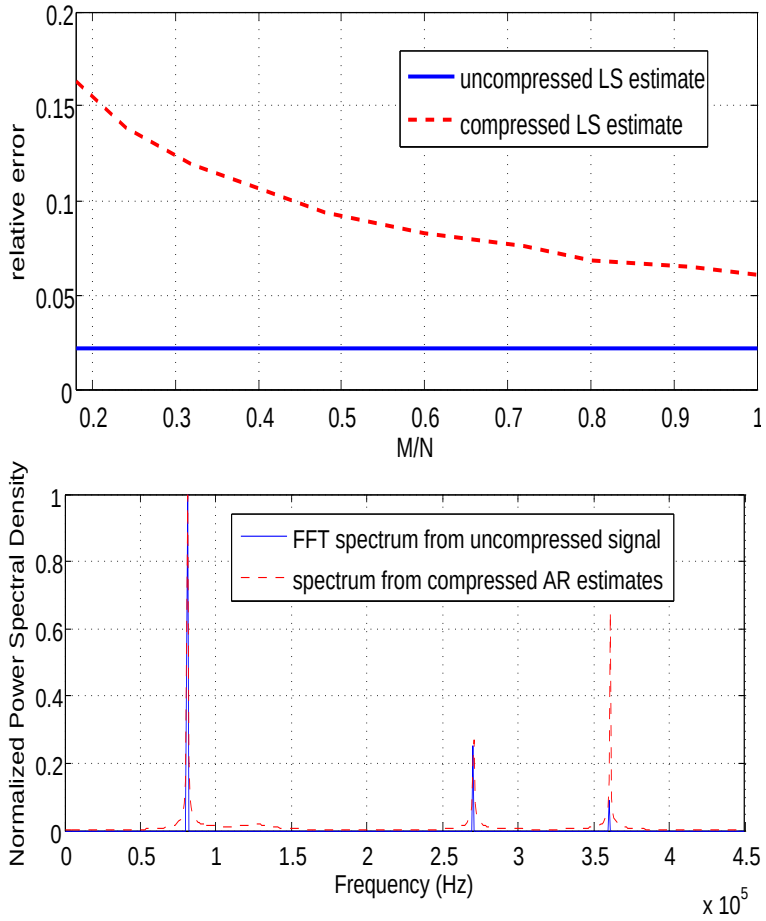


Figure 3.6: Performance evaluation of the proposed estimator. In (a) we show the performance loss deriving from using compressed measurements instead of uncompressed data, in (b) we show an example of compressive spectral estimation via estimated AR coefficients.

Chapter 4

Adaptive compressive imaging

4.1 Introduction

In this Chapter we propose a novel algorithm for adaptive compressive imaging based on Bayesian AR(1) inference. Ideally we want to use more measurements for the regions of the image which are more complex and less measurements for spatially smooth regions since we expect spatially smooth regions to be sparser than the complex counterparts. Additionally, since by CS theory we introduced in Chapter 2 it is known that the number of measurements required to have a good reconstruction depends on the sparsity of the signal, we want to correctly allocate the measurements in a content-adaptive fashion. Lastly, it is worth noting that we want this procedure to be fully compressive meaning that all the operations but the recovery must be performed in the compressed domain.

To motivate our algorithm, let us start by considering two blocks $\mathbf{b}_1, \mathbf{b}_2 \in \mathbb{R}^{B\times B}$ extracted from an image $\mathbf{I} \in \mathbb{R}^{NB\times NB}$.

In Figure 4.1 two common kinds of block characteristics are depicted: low ($\mathbf{b}_1$) and high ($\mathbf{b}_2$) spatial frequency blocks. Natural images typically involve large smooth regions [60] hence the number of low frequency blocks is significantly higher than the high frequency ones. Therefore, a compressive imaging scheme taking the same number of measurements on all blocks is going to provide significantly sub-optimal performance, since smooth blocks are going to be over represented ($\hat{\mathbf{b}}_1$), and high-frequency blocks under represented ($\hat{\mathbf{b}}_2$) in the compressed representation. This may also lead to significant blocky artifacts in the reconstructed image. This can be seen in the recovered blocks $\hat{\mathbf{b}}_1$ and $\hat{\mathbf{b}}_2$ in Fig. 4.1 where the block with high spatial frequencies has a noticeably less visually satisfying recovery. The algorithm we are going to introduce aims to adapt the sensing process by selecting a suitable number of measurements for each block depending on its statistics. Let us start by considering the block $\mathbf{b}_1$ which shows low spatial frequencies. The correlation

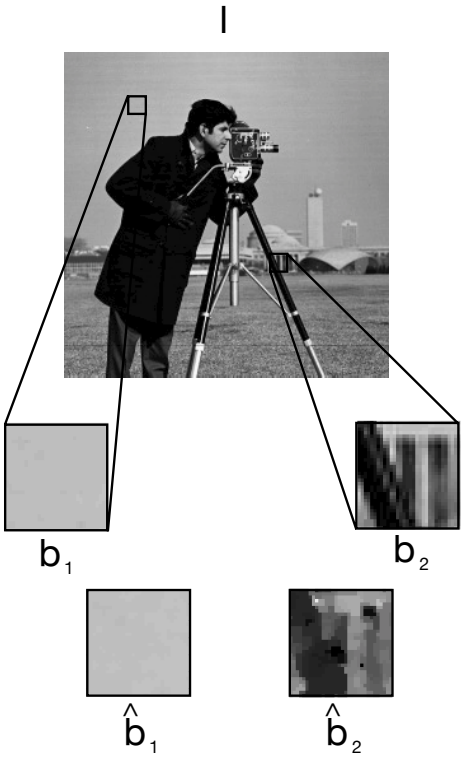


Figure 4.1: Blocks in natural images exhibit high ($\mathbf{b}_1$) and low ($\mathbf{b}_2$) spatial frequencies. $\hat{\mathbf{b}}_1$ and $\hat{\mathbf{b}}_2$ are the recovered blocks with $M/N = 0.1$

coefficient ρ computed using an AR(1) LS estimator (as described in (3.5)) using vec($\mathbf{b}_1$) is high, as one may expect, *i.e.,* $\rho = 0.99 \simeq 1$. The same coefficient, estimated from compressed measurements using Algorithm 1 (compression ratio of 0.3, with $B = 16$) is $\hat{\rho} = 0.95$.

On the other hand, a block which contains high spatial frequencies $\mathbf{b}_2$ will result in lower correlation coefficient ($\rho = 0.6$ and $\hat{\rho} = 0.55$ respectively computed from uncompressed and compressed block).

This means that by working in the compressed domain, hence without the need of recovering the signal, we can adapt the number of needed measurements depending on the complexity of each block. We remark that, in this specific application, since we are only interested in measuring the compressibility of each block of the image, an AR(1) model is perfectly adequate to the task at hand.

4.1.1 Adapting the compression ratio

In order to correlate the compression ratio $r = M/N$ and the AR(1) coefficient ρ we must define a function that will result in high compression ratio when $\rho \simeq 1$, and will reduce it as the correlation coefficient goes toward zero. In this Chapter we propose to use a function $f_c(\rho)$ which has been empirically shown to be effective. The function is defined as:

$$f_c(\rho) = \lfloor (M_{\max} - M_{\min}) \, |1 - \rho|^{\gamma} + M_{\min} \rfloor, \tag{4.1}$$

where $M_{\max}$ ($M_{\min}$) corresponds to the maximum (minimum) value of M which is desired for the problem of interest and γ is a parameter which can be used to tune the recovery quality. For values of γ higher than 0.5, as the signal becomes less complex (ρ approaches 1) the number of measurements slowly decreases. On the other hand, when γ is smaller than 0.5, as long as ρ is not very close 1 the number of required measurements does not decrease steeply. This behavior is highlighted in our experiments where we show that different γ lead to different reconstruction qualities.

4.1.2 Adaptive compressive imaging scheme

We now put it all together and introduce a scheme in which the number of measurements is adapted to the complexity of signal itself. The scheme we consider involves a sensor and a reconstruction unit. This approach is very general and allows us to include many different subproblems as special cases. In fact, this scheme can be employed for any kind of signal, although in this work we consider its application to block-based compressive imaging. The basic concept is that the sensor first acquires a small batch of measurements. From those measurements, using the proposed estimator the actual number of measurements needed to achieve the desired quality is calculated, and more measurements are acquired so as to reach this number. It should be noted that the estimation algorithm can run directly on the sensor; alternatively, one could envisage that the first batch is sent to the RU, which runs the estimation and then requests from the sensor the extra measurements needed.

Though not considered in this work, we could equivalently employ this scheme in a compressive imaging system [61] in which measurements are acquired at subsequent time intervals, instead of block by block.

We assume the sensor acquires the image by means of CS which is performed separately on each block $\mathbf{b}_i$ of size $B \times B$ which the image is composed of. At first, each block $\mathbf{b}_i \; \forall i \leq n_B$ is sensed using the minimum number of measurements $M_{\min}$, which is typically not sufficient for a visually satisfactory reconstruction. From this batch of measurements, the complexity can be efficiently estimated using Algorithm 1. At this point, the sensor is aware of the complexity of each block since

it has access to $\hat{\rho}_i \ \forall i \leq n_B$. The number of needed measurements M_R to achieve a satisfactory recovery is then computed using (4.1) for each block and the missing $M_i = M_R - M_{\min}$ measurements $\mathbf{y}_i^\star \in \mathbb{R}^{M_i \times 1} \ \forall i \leq n_B$ are requested from the sensor. The last step consist in the recovery of each block. To solve the problem we employ the BCS-SPL-DDWT technique introduced in [1] which performs block CS image recovery and showed superior recovery capabilities. In order to suit our problem, this algorithm has been adapted by substituting fixed sensing matrices with sensing matrices which are different in size for each block. More in detail the adaptive sensing problem defined at each block $\mathbf{b}_i$ becomes

$$\begin{bmatrix} \mathbf{y}_i^\star \\ \mathbf{y}_i \end{bmatrix} = \begin{bmatrix} \mathbf{\Phi}^\star \\ \mathbf{\Phi}_i \end{bmatrix} \ \mathrm{vec}(\mathbf{b}_i),$$

where $\mathbf{y}_i^\star$ and $\mathbf{\Phi}^\star$ are the measurements and the sensing matrix obtained during the first coarse compression at a fixed compression ratio given $M_{\min}$; $\mathbf{y}_i$ and $\mathbf{\Phi}_i$ are instead the measurements and the sensing matrix resulting after the estimation of the correlation coefficient ρ_i.

Block size

The choice of the block size is crucial to achieve good performance. In fact, if the blocks are not sufficiently small the approximation of the block with an AR(1) process will not hold since the block is more likely to contain subregions which exhibit different types of correlations among the pixels. More formally, we can state that we seek a dimension B for the blocks such that the stationarity assumption on the blocks is satisfied. We ran experiments over different images and we found that one of most common choices for block size *i.e.,* $B = 16$ is the largest block size that fits the stationarity requirements. Hence, this is the block size we have used for the experiments presented in the next section.

4.2 Results

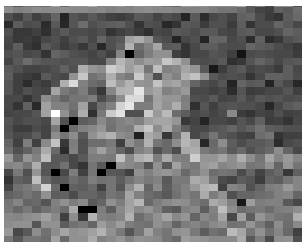
In this section we show the performance of the proposed adaptive compressive imaging scheme. The parameters of Algorithm 1 denoted by $\mathbf{\Theta}_h$ were set according the prior knowledge we have on the blocks statistics. We chose $\rho_\alpha = 0.8$ and $\rho_\beta = 0.8$ because, as previously discussed, this values lead to a distribution on the parameter ρ highly peaked around 1 since we expect most of the blocks to be smooth. The Bayesian parameters were set as $a = 2$, $b = 1$. These values were experimentally found to yield better recovery results. Then we set $M_{\min} = 22$ to be large enough to allow good complexity estimation from the first batch of measurements, but still not sufficient image recovery, and $M_{\max} = 250$ to be large but still having $M/N < 1$.

Given the image *Cameraman*, we start by showing in Fig. 4.2 the actual number of measurements chosen by the algorithm for each block by taking into account three different values of γ. Higher values of γ favor more complex blocks by allocating most of the measurements to this class of blocks. As the value of γ decreases, more measurements are added to blocks which show *medium* complexity. The result is an increased compression ratio for high γ values and a reduced compression for smaller values.

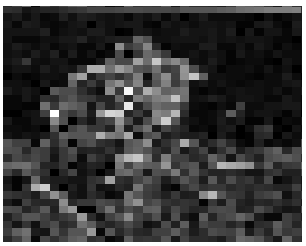
Next, we assess the end-to-end performance of the adaptive compressive imaging system by evaluating the PSNR and SSIM [62] values of recovered images compared with non-adaptive BCS-SPL-DDWT algorithm. For this experiment the parameters are set as above; in both the adaptive and non-adaptive case, the same total number of measurements is used; in the non adaptive case, the measurements are equally split among all blocks. The results of these experiments are shown in Table 4.1. We can see that in the vast majority of the cases considered, the ability to adapt the number of measurements according to the complexity of the block leads to superior recovery performance. More in detail, the proposed algorithm reached PSNR gains ranging from 0.4dB up to more than 6dB. We also considered a very sparse image containing well defined edges and smooth regions *i.e.,* the Shepp-Logan phantom image. The results are extremely good for the proposed adaptive algorithm which is able to efficiently allocate the measurements only where needed. For this particular case the gain reached up to 4 dB. Very high gains are also achieved when considering depth-map images (shown in Fig. 4.3) where allocating more measurements mainly in high complexity regions is crucial.

In Fig. 4.4 we show a crop of the image *Lena* acquired with the adaptive and non-adaptive algorithms. The detail shows a high frequency region which is good tesbed for evaluating the visual quality of the recovered images. As can be seen a higher visual quality is achieved when the adaptive algorithm is used. The details are better preserved due to the higher number of measurements allocated in this region.

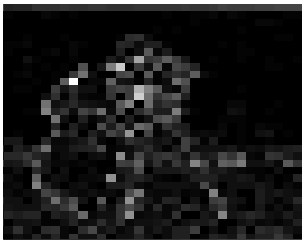
(a) Original image



(b) $\gamma = 0.2$

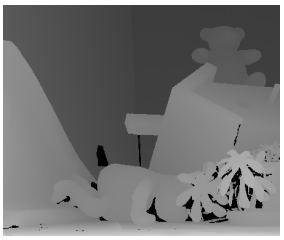


(c) $\gamma = 0.5$



(d) $\gamma = 0.8$

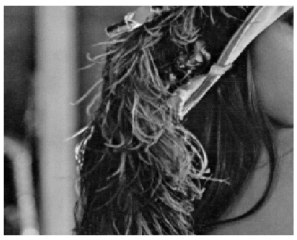
Figure 4.2: Number of required measurements for each block for different values of the parameter γ. In (b)—(d) gray intensity indicates the number of measurements needed for the corresponding block. Black corresponds to $M_{\min}$ and white to M_{max}.



(a) Teddy

(b) Ballet

Figure 4.3: Depth-map images taken from [4] and [5].



(a) Original image

(b) Proposed adaptive system

(c) Non-adaptive [1]

Figure 4.4: Detail of *Lena*; (a) original image; (b) image recovered using the proposed algorithm; (c) image obtained using [1].

Table 4.1: Comparison between the proposed adaptive compressive imaging system and BCS-SPL-DDWT [1].

	γ	**PSNR (dB)**		**SSIM**		**M/N**
		proposed	[1]	proposed	[1]	
Lena	0.2	**37.1390**	36.7500	**1.0000**	**1.0000**	0.5360
	0.5	**30.9200**	30.4680	**1.0000**	**1.0000**	0.1930
	0.8	**28.1990**	27.3360	**1.0000**	0.9970	0.1110
Cameraman	0.2	**34.2470**	33.4370	**1.0000**	**1.0000**	0.4760
	0.5	**29.0850**	28.8230	**1.0000**	0.9990	0.2550
	0.8	**25.9430**	25.0910	**1.0000**	0.9940	0.1320
Barbara	0.2	**28.4160**	27.8840	**1.0000**	0.9980	0.4820
	0.5	**24.3780**	23.9040	**1.0000**	0.9970	0.2320
	0.8	22.8490	**22.8590**	**1.0000**	0.9900	0.1430
Monarch	0.2	**36.5910**	35.3400	**1.0000**	0.9980	0.4860
	0.5	**28.1490**	27.6660	**1.0000**	0.9950	0.2040
	0.8	**25.1980**	23.9850	**1.0000**	0.9910	0.1260
Shepp-Logan Phantom	0.2	**32.2890**	31.1050	**1.0000**	0.9980	0.1640
	0.5	**31.7860**	28.1710	**0.9980**	0.9960	0.1010
	0.8	**30.0290**	26.2460	**0.9980**	0.9940	0.0830
Teddy	0.2	**39.2730**	33.4550	**1.0000**	0.9980	0.3750
	0.5	**29.3980**	27.2180	**0.9970**	0.9960	0.1710
	0.8	**26.5270**	24.9280	**0.9950**	0.9940	0.0990
Ballet	0.2	**34.7370**	30.8670	**0.9990**	0.9980	0.3500
	0.5	**30.4340**	26.2080	**0.9950**	0.9940	0.1360
	0.8	**26.5790**	26.1400	**0.9950**	**0.9950**	0.1010

Chapter 5

Compressive covariance estimation

Being able to estimate the covariance matrix of a process is of paramount importance in many signal processing operations. In fact, the statistics of the signal carried by the covariance matrix are, in most cases, a sufficient statistic for most of the signal processing problems. Interestingly, when this information can be directly estimated from the compressed sensing measurements, then it can be easily included in compressive signal processing tasks leading to improved performance.

In fact, the knowledge of the covariance matrix of a signal estimated in the compressed domain has a lot of applications, including compressive power-spectrum estimation [63] [64], wideband spectrum sensing [65], incoherent imaging [66] and direction-of-arrival estimation [67] [68]. All these techniques take advantage from compressive covariance estimation since the number of sensors needed for the signal acquisition can be dramatically reduced. In order to estimate the covariance matrix of a process in the compressed domain, the main approaches used in literature are: maximum likelihood estimation (MLE), least squares [69] and convex optimization [6]. However, while the second approach is not able to guarantee the positive semidefiniteness of the covariance matrix, the former requires to have samples which shows a good statistical significance. It is also worth noting that, in order to make the compressive covariance estimation possible, the first two approaches require the sensing process to be able to preserve the second order statistics of the signals which can the be used to recover the covariance matrix. In order to provide guarantees for the preservation of the correlation matrix structure, in [70] the authors propose optimal sensing matrix design through the use of the sparse rulers.

Differently from the aforementioned methods, the proposed approach focuses instead on the estimation of the parameters of a covariance matrix defined by the structure of the AR process. This approach is computationally light, and it also ensures the positive semidefiniteness of the covariance matrix.

5.1 Estimation

So far we have presented an efficient way to estimate the coefficients of a compressed AR(p) process. In the following we discuss an important application which arises from the aforementioned technique: the compressive estimation of a structured Toeplitz covariance matrix. This kind of matrices play an important role in different fields such as integral equations, spline functions, mathematics, statistics, and signal processing.

Due to the nature of AR(p) processes, the associated covariance matrix has a Toeplitz structure. Before derivating the compressed covariance matrix estimator, let us introduce some notation: the symbol tril($\bullet$) denotes the operator which extracts the lower triangular part of a matrix, while we will use toeplitz($\mathbf{a}$) to denote a matrix built as a Toeplitz matrix constructed from vector $\mathbf{a}$.

Let us start with a simple model of a set of observations of AR(p) processes:

$$\mathbf{AX} = \mathbf{V},$$

where $\mathbf{X} \in \mathbb{R}^{N\times O}$ are the column-wise AR processes, $\mathbf{V} \in \mathbb{R}^{N\times O}$ are the column-wise driving noise vectors and $\mathbf{A}$ is the regression matrix (common to all the observation vectors) which only depends on the coefficients vector $\mathbf{a} \in \mathbb{R}^{1\times p}$. In particular, we have $\mathbf{A} = \text{tril}\left(\text{toeplitz}(\mathbf{a}^\star)\right)$, having defined $\mathbf{a}^\star = [1 \ -\mathbf{a}\ 0 \ \dots \ 0]^\intercal \in \mathbb{R}^{1\times N}$.

We recall that, by definition, the driving noise processes $\mathbf{V}$ are distributed according to $\mathcal{N}\left(0, \mathbb{I}\sigma^2_{\mathbf{v}}\right)$, therefore:

$$\boldsymbol{\mu}_{\mathbf{X}} = \langle \mathbf{X} \rangle = \left\langle \mathbf{A}^{-1}\mathbf{V} \right\rangle = \underline{0}.$$

Then, we can write the covariance matrix of the AR(p) process as

$$\begin{aligned} \boldsymbol{\Sigma} &= \langle (\mathbf{X} - \boldsymbol{\mu}_{\mathbf{x}})(\mathbf{X} - \boldsymbol{\mu}_{\mathbf{x}})^\intercal \rangle = \langle \mathbf{X}\mathbf{X}^\intercal \rangle = \\ &= \mathbf{A}^{-1} \left\langle \mathbf{V}\mathbf{V}^\intercal \right\rangle \mathbf{A}^{-1\intercal} = \mathbf{A}^{-1}\sigma^2_{\mathbf{V}}\mathbb{I}\ \mathbf{A}^{-1\intercal}. \end{aligned} \tag{5.1}$$

Therefore, given the coefficients $\mathbf{a}$ of the process and the variance of the driving noise σ^2_V, the covariance matrix of the process is uniquely defined by these two parameters.

Given the compressed measurements $\mathbf{y} = \boldsymbol{\Phi}\mathbf{X}$, we can use (3.7) to estimate $\mathbf{a}$, without any prior knowledge of $\sigma^2_{\mathbf{V}}$. In order to estimate $\sigma^2_{\mathbf{V}}$ let us write

$$\begin{aligned} \langle \mathbf{y}^\intercal \mathbf{y} \rangle &= \langle \mathbf{X}^\intercal \boldsymbol{\Phi}^\intercal \boldsymbol{\Phi} \mathbf{X} \rangle = \left\langle \mathbf{V}^\intercal \mathbf{A}^{-1\intercal} \boldsymbol{\Phi}^\intercal \boldsymbol{\Phi} \mathbf{A}^{-1} \mathbf{V} \right\rangle = \\ &= \text{tr}(\mathbf{A}^{-1\intercal}\boldsymbol{\Phi}^\intercal\boldsymbol{\Phi}\mathbf{A}^{-1}\mathbb{I}\sigma^2_{\mathbf{V}}) + \\ &+ \langle \mathbf{V} \rangle^\intercal \mathbf{A}^{-1\intercal}\boldsymbol{\Phi}^\intercal\boldsymbol{\Phi}\mathbf{A}^{-1} \langle \mathbf{V} \rangle, \\ &= \sigma^2_{\mathbf{V}} \text{tr}(\mathbf{A}^{-1\intercal}\boldsymbol{\Phi}^\intercal\boldsymbol{\Phi}\mathbf{A}^{-1}). \end{aligned}$$

Thus, we can estimate $\sigma^2_{\mathbf{V}}$ as

$$\hat{\sigma^2_{\mathbf{v}}} = \frac{\langle \mathbf{y}^\intercal \mathbf{y} \rangle}{\mathrm{tr}(\mathbf{A}^{-1\intercal} \mathbf{\Phi}^\intercal \mathbf{\Phi} \mathbf{A}^{-1})}, \tag{5.2}$$

using the sample mean estimator to compute the term $\langle \mathbf{y}^\intercal \mathbf{y} \rangle$.

To summarize, the covariance $\mathbf{\Sigma}$ of a compressed AR(p) process can be estimated according to (5.1) exploiting the coefficients vector $\mathbf{a}$ computed with (3.7) and the driving noise variance $\sigma^2_{\mathbf{v}}$ computed with (5.2).

