Accelerated Sparse Coding with Overcomplete Dictionaries for Image Processing Applications

Submitted in partial fulfillment of the requirements for

the degree of

Doctor of Philosophy

in

ELECTRICAL AND COMPUTER ENGINEERING

Hugo R. Gonçalves

B.S., Electrical and Computer Engineering, FEUP
M.S., Electrical and Computer Engineering, FEUP

Carnegie Mellon University
Pittsburgh, PA

August, 2015

Abstract

Image processing problems have always been challenging due to the complexity of the signal. These problems comprise image enhancement, such as denoising, inpainting or digital zoom, decomposing the image into relevant parts, such as background subtraction and segmentation, image compression, object identification or recognition, and others. The complexity of the signal comes from the dimension of the images and by the large number of variations that can occur, such as contrast, luminance, occlusions, perspective, scale, rotation, etc.. This variety of problems has been tackled with different methods, with many of them depending on a transformation of the signal into a more meaningful representation. For instance, image compression relies on the fact that images can be accurately represented by a small number of elements of a proper basis. These include the Fourier or wavelet domains, but in fact any collection of elements can be used to efficiently represent images, as long as they constitute sources commonly found in images. This suggests that images belong to a type of signal that can be targeted by the sparse coding framework.

Sparse coding assumes that a certain class of signals can be expressed linearly by a small number of elements from a given set or frame. A range of image processing problems can then be solved by cleverly formulating an optimization problem with three main components: an objective function that matches the criterion to optimize, an appropriate frame or dictionary, and a penalty or constraint that enforces sparsity of the image representation. Strictly speaking, the sparsity of a vector is defined by

the number of non-zero elements (ℓ_0-norm), but this measure renders the problem NP-hard. Its convex relaxation, the sum of absolute values (ℓ_1-norm), also measures sparsity to some extent and is commonly used as an alternative. When the optimization problem is convex, the compressive sensing theory guarantees uniqueness (or at least bounded error) of the recovered sparse signal under certain conditions. This, in turn, leads to guarantees of performance for the solution of the image processing problems based on sparse coding. It turns out that many image processing problems that were solved with tailored methods can be solved with sparse coding with state-of-the-art results, specially when they use a dictionary learned from a dataset of images.

However, the toll that is paid for the state-of-the-art performance is a large computational cost. Although sparse coding solvers are now much faster than the first ones, the time to solve is still a major drawback. Finding faster solvers is still a compelling topic of research, since it will drive a large number of applications in the practical domain.

The first contribution of this dissertation is to benchmark several solvers under different conditions. These cover several values for the coherence of the dictionary, sparsity of the true signal and the regularization parameter. The benchmark confirms that the solver named Dual Alternating Direction Method of Multipliers (DADMM) has generally the best performance and therefore can be set as a standard and is a solid base to create faster solvers. The same algorithm is benchmarked with greedy methods for ℓ_0-norm based problems to compare speed and recovery properties. Contrary to the common belief that greedy algorithms are faster than convex solvers, DADMM outperforms most of them in terms of speed. Regarding recovery properties, the theoretical guarantees are better when the sparsity is measured by the ℓ_1-norm than for any greedy solver. In practice, the opposite is seen, i.e., greedy solvers have better recovery properties than ℓ_1-norm based solvers.

This work proposes a new version of DADMM, in which the penalty parameter can be adjusted at each iteration at no extra cost. As a consequence, additional simplifications are possible that accelerate the algorithm. A thorough study is undertaken to evaluate what is the best sequence of penalty parameter values, its relation to variables existent in the algorithm and how it improves convergence. A relation between optimal penalty parameter and reconstruction error is proposed, which improves performance for the problem in question. Several examples are presented that show that the proposed DADMM reaches the same convergence level in less than half the time compared with standard DADMM.

Furthermore, the unique properties of DADMM are further explored to accelerate Virtual Probe (VP), a method used to reduce wafer test cost by reducing the number of effective measurements required. The wafer map for each test item constitutes an image, for which compressive sensing can be applied. The properties of DADMM become an advantage when the dictionary is a partial fast transform, as is the case of the DCT transform used in VP. Results show that DADMM tailored to VP can reach speed-ups up to 38$\times$ compared with a standard interior-point solver.

The algorithm could potentially be further improved by reducing the complexity of matrix-vector multiplications, with particular advantage for highly overcomplete dictionaries. This could be achieved by predicting the zero elements in the next iterated solution and avoiding the computation of the corresponding vector dot-products, since their value is irrelevant. Furthermore, a better and broader penalty parameter rule could be attained if the suggested methodology was used in a larger range of test cases. The result would certainly be a parameter-free standard solver for ℓ_1-norm regularized problems.

Resumo

Problemas de processamento de imagens têm sido sempre um desafio devido à complexidade do sinal. Estes problemas incluem melhoria de imagem, como remoção de ruído, recuperação de pixeis ou zoom digital; decompor a imagem em partes relevantes, como a subtração de fundo e segmentação; compressão de imagem; identificação ou reconhecimento de objetos, entre outros. A complexidade do sinal advém da dimensão das imagens e pelas variações que podem ocorrer, tais como no contraste, na luminância, oclusões, perspectiva, escala, rotação, etc.. Estes vários problemas foram resolvidos com métodos diferentes, muitos deles dependendo de uma transformação do sinal para uma representação mais significativa. Por exemplo, a compressão de imagem baseia-se no facto de que as imagens podem ser representadas com precisão por um pequeno número de elementos de uma base adequada. Estas bases incluem os domínios de Fourier ou de wavelet, mas, de facto, qualquer conjunto de elementos pode ser usado para representar imagens de forma eficiente, desde que estes constituam padrões normalmente encontrados em imagens. Isto sugere que as imagens pertencem a um tipo de sinal que pode ser visado pelo método de codificação esparsa.

A codificação esparsa assume que uma certa classe de sinais pode ser expressa de forma linear por um pequeno número de elementos de um dado conjunto. Uma série de problemas de processamento de imagem pode ser resolvido através de uma inteligente formulação de um problema de otimização com três components: uma função objectivo que captura o critério a optimizar, um dicionário (conjunto de elementos)

adequado, e uma penalização ou restrição que impõe esparsidade na representação da imagem. Na realidade, a esparsidade de um vetor é definida pelo número de elementos diferentes de zero (norma ℓ_0), mas esta medida torna o problema NP-difícil. A sua relaxação convexa, a soma dos valores absolutos (norma ℓ_1), também mede esparsidade e é normalmente usada como alternativa. Quando o problema de optimização é convexo, a teoria de *compressive sensing* garante unicidade (ou pelo menos um erro limitado) do sinal esparso recuperado. Por sua vez, leva a garantia de desempenho para a solução dos problemas de processamento de imagem baseada em codificação esparsa. Sendo assim, acontece que muitos problemas de processamento de imagem que foram resolvidos com métodos adaptados podem ser resolvidos com a codificação esparsa com melhores resultados, especialmente quando eles usam um dicionário aprendido a partir de um conjunto de dados de imagens.

No entanto, o preço a pagar pelo desempenho acrescido é um grande custo computacional. Embora os métodos usados são agora mais rápido do que no início do desenvolvimento da codificação esparsa, o tempo de convergência ainda é uma grande desvantagem. Desenvolver métodos mais rápidos é ainda um tema de pesquisa importante, uma vez que irá conduzir um grande número de aplicações para o domínio prático.

A primeira contribuição desta tese é a comparação de vários solucionadores de codificação esparsa sob diferentes condições. Estas abrangem vários valores para a coerência do dicionário, esparsidade do verdadeiro sinal e para o parâmetro de regularização. A comparação confirma que o método denominado *Dual Alternating Direction Method of Multipliers* (DADMM) geralmente tem o melhor desempenho e, portanto, pode ser definido como um padrão e é uma base sólida para criar solucionadores mais rápidos. O mesmo algoritmo é comparado com solucionadores gulosos para problemas que usam a norma ℓ_0 para avaliar propriedades de velocidade e recuperação do sinal. Contrariamente à opinião comum de que algoritmos gulosos são mais rápidos

que solucionadores convexos, DADMM supera a maioria deles em termos de velocidade. Em relação às propriedades de recuperação, as garantias teóricas são melhores quando a esparsidade é medida pela norma ℓ_1 do que para qualquer solucionador guloso. Na prática, o oposto é visto, ou seja, solucionadores gananciosos têm melhores propriedades de recuperação do que solucionadores baseados na norma ℓ_1.

Este trabalho propõe uma nova versão do DADMM, no qual o parâmetro de penalização pode ser ajustado a cada iteração, sem nenhum custo extra. Como consequência, são possíveis simplificações adicionais que aceleram o algoritmo. Um estudo completo é realizado para avaliar qual é a melhor sequência de valores do parâmetro de penalização, sua relação com variáveis existentes no algoritmo e como ele melhora a convergência. Uma relação entre o parâmetro de penalização ideal e o erro de reconstrução é proposta, o que melhora o desempenho para o problema em questão. Vários exemplos são apresentados que mostram que o DADMM proposto atinge o mesmo nível de convergência em menos de metade do tempo em comparação com o DADMM original.

Adicionalmente, as propriedades únicas do DADMM são exploradas para acelerar *Virtual Probe* (VP), um método utilizado para reduzir o custo de teste de *wafers*, reduzindo o número de medições efectivamente necessárias. O mapa da *wafer* para cada item de teste constitui uma imagem, onde *compressive sensing* pode ser aplicado. As propriedades de DADMM tornam-se uma vantagem quando o dicionário é uma transformada rápida parcial, como é o caso da transformada discreta de cosseno usada em VP. Os resultados mostram que DADMM adaptado para VP pode atingir melhorias de velocidade até 38× em comparação com um método de ponto interior padrão.

O algoritmo poderia ser melhorado através da redução da complexidade das multiplicações matriz-vetor, com particular vantagem para dicionários altamente sobrecompletos. Isto poderia ser conseguido prevendo os elementos que estarão a zero na

próxima solução iterativa e evitando o cálculo dos produtos internos correspondentes, uma vez que o seu valor é irrelevante. Além disso, uma regra para a atualização do parâmetro de penalização melhor e mais ampla poderia ser obtida se a metodologia sugerida fosse utilizada numa gama maior de casos de teste. O resultado seria certamente um algoritmo de referência, livre de parâmetros, para problemas regularizados com a norma ℓ_1.

Acknowlegments

I would like to thank my advisors, Professor Miguel Correia and Professor Xin Li, for their guidance and for supporting my choices. The Carnegie Mellon | Portugal program was a unique opportunity for me to experience two different perspectives of high-level education and meet people with such diversified backgrounds. I would also like to express my gratitude to Professor Vitor Tavares for accompanying my research and providing helpful advices, and Professor Aswin Sankaranarayanan for his expertise in the area and suggestions for research topics. My Ph. D. committee members deserve my thanks for taking interest in reviewing my proposal and dissertation.

I would like to acknowledge the financial support from the Information and Communication Technologies Institute (ICTI), which is at the base of the Carnegie Mellon | Portugal program.

I am thankful to the friends I shared home with in Pittsburgh for their companionship. Indeed, they were my everyday company during the two school years I spent in the United States and made me have a great time.

I am grateful to my parents, for giving me the opportunity of the education that took me to where I am today. Their pride in my success is something that drives me to do better.

Lastly, I was blessed with a comprehensive wife, Helena, that encouraged me to follow a Ph.D. and gave me the support I needed throughout these years. She endured almost two years at a distance, so I could have the experience of a top University

education.

I end this section with a famous sentence of John F. Kennedy:

> "We do these things not because they are easy, but because they are hard."

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Signal Representation

Signal representation is an important field in engineering that has impacted our lifes, with applications such as compression, pattern recognition, classification, segmentation, signal enhancement, etc.. The major goal of signal representation is to transform a certain signal into the most *suitable* representation. The meaning of *suitable* depends on the application: for compression, it is a representation that leads to the least amount of data to be stored; for a classification problem, it is a representation that leads to a set of features that easily differentiates members of different classes; for object tracking, it is a representation that separates the target object from the background; or in pattern recognition, it may be a representation that decomposes the signal into relevant constituent parts. Therefore, transforming a raw signal is a fundamental step to many signal processing applications.

Any signal, represented as a vector, can be expressed by a linear combination of a set of patterns, as long as those patterns span the entire vector space. When the patterns are linearly independent, they form a basis. Principal Component Analysis (PCA) [32], for example, builds a basis from a set of observations in a way that

the patterns/components are ordered in terms of the energy they contribute to each observation. The first components make the largest contribution to the set of observations and the last ones are usually negligible. When the patterns are not linearly independent, their collection is more commonly known as a frame or dictionary. A dictionary is more general than a basis and it has the potential to express a signal with a smaller set of patterns. In addition, the vectors, or atoms, that compose the dictionary can be selected from any source, such as directly from a training set, from pre-defined bases, or learned to best represent the signal.

In the last decade, two promising frameworks are bringing to the signal processing field a new way to solve signal representation problems with dictionaries: sparse coding and dictionary learning. Sparse coding is related to finding the sparsest combination of coefficients given a certain dictionary and a set of observations. Dictionary Learning is concerned in finding the dictionary that produces the sparsest coefficients, given a dataset of observations, and uses sparse coding extensively. This dissertation focus on the sparse coding part. Although sparse coding is just a method of encoding a signal into a representation containing a small selection of atoms, it is formulated as an optimization problem. This, in turn, gives it enough flexibility to be adapted to a variety of signal processing problems. A number of them can be applied to images.

1.2 Sparse Coding

Sparse coding can be generally stated as the process of finding the set of sparsest coefficients that, linearly combined by the atoms of a dictionary, can provide a good approximation of the given observations. It is based on the assumption that the observation can indeed be explained by a small set of atoms. Sparse coding is often divided in the noiseless and noisy settings, for which there are different problem statements.

In the noiseless setting, the observations are entirely generated by the sparse coefficients and it is possible to have a zero approximation or reconstruction error. The problem is stated as

$$\text{minimize: (sparsity) subject to (reconstruction error)} = 0. \tag{1.1}$$

In some conditions, there is only one possible set of coefficients with the lowest sparsity. Therefore, sparse coding is able to recover that unique set of coefficients.

In the noisy case, noise is added to the original observations and the reconstruction is intended to be accurate up to the level of noise. In that case, the problem is always based on a compromise between the sparsity of the coefficients and the accuracy of the reconstruction, since both work in opposing directions. The compromise is often controlled by a parameter λ:

$$\text{minimize: (reconstruction error)} + \lambda\text{(sparsity)}. \tag{1.2}$$

In this case, the regularizing parameter plays an important role, since a value too small uses too much coefficients to explain the content of the observations below noise level and a value too high will prevent important coefficients from being selected.

The sparsity can be measured by two different norms, the ℓ_1-norm (the sum of absolute values of the coefficients) and the ℓ_0-norm (the number of non-zero coefficients). Although the ℓ_0-norm is the real measure of sparsity, the problems (1.1) and (1.2) become NP-hard, for which there is no optimal algorithm except brute force. Therefore, greedy algorithms are used that often lead to acceptable sub-optimal solutions. In this case, the problems are dubbed L0 problems in this dissertation. On the other hand, when the sparsity is defined by the ℓ_1-norm, the problems (1.1) and (1.2) are convex and a well defined theory named compressive sensing provides guarantees to the error bounds. For this case, they are named L1 problems.

1.2.1 Compressive Sensing

Compressive sensing theory, introduced by Candés and Tao [11] and Donoho [20], provides guarantees for the recovery of the original sparse signal even if its indirect observation has a lower dimensionality. It makes the assumption that the sparsity (number of zero coefficients) of the underlying signal under a given frame is below a certain level, i.e., that the observed signal was formed by a subset of the frame. Under the compressive sensing theory, the reconstruction is done by an ℓ_1-norm based minimization problem. One of the biggest drawbacks of this problem is the heavy computation necessary to solve it, since the ℓ_1-norm is non-differentiable and there is no closed-form solution. Hence, faster ways to solve this problem is still an important topic of research.

Consider that the transformation $\mathbf{y} = \mathbf{A}\mathbf{x}$ of the original signal $\mathbf{x}$ reduces its dimensionality. The dimensionality of $\mathbf{y} \in \mathbb{R}^m$ is smaller than the dimensionality of $\mathbf{x} \in \mathbb{R}^n$, hence the system is underdetermined: there is an infinite number of solutions $\mathbf{x}$ that can produce the same observations $\mathbf{y}$. Then, it seems that it is impossible to recover $\mathbf{x}$, knowing $\mathbf{y}$ and $\mathbf{A}$. Compressive sensing theory defines the conditions in which $\mathbf{x}$ can be recovered with a convex minimization problem. In particular, the theory relies on a single property of the mixing or sensing matrix $\mathbf{A}$: the *Restricted Isometry Constant (RIC)* δ_k is the smallest value for which the following equation is valid for any $k-$sparse (with only k non-zero elements) signal $\mathbf{x}_k$:

$$\sqrt{1-\delta_k}\left\|\mathbf{x}_k\right\|_2 \leq \left\|\mathbf{A}\mathbf{x}_k\right\|_2 \leq \sqrt{1+\delta_k}\left\|\mathbf{x}_k\right\|_2, \forall \mathbf{x}_k. \tag{1.3}$$

The matrix $\mathbf{A}$ is said to satisfy the *k-Restricted Isometry Property* (RIP) with RIC δ_k. The sharpest bound found so far for this constant that guarantees exact recovery of any $k-$sparse signal is $\delta_k \leq 0.3$ [8] or $\delta_{tk} \leq \sqrt{(t-1)/t}$ for $t > 4/3$ [9]. This constant decreases with the number of rows of $\mathbf{A}$ (which is the number of

observations), thus setting a lower limit to the number of samples that still allows exact recovery of the original signal. There are also error bounds for the cases where the observations are subjected to noise ($\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{n}$) and/or $\mathbf{x}$ is $k-$compressible (the remaining elements are not zero, but small compared to the highest k elements), making compressive sensing a highly practical theory.

The minimization problem that recovers the true solution is an ℓ_1-regularized regression:

$$\min_{\mathbf{x}} \|\mathbf{x}\|_1 \text{ s.t. } \mathbf{y} = \mathbf{A}\mathbf{x} \quad \text{, noiseless case} \tag{1.4}$$

$$\text{(L1)} \ \min_{\mathbf{x}} \frac{1}{2} \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 + \lambda \|\mathbf{x}\|_1 \quad \text{, noisy case.} \tag{1.5}$$

Other equivalent formulations of (1.5) are

$$\min_{\mathbf{x}} \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 \quad \text{s.t.} \ \ \|\mathbf{x}\|_1 \leq s \tag{1.6}$$

$$\min_{\mathbf{x}} \|\mathbf{x}\|_1 \quad \text{s.t.} \ \ \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 \leq \epsilon. \tag{1.7}$$

The imposition on the sparsity of $\mathbf{x}$ leads to a solution that captures the essential elements in $\mathbf{A}$ that represent $\mathbf{y}$. The regression in (1.6), known as LASSO [54], was originally used in statistics, where $\mathbf{y}$ was data, $\mathbf{A}$ was a full column rank matrix with $m > n$ representing a set of parameters (each column is a parameter), and $\mathbf{x}$ would provide the linear regression that explains the data with the smallest subset of parameters. With the advent of compressive sensing, this regression was also applied to cases where $\mathbf{A}$ was overcomplete ($m < n$) and not full-column rank, so long as it respected the bounds of the Restricted Isometry Property (RIP). Since then, its application in the image processing field exploded.

1.2.2 Applications

This section reports several applications of sparse coding found in the literature. Every application is based on the assumption that any observation vector $\mathbf{z}$ can be represented in a sparse (or at least compressible) set of coefficients $\mathbf{x}$ under some frame $\mathbf{A}$:

$$\mathbf{z} \approx \mathbf{A}\mathbf{x}. \tag{1.8}$$

1.2.2.1 Denoising

Often images are contaminated by noise, which can modeled by adding a random vector $\mathbf{n}$ to the original signal $\mathbf{z}$, producing the only observable signal $\mathbf{y}$:

$$\mathbf{y} = \mathbf{z} + \mathbf{n}. \tag{1.9}$$

The goal is to remove the noise from $\mathbf{y}$ to recover $\mathbf{z}$. Assuming that the dictionary represents the content of the image and not the noise, the optimization of equation (1.7) or equation (1.5) returns a sparse set of coefficients $\hat{\mathbf{x}}$ that will hopefully capture $\mathbf{z}$ and not $\mathbf{n}$. Depending on the value of the penalty parameter λ or error bound ϵ, a large amount of the final error $\|\mathbf{y} - \mathbf{A}\hat{\mathbf{x}}\|_2^2$ is the added noise, since it is not well captured with dictionary $\mathbf{A}$. The *clean* signal is then estimated with the operation $\hat{\mathbf{z}} = \mathbf{A}\hat{\mathbf{x}}$. The work in [23] focus on learning a proper dictionary, given the image dataset, and shows better results when that dictionary is used.

1.2.2.2 Inpainting

The goal of inpainting is to fill missing pixels (e.g. erased or painted over) based on the remaining pixels. Each pixel can be given a binary weight, 0 for the missing pixels and 1 otherwise. If $\mathbf{W}$ is a diagonal matrix containing those weights, the solution of:

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} E(\mathbf{x}) = \|\mathbf{W}(\mathbf{y} - \mathbf{A}\mathbf{x})\|_2^2 + \lambda \|\mathbf{x}\|_1 , \tag{1.10}$$

essentially finds the underlying signal under frame $\mathbf{A}$, since the painted pixels do not count for the reconstruction error. The *clean* signal is then estimated with the operation $\hat{\mathbf{z}} = \mathbf{A}\hat{\mathbf{x}}$. More advanced versions of inpainting based on sparse representation can be found in [48].

1.2.2.3 Background Subtraction

An important task in object tracking and video surveillance is background substraction, i.e., the removal of the background from current frames to point out the foreground objects. In [12], each image $\mathbf{x}$ is subsampled by a mixing matrix $\mathbf{\Phi}$ and the compressed samples $\mathbf{y} = \mathbf{\Phi}\mathbf{x}$ are transmitted by a constrained communication channel. Information about compressed background images $\mathbf{y}_{bi}$ is aggregated in a gaussian distribution: $\log \|\mathbf{y}_{bi} - \mathbf{y}_b\|_2^2 \sim \mathcal{N}(\mu_b, \sigma_b^2)$, where $\mathbf{y}_b = \frac{1}{N}\sum_{i=1}^{N} \mathbf{y}_{bi}$. Test images $\mathbf{y}_t$, differentiated from the background by the foreground objects, will follow the same distribution, but with different parameters $\log \|\mathbf{y}_t - \mathbf{y}_b\|_2^2 \sim \mathcal{N}(\mu_t, \sigma_t^2)$. As a result, the presence of an object is detected by a simple thresholding involving the metric $\log \|\mathbf{y}_t - \mathbf{y}_b\|_2^2$. In case of detection, since the difference image $\mathbf{x}_d = \mathbf{x}_t - \mathbf{x}_b$ is sparse in the spatial domain (assuming the foreground objects occupy only a fraction of the scene), the reconstruction of the foreground objects follows from solving the inverse problem:

$$\hat{\mathbf{x}}_d = \arg\min_{\mathbf{x}} \|\mathbf{y}_d - \mathbf{\Phi}\mathbf{x}\|_2^2 + \lambda \|\mathbf{x}\|_1 , \tag{1.11}$$

where $\mathbf{y}_d = \mathbf{y}_t - \mathbf{y}_b$. To improve robustness against background shifts, [74] builds the possible backgrounds as a collection of atoms in a dictionary $\mathbf{A}$. Hence, a background image $\mathbf{y}_b$ can be sparsely represented in $\mathbf{A}$ ($\mathbf{y}_b = \mathbf{A}\mathbf{x}_b$, where $\mathbf{x}_b$ is sparse). Given a

frame $\mathbf{y} = \mathbf{y}_f + \mathbf{y}_b$, where the foreground $\mathbf{y}_f = \mathbf{y} - \mathbf{y}_b = \mathbf{y} - \mathbf{A}\mathbf{x}_b$ is sparse in the spatial domain, the solution to

$$\hat{\mathbf{x}}_b = \arg\min_{\mathbf{x}} \left\| \begin{bmatrix} \mathbf{y} \\ 0 \end{bmatrix} - \begin{bmatrix} \mathbf{A} \\ \lambda\mathbf{I} \end{bmatrix} \mathbf{x} \right\|_1 \tag{1.12}$$

gives an estimation of the current background. The foreground comes from $\hat{\mathbf{y}}_f = \mathbf{y} - \mathbf{A}\hat{\mathbf{x}}_b$.

1.2.2.4 Digital Zoom

The goal of the digital zoom is to create high-resolution images out of low-resolution ones [66]. The method assumes that there is a dictionary $\mathbf{A}_{ld}$ from which low-resolution images are well reconstructed, as well as another dictionary $\mathbf{A}_{hd}$ with their high-resolution counterpart. First, a sparse representation for the observations, under dictionary $\mathbf{A}_{ld}$, is found by:

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} E(\mathbf{x}) = \|\mathbf{y} - \mathbf{A}_{ld}\mathbf{x}\|_2^2 + \lambda \|\mathbf{x}\|_1 . \tag{1.13}$$

Then, apply $\mathbf{A}_{hd}\hat{\mathbf{x}}$ to obtain the high-resolution image.

1.2.2.5 Single Pixel Camera

The single pixel camera (SPC) [21] is a type of spatial-multiplexing camera (SMC) that focuses the incident light in a single sensor element (or pixel). In general, spatial-multiplexing cameras are imaging architectures that employ a spatial light modulator (SLM), such as a digital micro-mirror device (DMD) or liquid crystal on silicon (LCOS), to optically compute a series of linear projections of the scene, which are equivalent to the rows of the dictionary $\mathbf{A}$. Since the absorption of light beyond the visual spectrum often requires sensors built from expensive materials, the use of a small sensoring area reduces the overall sensor cost significantly, in contrast to

full-frame sensors which intend to maximize the same area.

In particular, the SPC focuses the light from the scene onto a programmable DMD, which in turn directs it from only a subset of activated micro-mirrors onto the photodetector. The programmability of the DMD controls if the light incident in each micro-mirror is directed towards the photodetector or away from it. As a result, the voltage measured at the photodetector corresponds to an inner product of the image focused on the DMD and its activation pattern.

In the recovery stage, the original incident image is estimated from the collected compressive measurements, for example, by solving (1.5) or its variants.

1.2.2.6 Video Compressive Sensing for Spatial Multiplexing Cameras

Video compressive sensing is significantly harder than image compressive sensing with SMCs, since the captured scene changes in the time span of the SMC acquiring period. This, in turn, makes a poor estimate of an initial frame. Consecutive frames, obtained from motion flow vectors derived from the first frame, inherit its poor accuracy.

The framework in [47] obtains the initial estimate at a lower spatial resolution, which in turn requires less measurements from the SMC and therefore a shorter time window. In addition, the spatial downsampling also reduces the temporal resolution of the captured video. Both impact positively the accuracy of the estimate of the initial frame. A low-resolution video is reconstructed by using simple least-squares technique in each frame (the dimension of the low-resolution frame is equal to the dimension of the measurement vector). Secondly, this low-resolution video is used to obtain motion estimates between frames. Finally, a high-resolution video is recovered by enforcing a spatio-temporal gradient prior: the constraints induced by the compressive measurements (spatial) as well as the constraints due to motion estimates (temporal). The last step is a sparse coding problem, where sparsity is enforced in the spatio-temporal gradients using the total variation (TV) norm.