5.2 Results

In the following we show the performance of the proposed technique for compressive covariance estimation and compare it with the algorithm proposed by Eldar et al. in [6] for structured Toeplitz covariance matrices. For the experiments, we generated synthetic AR processes of order $p = 4$, size $N = 100$ and considered $O = 200$ observations. Then, we compressed the resulting $\mathbf{X} \in \mathbb{R}^{N \times O}$ with different M/N ratios. The results were then averaged by running 1000 different experiments employing random sensing matrices.

To assess the performance we employ the normalized mean squared error (NMSE) defined as $\frac{\|\mathbf{\Sigma} - \hat{\mathbf{\Sigma}}\|_F^2}{\|\mathbf{\Sigma}\|_F^2}$ where $\mathbf{\Sigma}$ and $\hat{\mathbf{\Sigma}}$ are the true and the estimated covariance matrices respectively. As we can see from Fig. 5.1, at all undersampling rates the proposed algorithm shows lower NMSE than [6]. We also analyze the effects of higher orders of the regression and model mismatch. In Fig. 5.2 we show the NMSE of both techniques when the order of the process is $p = 10$. We considered this value since this is typically the largest order considered for natural signals. The experiment shows that the proposed algorithm is able to achieve lower NMSE compared to [6] and the results are comparable with those obtained with a smaller order.

Next, we analyze the mismatch scenario. Typically when the order of the process p is unknown, a good practice is to use an order slightly larger than the guessed one. Therefore we show an experiment in which we purposely used an increased order for the proposed AR(p) compressive covariance estimation algorithm. As in the previous experiments, the true order of the process is $p = 4$ while the augmented one is $p + 3$. As can be seen in Fig. 5.3 the proposed methods deals well with model mismatch and achieves results similar to those obtained using the correct value of p.

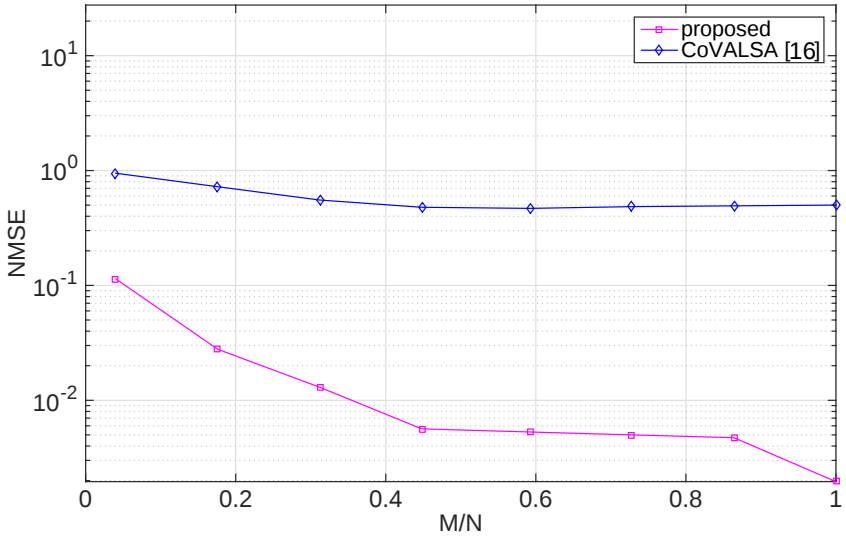


Figure 5.1: NMSE on compressive covariance estimation computed for the proposed technique and [6] evaluated at different compression ratios.

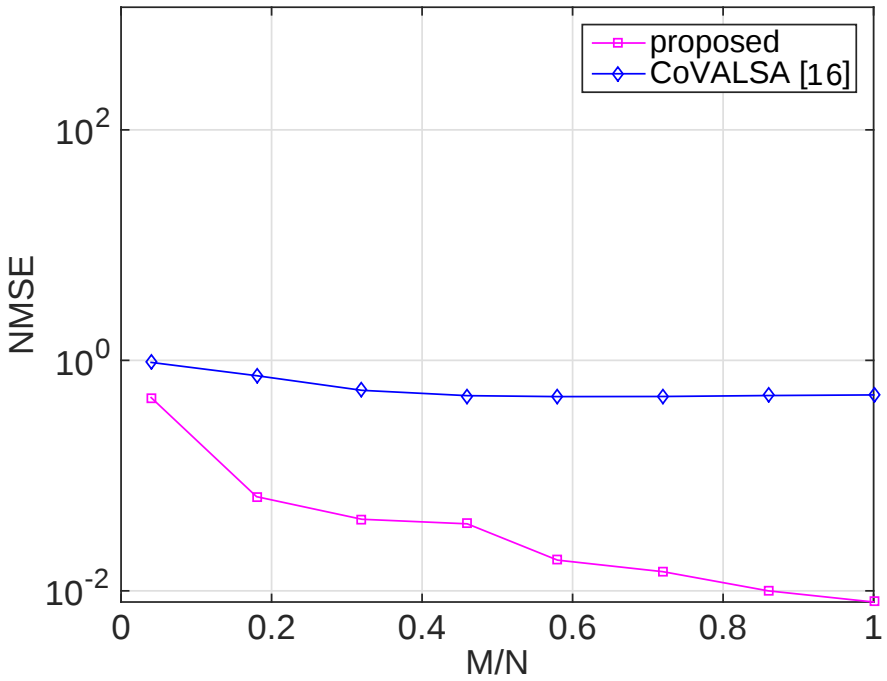


Figure 5.2: NMSE for the proposed technique and [6] techniques with $p = 10$.

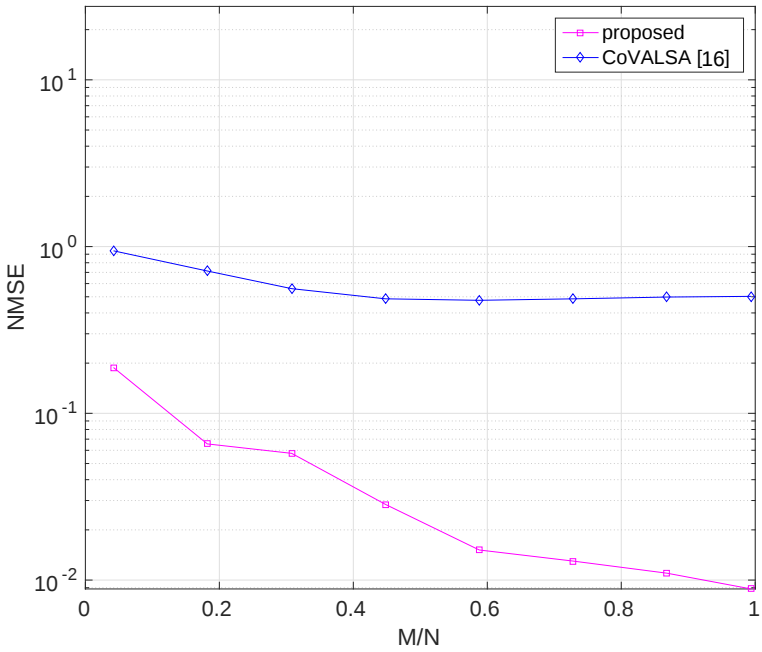


Figure 5.3: NMSE for the proposed technique and [6] techniques with a mismatched model order $p + 3$.

Chapter 6

Compressive AR classification

Heretofore we have discussed signal processing techniques and applications which take advantage (in terms of computational cost or improved accuracy) of being performed in the compressed domain. In this chapter we discuss another important application which directly falls in this category: compressive classification. The compressive classification attracted quite a lot of attention and few authors addressed this problem in their works which can be divided in two main categories: dictionary based and ℓ_2 distance based. Let us briefly discuss both categories.

The first category aims to use a dictionary whose atoms are related to the different classes to be identified. Hence, given a new compressed vector to be classified, after it has been recovered, ideally the largest element in the sparse representation will reveal the correct class. This approach has been used for robust face recognition in [71].

The second category exploits the ℓ_2 distance preservation due to the restricted isometry property (RIP) [46]. This property, which characterizes good sensing matrices used in CS, states that the ℓ_2 norm of a compressed signal is preserved up to a multiplicative constant. Hence, if the distance between two compressed vectors is preserved, distance based classification techniques can be directly applied to the compressed measurements. This approach is used in [72] [73]. However, if the distance between two vectors can not be directly used for classification, working in the compressed domain will not lead to any advantages. Other works, like the one in [74] put the focus on the compression of features extracted from the signal itself. The problem arising with this latter kind of compressive classification is that at some point it is necessary to have the knowledge of the original signal which is used to extract the feature vectors. While this is possible for signal acquired using traditional schemes and then compressed, the same can not be said for signals which have been acquired using CS hardware.

Differently from the techniques discussed above, the method we propose estimates the feature vectors used for the classification task in the compressed domain

achieving a fully compressive classification. As feature vectors we propose to use the coefficients of an autoregressive (AR) model of order p, which well approximate signals having a shaped spectrum. Many of the natural signals can be approximated using an AR model, among others the ones we consider in this work are speech signals and texture patches. In order to estimate the AR parameters in the compressed domain, we employ the technique described in Chapter 3 which consists in an ad-hoc sensing matrix construction and a least squares estimator. Differently from other techniques, the specific structure of the sensing matrix allows us to preserve the structure of the regression, which we propose to use for classification tasks. Then, the classification can be performed with well-known techniques such as support vector machines (SVM).

It is important to note that the proposed technique is not designed for a specific signal class, but is rather a more general approach. Thanks to ability of the AR processes to approximate a wide range of signals, our compressive classification technique can be employed in many different applications.

Moreover, as experimentally shown in Chapter 3, the proposed sensing matrix has excellent recovery capabilities thus allowing the recovery of the compressed signal if needed.

6.1 SVM classifier

The SVM classifier is one of most well-known algorithms for machine learning. It has been successfully applied in many different applications such as text and image recognition. Being a learning algorithm, at first it needs to be trained in order to learn the structure from the data itself. More formally, one wants to learn a mapping $\mathcal{X} \mapsto \mathcal{Y}$ where $\mathbf{x} \in \mathcal{X}$ is the available data and $\mathbf{y} \in \mathcal{Y}$ is the corresponding class label. When data is linearly separable, one may use as classifier the function $f(\mathbf{x}) = \text{sign}(\mathbf{w}^\intercal \mathbf{x} + \mathbf{b})$, that is check if the new data point to be classified is above or below the separating hyperplane defined by $\mathbf{w}^\intercal \mathbf{x} + \mathbf{b}$. This means that during the training stage the goal is to find the parameters $\{\mathbf{w}, \mathbf{b}\}$ which define the separating hyperplane with the additional constraint of maximum margin separation between the two classes. However, in most cases the data is not linearly separable, *i.e.*, it is not possible to find an hyperplane which can exactly separate the two classes of data. To overcome this, it is useful to use kernel functions which allow to map the data to a higher dimensional space in which it may be possible to find a separating hyperplane. Hence, the classifier function becomes $f(\mathbf{x}) = \mathbf{w}^\intercal \Phi(\mathbf{x}) + \mathbf{b}$ where $\Phi(\mathbf{x})$ is a kernel function which maps $\mathbf{x} \in \mathbb{R}^n$ to $\Phi(\mathbf{x}) \in \mathbb{R}^m$ with $n \ll m$. As for the linear classifier, finding the set of parameters $\{\mathbf{w}, \mathbf{b}\}$ which can guarantee data separation with the constraint of maximum margin separation can be written as a quadratic programming problem which can be efficiently solved.

6.2 Compressive classification with SVM

Here, we discuss the SVM classification of the compressive AR feature vectors. More formally, we want to map the compressively estimated AR parameters $\mathbf{a} \in \mathcal{A}$ to the class label $l \in \mathcal{L}$ using a SVM classifier. In order obtain better results, we employ a Gaussian radial basis function as kernel function for the SVM. This function, apart from being one of most used kernel functions, is the one who led to better experimental results.

We continue our discussion with a synthetic experiment: we generate 10000 AR processes of order $p = 6$ and length 1000 samples which are supposed to belong to two different classes. The first class is the one in which the first 3 elements of $\mathbf{a}$ are different from zero while the remaining elements are zero. The second class has instead the first 3 elements of $\mathbf{a}$ set to zero, while the others non null. It is also worth noting that the AR coefficients $\mathbf{a}$ have been generated in such a way to be wide sense stationary, *i.e.*, the roots of the polynomial defined by

$$\mathbf{z}^p - \mathbf{a}_1\mathbf{z}^{p-1} - \cdots - \mathbf{a}_{p-1}\mathbf{z} - \mathbf{a}_p = 0,$$

have to lie inside the unitary circle.

We investigate how SVM training and classification behave when working directly in the compressed domain at different compression ratios. For both compressed and uncompressed scenarios, we trained the SVM with 6000 out of a total of 10000 AR feature vectors. When considering the uncompressed classification, the AR parameters have been estimated using the Yule-Walker method since it is fast and achieves good results. We repeated the experiment over 100 different realizations and we averaged the results. The results of this experiment can be seen in Fig. 6.1. When the classification is performed in the original domain, we have that the 99% of the signals is correctly classified. On the other hand, when training and classification are performed in the compressed domain, the proposed method is still able to achieve good classification performance. As shown in Fig. 6.1, with less then 20% of the samples of the original signal, the compressive classification is able to correctly identify 95% of the signals. Moreover, as the M/N ratio increases the classification percentage in the compressed domain asymptotically tends to the one obtained in the original domain.

Next, we discuss what happens if training and classification are not performed in the same domain, namely compressed and original domains. This may be the case for the datasets which have not been compressively acquired, but rather compressed at a later stage and on which SVM classifiers have already been trained. In the same Fig. 6.1, it can be seen that the proposed technique is able to achieve the same good results and that the classification performance is almost the same independently from where training has been done: compressed or original domain.

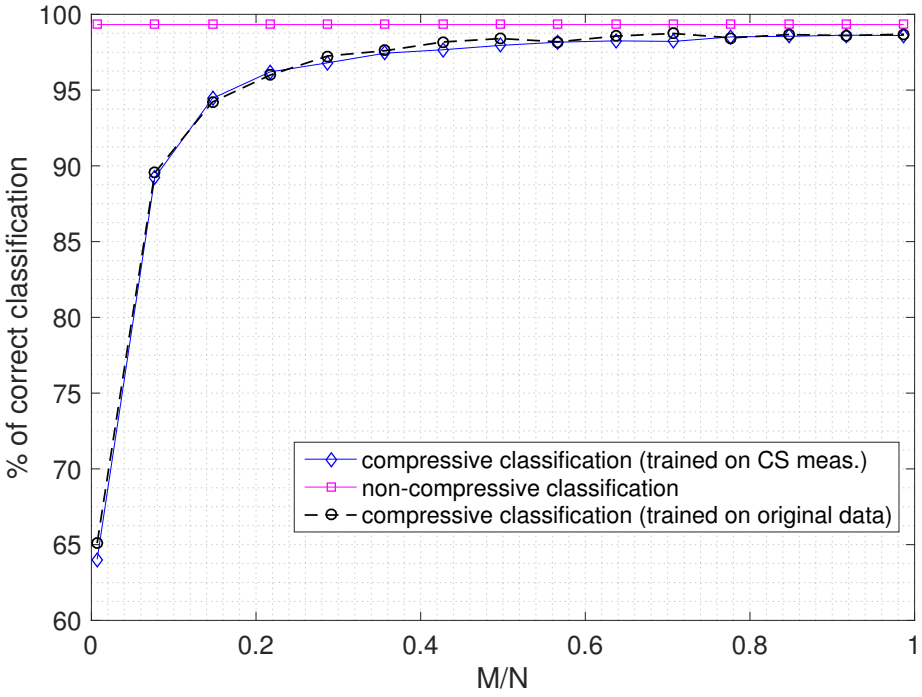


Figure 6.1: Results of the classification task of synthetic signals in the original and original domains. The black curve represents compressive classification performance using a SVM trained on non-compressed AR feature vectors.

The same performance is achieved because of the small distance between the true and compressively estimated AR coefficients as discussed in Chapter 3.

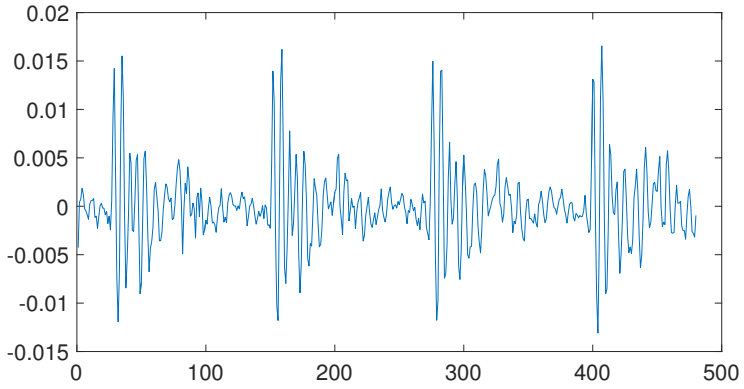
Hence, we can state that the compressive classification based on AR feature vectors can be used to efficiently classify signals based on their AR feature vector.

6.3 Experimental results

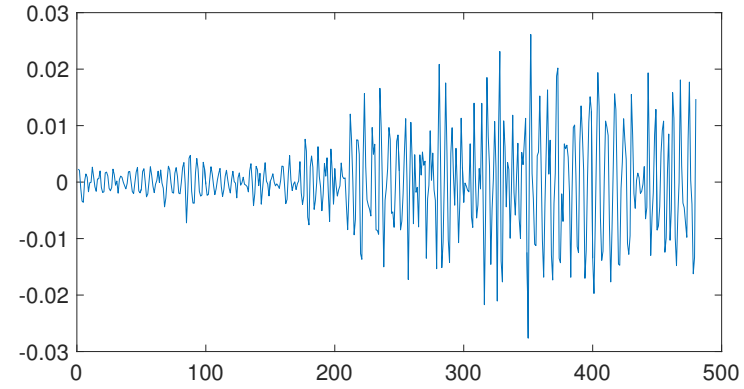
In this section we test the proposed classification scheme on two kinds of natural signals: speech and texture. Since AR models efficiently approximate the signals' spectrum, they are well suited to be used for classifying this classes of signals which exhibit a well defined spatial or temporal spectrum. More in detail speech signals are commonly approximated with AR models of order $p \sim 10$. For what concerns texture modeling, since the textures exhibit spatial frequencies with a well-defined spectrum AR approximation can be efficiently employed. Non-compressive autoregressive approximation of texture patches has been addressed in [75].

6.3.1 Speech voiced/unvoiced frame classification

The classification of frames extracted from speech signals into voiced or unvoiced is of paramount importance. Parametric voice compression and voice recognition are just few of the applications which need to be aware if a given frame is voiced or unvoiced. A frame is considered voiced if the sound is produced by the vibration of the vocal cords at specific frequencies *e.g.*, pronunciation of vowels as shown in Fig. 6.2(a). On the other hand, a frame is considered unvoiced if there is not a distinct frequency, but rather a filtered signal due to the modulation of the sound by the vocal tract *e.g.*, pronunciation of consonants as can be seen in Fig. 6.2(b).



(a)



(b)

Figure 6.2: Frames extracted from TIMIT dataset. (a) is a voiced frame, (b) is an unvoiced frame.

To test the proposed compressive classification for voiced/unvoiced classification, we used 30000 audio signals from the small TIMIT dataset [76]. We considered a

frame length of $30ms$ which, considering a sampling rate of $16KHz$ corresponds to frames of 480 samples each. To label the frames as voiced and unvoiced we used an algorithm used for this purpose [77] which takes into account the zero crossing and the signal energy to decide whether a frame is voiced or unvoiced. The order we set for the approximating AR model is $p = 6$ since is the one which resulted in better results. This order is smaller than the ones used in speech coding (typically $p \sim 10$) since it is enough to classify frames in the two classes we consider (voiced/unvoiced).

For a given dataset made of 30000 labeled frames, we used 30% of these frames to train the SVM. We performed the training and the classification in both uncompressed and compressed domain at different compression ratios. As done before, we averaged the results over 100 different realizations.

The results of this experiment are shown in Fig. 6.3. As can be seen, the classification in the original domain reaches 80% of correctly classified signals. The compressive classification reaches 79% of correctly classified signals with a compression ratio of $M/N = 0.26$ and then, as the compression ratio gets closer to 1, it surpasses the uncompressed classification reaching up to 81.3% of correctly classified signals.

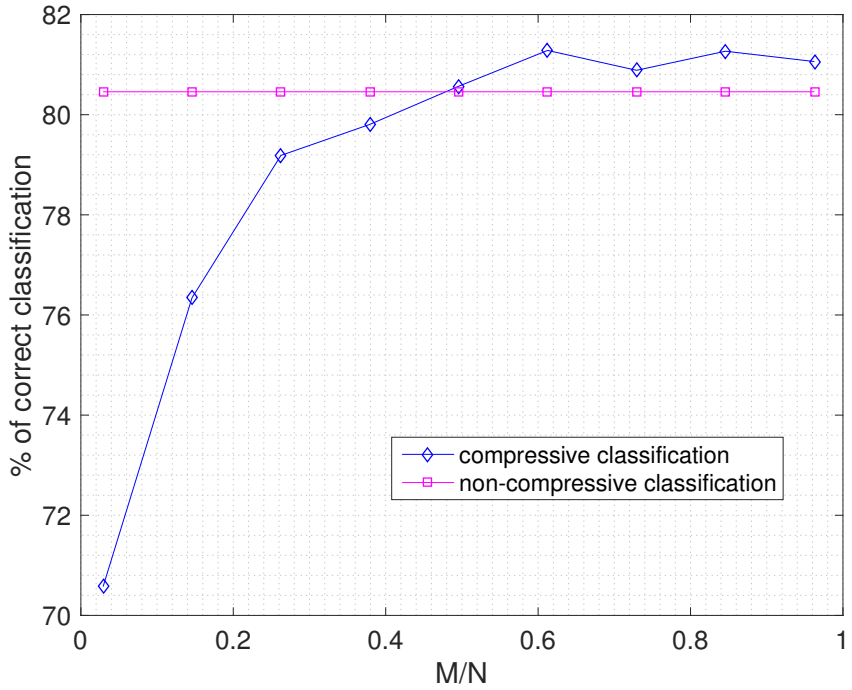


Figure 6.3: Results of the classification task of speech signals in the original and original domains.

The low classification accuracy obtained both in the compressed and original domains can be explained by noting that there is not an actual ground truth, in fact the data has been labeled using an algorithm which is not guaranteed to correctly label all the frames which have been used for this experiment.

However, it is worth noting that again the proposed compressive classification is able to reach and even surpass the classification performed in the original domain. As pointed out before, this can be explained by the fact that an actual ground truth does not exist, but rather the labeling is obtained with a voiced/unvoiced detection algorithm. Moreover, this could also be the result of the denoising property of CS.

6.3.2 Texture classification

The last kind of signal we consider for compressive classification is the texture class. Being able to correctly classify textures is a task which has applications for image recognition and segmentation *e.g.*, determine if a patch extracted from an image represents wood or grass.

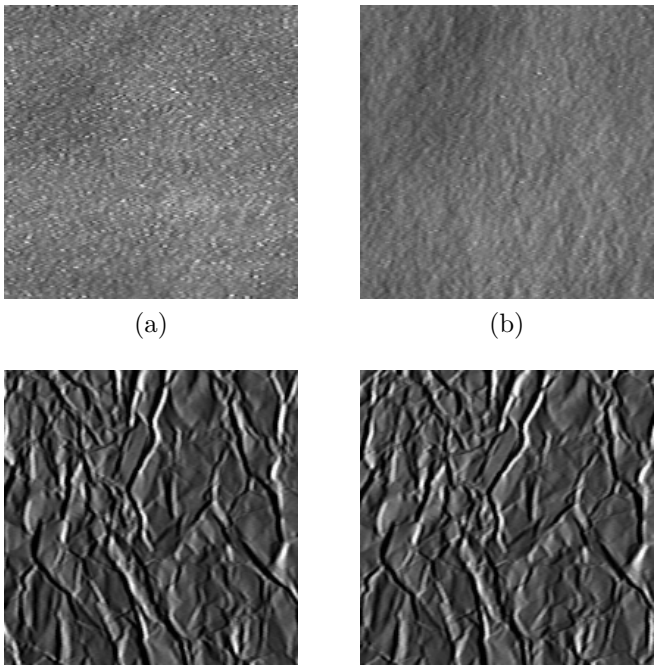


Figure 6.4: Example texture patches from the CUReT database [7]. (a)-(b) belong to class 1, (c)-(d) belong to class 28.