1.2.2.7 Face Recognition

In [63], sparse representation is used to robustly identify a query face $\mathbf{y}$ from a dictionary $\mathbf{A}$ formed by several faces. The dictionary is a conjunction of subdictionaries, each one containing faces of the same subject under different illumination conditions. Considering that the observed face may be occluded or corrupted, it can be represented as

$$\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{e} = \begin{bmatrix} \mathbf{A} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{e} \end{bmatrix} \tag{1.14}$$

where $\mathbf{x}$ is highly sparse, since only the atoms corresponding to the queried subject will be non-zero, and $\mathbf{e}$ will be non-zero in the locations of the corrupted pixels, since they cannot be well explained by the term $\mathbf{A}\mathbf{x}$. The variable $\mathbf{w} = \begin{bmatrix} \mathbf{x} \\ \mathbf{e} \end{bmatrix}$ can be found by solving

$$\min_{\mathbf{w}} \|\mathbf{w}\|_1 \qquad \mathbf{y} = \begin{bmatrix} \mathbf{A} & \mathbf{I} \end{bmatrix} \mathbf{w} \tag{1.15}$$

The structure of the dictionary is dubbed the cross-and-bouquet (CAB) model due to its appearance in a vector space diagram. In it, the matrix $\mathbf{I}$ marks a cross while the set of highly similar atoms $\mathbf{A}$ form a bouquet. Despite the high coherence of the dictionary, both $\mathbf{x}$ and $\mathbf{e}$ vectors can be exactly recovered, even when $\mathbf{e}$ is dense, as long as the bouquet is sufficiently tight and the dimensions of $\mathbf{x}$ and $\mathbf{y}$ are sufficiently high [62].

1.2.2.8 Wafer Test Cost Reduction

In the domain of integrated circuit testing, wafer probing is a stage where multiple performance items of each die in a wafer are measured. This process can be time-consuming and the test cost is proportional to the time it takes to complete. One way to reduce the test cost is through spatial variation modeling: assuming there is

spatial correlation in the wafer map, the goal is to measure a subset of the dies and infer the metrics of the remaining dies by some statistical algorithm. Virtual Probe [39] formulates the problem as a standard sparse coding problem:

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \frac{1}{2} \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 + \lambda \|\mathbf{x}\|_1 . \tag{1.16}$$

where $\mathbf{y}$ is the performance measurements of selected dies, $\mathbf{A}$ is a subsampled Discrete Cosine Transform (DCT) basis and $\mathbf{x}$ is the unknown DCT coefficient vector. The assumption that there is spatial correlation in the wafer map implies that the wafer map in the frequency domain $\mathbf{x}$ will be sparse. This is a classic application where taking measurements is an expensive step, and compressive sensing helps to reduce them. In this case, the time the solver takes to recover the DCT coefficients counts towards the test time and reducing it also reduces the test cost.

1.3 Contributions and Structure of the dissertation

This section summarizes the contributions and describes the structure of the dissertation:

- Chapter 2 introduces several methods developed in recent years to solve the sparse coding problem. The chapter is divided in solvers for ℓ_1-regularized problems and ℓ_0-norm based problems. The purpose of introducing greedy solvers is to later compare them with the best solver for ℓ_1-regularized problems and take conclusions from the benchmark. The same chapter further describes the Duality Theory (DT) and the Augmented Lagragian Method (ALM), which are both at the base of two particular solvers, named DALM and DADMM. DT is used to restate the formulation of the ℓ_1-regularized problem, while ALM

solves the new formulation in an efficient way. The chapter also briefly describes the idea behind dictionary learning, methods in the literature that solve it and its connection to sparse coding.

- Chapter 3 makes a contribution with the benchmark of several solvers under different conditions, sweeping several variables such as the coherence of the dictionary, sparsity of the true signal and the regularization parameter. The benchmark confirms that DADMM has the best performance and therefore is a solid base to create faster solvers. The chapter also benchmarks the same algorithm with greedy solvers for L0 problems to compare speed and recovery properties.

- Chapter 4 proposes a new version of DADMM, in which the penalty parameter can be adjusted at every iteration at no extra cost. A thorough study is undertaken to evaluate what is the best sequence of penalty parameter values and its relation to variables existent in the algorithm. A relation between optimal penalty parameter and reconstruction error is proposed, which improves performance for the addressed problem. Several examples are presented in which, for the same convergence level, the proposed DADMM finishes in less than half the time compared with the standard DADMM.

- Chapter 5 explores unique properties of DADMM to accelerate Virtual Probe (VP), a method used to reduce wafer test cost by reducing the number of effective measurements required. The properties of DADMM become an advantage when the dictionary is a partial fast-transform, as is the case of the DCT transform used in VP. Results show that DADMM tailored to VP can reach speed-ups up to 38× compared with a standard interior-point solver.

- Chapter 6 concludes the dissertation with a high-level summary of the work and discusses future improvements over the work presented here.

1.4 Conclusions

This chapter introduced the concept of sparse coding for signal representation. It covered the compressive sensing theory that provides recovery guarantees of the underlying signal when sparse coding is applied to overcomplete dictionaries. The variety of example applications that were presented show how broad and flexible the sparse coding framework is. However, compared with alternatives with closed-form solutions, it is often seen as computationally expensive and its future use depends on efficient implementations. The next chapter surveys previous works that aimed to speed up the ℓ_1-norm regularized minimization as well as greedy algorithms that approximately solve the ℓ_0-norm based minimization.

Chapter 2

Background/Related Work

Ever since the sparse coding framework and the compressive sensing theory emerged, its potential attracted many researchers to its field. The majority of the research can be divided in two paths: applications and methods.

Research on applications focuses on ways to use or adapt sparse coding to solve or improve solutions to problems in different fields. It turns out that sparse coding is a powerful and flexible tool to general problems of classification, prediction and others, many of them in the machine learning field.

However, the "achilles heel" of sparse coding is its computational cost. Therefore, the other major line of research focuses on reducing the cost of sparse coding methods. Since this is also the focus of this dissertation, the rest of this chapter describes the history of methods that solve the sparse coding problem and further recall the theory required for the next chapters.

2.1 Notation

Lowercase and uppercase boldface letters stand for column vectors and matrices, respectively. The i^{th} entry of a vector $\mathbf{x}$ is x_i, the i^{th} column of a matrix $\mathbf{A}$ is $\mathbf{a}_i$ and the i^{th} row of a matrix $\mathbf{A}$ is $\mathbf{a}^i$. For variables that are iterated, $\mathbf{x}_{(k)}$ or $\mathbf{x}^{(k)}$ represents

the k^{th} iteration of $\mathbf{x}$. $\mathbf{A}^T$ is the transpose of a matrix $\mathbf{A}$ and $\mathbf{A}^\dagger$ its pseudo-inverse. The ℓ_p norm of a vector $\mathbf{x}$ is denoted by $\|\mathbf{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p\right)^{1/p}$ and $|\mathbf{x}|$ or $\|\mathbf{x}\|_0$ denotes its cardinality or ℓ_0-pseudonorm (number of non-zero elements). The Frobenius norm of a matrix $\mathbf{A}$ is denoted as $\|\mathbf{A}\|_F = \sqrt{\sum_{i,j} a_{ij}^2}$. The support of a vector or matrix is the set of indices where the elements are non-zero and referred as $\text{supp}(\mathbf{x})$. $\max(\mathbf{x}, \mathbf{y})$ represents element-wise maximum between the first and second arguments and $\odot$ denotes the element-wise product. The function $\text{sign}(x)$ is 1 for $x > 0$, -1 for $x < 0$ and 0 for $x = 0$.

2.2 Sparse Coding Solvers

The vast amount of literature on this topic is impossible to be covered, therefore only the algorithms that standed out will be mentioned. They will be divided under two classes, the Basis Pursuit Denoising (BPDN)/LASSO algorithms, that solve (1.5)-(1.7) exactly, and greedy algorithms that solve approximately the same problems when the ℓ_1-norm is replaced by the ℓ_0-norm.

2.2.1 Basis Pursuit Denoising (BPDN)/LASSO Algorithms

This section describes the evolution of Basis Pursuit Denoising (BPDN)/LASSO algorithms, which solve the variations (1.5), (1.6) or (1.7).

In 1996, Tibshirani [55] presented the ℓ_1-regularization in the context of subset selection. There, he suggests the iterative ridge regression method

$$\begin{aligned} \mathbf{x}_{(t)} &= \arg\min_{\mathbf{x}} \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 + \lambda \left\|\mathbf{W}_{(t)}\mathbf{x}\right\|_2^2, \\ &= \left(\mathbf{A}^T\mathbf{A} + \lambda \mathbf{W}_{(t)}^T\mathbf{W}_{(t)}\right)^{-1} \mathbf{A}^T\mathbf{y}, \end{aligned} \tag{2.1}$$

where $\mathbf{W}_{(t)}$ is a diagonal matrix with each diagonal entry i equal to $1/\sqrt{\left|x_i^{(t-1)}\right|}$. As the algorithm converges ($\left\|\mathbf{x}_{(t)} - \mathbf{x}_{(t-1)}\right\|_2 \to 0$), the approximation $\|\mathbf{x}\|_1 = \sum_i |x_i| \approx \left\|\mathbf{W}_{(t)}\mathbf{x}\right\|_2^2 = \sum_i \left(\frac{x_i}{\sqrt{x_i^{(t-1)}}}\right)^2$ tends to become more and more accurate. However, he mentions this method is quite inefficient.

One year later, Gorodnitsky and Rao [28] developed FOCUSS (FOcal Underdetermined System Solver) summarized by:

Repeat

$$\begin{aligned}
\mathbf{W}_{(t)}^{-1} &= \text{diag}(\mathbf{x}_{(t-1)}^{l}) \\
\mathbf{z} &= \left(\left(\mathbf{A}\mathbf{W}_{(t)}^{-1}\right)^T \mathbf{A}\mathbf{W}_{(t)}^{-1} + \lambda\mathbf{I}\right)^{-1} \left(\mathbf{A}\mathbf{W}_{(t)}^{-1}\right)^T \mathbf{y} \\
\mathbf{x}_{(t)} &= \mathbf{W}_{(t)}^{-1}\mathbf{z} \\
t &= t+1
\end{aligned}$$

Until convergence,

where l is the element-wise power of $\mathbf{x}$. With the change of variable $\mathbf{z} = \mathbf{W}\mathbf{x}$, the method proposed by Tibshirani (2.1) becomes

$$\begin{aligned}
\mathbf{z}_{(t)} &= \arg\min_{\mathbf{z}} \left\|\mathbf{y} - \mathbf{A}\mathbf{W}_{(t)}^{-1}\mathbf{z}\right\|_2^2 + \lambda \|\mathbf{z}\|_2^2 \\
&= \mathbf{W}_{(t)}^{-1} \left(\left(\mathbf{A}\mathbf{W}_{(t)}^{-1}\right)^T \mathbf{A}\mathbf{W}_{(t)}^{-1} + \lambda\mathbf{I}\right)^{-1} \mathbf{A}\mathbf{W}_{(t)}^{-1}\mathbf{y} \qquad (2.2)
\end{aligned}$$

and $\mathbf{x}_{(t)} = \mathbf{W}_{(t)}^{-1}\mathbf{z}_{(t)}$, which is FOCUSS for $l = 0.5$. Therefore, FOCUSS generalizes iterated ridge regression for different powers of $\mathbf{x}$. In particular, for $l = 1$, it approximates the ℓ_0-norm. These early methods avoided the non-smoothness of the ℓ_1-norm by approximating it with a weighted ℓ_2-norm.

Shooting [25] was the first coordinate descent method to be proposed for ℓ_1-regularization. It minimizes the cost function w.r.t. each individual coefficient at a time, in a sequential fashion. Although not formulated as a proximal operator in the original work, the minimization involves the proximal operator of the ℓ_1-norm, which is the shrinkage function $x_i = \text{shrink}_\lambda(z_i) = \text{sign}(z_i)\max(|z_i| - \lambda, 0)$, where z_i is the result of the minimization of the reconstruction error w.r.t to x_i. Instead of optimizing the coefficients sequentially, other coordinate descent methods make an educated guess of the next best coefficient to update [49], such as the coefficient index corresponding to the atom with the strongest correlation with the residual. These methods are able to converge fast with dictionaries with low-correlated atoms, but struggled when atoms in the dictionary had some correlation between them.

Homotopy [43] is a method that solves (1.5) by finding critical points of the regularization parameter λ, in which a coefficient is added to or removed from an active set. At each of these critical points, the corresponding atom is added or removed from the active set, the Hessian is rank-one updated and a quadratic problem is solved with the exact $\Delta\lambda$ that takes the algorithm to the next critical point. The complete path of critical points is one that reduces the regularization parameter in a piecewise fashion. In this sense, they provide the complete regularization path, from $\lambda = \infty$ to a prescribed λ. Likewise, Least Angle Regression (LARS) [22] performs an identical procedure to solve (1.6), in this case sweeping from $s = 0$ to a prescribed s.

Active-set algorithms, which include Homotopy and LARS, keep track of the coefficients that are different from zero and progress by updating an active set, one atom at a time, towards the converging support. Feature-sign search [38] replaces the term $\left\|\mathbf{x}_{(k)}\right\|_1$ by the dot-product of the active coefficients $\mathbf{x}_{(k)}$ and their signs $\text{sign}\left(\mathbf{x}_{(k)}\right)$. The problem with the current active set is thus converted into a smooth quadratic program and its solution $\mathbf{x}_{(k+1)}$ is easily found by its closed-form expression. Then, the objective function is evaluated at all points in the line between $\mathbf{x}_{(k)}$ and $\mathbf{x}_{(k+1)}$ in

which the sign of a coefficient changes. The point with the lowest objective function value is selected and the step is repeated.

First-order algorithms that solve (1.5) are mostly based on the Forward-Backward splitting method [16], which can be seen as a generalization of the gradient descent method for functions with non-smooth terms. These are denoted as Iterative Shrinkage-Thresholding algorithms (ISTA), with its most basic form:

$$\mathbf{x}_{(k+1)} = \text{shrink}_{\lambda\gamma}\left(\mathbf{x}_{(k)} - \gamma\mathbf{A}^T\left(\mathbf{y} - \mathbf{A}\mathbf{x}_{(k)}\right)\right), \tag{2.3}$$

where γ is the gradient descent step size. FISTA [4] became quite popular, proving a complexity result $f(\mathbf{x}_{(k)}) - f(\mathbf{x}^\star) \propto 1/k^2$, where $\mathbf{x}^\star$ is the optimal solution, with almost no added computational complexity to each iteration compared to the most basic form of ISTA. It does so by using a combination of the last two iterations $\mathbf{x}_{(k)}$ and $\mathbf{x}_{(k-1)}$ to build $\mathbf{y}_{(k)}$ as the point to descend. SpaRSA [64] is an elaborate framework to minimize objective functions that include a non-smooth term. The framework includes extensions such as choosing the step size based on the Barzilai-Borwein (BB) method [3] or shrinking the regularization term from an upper bound down to the prescribed value throughout iterations (known as fixed-point continuation).

Augmented Lagrangian methods (ALM) solve constrained problems by iteratively progressing to the feasible space while optimizing the objective function. In its core, the Lagrange multiplier and a penalty term push each iteration closer to the feasible space. The Dual Augmented Lagrangian Method (DALM) [57] converts the primal problem (1.5) into its dual and solves it through the Augmented Lagrangian Method. The minimization of each AL step is done through the Newton method. The Alternating Direction Method of Multipliers (ADMM) is identical to ALM, with the exception that for composite functions $h(\mathbf{x}, \mathbf{z}) = f(\mathbf{x}) + g(\mathbf{z})$, each variable is updated separately, while keeping the other fixed. This divides a potentially difficult problem into two simpler subproblems. DADMM is similar to DALM, except that

it applies the ADMM to the dual problem [68]. By doing so, it simplifies the inner optimizations and provides closed-form solutions for them.

ProPPA [37] builds on top of projected proximal point algorithms, where the variable is alternatively updated by a proximal operator and projected to the feasible space of the problem. The evolution of the cost function is known to zig-zag in this class of methods because the projection often increases the cost, while the proximal operator reduces it. ProPPA uses a *buffer* feasible space whose progression towards the original feasible space is controlled by an update akin to the ALM step. As a consequence, the projection is smoothed out throughout iterations, reducing the zig-zag effect and improving convergence.

Although the motivation behind the improvement of these methods was the original compressive sensing formulation, they are usually general enough to solve other kinds of problems involving sparse coding. Hence, the range of applications they can reach is broader than the algorithms described in the next section. Furthermore, since they all solve exactly the same problem, they ensure the recovery guarantee presented by the compressive sensing theory.

Throughout this dissertation, this class of algorithms will be denoted as L1 algorithms.

2.2.2 Greedy Algorithms for ℓ_0-norm Regularization

A class of algorithms commonly used for compressive sensing is the one based on the minimization of the reconstruction error by using a limited number of coefficients

$$\begin{aligned} \hat{\mathbf{x}} &= \arg\min_{\mathbf{x}} \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 \\ &\quad S.T.\ \|\mathbf{x}\|_0 \leq s, \end{aligned} \tag{2.4}$$

or minimization of the ℓ_0-norm with the reconstruction error bounded from above

$$\begin{aligned} \hat{\mathbf{x}} &= \arg\min_{\mathbf{x}} \|\mathbf{x}\|_0 \\ & S.T.\ \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 \leq \varepsilon. \end{aligned} \tag{2.5}$$

Since both problems are NP-hard, greedy algorithms are employed to obtain a solution. Although it is not guaranteed they will reach the global optimum, they often provide good results. They are normally used when fast recovery is required. However, they are not as robust as ℓ_1-norm regularization. For example, they often have the target number of non-zeros as a parameter: an underestimation of the true value would truncate meaningful coefficients and overestimation results in loss of accuracy and robustness [19]. Since they work with an active set of atoms, the inverse problem is often the minimization of the reconstruction error with only the atoms belonging to the active set.

Orthogonal Matching Pursuit [44, 7] is, by far, the most widely used algorithm of its class. At each iteration, it selects the atom in the dictionary with the highest correlation with the residual and adds it to an active set. The inverse problem is then calculated with only the atoms in the active set and the residual is updated. The cycle is repeated until the error is sufficiently small or until the defined number of atoms is reached. Subspace Pursuit (SP) [17] and CoSAMP [41] are nearly identical algorithms and work as follows for the recovery of a $k-$sparse signal: they select the $2k$ atoms (for CoSAMP) or k atoms (for SP) in the dictionary with the highest correlation with the residual and add them to an active set. The inverse problem is then calculated with only the atoms in the active set. Afterwards, only the k coefficients with largest absolute values are kept and the residual is updated. The cycle is repeated until a halting criterion is met. SP recovers the signal exactly when

$\delta_{3k} < 0.165$ and CoSAMP when $\delta_{4k} < 0.1$. SAMP [19] differentiates from others by working without the knowledge of the sparsity of the original signal. It starts with a sparsity estimate k that needs to be smaller than the correct sparsity and proceeds in the same form as SP. If, at some iteration, the norm of the residual increases compared to the previous iteration, k is increased and the cycle is continued. The algorithm stops when a halting criterion is reached. SAMP recovers exactly the original signal if $\delta_{3k} < 0.06$. GraDeS [26] repeatedly takes gradient steps towards a reduction in the reconstruction error and keeps only the k largest (in absolute value) coefficients to restore the sparsity of the current solution. The step size for convergence depends on the Restricted Isometry Property (RIP) of the dictionary. This method recovers the true solution when $\delta_{2k} < 1/3$.

Since, in essence, these algorithms are solving only approximately the same problem, they have different recovery guarantees. Although they are generally worse than the compressive sensing guarantees, it does not mean they have worse recovery performance. The RIP upper bound only states the larger RIC *proven so far* for which they are guaranteed to recover the exact signal.

Throughout this dissertation, this class of algorithms will be denoted as L0 algorithms.

2.3 Duality Theory

As will become clear in this chapter, and confirmed experimentally in chapter 3, duality theory plays an important role in simplifying the problem statement for sparse coding applications. The *duality principle* states that a constrained problem can be viewed under two perspectives, the original *primal* version and a *dual* version, which is obtained by forming the Lagrangian on the original problem. Let us consider the general constrained case as the primal problem:

$$\begin{aligned} \min_{\mathbf{x}} \quad & f(\mathbf{x}) \\ S.T. \quad & h_i(\mathbf{x}) = 0,\ i \in \{1, ..., n_h\} \\ & g_i(\mathbf{x}) \leq 0,\ i \in \{1, ..., n_g\}, \end{aligned} \tag{2.6}$$

where $h_i(\mathbf{x})$ and $g_i(\mathbf{x})$ are scalar functions. For simplicity of notation, we define the feasible set as $\mathcal{X}$. The Lagrangian function is built by converting the constraints into penalties weighted by Lagrange multipliers μ and ν:

$$L(x, \mu, \nu) \quad = \quad f(\mathbf{x}) + \sum_{i=1}^{n_h} \mu_i h_i(\mathbf{x}) + \sum_{i=1}^{n_g} \nu_i g_i(\mathbf{x}). \tag{2.7}$$

It is important to know that, when solving for the Lagrangian function, the multipliers ν are confined to be non-negative because they are associated with *lower-than* inequalities. Since, at a feasible solution of the primal problem, we have $h_i(\mathbf{x}) = 0,\ i \in \{1, ..., n_h\}$, $g_i(\mathbf{x}) \leq 0,\ i \in \{1, ..., n_g\}$ and $\nu_i \geq 0,\ i \in \{1, ..., n_g\}$, we can confidently state that

$$L(\mathbf{x}, \mu, \nu) = f(\mathbf{x}) + \underbrace{\sum_{i=1}^{n_h} \mu_i h_i(\mathbf{x})}_{=0} + \underbrace{\sum_{i=1}^{n_g} \nu_i g_i(\mathbf{x})}_{\leq 0} \leq f(\mathbf{x}),\ \mathbf{x} \in \mathcal{X}. \tag{2.8}$$

Now we introduce the dual function, which is the minimization of the Lagrange over the primal variable:

$$q(\mu, \nu) = \inf_{\mathbf{x}} L(\mathbf{x}, \mu, \nu) \tag{2.9}$$

By joinining (2.8) and (2.9), we obtain

$$q(\mu, \nu) \leq L(\mathbf{x}, \mu, \nu) \leq f(\mathbf{x}),\ \nu \geq \mathbf{0},\ \mathbf{x} \in \mathcal{X} \tag{2.10}$$

Of course, the optimal solution $\mathbf{x}^\star$ of the primal problem is included in the feasible set. Therefore, we can also confirm that:

$$q(\mu, \nu) \leq f(\mathbf{x}^\star),\ \nu \geq \mathbf{0} \tag{2.11}$$

The dual problem is thus defined as the maximization of the dual function, with the constraint $\nu \geq \mathbf{0}$:

$$\begin{aligned} \max_{\mu,\nu} \quad & q(\mu, \nu) \\ & \nu \geq \mathbf{0} \end{aligned} \tag{2.12}$$

It is easy to see that if the primal problem was a maximization, the inequalities and optimizations respective to the dual problem would be inverted.

2.3.1 Duality Gap

Note that generally it is not guaranteed that the optima of the dual $q(\mu^\star, \nu^\star)$ and primal $f(\mathbf{x}^\star)$ problems will have the same value. The difference between them is called the duality gap and, in that sense, the dual problem can be seen as providing the best lower bound for the primal problem:

$$f(\mathbf{x}^\star) - q(\mu^\star, \nu^\star) = \text{duality gap} \tag{2.13}$$

When certain *constraint qualifications* are met, the duality gap is zero and is said that the problem has *strong duality*. In particular, one of these qualifications, *Slater's condition*, when applied to problems with only equality constraints, is simply that the primal problem is feasible [6]. As it will be seen in section 2.4, it is very simple to

add an equality constraint to unconstrained problems. Therefore, the unconstrained problem addressed by compressive sensing has strong duality as long as there is a solution for it.

2.4 Dual of the ℓ_1-regularized Least Mean Squares

Let us apply the duality theory to the compressive sensing problem:

$$\min_{\mathbf{x}} \frac{1}{2} \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 + \lambda \|\mathbf{x}\|_1 . \tag{2.14}$$

First, we add the auxiliary variable $\mathbf{r}$ with the equality constraint $\mathbf{r} = \mathbf{y} - \mathbf{A}\mathbf{x}$:

$$\begin{aligned} \min_{\mathbf{x},\mathbf{r}} \quad & \frac{1}{2} \|\mathbf{r}\|_2^2 + \lambda \|\mathbf{x}\|_1 \\ \text{s.t.} \quad & \mathbf{r} = \mathbf{y} - \mathbf{A}\mathbf{x} \end{aligned} \tag{2.15}$$

Next, we build the dual function:

$$\inf_{\mathbf{x},\mathbf{r}} L(\mathbf{x}, \mathbf{r}, \mu) = \inf_{\mathbf{x},\mathbf{r}} \left\{ \frac{1}{2} \|\mathbf{r}\|_2^2 + \lambda \|\mathbf{x}\|_1 + \mu^T (\mathbf{r} - (\mathbf{y} - \mathbf{A}\mathbf{x})) \right\} \tag{2.16}$$

and separate minimization problems:

$$\begin{aligned} & \inf_{\mathbf{r}} \left\{ \frac{1}{2} \|\mathbf{r}\|_2^2 + \mu^T \mathbf{r} \right\} + \inf_{\mathbf{x}} \left\{ \lambda \|\mathbf{x}\|_1 + \mu^T \mathbf{A}\mathbf{x} \right\} - \mu^T \mathbf{y} \\ = \; & - \sup_{\mathbf{r}} \left\{ -\frac{1}{2} \|\mathbf{r}\|_2^2 + (-\mu)^T \mathbf{r} \right\} - \sup_{\mathbf{x}} \left\{ -\lambda \|\mathbf{x}\|_1 + \left(-\mathbf{A}^T \mu\right)^T \mathbf{x} \right\} \\ & -\mu^T \mathbf{y} \end{aligned} \tag{2.17}$$

The first term of equation (2.17) can be recognized as the Legrendre-Fenchel

transform of $\frac{1}{2}\|\mathbf{r}\|_2^2$, while the second term as the Legendre-Fenchel transform of $\lambda\|\mathbf{x}\|_1$. Their relations are:

$$\begin{aligned} \sup_{\mathbf{r}}\left\{\mathbf{s}^T\mathbf{r}-\frac{1}{2}\|\mathbf{r}\|_2^2\right\} &= \frac{1}{2}\|\mathbf{s}\|_2^2 \\ \sup_{\mathbf{x}}\left\{\mathbf{s}^T\mathbf{x}-\lambda\|\mathbf{x}\|_1\right\} &= \begin{cases}\infty & \|\mathbf{s}\|_\infty > \lambda \\ 0 & \|\mathbf{s}\|_\infty \leq \lambda\end{cases} \end{aligned} \tag{2.18}$$

Compiling equations (2.17) and (2.18), and maximing the dual function, we obtain:

$$\begin{aligned} \max_{\mu} \quad & -\frac{1}{2}\|-\mu\|_2^2-\mu^T\mathbf{y} \\ \text{s.t.} \quad & \left\|-\mathbf{A}^T\mu\right\|_\infty \leq \lambda. \end{aligned} \tag{2.19}$$

With a final touch, $\alpha=-\mu$, equation (2.19) becomes

$$\begin{aligned} \max_{\alpha} \quad & -\frac{1}{2}\|\alpha\|_2^2+\alpha^T\mathbf{y} \\ \text{s.t.} \quad & \left\|\mathbf{A}^T\alpha\right\|_\infty \leq \lambda. \end{aligned} \tag{2.20}$$

In summary, equation (2.20) is the dual version of equation (2.14).

2.5 Dual Augmented Lagrangian Method (DALM)

The Dual Augmented Lagrangian Method applies the Augmented Lagrangian Method (ALM) to equation (2.20). Before moving on, ALM, which is also known as Method of Multipliers, will be briefly described.

2.5.1 Augmented Lagrangian Method (ALM)

As the name implies, the method builds an augmented version of the Lagrangian by adding penalty terms to the original Lagrangian. Consider the general problem:

$$\begin{aligned} \min_{\mathbf{x}} \quad & f(\mathbf{x}) \\ S.T. \quad & h_i(\mathbf{x}) = 0,\ i \in \{1, ..., n_h\}. \end{aligned} \tag{2.21}$$

The Augmented Lagrangian method builds the Augmented Lagrange function

$$L_\eta(\mathbf{x}, \mu) = f(\mathbf{x}) + \sum_{i=1}^{n_h} \mu_i h_i(\mathbf{x}) + \eta \sum_{i=1}^{n_h} h_i(\mathbf{x})^2, \tag{2.22}$$

which has an additional term controlled by the penalty parameter η. This term is always non-negative, which increases the cost function whenever the constraints are away from zero. The algorithm alternates between optimizing the Augmented Lagrangian $L_\eta(\mathbf{x}, \mu_{(k)})$ w.r.t. $\mathbf{x}$ and stepping the multipliers with the following rule:

$$\mu_{(k+1)} = \mu_{(k)} + \eta_{(k)} \mathbf{h}(\mathbf{x}_{(k+1)}), \tag{2.23}$$

where the subscript $(.)_{(k)}$ denotes the variable at iteration k. The update is a form of gradient ascent with the step size equal to the penalty parameter. It can be shown that if the sequence $\{\mathbf{x}_{(k)}\}$ tends to the optimum $\mathbf{x}^\star$, then the updates in equation (2.23) converge to the corresponding Lagrange multiplier $\mu^\star$ [6, Prop. 4.2.2]. The complete algorithm is described in Algorithm 2.1, where δ is a term that dictates the geometric progression of the parameter η. Usually, the penalty parameter is increased heuristically, not too slowly to speed up convergence and not too fast to maintain stability.

Algorithm 2.1 Augmented Lagrangian Method

Initialize		$k = 0, \mu_{(0)}$ and $\eta_{(0)}, \delta$
Repeat		
$\mathbf{x}_{(k+1)}$	$=$	$\arg\min_{\mathbf{x}} L_{\eta_{(k)}}(\mathbf{x}, \mu_{(k)})$
$\mu_{(k+1)}$	$=$	$\mu_{(k)} + \eta_{(k)}\mathbf{h}(\mathbf{x}_{(k+1)})$
$\eta_{(k+1)}$	$=$	$\delta\eta_{(k)}$
k	$\leftarrow$	$k + 1$
Until		convergence

2.5.2 ALM Applied to the Dual Problem

Starting from equation (2.20) and adding an auxiliary variable ν such that $\nu = \mathbf{A}^T\alpha$, the inequality constraint is replaced by $\|\nu\|_\infty \leq \lambda$ and the problem can be solved by first building the Augmented Lagrangian:

$$L_\eta(\alpha, \nu, \mathbf{x}) = -\frac{1}{2}\|\alpha\|_2^2 + \alpha^T\mathbf{y} - \mathbf{x}^T\left(\mathbf{A}^T\alpha - \nu\right) - \frac{\eta}{2}\left\|\mathbf{A}^T\alpha - \nu\right\|_2^2 - \delta_\lambda(\nu) \quad (2.24)$$

where η is the penalty parameter, $\mathbf{x}$ is the Lagrange Multiplier and $\delta_\lambda(\nu)$ is $+\infty$ if any element of ν is greater than λ and 0 otherwise. Recall that since this is a maximization problem, the signs of the terms in the Lagrange function are inverted. The solution is found by iteratively maximizing the objective function w.r.t. the joint variables (α, ν) and updating the multiplier $\mathbf{x}$ by

$$\mathbf{x}_{(k+1)} = \mathbf{x}_{(k)} + \eta_{(k)}\left(\mathbf{A}^T\alpha_{(k+1)} - \nu_{(k+1)}\right), \quad (2.25)$$

as in equation (2.23).

To obtain the joint maximization, we consider $\nu(\alpha)$ and $\alpha(\nu)$ as functions of each other. The maximum is found when the gradients w.r.t. both variables are zero:

$$\begin{aligned} \nabla_\nu L_\eta(\alpha, \nu, \mathbf{x}) &= 0 \\ \nabla_\alpha L_\eta(\alpha, \nu, \mathbf{x}) &= 0 \end{aligned} \tag{2.26}$$

For ν:

$$\begin{aligned} \nabla_\nu L_\eta(\alpha, \nu, \mathbf{x}) &= \mathbf{x} + \eta\left(\mathbf{A}^T\alpha - \nu\right) - \delta_\lambda(\nu) = 0 \\ \nu(\alpha) &= P_\lambda^\infty\left(\frac{\mathbf{x}}{\eta} + \mathbf{A}^T\alpha\right), \end{aligned} \tag{2.27}$$

where $P_\lambda^\infty(x)$ is an element-wise projection to the range $[-\lambda, \lambda]$. This is expected, since no element of ν can be greater than λ in absolute value.

For α:

$$\begin{aligned} & \nabla_\alpha L_\eta(\alpha, \nu, \mathbf{x}) \\ = \; & -\alpha + \mathbf{y} - \mathbf{A}\mathbf{x} - \eta\mathbf{A}\left(\mathbf{A}^T\alpha - \nu(\alpha)\right) \\ = \; & -\alpha + \mathbf{y} - \mathbf{A}\left(\mathbf{x} + \eta\mathbf{A}^T\alpha - \eta P_\lambda^\infty\left(\frac{\mathbf{x}}{\eta} + \mathbf{A}^T\alpha\right)\right) \\ = \; & -\alpha + \mathbf{y} - \mathbf{A}\left(\mathbf{x} + \eta\mathbf{A}^T\alpha - P_{\lambda\eta}^\infty\left(\mathbf{x} + \eta\mathbf{A}^T\alpha\right)\right) \\ = \; & -\alpha + \mathbf{y} - \mathbf{A}\cdot \mathit{shrink}_{\lambda\eta}\left(\mathbf{x} + \eta\mathbf{A}^T\alpha\right), \end{aligned} \tag{2.28}$$

where $\nu(\alpha)$ was replaced by (2.27) and $\text{shrink}_\lambda(x) = \max(|x| - \lambda, 0)\cdot\text{sign}(x)$ is an element-wise operator. The last step of (2.28) is derived from the Moreau's decomposition [46, Theorem 31.5] $\text{prox}_f(\mathbf{z}) = \mathbf{z} - \text{prox}_{f^\star}(\mathbf{z})$, where $f(\mathbf{z}) = \lambda\eta\left\|\mathbf{z}\right\|_1$ and $f^\star(\mathbf{z}) = \left\|\mathbf{z}\right\|_{\lambda\eta}^\infty$ are conjugate norms, $\text{prox}_f(\mathbf{z}) = \arg\min_{\mathbf{x}}\left\{f(\mathbf{x}) + \frac{1}{2}\left\|\mathbf{x} - \mathbf{z}\right\|_2^2\right\}$ is the proximal operator of function f and $\mathbf{z} = \mathbf{x} + \eta\mathbf{A}^T\alpha$. The proximal operators of the above functions are

$$\begin{aligned} \text{prox}_f(\mathbf{z}) &= \text{shrink}_\lambda(\mathbf{z}),\ f(\mathbf{z}) = \lambda \|\mathbf{z}\|_1 \\ \text{prox}_{f^\star}(\mathbf{z}) &= P_\lambda^\infty(\mathbf{z}), \qquad f^\star(\mathbf{z}) = \|\mathbf{z}\|_\lambda^\infty . \end{aligned} \tag{2.29}$$

Since there is no closed-form solution to the maximization, DALM [56] updates α through Newton steps with Hessian:

$$\nabla_\alpha^2 L_\eta(\alpha, \nu, \mathbf{x}) = -\left(\mathbf{I} + \eta \mathbf{A}_{\mathbf{\Omega}} {\mathbf{A}_{\mathbf{\Omega}}}^T\right), \tag{2.30}$$

where $\Omega = \left\{ i : x_i^{(k)} \neq 0 \right\}$ is the set of indices where the multipliers are non-zero.

The maximization is only done approximately, since conditions defined in [57] state that the gradient (2.28) only needs to be below a certain upper bound to guarantee that the Lagrangian is in fact increased.

The dual variable of the dual problem (which is the primal variable) is updated according to equation (2.25):

$$\begin{aligned} \mathbf{x}_{(k+1)} &= \mathbf{x}_{(k)} + \eta\left(\mathbf{A}^T\alpha_{(k+1)} - \nu(\alpha_{(k+1)})\right) \\ &= \mathbf{x}_{(k)} + \eta\left(\mathbf{A}^T\alpha_{(k+1)} - P_\lambda^\infty\left(\frac{\mathbf{x}_{(k)}}{\eta} + \mathbf{A}^T\alpha_{(k+1)}\right)\right) \\ &= \mathbf{x}_{(k)} + \eta\mathbf{A}^T\alpha_{(k+1)} - P_{\lambda\eta}^\infty\left(\mathbf{x}_{(k)} + \eta\mathbf{A}^T\alpha_{(k+1)}\right) \\ &= \text{shrink}_{\lambda\eta}\left(\mathbf{x}_{(k)} + \eta\mathbf{A}^T\alpha_{(k+1)}\right) \end{aligned} \tag{2.31}$$

where the last line is derived from the Moreau's decomposition [46, Theorem 31.5] as in (2.28), with $\text{prox}_f(\mathbf{z}) = \mathbf{z} - \text{prox}_{f^\star}(\mathbf{z})$, , with $f(\mathbf{z}) = \lambda\eta\|\mathbf{z}\|_1$, $f^\star(\mathbf{z}) = \|\mathbf{z}\|_{\lambda\eta}^\infty$, and $\mathbf{z} = \mathbf{x}_{(k)} + \eta\mathbf{A}^T\alpha_{(k+1)}$.

As a final note, given equation (2.27), it is not difficult to manipulate equation (2.24) to obtain the simpler form:

Algorithm 2.2 Dual Augmented Lagrangian Method.

Initialize $k = 0, \alpha_{(0)}, \mathbf{x}_{(0)}$ and $\eta_{(0)}, \delta$

Repeat

$$\alpha_{(k+1)} \simeq \arg\max_{\alpha} \left\{ -\frac{1}{2}\|\alpha\|_2^2 + \alpha^T \mathbf{y} - \frac{1}{2\eta_{(k)}} \left\| \text{shrink}_{\lambda\eta_{(k)}}(\mathbf{x}_{(k)} + \eta_{(k)}\mathbf{A}^T\alpha) \right\|_2^2 \right\}^{\dagger}$$

$$\mathbf{x}_{(k+1)} = \text{shrink}_{\lambda\eta}\left(\mathbf{x}_{(k)} + \eta_{(k)}\mathbf{A}^T\alpha_{(k+1)}\right)$$

$$\eta_{(k+1)} = \delta\eta_{(k)}$$

$$k \leftarrow k + 1$$

Until convergence

†The approximate maximization is obtained by Newton steps with gradient and Hessian defined in (2.28) and (2.30) respectively.

$$L(\alpha, \mathbf{x}) = -\frac{1}{2}\|\alpha\|_2^2 + \alpha^T \mathbf{y} - \frac{1}{2\eta}\left\|\text{shrink}_{\lambda\eta}(\mathbf{x} + \eta\mathbf{A}^T\alpha)\right\|_2^2, \tag{2.32}$$

which does not depend on the variable ν. The complete algorithm is described in Algorithm 2.2.

This approach requires very few iterations, however, the Hessian and its inverse needs to be recomputed for every outer iteration during the α update. Fortunately, the rank of the Hessian is equal to the number of active multipliers at a given iteration, which tends to be small.

2.6 Dual Alternating Direction Method of Multipliers (DADMM)

Although the original DALM converges in few steps, each step is computationally expensive, since the inverse needs to be recomputed with a new set of active columns. Furthermore, since there is no closed-form solution to the α update, DALM takes Newton steps as inner iterations. An alternative approach to solving ALM that has gain recent attention is the Alternating Direction Method of Multipliers (ADMM).

Algorithm 2.3 Alternating Direction Method of Multipliers

$$
\begin{array}{rcl}
\text{Initialize} & & k = 0, \mu_{(0)} \text{ and } \eta \\
\text{Repeat} & & \\
\mathbf{y}_{(k+1)} & = & \arg\min_{\mathbf{y}} L_\eta(\mathbf{x}_{(k)}, \mathbf{y}, \mu_{(k)}) \\
\mathbf{x}_{(k+1)} & = & \arg\min_{\mathbf{x}} L_\eta(\mathbf{x}, \mathbf{y}_{(k+1)}, \mu_{(k)}) \\
\mu_{(k+1)} & = & \mu_{(k)} + \eta \mathbf{h}(\mathbf{x}_{(k+1)}, \mathbf{y}_{(k+1)}) \\
k & \leftarrow & k + 1 \\
\text{Until} & & \text{convergence}
\end{array}
$$

2.6.1 Alternating Direction Method of Multipliers

This method is very similar to ALM, except that a potentially complex optimization is split into two variables and each variable is optimized separately [68]. The method is applied to problems that are in or can be converted to the form:

$$
\begin{array}{ll}
\min\limits_{\mathbf{x},\mathbf{y}} & f(\mathbf{x}) + g(\mathbf{y}) \\
S.T. & h_i(\mathbf{x}, \mathbf{y}) = 0,\ i \in \{1, ..., n_h\},
\end{array}
\tag{2.33}
$$

i.e., where the cost function is the sum of two separable functions. The Augmented Lagrangian is built as in ALM:

$$
L_\eta(\mathbf{x}, \mathbf{y}, \mu) = f(\mathbf{x}) + g(\mathbf{y}) + \sum_{i=1}^{n_h} \mu_i h_i(\mathbf{x}, \mathbf{y}) + \eta \sum_{i=1}^{n_h} h_i(\mathbf{x}, \mathbf{y})^2, \tag{2.34}
$$

and each variable is updated separately. The complete algorithm is described in Algorithm 2.3.

2.6.2 ADMM Applied to the Dual Problem

Note that if we still use the auxiliary variable ν such that $\nu = \mathbf{A}^T\alpha$, the problem has the form (2.33), except that it is a maximization, with

$$\begin{aligned} f(\alpha) &= -\frac{1}{2}\|\alpha\|_2^2 + \alpha^T\mathbf{y} \\ g(\nu) &= \begin{cases} 0 & \|\nu\|_\infty \leq \lambda \\ \infty & \|\nu\|_\infty > \lambda \end{cases} \\ \mathbf{h}(\alpha,\nu) &= \mathbf{A}^T\alpha - \nu. \end{aligned} \tag{2.35}$$

The optimization over α becomes:

$$\nabla_\alpha L_\eta(\alpha, \nu_{(k)}, \mathbf{x}_{(k)}) = -\alpha + \mathbf{y} - \mathbf{A}\mathbf{x}_{(k)} - \eta\mathbf{A}\left(\mathbf{A}^T\alpha - \nu_{(k)}\right) = 0, \tag{2.36}$$

which leads to the closed-form solution

$$\alpha = \left(\mathbf{I} + \eta\mathbf{A}\mathbf{A}^T\right)^{-1}\left(\mathbf{y} - \mathbf{A}\left(\mathbf{x}_{(k)} - \eta\nu_{(k)}\right)\right). \tag{2.37}$$

The variable ν and the multiplier $\mathbf{x}$ updates remain as in (2.27) and as in (2.31), respectively. The complete algorithm for DADMM is described in Algorithm 2.4.

If η is kept fixed and the inverse of the Hessian is precomputed, each iteration of DADMM amounts to three matrix-vector multiplications, although it runs more of them compared with DALM.

Using the correct penalty parameter η can improve the convergence rate. Recalling the α update (2.37), we can change η at the cost of recomputing the inverse of the Hessian, or keep it fixed throughout the algorithm and precompute the Hessian

Algorithm 2.4 Dual Alternating Direction Method of Multipliers.

$$
\begin{array}{rcl}
\text{Initialize} & & k = 0, \nu_{(0)}, \mathbf{x}_{(0)} \text{ and } \eta \\
\text{Repeat} & & \\
\alpha_{(k+1)} & = & \left(\mathbf{I} + \eta \mathbf{A}\mathbf{A}^T\right)^{-1} \left(\mathbf{y} - \mathbf{A}\left(\mathbf{x}_{(k)} - \eta \nu_{(k)}\right)\right) \\
\nu_{(k+1)} & = & P_{\lambda}^{\infty}\left(\dfrac{\mathbf{x}_{(k)}}{\eta} + \mathbf{A}^T \alpha_{(k+1)}\right), \\
\mathbf{x}_{(k+1)} & = & \text{shrink}_{\lambda\eta}\left(\mathbf{x}_{(k)} + \eta \mathbf{A}^T \alpha_{(k+1)}\right) \\
k & \leftarrow & k + 1 \\
\text{Until} & & \text{convergence}
\end{array}
$$

at the beginning. Chapter 4 suggests an alternative that not only allows to change the penalty parameter at no cost, but also makes each iteration faster. A precomputation based on eigendecomposition is required. Thankfully, in many applications, the dictionary is reused and this decomposition needs to be *done only once*.

2.7 Dictionary Learning

For the sake of completion, it is worth mentioning a concept that propelled sparse coding usefulness. Most of the benefit of sparse coding comes from the large number of atoms that the dictionary has available for selection, which increases the probability of representing an observation with a small number of atoms.

Originally, the dictionary was a composition of bases, such as Discrete Cosine Transform (DCT), wavelets, etc., or their overcomplete or subsampled versions. An analysis would be performed to ascertain which type of basis or conjunction of bases would provide the best results. On the other hand, dictionary learning attempts to infer the patterns that are common to a class of signals, under the assumption that each signal used to train the dictionary is formed by a small subset of these patterns. This approach has the potential to provide a uniform method to learn the best frame to a certain type of signal, solely based on training samples. It is specially suited to

new types of data, for which an adequate representation is not yet found. In fact, the genesis of dictionary learning comes from a paper that aimed to study the underlying patterns in natural images and point out the similarity to the patterns to which cells in the primary visual cortex respond best [42]. Although there is not yet a solid theory for dictionary learning, empirical results of existing algorithms show that is possible to recover the true dictionary if the coefficients used to produce the training samples are sparse enough [1, 40, 50, 18, 75]. Since the learned dictionary is, in principle, better matched to the observations than pre-designed dictionaries, the sparse coding model reaches higher levels of sparsity for the same reconstruction error, or better reconstruction for the same level of sparsity.

Much of the current work takes the approach of alternating the minimization of the reconstruction error between the dictionary and the coefficients. Since the minimization w.r.t to the coefficients is usually the ℓ_0- or ℓ_1-regularized problem, most of the research focuses on the update of the dictionary. The update of the dictionary is concerned in reducing the reconstruction error while keeping the support of the coefficient matrix $\mathbf{X}$ unchanged:

$$\left(\mathbf{A}^{(k+1)}, -\right) = \arg\min_{\mathbf{A},\mathbf{X}} \frac{1}{2} \|\mathbf{Y} - \mathbf{A}\mathbf{X}\|_F^2, \ \text{supp}(\mathbf{X}) = \text{supp}(\mathbf{X}^{(k)}), \tag{2.38}$$

where $\mathbf{X}^{(k+1)}$ is obtained by applying sparse coding with dictionary $\mathbf{A}^{(k+1)}$. The result $(-)$ in (2.38) means that the solution of $\mathbf{X}$ is not relevant, but it is allowed to change during the dictionary update. The first paper to work in the perspective of dictionary learning was the Method of Optimal Directions (MOD) [24], in which the update of the dictionary is a direct least squares minimization of the reconstruction error w.r.t. to the dictionary:

$$\mathbf{A}^{(k+1)} = \arg\min_{\mathbf{A}} \frac{1}{2} \left\|\mathbf{Y} - \mathbf{A}\mathbf{X}^{(k+1)}\right\|_F^2 = \left(\mathbf{Y}\ \mathbf{X}^{(k+1)\,T}\right) \left(\mathbf{X}^{(k+1)}\ \mathbf{X}^{(k+1)\,T}\right)^{-1} \tag{2.39}$$

and a subsequent normalization of each atom of the dictionary to the unit norm. K-SVD [1] gives a more agressive update of $\mathbf{A}$ by considering that if the support of $\mathbf{X}$ is kept constant, then the objective function could be minimized w.r.t $\mathbf{A}$ and $\mathbf{X}$ simultaneously without changing the sparsity of $\mathbf{X}$. Each atom is updated individually, by applying singular value decomposition (SVD) to an error matrix and obtaining the largest left and right singular vectors, which correspond to the updated atom and its new coefficients, respectively. Online Dictionary Learning (ODL) [40] realizes that the terms $\mathbf{Y}\ \mathbf{X}^{(k+1)\,T}$ and $\mathbf{X}^{(k+1)}\ \mathbf{X}^{(k+1)\,T}$ can be built with aggregated data from previous samples. For any new measurement $\mathbf{y}^{(k+1)}$ and the corresponding estimated coefficients $\mathbf{x}^{(k+1)}$, the new matrices are computed as:

$$\begin{aligned} \mathbf{Y}^{(k+1)}\ \mathbf{X}^{(k+1)\,T} &= \mathbf{Y}^{(k)}\ \mathbf{X}^{(k)\,T} + \mathbf{y}^{(k+1)}\ \mathbf{x}^{(k+1)\,T} \\ \mathbf{X}^{(k+1)}\ \mathbf{X}^{(k+1)\,T} &= \mathbf{X}^{(k)}\ \mathbf{X}^{(k)\,T} + \mathbf{x}^{(k+1)}\ \mathbf{x}^{(k+1)\,T} \end{aligned} \tag{2.40}$$

As a result, storage complexity is independent of the number of samples used for training and training can be done in a sample by sample basis (online learning). Recursive Least Squares - Dictionary Learning Algorithm (RLS-DLA) [50] uses the Woodbury matrix identity to simplify the operation (2.39) based on the previous iteration. This is identical in nature to Recursive Least Squares, except that it is applied in the dictionary learning context. Finally, the authors of [18, 75] identified that the main cause of fixed points are singularities in the dictionary: points in which the dictionary becomes insufficiently incoherent for the update of $\mathbf{X}$ to proceed. By changing the formulation of the problem, the singularities are avoided, thus leading to a better probability of reaching the global minimum.

2.8 Conclusions

The introduced algorithms for sparse coding take different approaches to solving the same problem. The variety of options in the literature hint to the facts that a) the computational time of the proposed algorithms is not yet satisfactory and b) there is no clear winner for all situations. Depending on the conditions of the problem, some methods are preferable than others. Dictionary Learning is bringing dictionaries that closely match the new data to the sparse coding framework, promoting better results. After reviewing the history of the sparse coding algorithms, this chapter goes in depth into the ones based on Duality and Augmented Lagrangian, describing its theory. Their importance will be shown in the benchmarks done in the following chapter. The remainder of the work in this dissertation is based on this class of algorithms.

Chapter 3

Benchmarking L1 and L0 Solvers

In this chapter, a benchmark of several L1 solvers is done, proving that DADMM has the best performance for the considered test cases. In addition, this chapter performs a comparison between DADMM and fast greedy algorithms that solve L0 problems. Although originally there was a large gap between greedy algorithms and L1 solvers in terms of speed, this chapter shows that this gap has shrinked significantly.

3.1 Benchmarking DADMM with L1 Algorithms

This section compares DADMM and DALM with other algorithms presented in section 2.2.1 and shows that DADMM is among the fastest algorithms, while DALM takes the least number of iterations to converge.

3.1.1 Benchmark Setup

The conditions for image representation may be adverse for sparse coding algorithms. For instance, images are contaminated by noise and objects in it are often obstructed, the sparsity level varies in a wide range and the atoms that constitute the dictionaries learned from natural images have, in general, some level of correlation. Therefore, it

is important that an algorithm behaves well under this range of conditions. Namely, their performance will be studied for a range of:

- correlations between basis of the dictionary;
- sparsity of the original signal;
- regularization parameter;

Unless otherwise stated, the following settings hold: the results are averaged across 50 trials; the dictionary $\mathbf{A} \in \mathbb{R}^{M \times N}$ was made of samples taken from the standard normal distribution and its columns were subsequently normalized to their ℓ_2-norm; For each trial, the true solution $\mathbf{x}^\star \in \mathbb{R}^N$ was generated by selecting S positions at random and filling them with random values from the same normal distribution. The observation was obtained by $\mathbf{y} = \mathbf{A}\mathbf{x}^\star + \mathbf{n}$, where each element of $\mathbf{n}$ was obtained from the same distribution and multiplied by $0.01/\left\|\mathbf{A}\mathbf{x}^\star\right\|_2$. In each test, the processing time, the number of iterations, the objective function value and the relative error

$$\frac{\left\|\hat{\mathbf{x}}^\star - \hat{\mathbf{x}}\right\|_2}{\left\|\hat{\mathbf{x}}^\star\right\|_2}, \tag{3.1}$$

where $\hat{\mathbf{x}}$ is the estimated solution and $\hat{\mathbf{x}}^\star$ is the optimal solution to the problem when solved with high accuracy, were measured against the test variable. All algorithms, with the exception of the homotopy method, stopped when the condition:

$$\frac{f(\mathbf{x}_{(k)}) - f(\hat{\mathbf{x}}^\star)}{f(\hat{\mathbf{x}}^\star)} < \varepsilon \tag{3.2}$$

was met, where $f(\mathbf{x}) = \frac{1}{2}\left\|\mathbf{y} - \mathbf{A}\mathbf{x}\right\|_2^2 + \lambda\left\|\mathbf{x}\right\|_1$. $\varepsilon = 10^{-3}$, $M = 256$, $N = 512$ and $S = 40$. The regularization parameter λ was set to 0.01. Setup times of the algorithms, including operations that do not depend on the observation vectors, were excluded from the time measurements. All algorithms were implemented in Matlab and simulated in a Quad-Core i5 @ 2.50GHz CPU with 4GB memory.

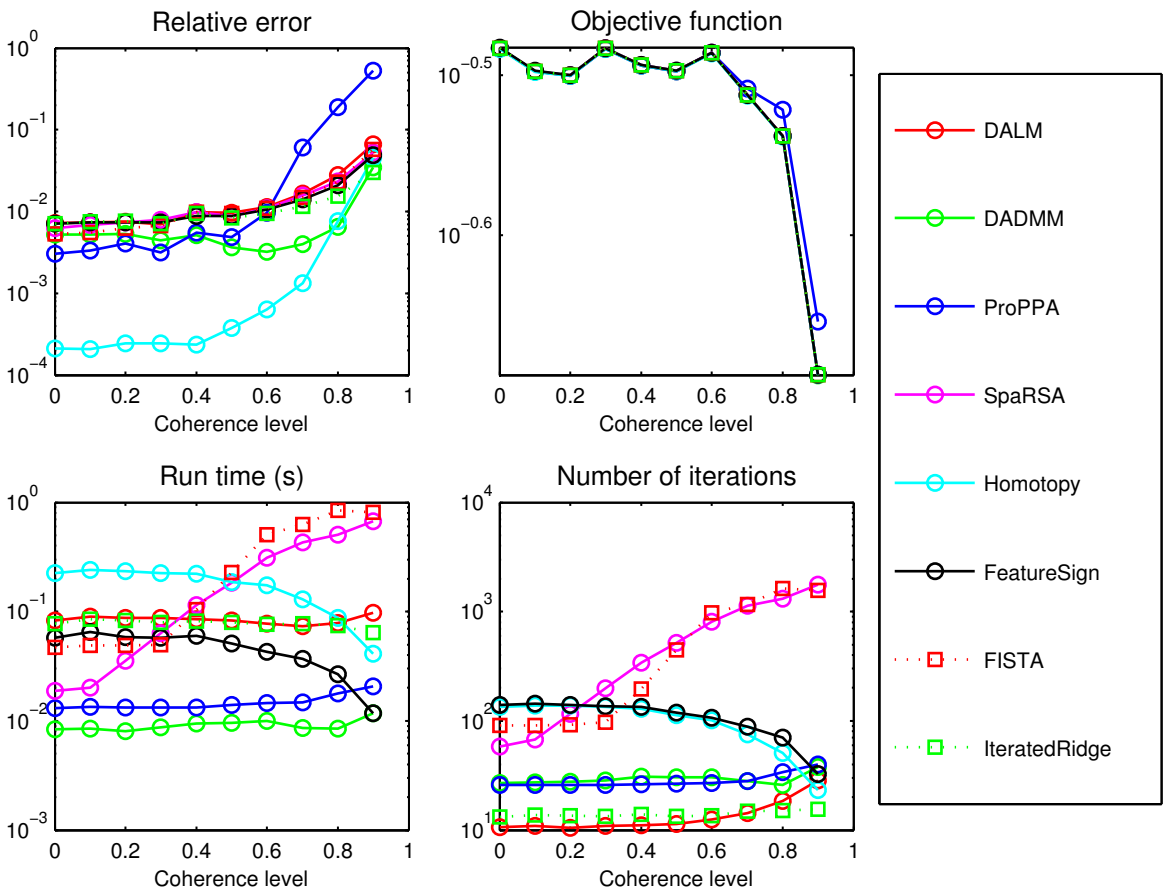


Figure 3.1: Relative error of solution, objective function, processing time and number of iterations vs. τ, for several L1 algorithms.