For this experiment we used the texture data from the CUReT database [7] which contains 92 image patches for each of the 61 represented texture classes. We

considered the task of classifying a patch to belong to the class 1 or 28. In Fig. 6.4, example patches for each of the considered classes are depicted.

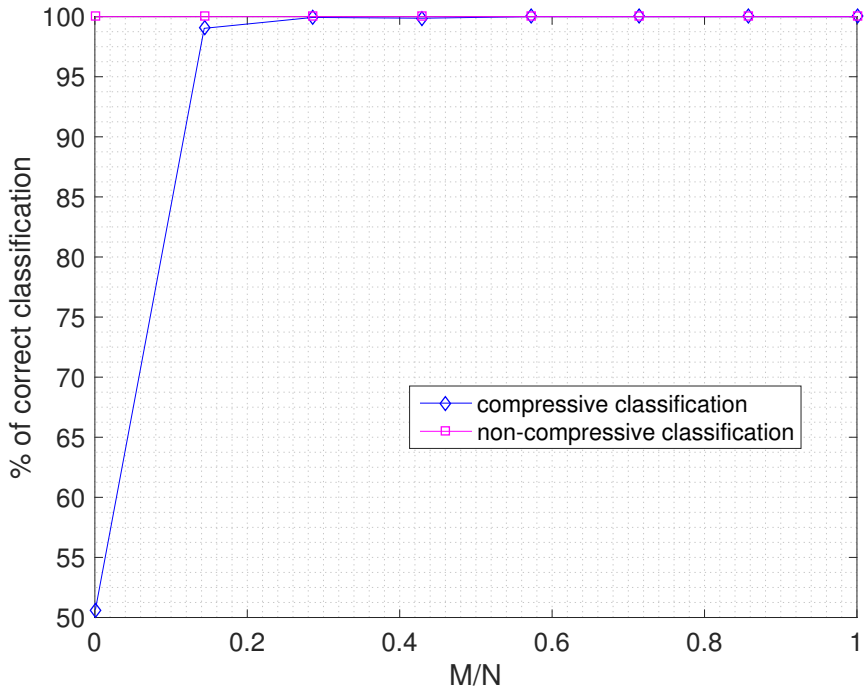


Figure 6.5: Results of the classification task of texture patches in the original and original domains.

Each patch of size 200×200 pixels has been further divided into smaller sub-patches of size 100×100 pixels which have been vectorized according to the zig-zag reordering to be then used for the classification task. The zig-zag re-ordering, rather than raster scan vectorization experimentally showed to lead to superior performance.

For the experiment 350 patches out of 1472 total patches were used for training, the remaining patches were used as test. The choice of the patches to be put in the training and test set has been done using a random permutation, but still keeping an even number of patches for each of the two considered classes in both training/test sets. The order of the AR process we employed for the computation of the feature vectors used in this experiment is $p = 3$. Even though small, this is the value which showed better classification performance.

Then, the results were averaged over 100 realizations to assure consistency in the results.

The classification results are shown in Fig. 6.5. As can be seen, the performance achieved in the original domain is excellent with 100% of correctly classified patches. When the training and classification are performed in the compressed domain, with as low as 14% of measurements with respect to the original signal length, the classification performance is already higher than 99%.

Chapter 7

Distributed covariance estimation

7.1 Introduction

In recent years, wireless sensor networks (WSN) emerged as an inexpensive way to collect spatially distributed data such as temperature or gas concentrations. The aim of WSN is not only that of collecting distributed data, but also that of collaborating to accomplish a task (*e.g.,* estimating the covariance matrix of a process) given that each node has only a portion of the whole data required for the task to be accomplished.

Since signal transmission among nodes in a WSN is limited due to energy consumption of the radio interface, CS which allows to *compress while acquiring* a signal, emerged as a viable solution to reduce the communication load in WSN. Moreover, since the recovery of a compressed signal is computationally expensive and often is not required (*e.g.,* only some parameters of the signals are needed), detection and estimation problems from CS measurements [11] are extremely suitable to be performed directly at nodes. A common assumption is to model the noise affecting the signal with additive white gaussian noise (AWGN), in which case the noise variance is a sufficient statistic to perform signal processing operations. However, when the noise affecting the signal is not AWGN, estimating the covariance matrix of the noise process is fundamental to improve the detection/estimation performance in a variety of inference techniques [78].

The problem of distributed covariance estimation, which has received a lot of attention recently in the non-compressed case, is mainly addressed by the distributed estimation of the principal eigenvectors or eigenvalues of the covariance matrix. Moreover, a large number of techniques proposed in literature tackle this problem using a fusion center to which the nodes send the measurements for processing, which however, it introduces a single point of failure. Differently, the setting we

consider in this Chapter is a fully distributed one with no fusion center and local computations at the nodes. The distributed eigenvector problem is considered in [79] via distributed estimation of the sample covariance matrix followed by local eigen-decomposition. Other authors [80] propose instead algorithms able to directly estimate the smallest (largest) eigenvectors. Ad hoc algorithms for the distributed estimation of the largest eigenvalues have also been proposed as in [81]. Moreover, very few papers in literature consider the distributed architecture when the covariance matrix is that of a noise process having observation matrix Y, whose columns, distributed among the nodes, contains different realizations of the process. This setting is important for detection and estimation tasks because the resulting covariance matrix captures the statistics of the process common to the nodes. This model is considered in [82] where the authors compare the centralized estimation with two different fully distributed approaches based on the average consensus protocol to estimate the largest eigenvector of the covariance matrix of the process.

The drawback of such approaches is the potential increase of the exchanged data among nodes and hence energy consumption. In particular, the aforementioned techniques requires the nodes to exchange (with all the other nodes or with a smaller subset) signals whose length is the same as the one of the acquired signal. In this Chapter we also consider the problem of distributed covariance estimation. Specifically, we aim at developing estimation techniques that require a small communication load to perform the distributed estimation task, thereby requiring much less energy than existing techniques. Consequently, the proposed method differs markedly from existing methods for two main reasons: instead of estimating the eigenvectors of the covariance matrix we estimate the whole covariance matrix but in a parametric fashion in order to reduce the communication load; moreover, we estimate the covariance matrix of a noise process corrupting CS measurements instead of the original signal samples. This latter difference, besides complicating the estimation process, has interesting practical implications. In fact, once the covariance has been estimated, each node can perform signal processing operations on the compressed data, *e.g.,* inference and detection tasks [11] exploiting the knowledge of the noise statistics to obtain improved accuracy. In particular, in this Chapter we propose a new distributed algorithm for the estimation of the covariance matrix of a colored noise process affecting CS measurements using a parametric approach. In fact, we propose to model the colored noise with an autoregressive process (AR) of order p since this model is able to characterize the colored noise process using only few parameters. Hence, given that we model the colored noise with only few parameters, the approach we propose is extremely parsimonious on the communication cost among the nodes. Moreover, the noise covariance matrix estimate obtained with the proposed algorithm can be used to improve the performance of compressive detection and estimation tasks in non-AWGN noise. In order to show this, we derive a compressive detector and assess its performance, showing that with the distributed

estimation of the covariance matrix indeed improves the detection accuracy.

7.2 Preliminaries

7.2.1 Model and assumptions

The topology of the sensor network we are considering throughout this work is represented by a graph $G = (V, E)$. For each node of the network $v \in V$ a signal is acquired according to the model

$$\mathbf{y}^{(v)} = \mathbf{\Phi}^{(v)}\mathbf{x}^{(v)} + \mathbf{n}^{(v)},$$

where $\mathbf{x}^{(v)} \in \mathbb{R}^n$ is a sparse or compressible signal in some domain (*e.g.,* Fourier or DCT) and $\mathbf{\Phi}^{(v)} \in \mathbb{R}^{m \times n}$ with $m < n$ is the sensing matrix: the operator which performs the dimensionality reduction. According to CS theory [10] $\mathbf{\Phi}$ is chosen to be a random sensing matrix whose entries are distributed according to $N \sim (0, \frac{1}{m})$. Then, $\mathbf{\Phi}^{(v)}\mathbf{x}^{(v)}$ can be approximated as white noise [83]. The vector $\mathbf{n}^{(v)} \in \mathbb{R}^m$ is assumed to be a colored noise process that corrupts the linear measurements. In this work, the colored noise is approximated with a parametric model, *i.e.,* an autoregressive process of a order p, denoted as AR(p). The goal of this work is to perform distributed estimation of the covariance matrix of the noise given that each node is corrupted by a different realization of the same noise process. Having the colored noise n approximated with an AR(p) process, we can formally define the time-varying nature of such a process as $\mathbf{n}_t = \sum_{i=1}^{p} \mathbf{n}_{t-i}\mathbf{a}_i + \mathbf{w}_t$ being $\mathbf{a} = [\mathbf{a}_1 \ \ldots \mathbf{a}_i \ \mathbf{a}_p]^\top$ the coefficients of the regression and $\mathbf{w}$ the driving noise process. Then the covariance matrix of the process can be written as $\mathbf{C} = \mathbf{A}^{-1}\mathbf{A}^{-\intercal}$, where $(\cdot)^{-\intercal}$ denotes the transpose of the inverse operator, $\mathbf{A} = \text{tril}(\text{toeplitz}(\mathbf{u}))$ and $\mathbf{u} = [1 \ \mathbf{a} \ 0 \ \ldots \ 0]^\intercal \in \mathbb{R}^m$. In order to estimate $\mathbf{C}$ we need to estimate the parameter vector $\mathbf{a}$. Although many estimators have been proposed in literature [84], our work relies on the least squares estimator as it allows us to obtain closed form expression for the covariance estimator.

7.2.2 Unbiased least-squares estimate

For notation simplicity let us drop the superscript (v) and consider a single node in this subsection. Given a realization $\mathbf{n} \in \mathbb{R}^m$ of an AR(p) process and the associated regression vector $\mathbf{n}_t^\intercal = [\mathbf{n}_{t-1} \ldots \mathbf{n}_{t-p}]$, the t-th sample can be written as $\mathbf{n}_t = \mathbf{n}_t^\intercal a + \mathbf{w}_t$, where $\mathbf{w}_t \sim N(0, \sigma_v^2)$ is the t-th sample of the driving noise process vector $\mathbf{w}$.

The least-squares estimator of the regression coefficient vector $\mathbf{a}$ is then given by

$$\mathbf{a}_{LS} = \min_{\mathbf{a}} \|\mathbf{n} - \mathbf{N}\mathbf{a}_{LS}\|_2^2 \text{ where } \mathbf{N} = \begin{bmatrix} \mathbf{n}_{p+1}^\intercal \\ \vdots \\ \mathbf{n}_m^\intercal \end{bmatrix}. \tag{7.1}$$

By means of the bias compensation principle [85], the least-squares estimate can be decomposed in the unbiased estimate term $\mathbf{a}$ and the bias term as

$$\mathbf{a} = \mathbf{a}_{LS} + \sigma_w^2 \mathbf{R}^{-1}\mathbf{a}, \tag{7.2}$$

where $\mathbf{R} = \sum_{t=1}^{m} \mathbf{n}_t \mathbf{n}_t^\top$ and $\sigma_w^2 = \text{var}(\mathbf{\Phi x})$. As we can see from (7.2), there are two unknowns: $\mathbf{a}$ and σ_w^2. Therefore we have developed an iterative algorithm based on alternating the estimation of the unknowns. The distributed algorithm we introduce in Section 7.3.1 unbiases the least-squares estimate relying on the technique developed in [85], where the authors obtain the unbiased estimate $\hat{\mathbf{a}}_{ILS}$ according to Algorithm **??**.

Algorithm 2 Iterative AR(p) LS estimate unbiasing

1: $k^2 = \frac{\sigma_v^2}{\sigma_w^2}$
2: $\hat{J}(\mathbf{a}_{LS}) = \sigma_w^2(k^2 + 1 + \mathbf{a}_{LS}^\intercal \mathbf{a})$
3: **while** not stopIter **do**
4: $\quad \hat{\sigma}_w^2(i) \leftarrow \frac{\hat{J}(\hat{\mathbf{a}}_{LS})}{k^2+1+\hat{\mathbf{a}}_{LS}^\intercal \hat{\mathbf{a}}_{ILS}(i-1)}$
5: $\quad \hat{\mathbf{a}}_{ILS}(i) \leftarrow \hat{\mathbf{a}}_{LS} + \hat{\sigma}_w^2(i)\mathbf{R}^{-1}\hat{\mathbf{a}}_{ILS}(i-1)$
6: $\quad \hat{\sigma}_n^2(i) \leftarrow k^2 \hat{\sigma}_w^2(i)$
7: **end while**

7.3 Distributed covariance estimation algorithm

7.3.1 Proposed method

As introduced in section 7.2.1, the estimation of the covariance matrix can be reduced to the estimation of the common parameter vector a of the AR(p) process realizations among the nodes. The idea of using a parametric representation for the covariance matrix of the process allow us to distribute the estimation task keeping the communication cost low. Namely, at each iteration the nodes only need to exchange with their neighbors a small parameter vector of length $p \ll m$. To do so, we split the problem into sub-problems to be iteratively solved at each node, in such a way that at the end of the iterations each node v has a consistent estimate of the covariance matrix $\hat{\mathbf{C}}$ across the whole network.

Let us start by defining a global functional in which we have a term related to the node-dependent least-squares estimation of the AR parameters $\hat{\mathbf{a}}_{LS}^{(v)}$ and a second term, derived from (7.2), for bias removal which contains the vector variable $\mathbf{a}$ that is common to all nodes. In fact, since the noise component affecting the nodes corresponds to a different realization of the same noise process, the consensus is on the parameters $\mathbf{a}^{(v)} \forall v \in V$ that are the local *unbiased* coefficients vector estimates. Since the local least-squares estimates at each node are biased according to the compressed measurements the node has acquired, this parameter is updated and kept local. Hence, the functional $\mathcal{F}$ is defined as:

$$\mathcal{F}(\mathbf{a}, \mathbf{a}_{LS}^{(1)}, \dots, \mathbf{a}_{LS}^{(v)}, \dots, \mathbf{a}_{LS}^{(|V|)}) = \min \sum_{v \in V} f^{(v)}(\mathbf{a}, \mathbf{a}_{LS}^{(v)}) = \tag{7.3}$$

$$= \min \sum_{v \in V} \underbrace{\|\mathbf{y}_{+}^{(v)} - [\mathbf{0}\ \mathbf{Y}^{(v)}]\bar{\mathbf{a}}^{(v)}\|^2}_{\text{least squares term}} + \lambda \underbrace{\|[(\mathbb{I} - \sigma_w^{2\,(v)}\mathbf{R}^{-1(v)}) - \mathbb{I}]\ \bar{\mathbf{a}}^{(v)}\|^2}_{\text{unbiasing term}},$$

given $\bar{\mathbf{a}}^{(v)} = [\mathbf{a}_{LS}^{(v)}\ \mathbf{a}]^\intercal$, $\mathbf{y}_{+}^{(v)} = [\mathbf{y}_{p+1}^{(v)}\ \dots\ \mathbf{y}_{m}^{(v)}]$ and $\mathbf{Y}^{(v)} = [\mathbf{y}_{p+1\to 1}^{(v)\intercal}\ \dots\ \mathbf{y}_{m\to m-p+1}^{(v)\intercal}]^\intercal$, where $\mathbf{y}_{a\to b} \triangleq [\mathbf{y}_a\ \mathbf{y}_{a-1}\ \dots\ \mathbf{y}_{b+1}\ \mathbf{y}_b]$. In particular, the first term of the functional, which only updates the least-squares component $\mathbf{a}_{LS}^{(v)}$ of the vector $\bar{\mathbf{a}}^{(v)}$, is equivalent to (7.1) as its solution corresponds to the AR(p) least-square estimate. The second term instead links the local least-squares estimates with the unbiased estimate $\mathbf{a}$ (that is common to all nodes) thought $\sigma_w^{2\,(v)}$ and $\mathbf{R}^{-1(v)}$. Hence, this latter term is defined as the least-squares solution of the bias equation (7.2). Moreover, a regularization parameter λ is used to weight the second term. To distribute the functional $\mathcal{F}$ we use the subgradient consensus method proposed in [86], where each node v alternates between a gradient descent step towards the minimum of the function $f^{(v)}$ and a consensus step. Since, as already pointed out, by nature of the noise process all the nodes share the same unbiased coefficients, the consensus step is performed on the coefficients $\mathbf{a}^{(v)}$. Therefore, in order to perform the consensus, each node v only needs to exchange a vector made of p parameters with its N_v neighbors corresponding to the coefficients vector $\mathbf{a}^{(v)}$. The resulting step to be performed at each node at each iteration is then:

$$\mathbf{a}^{(v)}(i+1) = \sum_{j=1}^{|V|} [\mathbf{W}]_{vj}^{h} (\mathbf{a}^{(v)}(i) - \tau \mathbf{g}^{(v)}(\mathbf{a}^{(v)}(i))), \tag{7.4}$$

where $\mathbf{g}^{(v)}(\mathbf{a}^{(v)}) = \partial f^{(v)}(\mathbf{a}^{(v)}(i))$, τ is the gradient descent step size and the notation $[\mathbf{W}]^h$ indicates that h consensus iterations are performed according to the adjacency matrix $\mathbf{W}$. However, computing the step in (7.4) requires the knowledge of $\sigma_w^{2\,(v)}$ due to its dependency on $\mathbf{g}^{(v)}$. Since this variable is assumed to be unknown, it is iteratively estimated according to step 5 of the unbiasing algorithm described in Algorithm **??**. The distributed algorithm procedure is summarized in Algorithm **??**.

Once the algorithm has reached the convergence, namely $\|\hat{\mathbf{a}}^{(v)}(i)-\hat{\mathbf{a}}^{(v)}(i-1)\|_2 < \alpha$ given the arbitrary parameter $\alpha \ll 1$, each node can hence locally build the estimate of the covariance matrix $\hat{\mathbf{C}}^{(v)}$.

The parameter h specifies the number of the consensus iterations of the algorithm and it is a trade-off parameter between consistency of the estimates and communication cost across the network. More formally, the communication cost (expressed in exchanged data samples) can be written as

$$D = h(|V||N_v|Rp),$$

where $|N_v|$ is constant and R is the number of iterations of the algorithm. As we experimentally show in Sec. 7.4, $h = 2$ is a good compromise between estimation quality and communication cost.

Algorithm 3 Distributed parametric covariance matrix estimation

Intialize:
$\bar{\mathbf{a}}^{(v)} \leftarrow \mathbf{0}$
${\sigma_w}^{2(v)} \leftarrow var(\mathbf{y}^{(v)})$
$k^{2(v)} = \frac{\sigma_v^2}{\sigma_w^{2(v)}}$
while not stopIter **do**
For each node $v \in V$
$\bar{\mathbf{a}}^{(v)}(i+1) \leftarrow \bar{\mathbf{a}}(i) - \tau \mathbf{g}(\bar{\mathbf{a}}^{(v)}(i), \sigma_w^{2\,(v)}(i))$
$\hat{\sigma_w}^{2(v)}(i+1) \leftarrow \frac{\hat{J}(\bar{\mathbf{a}}^{(v)}(i))}{k^{(v)2}+1+\mathbf{a}_{LS}^{\intercal}{}^{(v)}(i)\mathbf{a}(i)^{(v)}}$
$\mathbf{a}^{(v)}(i+1) = \sum_{v\in N_v} \mathbf{W}\mathbf{a}^{(v)}(i)$
$\bar{\mathbf{a}}^{(v)}(i+1) = [\mathbf{a}^{(v)}(i+1)\ \mathbf{a}_{LS}^{(v)}(i+1)]^{\intercal}$
end while
$\hat{\mathbf{A}}^{(v)} \leftarrow \text{tril}(\text{toeplitz}([1\ \mathbf{a}^{(v)}\ 0\ \dots\ 0]))$, $\hat{\mathbf{C}}^{(v)} \leftarrow \hat{\mathbf{A}}^{(v)-1}\hat{\mathbf{A}}^{(v)-\intercal}$

7.3.2 Convergence

Although in this work we are not presenting a proof of convergence of the proposed algorithm, we discuss its convergence properties by separately analyzing the two directions in which the algorithm moves that are: the subgradient consensus step and the variance estimation. At first, let us assume that $\hat{\sigma}_w^2$ is known at each node. Then the following theorem can be stated:

Theorem 1. *Having the sequence* $\{\bar{\mathbf{a}}(i)^{(1)}, \dots, \bar{\mathbf{a}}(i)^{(|V|)}\}_{i=0}^{\infty}$ *generated by algorithm 2, with* $h \geq (\log(\beta) - \log(4NM(\beta+\alpha C)))/\log(\gamma)$ *and* $f^* > -\infty$ *we have that:*

$$\lim_{i\to\infty} \inf f(\mathbf{a}(i)^{(v)}) \leq f^* + \alpha NC^2/2 + 3nc\beta\ , \forall v \in V$$

Proof. This theorem comes from Th. 1 in [86], we hence need to prove that in our case all the related assumptions are satisfied. Assumption 1 is satisfied since we have that $\|\mathbf{g}^{(v)}(\mathbf{a})\| \leq C = (2\|\mathbf{Y}^\intercal\mathbf{Y}\| + \|\lambda\mathbf{S}^\intercal\mathbf{S}\|)\|\mathbf{a}\| + \|2\mathbf{Y}^\intercal\mathbf{y}^+\|$ with $\mathbf{S} = [(\mathbb{I} - \sigma_w^{2\,(v)}\mathbf{R}^{-1(v)}) - \mathbb{I}]$, then if $\|\mathbf{a}\| \leq \infty$ we have that $C \leq \infty$. Assumption 2 is satisfied for the topology we are mainly considering throughout this Chapter: the ring topology. Lastly, Assumption 3 is satisfied since the problem we are considering is unconstrained. □

In the proposed algorithm we are assuming that σ_w^2 is not known at each node and hence must be estimated. The unbiasing estimator given by the Algorithm 1 is proven to converge in [85] to the unbiased estimate and hence σ_w^2 converges to $\sigma_w^{2\,*}$. Differently from this estimator, the proposed algorithm does not have the knowledge of the true least squares estimate $\mathbf{a}_{LS}^{(v)}$. It has instead, at each step, a local estimate of $\mathbf{a}_{LS}^{(v)}$ given by the subgradient step which is going towards the optimum $\mathbf{a}_{LS}^{(v)*}$. As we numerically show in the next section, the proposed algorithm shows empirical convergence. Intuitively we can hence say that, even though the convergence is not proved when σ_w^2 is not known in advance, the alternated estimation of least squares term and the noise variance, tends to go towards the optimum of the functional defined in (7.3).

7.4 Experimental results

7.4.1 Estimation of AR coefficients

For the purpose of numerical evaluation of the proposed algorithm, at first we consider the estimation error and the consensus reached on the parameters of the AR process in a given network after a suitable number of iterations. The considered network is arranged in a ring topology and is given by the graph $\mathcal{G}$ with $|V| = 10$, and $|N_v| = 2$. Each node acquires a signal $\mathbf{y}^{(v)} = \mathbf{\Phi}^{(v)}\mathbf{x}^{(v)}$ where $\mathbf{\Phi}^{(v)}$ are gaussian random sensing matrices whose entries are drawn from $N \sim (0, \frac{1}{m})$ and $\mathbf{x}^{(v)}$ are sinusoidal signals (sparse in frequency domain) for which different signal to noise ratios are taken into account. The length of $\mathbf{x}^{(v)}$ is $n = 800$, and that of $\mathbf{y}^{(v)}$ is $m = 200$. For the experiments, along with AR noise, we also considered pink noise. This kind of noise, which is a non-white gaussian noise, exhibits a decreasing spectrum and is present in many physical, biological and economical systems. Hence, for this experiment we considered both synthetic AR noise of order $p = 3$ and pink noise which is approximated with an AR process. Since from our experiments the pink noise is well approximated with an AR process of third order, we choose $p = 3$. When considering pink noise, since the true AR coefficients are not available, the relative error we consider is given by $\frac{\|\mathbf{a}^* - \hat{\mathbf{a}}^{(v)}\|}{\|\mathbf{a}^*\|}$ where $\mathbf{a}^*$ is the estimate obtained in

a centralized fashion. The results in Fig. 7.1 show that a relatively small number of exchanged data samples (160000 samples) are sufficient to reach very low errors on the estimates $\hat{\mathbf{a}}^{(v)}$. Along with a low estimation error, we also show that the local estimates at each node reach consensus. More in detail, for the AR process with SNR $= -3$dB, the maximum distance among the nodes reduces down to $9e{-}3$ as the number of iterations are bigger than 80.