3.1.2 Results

3.1.2.1 Correlated Basis of the Dictionary

To produce correlated dictionary basis, a temporary matrix $\tilde{\mathbf{A}}$ was generated by taking samples from the standard normal distribution. Then, each column of a matrix $\mathbf{A}$ was computed as $\mathbf{a}_i = \tau\tilde{\mathbf{a}}_1 + (1-\tau)\tilde{\mathbf{a}}_i$, for $i = 1, ..., N$ and $\tau \in [0, 1]$. Then, all columns were ℓ_2-normalized. Therefore, $\tau = 1$ gave exact replicas of the first column while $\tau = 0$ disrupted the correlation entirely. Figure 3.1 shows the relative error, objective function value, processing time and number of iterations for several algorithms described in section 2.2.1 depending on parameter τ.

As shown in figure 3.1, first-order methods such as FISTA and SpaRSA increase

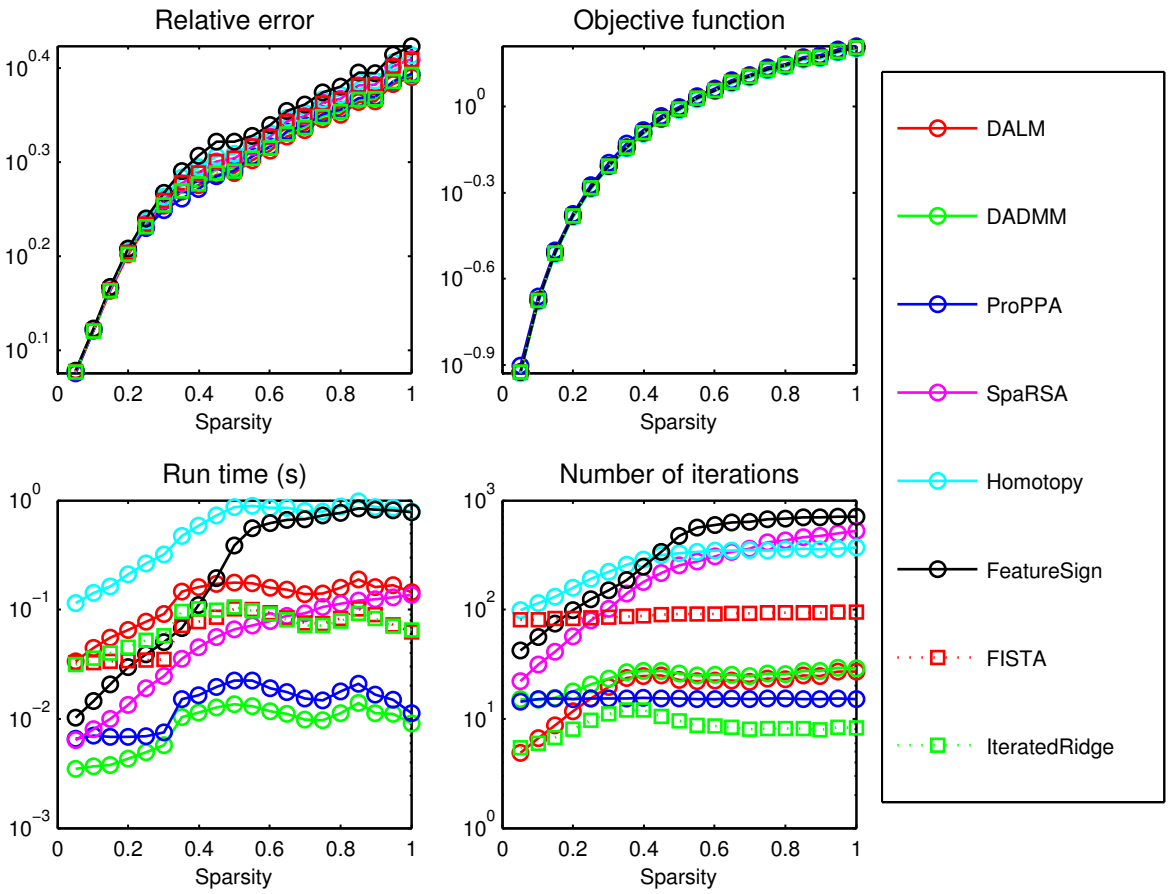


Figure 3.2: Relative error of solution, objective function, processing time and number of iterations vs. S/M, for several L1 algorithms.

their processing time with increased coherence in the dictionary. Other methods behave more or less steady for the whole range of dictionary coherency. DALM stands out for its low number of iterations, while DADMM excels in runtime.

3.1.2.2 Sparsity of the Signal

The number of non-zero elements of $\mathbf{x}^\star$ was increased from 1 to M. Figure 3.2 shows the relative error, objective function value, processing time and number of iterations for the same algorithms depending on parameter S/M.

As the number of non-zero elements S is increased, all algorithms reduce their accuracy. However, active-set algorithms such as Homotopy or Feature-sign search,

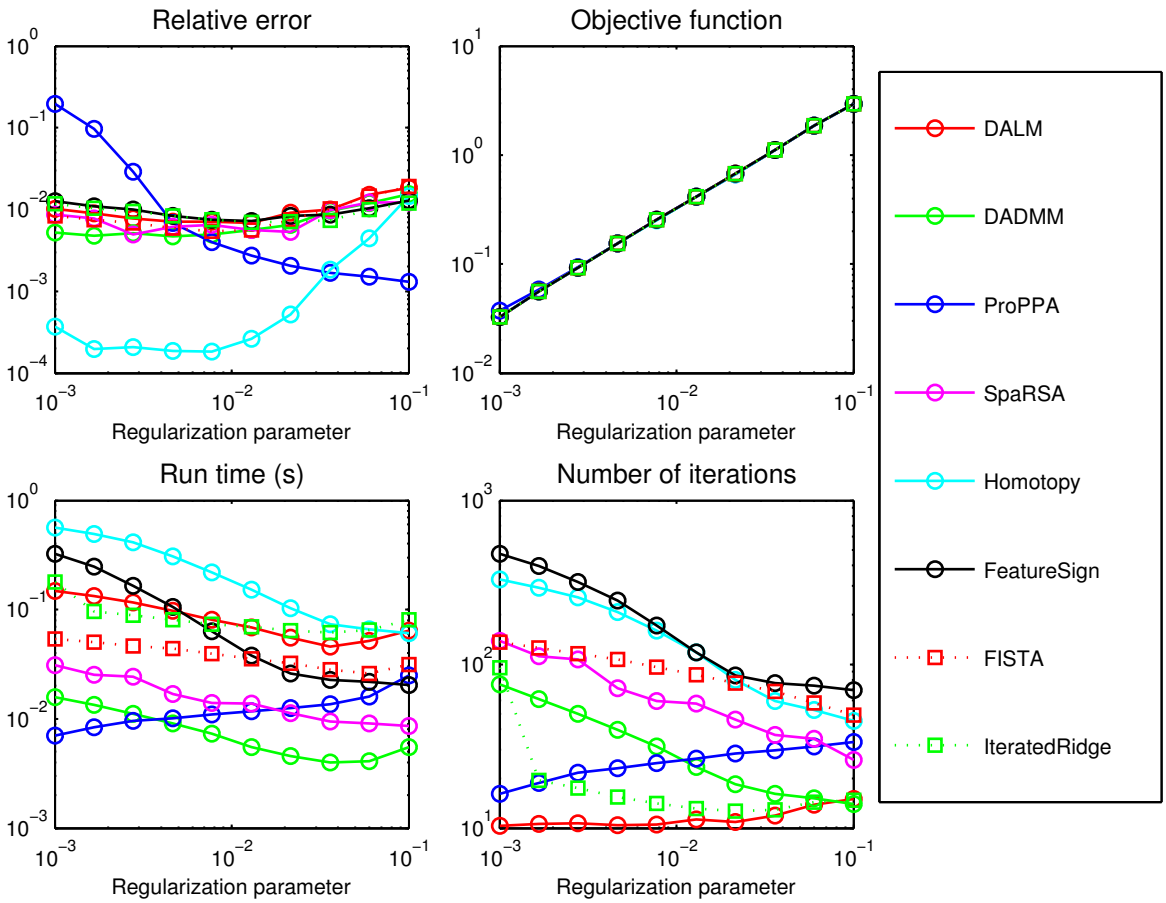


Figure 3.3: Relative error of solution, objective function, processing time and number of iterations vs. λ, for several L1 algorithms.

as well SpaRSA and DALM, suffer a penalty in processing time as S is increased. As before, DADMM has the best runtime.

3.1.2.3 Regularization Parameter

This test is intented to show how each algorithm performs for different regularization parameters. For example, the homotopy method is expected to run longer for smaller λ, since it traces a longer path of λ.

In fact, figure 3.3 confirms that most algorithms increase their processing time for smaller λ, with the exception of ProPPA. However, it does not perform well for small λ, as can be seen by the large relative error of the solution. Once again, DALM

stands out for its low number of iterations and DADMM has the best runtime more often.

3.2 Benchmarking DADMM with L0 algorithms

This section compares the DADMM L1 algorithm with greedy L0 algorithms presented in section 2.2.2. Although it is an apples with oranges comparison, this is relevant to demonstrate that state of the art L1 algorithms do not fall behind L0 algorithms in terms of run time. In fact, the evolution of L0 algorithms has been to improve theoretical recovery performance at the cost of slower iterations and the evolution of L1 algorithms has been simply to become faster. This, in turn, shortens the gap between L0 and L1 algorithms.

3.2.1 Benchmark Setup

The benchmark setup is identical to section 3.1.1, with the following exceptions: for DADMM, the stopping criterion was now set to

$$\text{duality gap} < \varepsilon_{dg}, \tag{3.3}$$

with $\varepsilon_{dg} = 10^{-4}$. For the L0 algorithms, the stopping criterion was still the objective function in equation (3.2), but with $\varepsilon = 0.02$, since some algorithms had a zig-zagging effect that did not converge with the original $\varepsilon = 10^{-3}$. Since we are interested in evaluating the recovery performance between L0 and L1 solvers, the relative error was defined as

$$\frac{\|\mathbf{x}^\star - \hat{\mathbf{x}}\|_2}{\|\mathbf{x}^\star\|_2}, \tag{3.4}$$

where $\mathbf{x}^\star$ is the true solution.

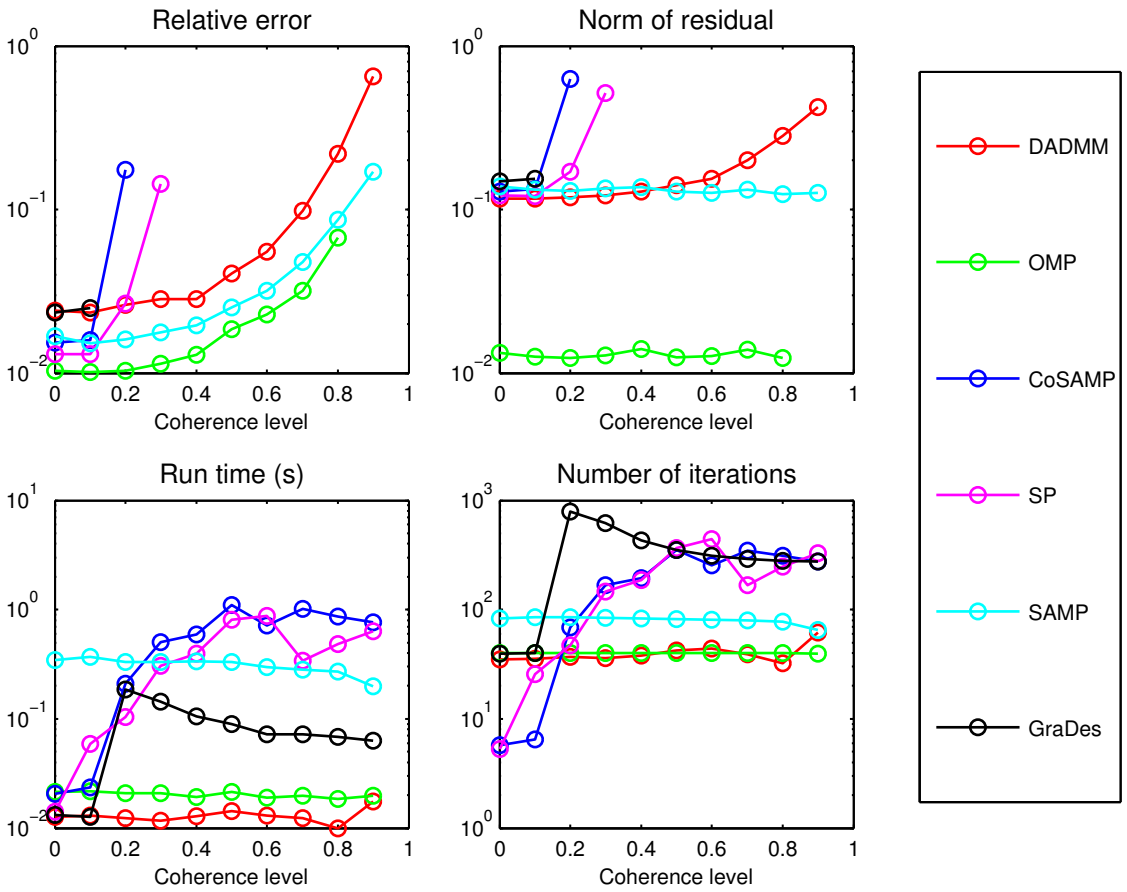


Figure 3.4: Relative error of solution, norm of residual, processing time and number of iterations vs. τ, for several L0 algorithms and DADMM.

3.2.2 Results

3.2.2.1 Correlated Basis of the Dictionary

The dictionary was constructed in the same way as described in section 3.1.2.1. Figure 3.4 shows the relative solution error, residual norm, processing time and number of iterations for several algorithms described in section 2.2.2 and DADMM depending on parameter τ.

The figure confirms that DADMM has a competitive run time, only compared with OMP, albeit with worse recovery properties. In particular, OMP has the knowledge of the true sparsity of the signal and, although SAMP does not, it takes a considerable amount of time to recover the coefficients. SP, CoSAMP and GraDes cannot recover

any coefficients if the dictionary is moderately/heavily coherent.

3.2.2.2 Sparsity of the Signal

Greedy algorithms are usually heavily dependent on the number of non-zero elements S defined for the solution. For instance, OMP iterates S times, while the time complexity of each iteration of SP and CoSAMP is $\mathcal{O}(S^2)$. Figure 3.5 shows the relative error, residual norm, processing time and number of iterations for the same algorithms depending on parameter S. SP, CoSAMP and SAMP share a common structure and it is not surprising that they behaved similarly in some aspects. Here, they show the best recovery properties, along with OMP, but have the quickest transition to non-recoverability. Once again, DADMM presents competitive run time, but falls a bit short in recovery performance. Take note that L0 algorithms recover the coefficients with the knowledge of the true sparsity, with the exception of SAMP. DADMM, on the other hand, only has the λ parameter to indirectly control the sparsity of the solution, which was left fixed in this case.

3.3 Conclusions

This section takes several conclusions from the information previously presented in this chapter.

- Among the algorithms that solve the compressive sensing problem, DADMM is one of the most consistent and fast algorithms in the literature. The benchmarks done in section 3.1 confirm that DADMM is most of the time the algorithm with the fastest convergence. The problem under test has an overcomplete dictionary, with more columns than rows. Although this condition is favorable for algorithms that solve the dual problem, which corroborates the presented data, it covers most of the applications using sparse coding. The superior performance

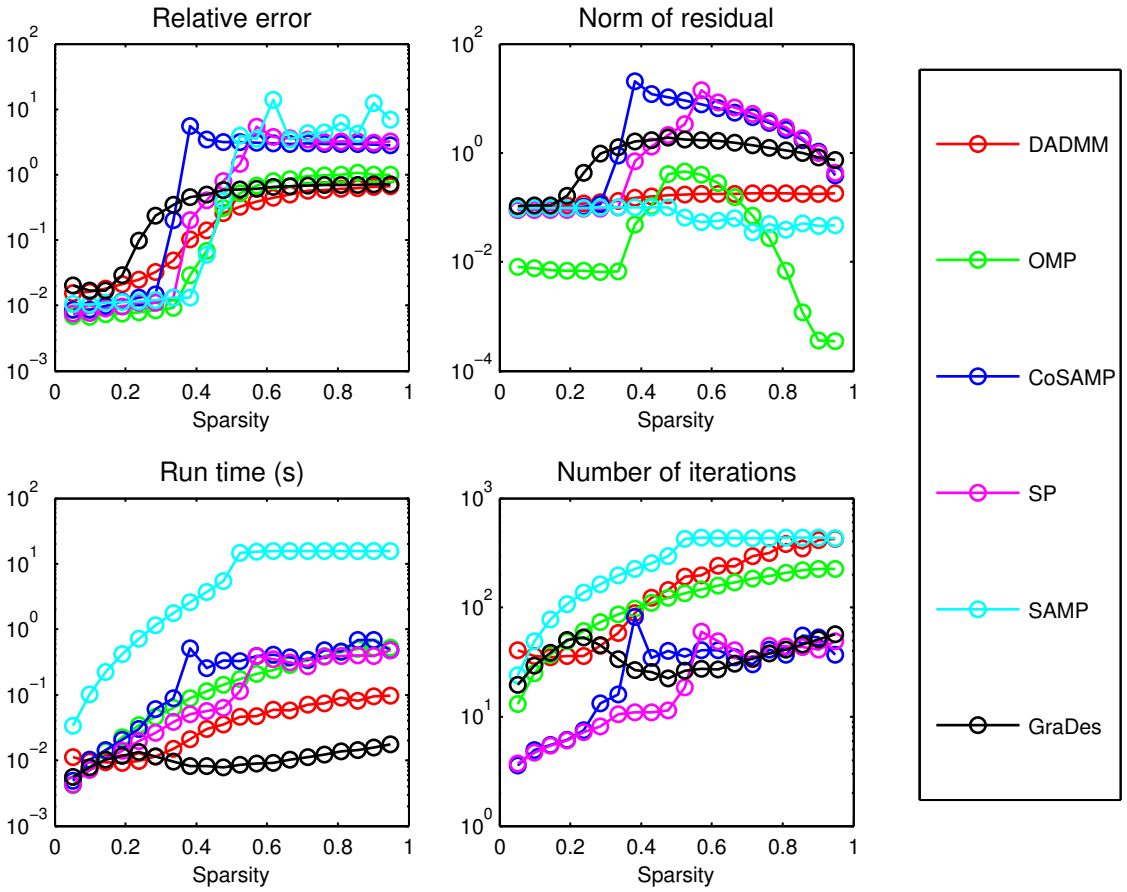


Figure 3.5: Relative error of solution, norm of residual, processing time and number of iterations vs. S/M, for several L0 algorithms and DADMM.

of the noiseless version of DADMM has also been confirmed in the context of robust face recognition [65], where the dictionaries are very overcomplete.

- The trend of greedy L0 algorithms has been to have better theoretical recovery properties at a higher computational cost. The trend of L1 algorithms has been to become faster, since the reconstruction properties are guaranteed by the problem itself. Both trends are converging into algorithms with similar theoretical recovery guarantees and run times. In practice, L0 algorithms show better recovery properties than L1 algorithms. The reason is that ℓ_1-regularized problems require $O(k\log(n/k))$ measurements to guarantee the RIP with high probability, while problems based on the ℓ_0-norm require only $k+1$ measurements to recover the true signal. However, only an exaustive search over the solutions with a support of size k could find the exact solution of the latter problems, and greedy algorithms do the same with a number of measurements somewhere between $k+1$ and $O(k\log(n/k))$.

- The problems being solved by L1 and L0 algorithms are not the same, which renders the comparison done in section 3.2 a bit faulty. The L0 algorithms, with the exception of SAMP, have the prior knowledge of the exact sparsity of the true signal, which gives them a recovery performance advantage. This fact potentially deviates the comparison of the recovery performance between DADMM (or in fact L1 algorithms) and L0 algorithms. Nevertheless, section 3.2 shows that DADMM is comparable or even surpasses greedy algorithms in terms of run time when the solution is not extremelly sparse, while having recovery properties that do not fall much behind L0 algorithms. In particular, algorithms with excellent recovery properties such as Threshold-ISD [61] run truncated versions of L1 algorithms in a small number of iterations. Therefore, L1 algorithms (and their runtime) are fundamental components in this type of

methods.

In the previous benchmarks, DADMM had a fixed η parameter. As referred before, using a variable penalty parameter η can improve the convergence rate. However, changing η during the progression of the algorithm comes at the cost of recomputing the inverse of the Hessian. The next chapter suggests an alternative that not only allows to change the penalty parameter at no cost, but also makes each iteration faster. It also analyzes which η sequence improves convergence and how it relates with variables existent in the algorithm.

Chapter 4

Accelerating DADMM with Eigendecomposition

This chapter presents an adaptation to DADMM that allows the penalty parameter η to be changed at no extra cost. Furthermore, several simplifications are possible with the proposed method that make each iteration faster. A thorough analysis will reveal which η sequence improves convergence the most and how it relates with variables existent in the algorithm. It will be shown that reducing η as the method progresses improves convergence. Speeding up sparse coding solvers is particularly important in time-sensitive applications, such as object tracking [60] or video compressive sensing [47], or where time equals financial cost [39]. Apart from these, "shaving" time from methods that are heavily based on sparse coding, which are usually time consuming, helps the argument to use them.

4.1 Applying Eigendecomposition

The proposed adaptation is based on the eigendecomposition of a term inside the inverse in algorithm 2.4. Before moving on, recall the α update expression (2.37):

$$\alpha_{(k+1)} = \left(\mathbf{I} + \eta\mathbf{A}\mathbf{A}^T\right)^{-1}\left(\mathbf{y} - \mathbf{A}\left(\mathbf{x}_{(k)} - \eta\nu_{(k)}\right)\right). \tag{4.1}$$

The matrix $\mathbf{A}\mathbf{A}^T$ is a square symmetric matrix, for which there is an eigendecomposition, $\mathbf{A}\mathbf{A}^T = \mathbf{U}\mathbf{\Sigma}\mathbf{U}^T$, where the matrix $\mathbf{U}$ is orthogonal and $\mathbf{\Sigma}$ is diagonal. This matrix has the same dimensions as the number of measurements m. Its eigendecomposition takes $\mathcal{O}(m^3)$ time to compute and complexity $\mathcal{O}(m^2)$ to store. On the other hand, second-order methods that solve the primal problem have the term $\mathbf{A}^T\mathbf{A}$, whose dimensions are proportional to the (potentially large) number of atoms n in the dictionary. The storage requirements for the decomposition of $\mathbf{A}^T\mathbf{A}$ is $\mathcal{O}(n^2)$ and its computation complexity is $\mathcal{O}(n^3)$, which becomes impractical as the number of atoms in the dictionary grows. Therefore, the proposed adaptation, as well as the original DADMM, becomes more efficient in applications where the number of measurements is considerably smaller than the number of atoms. These include applications where the dictionary is built from training samples, such as [63], or applications where larger dictionaries improve performance, such as image classification based on sparse coding [67].

By applying eigendecomposition to $\mathbf{A}\mathbf{A}^T$ in (4.1), we obtain

$$\begin{aligned}
\alpha_{(k+1)} &= \left(\mathbf{I} + \eta\mathbf{U}\mathbf{\Sigma}\mathbf{U}^T\right)^{-1}\left(\mathbf{y} - \mathbf{A}\left(\mathbf{x}_{(k)} - \eta\nu_{(k)}\right)\right) \\
\alpha_{(k+1)} &= \left(\mathbf{U}\mathbf{U}^T + \eta\mathbf{U}\mathbf{\Sigma}\mathbf{U}^T\right)^{-1}\left(\mathbf{y} - \mathbf{A}\left(\mathbf{x}_{(k)} - \eta\nu_{(k)}\right)\right) \\
\alpha_{(k+1)} &= \mathbf{U}\left(\eta\mathbf{\Sigma} + \mathbf{I}\right)^{-1}\mathbf{U}^T\left(\mathbf{y} - \mathbf{A}\left(\mathbf{x}_{(k)} - \eta\nu_{(k)}\right)\right) \\
\underbrace{\mathbf{U}^T\alpha_{(k+1)}}_{\alpha'_{(k+1)}} &= \left(\eta\mathbf{\Sigma} + \mathbf{I}\right)^{-1}\left(\underbrace{\mathbf{U}^T\mathbf{y}}_{\mathbf{y}'} - \underbrace{\mathbf{U}^T\mathbf{A}}_{\mathbf{A}'}\left(\mathbf{x}_{(k)} - \eta\nu_{(k)}\right)\right) \\
\alpha'_{(k+1)} &= \left(\eta\mathbf{\Sigma} + \mathbf{I}\right)^{-1}\left(\mathbf{y}' - \mathbf{A}'\left(\mathbf{x}_{(k)} - \eta\nu_{(k)}\right)\right)
\end{aligned} \tag{4.2}$$

where steps 2 and 3 are possible because $\mathbf{U}$ is an orthogonal matrix. The update of vector α did not change, yet the matrix to be inverted is diagonal, and therefore, easily inverted. As a consequence, there is no restriction for the penalty parameter η to be fixed every iteration.

In addition, note that α is only used in algorithm 2.4 by the operator $\mathbf{A}^T$. The following relation can be confirmed between the original α and the new α':

$$\mathbf{A}^T\alpha = \mathbf{A}^T\mathbf{U}\mathbf{U}^T\alpha = \mathbf{A}'^T\alpha'. \tag{4.3}$$

Concerning the updates of the remaining variables, $\mathbf{z}$ and $\mathbf{x}$, they remain unchanged, except that the term $\mathbf{A}^T\alpha$ is replaced by $\mathbf{A}'^T\alpha'$, according to (4.3). For the calculation of the dual objective value,

$$\begin{aligned} -\frac{1}{2}\left\|\alpha\right\|_2^2 + \alpha^T\mathbf{y} &= -\frac{1}{2}\left\|\mathbf{U}\alpha'\right\|_2^2 + \left(\mathbf{U}\alpha'\right)^T\mathbf{y} \\ &= -\frac{1}{2}\left\|\alpha'\right\|_2^2 + \alpha'^T\mathbf{U}^T\mathbf{y} \\ &= -\frac{1}{2}\left\|\alpha'\right\|_2^2 + \alpha'^T\mathbf{y}' \end{aligned} \tag{4.4}$$

which is identical to $f(\alpha)$ in (2.35), except the new set of variables is used. Regarding the primal objective function

$$\begin{aligned} \frac{1}{2}\left\|\mathbf{y}-\mathbf{A}\mathbf{x}\right\|_2^2 + \lambda\left\|\mathbf{x}\right\|_1 &= \frac{1}{2}\left\|\mathbf{U}^T\left(\mathbf{y}-\mathbf{A}\mathbf{x}\right)\right\|_2^2 + \lambda\left\|\mathbf{x}\right\|_1 \\ &= \frac{1}{2}\left\|\mathbf{y}'-\mathbf{A}'\mathbf{x}\right\|_2^2 + \lambda\left\|\mathbf{x}\right\|_1, \end{aligned} \tag{4.5}$$

where the first step can be taken because the multiplication by an orthogonal matrix does not change the ℓ_2-norm. In summary, the new terms α', $\mathbf{A}'$ and $\mathbf{y}'$ can be used instead of the original counterparts in all parts of the algorithm.