Since the number of consensus iterations h is a trade-off between closer estimates and higher communication cost, we experimentally evaluate the role of the parameter h. We can see from Fig. 7.2 that, given a fixed error on the AR coefficients estimates to be reached, increasing this consensus parameter leads to closer estimates but at a higher communication cost. For the experiments we show in the following sections, we chose $h = 2$ as it is a good trade-off for keeping the communication cost low while maintaining accurate estimates across the network.

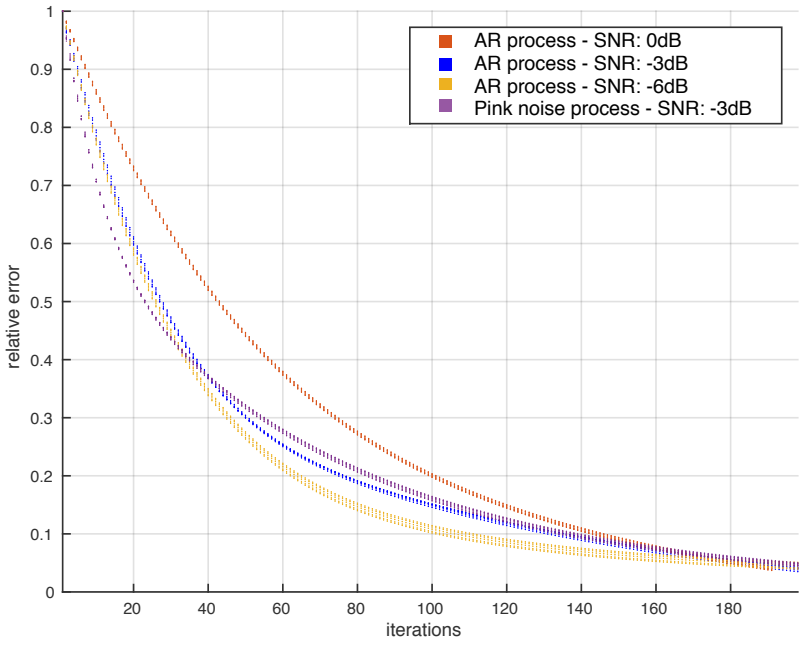


Figure 7.1: Relative error $\frac{\|\mathbf{a}-\hat{\mathbf{a}}^{(v)}\|_2}{\|\mathbf{a}\|_2}$ $\forall v \in V$. The vertical bars at each iteration corresponds to the range of errors of all the nodes in the network at the given iteration.

7.4.2 Estimation of covariance matrix

Starting from the good results obtained for the AR parameters estimates, in this section we present some results on the estimated covariance matrix in order to assess the performance of the proposed technique. The distance metric we use to compare the estimate is the Forstner distance [87], a widely used method for evaluating the

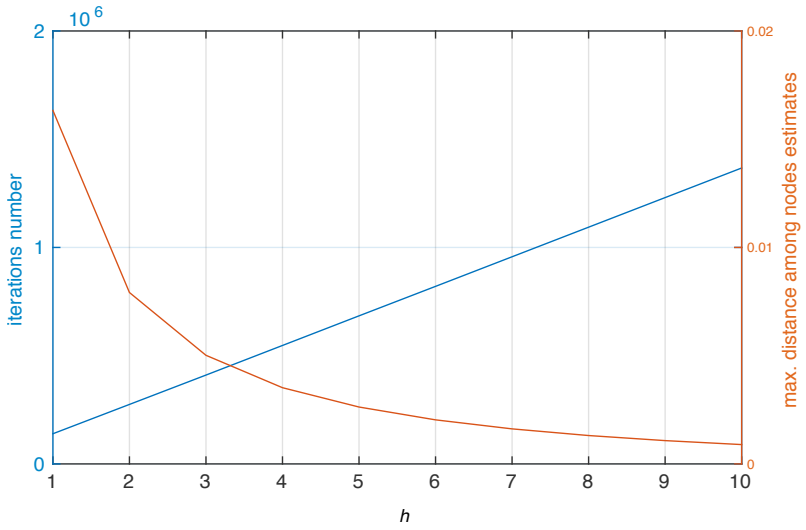


Figure 7.2: Trade-off curve for different values of consensus parameter h.

distance of two positive semi-definite covariance matrices. It is defined as $d(\mathbf{A}, \mathbf{B}) = \mathrm{tr}(\ln^2(\sqrt{\mathbf{A}^{-1}}\mathbf{B}\sqrt{\mathbf{A}^{-1}}))$, where in our setting $\mathbf{A}$ is $\mathbf{C}$, the sample covariance matrix of the process (averaged over 5000 realizations), and $\mathbf{B}$ is the local estimate of the covariance $\hat{\mathbf{C}}^{(v)}$. As shown in Fig. 7.3, the distance between the matrices $\mathbf{C}$ and $\hat{\mathbf{C}}^{(v)}$ decreases as the proposed algorithm iterates, moreover the local distances (*i.e.*, computed using the nodes local estimates) tend to converge. Then, we compare our technique with the one proposed in [82] for the estimation of the eigenvector associated with the largest eigenvalue of the covariance matrix in a distributed setting. More in detail, among the techniques the authors propose, we use the technique *adapt then combine* that, according to their experiments, leads to better results. The setting we use for the comparison is the same as described above, and the number of iterations for the compared algorithm is fixed to 200. The metric used in this experiment is the angle between the principal axes of the true and estimated covariance matrices.

In Table 7.1 we summarize the results of the comparison. As can be seen, the proposed technique leads to lower angles between the principal axes while keeping the communication cost extremely low. Moreover, the proposed technique estimates the whole covariance matrix of the noise process instead of just the principal eigenvector and hence it allows a wider range of applications.

7.4.3 Compressive detection in distributed setting

The proposed algorithm allows us to estimate in a distributed fashion the covariance matrix of a non-white noise process keeping the communication cost very low. Hence,

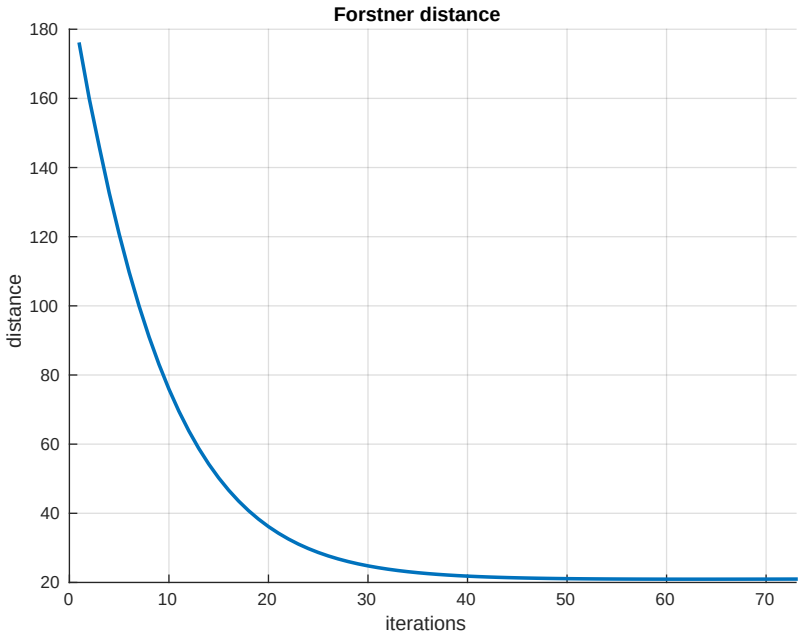


Figure 7.3: Forstner distance between $\hat{\mathbf{C}}^v$ and $\mathbf{C}$ computed for each estimate $\hat{\mathbf{C}}^{(v)} \forall v \in \mathbf{V}$. This figure only shows the Forstner distance up to 70 iterations since in the remaining iterations the distance does not change having reached convergence.

Table 7.1

	exchanged data (samples)	angle (rad)
Proposed algorihtm	160000	0.24
Ghadban *et al.* [82]	2000000	0.45

we further validate the proposed technique by including it in a compressive signal processing task. More in detail we consider the detection of a compressed signal given its CS measurements corrupted by additive colored noise as introduced in [11]. Similarly to Sec. 7.4.1, let us assume to have a wireless sensor network represented by the graph $\mathcal{G}$ arranged in a ring topology with $|V| = 10$. Then, we assume that each node v acquiring a CS signal is then connected to its neighbors given $|N_v| = 2$. We are now interested in the signal detection problem at node $v^\star$, more in detail we want to distinguish between the hypotheses:

$$\begin{cases} \mathcal{H}_0 & : \mathbf{y}^{(v^\star)} = \mathbf{n}^{(v^\star)} \\ \mathcal{H}_1 & : \mathbf{y}^{(v^\star)} = \mathbf{\Phi}^{(v^\star)}\mathbf{x}^{(v^\star)} + \mathbf{n}^{(v^\star)}, \end{cases}$$

where $\mathbf{\Phi}^{(v^\star)}$, $\mathbf{x}^{(v^\star)}$ are known, and $\mathbf{n}^{(v^\star)} \sim \mathcal{N}(0, C)$ is the colored noise with unknown covariance matrix $\mathbf{C}$.

In order to improve the performance of the detector, the knowledge of the noise statistics is required. In particular we need the covariance matrix of the noise process. Hence, assuming that the node $v^\star$ needs an estimate of the covariance matrix of the noise process to accurately solve the detection problem, it runs together with its neighbors the distributed covariance estimation algorithm described in Algorithm 2. From now on, assuming that the node $v^\star$ has obtained the estimate $\hat{\mathbf{C}}^\star$, we will drop the superscript "$\star$" to improve readability.

Relying on standard detection theory [78] we can show that, in our setting, the *Neyman-Pearson* (NP) optimal detector, namely the likelihood ratio test, can be written as $t = \mathbf{y}^\intercal \hat{\mathbf{C}}^\intercal \mathbf{\Phi}\mathbf{x}$. Then, denoting the probability of false alarm as P_F and that of detection as P_D, it can be shown that the P_D in function $P_F = \alpha$ is given by:

$$P_D = Q\left(\frac{\sqrt{V_0}Q^{-1}(\alpha) + \mu^\intercal \hat{\mathbf{C}}^{-1}\mathbf{\Phi}\mathbf{x} - E_1}{\sqrt{V_1}}\right), \tag{7.5}$$

where

$$\begin{aligned}
\mu = \mathbb{E}[n], V_0 &= \sigma^2 \mathbf{x}^\intercal \mathbf{\Phi}^\intercal (\hat{\mathbf{C}}\hat{\mathbf{C}}^\intercal)^{-1}\mathbf{\Phi}x - (\mu^\intercal \hat{\mathbf{C}}^{-1}\mathbf{\Phi}\mathbf{x})^2, \\
E_1 &= \mathbf{x}^\intercal \mathbf{\Phi}^\intercal \hat{\mathbf{C}}^{-\intercal}(\mathbf{\Phi}x + \mu^\intercal), \\
V_1 &= \mathbf{x}^\intercal \mathbf{\Phi}^\intercal \hat{\mathbf{C}}^{-\intercal}\mathbf{\Phi}\mathbf{x} - \mathbf{x}^\intercal \mathbf{\Phi}^\intercal \hat{\mathbf{C}}^{-\intercal}\mu^\intercal \mu \mathbf{C}^{-\intercal}\mathbf{\Phi}\mathbf{x}.
\end{aligned}$$

We hence show the receiver operating characteristics (ROC) curves for the detection problem we introduced. We compare the covariance matrix estimate $\hat{\mathbf{C}}$ with those obtained by assuming the noise to be white and hence having no knowledge of noise statistics averaging the results over 50 different runs of the algorithm. Using the estimated covariance $\hat{\mathbf{C}}$, we show both theoretical results in (7.5) and experimental ROC curves obtained by running 500 Monte Carlo (MC) tests. Then, we compare them with those obtained by a standard detector assuming the noise to be white, hence being unaware of the noise statistics.

The results we present are for two different compression ratio values, namely $\frac{m}{n} = \{0.11, 0.22\}$. Moreover, the SNR we consider is extremely low SNR $= -19$dB, this choice is made to show the detection performances for extremely noisy signals, using the proposed setup. It is worth noting that when higher SNRs are considered, the ROC curve reaches the optimality. As we can see from Fig. 7.4, the compressive signal detection using the estimated covariance matrix according to the proposed method, outperforms a compressive signal detection algorithm unaware of the noise covariance matrix. Moreover it can be seen that even though the number of measurements m and the SNR value are very low, the detection task is efficiently solved. Lastly, we can see that the experimental results confirm the theoretical performance.

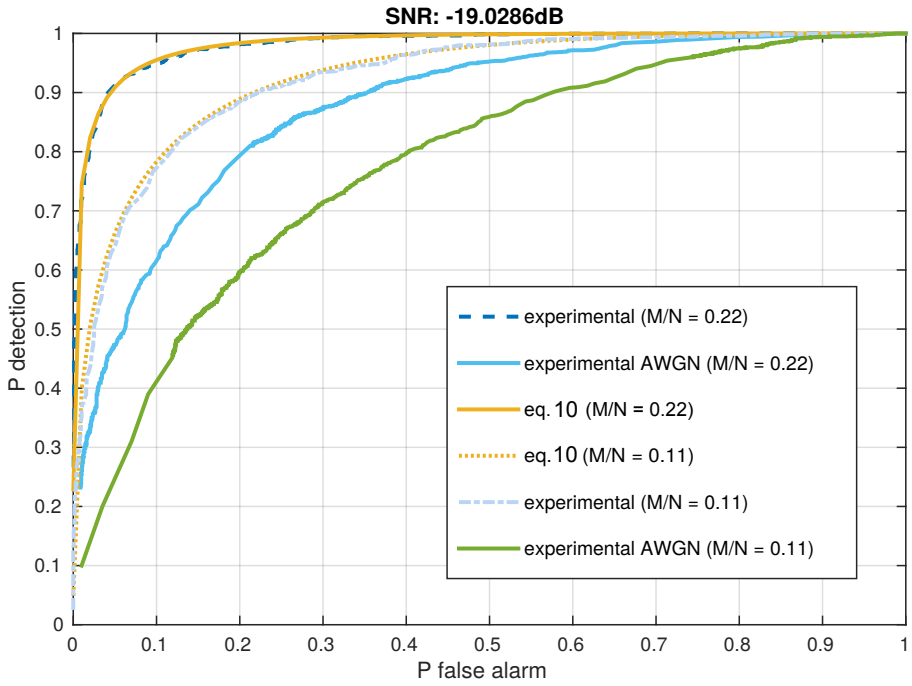


Figure 7.4: ROC performance comparison of noise unaware signal detection (AWGN) and using the proposed covariance matrix estimate.

Chapter 8

Bayesian KSVD

Heretofore we discussed how to extract information from CS measurements and how to eventually improve the recovery process. In this Chapter, we focus instead on the sparse representation which is one of the key components of the CS. In fact, being able to efficiently sparsify a signal may lead to better recovery with less measurements. As previously discussed, the sparser a signal, the smaller the number of required measurements to achieve a good signal recovery. In particular, in this Chapter we mainly focus on non-CS sparse representations before moving, in Chapter 9, towards CS dictionary learning.

Let us now introduce the dictionary learning problem in a more formal way: we aim to find a sparse representation of each signal in a database of Q natural signals to $\mathbb{R}^P$ concatenated column-wise into a matrix as $\mathbf{Y} = [\mathbf{y}_1, \cdots, \mathbf{y}_Q] \in \mathbb{R}^{P \times Q}$. We do this by finding a set of K atoms in the signals' ambient space, concatenated into a dictionary matrix $\mathbf{D} = [\mathbf{d}_1, \cdots, \mathbf{d}_K] \in \mathbb{R}^{P \times K}$. This dictionary, and the corresponding assignment matrix $\mathbf{X} = [\mathbf{x}_1, \cdots, \mathbf{x}_Q] \in \mathbb{R}^{K \times Q}$ for the signals, are recovered by solving an optimization problem where we seek the best reconstruction of our signals given a budget T for the number of non-zero entries allowed in each column of $\mathbf{X}$. Formally this problem takes the form

$$\begin{aligned} &\min_{\mathbf{D},\mathbf{X}} \|\mathbf{Y} - \mathbf{D}\mathbf{X}\|_{\mathrm{F}}^2 \\ &\text{s.t. } \|\mathbf{x}_q\|_0 \leq T, \;\; q = 1, \ldots, Q, \end{aligned} \tag{8.1}$$

where $\|\cdot\|_0$ denotes the ℓ_0-(pseudo)norm, which counts the non-zero entries in a vector, and $\|\cdot\|_{\mathrm{F}}$ denotes the Frobenius norm.

Since the objective function $\|\mathbf{Y} - \mathbf{D}\mathbf{X}\|_{\mathrm{F}}^2$ is not convex in $\mathbf{X}$ and $\mathbf{D}$ jointly, but biconvex, that is, convex in $\mathbf{X}$ and $\mathbf{D}$ individually, this problem can be addressed by alternating minimization over each variable separately. However, the exact minimization over $\mathbf{X}$ is well known to be NP-hard. Therefore greedy methods, among which the popular K-SVD algorithm [2], are used to approximate the true solution. Alternatively, the sparsity constraint can be relaxed, resulting in the following

problem

$$\begin{aligned} &\min_{\mathbf{D},\mathbf{X}} \|\mathbf{Y}-\mathbf{D}\mathbf{X}\|_{\mathrm{F}}^2 \\ &\text{s.t. } \|\mathbf{x}_q\|_1 \leq T, \;\; q=1,\ldots,Q, \end{aligned} \tag{8.2}$$

where $\|\cdot\|_1$ denotes the vector ℓ_1-norm. A wide array of techniques from convex optimization can be applied to solve this problem (e.g., [2, 88–90]).

An alternative approach to the problem is studied in the work by Skretting and Engan [91], Recursive Least Squares Dictionary Learning Algorithm (RLS-DLA), which performs a continuous update of the dictionary after each training vector is processed. Therein lies the main difference between RLS-DLA and other previous approaches such as its precedent ILS-DLA [92] or K-SVD [2]. However, its convergence has not been established.

Along with deterministic methods to solve the dictionary learning problem, other authors proposed a probabilistic approach. In their seminal works, Olshausen and Field [20] and Lewicki and Sejnowski [93] introduced a generative model for the data which allowed them to develop a Maximum Likelihood (ML) estimator for both the sparse coding and the dictionary. According to this model, when the prior on the sparse signal is a heavily peaked Laplacian distribution around zero and the residual is approximated by a zero-mean Gaussian distribution, the dictionary learning problem reduces to the one in (8.2). Following this work, other authors proposed modifications to either the sparse approximation step, the dictionary update, or both. In [94], using the same generative model introduced in [20], the authors proposed the use of Orthogonal Matching Pursuit (OMP) to solve the sparse coding problem and a closed form solution for the dictionary update equation. Later papers focused on the use of a Maximum a Posteriori (MAP) approach instead, which allows to impose constraints on the dictionary as well. For instance, in the work of Kreutz-Delgado *et al.* [95] a unit-norm Frobenius prior is placed on the dictionary. However, due to the intractability of such a prior, they propose to use an approximate solution and the FOCUSS [96] algorithm in order to obtain the sparse solution. Other choices of priors involve smoother (less sparse) priors based on the Kullback-Leibler divergence for the ℓ_1 regularization as in [97]. The advantage of this latter approach lies in the increased stability of the sparse solution and the efficient convex inference.

All of the aforementioned techniques use ML or MAP estimators to solve the dictionary learning problem. However, the main drawback of such approaches is that they do not take into account the uncertainty of the estimated sparse representation coefficients, which, as we will later examine, leads to reduced algorithmic performance.

Moreover, since the variance of the noise is not explicitly taken into account in the model, these algorithms have to rely on other techniques for noise estimation. The importance of having a good estimate of the noise variance is discussed in [89] where

the authors show that when using K-SVD for image denosing [21], the resulting PSNR is highly affected by the precision of the noise variance estimate.

To overcome these problems a few techniques have been developed. These include the incorporation of the noise variance/covariance information in the model as a parameter that can be estimated and taking into account the uncertainty of the estimates. The author in [98] proposed an Expectation Maximization (EM) algorithm in which the posterior of the sparse signal x is estimated along with the dictionary. In more detail, each column of X is modeled using a Laplacian prior which, however, leads to an intractable posterior distribution, for which the authors propose to use a variational approximation of the prior which tranforms the posterior of the sparse signal into a Gaussian form. Finally, an EM algorithm is developed in order to estimate the parameters of the model. However, with this approach, the authors do not place a prior on the entries of the dictionary.

Following a similar strategy, but introducing more complex models, Zhou *et al.*, [89] and [99], introduced a beta-Bernoulli prior for the selection of the active-set; that is, the smallest possible set of atoms in the dictionary which is capable of efficiently explaining the underlying signal structure. In addition, they introduced a Dirichlet patch clustering in order to cluster the data which have the same probability of being represented using a fixed set of atoms. Finally, the full posterior distribution is estimated through Gibbs sampling. In order to deal with large data sets, the same model was also adapted to process randomly partitioned data [100]. In more detail, the parameters of the model are inferred locally for each set of partitioned data and then aggregated using a moving average to generate the global parameters of the model resulting in an increased robustness to local minima and reduced memory requirements.

In this Chapter we propose a novel Bayesian algorithm for solving ℓ_1 dictionary learning problems. Our approach aims at estimating the whole posterior distribution of $\mathbf{X}$ (thus taking into account the uncertainty of the estimated coefficients) but with an automatic technique for the estimation of the parameters which originates with the introduced models. The proposed approach is applied to image denoising and inpainting in order to test its performance in different applications of interest in image processing.

Notation: In order to improve readability, we discuss the notation we are going to use throughout this chapter. For a matrix $\mathbf{X}$, its ith column and jth row are denoted by $\mathbf{x}_i$ and $\mathbf{x}^j$, respectively. The (i,j)-th entry of a matrix $\mathbf{X}$ is denoted by either x_{ij} or $\mathbf{X}(i,j)$, whichever makes the notation clearer. Given a vector $\mathbf{x}$, $\mathrm{diag}(\mathbf{x})$ represents the square matrix with the entries of $\mathbf{x}$ on its diagonal, while given a square matrix $\mathbf{X}$, $\mathrm{diag}(\mathbf{X})$ extracts its diagonal into a vector. Given a square matrix $\mathbf{X}$, $\mathrm{Tr}(\mathbf{X})$ and $|\mathbf{X}|$ denote the trace and determinant operators, respectively. The $M \times 1$ all-zero vector is denoted by $\mathbf{0}_M$, and finally, the $M \times M$ identity matrix is denoted by $\mathbf{I}_M$.

8.1 The K-SVD algorithm

Among the most popular algorithms for dictionary learning, K-SVD [2] is a greedy approach that approximately solves the standard ℓ_0 problem in (8.1). In K-SVD the optimization is performed coordinate-wise alternating between $\mathbf{X}$ and $\mathbf{D}$.

At each iteration of the K-SVD algorithm, given the current state update of the dictionary, the Orthogonal Matching Pursuit (OMP) algorithm [101] is first applied to determine the support of $\mathbf{X}$, i.e., the locations of the non-zeros in $\mathbf{X}$, while the values at these non-zero locations obtained from OMP are discarded. Notice that this requires manually fixing the number of non-zero components in each column of $\mathbf{X}$.