The decomposition of $\mathbf{A}\mathbf{A}^T$ is done prior to applying the proposed method. This method is only useful when the dictionary is reused several times, since the decomposition is done only once, but the algorithm is applied to a dataset of observations. By working with the decomposed matrix, the term to be inverted during the α update becomes diagonal, which brings two benefits: the inverse is easily computed and multiplied and, as a consequence, the penalty parameter can be updated at each iteration.

4.1.1 Reducing Computational Complexity of α Update

In practice, the algorithm has some room for mathematical manipulation. Note that the term $(\mathbf{x} - \eta\nu)$ in (4.2) is not sparse and the multiplication $\mathbf{A}'(\mathbf{x} - \eta\nu)$ has complexity $\mathcal{O}(mn)$. According to equations (2.31) and (4.3), we now have:

$$\begin{aligned} \mathbf{x}_{(k)} &= \text{shrink}_{\eta\lambda}\left(\mathbf{x}_{(k-1)} + \eta\mathbf{A}'^T\alpha'_{(k)}\right) \\ &= \mathbf{x}_{(k-1)} + \eta\left(\mathbf{A}'^T\alpha'_{(k)} - \nu_{(k)}\right). \end{aligned} \tag{4.6}$$

By replacing $\eta\nu_{(k)} = -\mathbf{x}_{(k)} + \mathbf{x}_{(k-1)} + \eta\mathbf{A}'^T\alpha'_{(k)}$ in (4.2), the α' update becomes

$$\begin{aligned} \alpha'_{(k+1)} &= (\eta\boldsymbol{\Sigma} + \mathbf{I})^{-1}\left(\mathbf{y}' - \mathbf{A}'\left(\mathbf{x}_{(k)} - \eta\nu_{(k)}\right)\right) \\ &= (\eta\boldsymbol{\Sigma} + \mathbf{I})^{-1}\left(\mathbf{y}' - \mathbf{A}'\left(\mathbf{x}_{(k)} - \mathbf{x}_{(k-1)} + \mathbf{x}_{(k)} - \eta\mathbf{A}'^T\alpha'_{(k)}\right)\right) \\ &= (\eta\boldsymbol{\Sigma} + \mathbf{I})^{-1}\left(\eta\Sigma\alpha'_{(k)} + \left(\mathbf{y}' - \mathbf{A}'\left(2\mathbf{x}_{(k)} - \mathbf{x}_{(k-1)}\right)\right)\right), \end{aligned} \tag{4.7}$$

where the last step is valid since $\mathbf{A}'\mathbf{A}'^T = \mathbf{U}^T\mathbf{A}\left(\mathbf{U}^T\mathbf{A}\right)^T = \mathbf{U}^T\mathbf{A}\mathbf{A}^T\mathbf{U} = \mathbf{U}^T\mathbf{U}\boldsymbol{\Sigma}\mathbf{U}^T\mathbf{U} = \boldsymbol{\Sigma}$.

In this case, $\mathbf{x}$ is a sparse variable due to the shrinkage operator in (4.6). The complexity of the multiplication $\mathbf{A}'\mathbf{x}_{(k)}$ is proportional to the number of non-zero elements in the term $\mathbf{x}_{(k)}$ and not its dimensionality. Therefore, compared to the complexity $\mathcal{O}(mn)$ of $\mathbf{A}(\mathbf{x} - \eta\nu)$ in the original DADMM, the operation $\mathbf{A}'\mathbf{x}_{(k)}$ requires only $ms_{(k)}$ computations, where $s_{(k)}$ is the sparsity of $\mathbf{x}_{(k)}$. This is particularly useful when the dictionary has a large number of atoms.

In addition, the term $\mathbf{A}'\mathbf{x}_{(k)}$ is used to calculate the objective function and therefore the duality gap, both of which are commonly used for the stopping criterion. As a result, the original DADMM would require an extra matrix-vector multiplication. In the following experiments, extra operations necessary to validate the stopping criterion were not accounted for in the run time.

4.2 Update Rule of Penalty Parameter

An optimal selection of the penalty parameter, fixed or adaptive, for ADMM has remained elusive. On the theoretical side, [27, 45] determine the optimal fixed parameter for quadratic problems, which is dependent on the problem properties. No article was found for the case of ℓ_1-regularized problems or for adaptive selection (selection at each iteration). On the empirical side, it has been shown that an adaptive step improves convergence and several heuristics were proposed. [34] chooses to monotonically increase or reduce η depending on the problem conditions and in [13], η is increased every iteration. Ultimately, the best η sequence is problem dependent and should be analyzed in a case-by-case basis.

This dissertation takes a pratical approach by determining the trend of the optimal parameter throughout iterations, during a discovery phase. To that end, in this phase and at iteration k, the penalty parameter that reduces the term $\left\|\mathbf{x}_{(k+1)} - \mathbf{x}^\star\right\|_2$ the most, where $\mathbf{x}^\star$ is the true coefficient vector, is determined and stored. All steps

are taken with their best penalty parameter $\eta^\star_{(k)}$, leading to the optimal parameter sequence $\{\eta^\star_{(k)}\}$ for each trial. Note that this is an exploratory phase and it is only possible because $\mathbf{x}^\star$ is known in advance. This approach is valid when a representative dataset is available, from which an average penalty parameter sequence can be obtained.

Analysis of the discovery phase was done for different settings of the following parameters: number of observations m (figure 4.1), sparsity k of the true signal $\mathbf{x}^\star$ (figure 4.2), regularization parameter λ (figure 4.3), levels of noise (figure 4.4) and dictionaries (figure 4.5). Every figure shows in the top row the solution error against each iteration for different DADMM η configurations (including the discovery procedure), averaged over 20 trials, while the bottom row has scatter plots of the η parameter for DADMM in the discovery phase against the residual. Each column represents a different setting for the parameter under testing. The purpose of the top row is to show the improvement of the convergence of the algorithm when η is changed each iteration. The bottom row depicts the relation found between the optimal $\eta^\star_{(k)}$ and the residual $\left\|\mathbf{y} - \mathbf{A}\mathbf{x}_{(k)}\right\|_2$ for a given iteration. The red line in the scatter plots of the bottom row of all figures follows the same equation, described later. The parameters that were not under testing were defined as $\lambda = 10^{-4}$, $n = 256$, $m = 144$, $k = 26$, $SNR = \|\mathbf{y}\|_2 / \|\mathbf{n}\|_2 = \infty$, dictionary filled with entries obtained from a standard normal distribution and each column normalized to unit-norm. Data obtained by $\mathbf{y} = \mathbf{A}\mathbf{x}^\star + \mathbf{n}$.

Figure 4.1 shows the results for different undersampling m/n values. In all of them, compared to any fixed η, the improvement of the discovery procedure is significant. A lower undersampling does not guarantee accurate recovery of the coefficients, which scatters the relation between $\eta^\star$ and the residual.

Figure 4.2 shows the results for different sparsity levels k/m of the true coefficient vector $\mathbf{x}^\star$. In all of them, compared to any fixed η, the improvement of the discovery

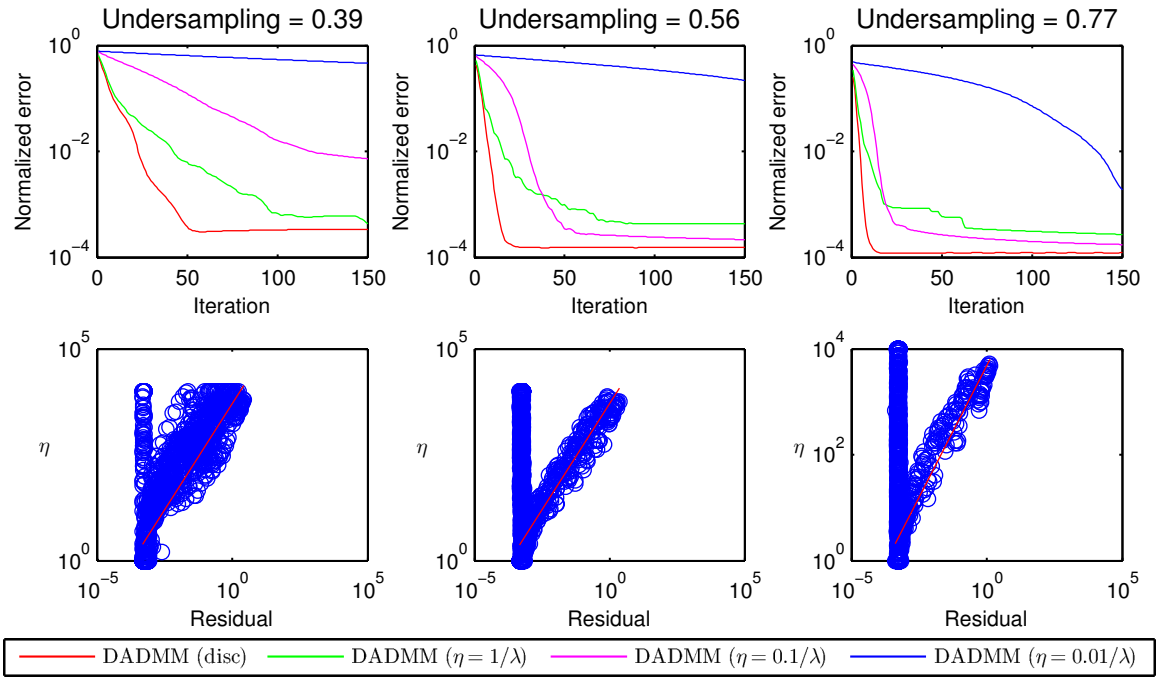


Figure 4.1: Performance comparison for DADMM in discovery phase for several undersampling conditions.
Top row: Solution error for a sequence of steps of DADMM. DADMM in discovery mode traces the penalty parameter that descends the solution error the most. Bottom row: Scatter plots of optimal η vs. residual for each iteration of the DADMM. Each column represents a different undersampling ratio. The red line follows the equation $\eta = 0.5 \left\| \mathbf{y} - \mathbf{A}\mathbf{x}_{(k)} \right\|_2 / \lambda$.

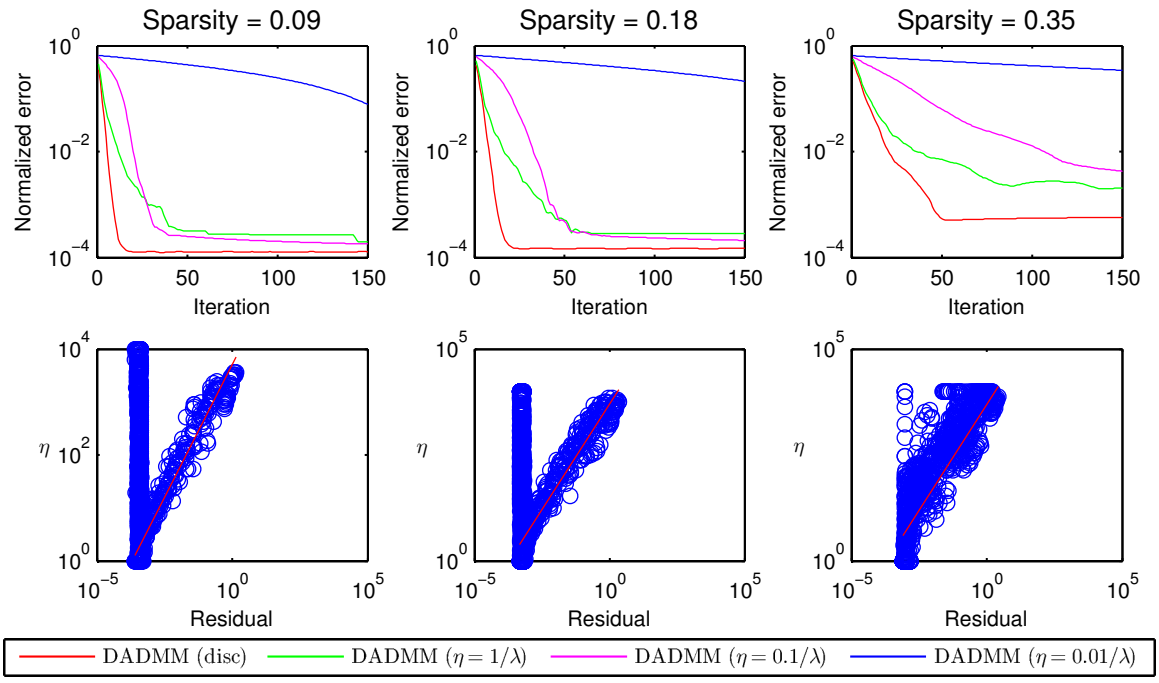


Figure 4.2: Performance comparison for DADMM in discovery phase for several sparsity conditions.
Top row: Solution error for a sequence of steps of DADMM. DADMM in discovery mode traces the penalty parameter that descends the solution error the most. Bottom row: Scatter plots of optimal η vs. residual for each iteration of the DADMM. Each column represents a different sparsity level of the true coefficient vector $\mathbf{x}^\star$. The red line follows the equation $\eta = 0.5 \left\|\mathbf{y} - \mathbf{A}\mathbf{x}_{(k)}\right\|_2 / \lambda$.

procedure is significant. A larger number of non-zero elements does not guarantee accurate recovery of the coefficients, which scatters the relation between $\eta^\star$ and the residual.

Figure 4.3 shows the results for different values of the regularization parameter λ. The improvement of changing η reduces as λ increases. In general, a larger λ is a favorable condition for DADMM, and the margin for improvement gets smaller. A larger λ also introduces scatter in the relation between the optimal η and the residual. Among the settings with fixed η, although the setting with the largest $\eta = 1/\lambda$ descends faster, it is the setting $\eta = 0.1/\lambda$ that converges first.

Figure 4.4 shows the results for different levels of SNR $\|\mathbf{y}\|_2 / \|\mathbf{n}\|_2$. The improvement of changing η reduces as the noise level increases, although the relation between

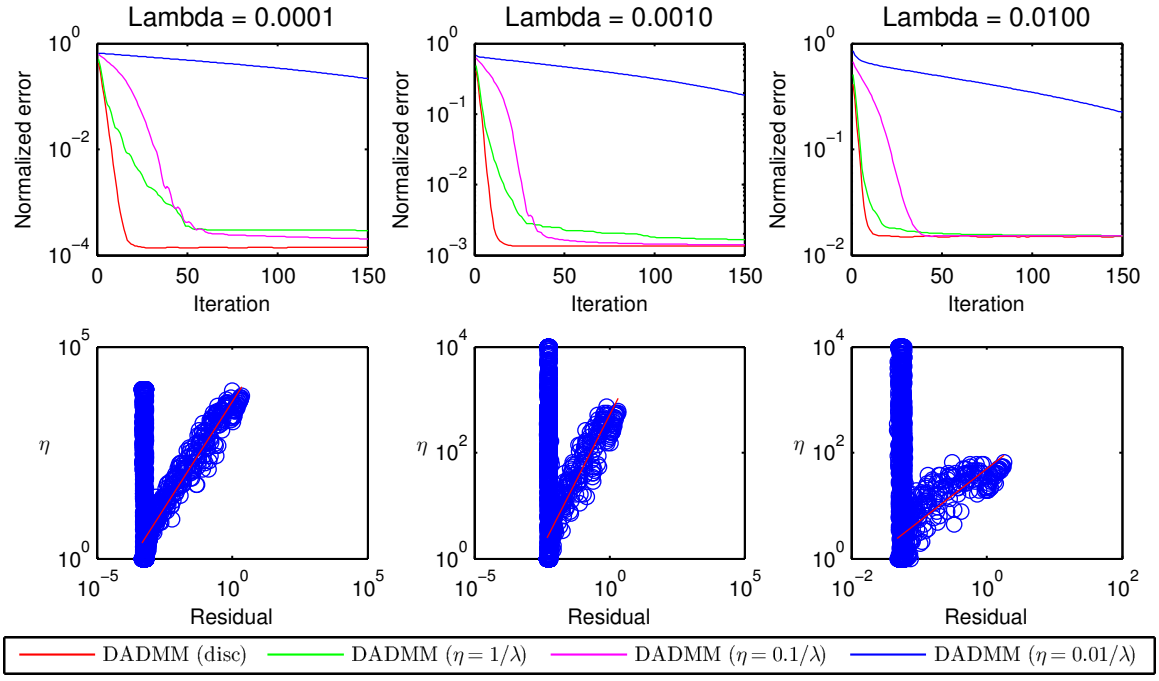


Figure 4.3: Performance comparison for DADMM in discovery phase for several regularization conditions.
Top row: Solution error for a sequence of steps of DADMM. DADMM in discovery mode traces the penalty parameter that descends the solution error the most. Bottom row: Scatter plots of optimal η vs. residual for each iteration of the DADMM. Each column represents a different regularization parameter λ. The red line follows the equation $\eta = 0.5 \left\| \mathbf{y} - \mathbf{A}\mathbf{x}_{(k)} \right\|_2 / \lambda$.

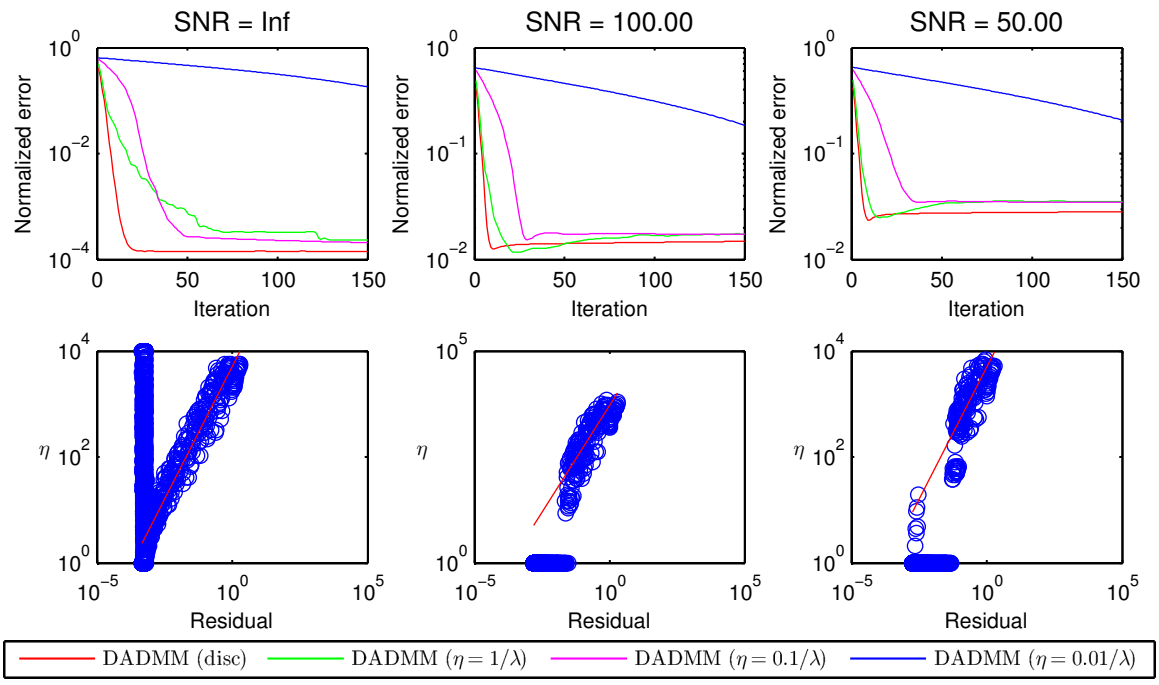


Figure 4.4: Performance comparison for DADMM in discovery phase for several noise levels.
Top row: Solution error for a sequence of steps of DADMM. DADMM in discovery mode traces the penalty parameter that descends the solution error the most. Bottom row: Scatter plots of optimal η vs. residual for each iteration of the DADMM. Each column represents a different level of signal-noise-ratio. The red line follows the equation $\eta = 0.5 \left\| \mathbf{y} - \mathbf{A}\mathbf{x}_{(k)} \right\|_2 / \lambda$.

the optimal η and the residual still holds. As the SNR reduces, the optimal solution deviates from the true coefficients, which "fools" the discovery phase: the discovery phase steps towards the $\mathbf{x}^\star$, not the optimal coefficients. As before, although the setting with the largest $\eta = 1/\lambda$ descends faster, it is the setting $\eta = 0.1/\lambda$ that converges first.

Figure 4.5 shows the results for different dictionaries. From left to right, 1) a dictionary $\mathbb{R}^{m \times n}$ with entries obtained from a normal distribution, 2) a $\mathbb{R}^{m \times n}$ over-complete 2D-DCT matrix, 3) a complete 2D-DCT $\mathbb{R}^{n \times n}$ matrix, with m random rows selected, 4) a $\mathbb{R}^{n \times n}$ haar wavelet matrix premultiplied by a matrix $\mathbb{R}^{m \times n}$ with random entries obtained from a normal distribution. The columns of all dictionaries were normalized to unit-norm. From all the dictionaries, an improvement was not seen only

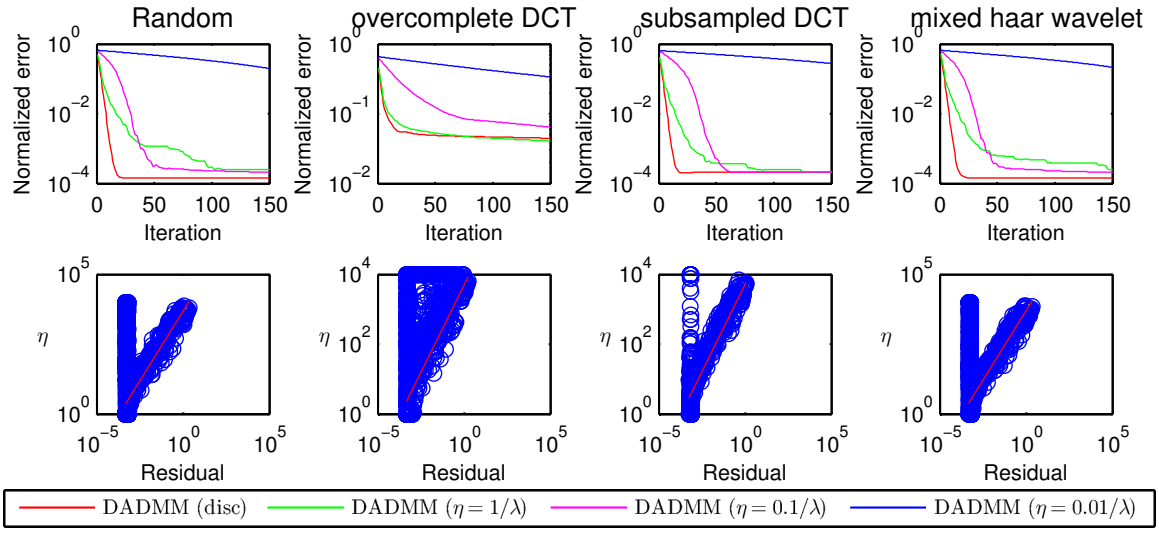


Figure 4.5: Performance comparison for DADMM in discovery phase for several dictionaries.
Top row: Solution error for a sequence of steps of DADMM. DADMM in discovery mode traces the penalty parameter that descends the solution error the most. Bottom row: Scatter plots of optimal η vs. residual for each iteration of the DADMM. Each column represents a different dictionary. The red line follows the equation $\eta = 0.5 \left\| \mathbf{y} - \mathbf{A}\mathbf{x}_{(k)} \right\|_2 / \lambda$.

in the overcomplete 2D-DCT dictionary. The large recovery error indicates that the settings used were not sufficient to fulfill the Restricted Isometry Propery for this dictionary and an accurate recovery was not possible. As a result, DADMM in the discovery phase is stepping towards the optimal solution while selecting η that steps towards $\mathbf{x}^\star$. This incoherency breaks down the relation between $\eta^\star$ and the residual.

In all scatter plots with a clear relation between $\eta^\star$ and the residual, the red line, defined as:

$$\eta_{(k)} = 0.5 \frac{\left\| \mathbf{y} - \mathbf{A}\mathbf{x}_{(k)} \right\|_2}{\lambda}, \tag{4.8}$$

is a good linear fitting. Therefore, given the newly obtained freedom of the penalty parameter, we would like the progression of η for DADMM with unknown data to mimick the progression of $\eta^\star$ in the discovery phase. The equation (4.8) must be slightly adapted for a practical use:

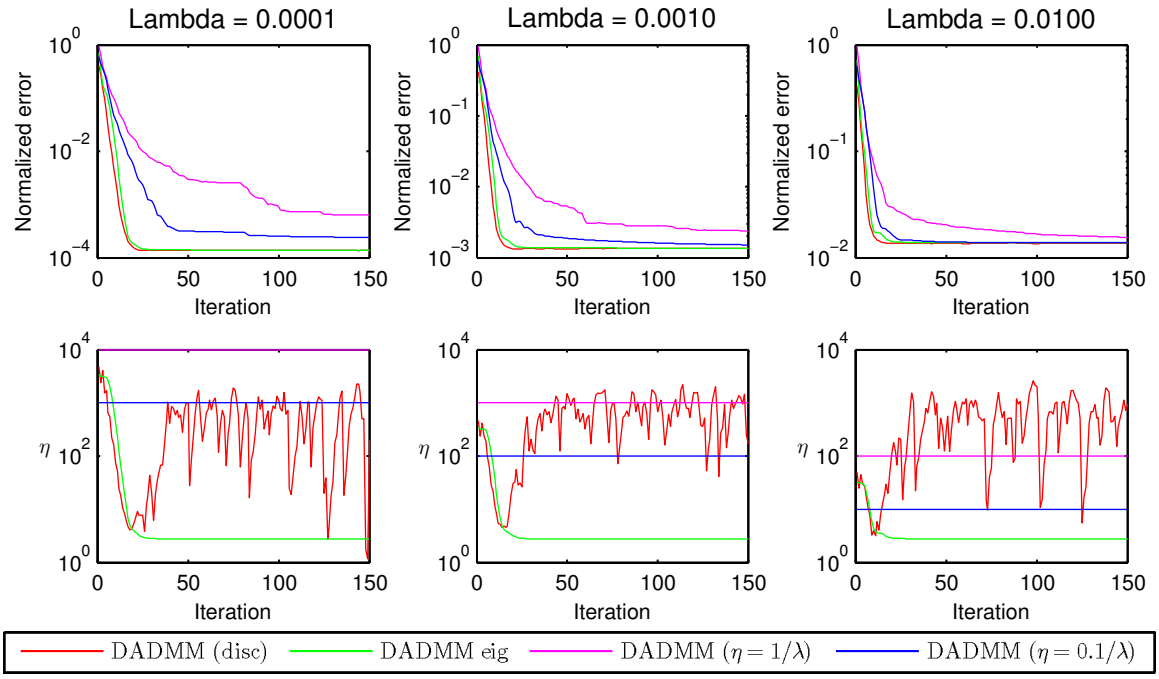
Algorithm 4.1 DADMM-eig.

$$
\begin{array}{rcl}
\text{Initialize} & & k = 0, \alpha'_{(0)}, \mathbf{x}_{(0)} \text{ and } \eta_{(0)} \\
\text{Repeat} & & \\
\alpha'_{(k+1)} & = & (\eta\boldsymbol{\Sigma} + \mathbf{I})^{-1} \left(\eta_{(k)}\Sigma\alpha'_{(k)} + \left(\mathbf{y}' - \mathbf{A}' \left(2\mathbf{x}_{(k)} - \mathbf{x}_{(k-1)}\right)\right)\right) \\
\mathbf{x}_{(k+1)} & = & \text{shrink}_{\lambda\eta} \left(\mathbf{x}_{(k)} + \eta_{(k)}\mathbf{A}'^{T}\alpha'_{(k+1)}\right) \\
\eta_{(k+1)} & = & \min \left(0.5\dfrac{\left\|\mathbf{y}' - \mathbf{A}'\mathbf{x}_{(k)}\right\|_2}{\lambda} \dfrac{\left\|\mathbf{x}_{(k)}\right\|_2}{\left\|\mathbf{A}'\mathbf{x}_{(k)}\right\|_2}, \dfrac{\|\mathbf{y}\|_1}{\lambda m}\right) \\
k & \leftarrow & k + 1 \\
\text{Until} & & \text{convergence}
\end{array}
$$

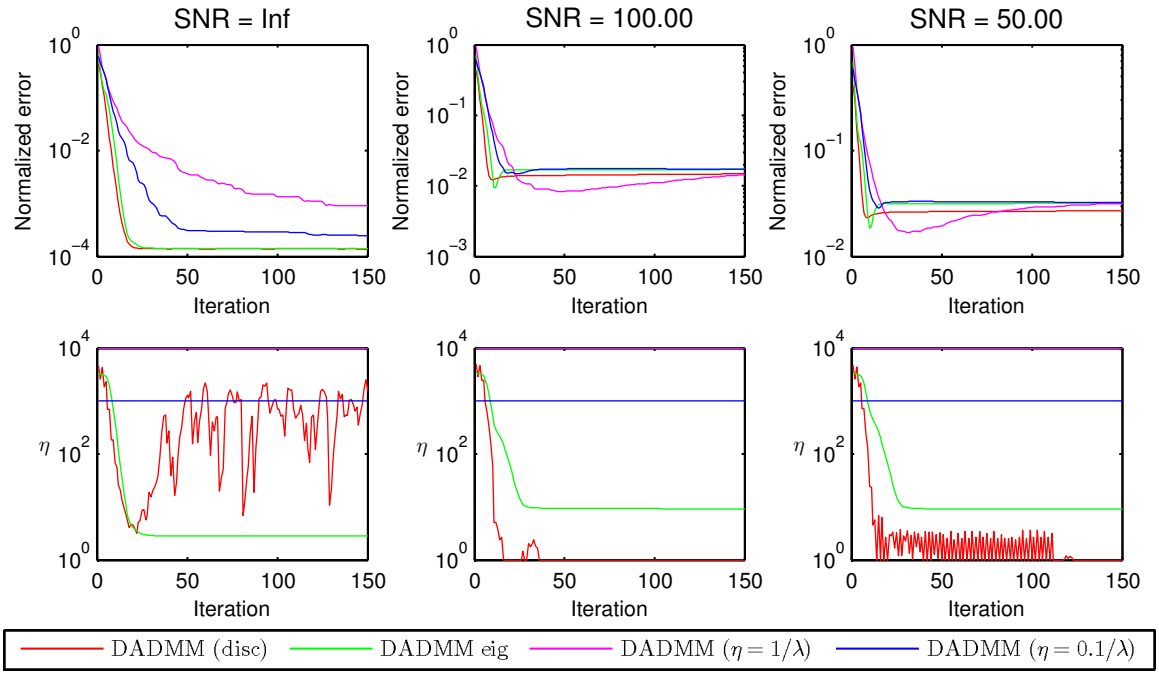
$$
\eta_{(k)} = \min \left(0.5\frac{\left\|\mathbf{y}' - \mathbf{A}'\mathbf{x}_{(k)}\right\|_2}{\lambda} \frac{\left\|\mathbf{x}_{(k)}\right\|_2}{\left\|\mathbf{A}'\mathbf{x}_{(k)}\right\|_2}, \frac{\|\mathbf{y}\|_1}{\lambda m}\right). \tag{4.9}
$$

The term $\left\|\mathbf{x}_{(k)}\right\|_2 / \left\|\mathbf{A}'\mathbf{x}_{(k)}\right\|_2$ compensates for the properties of the dictionary and $\eta_{(k)}$ needs to be bounded from above, since, depending on initialization, an unconstrained update can lead to large η values that may preclude the convergence of the algorithm. The term $\|\mathbf{y}\|_1 / (\lambda m)$ proved to be an upper bound that prevented these situations. The proposed algorithm, here dubbed DADMM-eig, is summarized in algorithm 4.1.

Repeating a part of the analysis done for DADMM in discovery mode and adding DADMM-eig with automatic η update, it can be seen in figure 4.6 that the automatic update of η follows closely the average η of DADMM in the discovery phase, specially in the noiseless and small lambda settings. As a result, DADMM-eig sees the same improvements from updating η in each iteration. Note that the η values for DADMM-eig in the discovery phase after it converges are irrelevant and do not need to be followed by the update rule.



(a)



(b)

Figure 4.6: Performance comparison for DADMM-eig with the proposed update rule. For each subfigure, top row: Average solution error for a sequence of steps of DADMM and DADMM-eig; bottom row: Trace of the average penalty parameter η for DADMM-eig (variable) and DADMM (fixed) for a sequence of steps of the algorithm. a) Plots for different values of λ. b) Plots for different levels of noise. Averaged over 20 trials.

4.3 Experiments

4.3.1 Compressive Sensing

Compressive sensing is the standard example use of sparse coding. The goal is to infer the underlying signal from a limited number of measurements, obtained directly or after the transformation by a sensing matrix. When the signal is transformed by a sensing matrix, each measurement is a linear combination of the elements of the signal, allowing the use of dictionaries with local atoms such as wavelet basis, where direct measurements would not.

The compressive sensing problem is formulated as:

$$\min_{\mathbf{x}} \frac{1}{2} \|\mathbf{y} - \mathbf{SDx}\|_2^2 + \lambda \|\mathbf{x}\|_1 , \tag{4.10}$$

where the dictionary $\mathbf{D}$ is known to represent the underlying signal $\mathbf{z}$ with a sparse vector $\mathbf{x}$, $\mathbf{S}$ is the sensing matrix and $\mathbf{y}$ is the available measurements vector. The dictionary used in this experiment was the complete DCT transform. DADMM is applicable to any dictionary, but the DCT transform was used here for simplicity and because it is known that images have sparse (or at least compressible) representations in the spatial frequency domain. The sensing matrix was filled with values extracted from a standard normal distribution. The image to be recovered was divided in patches of size 32×32 and each patch was transformed by the same sensing matrix. The number of rows of the sensing matrix was swept between 20% of the number of pixels (1024) up to 50%. This ratio is denoted by f in the following experiment. DADMM-eig was analyzed for the image *lena*, tracking the reconstruction error as the time and iterations grow, when solving equation (4.10) with $\lambda = 0.1$. Its performance was compared with DADMM with fixed $\eta = 1/\lambda$ and $\eta = 0.1/\lambda$, under the same conditions. The reconstruction error tracking is shown in figure 4.7, for each DADMM version and undersampling factor. Although in terms of iterations, the improvement

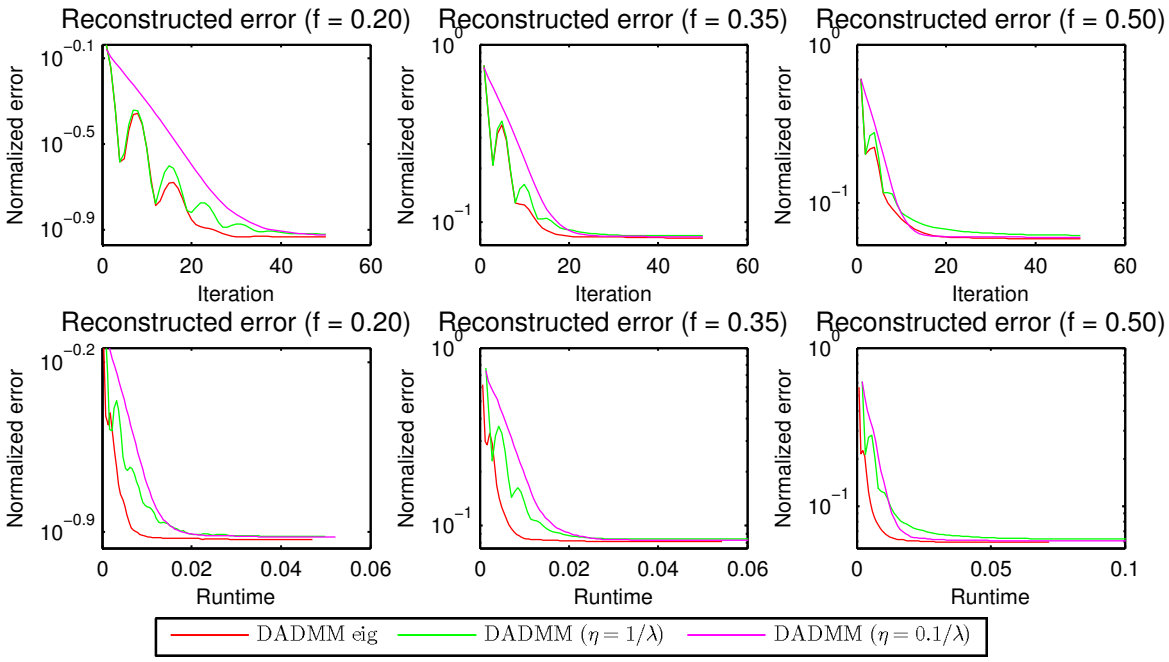


Figure 4.7: Performance of DADMM-eig for compressive sensing of image *lena*. Average reconstruction error of a patch of image *lena* for DADMM-eig and DADMM. Top row: Reconstruction error vs. iteration. Bottom row: Reconstruction error vs. time. Each column represents different number of rows of the sensing matrix.

is not significant, when the runtime is taken into account, DADMM-eig converges faster. The final recovered image for each undersampling ratio, which is identical for any DADMM version, can be seen in figure 4.8.

DADMM-eig was also analyzed for several other images under this experiment, shown in figure 4.9. The procedure applied to image *lena* was also applied to each of these images. In this case, the algorithm ran until the normalized duality gap was below a certain tolerance ε_{dg} or the maximum number of iterations was reached. For each combination of image, number of measurements and DADMM version, the time and number of iterations to reach convergence, as well as the final reconstruction error were measured and reported in table 4.1. $\varepsilon_{dg} = 10^{-3}$, $\lambda = 0.1$ and maximum number of iterations= 100. DADMM-eig meets convergence significantly sooner than DADMM with any fixed η. Naturally, a larger number of measurements reduces the reconstruction error. More surprisingly, the number of iterations reduces with more

Figure 4.8: Final recovered *lena* image for different values of undersampling f. The PSNR value is defined as $20\log_{10}\left({}^{\max(I)}/\sqrt{\sum_{i=1}^{N}(I_i-J_i)^2/N}\right)$, where I and J are the vectorized original and reconstructed images respectively and N is the number of pixels.

(a) Lady (b) House

(c) Baboon (d) Vegetables

Figure 4.9: Image dataset

available measurements, but since each computation involving the dictionary becomes heavier, the outcome is a larger run time. This trend is present in all images.

The setup of the original DADMM involves the precomputation of the inverse of the Hessian, while the setup of DADMM-eig consists in the eigendecomposition of the matrix $\mathbf{A} = \mathbf{SD}$ and computing the term $\mathbf{A}' = \mathbf{U}^T\mathbf{A}$. The setup times are reported in table 4.2 for DADMM-eig and the original DADMM for the same sensing matrix variations used in table 4.1. It can be seen that the setup time is only a small fraction of the time necessary to complete the recovery of an entire image.

4.3.2 Unsupervised Inpainting

Section 1.2.2.2 presented the supervised version of inpainting as an application of sparse coding, in which the location of missing (or painted) pixels are known in advance and excluded from the optimization problem. The unsupervised version of inpainting is not aware of the location of the missing pixels and assumes that they are significantly different from the remaining image. Therefore, the missing pixels can be considered outliers among the image data. Another example of outlier pixels is the salt-and-pepper noise, recognizable as sparsely occurring white and black pixels. An image with outliers $\mathbf{y}$ can be represented as

$$\mathbf{y} = \mathbf{Ax} + \mathbf{e} = \begin{bmatrix} \mathbf{A} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{e} \end{bmatrix}, \tag{4.11}$$

where the term $\mathbf{Ax}$ captures the image without outliers and $\mathbf{e}$ is a vector containing the outliers. As long as matrix $\mathbf{A}$ represents sparsely the image without outliers and the outliers themselves are sparse, the variable $\mathbf{w} = \begin{bmatrix} \mathbf{x} \\ \mathbf{e} \end{bmatrix}$ can be found by solving

Table 4.1: Comparison between DADMM and DAMM-eig for compressive sensing for an image dataset

Normalized reconstruction error, time and number of iterations for convergence until normalized duality gap $< 10^{-3}$, for several images, number of measurements and DADMM versions. Number of iterations per patch were limited to 150.$\lambda = 0.1$.

Image (dimensions)	# measurements	DADMM	Rec error	Time (s)	# Iterations
Lady (256×256)	0.2	eig	0.1702	**1.600**	**6030**
		$\eta = 1/\lambda$	0.1739	4.407	9600
		$\eta = 0.1/\lambda$	0.1724	4.080	9588
	0.35	eig	0.1238	**3.307**	**5165**
		$\eta = 1/\lambda$	0.1258	8.975	9600
		$\eta = 0.1/\lambda$	0.1247	8.606	9573
	0.5	eig	0.0903	**8.109**	**4471**
		$\eta = 1/\lambda$	0.0921	19.585	9600
		$\eta = 0.1/\lambda$	0.0911	35.834	9600
Image (dimensions)	# measurements	DADMM	Rec error	Time (s)	# Iterations
House (256×256)	0.2	eig	0.0966	**2.664**	**7004**
		$\eta = 1/\lambda$	0.0991	4.326	9600
		$\eta = 0.1/\lambda$	0.0981	4.155	9546
	0.35	eig	0.0632	**4.408**	**6590**
		$\eta = 1/\lambda$	0.0649	9.171	9600
		$\eta = 0.1/\lambda$	0.0639	8.521	9511
	0.5	eig	0.0423	**11.500**	**6083**
		$\eta = 1/\lambda$	0.0434	18.799	9600
		$\eta = 0.1/\lambda$	0.0429	30.531	9574
Image (dimensions)	# measurements	DADMM	Rec error	Time (s)	# Iterations
Baboon (512×512)	0.2	eig	0.1888	**10.659**	**27890**
		$\eta = 1/\lambda$	0.1913	16.819	38400
		$\eta = 0.1/\lambda$	0.1900	16.970	38101
	0.35	eig	0.1544	**19.551**	**27463**
		$\eta = 1/\lambda$	0.1566	47.927	38400
		$\eta = 0.1/\lambda$	0.1557	69.746	38149
	0.5	eig	0.1231	**43.418**	**25922**
		$\eta = 1/\lambda$	0.1250	140.894	38400
		$\eta = 0.1/\lambda$	0.1242	165.290	38119
Image (dimensions)	# measurements	DADMM	Rec error	Time (s)	# Iterations
Vegetable (512×512)	0.2	eig	0.0915	**8.956**	**27247**
		$\eta = 1/\lambda$	0.0938	16.266	38400
		$\eta = 0.1/\lambda$	0.0928	16.689	37985
	0.35	eig	0.0640	**25.477**	**25167**
		$\eta = 1/\lambda$	0.0654	57.658	38400
		$\eta = 0.1/\lambda$	0.0646	90.058	38275
	0.5	eig	0.0465	63.129	**23179**
		$\eta = 1/\lambda$	0.0478	91.605	38400
		$\eta = 0.1/\lambda$	0.0470	**52.591**	38177

Table 4.2: Setup times of DADMM and DADMM-eig.

Fraction of measurements	DADMM-eig (s)	DADMM (s)	Ratio
0.2	0.0143	**0.0091**	1.57
0.35	0.0695	**0.0228**	3
0.5	0.1310	**0.0420**	3.12

$$\min_{\mathbf{w}} \|\mathbf{w}\|_1 \qquad \mathbf{y} = \begin{bmatrix} \mathbf{A} & \mathbf{I} \end{bmatrix} \mathbf{w} \tag{4.12}$$

or, more robustly,

$$\min_{\mathbf{w}} \frac{1}{2} \left\| \mathbf{y} - \begin{bmatrix} \mathbf{A} & \mathbf{I} \end{bmatrix} \mathbf{w} \right\|_2^2 + \lambda \|\mathbf{w}\|_1 . \tag{4.13}$$

The image *lena* of size 512×512 containing salt-and-pepper noise was divided in 1024 patches of size 16×16. This problem with $\mathbf{y}$ representing each patch was solved with DADMM and DADMM-eig. The dictionary $\mathbf{A}$ was the complete DCT transform of size 256×256. DADMM ran with $\eta = 1/\lambda$ and $\eta = 0.1/\lambda$, since the optimum penalty parameter changed with the fraction of corrupted pixels f. The algorithms stopped when the relative duality gap was below ε_{dg}. The stopping criterion tolerance was set to $\varepsilon_{dg} = 10^{-4}$ and the regularization parameter to $\lambda = 0.01$.

The results are shown in figure 4.10, with a trace of the reconstruction error throughout iterations and time, for different levels of pixel corruption f =number of corrupted pixels/total number of pixels. This example shows that the improvement obtained by DADMM-eig is not significant in some situations. The proposed method is in advantage only by a small amount. More interestingly, note that the best λ value depends on the number of corrupted pixels in this example: $\eta = 1/\lambda$ clearly works best for $f = 0.3$, but not for $f = 0.1$. On the other hand, the adaptive η has the best results for all levels of image corruption.

The corrupted and reconstructed images for different levels of image corruption is shown in figure 4.11. Even with 30% of corrupted pixels, the image *lena* is mostly

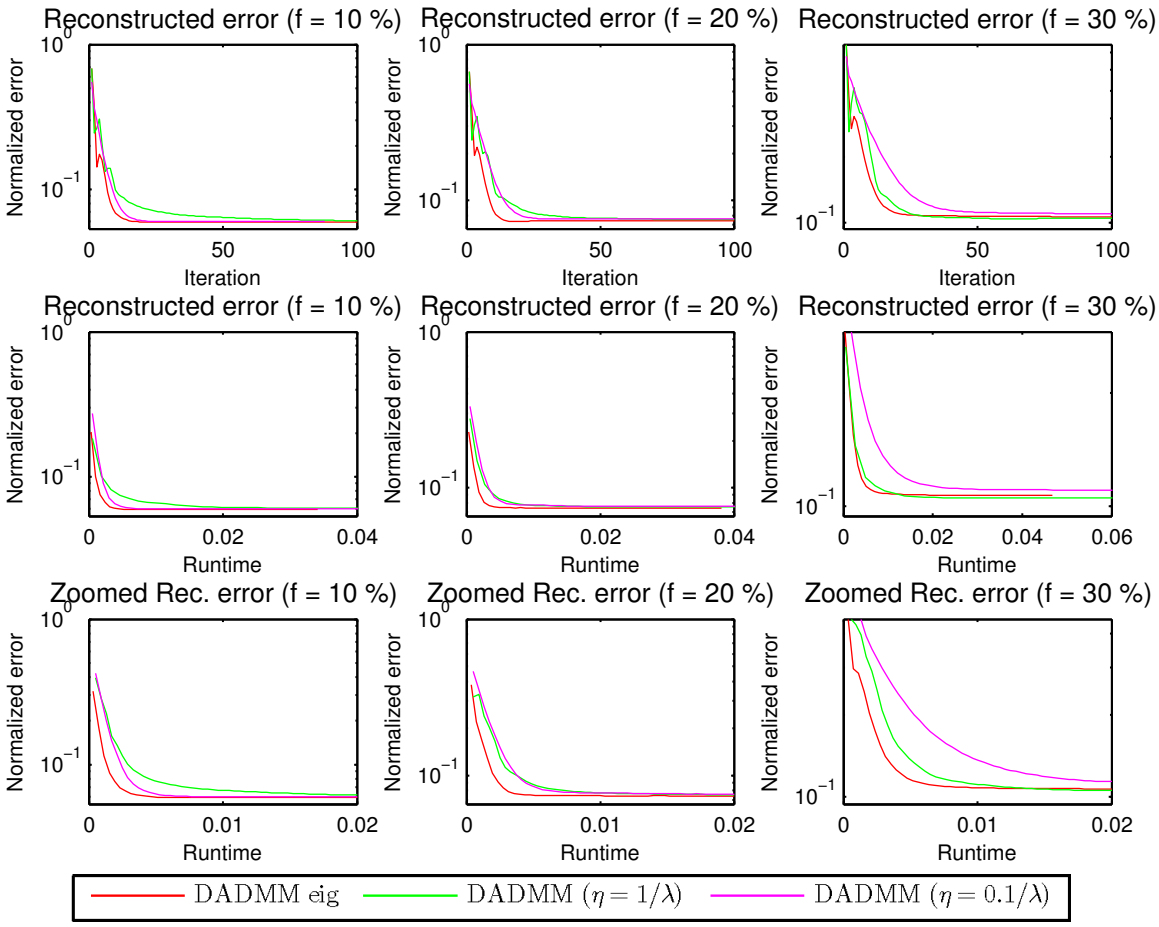


Figure 4.10: Average reconstruction error of a patch of image *lena* for DADMM-eig and DADMM.
Top row: Reconstruction error vs. iteration. Middle row: Reconstruction error vs. time. Bottom row: zoomed area of middle row. Each column represents different fractions of corrupted pixels f.

recovered.

4.4 Better Sparse Recovery From ℓ_1-regularization

Section 3.2 showed that DADMM presents runtimes comparable or even surpassing L0 algorithms, although lagging behind in terms of practical recovery properties. However, some methods based on L1 algorithms show better recovery properties [10, 61] than regular L1 minimization. For instance, Iterative Reweighted ℓ_1-minimization (IRL1) [10] uses a regularization vector instead of a scalar. The L1 minimization is repeated for different values of the regularization vector. In each iteration, the regularization value for each element is set as the inverse of the solution found in the previous iteration plus a positive parameter ε, $\lambda_i^{(k+1)} = \left(x_i^{(k)} + \varepsilon\right)^{-1}$. The ε parameter provides stability and ensures that a zero-valued element in $\mathbf{x}_{(k)}$ does not prohibit a non-zero estimate at the next step. Thus, the sparsity measure at iteration $k+1$ is:

$$\sum_i \left|\lambda_i^{(k+1)} x_i^{(k+1)}\right| = \sum_i \left|\frac{x_i^{(k+1)}}{x_i^{(k)} + \varepsilon}\right|. \tag{4.14}$$

When the iterations converge, the sparsity measure approximates the ℓ_0-norm.

Threshold-Iterative Support Detection (threshold-ISD) [61] presents recovery performance similar to IRL1, although taking a different approach. Like IRL1, threshold-ISD runs ℓ_1-norm based minimization during several iterations until convergence. Unlike IRL1, threshold-ISD changes the support of the solution accounting for the sparsity measure. After an iteration finds a solution with support T, a subset $I \subset T$ is selected by thresholding and the next minimization problem is the truncated Basis-Pursuit:

$$\min_{\mathbf{x}} \|\mathbf{x}_{I^C}\|_1 \qquad S.T. \quad \mathbf{A}\mathbf{x} = \mathbf{y}, \tag{4.15}$$

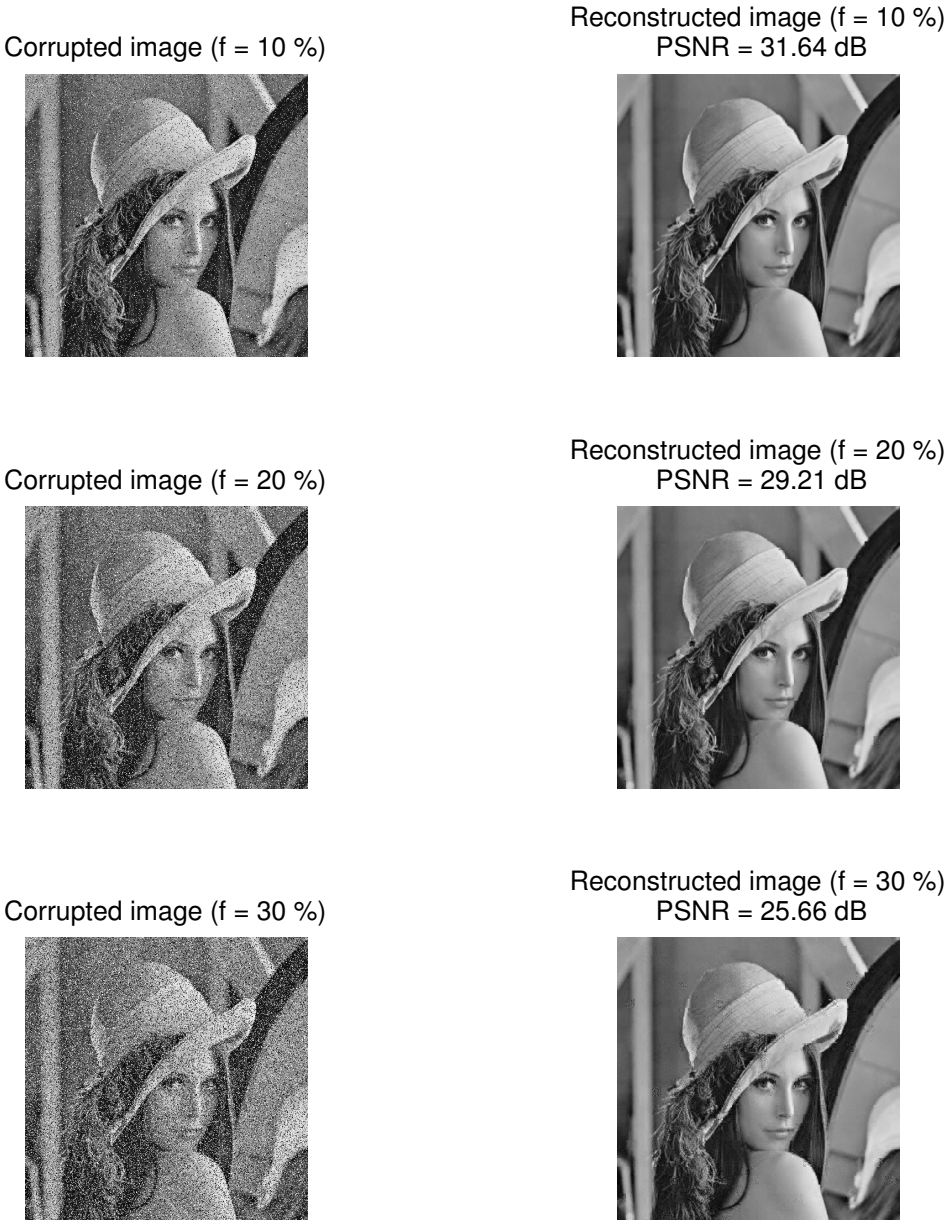


Figure 4.11: Final recovered *lena* image for different levels of corrupted pixels. The PSNR value is defined as $20\log_{10}\left({}^{\max(I)}/\sqrt{\sum_{i=1}^{N}(I_i-J_i)^2/N}\right)$, where I and J are the vectorized original and reconstructed images respectively and N is the number of pixels.

where I^C is the complement support of I. The algorithm rapidly converges to a situation where the zero elements contribute to the sparsity measure, but elements that were confirmed to be part of the solution with high probability are excluded from it. In particular, since the intermediate iterations do not need to reach high precision and may end early, [61] shows that threshold-ISD can take the same time to solve all iterations as a standard L1 minimization.

DADMM-eig supports the variations proposed in IRL1 and threshold-ISD and can be used to speed-up these algorithms as well. It would be interesting to evaluate how IRL1 or threshold-ISD compare to greedy L0 algorithms in terms of practical recovery properties.

4.5 Other Forms of ℓ_1-norm Based Optimization

The exploration of sparse coding often leads to formulations that are not strictly as the one presented in (1.5). It thus begs the question: in what other formulations DADMM-eig can be applied? DADMM-eig relies on the fact that the terms in the Hessian in (2.37) are fixed, with the exception of the penalty parameter, and that a change in the penalty parameter would require the recomputation of the inverse of the Hessian. Instead, the eigendecomposition done in DADMM-eig is done only once and reused in every iteration.