After OMP, $\mathbf{D}$ and the non-zeros of $\mathbf{X}$ are updated. The term $\mathbf{DX}$ can be decomposed as

$$\mathbf{DX} = \sum_{k=1}^{K} \mathbf{d}_k \mathbf{x}^k \, .$$

This decomposition forms the basis of a cyclic update procedure, where each pair of $\{(\mathbf{d}_k, \mathbf{x}^k)\}_{k=1}^K$ is updated individually while all other pairs are held constant at their most recent values. Specifically, for the jth pair, the objective function in (8.1) can be expressed as the sum of a residual and a rank-one matrix, i.e.,

$$\min_{\mathbf{d}_j, \mathbf{x}^j} \|\mathbf{Y} - \mathbf{DX}\|_{\mathrm{F}}^2 = \min_{\mathbf{d}_j, \mathbf{x}^j} \|\mathbf{R}_j - \mathbf{d}_j \mathbf{x}^j\|_{\mathrm{F}}^2, \tag{8.3}$$

where the residual term

$$\mathbf{R}_j = \mathbf{Y} - \sum_{i \neq j} \mathbf{d}_i \mathbf{x}^i$$

does not depend on $(\mathbf{d}_j, \mathbf{x}^j)$. Because $\mathbf{d}_j \mathbf{x}^j$ has at most rank one, the minimization in (8.3) is precisely a low-rank approximation problem, which can be solved via the Singular Value Decomposition (SVD) [102].

Before computing the SVD of $\mathbf{R}_j$, we note that the support of $\mathbf{X}$ has already been determined using OMP. Resetting $\mathbf{x}^j$ via the SVD of $\mathbf{R}_j$ would destroy its sparse structure. To resolve this issue, instead of considering $\mathbf{R}_j$, we consider $\widetilde{\mathbf{R}}_j$, which is formed by retaining the columns of $\mathbf{R}_j$ that correspond to the non-zero entries in $\mathbf{x}^j$. We will see later that this restricted processing has a clear justification in the Bayesian context. Concretely, we have

$$\mathbf{d}_j \widetilde{\mathbf{x}}^j = \sigma_1 \mathbf{u}_1 \mathbf{v}^1,$$

where σ_1 is the largest singular value of $\widetilde{\mathbf{R}}_j$, $\mathbf{u}_1$ and $\mathbf{v}^1$ are its corresponding left and right singular vectors, and $\widetilde{\mathbf{x}}^j$ denotes the row vector $\mathbf{x}^j$ after imposing the known sparsity support. After this step, the values at the non-zero locations of

$\mathbf{x}^j$ are set equal to $\widetilde{\mathbf{x}}^j$. Notice that this restricted non-zero update does not have a mathematical justification and will reduce the quality of the SVD fitting. A justified way to alternate between atom and representation updates will be proposed in the coming sections.

The advantage of the K-SVD algorithm is its simplicity, as the update steps are greedy in nature. One major drawback, though, is that the uncertainty of the estimates of $\mathbf{D}$ and $\mathbf{X}$ is not taken into account in the estimation procedure. While not taking into account the uncertainty in the atoms of $\mathbf{D}$ may not be a problem due to the generally large number of columns in $\mathbf{X}$, each column of $\mathbf{X}$ normally has a reduced number of non-zero components and their inherent uncertainty should be accounted for. Furthermore, K-SVD requires to know the number of non-zero components in each column of $\mathbf{X}$, information that may not be available or may even be column dependent. In this Chapter we will show how these problems can be tackled in a principled manner using Bayesian modeling and inference.

8.2 Hierarchical Bayesian Model

8.2.1 Noise Modeling

The use of the sparsity inducing ℓ_1 norm in (8.2) requires an elaborate modeling, following our previous work in [103], we begin by modeling the observation process by using

$$\mathrm{p}(\mathbf{Y}|\mathbf{D},\mathbf{X},\beta) \propto \beta^{\frac{PQ}{2}} \exp\left\{-\frac{\beta}{2}\left\|\mathbf{Y}-\mathbf{D}\mathbf{X}\right\|_{\mathrm{F}}^2\right\}, \tag{8.4}$$

where β is the noise precision. We assume that

$$\mathrm{p}(\beta|a^\beta,b^\beta) = \Gamma(\beta|a^\beta,b^\beta) \propto \beta^{a^\beta-1}\exp(-b^\beta\beta),$$

with $a^\beta > 0$ and $b^\beta > 0$ being the shape and inverse scale parameters, respectively.

8.2.2 Modeling of D and X

Since we expect the columns of $\mathbf{D}$ to be normalized vectors we utilize the following prior on $\mathbf{D}$

$$\mathrm{p}(\mathbf{D}) = \prod_{k=1}^{K} \mathrm{p}(\mathbf{d}_k)$$

where

$$\mathrm{p}(\mathbf{d}_k) = \begin{cases} \text{const} & \text{if } \| \mathbf{d}_k \| = 1 \\ 0 & \text{elsewhere} \end{cases}$$

We now proceed to model the columns of $\mathbf{X}$. Although various general sparsity promoting priors could be considered here, see [104], we will only investigate the use of the Laplace prior on the components of the columns of $\mathbf{X}$ in this work. The non-conjugacy of the likelihood in (8.4) and Laplace prior distributions makes the use of this prior for the columns of $\mathbf{X}$ intractable. In our approach we address this issue by applying instead a three-tiered hierarchical prior on each column of $\mathbf{X}$, which has the same sparsifying effect as a Laplace prior while rendering the inference tractable.

For each column $\mathbf{x}_q, q = 1, \ldots, Q$ of $\mathbf{X}$, we utilize

$$\begin{aligned} \mathrm{p}(\mathbf{x}_q|\boldsymbol{\gamma}_q) &= \prod_{k=1}^{K} \mathcal{N}(x_{kq}|0, \gamma_{kq}) \\ &= \mathcal{N}(\mathbf{x}_q|\mathbf{0}_K, \boldsymbol{\Gamma}_q) \end{aligned}$$

where $\boldsymbol{\gamma}_q$ is a $K \times 1$ column vector with elements γ_{kq}, $k = 1, \cdots, K$, and $\boldsymbol{\Gamma}_q = \mathrm{diag}(\boldsymbol{\gamma}_q)$ along with the tiered hyperpriors

$$\mathrm{p}(\boldsymbol{\gamma}_q|\lambda_q) = \prod_{k=1}^{K} \Gamma(\gamma_{kq}|1, \lambda_q/2)$$

and

$$\mathrm{p}(\lambda_q|\nu_q) = \Gamma(\lambda_q|\nu_q/2, \nu_q/2),$$

where we assume a flat distribution on ν_q.

With marginalization, this hierarchical model yields a Laplace distribution of $\mathbf{x}_q$ conditioned on λ_q

$$\mathrm{p}\left(\mathbf{x}_q|\lambda_q\right) = \frac{\lambda_q^{K/2}}{2^K} \exp\left\{-\sqrt{\lambda_q}\|\mathbf{x}_q\|_1\right\} .$$

8.2.3 Complete System Modeling

Throughout this Chapter we will denote by

$$\boldsymbol{\Gamma} = [\boldsymbol{\gamma}_1, \cdots, \boldsymbol{\gamma}_Q] , \ \boldsymbol{\lambda} = [\lambda_1, \cdots, \lambda_Q] , \ \boldsymbol{\nu} = [\nu_1, \cdots, \nu_Q]$$

the hyperparameters associated with $\mathbf{X}$.

We also denote the entire set of unknowns as

$$\boldsymbol{\Theta} = \left\{\{\mathbf{d}_k\}_{k=1}^{K}, \{\mathbf{x}_q\}_{q=1}^{Q}, \boldsymbol{\Gamma}, \boldsymbol{\lambda}, \boldsymbol{\nu}, \beta\right\} .$$

Based on the above presented modeling, the complete system modeling is therefore given by the joint distribution

$$\mathrm{p}(\mathbf{Y}, \boldsymbol{\Theta}) = \mathrm{p}(\mathbf{Y}|\mathbf{D}, \mathbf{X}, \beta)\mathrm{p}(\beta)\mathrm{p}(\mathbf{D})\mathrm{p}(\mathbf{X}|\boldsymbol{\Gamma})\mathrm{p}(\boldsymbol{\Gamma}|\boldsymbol{\lambda})\mathrm{p}(\boldsymbol{\lambda}|\boldsymbol{\nu})\mathrm{p}(\boldsymbol{\nu}). \tag{8.5}$$

8.3 Inference

Our scheme for estimating $\mathbf{D}$ and $\mathbf{X}$ depends on our ability to estimate the posterior distribution $\mathrm{p}(\boldsymbol{\Theta}|\mathbf{Y})$. We do this using variational distribution approximation [105]. Specifically, with Mean-Field Factorization, the joint posterior distribution is approximated as

$$\mathrm{q}(\boldsymbol{\Theta}) = \mathrm{q}(\beta)\mathrm{q}(\boldsymbol{\Gamma})\,\mathrm{q}(\boldsymbol{\lambda})\,\mathrm{q}(\boldsymbol{\nu})\mathrm{q}(\mathbf{X})\prod_{k=1}^{K}\mathrm{q}(\mathbf{d}_k),$$

where in our case it is assumed that $\mathrm{q}(\boldsymbol{\Gamma})$, $\mathrm{q}(\boldsymbol{\lambda})$, and $\mathrm{q}(\boldsymbol{\nu})$ are degenerate distributions. We also assume that each $\mathrm{q}(\mathbf{d}_k), k = 1, \ldots, K$ is a degenerate distribution on a vector with $\| \mathbf{d}_k \|_2 = 1$.

For each $\boldsymbol{\theta}_i \in \boldsymbol{\Theta}$ where $\mathrm{q}(\boldsymbol{\theta}_i)$ is assumed to be degenerate, we can update its value by calculating

$$\hat{\boldsymbol{\theta}}_i = \arg\max_{\boldsymbol{\theta}_i} \ln \mathrm{q}(\boldsymbol{\theta}_i) = \arg\max_{\boldsymbol{\theta}_i} \langle \ln \mathrm{p}(\mathbf{Y}, \boldsymbol{\Theta}) \rangle_{\boldsymbol{\Theta}\setminus\boldsymbol{\theta}_i}, \tag{8.6}$$

where $\langle\cdot\rangle_{\boldsymbol{\Theta}\setminus\boldsymbol{\theta}_i}$ denotes the expectation taken with respect to all approximating factors $\mathrm{q}(\boldsymbol{\theta}_j)$, $j \neq i$.

For each $\boldsymbol{\theta}_i$ where $\mathrm{q}(\boldsymbol{\theta}_i)$ is assumed to be non-degenerate, we apply calculus of variations and obtain

$$\ln \mathrm{q}(\boldsymbol{\theta}_i) = \langle \ln\, \mathrm{p}(\mathbf{Y}, \boldsymbol{\Theta}) \rangle_{\boldsymbol{\Theta}\setminus\boldsymbol{\theta}_i} + C, \tag{8.7}$$

where C denotes a constant independent of the variable of current interest. For non-degenerate distributions $\mathrm{q}(\boldsymbol{\theta}_i)$, the updated value $\hat{\boldsymbol{\theta}}_i$ will denote its mean.

8.3.1 Estimation of X, Γ, λ, and ν

In order to find an approximate posterior distribution of $\mathbf{X}$, we apply (8.7) and obtain

$$\begin{aligned}
\ln \mathrm{q}(\mathbf{X}) &= \langle \ln \mathrm{p}(\mathbf{Y}|\mathbf{D}, \mathbf{X}, \beta) + \ln \mathrm{p}(\mathbf{X}|\boldsymbol{\Gamma}) \rangle_{\boldsymbol{\Theta}\setminus\mathbf{X}} + C \\
&= \sum_{q=1}^{Q} \left\langle -\frac{\beta}{2} \|\mathbf{y}_q - \mathbf{D}\mathbf{x}_q\|_2^2 - \frac{1}{2}\mathbf{x}_q^{\mathrm{T}}\boldsymbol{\Gamma}_q^{-1}\mathbf{x}_q \right\rangle_{\boldsymbol{\Theta}\setminus\mathbf{x}_q} + C \\
&= \sum_{q=1}^{Q} \left\{ -\frac{\hat{\beta}}{2} \left\|\mathbf{y}_q - \hat{\mathbf{D}}\mathbf{x}_q\right\|_2^2 - \frac{1}{2}\mathbf{x}_q^{\mathrm{T}}\hat{\boldsymbol{\Gamma}}_q^{-1}\mathbf{x}_q \right\} + C.
\end{aligned} \tag{8.8}$$

It is clear from (8.8) that the columns of $\mathbf{X}$ in the posterior distribution approximation are independent with

$$\ln \mathrm{q}(\mathbf{x}_q) = -\frac{\hat{\beta}}{2}\|\mathbf{y}_q - \hat{\mathbf{D}}\mathbf{x}_q\|_2^2 - \frac{1}{2}\mathbf{x}_q^{\mathrm{T}}\hat{\mathbf{\Gamma}}_q^{-1}\mathbf{x}_q + C\,.$$

It is straightforward to see that $\mathrm{q}(\mathbf{x}_q)$ is a Gaussian distribution

$$\mathrm{q}(\mathbf{x}_q) = \mathcal{N}(\mathbf{x}_q|\hat{\mathbf{x}}_q, \mathbf{\Sigma}_{\mathbf{x}_q})$$

with covariance matrix and mean vector defined respectively as

$$\mathbf{\Sigma}_{\mathbf{x}_q} = \left(\hat{\beta}\hat{\mathbf{D}}^{\mathrm{T}}\hat{\mathbf{D}} + \hat{\mathbf{\Gamma}}_q^{-1}\right)^{-1} \tag{8.9}$$

$$\hat{\mathbf{x}}_q = \hat{\beta}\mathbf{\Sigma}_{\mathbf{x}_q}\hat{\mathbf{D}}^{\mathrm{T}}\mathbf{y}_q. \tag{8.10}$$

Next, taking the appropriate expectation and finding a solution to (8.6) we can calculate the updates for the hyperparameters associated with $\mathbf{X}$. Specifically, the optimal $\hat{\gamma}_{kq}$ is given by

$$\hat{\gamma}_{kq} = -\frac{1}{2\hat{\lambda}_q} + \sqrt{\frac{1}{4\hat{\lambda}_q^2} + \frac{\hat{x}_{kq}^2 + \mathbf{\Sigma}_{\mathbf{x}_q}(k,k)}{\hat{\lambda}_q}};$$

the optimal $\hat{\lambda}_q$ is given by

$$\hat{\lambda}_q = \frac{\hat{\nu}_q + 2K - 2}{\hat{\nu}_q + \sum_{k=1}^{K}\hat{\gamma}_{kq}};$$

and finally, the optimal $\hat{\nu}_q$ is found via numerically maximizing

$$\frac{\nu_q}{2}\ln\frac{\nu_q}{2} - \ln\left(\Gamma\left(\frac{\nu_q}{2}\right)\right) + \frac{\nu_q}{2}\left(\ln\hat{\lambda}_q - \hat{\lambda}_q\right)\,.$$

8.3.2 Estimation of D

First notice that we assume that the columns of $\mathbf{D}$ are independent of each other in the posterior distribution approximation, i.e.,

$$\mathrm{q}(\mathbf{D}) = \prod_{k=1}^{K}\mathrm{q}(\mathbf{d}_k)\,,$$

with these distributions degenerate on a point in $\|\ \mathbf{d}_k\ \| = 1$.

Focusing on a single $\mathbf{d}_k$ and applying (8.6), we have

$$\hat{\mathbf{d}}_k = \arg\min_{\mathbf{d}_k}\left\langle\|\mathbf{Y} - \mathbf{D}\mathbf{X}\|_{\mathrm{F}}^2\right\rangle_{\mathbf{\Theta}\setminus\mathbf{d}_k}$$
$$s.t. \quad \|\ \mathbf{d}_k\ \| = 1.$$

We can write

$$\left\langle \|\mathbf{Y}-\mathbf{D}\mathbf{X}\|_{\mathrm{F}}^2 \right\rangle_{\boldsymbol{\Theta}\setminus\mathbf{d}_k} = \left\langle \left\|\mathbf{Y}-\mathbf{D}\hat{\mathbf{X}}\right\|_{\mathrm{F}}^2 \right\rangle_{\boldsymbol{\Theta}\setminus\mathbf{d}_k} + \left\langle \left\|\mathbf{D}\left(\hat{\mathbf{X}}-\mathbf{X}\right)\right\|_{\mathrm{F}}^2 \right\rangle_{\boldsymbol{\Theta}\setminus\mathbf{d}_k} \tag{8.11}$$

where we have

$$\begin{aligned}\left\langle \left\|\mathbf{Y}-\mathbf{D}\hat{\mathbf{X}}\right\|_{\mathrm{F}}^2 \right\rangle_{\boldsymbol{\Theta}\setminus\mathbf{d}_k} &= \left\| \mathbf{Y}-\sum_{i\neq k}\hat{\mathbf{d}}_i\hat{\mathbf{x}}^i-\mathbf{d}_k\hat{\mathbf{x}}^k \right\|_{\mathrm{F}}^2 \\ &= C+\|\hat{\mathbf{x}}^k\|_2^2\mathbf{d}_k^{\mathrm{T}}\mathbf{d}_k-2\mathbf{b}_k^{\mathrm{T}}\mathbf{d}_k \end{aligned} \tag{8.12}$$

with

$$\mathbf{b}_k=\left(\mathbf{Y}-\sum_{i\neq k}\hat{\mathbf{d}}_i\hat{\mathbf{x}}^i\right)\left(\hat{\mathbf{x}}^k\right)^{\mathrm{T}}.$$

Notice that (8.12) is the only term used in K-SVD to update the atoms of the dictionary.

The uncertainty of the estimate of $\mathbf{x}_q$ is incorporated in the estimation of $\mathbf{d}_k$ by the second term on the right hand side of (8.11) which we now calculate. It can be expressed as

$$\begin{aligned}\left\langle \left\|\mathbf{D}\left(\hat{\mathbf{X}}-\mathbf{X}\right)\right\|_{\mathrm{F}}^2 \right\rangle_{\boldsymbol{\Theta}\setminus\mathbf{d}_k} &= \left\langle \left\|\mathbf{d}_k(\hat{\mathbf{x}}^k-\mathbf{x}^k)\right\|_{\mathrm{F}}^2 \right\rangle_{\boldsymbol{\Theta}\setminus\mathbf{d}_k} \\ &+2\left\langle \mathrm{Tr}\left(\mathbf{d}_k(\hat{\mathbf{x}}^k-\mathbf{x}^k)\left(\sum_{i\neq k}\hat{\mathbf{d}}_i(\hat{\mathbf{x}}^i-\mathbf{x}^i)\right)^{\mathrm{T}}\right)\right\rangle_{\boldsymbol{\Theta}\setminus\mathbf{d}_k}+C. \end{aligned} \tag{8.13}$$

Now, the first term on the right hand side of (8.13) can be written as

$$\left\langle \|\mathbf{d}_k(\hat{\mathbf{x}}^k-\mathbf{x}^k)\| \right\rangle_{\boldsymbol{\Theta}\setminus\mathbf{d}_k}=c_k\mathbf{d}_k^{\mathrm{T}}\mathbf{d}_k, \tag{8.14}$$

where

$$c_k=\left\langle \|\hat{\mathbf{x}}^k-\mathbf{x}^k\|_2^2 \right\rangle_{\boldsymbol{\Theta}\setminus\mathbf{d}_k}=\sum_{q=1}^{Q}\boldsymbol{\Sigma}_{\mathbf{x}_q}(k,k),$$

and $\boldsymbol{\Sigma}_{\mathbf{x}_q}(k,k)$ denotes the (k,k)-th element of $\boldsymbol{\Sigma}_{\mathbf{x}_q}$ defined in (8.9).

Similarly, the second term on the right hand side of (8.13) can be written as

$$\left\langle \mathrm{Tr}\left(\mathbf{d}_k(\hat{\mathbf{x}}^k-\mathbf{x}^k)\left(\sum_{i\neq k}\mathbf{d}_i(\hat{\mathbf{x}}^i-\mathbf{x}^i)\right)^{\mathrm{T}}\right)\right\rangle_{\boldsymbol{\Theta}\setminus\mathbf{d}_k}=\mathbf{a}_k^{\mathrm{T}}\mathbf{d}_k, \tag{8.15}$$

where

$$\mathbf{a}_k = \sum_{q=1}^{Q} \sum_{i \neq k} \mathbf{\Sigma}_{\mathbf{x}_q}(i,k) \hat{\mathbf{d}}_i \, .$$

Substituting (8.14) and (8.15) into (8.13), we obtain

$$\left\langle \left\| \mathbf{D}\left(\hat{\mathbf{X}} - \mathbf{X}\right) \right\|_{\mathrm{F}}^2 \right\rangle_{\mathbf{\Theta} \setminus \mathbf{d}_k} = c_k \mathbf{d}_k^{\mathrm{T}} \mathbf{d}_k + 2\mathbf{a}_k^{\mathrm{T}} \mathbf{d}_k + C, \tag{8.16}$$

and substituting (8.12) and (8.16) into (8.11), we obtain

$$\begin{aligned} \left\langle \|\mathbf{Y} - \mathbf{D}\mathbf{X}\|_{\mathrm{F}}^2 \right\rangle_{\mathbf{\Theta} \setminus \mathbf{d}_k} &= e_k \mathbf{d}_k^{\mathrm{T}} \mathbf{d}_k - 2(\mathbf{b}_k - \mathbf{a}_k)^{\mathrm{T}} \mathbf{d}_k + C \\ &= \left\| \sqrt{e_k}\mathbf{d}_k - \frac{1}{\sqrt{e_k}}(\mathbf{b}_k - \mathbf{a}_k) \right\|^2 + C, \end{aligned}$$

where

$$e_k = \|\hat{\mathbf{x}}^k\|^2 + c_k \, .$$

Defining

$$\mathbf{t}_k = \frac{1}{\sqrt{e_k}}(\mathbf{b}_k - \mathbf{a}_k)$$

we obtain

$$\left\langle \|\mathbf{Y} - \mathbf{D}\mathbf{X}\|_{\mathrm{F}}^2 \right\rangle_{\mathbf{\Theta} \setminus \mathbf{d}_k} = \|\mathbf{t}_k - \sqrt{e_k}\mathbf{d}_k\|^2 + C \, .$$

We can therefore finally write

$$\begin{aligned} \hat{\mathbf{d}}_k = & \arg\min \|\mathbf{t}_k - \sqrt{e_k}\mathbf{d}_k\|^2 \\ & s.t. \parallel \mathbf{d}_k \parallel = 1, \end{aligned}$$

which produces

$$\hat{\mathbf{d}}_k = \frac{1}{\parallel \mathbf{t}_k \parallel} \mathbf{t}_k = \frac{\mathbf{b}_k - \mathbf{a}_k}{||\mathbf{b}_k - \mathbf{a}_k||}. \tag{8.17}$$

8.3.3 Estimation of Noise Precision β

Keeping the terms dependent on β in (8.5) and applying (8.7), we obtain

$$\begin{aligned} \ln \mathrm{q}(\beta) = & \frac{PQ}{2} \ln \beta - \frac{\beta}{2} \left\langle \|\mathbf{Y} - \mathbf{D}\mathbf{X}\|_{\mathrm{F}}^2 \right\rangle_{\mathbf{\Theta} \setminus \beta} \\ & + (a^\beta - 1) \ln \beta - b^\beta \beta + C, \end{aligned}$$

from which we see that $\mathrm{q}(\beta)$ is a Gamma distribution with mean

$$\hat{\beta} = \frac{PQ + 2a^\beta}{\sum_{q=1}^{Q} \langle \|\mathbf{y}_q - \hat{\mathbf{D}}\mathbf{x}_q\|^2 \rangle_{\mathbf{x}_q} + 2b^\beta} \, . \tag{8.18}$$

8.4 Fast Inference Procedure Based on Empirical Bayes

The inference procedure introduced in the previous section is mathematically sound but has a practical limitation: computing $\mathbf{\Sigma}_{\mathbf{x}_q}$ in (8.9) for each q requires the inversion of a $K \times K$ matrix. This matrix inversion at each iteration step can be computationally costly and memory intense.