In the noiseless formulation (1.4), the approach taken in DADMM-eig does not bring any advantage. The dual of the noiseless case is:

$$\begin{aligned} \max_{\alpha} \quad & \mathbf{y}^T\alpha \\ S.T. \quad & \nu = \mathbf{A}^T\alpha \\ & \|\nu\|_\infty \leq 1. \end{aligned} \tag{4.16}$$

In this case, the second derivative of the dual objective function is 0, which leads to an Hessian of $\eta\mathbf{A}\mathbf{A}^T$ for the α update. It is clear that the penalty parameter can be changed freely without the need for DADMM-eig.

The robust sparse model

$$\min_{\mathbf{x}} \|\mathbf{y}-\mathbf{A}\mathbf{x}\|_1 + \lambda \|\mathbf{x}\|_1 \tag{4.17}$$

handles non-gaussian noise much better than the standard ℓ_1-norm minimization. It can be reformulated as:

$$\begin{aligned} \min_{\mathbf{x},\mathbf{r}} \|\mathbf{r}\|_1 + \lambda \|\mathbf{x}\|_1 \quad S.T. \quad \mathbf{r} = \mathbf{y}-\mathbf{A}\mathbf{x} \\ \min_{\hat{\mathbf{x}}} \|\hat{\mathbf{x}}\|_1 \quad S.T. \quad \hat{\mathbf{y}} = \hat{\mathbf{A}}\hat{\mathbf{x}}, \end{aligned} \tag{4.18}$$

where

$$\hat{\mathbf{x}} = \begin{bmatrix} \lambda\mathbf{x} \\ \mathbf{r} \end{bmatrix}, \hat{\mathbf{y}} = \frac{\lambda\mathbf{y}}{\sqrt{1+\lambda^2}}, \hat{\mathbf{A}} = \frac{\begin{bmatrix} \mathbf{A} & \lambda\mathbf{I} \end{bmatrix}}{\sqrt{1+\lambda^2}}. \tag{4.19}$$

Therefore, this problem can be solved with the dual formulation (4.16), with no improvement from DADMM-eig.

However, problems that change the sparsity measure or include additional constraints to the solution can benefit from DADMM-eig. For example, when non-negativity constraint is added to the solution, (1.5) is changed to:

$$\begin{aligned} \min_{\mathbf{x}} \quad & \frac{1}{2}\|\mathbf{y}-\mathbf{A}\mathbf{x}\|_2^2 + \lambda\|\mathbf{x}\|_1 \\ S.T. \quad & \mathbf{x} \geq 0. \end{aligned} \tag{4.20}$$

The dual of this problem is:

$$\begin{aligned} \max_{\alpha} \quad & -\frac{1}{2}\left\|\alpha\right\|_2^2 + \alpha^T \mathbf{y} \\ S.T. \quad & \nu = \mathbf{A}^T\alpha \\ & \nu \leq \mathbf{1}\lambda. \end{aligned} \tag{4.21}$$

As a consequence, this will only change the projection and proximal operators. The Hessian is kept unchanged and DADMM-eig can improve convergence and speed-up. In general terms, any function $\Phi(\mathbf{x})$ that measures sparsity only affects the projection and proximal operators of DAMM-eig, since the Hessian of the Lagrange w.r.t. α is only defined by the objective function and the equality constraint. Therefore, any primal problem defined as:

$$\min_{\mathbf{x}} \frac{1}{2}\left\|\mathbf{y} - \mathbf{A}\mathbf{x}\right\|_2 + \Phi(\mathbf{x}) \tag{4.22}$$

has a dual formulation

$$\begin{aligned} \max_{\alpha} \quad & -\frac{1}{2}\left\|\alpha\right\|_2^2 + \alpha^T \mathbf{y} \\ S.T. \quad & \nu = \mathbf{A}^T\alpha \\ & \phi(\nu). \end{aligned} \tag{4.23}$$

for some correspondence between the sparsity measure $\Phi(\mathbf{x})$ and the constraint $\phi(\nu)$. Some examples are shown in table 4.3. Group sparsity [5] motivates sparsity in groups of elements of the solution. For disjoint groups $G_1, ..., G_N$, the final solution tends to aggregate non-zero elements in the same group. The group structure is defined beforehand and its dependent on the dictionary. The use of a regularization vector instead of a scalar is also relevant, for example for IRL1 [10], threshold-ISD [61] or

Table 4.3: Relation between primal function for sparsity measure and dual constraint.

Sparsity Measure	$\Phi(\mathbf{x})$	$\phi(\nu)$
ℓ_1-norm	$\lambda \sum_i \lvert x_i \rvert$	$\lVert \nu \rVert_\infty \leq \lambda$
Group sparsity	$\lambda \sum_{i \in 1..N} \lVert \mathbf{x}_{G_i} \rVert_2$	$\lVert \nu_{G_i} \rVert_2 \leq \lambda, i = 1, ..., N$
Element-wise regularization	$\sum_i \lvert \lambda_i x_i \rvert$	$\lvert \nu_i \rvert \leq \lambda_i$

generally if some elements of the solution are more important to be inactive than others.

4.6 Conclusions

In this section, an adaptation to DADMM is proposed that transforms the matrix to be inverted into a diagonal matrix. As a result, the penalty parameter can be changed at no extra cost. In addition, the computational complexity of one of the matrix-vector multiplications is simplified, since the vector becomes sparse. A thorough analysis of the best sequence for the penalty parameter was done and it was shown that the proportionality to the norm of the residual at each iteration is an important factor. The improvement of adapting the penalty parameter was reduced under large regularization parameter and/or low signal-noise-ratio of the observations. Real data experiments under these circunstances still showed improvement in the runtime, in particular when measuring the runtime until the duality gap was below a certain tolerance. The setup time is larger in DADMM-eig than in DADMM, but since it is performed only once, it had a small expression when recovering many patches of an entire image.

The use of the eigendecomposition is prohibitive for very large dictionaries, in the order of tens to the hundreds of thousands of measurements. We mention measurements because the eigendecomposition in DADMM-eig is done on a square matrix with each side with dimensions equal to the number of measurements. The number of atoms can be extended with no effect on the eigendecomposition. Therefore,

DADMM-eig cannot be used, for example, in the sparse coding of an entire image or in the recovery of a Computerized Tomography (CT) or Magnetic Resonance Imaging (MRI) exam. DADMM-eig is better suited for medium-sized problems, such as dictionaries obtained from dictionary learning algorithms, which are themselves limited; or when a high-dimensional signal is mapped to a low-dimensional feature space. In particular, wafer probe testing, introduced in the next chapter, contains data with thousands of measurements, well within DADMM-eig range. Virtual Probe is a sparse coding based algorithm that aims to reduce wafer test cost.

Chapter 5

Accelerating Virtual Probe with DADMM

5.1 Integrated Circuits Test Cost Reduction

Large-scale parametric variations in nanoscale transistors have made analog/RF circuits increasingly difficult to design and test. The performance of an analog/RF circuit can substantially vary from lot to lot, from wafer to wafer, and from die to die. For this reason, analog/RF circuit testing has contributed to a large, or even dominant, portion of the overall test cost for today's complex systems on chip (SOCs) [15, 2]. To reduce the test cost and, consequently, the manufacturing cost of analog/RF circuits, a large number of algorithms and methodologies have been extensively studied over the past decade [58, 52, 59, 51, 53, 69]. Among them, spatial variation modeling [39, 73, 71, 14, 72, 29, 35, 36, 30, 31] has emerged as a promising technique for wafer-level test cost reduction in recent years. The key idea is to accurately capture the spatial variation pattern of an entire wafer by using advanced statistical algorithms, such as Virtual Probe (VP) [39, 73, 71, 14, 72, 29] and Gaussian process (GP) [35, 36, 30, 31], over only a small subset of dies. In particular, Virtual

Probe uses sparse reconstruction as the statistical method.

In more detail, a wafer comprises a large number of dies, each one with the same set of test items. A test item is a metric, such as power consumption, bandwidth, operating points, slew-rate, or any other electrical measurement, that must be within certain bounds for the die to pass the test item. A die must pass all test items to be considered within specification. Given the large number of test items and dies in a complete wafer, wafer probing for testing purposes is a time-consuming process. This, in turn, creates a bottleneck in production, which can be mitigated by effectively reducing the number of dies that need to be measured. Given that some test items present a predictable spatial pattern, the complete pattern can be inferred from a subset of dies.

5.2 Virtual Probe

Virtual Probe was proposed in [39] and further improved in [72]. It infers the unmeasured dies by sparse reconstruction, assuming that the pattern is sparse under the frequency domain. Figure 5.1 depicts the concept behind VP: for a given test item, a selected subset of dies is measured from the wafer; these measurements are then used to estimate the coefficients in the frequency domain by sparse reconstruction; finally, the spectral coefficients are used to estimate all measurements in the wafer. The quality of the reconstruction is defined by the sparsity of the spectral coefficients and the number of measurements used for reconstruction.

Formally, the test item i at the location j of the wafer is denoted as y_j^i. The complete collection of measurements of the same test item in a wafer is represented as a vector $\mathbf{y}^i$. The set of spatial locations of measured dies for test item i is denoted as M_i. Let us denote $\mathbf{x} = \mathcal{D}(\mathbf{y})$ as the fast 2-D Discrete Cosine Transform (DCT) transform, $\mathbf{y} = \mathcal{D}^{-1}(\mathbf{x})$ as the fast 2-D inverse DCT transform, while $\mathbf{y}_M = \mathcal{D}_M^{-1}(\mathbf{x})$

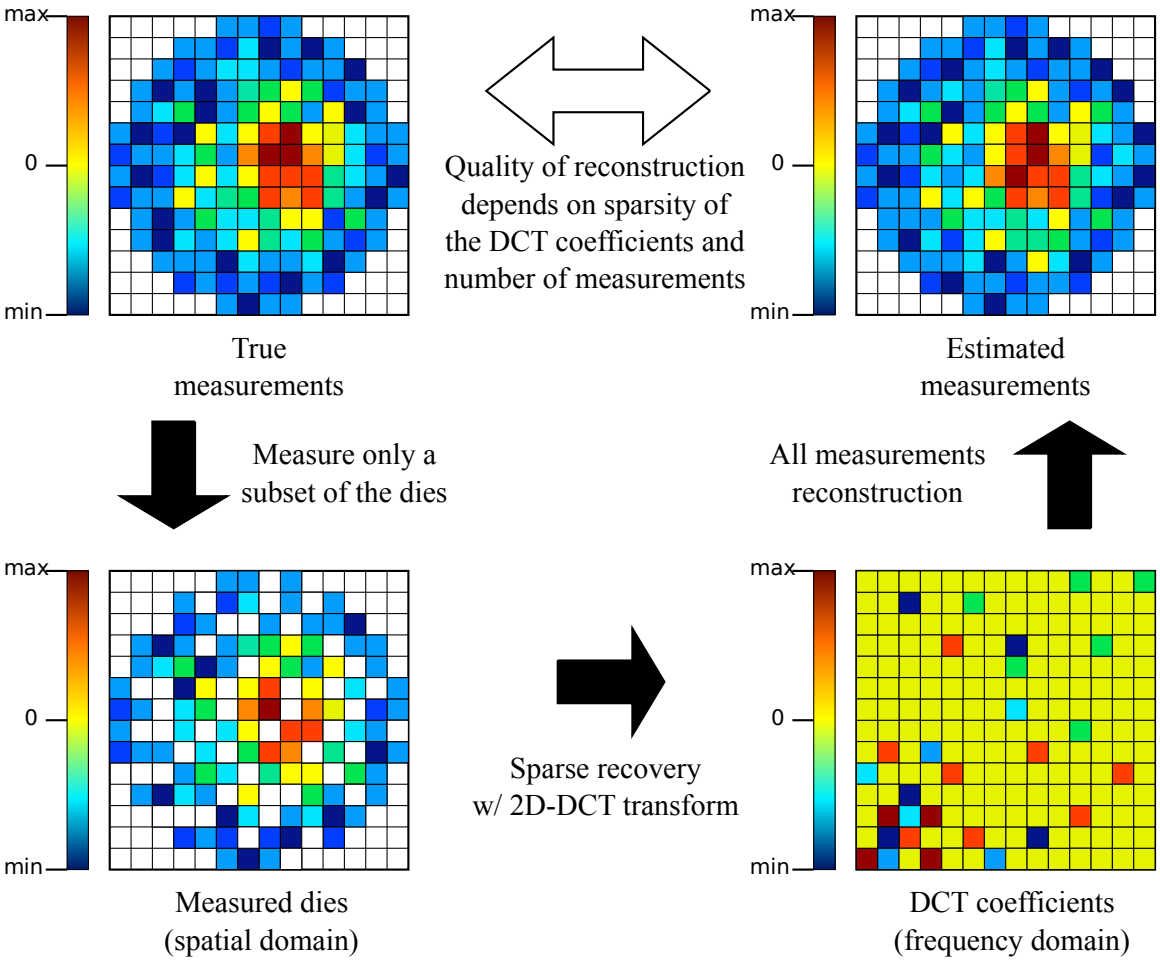


Figure 5.1: Overview of Virtual Probe.
For a given test item, only a subset of dies is effectively measured from the available wafer measurements. Those measurements are used for sparse reconstruction of 2D-DCT coefficients corresponding to the complete wafer by sparse recovery. The unmeasured dies are then predicted by reconstruction of all wafer measurements from the 2D-DCT coefficients. The quality of the reconstruction is mostly dependent on the sparsity of the coefficients when the transform is applied to the full wafer measurements and the number of measurements used for sparse recovery.

is equivalent to $\mathcal{D}^{-1}(\mathbf{x})$ with only the result of the indices in set M. In what follows, assume m dies are measured out of a total of n dies.

Virtual Probe works under the premise that the measurements obtained from a test item i can be expressed as a linear combination of a small set of DCT coefficients, i.e.:

$$\mathbf{y}^i \approx \mathcal{D}^{-1}(\mathbf{x}^i), \tag{5.1}$$

where $\mathbf{x}^i$ is sparse. For test items that fulfill this premise, compressive sensing theory dictates that the solution of

$$\hat{\mathbf{x}}^i = \arg\min_{\mathbf{x}} \frac{1}{2} \left\| \mathbf{y}^i_{M_i} - \mathcal{D}^{-1}_{M_i}(\mathbf{x}) \right\|_2^2 + \lambda \left\| \mathbf{x} \right\|_1 \tag{5.2}$$

will be close to $\mathbf{x}^i$. From here, the values of the test item i at all locations can be recovered by $\hat{\mathbf{y}}^i = \mathcal{D}^{-1}(\hat{\mathbf{x}}^i)$.

Regarding the use of the 2D-DCT matrix as dictionary, other options cannot be used for different reasons. On one hand, since the process parameters shift throughout the wafers in the same lot, there is the possibility of overfitting when very specific dictionaries, such as template based dictionaries, are used. On the other hand, dictionaries with local atoms, such as wavelet transforms, could not be used either, since the local atoms that match the location of the measured dies, and therefore are part of solution, would not contain sufficient information in the location of the unmeasured dies. Therefore, the use of a global and general dictionary, such as the 2D-DCT matrix, is a convenient choice.

Any prediction is associated with a level of accuracy, in this case defined by the sparseness of the coefficients $\mathbf{x}^i$ and the number of measured dies, according to the compressive sensing theory. Since the coefficients are defined by the wafer spatial pattern, the only control "knob" that VP has to improve accuracy is the number of measured dies. As depicted in figure 5.2, the prediction error, defined by Yield Loss

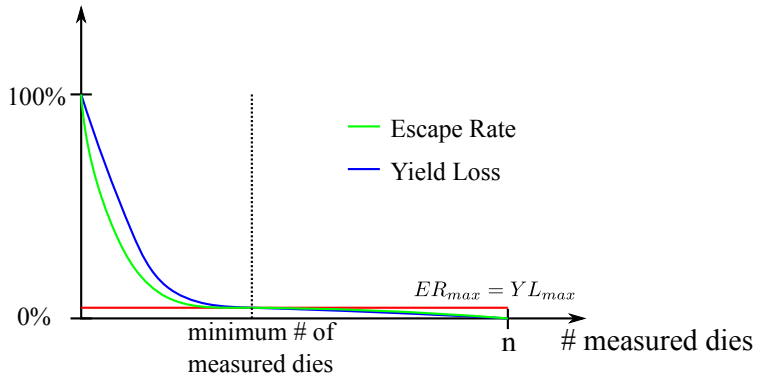


Figure 5.2: Number of measured dies limited from below by Yield Loss and Escape Rate.

(YL) and Escape Rate (ER), reduces as the number of measured dies is increased. YL is the percentage of dies that are predicted to fail, but actually pass the specification, while ER is the percentage of dies that are predicted to pass, but in reality fail the specification. For a pre-defined target of YL and ER, there is a minimum number of measurements required for each test item.

Accuracy is also affected by defective dies, i.e., dies that are damaged and produce erratic or no measurements. Obviously, if these dies are not measured, they cannot be predicted correctly, since they are random in nature. However, defective dies can be detected: for example, prior to VP, fast tests can be run in all dies to simply test if a die is faulty. As will later be seen, some test items do not have any predictable spatial structure and therefore need to be measured in all dies anyway. Therefore, these tests can also be used to detect defective dies.

A general convex solver requires the use of the 2D-DCT matrix instead of the 2D-DCT transform. In that case, the 2D-DCT transform is replaced by the corresponding orthogonal matrix $\mathbf{D}^T$ and its inverse by $\mathbf{D}$. An interior-point solver may take a few minutes to recover the spatial variation pattern for a single test item of a single wafer. As a result, it substantially slows down the testing process and poses a major limitation of the VP-based method for test cost reduction. The fact that Virtual Probe uses a partial DCT gives it unique properties that can be exploited by a solver

such as DADMM in order to improve its performance. Namely, the overcompleteness of the partial DCT favors the dual formulation of equation (5.2), the orthogonality of its rows simplifies the closed-form expression of the dual variable, and the use of a fast DCT transform to efficiently calculate matrix-vector multiplications within the iteration loop of DADMM further reduces the computational cost. The next section presents a detailed explanation of these properties and how they can be exploited. A comparison is done between a standard Interior Point Method (IPM) [33] and DADMM, after the suggested alterations are implemented. The results section shows that adopting DADMM to solve equation (5.2) can significantly reduce computational time, and therefore, test time, without loss of accuracy compared with IPM.

5.3 DADMM Applied to Virtual Probe

As discussed in section 2.6, DADMM solves the dual problem of equation (5.2), where the number of variables is proportional to the number of measurements, instead of the number of coefficients. In the problem set by VP, the number of measurements is a fraction of the total number of dies. Since the dimension of the Hessian is the square of the number of variables, the Hessian is much smaller than compared to solving the primal version.

Still concerning the Hessian, let us first state the following equivalences between the 2D-DCT transforms and their explicit matrices:

$$\begin{aligned} \mathcal{D}(\mathbf{y}) &\equiv \mathbf{D}^T\mathbf{y} \\ \mathcal{D}^{-1}(\mathbf{x}) &\equiv \mathbf{D}\mathbf{x}. \\ \mathcal{D}_M^{-1}(\mathbf{x}) &= \mathbf{D}^M\mathbf{x} \end{aligned} \tag{5.3}$$

Matrices $\mathbf{D}$ and $\mathbf{D}^T$ are square matrices containing the DCT basis, while $\mathbf{D}^M$ is a subselection of rows of $\mathbf{D}$. To generate $\mathbf{D}^M$, the rows corresponding to the unmeasured dies are removed from $\mathbf{D}$, which destroys the orthogonality of its columns, but not the orthogonality of its rows. Second-order methods that solve the primal problem have the term $\left(\mathbf{D}^M\right)^T \mathbf{D}^M \in \mathbb{R}^{n\times n}$ in their Hessian, which results in a non-diagonal dense matrix. On the other hand, as seen in DADMM, second-order methods that solve the dual problem have the term $\mathbf{D}^M \left(\mathbf{D}^M\right)^T \in \mathbb{R}^{m\times m}$ in their Hessian, which is indeed equal to the identity matrix, regardless of the set M. This property makes the α update in equation (2.37) computed by DADMM much simpler:

$$\begin{aligned}
\alpha_{(k+1)} &= \left(\mathbf{I} + \eta\mathbf{D}^M \left(\mathbf{D}^M\right)^T\right)^{-1} \left(\mathbf{y}_M - \mathbf{D}^M \left(\mathbf{x}_{(k)} - \eta\nu_{(k)}\right)\right) \\
&= \left(\mathbf{I} + \eta\mathbf{I}\right)^{-1} \left(\mathbf{y}_M - \mathbf{D}^M \left(\mathbf{x}_{(k)} - \eta\nu_{(k)}\right)\right) \\
&= \frac{1}{1+\eta} \left(\mathbf{y}_M - \mathbf{D}^M \left(\mathbf{x}_{(k)} - \eta\nu_{(k)}\right)\right) \qquad (5.4)
\end{aligned}$$

Furthermore, the matrix $\mathbf{D}^M$ is only involved in the operations $\left(\mathbf{D}^M\right)^T \mathbf{y}_M$ and $\mathbf{D}^M\mathbf{x}$, therefore the explicit matrix is not really necessary. Since $\mathbf{D}^M$ represents the partial DCT transform, we have the equivalencies:

$$\begin{aligned}
\mathcal{D}_M\left(\mathbf{y}_M\right) &\equiv \left(\mathbf{D}^M\right)^T \mathbf{y}_M \\
\mathcal{D}_M^{-1}\left(\mathbf{x}\right) &\equiv \mathbf{D}^M\mathbf{x}. \qquad (5.5)
\end{aligned}$$

The transforms have time complexity $\mathcal{O}(n \cdot \log n)$, while the matrix-vector multiplications are done in $\mathcal{O}(mn)$. Above a certain m/n ratio, the transforms are faster compared to the matrix-vector multiplications. The next section describes how the equivalencies in equation (5.5) are implemented.

5.3.1 Applying Partial Transforms

The transforms $\mathcal{D}_M(\mathbf{y}_M)$ and $\mathcal{D}_M^{-1}(\mathbf{x})$ are partial in the sense that $\mathcal{D}_M^{-1}(\mathbf{x})$ only returns the elements indexed by M and $\mathcal{D}_M(\mathbf{y}_M)$ only uses the DCT basis indexed by M. The matrix $\mathbf{D}^M$ is created by down-sampling the full 2-D IDCT matrix where only the rows corresponding to the spatial locations in M are chosen. To compute the matrix-vector multiplication $\mathbf{y}_M = \mathbf{D}^M\mathbf{x} \equiv \mathcal{D}_M^{-1}(\mathbf{x})$, we can first compute $\mathbf{y} = \mathcal{D}^{-1}(\mathbf{x})$ by the fast 2-D IDCT transform and then select the corresponding rows of $\mathbf{y}$ to form the vector $\mathbf{y}_M$.

Likewise, the matrix $\left(\mathbf{D}^M\right)^T$ is created by down-sampling the full 2-D DCT matrix where only the columns corresponding to the spatial locations in M are selected. To compute the matrix-vector multiplication $\mathbf{x} = \left(\mathbf{D}^M\right)^T \mathbf{y}_M \equiv \mathcal{D}_M(\mathbf{y}_M)$, we need to first create a vector $\mathbf{v}$

$$\begin{aligned} \mathbf{v}_M &= \mathbf{y}_M \\ \mathbf{v}_{\tilde{M}} &= \mathbf{0}, \end{aligned} \tag{5.6}$$

where $\mathbf{v}_{\tilde{M}}$ represents the elements $\{v_i : i \notin M\}$. Namely, for the indices belonging to the set M, the corresponding elements of $\mathbf{v}$ are equal to the vector $\mathbf{y}_M$. The remaining indices of $\mathbf{v}$ are filled with zeros. Next, we apply the fast 2-D DCT transform $\mathbf{x} = \mathcal{D}(\mathbf{v})$.

The complete algorithm is summarized in algorithm 5.1.

5.4 Test Flow

The various test items required for wafer probing impose some restrictions in the prediction method used in Virtual Probe. For instance, some test items simply do not have spatial correlation and cannot be predicted. Different test items may require

Algorithm 5.1 DADMM for Virtual Probe

$$
\begin{aligned}
&\text{Initialize} && k = 0, \nu_{(0)}, \mathbf{x}_{(0)} \text{ and } \eta \\
&\text{Repeat} && \\
\alpha_{(k+1)} &= && \frac{1}{1+\eta}\left(\mathbf{y}_M - \mathcal{D}_M^{-1}\left(\mathbf{x}_{(k)} - \eta\nu_{(k)}\right)\right) \\
\nu_{(k+1)} &= && P_\lambda^\infty\left(\frac{\mathbf{x}_{(k)}}{\eta} + \mathcal{D}_M^{-1}\left(\alpha_{(k+1)}\right)\right), \\
\mathbf{x}_{(k+1)} &= && \text{shrink}_{\lambda\eta}\left(\mathbf{x}_{(k)} + \eta\mathcal{D}_M^{-1}\left(\alpha_{(k+1)}\right)\right) \\
k &\leftarrow && k+1 \\
&\text{Until} && \text{convergence}
\end{aligned}
$$

a different number of measurements to prompt accurate wafer map reconstruction. Even for predictable items, the process parameters drift during production and a test item may reduce its spatial correlation between wafers in the same lot. As a consequence, the quality of reconstruction has to be checked for each test item and each wafer.

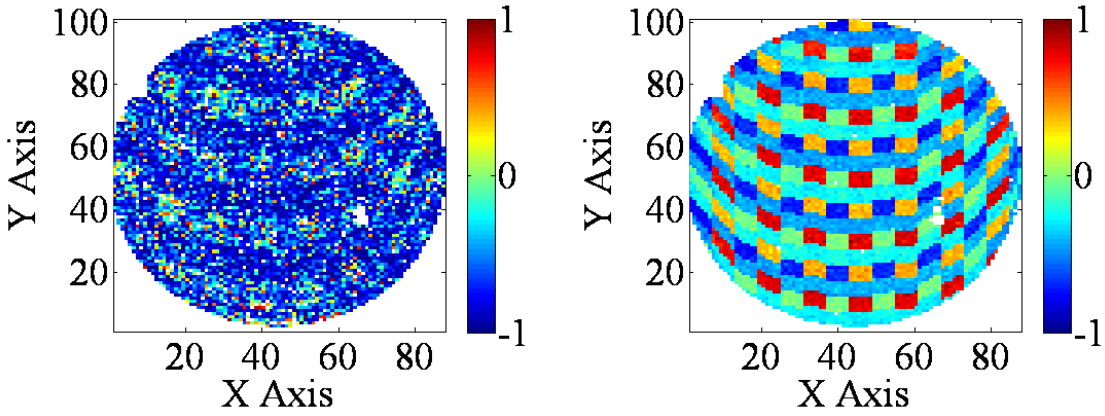
To apply VP for test cost reduction, the same test flow that was proposed in [70] is adopted here. It consists of two major steps: (i) pre-test analysis, and (ii) test application. During pre-test analysis, all test items of all dies on a first wafer are physically measured. Based on the measurement data, the goal of this step is to determine whether a test item is spatially correlated, and therefore predictable, at the wafer level. The test item is considered to be "predictable" if a strong spatial correlation is observed. If that is the case, the number of dies that should be measured on a wafer to accurately predict the spatial wafer map associated with the test item by using VP is decided. As such, the test item will only be physically measured at a number of spatial locations and its values at other unmeasured locations are estimated by VP with sufficiently small escape rate and yield loss.

During test application, firstly all test items at a subset of dies are measured, where the number of these dies is determined by pre-test analysis. Next, for the

predictable test items, VP is applied to estimate their values for other unmeasured dies on the same wafer. Escape rate and yield loss are closely monitored during the prediction process. If the escape rate or the yield loss exceeds a pre-defined target for a specific test item, the test item is temporarily labelled as "unpredictable" for the current wafer. Finally, the unpredictable test items are physically measured for the remaining dies on the wafer. Here, a test item is considered to be unpredictable, if it is classified as an unpredictable item during pre-test analysis or its escape rate or yield loss is not sufficiently small for the current wafer. In either case, the unpredictable test item must be measured for all dies on the current wafer. More details can be found in [70].