In order to reduce the computational complexity and alleviate memory requirement, we propose a fast inference procedure based on the use of Empirical Bayes [103, 106, 107]. The principle of this approach is first presented in [106] in the context of Sparse Bayesian Learning (SBL) and later adapted in [103] and [107] for recovery of sparse signals. Here we adapt it for the sparse dictionary learning problem.

Specifically, for each $\mathbf{x}_q$, we adopt a constructive approach for identifying its support, i.e., the locations where it assumes non-zero values. The values of the hyperparameters at these non-zero locations are obtained via Maximum *A Posteriori* (MAP) estimation. With such support identification and hyperparameter estimation, the effective problem dimensions are drastically reduced due to sparsity. Finally, the estimated values of $\mathbf{x}_q$ in its support are obtained via (8.10).

8.4.1 Fast Bayesian Inference for $\mathbf{\Gamma}$ and $\mathbf{X}$

We will derive in this section a fast inference approach for $\mathbf{\Gamma}$ and $\mathbf{X}$. We have

$$\mathrm{p}(\mathbf{y}_q|\hat{\beta}, \hat{\mathbf{D}}, \boldsymbol{\gamma}_q) = \mathcal{N}(\mathbf{y}_q|\mathbf{0}_P, \mathbf{C}_q)$$

with covariance matrix

$$\mathbf{C}_q = \hat{\beta}^{-1}\mathbf{I}_P + \hat{\mathbf{D}}\mathbf{\Gamma}_q\hat{\mathbf{D}}^{\mathrm{T}}.$$

Separating the contribution of a single γ_{kq} from $\mathbf{C}_q$, we have

$$\mathbf{C}_q = {}^{-k}\mathbf{C}_q + \gamma_{kq}\hat{\mathbf{d}}_k\hat{\mathbf{d}}_k^{\mathrm{T}},$$

Using the matrix inversion lemma we obtain

$$\mathbf{C}_q^{-1} = {}^{-k}\mathbf{C}_q^{-1} - \frac{{}^{-k}\mathbf{C}_q^{-1}\hat{\mathbf{d}}_k\hat{\mathbf{d}}_k^{\mathrm{T}}\ {}^{-k}\mathbf{C}_q{}^{-1}}{\gamma_{kq}^{-1} + \hat{\mathbf{d}}_k^{\mathrm{T}}{}^{-k}\mathbf{C}_q^{-1}\hat{\mathbf{d}}_k},$$

and using the determinant identity we obtain

$$|\mathbf{C}_q| = |{}^{-k}\mathbf{C}_q||1 + \gamma_{kq}\hat{\mathbf{d}}_k^{\mathrm{T}}\ {}^{-k}\mathbf{C}_q{}^{-1}\hat{\mathbf{d}}_k|\,.$$

Utilizing $\mathrm{p}(\mathbf{y}_j|\hat{\beta}, \hat{\mathbf{D}}, \boldsymbol{\gamma}_q)$ and $\mathrm{p}(\boldsymbol{\gamma}_q|\hat{\lambda}_q)$ our goal is to maximize the following function of $\boldsymbol{\gamma}_q$

$$\begin{aligned}\mathcal{L}(\boldsymbol{\gamma}_q) &= -\frac{1}{2}\left[\log|{}^{-k}\mathbf{C}_q| + \mathbf{y}_q^{\mathrm{T}-k}\mathbf{C}_q^{-1}\mathbf{y}_q + \lambda_q\sum_{n\neq k}\gamma_{nq}\right] \\ &+ \frac{1}{2}\left[\log\frac{1}{1+\gamma_{kq}s_{iq}} + \frac{h_{kq}^2\gamma_{kq}}{1+\gamma_{kq}s_{kq}} - \lambda_q\gamma_{iq}\right] \\ &= \mathcal{L}({}^{-k}\boldsymbol{\gamma}_q) + l(\gamma_{kq}),\end{aligned} \tag{8.19}$$

where

$$l(\gamma_{kq}) = \frac{1}{2}\left[\log\frac{1}{1+\gamma_{kq}s_{kq}} + \frac{h_{kq}^2\gamma_{kq}}{1+\gamma_{kq}s_{kq}} - \lambda_q\gamma_{kq}\right]$$

and s_{kq} and h_{kq} are defined as

$$\begin{aligned}s_{kq} &= \hat{\mathbf{d}}_k^{\mathrm{T}}\left({}^{-k}\mathbf{C}_q\right)^{-1}\hat{\mathbf{d}}_k \\ h_{kq} &= \hat{\mathbf{d}}_k^{\mathrm{T}}\left({}^{-k}\mathbf{C}_q\right)^{-1}\mathbf{y}_q\end{aligned}.$$

The quantities s_{kq} and h_{kq} do not depend on γ_{kq}. Therefore, the terms related to a single hyperparameter γ_{kq} are now separated from others. A closed form solution of the maximization of $\mathcal{L}(\boldsymbol{\gamma}_q)$, when only its kth component is changed, can be found by holding the other hyperparamters fixed, taking its derivative with respect to γ_{kq} and setting it equal to zero.

The optimal $\hat{\gamma}_{kq}$ can be obtained as follows

$$\hat{\gamma}_{kq} = m_{kq}\mathbf{1}_{[h_{kq}^2 - s_{kq} \geq \lambda_q]} \tag{8.20}$$

where

$$m_{kq} = \frac{-s_{kq}(s_{kq}+2\hat{\lambda}_q) + s_{kq}\sqrt{(s_{kq}+2\hat{\lambda}_q)^2 - 4\hat{\lambda}_q(s_{kq}-h_{kq}^2+\hat{\lambda}_q)}}{2\hat{\lambda}_q s_{kq}^2}$$

It is crucial for computational efficiency that once a hyperparameter γ_{kq} is updated using (8.20), the quantities s_{kq}, h_{kq}, $\hat{\mathbf{x}}_q$, and $\Sigma_{\mathbf{x}_q}$ are efficiently updated. Similarly to [103], for a given q the parameters s_{kq} and h_{kq} can be calculated for all basis atoms $\hat{\mathbf{d}}_k$ efficiently using the following identities

$$\begin{aligned}S_{kq} &= \hat{\beta}\hat{\mathbf{d}}_k^t\hat{\mathbf{d}}_k - \hat{\beta}^2\hat{\mathbf{d}}_k^t\hat{\mathbf{D}}\Sigma_{\mathbf{x}_q}\hat{\mathbf{D}}^t\hat{\mathbf{d}}_k \\ Q_{kq} &= \hat{\beta}\hat{\mathbf{d}}_k^t\mathbf{y}_q - \hat{\beta}^2\hat{\mathbf{d}}_k^t\hat{\mathbf{D}}\Sigma_{\mathbf{x}_q}\hat{\mathbf{D}}^t\mathbf{y}_q \\ s_{kq} &= \frac{S_{kq}}{1-\gamma_{kq}S_{kq}} \\ q_{kq} &= \frac{Q_{kq}}{1-\gamma_{kq}S_{kq}},\end{aligned}$$

where $\mathbf{\Sigma}_{\mathbf{x}_q}$ and $\hat{\mathbf{D}}$ include only the columns k that are included in the model ($\gamma_{kq} \neq 0$). Moreover, $\mathbf{\Sigma}_{kq}$ and $\hat{\mathbf{x}}_q$ can be updated very efficiently when only a single coefficient γ_{kq} is considered, as in [103]. With the fast updates for $\{\mathbf{x}_q\}$ in hand we can formally state the full **Algorithm 4** which we call the Bayesian K-SVD (BKSVD) method.

At step 9 of the algorithm, a candidate γ_{kq} must be selected for updating. This can be done by randomly choosing a basis vector $\hat{\mathbf{d}}_k$, or by calculating each γ_{kq} and choosing the one that results in the greatest increase in $\mathcal{L}(\boldsymbol{\gamma})$ in (8.19), which results in a faster convergence. The latter is the method implemented in this work.

An important contribution of the algorithm is the estimation of the noise precision β, which is derived in the previous section using (8.18).

In the approach presented in [108], the estimation of the noise precision is unreliable at early iterations and hence it is necessary to set it to a fixed value at the beginning of the algorithm. Unreliable estimates can indeed significantly affect the performances of the technique. However, in the proposed method β is estimated using a set of signals which are assumed to share the same noise variance, thus leading to reliable estimates even at early BKSVD iterations.

Let us complete this section by analyzing the proposed approach and comparing it to K-SVD. First we examine the variance estimates provided by the Relevance Vector Machine and the ones provided by the proposed method in terms of sparsity. The RVM model corresponds to the particular case of $\lambda = 0$ in our model. For the RVM we have

$$\gamma_{kq}^{\mathrm{RVM}} = \begin{cases} \frac{h_{kq}^2 - s_{kq}}{s_{kq}^2} & \text{if } h_{kq}^2 - s_{kq} > 0 \\ 0 & \text{otherwise.} \end{cases}$$

Let us now examine the difference $\gamma_i^{\mathrm{RVM}} - \gamma_i^{\mathrm{L}}$, where γ_i^{L} denotes the estimate using the Laplace prior as presented in this work. When $h_{kq}^2 - s_{kq} < \lambda$ we have

$$\gamma_{kq}^{\mathrm{RVM}} - \gamma_{kq}^{\mathrm{L}} = \begin{cases} 0 & \text{if } h_{kq}^2 - s_{kq} < 0 \\ \gamma_{kq}^{\mathrm{RVM}} & \text{if } 0 \leq h_{kq}^2 - s_{kq} < \lambda. \end{cases}$$

When $h_{kq}^2 - s_{kq} \geq \lambda$, the derivative of the function $l(\gamma_{kq})$ at $\gamma_{kq} = \gamma_{kq}^{\mathrm{RVM}}$ is $-\lambda < 0$. Since $\frac{dl(\gamma_{kq})}{d\gamma_{kq}}|_{\gamma_{kq}=0} > 0$, the maximum of $l(\gamma_{kq})$ occurs at a smaller value γ_{kq}^{L} than γ_i^{RVM}. Consequently we always have

$$\gamma_{kq}^{\mathrm{RVM}} \geq \gamma_{kq}^{\mathrm{L}},$$

That is, the estimates γ_{kq}^{L} using the Laplace prior are always not greater than the estimates $\gamma_{kq}^{\mathrm{RVM}}$ of the relevance vector machine. Note also that compared to RVM more components will possibly be pruned out from the model when $\lambda > 0$,

since the cardinality of the set $\mathbf{x}_q(k)$ for which $h_{kq}^2 - s_{kq} > \lambda$ is smaller than that of the set $\mathbf{x}_q(k)$ for which $h_{kq}^2 - s_{kq} > 0$. These observations imply that the solution obtained by the proposed method is at least as sparse as the one provided by the RVM. This will also be shown empirically in the experimental section.

We now relate the proposed BKSVD model to K-SVD. In K-SVD the number of non-zero components, S, in $\mathbf{x}_q$ is fixed in advance. In BKSVD we can update $\boldsymbol{\gamma}_q$ until convergence and then keep only its S largest values. We can also run BKSVD in a greedy fashion until S non-zero components are incorporated.

Finally, let us compare the iterative procedures for BKSVD and K-SVD. In K-SVD to update the kth atom we select the non-zero components in $\hat{\mathbf{x}}^k$. If the qth component is selected, this means that γ_{kq} is non-zero in our fast formulation. Notice that the components selected by BKSVD ($\gamma_{kq} \neq 0$) and the ones selected by K-SVD ($\hat{x}_{kq} \neq 0$) coincide almost surely. K-SVD then proceeds to find the rank one SVD decomposition of the residual term

$$\mathbf{R}_k = \mathbf{Y} - \sum_{i \neq k} \hat{\mathbf{d}}_i \hat{\mathbf{x}}^i,$$

where only the columns $\mathbf{y}_q$ with non-zero γ_{kq} are considered. This produces an update of $\mathbf{d}_k$ and the non-zero components of $\hat{\mathbf{x}}^k$. On the other hand, BKSVD not only takes into account the residual $\mathbf{R}_k$ in $\|\mathbf{Y} - \sum_{i \neq k} \hat{\mathbf{d}}_i \hat{\mathbf{x}}^i - \mathbf{d}_k \hat{\mathbf{x}}^k\|_{\mathrm{F}}^2$, see (8.12), but also makes the uncertainty of the estimation of $\mathbf{X}$ reponsible for some of the variation of the model, see $\langle \|\mathbf{D}\left(\hat{\mathbf{X}} - \mathbf{X}\right)\|_{\mathrm{F}}^2 \rangle_{\boldsymbol{\Theta} \setminus \mathbf{d}_k}$ in (8.13).

To update the kth atom BKSVD utilizes (8.17), while K-SVD utilizes the rank one decomposition of $\mathbf{R}_k$ to update $\hat{\mathbf{d}}_k$ and the non-zero elements in $\mathbf{x}^k$. For BKSVD, once the kth atom has been updated we can also update the non-zero components in $\mathbf{x}^k$. Both strategies will be compared in the experimental section.

8.4.2 Suboptimal greedy version

Since the Bayesian K-SVD algorithm we introduced in the previous sections takes into account the uncertainties of the coefficients to improve the estimation, it is computationally expensive. As an example, using non-optimized code on a server equipped with Intel Xeon® CPU E5-4640 @ 2.40 GHz processor, the learning and reconstruction phases for a 64×255025 $\mathbf{Y}$ matrix using $Q = 256$ atoms require 3 hours and 30 minutes, respectively. During training and reconstruction, the bottleneck is in the computation of the sparse representation in which atoms are added, deleted or reestimated.

To improve the overall speed of the Bayesian K-SVD algorithm, and in particular, that of the sparse representation computation, we introduce a faster version. Inspired by the approach of greedy algorithms like OMP, we propose to compute

Algorithm 4 Pseudocode for BKSVD algorithm

Input: $\mathbf{Y}$, initial normalized $\mathbf{D}$
Output: $\hat{\mathbf{D}}$, $\hat{\mathbf{\Gamma}}$, the posterior approximations q($\mathbf{x}_q$),
$q = 1, \ldots, Q$
initialize $\mathbf{\Gamma}$ and $\boldsymbol{\lambda}$ to zero
while not converged **do**
    for q in $1, \ldots, Q$ **do**
        while not converged **do**
            Choose a $k \in \{1, \ldots, K\}$
    (or equivalently choose a γ_{kj})
            Find optimal $\hat{\gamma}_{kq}$ using (8.20)
            Update $\mathbf{\Sigma}_{\mathbf{x}_q}$ and $\hat{\mathbf{x}}_q$ based on $\hat{\gamma}_{kq}$
            Update s_{kq} and h_{kq}
            Update $\hat{\lambda}_j$ and $\hat{\nu}_j$
        end while
    end for
    for k in $1, \ldots, K$ **do**
        Update $\mathbf{d}_k$ using (8.17)
    end for
    Update $\hat{\beta}$
end while

the sparse representation in an additive suboptimal fashion. This faster version only adds atoms instead of reestimating or deleting elements in the support of the sparse signals. That is, when the likelihood is maximized only by removing or reestimating a new atom, the sparse representation calculation stops. This approach allows for the whole BKSVD algorithm to perform fewer operations and hence leads to faster iterations.

To validate the proposed approach, we ran the following synthetic experiment. We generated a $\mathbf{D}_{64\times150}$ dictionary and a sparse matrix $\mathbf{X}_{150\times1500}$ with different numbers of non-zero components per column, see Table 8.1, and calculated $\mathbf{Y} = \mathbf{DX}$ with no noise.

We compared the BKSVD algorithm and its faster version by examining the percentage of columns in $\mathbf{X}$ for which each method correctly selects at least 80% of the atoms, as shown in Table 8.1. As can be seen in it, the performance of the two algoritms is comparable for smaller values of s.

We show next in Figure 8.1 the required computation times using different dictionary sizes K with $K = iP$, $i = 2,3,4$, $P = 64$ and the values of Q corresponding to the total number of overlapping patches in 128×128,256×256 and 512×512 images assuming full overlap. As can be seen from it, the computational savings are significant. All the experiments we present in Sec. 8.5 are performed using this faster and greedy method.

Table 8.1: Performance of the BKSVD algorithm and its faster greedy version for different percentages of non-zero components s

s (%)	Optimal sparse coding	Greedy sparse coding
3	100.00	100.00
6	100.00	99.87
10	99.67	98.50
13	92.00	70.00

8.5 Experimental results

We carried out experiments on denoising and inpainting to demonstrate the performance of the proposed BKSVD algorithm on real data. Experiments were performed on four typical grayscale images, namely *Barbara*, *Boat*, *Lena* and *Peppers*.

For both denoising and inpainting a dictionary of 256 atoms was learned. The dictionary was initialized with an overcomplete DCT dictionary. To have an unbiased dictionary, the mean is removed from each patch before running the BKSVD algorithm and then added back to the processed patches. Images are divided into

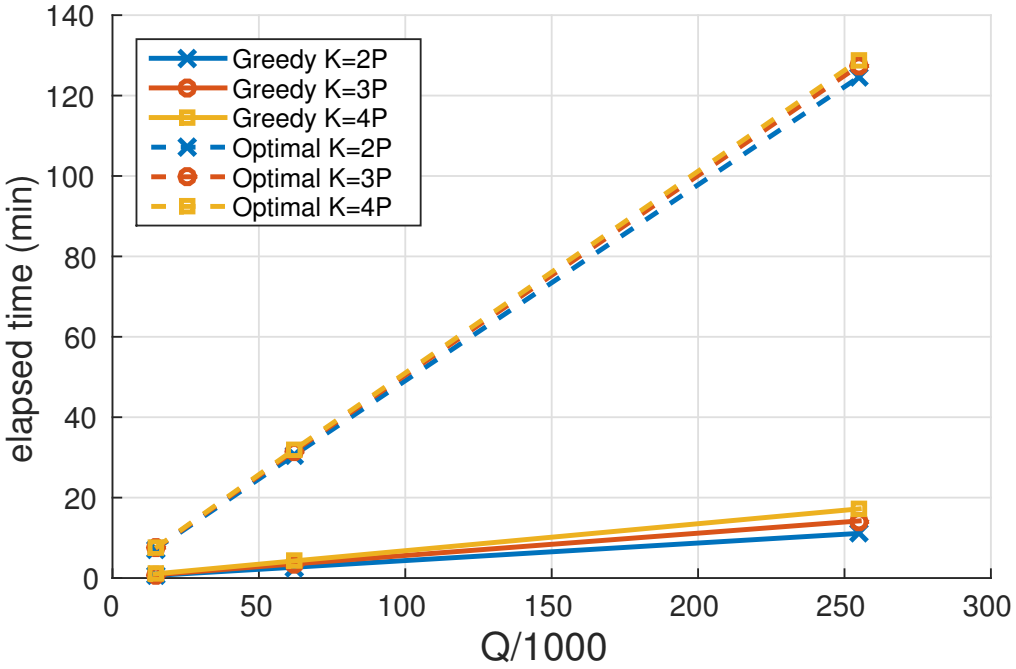


Figure 8.1: Time required to compute the sparse representation of synthetic data (as in the experiment in Table 8.1) for the proposed methods.

8×8 overlapping patches vectorized in columns and stacked into a matrix. We use maximum overlap for better performance, although it slows down the representation task. Recovered overlapping patches are then averaged according to their pixel contribution to the image. We used $a = b = 1$ for the Gamma hyperprior distribution.

(a) Noisy image ($\sigma = 20$) (b) BKSVD Denoising (c) K-SVD Denoising

Figure 8.2: Comparison of the denoising performances of BKSVD and K-SVD algorithms.

(a) $r = 75\%$ missing pixels (b) BKSVD inpainting (c) K-SVD inpainting

Figure 8.3: Example of inpainting results.

8.5.1 Denoising

To assess the performance of the proposed fast BKSVD algorithm, we compare it with K-SVD . Differently from K-SVD, which requires knowledge of the exact noise variance, information rarely available in real problems, the proposed Bayesian approach automatically estimates its value. To perform a fair comparison, K-SVD is run with both the noise variance estimated by our method and the true added noise.

We learned the dictionaries for both K-SVD and BKSVD using the noisy patches of the image itself (of size 256×256 pixels). We corrupted the images with additive white Gaussian noise (AWGN) with standard deviation $\sqrt{1/\beta} \in \{5{,}10{,}15{,}20{,}25{,}50\}$.

We show in Table 8.3 a comparison of the two techniques. As we have already mentioned, we compared the proposed BKSVD algorithm with both the K-SVD aware of the true noise standard deviation and using the one estimated by our algorithm. Notice that, since our noise estimate is very close to the true one, these two experiments we performed using K-SVD present similar results.

Table 8.2: Estimated σ using the "Lena" image.

σ	5	10	15	20	25	50
$\hat{\sigma}$	5.54	10.39	15.01	19.55	24.46	46.80

The proposed method performs equally or better than K-SVD in 20 out of 24 experiments and also is capable of estimating the noise variance. Notice also that unlike our method, K-SVD is very sensitive to noise variance mismatch. This mismatch can decrease its PSNR performance by a few dBs [89]. On the other hand, our technique performs a completely automatic noise variance estimation and is more robust to high noise levels because it takes into account the uncertainty of the estimates.

We show in Table 8.3 the average percentage of non-zero components in the estimated $\mathbf{X}$. As can be seen, while PSNR and SSIM values are similar for both techniques, BKSVD always obtains sparser solutions which indicates that the learned dictionary with our method contains atoms which can better represent the signal.

An example of denoising is shown in Figure 8.2 where we can see the denoised "Lena" provided by both methods. As shown in this figure, the proposed technique preserves edges and high spatial frequencies better than K-SVD, which leads to a flatter and more blurry image. Moreover, as we already pointed out previously, our method does not require any prior information on the noise corrupting the images since its variance is automatically estimated.

Table 8.2 shows both the true synthetic noise standard deviation and the corresponding estimation by our method for the denoising experiment using the "Lena" image.

8.5.2 Inpainting

Sparse coding is also capable of dealing with missing information. The problem stated in (8.1) needs to be adapted to handle this lack of information at the reconstruction phase, that is, after the dictionary has been learned.

For this experiment, a dictionary of 256 atoms was learned from a database of 23 images. From every image, we selected the 4096 patches with the highest variances. Following the approach in [2], these images did not contain missing values. During testing, and for images not in the training set, 25%, 50% and 75% of the pixels in those images were removed (set to zero) from every non-overlapping patch of each 512×512 test image. No noise was added. Regarding the K-SVD parameters, a very small σ was used since the image has no noise.

During testing, the process was adapted to deal with the missing information. Let n_q denote the position of the pixels in a patch q where the information is available. We create the set of truncated vectors $\tilde{\mathbf{y}}_q = \mathbf{y}_q(n_q)$ which contain the entries of $\mathbf{y}_q$ restricted to the indices in n_q, and consider the set of truncated dictionaries for these signals $\tilde{\mathbf{D}}^{(q)} = [\mathbf{d}_1(n_q), \cdots, \mathbf{d}_K(n_q)]$. We then estimate $\mathbf{x}_q$ from the observation model

$$\tilde{\mathbf{y}}_q = \tilde{\mathbf{D}}^{(q)}\mathbf{x}_q, \quad q = 1, \ldots, Q.$$

Finally, the image is recovered from the estimated representations $\hat{x}_q$ and the full dictionary, $\hat{\mathbf{Y}} = \mathbf{D}\hat{\mathbf{X}}$. This process is depicted in Figure 8.4.

As we can see in table 8.4, the results obtained by the proposed method outperform those obtained by K-SVD, suggesting an improved capability of representation by the learned BKSVD dictionary. Notice that for high percentages r of missing pixels and due to the scarcity of data both methods perform similarly, although the proposed on still performs slightly better. We show a graphical example in figure 8.3

for the highest ratio of missing pixels ($r = 75\%$). There is a noticeable improvement in the visual quality of the image recovered by our method in contrast to the too smooth K-SVD reconstruction.