5.5 Experiment Setup

To validate the efficiency of the proposed solver for test cost reduction, a set of wafer probe measurement data of an industrial RF transceiver is used. The data set is collected from 1,190,816 dies distributed over 176 wafers and 9 lots where each wafer contains 6,766 dies. For each die, 51 test items are considered in the experiment. For testing and comparison purposes, two different solvers are implemented for VP and applied to the same test flow: (i) the conventional interior-point solver for large-scale L1-regularized least-squares (IPM) [33] and (ii) the DADMM solver (Algorithm 5.1). All numerical experiments are performed on a Linux server with 3.4 GHz CPU and 16 GB memory.



(a) (b)

Figure 5.3: Spatial variations (normalized) for spatially uncorrelated and correlated test items.
(a) test item #1 that is spatially uncorrelated, and (b) test item #48 that is spatially correlated.

5.6 Results

5.6.1 Spatial Variation Modeling

Figure 5.3 shows the spatial variations for two different test items. Studying figure 5.3 reveals an important observation that some test items may not be spatially correlated (e.g., test item #1), while others carry strong spatial correlations (e.g., test item #48). In turn, it demonstrates the importance of the pre-test analysis described in section 5.4 where the objective is to identify the spatially correlated test items for test cost reduction.

Figure 5.4 compares the modeling error for IPM and DADMM. In this experiment, the modeling error for a test item i is defined as:

$$error_i = \frac{\|\hat{\mathbf{y}}^i - \mathbf{y}^i\|_2}{\|\mathbf{y}^i\|_2}, \tag{5.7}$$

where $\mathbf{y}^i$ and $\hat{\mathbf{y}}^i$ denote the actual and predicted values of the i-th test item respectively. Figure 5.5 further shows the spatial variations predicted by IPM and DADMM

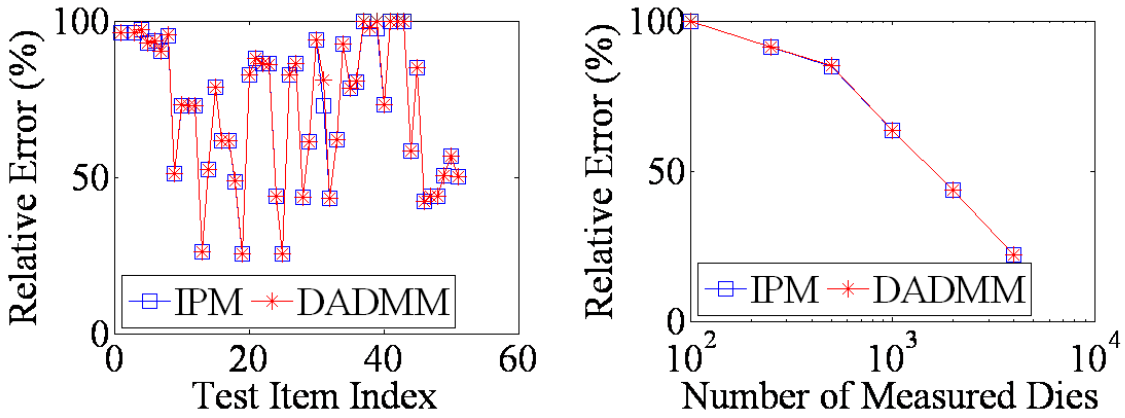
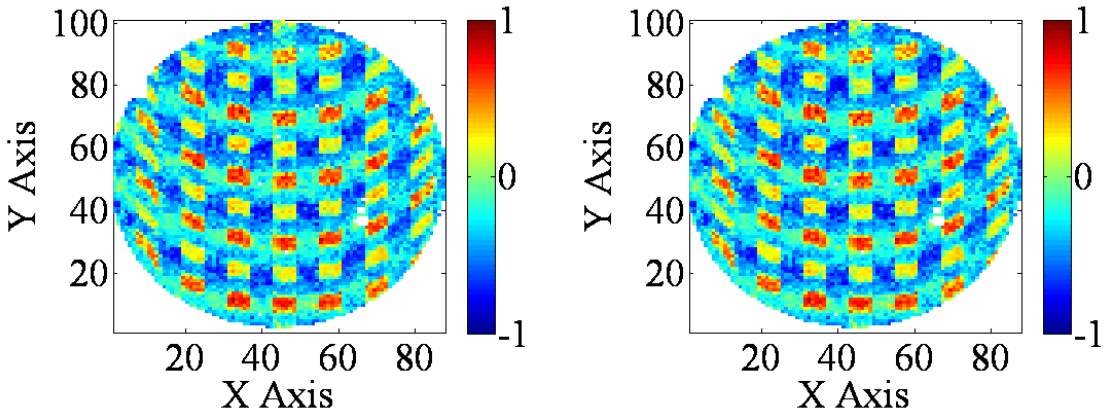


Figure 5.4: Modeling error is compared between IPM and DADMM (a) all test items where 2000 dies are physically measured on a wafer, and (b) test item #48 where the number of physically measured dies varies from 100 to 4000.

for test item #48. Note that the results in figure 5.4 and figure 5.5 are almost identical for both solvers, implying that the proposed DADMM solver is as accurate as the conventional IPM.

Table 5.1 compares the computational time for IPM and DADMM. Based on the results in table 5.1, two important observations can be made. First, DADMM achieves up to 38× runtime speed-up over IPM. Hence, DADMM is the preferred method for this test application where the spatial variations must be accurately estimated in real time during the testing process. Second, and more interestingly, the computational time of DADMM decreases with the number of measured dies (i.e., N). As N becomes larger, the underdetermined linear equation in (5.1) is better constrained and DADMM requires less number of iterations to converge. Since the applied fast transforms do not depend on the number of measurements, the computational time is reduced in proportion, as shown in table 5.1.



(a) (b)

Figure 5.5: Spatial variations (normalized) are predicted by IPM and DADMM (a) IPM and (b) DADMM for test item #48 where 2000 dies are physically measured on a wafer.

Table 5.1: Computational time for IPM and DADMM to model the spatial variation of a single test item for a certain wafer.

Number of measured dies	IPM	DADMM		
	Runtime (s)	Runtime (s)	# of iterations	Speed-up
100	48.3	12.2	7027	3.96
250	62.7	10.3	5664	6.07
500	84.7	8.9	5083	9.52
1000	119.9	8.1	4504	14.88
2000	171.2	7.3	3922	23.56
4000	255.2	6.7	3580	37.86

Table 5.2: Pre-test analysis results for IPM/DADMM

Pre-defined Target for Escape Rate	5×10^{-3}
Pre-defined Target for Yield Loss	5×10^{-3}
Number of Measured Dies (N) per Wafer	2000
Number of Predictable Test Items	38

5.6.2 Test Application

Table 5.2 summarizes the setup for the pre-test analysis and the corresponding results. Since both IPM and DADMM give the same results in this example, we do not explicitly distinguish these two solvers in table 5.2.

Two important clarifications should be made here. First, a relatively large escape rate is allowed, because the "escaped" dies from wafer probe testing can be further captured during the final test after packaging. Second, most of the test items (i.e., 38 items) are considered to be predictable during our pre-test analysis. During test application, the escape rate and the yield loss are further monitored for these 38 test items that are predictable. If the escape rate or the yield loss of a test item exceeds the pre-defined target in table 5.2 for a specific wafer, the test item will be temporarily set as unpredictable for that wafer.

The test flow is applied to all wafers. Figure 5.6 shows the total number of measured dies for each test item across all wafers. Here, two different cases are studied: (i) without test cost reduction (Full) and (ii) with test cost reduction (IPM/DADMM). Note that IPM/DADMM achieves significant cost reduction for most test items in this example. Table 5.3 summarizes the overall test cost, the escape rate and the yield loss. In this example, since the measurement time of each test item is not disclosed by the industrial collaborator, we simply use the total number of measurements to assess the test cost. Based on table 5.3, IPM/DADMM achieves 1.875× reduction in test cost for this example.

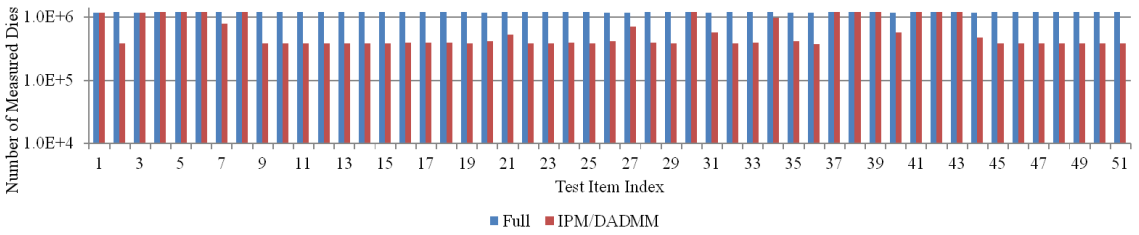


Figure 5.6: Test cost reduction summary
The total number of measured dies across all wafers is shown for each test item without test cost reduction (Full) and with test cost reduction (IPM/DADMM).

Table 5.3: Test cost reduction by IPM/DADMM

	Full	IPM/DADMM
Overall Test Cost	6.0×10^7	3.2×10^7
Escape Rate	-	1.2×10^{-3}
Yield Loss	-	2.0×10^{-3}

5.7 Conclusions

Virtual Probe is a method for wafer probe test cost reduction based on sparse reconstruction. The full wafer map is estimated from the measurement of a small subset of dies when the test item has a predictable spatial pattern. Given that the dictionary is a partial IDCT transform, DADMM presents several properties that are an advantage to VP. Namely, it solves the dual problem instead of the primal, which has a smaller number of variables; the Hessian of the Lagrange w.r.t α is a diagonal matrix, instead of a dense matrix; and there is no need to use the explicit DCT or IDCT matrices, only its fast transforms are used instead.

The results show that speed-ups up to 38× can be achieved with DADMM compared with an interior-point solver, without loss of accuracy. The test cost reduction is the same for both methods, which is a 1.875× reduction for the given example of an industrial RF transceiver.

Chapter 6

Summary Conclusions

This dissertation focused on studying ways to reduce runtime of convex solvers for the sparse coding problem. To that end, it made the following contributions:

- The first contribution is a thorough benchmark of several solvers under different conditions, for several values of the coherence of the dictionary, sparsity of the true signal and the regularization parameter. The benchmark confirms that DADMM has generally the best performance and therefore is a good base to create faster solvers for overcomplete dictionaries. A benchmark of the same algorithm is done with greedy solvers for ℓ_0-norm based problems to compare speed and recovery properties. Although this latter benchmark is not entirely fair in terms of recovery properties, it is shown that, contrary to common belief, the fastest L1 algorithm is comparable or even surpasses the greedy algorithms in terms of runtime.

- A new version of DADMM is proposed, named DADMM-eig, in which the penalty parameter can be changed at each iteration with no extra cost. A study is made to evaluate what is the best sequence of penalty parameter values and its relation to variables existent in the algorithm. The strategy is to exaustively search for the best penalty parameter in a discovery phase, in which the true so-

lution is known in advance. A relation between the optimal penalty parameter and reconstruction error is proposed, which improves performance compared to DADMM with fixed penalty parameter. Furthermore, simplifications derived from DADMM-eig reduce the complexity of its matrix-vector multiplications, since it takes advantage of the sparsity of the intermediate solutions. Several examples are presented that, for the same convergence level, DADMM-eig finishes in less than half the time compared with standard DADMM.

- We also explore unique properties of DADMM to accelerate Virtual Probe (VP), a method used to reduce wafer test cost by reducing the number of effective measurements required. The properties of DADMM become an advantage when the dictionary is a partial fast-transform, as is the case of the DCT transform used in VP. Namely, the overcompleteness of the partial DCT favors the dual formulation of the VP optimization problem, the orthogonality of its rows simplifies the closed-form expression of the dual variable, and the use of a fast DCT transform to efficiently calculate matrix-vector multiplications within the iteration loop of DADMM further reduces the computational cost. Results show that DADMM tailored to VP can reach speed-ups up to 38× compared with a standard interior-point solver. The use of adaptive penalty parameter could further improve the speed-up values.

6.1 Future Work

This work also leaves some open questions that could be addressed to further accelerate L1 solvers.

6.1.1 Universal Penalty Parameter Update

This dissertation suggests an approach to finding optimal penalty parameter sequences for DADMM during a discovery phase. From there, for the test cases used, a correlation was found between the best penalty parameter and the residual for a given iteration. The created update rule works well for the test cases presented in this dissertation, but their scope is still limited. To validate the rule further, it should be tested against other datasets and conditions. With a larger range of cases, a better and broader rule can potentially be found.

6.1.2 Highly Overcomplete Dictionaries

Very large dictionaries are particularly demanding for second-order solvers of the primal formulation, since they have superlinear complexity with n. Dual formulation solvers mitigate the increasing size of the Hessian, but still involve matrix-vector multiplications with complexity $\mathcal{O}(mn)$, for m observations and n atoms. DADMM-eig has two matrix-vector multiplications: $\mathbf{A}\mathbf{x}_{(k)}$, with complexity $O(ms_{(k)})$, where $s_{(k)}$ is the sparsity of $\mathbf{x}_{(k)}$, and $\mathbf{A}^T\alpha_{(k)}$, which still has complexity $\mathcal{O}(mn)$, since $\alpha_{(k)}$ is not sparse. However, many elements of this operation lead independently to zero elements in $\mathbf{x}_{(k+1)}$ through equation (4.6). Therefore, its computation is unnecessary, if it is known in advance that the intermediate solution will be zero. In the pursuit of fast algorithms for highly overcomplete dictionaries, it would be interesting to derive conditions to guarantee that resulting solutions for a given iteration would be zero even before computing the corresponding element of $\mathbf{A}^T\alpha$.

The acceleration of L1 solvers is extremely important to open a vast range of sparse coding applications into the practical domain. In particular, overcomplete dictionaries are a fundamental part of most applications, for which DADMM is a state-of-the-art solver. Formulating an update rule that reaches close to optimal descent in all situations makes DADMM a parameter-free algorithm. Accelerating it

even further and making it work more efficiently for a broader range of conditions can eventually set it as a model solver for L1 problems.

Bibliography

[1] M. Aharon, M. Elad, and A. Bruckstein. K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *Signal Processing, IEEE Transactions on*, 54(11):4311–4322, Nov 2006.

[2] K. Arabi. Special session 6c: New topic mixed-signal test impact to soc commercialization. In *VLSI Test Symposium (VTS), 2010 28th*, pages 212–212, April 2010.

[3] J. Barzilai and J. Borwein. Two-point step size gradient methods. *IMA Journal of Numerical Analysis*, 8(1):141–148, 1988.

[4] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

[5] S. Bengio, F. Pereira, Y. Singer, and D. Strelow. Group sparse coding. In Y. Bengio, D. Schuurmans, J.D. Lafferty, C.K.I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 82–89. Curran Associates, Inc., 2009.

[6] Dimitri P Bertsekas. *Nonlinear programming*. Athena Scientific, 1999.

[7] T.T. Cai and L. Wang. Orthogonal matching pursuit for sparse signal recovery with noise. *Information Theory, IEEE Transactions on*, 57(7):4680–4688, July 2011.

[8] T.T. Cai and A. Zhang. Sharp rip bound for sparse signal and low-rank matrix recovery. *Applied and Computational Harmonic Analysis*, 35(1):74 – 93, 2013.

[9] T.T. Cai and A. Zhang. Sparse representation of a polytope and recovery of sparse signals and low-rank matrices. *Information Theory, IEEE Transactions on*, 60(1):122–132, Jan 2014.

[10] E. J. Candès, M. B. Wakin, and S. P. Boyd. Enhancing Sparsity by Reweighted l1 Minimization. *Journal of Fourier Analysis and Applications*, 14(5-6):877–905, 2008.

[11] E.J. Candes and T. Tao. Near-optimal signal recovery from random projections: Universal encoding strategies? *Information Theory, IEEE Transactions on*, 52(12):5406–5425, Dec 2006.

[12] V. Cevher, A. Sankaranarayanan, M. F. Duarte, D. Reddy, and R. G. Baraniuk. Compressive sensing for background subtraction. In *European Conf. Comp. Vision (ECCV)*, pages 155–168, 2008.

[13] R. H. Chan, J. Yang, and X. Yuan. Alternating direction method for image inpainting in wavelet domains. *SIAM J. Img. Sci.*, 4(3):807–826, September 2011.

[14] H.-M. Chang, K.-T. Cheng, W. Zhang, Xin Li, and K.M. Butler. Test cost reduction through performance prediction using virtual probe. In *Test Conference (ITC), 2011 IEEE International*, pages 1–9, Sept 2011.

[15] K.-T. Cheng and H.-M. Chang. Recent advances in analog, mixed-signal, and rf testing. *IPSJ Transactions on System LSI Design Methodology*, 3:19–46, 2010.

[16] P. L. Combettes and V. R. Wajs. Signal recovery by proximal forward-backward splitting. *Multiscale Modeling & Simulation*, 4(4):1168–1200, 2005.

[17] W. Dai and O. Milenkovic. Subspace pursuit for compressive sensing signal reconstruction. *Information Theory, IEEE Transactions on*, 55(5):2230–2249, May 2009.

[18] W. Dai, T. Xu, and W. Wang. Simultaneous codeword optimization (simco) for dictionary update and learning. *Signal Processing, IEEE Transactions on*, 60(12):6340–6353, Dec 2012.

[19] T.T. Do, Lu Gan, N. Nguyen, and T.D. Tran. Sparsity adaptive matching pursuit algorithm for practical compressed sensing. In *Signals, Systems and Computers, 2008 42nd Asilomar Conference on*, pages 581–587, Oct 2008.

[20] D.L. Donoho. Compressed sensing. *Information Theory, IEEE Transactions on*, 52(4):1289–1306, April 2006.

[21] M.F. Duarte, M.A. Davenport, D. Takhar, J.N. Laska, Ting Sun, K.F. Kelly, and R.G. Baraniuk. Single-pixel imaging via compressive sampling. *Signal Processing Magazine, IEEE*, 25(2):83–91, March 2008.

[22] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least Angle Regression. *The Annals of Statistics*, 32(2):407–451, 2004.

[23] M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *Image Processing, IEEE Transactions on*, 15(12):3736–3745, Dec 2006.

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

[25] W. J. Fu. Penalized Regressions: The Bridge versus the Lasso. *Journal of Computational and Graphical Statistics*, 7(3):pp. 397–416, 1998.

[26] R. Garg and R. Khandekar. Gradient descent with sparsification: An iterative algorithm for sparse recovery with restricted isometry property. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 337–344, New York, NY, USA, 2009. ACM.

[27] E. Ghadimi, A. Teixeira, I. Shames, and M. Johansson. Optimal parameter selection for the alternating direction method of multipliers (admm): Quadratic problems. *Automatic Control, IEEE Transactions on*, 60(3):644–658, March 2015.

[28] I. F. Gorodnitsky and B. D. Rao. Sparse signal reconstruction from limited data using FOCUSS: a re-weighted minimum norm algorithm. *IEEE Transactions on Signal Processing*, 45(3):600–616, 1997.

[29] C.-K. Hsu, Fan Lin, K.-T. Cheng, W. Zhang, Xin Li, J. M. Carulli Jr, and K. M. Butler. Test data analytics - exploring spatial and test-item correlations in production test data. In *International Test Conference (ITC), 2013 IEEE*, 2013.

[30] Ke Huang, N. Kupp, J. M. Carulli, and Y. Makris. Handling discontinuous effects in modeling spatial correlation of wafer-level analog/rf tests. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2013*, pages 553–558, March 2013.

[31] Ke Huang, N. Kupp, J.M. Carulli, and Y. Makris. On combining alternate test with spatial correlation modeling in analog/rf ics. In *Test Symposium (ETS), 2013 18th IEEE European*, pages 1–6, May 2013.

[32] Ian Jolliffe. *Principal component analysis*. Wiley Online Library, 2002.

[33] S.-J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky. An interior-point method for large-scale l1-regularized least squares. *Selected Topics in Signal Processing, IEEE Journal of*, 1(4):606–617, Dec 2007.

[34] S. Kontogiorgis and R. Meyer. A variable-penalty alternating directions method for convex optimization. *Mathematical Programming*, 83(1-3):29–53, 1998.

[35] N. Kupp, K. Huang, J. Carulli, and Y. Makris. Spatial estimation of wafer measurement parameters using gaussian process models. In *Test Conference (ITC), 2012 IEEE International*, pages 1–8, Nov 2012.

[36] N. Kupp, K. Huang, J.M. Carulli, and Y. Makris. Spatial correlation modeling for probe test cost reduction in rf devices. In *Computer-Aided Design (ICCAD), 2012 IEEE/ACM International Conference on*, pages 23–29, Nov 2012.

[37] R. Y. Q. Lai and P. C. Yuen. Proppa: A fast algorithm for ℓ_1 minimization and low-rank matrix completion. *CoRR*, abs/1205.0088, 2012.

[38] H. Lee, A. Battle, R. Raina, and A. Ng. Efficient sparse coding algorithms. In *Advances in neural information processing systems*, pages 801–808, 2006.

[39] X. Li, R.R. Rutenbar, and R.D. Blanton. Virtual probe: A statistically optimal framework for minimum-cost silicon characterization of nanoscale integrated circuits. In *Computer-Aided Design - Digest of Technical Papers, 2009. ICCAD 2009. IEEE/ACM International Conference on*, pages 433–440, Nov 2009.

[40] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online dictionary learning for sparse coding. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 689–696, New York, NY, USA, 2009. ACM.

[41] D. Needell and J.A. Tropp. Cosamp: Iterative signal recovery from incomplete and inaccurate samples. *Applied and Computational Harmonic Analysis*, 26(3):301 – 321, 2009.

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

[43] M.R. Osborne, B. Presnell, and B.A. Turlach. A new approach to variable selection in least squares problems. *IMA journal of numerical analysis*, 20(3):389, 2000.

[44] Y.C. Pati, R. Rezaiifar, and P.S. Krishnaprasad. Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition. In *Conference Record of The Twenty-Seventh Asilomar Conference on Signals, Systems and Computers*, pages 40–44 vol.1, Nov 1993.

[45] A.U. Raghunathan and S. Di Cairano. Alternating direction method of multipliers for strictly convex quadratic programs: Optimal parameter selection. In *American Control Conference (ACC), 2014*, pages 4324–4329, June 2014.

[46] R. Tyrrell Rockafellar. *Convex analysis*. Princeton Mathematical Series. Princeton University Press, Princeton, N. J., 1970.

[47] A. C. Sankaranarayanan, C. Studer, and R. Baraniuk. CS-MUVI: Video compressive sensing for spatial-multiplexing cameras. In *IEEE Conference on Computational Photography (ICCP)*, 2012.

[48] B. Shen, W. Hu, Y. Zhang, and Y.-J. Zhang. Image inpainting via sparse representation. In *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, pages 697–700, April 2009.

[49] S. K. Shevade and S. S. Keerthi. A simple and efficient algorithm for gene selection using sparse logistic regression. *Bioinformatics*, 19(17):2246–2253, 2003.

[50] K. Skretting and K. Engan. Recursive least squares dictionary learning algorithm. *IEEE Transactions on Signal Processing*, 58(4):2121–2130, 2010.

[51] H.-G. Stratigopoulos and Y. Makris. Error moderation in low-cost machine-learning-based analog/rf testing. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 27(2):339–351, Feb 2008.

[52] H.-G.D. Stratigopoulos, P. Drineas, M. Slamani, and Y. Makris. Non-rf to rf test correlation using learning machines: A case study. In *VLSI Test Symposium, 2007. 25th IEEE*, pages 9–14, May 2007.

[53] H.G. Stratigopoulos, P. Drineas, M. Slamani, and Y. Makris. Rf specification test compaction using learning machines. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 18(6):998–1002, June 2010.

[54] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1994.

[55] R. Tibshirani. Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[56] R. Tomioka and M. Sugiyama. Dual-augmented lagrangian method for efficient sparse reconstruction. *IEEE Signal Processing Letters*, 16(12):1067–1070, 2009.

[57] R. Tomioka, T. Suzuki, and M. Sugiyama. Super-Linear Convergence of Dual Augmented Lagrangian Algorithm for Sparsity Regularized Estimation. *Journal of Machine Learning Research*, 12:1537–1586, 2011.

[58] P.N. Variyam, S. Cherubal, and A. Chatterjee. Prediction of analog performance parameters using fast transient testing. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 21(3):349–361, Mar 2002.

[59] R. Voorakaranam, S.S. Akbay, S. Bhattacharya, S. Cherubal, and A. Chatterjee. Signature testing of analog and rf circuits: Algorithms and methodology. *Circuits and Systems I: Regular Papers, IEEE Transactions on*, 54(5):1018–1031, May 2007.

[60] Q. Wang, F. Chen, W. Xu, and M.-H. Yang. Online discriminative object tracking with local sparse representation. In *Applications of Computer Vision (WACV), 2012 IEEE Workshop on*, pages 425–432, Jan 2012.

[61] Y. Wang and W. Yin. Sparse signal reconstruction via iterative support detection. *SIAM J. Img. Sci.*, 3(3):462–491, August 2010.

[62] J. Wright and Yi Ma. Dense error correction via l1-minimization. In *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, pages 3033–3036, April 2009.

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

[64] S.J. Wright, R.D. Nowak, and M.A.T. Figueiredo. Sparse reconstruction by separable approximation. *Signal Processing, IEEE Transactions on*, 57(7):2479–2493, July 2009.

[65] A.Y. Yang, Zihan Zhou, A.G. Balasubramanian, S.S. Sastry, and Yi Ma. Fast ℓ_1-minimization algorithms for robust face recognition. *Image Processing, IEEE Transactions on*, 22(8):3234–3246, Aug 2013.

[66] J. Yang, J. Wright, T.S. Huang, and Y. Ma. Image super-resolution via sparse representation. *Image Processing, IEEE Transactions on*, 19(11):2861–2873, Nov 2010.

[67] J. Yang, K. Yu, Y. Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *in IEEE Conference on Computer Vision and Pattern Recognition(CVPR*, 2009.

[68] J. Yang and Y. Zhang. Alternating direction algorithms for l1-problems in compressive sensing. *SIAM J. Sci. Comput.*, 33(1):250–278, 2011.

[69] E. Yilmaz, S. Ozev, and K.M. Butler. Per-device adaptive test for analog/rf circuits using entropy-based process monitoring. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 21(6):1116–1128, June 2013.

[70] S. Zhang, X. Li, R.D. Blanton, J. Machado da Silva, J.M. Carulli, and K.M. Butler. Bayesian model fusion: Enabling test cost reduction of analog/rf circuits via wafer-level spatial variation modeling. In *Test Conference (ITC), 2014 IEEE International*, pages 1–10, Oct 2014.

[71] W. Zhang, Xin Li, E. Acar, F. Liu, and R. Rutenbar. Multi-wafer virtual probe: Minimum-cost variation characterization by exploring wafer-to-wafer correlation. In *Computer-Aided Design (ICCAD), 2010 IEEE/ACM International Conference on*, pages 47–54, Nov 2010.

[72] W. Zhang, Xin Li, F. Liu, E. Acar, R.A. Rutenbar, and R.D. Blanton. Virtual probe: A statistical framework for low-cost silicon characterization of nanoscale integrated circuits. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 30(12):1814–1827, Dec 2011.

[73] W. Zhang, Xin Li, and R. Rutenbar. Bayesian virtual probe: Minimizing variation characterization cost for nanoscale ic technologies via bayesian inference. In

Proceedings of the 47th Design Automation Conference, DAC '10, pages 262–267, New York, NY, USA, 2010. ACM.

[74] C. Zhao, X. Wang, and W. Cham. Background subtraction via robust dictionary learning. *EURASIP Journal on Image and Video Processing*, 2011(1):961–972, 2011.

[75] X. Zhao, G. Zhou, and W. Dai. Smoothed simco for dictionary learning: Handling the singularity issue. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 3292–3296, May 2013.