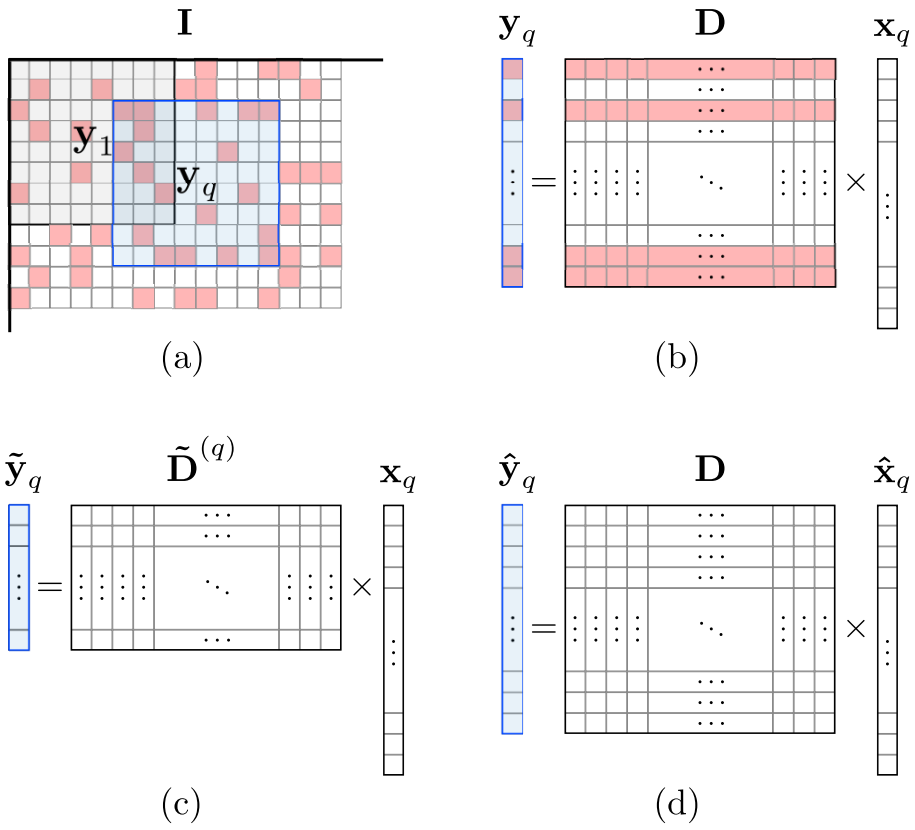


Figure 8.4: Inpainting process: (a) two patches from image $\mathbf{I}$, $\mathbf{y}_1$ and $\mathbf{y}_q$ (missing pixels in red), (b) vectorization of patch $\mathbf{y}_q$; rows from $\mathbf{D}$ corresponding to the missing pixels in $\mathbf{y}_q$ are also highlighted in red, (c) highlighted entries are discarded from the problem formulation, (d) recovery using the full dictionary $\mathbf{D}$.

Table 8.3: Comparison of the proposed BKSVD algorithm with K-SVD [2]. K-SVD was tested with estimated and true σ. Top to bottom, we show PSNR (dB), SSIM and average sparsity (non-zero coefficients over the number of coefficients).

σ	Barbara			Boat			Lena			Peppers		
	BKSVD	K-SVD $\hat{\sigma}$	K-SVD σ	BKSVD	K-SVD $\hat{\sigma}$	K-SVD σ	BKSVD	K-SVD $\hat{\sigma}$	K-SVD σ	BKSVD	K-SVD $\hat{\sigma}$	K-SVD σ
5	38.02	38.00	**38.10**	**36.71**	36.70	36.40	38.90	38.92	**38.93**	**38.98**	38.91	38.89
	0.97	**0.97**	**0.97**	**0.96**	**0.96**	**0.96**	**0.97**	**0.97**	**0.97**	**0.97**	**0.97**	**0.97**
	0.05	0.08	0.10	**0.03**	0.08	0.12	**0.04**	0.08	0.08	**0.04**	0.07	0.08
10	**33.97**	33.95	**33.97**	32.8	32.7	**33.0**	**34.96**	34.94	34.95	**35.27**	35.08	35.09
	0.93	**0.93**	**0.93**	0.91	0.90	**0.92**	**0.94**	**0.94**	**0.94**	**0.94**	**0.94**	**0.94**
	0.02	0.04	0.06	**0.02**	0.04	0.04	**0.02**	0.04	0.04	**0.01**	0.03	0.02
15	**31.85**	31.82	31.84	**31.01**	30.90	31.00	**32.78**	32.73	32.70	**33.12**	33.00	33.04
	0.90	**0.90**	**0.90**	**0.87**	**0.87**	**0.87**	**0.91**	**0.91**	**0.91**	**0.92**	**0.92**	**0.92**
	0.01	0.03	0.03	**0.01**	0.03	0.02	**0.01**	0.03	0.03	**0.01**	0.03	0.02
20	**30.30**	30.24	30.28	**29.44**	29.42	**29.44**	**31.32**	31.27	31.30	**31.36**	31.35	**31.36**
	0.87	**0.87**	**0.87**	**0.83**	**0.83**	**0.83**	**0.88**	**0.88**	**0.88**	**0.89**	**0.89**	**0.89**
	0.01	0.02	0.02	**0.01**	0.02	0.02	**0.01**	0.02	0.02	**0.01**	0.02	0.02
25	**29.10**	**29.10**	29.05	**28.38**	**28.38**	28.36	29.99	29.87	**30.00**	**30.20**	30.12	30.15
	0.84	0.83	0.83	**0.80**	**0.80**	**0.80**	**0.86**	**0.86**	**0.86**	**0.88**	0.87	0.87
	0.01	0.02	**0.01**	**0.01**	0.02	**0.01**	**0.01**	0.02	**0.01**	**0.01**	0.02	**0.01**
50	**25.60**	25.51	25.43	**25.10**	25.01	24.92	**26.31**	26.16	26.23	**26.41**	26.19	26.32
	0.72	0.71	0.71	**0.67**	0.66	**0.67**	**0.73**	0.72	0.72	**0.77**	0.76	**0.77**
	0.01	**0.01**	**0.01**	**0.01**	**0.01**	**0.01**	**0.01**	**0.01**	**0.01**	**0.01**	**0.01**	**0.01**

Table 8.4: Inpainting results. Comparison of the proposed BKSVD algorithm with K-SVD for different ratios (r) of missing pixels. PSNR and SSIM values are given.

r (%)	Barbara		Boat		Lena		Peppers	
	BKSVD	K-SVD	BKSVD	K-SVD	BKSVD	K-SVD	BKSVD	K-SVD
25	**39.1773**	30.9816	**38.5391**	34.3484	**41.5410**	36.9890	**39.4880**	35.1438
	0.9869	0.9524	**0.9743**	0.9455	**0.9811**	0.9700	**0.9676**	0.9389
50	**33.1485**	26.4042	**32.6518**	29.7701	**36.1429**	32.3366	**34.7303**	31.6623
	0.9568	0.8600	**0.9239**	0.8763	**0.9512**	0.9257	**0.9261**	0.8951
75	**27.3552**	23.8155	**27.3947**	26.1601	**30.3712**	28.7631	**29.2929**	28.3120
	0.8605	0.7213	**0.8021**	0.7568	**0.8800**	0.8477	**0.8542**	0.8318

Chapter 9

Compressive BKSVD

Heretofore we described a novel dictionary learning technique which, exploiting the Bayesian inference, enables a fully automatic estimation of the parameters of the model and improves the performance over the well-known KSVD. The natural evolution of such technique is the extension in the compressed domain. In fact, CS heavily depends on the sparsity of the original signal and in turn on the quality of the sparsifying basis. Being able to estimate a good sparsifying basis given a set of random projections would be extremely useful in many applications in which the basis is not known a-priori or when standard bases are not able to correctly sparsify a given signal. In this chapter we introduce the compressive BKSVD by leveraging the BKSVD model we described in Chapter 8.

9.1 Modeling

The modeling we consider for the C-BKSVD is the same as the one we introduced for the BSKVD. In fact, the hierarchical structure we employed on $\mathbf{X}$ to enforce the sparsity as well as the modeling of the columns of the dictionary $\mathbf{D}$ do not require any modification. The transformation induced by CS is a linear transformation which does not involve any stochastic quantities. However, we need to take into account that projecting all the signals in $\mathbf{X}$ onto the same subspace spanned by a single sensing matrix $\mathbf{\Phi}$ may lead to errors. As pointed out in [8], projecting all the training signals onto the same low-dimensional subspace may lead to the loss of the original signal space and the ability to correctly recover the signals. It is worth noting that this approach does not require extra memory or transmission costs because there is no need to store or transmit all the sensing matrices since a seed used to generate the random matrix is enough to build the sensing matrices on the fly.

This said, the model we take into account is given by

$$\mathbf{y}_q = \mathbf{\Phi}^{(q)}\mathbf{D}\mathbf{x}_q \; \forall q \in 1 \dots Q$$

where each training signal is projected by means of a different sensing matrix.

9.2 Inference

The C-BKSVD inference follows the same structure as the one for the BKSVD algorithm, except for few differences we need to take into account which are due to presence of different sensing matrices. For what concerns the inference on the sparse representation $\mathbf{X}$, since it is performed independently for each column $\mathbf{x}^q$, we can directly refer to the method in Section 8.3.1 with the only difference that each signal is projected by a different $\mathbf{\Phi}^{(q)}$. The main differences rely in the dictionary update step in which the uncertainties coming from the sparse representation are joined to improve the dictionary inference. More in detail, focusing on a single $\mathbf{d}_k$, applying (8.6) we have

$$\begin{aligned} \hat{\mathbf{d}}_k &= \arg\min_{\mathbf{d}_k} \sum_q \left\langle \|\mathbf{y}_q - \mathbf{\Phi}^{(q)}\mathbf{D}\mathbf{x}_q\|_2^2 \right\rangle_{\mathbf{\Theta}\setminus\mathbf{d}_k} \\ s.t. \quad &\| \mathbf{d}_k \| = 1, \end{aligned}$$

then, the argument of the minimization problem can be written as

$$\sum_q \left\langle \|\mathbf{y}_q - \mathbf{\Phi}^{(q)}\mathbf{D}\hat{\mathbf{x}}_q\|_2^2 \right\rangle_{\mathbf{\Theta}\setminus\mathbf{d}_k} + \sum_q \left\langle \|\mathbf{\Phi}^{(q)}\mathbf{D}\left(\hat{\mathbf{x}}_q - \mathbf{x}_q\right)\|_2^2 \right\rangle_{\mathbf{\Theta}\setminus\mathbf{d}_k} \tag{9.1}$$

If we focus on the first term we have

$$\begin{aligned} \sum_q \left\langle \|\mathbf{y}_q - \mathbf{\Phi}^{(q)}\mathbf{D}\hat{\mathbf{x}}_q\|_2^2 \right\rangle_{\mathbf{\Theta}\setminus\mathbf{d}_k} &= \sum_q \left\langle \|\mathbf{y}_q - \mathbf{\Phi}^{(q)} \sum_{i\neq k} \hat{\mathbf{d}}_i \hat{\mathbf{x}}_q^i - \mathbf{\Phi}^{(q)}\mathbf{d}_k\mathbf{x}_q^k\|_2^2 \right\rangle_{\mathbf{\Theta}\setminus\mathbf{d}_k} \\ &= C + \mathbf{d}_k^\intercal \mathbf{A}_k \mathbf{d}_k + \mathbf{b}_k\mathbf{d}_k, \end{aligned} \tag{9.2}$$

with

$$\mathbf{A}_k = \sum_q (\mathbf{x}_q^k)^2 \mathbf{\Phi}^{(q)\intercal}\mathbf{\Phi}^{(q)},$$

and

$$\mathbf{b}_k = -2\sum_q \mathbf{x}_q^k \left(\mathbf{y}_q - \mathbf{\Phi}^{(q)} \sum_{i\neq k} \hat{\mathbf{d}}_i \mathbf{x}_q^i\right)^\intercal \mathbf{\Phi}^{(q)}.$$

Now let us focus on the second term on the right hand side of (9.1). As it will become clearer in the following steps, this term includes the uncertainty related to

the selection of the atoms of the dictionary in terms of the variance of the components of $\mathbf{X}$. It can be written as

$$\sum_q \left\langle \| \mathbf{\Phi}^{(q)} \mathbf{d}_k \left(\hat{\mathbf{x}}_q^k - \mathbf{x}_q^k \right) \|_2^2 \right\rangle_{\mathbf{\Theta} \setminus \mathbf{d}_k} + 2 \sum_q \left\langle \left(\mathbf{\Phi}^{(q)} \mathbf{d}_k \left(\hat{\mathbf{x}}_q^k - \mathbf{x}_q^k \right) \right)^\intercal \mathbf{\Phi}^{(q)} \sum_{i \neq k} \hat{\mathbf{d}}_i \left(\hat{\mathbf{x}}_q^i - \mathbf{x}_q^i \right) \right\rangle_{\mathbf{\Theta} \setminus \mathbf{d}_k} . \tag{9.3}$$

At this point, the first term in (9.3) is

$$\sum_q \left\langle \| \mathbf{\Phi}^{(q)} \mathbf{d}_k \left(\hat{\mathbf{x}}_q^k - \mathbf{x}_q^k \right) \|_2^2 \right\rangle_{\mathbf{\Theta} \setminus \mathbf{d}_k} = \mathbf{d}_k^\intercal \mathbf{c}_k \mathbf{d}_k, \tag{9.4}$$

with

$$\mathbf{c}_k = \sum_q \mathbf{\Phi}^{(q)\intercal} \mathbf{\Phi}^{(q)} \mathbf{\Sigma}_{\mathbf{x}_q}(k,k).$$

In the same way, if we consider the second term of (9.3) we can write

$$2 \sum_q \left\langle \left(\mathbf{\Phi}^{(q)} \mathbf{d}_k \left(\hat{\mathbf{x}}_q^k - \mathbf{x}_q^k \right) \right)^\intercal \mathbf{\Phi}^{(q)} \sum_{i \neq k} \hat{\mathbf{d}}_i \left(\hat{\mathbf{x}}_q^i - \mathbf{x}_q^i \right) \right\rangle_{\mathbf{\Theta} \setminus \mathbf{d}_k} = \mathbf{a}_k \mathbf{d}_k \tag{9.5}$$

with

$$\mathbf{a}_k = 2 \sum_q \left(\sum_{i \neq k} \mathbf{\Sigma}_{\mathbf{x}_q}(i,k) \hat{\mathbf{d}}_k \right)^\intercal \mathbf{\Phi}^{(q)\intercal} \mathbf{\Phi}^{(q)}.$$

If we then substitute (9.2) (9.4) and (9.5) into (9.1) we obtain

$$\sum_q \left\langle \| \mathbf{y}_q - \mathbf{\Phi}^{(q)} \mathbf{D} \mathbf{x}_q \|_2^2 \right\rangle_{\mathbf{\Theta} \setminus \mathbf{d}_k} = \mathbf{d}_k^\intercal \mathbf{H} \mathbf{d}_k + \mathbf{g}_k \mathbf{d}_k + C,$$

with

$$\begin{aligned} \mathbf{H} &= \mathbf{A}_k + \mathbf{c}_k, \\ \mathbf{g}_k &= \mathbf{b}_k + \mathbf{a}_k, \end{aligned}$$

which is a quadratic problem that can be solved by putting the derivative equal to zero as

$$(\mathbf{H} + \mathbf{H}^\intercal) \mathbf{d}_k + \mathbf{g}_k^\intercal = 0,$$

leading to the update step for the $\mathbf{d}_k$ column of the dictionary $\mathbf{D}$

$$\mathbf{d}_k = -(\mathbf{H} + \mathbf{H}^\intercal)^\dagger \mathbf{g}_k^\intercal. \tag{9.6}$$

The whole compressive BKSVD algorithm is summarized in Algorithm 5.

Algorithm 5 Pseudocode for compressive BKSVD algorithm

Input: $\mathbf{Y}$, initial normalized $\mathbf{D}$ and $\mathbf{\Phi}^{(q)}$ $\forall q \in 1 \dots Q$
Output: $\hat{\mathbf{D}}$, $\hat{\mathbf{\Gamma}}$, the posterior approximations q($\mathbf{x}_q$),
$q = 1, \dots, Q$
initialize $\mathbf{\Gamma}$ and $\boldsymbol{\lambda}$ to zero
while not converged **do**
    for q in $1, \dots, Q$ **do**
        while not converged **do**
            Choose a $k \in \{1, \dots, K\}$
    (or equivalently choose a γ_{kj})
            Find optimal $\hat{\gamma}_{kq}$ using (8.20)
            Update $\mathbf{\Sigma}_{\mathbf{x}_q}$ and $\hat{\mathbf{x}}_q$ based on $\hat{\gamma}_{kq}$
            Update s_{kq} and h_{kq}
            Update $\hat{\lambda}_j$ and $\hat{\nu}_j$
        end while
    end for
    for k in $1, \dots, K$ **do**
        Update $\mathbf{d}_k$ using (9.6)
    end for
    Update $\hat{\beta}$
end while

9.3 Experimental results

In this section we numerically evaluate the proposed compressive BKSVD and compare it with the compressive KSVD introduced in [8]. The compressive KSVD follows the standard KSVD approach which alternates the use of OMP-like algorithms to obtain a sparse representation with the dictionary update step. It is in this last step that lies the main difference with respect to the KSVD since a different dictionary update step is needed due to the fact that different sensing matrices are employed and the SVD step can not be used.

Our experiments are performed using synthetically generated data. More in detail, we generate a random dictionary $\mathbf{D} \in \mathbb{R}^{50\times 10}$ which is column normalized. Then, Q sparse signals $\mathbf{x}_q$ are generated with 3 non-zero elements each having values drawn from a normal distribution according to $\mathcal{N}(0,80)$. Having the dictionary and the sparse representation, we obtain $\mathbf{s}_q = \mathbf{D}\mathbf{x}_q$ and the compressed measurements as $\mathbf{y}_q = \mathbf{\Phi}^{(q)}\mathbf{s}_q$.

At this point, in order to evaluate the performance of the proposed method we compare the compressive BKSVD with the compressive KSVD in terms of correctly learned atoms in the dictionary and recovery error. To detect the number of correctly learned atoms we identify an atom as correctly learned if the correlation between this atom and any of the atoms of true dictionary is larger than 0.98. For what concerns the recovery error we define the mean relative recovery error as $e_r = (1/Q)\sum_q \|\hat{\mathbf{D}}\hat{\mathbf{x}}_k - \mathbf{D}\mathbf{x}_k\|_2^2/\|\mathbf{D}\mathbf{x}_k\|_2^2$. We tested these parameters using different compression ratios M/N and different number of training signals Q. The results are then averaged over 10 different runs. It is worth noting that since the compressive BKSVD typically requires more iterations to reach convergence, we fixed for both algorithms the number of iterations to 1000. Moreover, since the compressive KSVD requires the knowledge of the *exact* sparsity of each column of $\mathbf{X}$, we gave to the algorithm the true sparsity level we set to 3.

As we can see in Fig. 9.1 and 9.2, except for very small compression ratios, the proposed technique outperforms the compressive KSVD. In particular, a compression ratio value higher than 0.3 allows the proposed technique to reach very small recovery errors and to correctly learn all the atoms in the dictionary. On the other hand, smaller values of the compression ratio seem to favor the compressive KSVD. Nevertheless at such compression ratios even though the technique in [8] performs better is still unable to reach acceptable recovery errors making the recovered signal too different from the true one. This behavior at small compression ratio values can be explained by the fact that the proposed technique does not have access to any prior information differently from the compressive KSVD which has the knowledge of the sparsity of the signals. In fact, when the signals are extremely compressed, having access to some prior information can lead to improved performance. Moreover, we can state that the number of training signals is a crucial parameter which

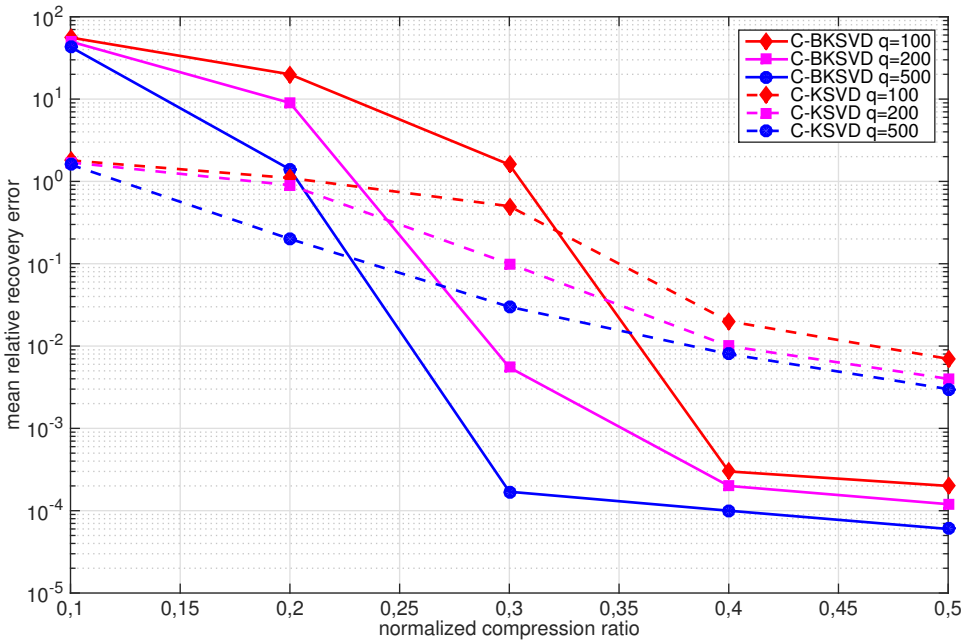


Figure 9.1: Mean relative error comparison for the proposed compressive BKSVD and [8] at different compression ratios using a different number of training signals $Q = \{100,200,500\}$.

can determine the success of the dictionary learning and thus the recovery. For the experiments we performed it can be seen that a number of training signals $Q < 200$ is not enough to reach acceptable performance.

The advantage of such technique over the compressive KSVD is not only that it reaches better performance, but also that we do not need any additional information in the process. As in the BKSVD all the quantities are automatically estimated as an additional feature of the method. More in detail, the knowledge of the true sparsity is typically hard to guess and wrong guesses may lead to inconsistent results.

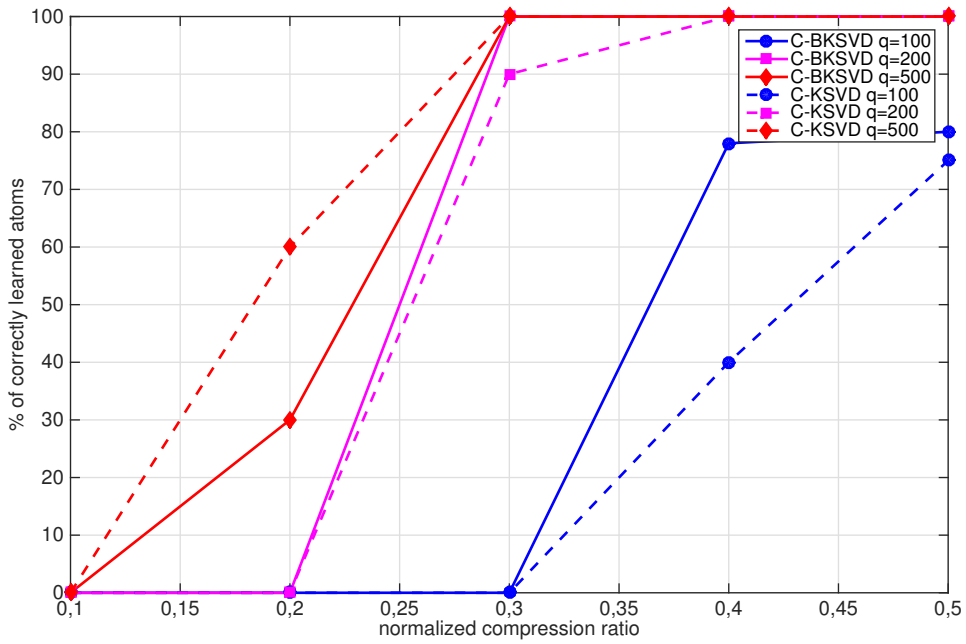


Figure 9.2: Percentage of correctly learned atoms for the proposed compressive BKSVD and [8] at different compression ratios using a different number of training signals $Q = \{100,200,500\}$.

Chapter 10

Conclusions

In this thesis we explored the possibility of extracting information about the original signal directly in the compressed domain. In order to achieve this goal we introduced an AR modeling of the underlying signal which allowed us to characterize a signal by means of few parameters. Next, we showed that these parameters can be efficiently estimated with good accuracy in the compressed domain employing an ad-hoc sensing matrix coupled with a LS or Bayesian estimators depending on the model order. On top of this novel framework we developed few interesting applications:

- Compressive covariance sensing
- Compressive classification
- Adaptive compressive imaging

Then, we focused on compressive signal processing in the distributed setting. In this case when the noise corrupting CS measurements is non-white the knowledge of the covariance matrix of the noise process is of paramount importance to perform compressive signal processing applications. We addressed this problem by introducing an iterative and fully distributed algorithm thanks which each node of the network can obtain the covariance matrix of the colored noise process. We also showed that this technique allows the nodes to improve the performance of compressive signal processing operations, *e.g.,* signal detection. The main advantage of such technique comes from the noise modeling which has been modeled according to an AR process. Hence, being able to define the covariance matrix of the process with few parameters allowed us to reach excellent performance at very low communication costs.

The last topic we covered in this thesis was dictionary learning. In particular, concerning the non-compressed case, we proposed a Bayesian alternative to the well-known KSVD algorithm for dictionary learning. Going Bayesian allowed us to

improve the overall performance by incorporating all the uncertainties in the process and to automatically estimate all the parameters of the model which would otherwise hard to tune. Moreover, sparse representations are not only useful *per se* but are part of the foundation of the CS. This said, we also showed how the CS framework can take advantage by using this technique to estimate the best sparsifying basis which would ultimately lead to a better recovery.

10.1 Future work

With the techniques we described in this work, we expanded the ways in which CS measurements can be used as a direct source of information rather than a proxy for the signal recovery. However, there are still some paths which can be followed and which may lead to interesting research problems. These would eventually lead to advances in compressive signals processing and signal recovery. In the follow we describe some future work which may be interesting to further investigate.

- *Provide theoretical guarantees for the ad-hoc sensing matrix* In this thesis we experimentally showed that the proposed ad-hoc sensing matrix construction provides good CS recovery capabilities. Even though is quite a tough task to provide RIP guarantees for the sensing matrix we introduce, it would be useful to have such results in order to have a more complete characterization of the sensing process we proposed in this work.

- *Consider different signal modeling* Whilst the AR modeling we extensively discussed in this thesis turned out to be an effective way yo model a signal in order to extract information from CS measurements, it would be of interest to consider other kinds of modeling. If we consider 2D signals such as images, it would be natural to consider other kinds of models which exploit the bidimensional structure of the signal. Since we are more interested in parametric models because of their compactness, simultaneous AR models are an interesting model to investigate.

- *Extend compressive BKSVD* As shown in Chapter 9, the Bayesian KSVD algorithm can be successfully adapted to the CS scheme in order to learn a good sparsifying basis which can improve the recovery process. However, the results we showed are preliminary and more work is needed. In particular it would be interesting to perform a more complete analysis of the probable gains that this technique can provide when the signals to be recovered are sparse in the subspace spanned by the columns of a suited dictionary but are compressible in a standard fixed basis. Moreover, it would also be of interest

to assess the performance of this approach with some real applications which take advantage of a good basis such as compressive image denoising.

Bibliography

[1] S. Mun and J. E. Fowler, "Block compressed sensing of images using directional transforms," in *Image Processing (ICIP), 2009 16th IEEE International Conference on*. IEEE, 2009, pp. 3021–3024.

[2] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, p. 4311, 2006.

[3] D. Giacobello, "Retrieving Sparse Patterns Using a Compressed Sensing Framework: Applications to Speech Coding Based on Sparse Linear Prediction," 2010.

[4] D. Scharstein and R. Szeliski, "High-accuracy stereo depth maps using structured light," in *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, vol. 1. IEEE, 2003, pp. I–195.

[5] C. L. Zitnick, S. B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski, "High-quality video view interpolation using a layered representation," in *ACM Transactions on Graphics (TOG)*, vol. 23, no. 3. ACM, 2004, pp. 600–608.

[6] J. M. Bioucas-Dias, D. Cohen, and Y. C. Eldar, "Covalsa: Covariance estimation from compressive measurements using alternating minimization," in *Signal Processing Conference (EUSIPCO), 2014 Proceedings of the 22nd European*. IEEE, 2014, pp. 999–1003.

[7] K. Dana, B. Van-Ginneken, S. Nayar, and J. Koenderink, "Reflectance and Texture of Real World Surfaces," *ACM Transactions on Graphics (TOG)*, vol. 18, no. 1, pp. 1–34, Jan 1999.

[8] F. P. Anaraki and S. M. Hughes, "Compressive k-svd," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, May 2013, pp. 5469–5473.

[9] E. J. Candès, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," *Information Theory, IEEE Transactions on*, vol. 52, no. 2, pp. 489–509, 2006.

[10] D. L. Donoho, "Compressed sensing," *Information Theory, IEEE Transactions on*, vol. 52, no. 4, pp. 1289–1306, 2006.

[11] M. A. Davenport, P. T. Boufounos, M. B. Wakin, and R. G. Baraniuk, "Signal

processing with compressive measurements," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 4, no. 2, pp. 445–460, 2010.

[12] S. R. Becker, "Practical compressed sensing: modern data acquisition and signal processing," Ph.D. dissertation, Citeseer, 2011.

[13] G. W. Wornell, "Wavelet-based representations for the 1/f family of fractal processes," *Proceedings of the IEEE*, vol. 81, no. 10, pp. 1428–1450, 1993.

[14] R. G. Baraniuk, V. Cevher, M. F. Duarte, and C. Hegde, "Model-based compressive sensing," *Information Theory, IEEE Transactions on*, vol. 56, no. 4, pp. 1982–2001, 2010.

[15] R. G. Baraniuk and M. B. Wakin, "Random projections of smooth manifolds," *Foundations of Computational Mathematics*, vol. 9, no. 1, pp. 51–77, 2009.

[16] A. Eftekhari and M. B. Wakin, *New Analysis of Manifold Embeddings and Signal Recovery from Compressive Measurements*. Elsevier Inc., 2013, vol. 39, no. 1. [Online]. Available: http://dx.doi.org/10.1016/j.acha.2014.08.005

[17] B. R. Rubinstein, A. M. Bruckstein, and M. Elad, "Dictionaries for Sparse Representation Modeling," *Proceedings of the IEEE*, vol. 98, no. 6, pp. 1045–1057, 2010.

[18] E. P. Simoncelli, W. T. Freeman, E. H. Adelson, and D. J. Heeger, "Shiftable multiscale transforms," *IEEE Transactions on Information Theory*, vol. 38, no. 2, pp. 587–607, 1992.

[19] B. a. Olshausen and D. J. Field, "Emergence of simple-cell receptive field properties by learning a sparse code for natural images." *Nature*, vol. 381, no. 6583, pp. 607–609, 1996.

[20] B. A. Olshausen and D. J. Field, "Sparse coding with an overcomplete basis set: A strategy employed by v1?" *Vision research*, vol. 37, no. 23, pp. 3311–3325, 1997.

[21] M. Elad and M. Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Transactions on Image Processing*, vol. 15, no. 12, pp. 3736–3745, 2006.

[22] M. Elad, M. A. Figueiredo, and Y. Ma, "On the role of sparse and redundant representations in image processing," *Proceedings of the IEEE*, vol. 98, no. 6, pp. 972–982, 2010.

[23] J. Yang, J. Wright, T. Huang, and Y. Ma, "Image super-resolution as sparse representation of raw image patches," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2008, pp. 1–8.

[24] J. Mairal, M. Elad, and G. Sapiro, "Sparse representation for color image restoration," *IEEE Transactions on Image Processing*, vol. 17, no. 1, pp. 53–69, 2008.

[25] I. Ramirez, P. Sprechmann, and G. Sapiro, "Classification and clustering via dictionary learning with structured incoherence and shared features," in *IEEE*

Conference on Computer Vision and Pattern Recognition, 2010, pp. 3501–3508.

[26] Q. Zhang and B. Li, "Discriminative k-svd for dictionary learning in face recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2010, pp. 2691–2698.

[27] G. K. Wallace, "The jpeg still picture compression standard," *Consumer Electronics, IEEE Transactions on*, vol. 38, no. 1, pp. xviii–xxxiv, 1992.

[28] A. Skodras, C. Christopoulos, and T. Ebrahimi, "The jpeg 2000 still image compression standard," *Signal Processing Magazine, IEEE*, vol. 18, no. 5, pp. 36–58, 2001.

[29] K. Brandenburg, "Mp3 and aac explained," in *Audio Engineering Society Conference: 17th International Conference: High-Quality Audio Coding.* Audio Engineering Society, 1999.

[30] E. Candès and T. Tao, "Decoding by linear programming," *IEEE Trans. Inform. Theory, vol 52, no. 4, pp. 4203-4215*, Dec. 2005.

[31] J. Bourgain, S. Dilworth, K. Ford, S. Konyagin, and D. Kutzarova, "Explicit construction of rip matrices and related problems," *To appear - Duke Math*, Jan. 2011.

[32] R. DeVore, "Deterministic constructions of compressed sensing matrices," *J. Complex, pp. 918-925*, 2007.

[33] E. Candés and M. B. Wakin, "An introduction to compressive sampling," *IEEE Signal Processing Magazine, pp. 23-24*, Mar. 2008.

[34] T. Strohmer and R. Heath, "Grassmannian frames with applications to coding and communication," *Appl. Comput. Harmon. Anal., 14(3) pp. 257 – 275*, 2003.

[35] R. Baraniuk, M. Davenport, R. DeVore, and M. Wakin, "A simple proof of the restricted isometry property for random matrices," *Const. Approx., 28(3):253-263*, 2008.

[36] D. Achliotpas, "Database-friendly random projections," *Proc. 20th Annual ACM SIGACT-SIGMOD-SIGART Symp. on Principles of Database Systems, pp. 274-281*, 2001.

[37] M. Fornasier and H. Rauhut, "Compressive sensing," in *Handbook of mathematical methods in imaging.* Springer, 2011, pp. 187–228.

[38] E. Candès and J. Romberg, "Sparsity and incoherence in compressive sampling," *Inverse prob., vol. 23, no. 3, pp. 969-985*, 2007.

[39] M. F. Duarte, S. Sarvotham, D. Baron, M. B. Wakin, and R. G. Baraniuk, "Distributed compressed sensing of jointly sparse signals," in *Asilomar Conf. Signals, Sys., Comput*, 2005, pp. 1537–1541.

[40] B. K. Natarajan, "Sparse approximate solutions to linear systems," *SIAM J. Comput., 24:227-234*, 1995.

[41] G. Cormode and S. Muthukrishnan, "Combinatorial algorithms for compressed sensing," in *Structural Information and Communication Complexity*. Springer, 2006, pp. 280–294.

[42] J. A. Tropp, "Greed is good: Algorithmic results for sparse approximation," *IEEE Trans. Inform. Theory, 51(3)1030-1051*, 2006.

[43] R. Berinde, A. Gilbert, P. Indyk, H. Karloff, and M. Strauss, "Combining geometry and combinatorics: A unified approach to sparse signal recovery," *Communication, Control, and Computing, 2008 46th Annual Allerton Conference on, pp. 798-805*, 2008.

[44] A. Moghadam and H. Radha, "On combinatorial approaches to compressed sensing," *http://www.egr.msu.edu/waves/people/nima/isit-2/isit-draft-extended-version.pdf*.

[45] S. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Trans. Signal Processing, 41(12), pp. 3397 – 3415*, 1993.

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

[47] M. A. Davenport, M. F. Duarte, Y. C. Eldar, and G. Kutyniok, "Introduction to compressed sensing," *Preprint*, vol. 93, no. 1, p. 2, 2011.

[48] Z. Ben-Haim and Y. C. Eldar, "The cramer-rao bound for estimating a sparse parameter vector," *IEEE Trans. Signal Processing, 58(6):3384 – 3389*, 2010.

[49] D. Donoho, M. Elad, and V. Temlyahov, "Stable recovery of sparse overcomplete representations in the presence of noise," *IEEE Trans. Inform. Theory, 52(1):6 – 18*, 2006.

[50] Z. Ben-Haim, Y. C. Eldar, and M. Elad, "Coherence-based performance guarantees for estimating a sparse vector under random noise," *IEEE Trans. Signal Processing, 58(10):5030 – 5043*, 2010.

[51] B. S. Atal and J. Remde, "A new model of lpc excitation for producing natural-sounding speech at low bit rates," in *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'82.*, vol. 7. IEEE, 1982, pp. 614–617.

[52] M. Akay, *Biomedical signal processing*. Academic Press, 2012.

[53] X. Wu and X. Zhang, "Adaptive structured recovery of compressive sensing via piecewise autoregressive modeling," in *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*. IEEE, 2010, pp. 3906–3909.

[54] R. Bos, S. De Waele, and P. M. Broersen, "Autoregressive spectral estimation by application of the burg algorithm to irregularly sampled data," *Instrumentation and Measurement, IEEE Transactions on*, vol. 51, no. 6, pp. 1289–1294, 2002.

[55] S. M. Kay, *Modern spectral estimation*. Pearson Education India, 1988.

[56] H. Rauhut, "Circulant and toeplitz matrices in compressed sensing," *arXiv preprint arXiv:0902.4394*, 2009.

[57] E. J. Candes and T. Tao, "Near-optimal signal recovery from random projections: Universal encoding strategies?" *Information Theory, IEEE Transactions on*, vol. 52, no. 12, pp. 5406–5425, 2006.

[58] J. D. Cryer and N. Kellet, *Time series analysis*. Springer, 1986, vol. 101.

[59] D. G. Tzikas, A. C. Likas, and N. P. Galatsanos, "The variational approximation for bayesian inference," *Signal Processing Magazine, IEEE*, vol. 25, no. 6, pp. 131–146, 2008.

[60] A. Torralba and A. Oliva, "Statistics of natural image categories," *Network: computation in neural systems*, vol. 14, no. 3, pp. 391–412, 2003.

[61] M. B. Wakin, J. N. Laska, M. F. Duarte, D. Baron, S. Sarvotham, D. Takhar, K. F. Kelly, and R. G. Baraniuk, "An architecture for compressive imaging," in *Image Processing, 2006 IEEE International Conference on*. IEEE, 2006, pp. 1273–1276.

[62] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *Image Processing, IEEE Transactions on*, vol. 13, no. 4, pp. 600–612, 2004.

[63] M. A. Lexa, M. E. Davies, J. S. Thompson, and J. Nikolic, "Compressive power spectral density estimation," in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*. IEEE, 2011, pp. 3884–3887.

[64] C.-P. Yen, Y. Tsai, and X. Wang, "Wideband spectrum sensing based on sub-nyquist sampling," *Signal Processing, IEEE Transactions on*, vol. 61, no. 12, pp. 3028–3040, 2013.

[65] Z. Tian and G. B. Giannakis, "Compressed sensing for wideband cognitive radios," in *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, vol. 4. IEEE, 2007, pp. IV–1357.

[66] J. D. Krieger, Y. Kochman, and G. W. Wornell, "Design and analysis of multi-coset arrays," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 3781–3785.

[67] P. Pal and P. Vaidyanathan, "Nested arrays: a novel approach to array processing with enhanced degrees of freedom," *Signal Processing, IEEE Transactions on*, vol. 58, no. 8, pp. 4167–4181, 2010.

[68] Y. Abramovich, D. Gray, A. Y. Gorokhov, N. K. Spencer *et al.*, "Positive-definite toeplitz completion in doa estimation for nonuniform linear antenna arrays. i. fully augmentable arrays," *Signal Processing, IEEE Transactions on*, vol. 46, no. 9, pp. 2458–2471, 1998.

[69] D. Romero, D. D. Ariananda, Z. Tian, and G. Leus, "Compressive covariance sensing."

[70] D. Romero, R. Lopez-Valcarce, and G. Leus, "Compression limits for random vectors with linearly parameterized second-order statistics," *Information Theory, IEEE Transactions on*, vol. 61, no. 3, pp. 1410–1425, 2015.

[71] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 31, no. 2, pp. 210–227, 2009.

[72] G. Tzagkarakis, G. Tsagkatakis, J.-L. Starck, and P. Tsakalides, "Compressive video classification in a low-dimensional manifold with learned distance metric," in *Signal Processing Conference (EUSIPCO), 2012 Proceedings of the 20th European*. IEEE, 2012, pp. 155–159.

[73] M. A. Davenport, M. F. Duarte, M. B. Wakin, J. N. Laska, D. Takhar, K. F. Kelly, and R. G. Baraniuk, "The smashed filter for compressive classification and target recognition," in *Electronic Imaging 2007*. International Society for Optics and Photonics, 2007, pp. 64 980H–64 980H.

[74] K. Zhang, L. Zhang, and M.-H. Yang, "Real-time compressive tracking," in *Computer Vision–ECCV 2012*. Springer, 2012, pp. 864–877.

[75] M. Szummer and R. W. Picard, "Temporal texture modeling," in *Image Processing, 1996. Proceedings., International Conference on*, vol. 3. IEEE, 1996, pp. 823–826.

[76] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, D. S. Pallett, and N. L. Dahlgren, "DARPA TIMIT acoustic phonetic continuous speech corpus CDROM," 1993. [Online]. Available: http://www.ldc.upenn.edu/Catalog/LDC93S1.html

[77] R. Bachu, S. Kopparthi, B. Adapa, and B. Barkana, "Separation of voiced and unvoiced using zero crossing rate and energy of the speech signal," in *American Society for Engineering Education (ASEE) Zone Conference Proceedings*, 2008, pp. 1–7.

[78] S. M. Kay, "Fundamentals of statistical signal processing, vol. ii: Detection theory," *Signal Processing. Upper Saddle River, NJ: Prentice Hall*, 1998.

[79] S. V. Macua, P. Belanovic, and S. Zazo, "Consensus-based distributed principal component analysis in wireless sensor networks," in *Signal Processing Advances in Wireless Communications (SPAWC), 2010 IEEE Eleventh International Workshop on*. IEEE, 2010, pp. 1–5.

[80] A. Bertrand and M. Moonen, "Low-complexity distributed total least squares estimation in ad hoc sensor networks," *Signal Processing, IEEE Transactions on*, vol. 60, no. 8, pp. 4321–4333, 2012.

[81] F. Penna and S. Stanczak, "Decentralized Eigenvalue Algorithms for Distributed Signal Detection in Wireless Networks," *IEEE Transactions on Signal Processing*, no. c, pp. 1–1, 2014. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6963379

[82] N. Ghadban, P. Honeine, C. Francis, F. Mourad-Chehade, and J. Farah,

"Strategies for principal component analysis in wireless sensor networks," in *Proc. eighth IEEE Sensor Array and Multichannel Signal Processing Workshop*, 2014.

[83] G. Coluccia, A. Roumy, and E. Magli, "Operational rate–distortion performance of single-source and distributed compressed sensing," 2013.

[84] J. D. Hamilton, *Time series analysis*. Princeton university press Princeton, 1994, vol. 2.

[85] W. Zheng, "Study of a least-squares-based algorithm for autoregressive signals subject to white noise," *Mathematical Problems in Engineering*, vol. 3, no. October 2002, pp. 93–101, 2003. [Online]. Available: http://www.hindawi.com/journals/mpe/2003/526073/abs/

[86] B. Johansson, T. Keviczky, M. Johansson, and K. H. Johansson, "Subgradient methods and consensus algorithms for solving convex optimization problems," in *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*. IEEE, 2008, pp. 4185–4190.

[87] W. Förstner and B. Moonen, "A Metric for Covariance Matrices," *Quo vadis geodesia*, vol. 66, pp. 113–128, 1999. [Online]. Available: http://www.uni-stuttgart.de/gi/research/schriftenreihe/quo_vadis/pdf/foerstner.pdf

[88] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online dictionary learning for sparse coding," in *Proceedings of the 26th Annual International Conference on MachineLearning*, 2009, pp. 689–696.

[89] M. Zhou, H. Chen, J. Paisley, L. Ren, G. Sapiro, and L. Carin, "Non-parametric Bayesian dictionary learning for sparse image representations," *IEEE Transactions on Image Processing*, 2009.

[90] J. Shi, X. Ren, G. Dai, J. Wang, and Z. Zhang, "A non-convex relaxation approach to sparse dictionary learning," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 1809–1816.

[91] K. Skretting and K. Engan, "Recursive least squares dictionary learning algorithm," *IEEE Transactions on Signal Processing*, vol. 58, no. 4, pp. 2121–2130, 2010.

[92] K. Engan, K. Skretting, and J. H. Husø y, "Family of iterative LS-based dictionary learning algorithms, ILS-DLA, for sparse signal representation," *Digital Signal Processing: A Review Journal*, vol. 17, no. 1, pp. 32–49, 2007.

[93] M. S. Lewicki and T. J. Sejnowski, "Learning overcomplete representations." *Neural computation*, vol. 12, no. 2, pp. 337–365, 2000.

[94] K. Engan, S. O. Aase, and J. Hakon Husoy, "Method of optimal directions for frame design," in *Acoustics, Speech, and Signal Processing, 1999. Proceedings., 1999 IEEE International Conference on*, vol. 5. IEEE, 1999, pp. 2443–2446.

[95] K. Kreutz-Delgado, J. F. Murray, B. D. Rao, K. Engan, T.-W. Lee, and T. J. Sejnowski, "Dictionary learning algorithms for sparse representation." *Neural computation*, vol. 15, no. 2, pp. 349–396, 2003.

[96] I. F. Gorodnitsky and B. D. Rao, "Sparse signal reconstruction from limited data using focuss: A re-weighted minimum norm algorithm," *Signal Processing, IEEE Transactions on*, vol. 45, no. 3, pp. 600–616, 1997.

[97] D. M. Bradley and J. A. Bagnell, "Differential Sparse Coding," *Neural Information Processing Systems*, 2008.

[98] M. Girolami, "A variational method for learning sparse and overcomplete representations." *Neural computation*, vol. 13, no. 11, pp. 2517–2532, 2001.

[99] M. Zhou, H. Chen, J. Paisley, L. Ren, L. Li, Z. Xing, D. Dunson, G. Sapiro, and L. Carin, "Nonparametric bayesian dictionary learning for analysis of noisy and incomplete images," *IEEE Transactions on Image Processing*, vol. 21, no. 1, pp. 130–144, 2012.

[100] L. Li, J. Silva, M. Zhou, and L. Carin, "Online Bayesian dictionary learning for large datasets," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2012, pp. 2157–2160.

[101] M. Elad, *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing*. Springer Verlag, 2010.

[102] J. Cai, E. J. Candès, and Z. Shen, "A singular value thresholding algorithm for matrix completion," *SIAM Journal on Optimization*, vol. 20, no. 4, pp. 1956–1982, 2010.

[103] S. Babacan, R. Molina, and A. Katsaggelos, "Bayesian compressive sensing using Laplace priors," *IEEE Transactions on Image Processing*, vol. 19, no. 1, pp. 53–63, 2010.

[104] S. D. Babacan, R. Molina, M. N. Do, and A. K. Katsaggelos, "Bayesian blind deconvolution with general sparse image priors," in *Computer Vision–ECCV 2012*. Springer, 2012, pp. 341–355.

[105] C. M. Bishop, *Pattern recognition and machine learning*. Springer New York, 2006.

[106] M. E. Tipping and A. Faul, "Fast marginal likelihood maximisation for sparse Bayesian models," in *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*, 2003, pp. 3–6.

[107] Z. Chen, R. Molina, and A. K. Katsaggelos, "Automated recovery of compressedly observed sparse signals from smooth background," *IEEE Signal Processing Letters*, vol. 21, no. 8, pp. 1012–1016, Aug 2014.

[108] S. D. Babacan, L. Mancera, R. Molina, and A. K. Katsaggelos, "Bayesian Compressive Sensing Using Non-Convex Priors," *European Signal Processing Conference 2009 EUSIPCO09*, 2009.